

Article

A Bottom-Up Approach for Automatically Grouping Sensor Data Layers by their Observed Property

Ben Knoechel, Chih-Yuan Huang and Steve H.L. Liang *

GeoSensorweb Lab, University of Calgary, 2500 University Dr NW, Calgary, AB T2N 1N4, Canada;
E-Mails: ben.knoechel@gmail.com (B.K.); huangcy@ucalgary.ca (C.-Y.H.)

* Author to whom correspondence should be addressed; E-Mail: steve.liang@ucalgary.ca;

Received: 25 November 2012; in revised form: 5 January 2013 / Accepted: 7 January 2013 /

Published: 30 January 2013

Abstract: The Sensor Web is a growing phenomenon where an increasing number of sensors are collecting data in the physical world, to be made available over the Internet. To help realize the Sensor Web, the Open Geospatial Consortium (OGC) has developed open standards to standardize the communication protocols for sharing sensor data. Spatial Data Infrastructures (SDIs) are systems that have been developed to access, process, and visualize geospatial data from heterogeneous sources, and SDIs can be designed specifically for the Sensor Web. However, there are problems with interoperability associated with a lack of standardized naming, even with data collected using the same open standard. The objective of this research is to automatically group similar sensor data layers. We propose a methodology to automatically group similar sensor data layers based on the phenomenon they measure. Our methodology is based on a unique bottom-up approach that uses text processing, approximate string matching, and semantic string matching of data layers. We use WordNet as a lexical database to compute word pair similarities and derive a set-based dissimilarity function using those scores. Two approaches are taken to group data layers: mapping is defined between all the data layers, and clustering is performed to group similar data layers. We evaluate the results of our methodology.

Keywords: GIS; data mining; information retrieval; data interoperability; OGC; SOS

1. Introduction

The World Wide Web (WWW) has had a profound impact on almost all aspects of life. Over the past 20 years it came up from obscurity into the general public's consciences, and rightly so. The WWW has revolutionized communication. Although the Internet had been around for decades, the web was realized in the early 1990s. Tim Berners-Lee developed Hyper Text Markup Language (HTML), which allowed text documents to be shared via hyperlinks. As well, he developed protocols for sharing HTML, namely Hyper Text Transfer Protocol (HTTP). These technologies were the basis for the WWW, and with the availability of a user friendly browser, the WWW exploded in 1993. In a similar way, the realization of the Sensor Web is quickly approaching, and will be an important and defining factor in the next generation of the Internet.

The term Sensor Web was first used by NASA [1], which was described as “the Sensor Web consists of a system of wireless, intra-communicating, spatially distributed sensor pods that can be easily deployed to monitor and explore new environments.” Liang *et al.* [2] extended the definition of the Sensor Web to include a wide variety of applications and sensors. They discuss the wide variety of possible sensors, such as wireless sensor networks, flood gauges, weather towers, air pollution monitors, stress gauges on bridges, mobile bio-sensors, webcams, and satellite-borne earth imaging devices. As well, they argue that the Sensor Web can be thought of as a “global sensor” that connects to all sensors and its observations. We use this notion of the Sensor Web throughout the rest of this paper.

To allow machines to communicate over the Sensor Web, a common language is needed. Just as the WWW has been successful due to the adaptation of HTML and HTTP, the Sensor Web will have a set of commonly used standards. The Open Geospatial Consortium (OGC), a standards organization, has been involved in developing these open standards for many years. They have developed the Sensor Web Enablement (SWE) standards. These standards define information models and communication protocols to make sensors accessible and interoperable on the Internet. Botts *et al.* [3] describe the impact of the realization of the Sensor Web via the SWE, “This has extraordinary significance for science, environmental monitoring, transportation management, public safety, facility security, disaster management, utilities' Supervisory Control And Data Acquisition (SCADA) operations, industrial controls, facilities management and many other domains of activity”.

Of particular importance is the Sensor Observation Service (SOS) standard [4]. It is a service to communicate observations generated from procedures. A procedure is often a sensor, because it produces an observation based on some physical phenomenon, but a procedure is more general and could be an equation or system that generates an observation. For brevity, we will use the term sensor throughout this paper. The SOS standard is a core standard for sharing sensor data. In this paper, the discussion of sensor data layers refers to data extracted from SOSs. In this paper, we only consider the SOS standard version 1.0. From here on, any mention of SOS is an abbreviated form of SOS version 1.0. As of 2012, OGC has published SOS version 2.0. However, since few tools support the new version, we rely on the existing data and software compliant with version 1.0 to test our methodology.

The nature of the Sensor Web is highly spatial-temporal. All physical sensors have some physical location, which makes all sensor data highly dependent on the proper modelling and understanding of location. As well, an observation consists of a value, computed, generated, or collected at some point in

time, giving all sensor observations a timestamp. Observations and Measurements (O&M) is an OGC SWE standard for encoding sensor data [5], and the O&M model reinforces this spatial-temporal view of sensor data. The OM_Observation element has multiple time attributes, resultTime, validTime, and phenomenonTime, and inherits location from the GF_FeatureType.

In reaction to the huge amount of data to be generated from the Sensor Web, many research groups from all around the world are designing and researching systems to handle and process these new data. Geographic Information System (GIS) is the term commonly used to refer to software packages that are capable of integrating spatial and non-spatial data to yield the spatial information that is used for decision making. Nebert [6] explains that the term Spatial Data Infrastructure (SDI) is “often used to denote the relevant base collection of technologies, policies and institutional arrangements that facilitate the availability of and access to spatial data.” Coleman and Nebert [7] argue that the main components of a SDI include data providers, databases and metadata, data networks, technologies, institutional arrangements, policies and standards, and end-users. Nogueras-Iso *et al.* [8] explain, “It must be remarked that spatial data infrastructures are just like other forms of better known infrastructures, such as roads, power lines or railways. The whole concept of spatial data infrastructures, and other forms of infrastructure, is that they allow authorized and/or participating members of the community to use them.” We see that the concept of a SDI encompasses the technologies needed to harness the power of the Sensor Web.

A SDI that is able to interact with the Sensor Web is a complex system, with many important software components. The ultimate goal of such a system is to allow users to collect sensor data relevant to their needs. Typical abilities include being able to visualize data, connect to different data providers, search by geographic location and by time period. One important use case is to allow users to search for sensor data, based on the real world phenomenon the sensor measures. This is known as a thematic filter: instead of removing data outside the location of interest or the time frame of interest, we are removing data that do not match the user’s thematic request. The phenomenon is only one possible thematic category.

The objective of this research is to provide the functionality of filtering sensor data by the phenomenon it measures, accomplished by grouping similar sensor data. The notion of a layer is introduced to help define the objective; a sensor layer is defined as a discrete entity consisting of a collection of sensor data from one source, based on a single phenomenon. As well, we want to automate the process of grouping similar sensor data, since the high volume of data will make manual or semi-manual methods too costly to perform. Therefore, we may reword our objective as defining a methodology for automatically grouping similar sensor data layers.

1.1. Problems

There are two fundamental problems associated with grouping data layers by their observed property from different data sources: syntactic interoperability and semantic interoperability. Although the use of open standards does eliminate many of the problems associated with data transfer, these two problems must be defined and addressed.

Syntactic interoperability is the first problem of automatically grouping data layers. Bishr [9] introduced this idea while discussing the heterogeneity in a multi-GIS environment. “Syntactical

investigation deals with formalizing the grammar of schemata and semantics expressions, without any reference to what they actually mean. There could be different grammar that results in syntactic heterogeneity.” In other words, syntactic interoperability arises because observed properties are represented as text strings, and unless the two sequences of characters match exactly, the computer considers them as different.

Bishr [9] also defined the idea of semantic interoperability as the goal of interoperating GISs. Semantic interoperability is described as “...to provide seamless communication between remote GISs without having prior knowledge of the underlying semantics.” They go on to note that semantic heterogeneity is when a real world fact may have more than one underlying description. Kuhn [10] mentions the idea of a semantic reference system and describes semantic interoperability as the capacity of information systems or services to work together without the need for human intervention. The problem of semantic interoperability comes from different words or descriptions to represent the same concept. For example, consider the two strings “precipitation” and “rainfall”. Since rainfall is a type of precipitation, a user interested in precipitation data would be interested in rainfall, as well as snowfall, hail, rainfall intensity, and other related observed properties. Although these concepts are intuitively related to any human, to any computer these are simply different sequences of characters.

These problems make basic string matching ineffective for automatically grouping sensor data layers together. For example, we describe the variety of assigned names to the observed properties in sensor data layers, using the SOS standard [11]. Table 1 shows the various observed properties, which all correspond to the same concept of wind speed. However, different data providers will label their data differently. This makes it difficult to design systems to simply return to the user all sensor data layers that measure wind speed.

Table 1. Various observed properties of the concept of wind speed.

1	urn:x-ogc:def:property:OGC::WindSpeed
2	urn:ogc:def:property:universityofsaskatchewan:ip3:windspeed
3	urn:ogc:def:phenomenon:OGC:1.0.30:windspeed
4	urn:ogc:def:phenomenon:OGC:1.0.30:WindSpeeds
5	urn:ogc:def:phenomenon:OGC:windspeed
6	urn:ogc:def:property:geocens:geocensv01:windspeed
7	urn:ogc:def:property:noaa:ndbc:Wind Speed
8	urn:ogc:def:property:OGC::WindSpeed
9	urn:ogc:def:property:ucberkeley:odm:Wind Speed Avg MS
10	urn:ogc:def:property:ucberkeley:odm:Wind Speed Max MS
11	http://marinemetadata.org/cf#wind_speed
12	http://mmisw.org/ont/cf/parameter/winds

1.2. Previous Solutions

There have been many different approaches to solving interoperability issues, especially semantic interoperability. We outline notable and recent solutions to this problem in the context of the Sensor Web.

One proposed method for finding and grouping similar sensor data layers is using a Sensor Observable Registry (SOR) and semantic annotations [12]. The SOR comprises of a dictionary of URNs identifying observed properties, as well as definitions of the observed properties and references to concepts for those observed properties in an ontology. This is a practical way to manage sensor data layers, but requires a certain level of manual work. New observed properties must be manually linked to some agreed upon ontology. As well, if multiple ontologies are used, then some method of matching different ontologies must be implemented. This solution fits into the work described in [13]. They describe a Sensor Plug and Play infrastructure, including a description on semantically-enabled matchmaking. Although they discuss the use of syntactic metrics for matching, their emphasis is on a large scale Sensor Web architecture. The focus of this research is on grouping sensor data layers from existing SOS services, and with minimal help from the data provider. This is based on the assumption that many sensor data providers will not register or annotate their sensors but rather simply provide the data according to the SOS specification.

Bermudez [14] defined categories for searching sensor data in the portal. Next, an ontology was created to represent the concepts used by the service providers. Mapping between the portal categories and service provider terms was achieved using an ontological mapping tool. This allowed the user to select a portal category, and the system in turn could filter the results based on the observed property the user was interested in. This proved to be an effective solution, but requires the data providers or data integrators to create and maintain ontologies and also trust that the mapping process was effective.

A folksonomy-based recommendation system has been proposed to handle large volumes of sensor data [15]. Although these systems are very effective, these systems often suffer from cold-start problems. There is a potential benefit in building hybrid systems that utilize both external knowledge sources and user defined annotations, but that work is outside the scope of this paper.

We make a key assumption that not all data providers will provide usable ontologies, and also that not all data providers will follow recommended naming conventions. Based on much of the available data, this assumption holds true, to the best of our knowledge. Instead of relying on data providers to offer semantic cues, we propose a methodology that will consume text information directly from open standard based data services and use that data, along with some general-purpose lexical database, to infer semantics between data layers.

Another key assumption is that the “title” of the sensor data layer is a concise word or phrase to describe the data. This assumption is justified because the use of open standards makes sure that we can expect a certain consistency with the title, even if the data provider does not use or know about a naming registry.

1.3. Contributions

This paper contributes to the SDI community. Our first and foremost contribution is the evaluation of various syntactic and semantic string functions for the purpose of grouping similar sensor data layers. This provides the community with some initial work on using a solid bottom-up string matching approach for data interoperability. String processing and string matching can be more generally applied to other

aspects of data interoperability in SDIs, and our evaluation shows how these techniques performed for our data set.

Another important contribution of this work is that it highlights the current progress of OGC's SWE, namely the SOS. This paper discusses the SOS in detail, including example data from currently deployed SOS data providers. Our work focuses on the current problem in SOS regarding inconsistent naming, and serves as a record of the current progress of the OGC standard. This could possibly benefit those wishing to design other open standards for information sharing, both within and outside the GIS community.

A major contribution of this work is the unique data set we use for both clustering and classification. As far as we know, there is no other research that has attempted to cluster or classify data layers. This is similar to some of the work done in clustering tags, except that this data set is fundamentally different from tags. This provides a unique case to those interested in Information Retrieval or data mining, on how techniques may vary from data set to data set.

2. Related Work

We discuss work relevant to automatic grouping of similar sensor data layers. First, we introduce Information Retrieval as a general body of work. We then discuss how WordNet can be used as a semantic resource. Finally, we look at other methodologies for interoperability based on ontologies in SDIs.

2.1. Information Retrieval

Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers) [16]. We use an IR approach to grouping similar sensor data layers, by treating data layers like documents. Shehata *et al.* implement a concept based mining model for clustering text-based documents [17]. We follow their approach by applying text preprocessing on observed properties, using string functions to determine data layer similarities, and then clustering data layers into similar groups.

For determining the relationship between two data layers, we rely on text data for cues on how close the two data layers are. This requires more sophisticated string matching algorithms than exact string matching. Cohen *et al.* [18] compare string distance metrics for name-matching tasks. The authors evaluated three categories of distance metrics: edit-distance like functions, token-based distance functions, and hybrid functions. Edit-distance functions include Levenshtein distance, Monger–Elkan distance function, Jaro metric and Jaro–Winkler metric. For token-based functions, they consider Jaccard similarity, cosine similarity, Jensen–Shannon distance, as well as a method proposed by Fellegi and Sunter. Overall, they found the best performance from a hybrid scheme combining tf-idf weights with Jaro–Winkler string distance scheme.

Cruz *et al.* [19] propose a methodology for ontological alignment that we use as the basis for our methodology. They design and develop the AgreementMake system for ontological alignment, based on different techniques for comparing the text in the ontological class elements. Ontological alignment is based on finding matching classes between two ontologies. They propose a Parametric String-based Matcher (PSM), where they take sub-elements of the ontology class element (localname,

label, comments, *etc.*), normalize them, apply string metrics to develop similarity values, which are then weighted in a final similarity measure. The string matchers include edit-distance, Jaro–Winkler, and a substring-based measure devised by them. They also use a Vector-based Multi-word Matcher (VMM), which tokenizes ontological classes, builds tf-idf vectors and applies a cosine similarity.

2.2. WordNet as a Semantic Resource

In order to define string matching functions that take advantage of the meaning of words, we will use WordNet [20] as a semantic resource. We utilize the work presented in [21,22], where several different approaches are used to generate similarities between words using WordNet. Once similarity values for word pairs are generated, we can then define a semantic dissimilarity function for establishing how similar two data layers are based on word relationships.

WordNet is a lexical network of English words. Nouns, verbs, adjectives, and adverbs are organized into sets of synonyms, or synsets. WordNet is commonly used for its extensive structure of nouns. The backbone of the noun network is the subsumption hierarchy, consisting of parent-child relationships. Synsets are connected by various relationships, including hyponymy (is-a), its inverse hypernymy, six meronymic (part-of) relations, and antonymy (complement-of). To use WordNet, we define a global root element such that all the synsets are contained within a graph.

Many of the word similarity approaches do not have formal names, and so we adapt the naming convention from [22]. Table 2 is a summary of the abbreviations and approaches used in this paper. The authors of [22] provide a free tool to calculate word similarities, which is used by our group to compute all word pair similarities. As well, we introduce our own basic word pair similarity algorithm, named *kno*. All these approaches are discussed separately based on their general approach.

Table 2. Summary of approaches to defining word relatedness using WordNet.

Approach	Authors	Year	Acronym
Information Content Approach	Resnik [23]	1995	res
	Lin [24]	1998	lin
	Jiang and Conrath [25]	1997	jcn
Path Length Approach	Leacock and Chodorow [21]	1998	lch
	Wu and Palmer [21]	1994	wup
Word Relatedness Measures	Hirst and St-Onge [26]	1998	hso
	Banerjee and Pedersen [27]	2003	lesk
	Patwardhan [22]	2003	vector
Proposed	Knoechel	2012	kno

There are three approaches based on information content, *res* [23], *lin* [24], and *jcn* [25]. These approaches use a corpus to generate the frequency a given word will appear, and combine that information with the length between two concept's lowest common parent. The basic idea is that as one moves upwards through a taxonomy, the probability of encountering a concept increases. The information content of general high-level concepts is therefore quite low, because they are related to many other concepts.

There are two approaches based on path lengths between concepts, *lch* [21] and *wup* [21]. These are two measures based on the path length between two concepts.

There are three approaches based on word relatedness measures, *hso* [26], *lesk* [27], and *vector*. Hirst and St-Onge's approach, or *hso*, is based on path length, and also paths having direction based on the nature of the relationship between two concepts. The *lesk* is a semantic relatedness measure that is based on the number of shared words in their glosses [27]. A gloss is defined here as an alternative word for the description or definition of a word. The *vector* measure creates a co-occurrence matrix from a corpus made up of the WordNet glosses, defining each concept as a gloss vector. The relatedness between concepts is found by computing the cosine between a pair of gloss vectors.

The approach *kno* was devised with a simple strategy. It assigns a word pair similarity of one to parent-child relationships, and a zero to all other word pairs. This is useful because it helps define obvious and useful relationships, and ignores all other relationships between words. This is an overly-simplified word pair similarity score to help evaluate the effectiveness of the other strategies.

WordNet is an expansive database of English words, but in our research there are tokens that WordNet does not recognize. These are often abbreviations, acronyms, or slang (e.g., "tempc" as an abbreviation for "temperature Celsius"). If a token is not identified as a word according to WordNet, then a similarity score of zero is assigned to any word pair that contains this unknown token.

2.3. Ontology-Based SDIs

We look at other researchers who use ontologies in their SDI, as this has been the preferred methodology for achieving semantic interoperability. We select the most relevant papers, with systems closest to our work.

Henson *et al.* designed a system to add intelligence to sensor data [28]. They semantically enable SOS by adding semantic annotations to sensor data and using ontology models to reason over observations. Their system architecture has a SOS front end connected to a SPARQL Query Engine, linked to their knowledge base. To make our system as usable and practical as possible, we focus on a bottom-up approach that does not necessitate large, complex ontologies from domain experts. We only use word semantics to provide groups of related data layers.

Janowicz *et al.* [29] discuss the semantics of similarity, in a geographic information retrieval context. Their framework can be used to specify the semantics of similarity. They argue that semantic similarity can only be computed between concepts, often derived from an ontology. In this context, a bottom-up approach would be a sophisticated syntactic similarity measure. However, since ontologies are not available from a SOS service, it is impossible to use semantically rich ontologies for semantic similarity measures. The proposed methodology in this paper includes WordNet derived word pair similarity scores, which identifies relationships between concepts. Using the word pair similarity score in an algorithm solves the problem of semantic interoperability in the context of grouping observed properties.

Lutz *et al.* [30] discuss overcoming semantic heterogeneity in SDIs. Their system is based on a hybrid ontology approach, where each information system has its own application ontology, and each of these ontologies is based on a shared vocabulary, an upper level ontology. To connect the data sources to the application level ontologies, registration mappings are used. The application ontologies and the

registration mappings are not easy to generate, and certainly not provided by all, if any, data providers. Although a solid framework, until service providers integrate the use of ontologies, this system is isolated from many of the real data sources we want to connect to.

We do not use any other high level ontology in our methodology. We emphasize a strong bottom-up approach by only using word pair similarities, and as a result our approach is extremely flexible and can be reapplied in a variety of SDI contexts. By subscribing to one ontology, we would need to define mapping or translations from all external data sources to our ontology, which is a difficult problem to solve with a fully automatic solution.

3. Methodology

For this paper, we describe a bottom-up approach to automatically grouping similar sensor data layers. First, we discuss the data used to represent data layers. Next, we describe dissimilarity functions to produce numerical values of how dissimilar two data layers are. We then use the dissimilarity functions to perform both property layer mapping and clustering.

3.1. Data

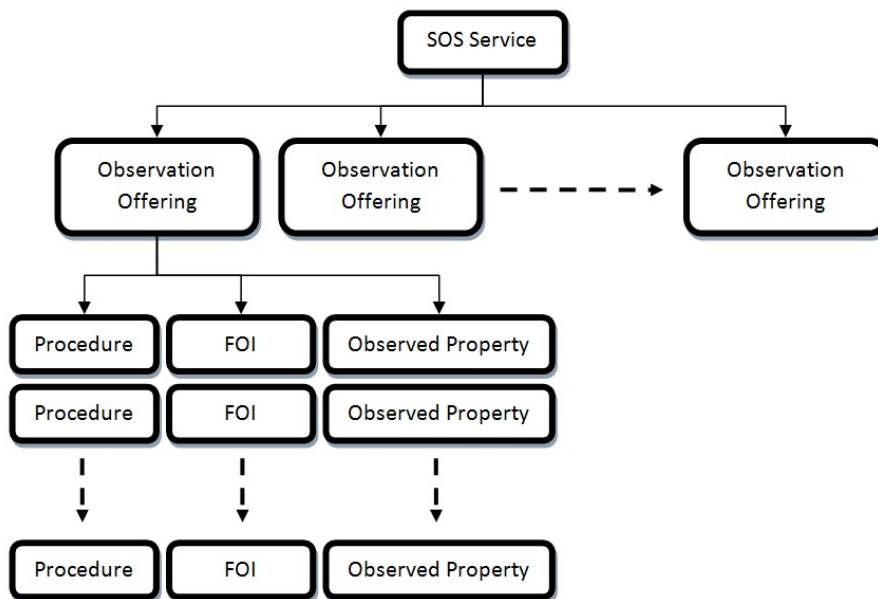
First, we must define what a data layer is in the context of our research. All of the sensor data we use in our research is downloaded via the SOS standard. The SOS standard must be explained in order to describe the definition of a sensor data layer.

SOS runs over HTTP via a client–server architecture. Content is negotiated via XML documents. Some important terminology from the SOS standard will be used throughout this paper, and is defined here.

- **Observation Offering**—A logical grouping of data sources.
- **Phenomenon**—Some naturally occurring event in the real world that can be measured, (*i.e.*, wind speed, air temperature).
- **Observed Property**—Another term for phenomenon.
- **Procedure**—A procedure is another term for sensor, except that procedure is more general to include any process that generates an observational value.
- **Feature of Interest (FOI)**—An object associated with an observation, such as a lake where a temperature measurement was taken.

The relationship of a SOS is presented in Figure 1. A SOS will have one or more observation offerings. Each observation offering will have lists of one or more observed properties, FOIs, and procedures. Our research group has defined a Property Layer (PL) as a sensor data layer extracted from a SOS. The terms data layer, sensor data layer, and PL will be used interchangeably throughout the rest of this paper. A PL is a unique data layer, defined by a SOS service URL, an observation offering, and an observed property [11]. Since a SOS, or even a single observation offering within a SOS, may offer a variety of data sources, a PL is the single most atomic data layer available from a SOS.

Figure 1. SOS structure overview.



For this paper, we will only use the observed property of a PL for the purpose of determining similarity. This is because similarity is based on how similar the two data layer’s phenomenon is, and the observed property is the most consistent, useful, and direct piece of information for determining what real world phenomenon a sensor is measuring.

For this paper, we will be extracting data from 27 different SOSs. The SOS services were discovered using a Peer-to-Peer (P2P) resource discovery system [31]. There are 212 PLs in our data set. A small example of the dataset is shown in Table 3. It must be noted that many of the existing SOS services are in the testing phase, simply due to the gradual development and deployment of the SOS standard. As well, many of the current SOS services online are run by the GeoSensorweb Lab, our own research group. This may cause a bias in the data source, which may influence the evaluation. However, the presented methodology is not influenced by this, and it remains perfectly valid for data collected by future SOS services. A full list of SOS services is presented in Table 4.

Table 3. Example property layers.

Observed Property URL
urn:ogc:def:property:geocens: rocky_view_groundwater:groundwater
urn:ogc:def:property:noaa:ndbc:Wind Direction
urn:ogc:def:property:noaa:ndbc:Wind Speed
urn:ogc:def:property:noaa:ndbc:Wind Gust
urn:ogc:def:property:ucberkeley: odm:Solar Radiation Total kW/m ²
urn:ogc:def:phenomenon:OGC:1.0.30: watertemperature
urn:ogc:def:phenomenon:OGC:waterlevel

Table 4. List of SOS services used to create property layers.

Sensor Observation Service URL
http://app.geocens.ca:8155/sos
http://app.geocens.ca:8171/sos
http://app.geocens.ca:8191/sos
http://sensorweb.vito.be:8080/52nSOSv3/sos
http://app.geocens.ca:8175/sos
http://alec192.geocens.ca:9182/GSWSOSTJSv1ABW3/sos
http://alec192.geocens.ca:9182/GSWSOSTJSv1CAWeather2d/sos
http://app.geocens.ca:8181/sos
http://app.geocens.ca:8203/sos
http://app.geocens.ca:8206/sos
http://app.geocens.ca:8208/sos
http://app.geocens.ca:8210/sos
http://app.geocens.ca:8212/sos
http://app.geocens.ca:8215/sos
http://app.geocens.ca:8217/sos
http://app.geocens.ca:8221/sos
http://app.geocens.ca:8187/sos
http://app.geocens.ca:8189/sos
http://app.geocens.ca:8195/sos
http://app.geocens.ca:8197/sos
http://www.csiro.au/sensorweb/BOM_SOS/sos
http://ccip.lat-lon.de/ccip-sos/services
http://infotrek.er.usgs.gov/ogc-ie/sosbbox
http://mmisw.org/oostethys/sos
http://security.demo.52north.org/wss/service/sos_weather/noauth
http://www.csiro.au/sensorweb/CSIRO_SOS/sos
http://www.pegelonline.wsv.de/webservices/gis/sos

3.2. Text Processing

Text processing is an essential step in our methodology. We consider two types of text processing, *normalization* and *tokenization*. Normalization is the process of canonicalizing strings such that superficial differences between strings are removed, and tokenization is the process of converting text into distinct tokens. For our normalization, we strip the OGC URI prefix, so that we are left with only the observed property text. We also convert all uppercase characters to lowercase characters, and remove whitespace and other separating characters, like underscores.

Tokenization is done as a second step after normalization. We first normalize the observed properties, as described above. Next, we use WordNet as a dictionary and split the observed property into distinct words. As a result, we are left with a list of strings, each string being a distinct word. For example, consider the observed property “urn:ogc:def:property:geocens:rocky_view_groundwater:groundwater”. After normalization, we are left with “groundwater”, and after tokenization, we are left with a list of two strings, (“ground”, “water”). An edit-based string function uses only a single string, which is generated

from normalization. A set-based string function uses the list of tokens as input, which is generated from tokenization.

3.3. Dissimilarity Functions

The similarity between two objects is a numerical measure of the degree to which the two objects are alike [32]. However, we will use the notion of dissimilarity for this paper, which is a numerical measure of the degree to which the two objects are different. We use the term dissimilarity over distance because not all of these functions satisfy the triangle inequality, which we believe needs to be satisfied to appropriately use the term distance.

We define and use three dissimilarity functions for this work, a Length Adjusted Levenshtein Dissimilarity function, a Jaccard-based Dissimilarity function, and a Semantic Dissimilarity function. The Length Adjusted Levenshtein Dissimilarity function is an edit-based function, while the latter two are set-based functions.

3.3.1. Length Adjusted Levenshtein Dissimilarity

The Length Adjusted Levenshtein Dissimilarity (LALD) is a modification of the Levenshtein Distance [33]. It is used over the basic edit distance to (1) reduce the impact of string length on the dissimilarity between strings and (2) normalize all dissimilarity values between 0 and 1.

The Levenshtein Distance counts the number of additions, subtractions, and substitutions required to traverse from one string to another. Our modification is this string length divided by the maximum string length between both words.

$$d_{LALD} = \frac{d_{LD}}{\max(|s1|, |s2|)} \quad (1)$$

3.3.2. Jaccard Dissimilarity

The Jaccard coefficient is a measure of similarity between two data objects. Given two objects, the Jaccard coefficient is the number of shared binary attributes divided by the total number of binary attributes of both data objects. Therefore, this function requires the input of an array of tokens, where each token is a string. To use this as a dissimilarity function, we simply difference one by the Jaccard coefficient.

$$d_{jaccard} = 1 - \frac{m_{11}}{m_{11} + m_{10} + m_{01}} \quad (2)$$

m_{11} is the number of words that exists in both strings, m_{10} is the number of words that exist only in string 1, and m_{01} is the number of words that exist only in string 2. We use Jaccard over Cosine as a dissimilarity because words are not repeated in observed properties. We therefore use a boolean measure of dissimilarity.

3.3.3. Semantic Dissimilarity Function

This semantic dissimilarity function will be applied between tokens, using the set-based dissimilarity approach. The word pair similarities generated from WordNet, described in the previous section, are used in this dissimilarity function. We base our dissimilarity function off of the Jaccard dissimilarity measure, presented above.

The denominator contains the total number of distinct tokens in both token lists in Table 5. For example, if we have two arrays of tokens A: [“X”, “Y”, “Z”] and B: [“X”, “W”, “V”], we would say there are 5 distinct tokens, and a single matching token, so that the dissimilarity would be $d_{jaccard} = 1 - \frac{1}{5} = 0.8$. But let us assume we have word pair similarities available to us. We use this information to combine tokens into similar token pairs. To continue with the example for the semantic dissimilarity, we see that “X” from A is the same token as “X” from B. Now we are left with four distinct tokens, “Y”, “Z”, “W”, and “V”. We create a matrix of the word pair similarities, and rank it from most similar to least similar.

Table 5. Word pair similarity scores for semantic dissimilarity function example.

Token A	Token B	Similarity
Y	W	0.8
Y	V	0.2
Z	W	0.1
Z	V	0.1

We see that Y and W have the highest similarity. We assume that these two tokens are related, and these two tokens will form a single token pair, “YW”. The other two tokens, “Z” and “V” are also assumed to be related, and form a token pair, “ZV”. The total number of distinct tokens is now 3, and they are [“X”, “YW”, “ZV”].

However, A does not contain “YW”, it only contains “Y”, which is only 0.8 of the pair “YW”. We modify our dissimilarity function to be

$$d_{semantic} = 1 - \frac{m_{11}}{|A| + |B| - m_{11}} \quad (3)$$

where m_{11} is the total sum of the token pair similarities. This works out to be $m_{11} = 1.0 + 0.8 + 0.1 = 0.9$. Therefore, the overall semantic dissimilarity is $d_{semantic} = 1 - \frac{1.9}{3+3-1.9} = 0.54$.

3.4. Example of Dissimilarity Calculation

To help explain the methodology, two PLs are introduced and the dissimilarity is calculated between them using different dissimilarity functions. Table 6 shows two different PLs from two different SOS services, including the result of normalization and tokenization. Table 7 shows the word pair similarity values for the tokens, as well as all the dissimilarity values between the two PLs.

Table 6. Example of tokenization and normalization for two different property layers.

SOS Service URL	http://app.geocens.ca:8155/sos	http://app.geocens.ca:8197/sos
Observation Offering	rocky_view_groundwater	st_denis_evaporation
Observed Property	urn:ogc:def:property: geocens:rocky_view_ground- water:groundwater	urn:ogc:def:property: GeoCENS:st_denis_evapora- tion:SoilMoisture
Normalized Observed Property	groundwater	soilmoisture
Tokenized Observed Property	[ground,water]	[soil,moisture]

Table 7. Word pair similarity scores and dissimilarity values between *ground water* and *soil moisture*.

Property Layer 1		Ground Water								
Property Layer 2		Soil Moisture								
word1	word2	res	lin	jen	lch	wup	hso	lesk	vector	kno
ground	soil	0.70	1.00	1.00	1.00	1.00	1.00	0.53	0.78	0.00
ground	moisture	0.07	0.09	0.07	0.40	0.43	0.00	0.00	0.01	0.00
water	soil	0.38	0.63	0.19	0.56	0.71	0.40	0.07	0.07	0.00
water	moisture	0.07	0.09	0.07	0.38	0.38	0.00	0.01	0.02	0.00
Semantic Dissimilarity		0.76	0.62	0.64	0.48	0.48	0.67	0.84	0.75	1.00
		LALD Dissimilarity								0.83
		Jaccard Dissimilarity								1.00

3.5. Property Layer Mapping

The first part of our methodology is to apply the string functions to define maps between PLs. A map is a symbolic link between two PLs, and the existence of a map between two PLs indicates that those PLs are similar. For our purposes, two data layers are similar if their observed properties have a direct relationship. Ultimately, we want to group PLs together if a scientist would consider those two data layers the same source of data.

A set of maps is collectively referred to as mapping. Therefore, it is important to define mapping between PLs. A PL cannot map to itself. A map is bidirectional. It has no value, it either exists or it does not exist.

The methodology to define Property Layer Mapping requires two things, a dissimilarity function and a threshold. The process is very straightforward. Every PL is compared with every other PL that is not itself. If the value of the dissimilarity function is less than the value of the threshold, then a map is defined between the two PLs. Otherwise, no map is defined between the two PLs.

3.6. Clustering

Clustering is performed to automatically group PLs into discrete non-overlapping clusters. The input for clustering is a clustering algorithm, a dissimilarity function, and a threshold. Each clustering algorithm uses the threshold and the dissimilarity function, albeit in different ways. Therefore, the actual clustering methodology depends on the clustering algorithm.

For this methodology, we implemented three different clustering algorithms, K-medoids, DBSCAN, and HAC. K-medoids is a well known variation of the K-means clustering algorithm. DBSCAN is a density based clustering algorithm, which is fundamentally a different kind of clustering algorithm. Lastly, HAC is considered a standard document clustering technique. Shehata *et al.* [17] used HAC, Single-Pass Clustering and k-Nearest Neighbor (k-NN) as clustering techniques in their research for text clustering. Our clustering algorithm selection is based on fundamentally different algorithms; since this is a new data type used in clustering, we experiment with a variety of algorithms. We evaluate these three different clustering techniques to see which algorithm works the best with our data set. Each clustering algorithm is discussed in detail in this section.

K-medoids, also known as Partitioning Around Medoids (PAM), is a similar clustering algorithm to K-means [34]. K-means is a commonly used clustering algorithm. It involves the selection of starting points as seeds, and associating every data point or object to each seed, forming clusters. The centroids of each cluster are calculated, and each data object is re-assigned to a cluster based on the new centroids. This is done recursively until the clusters no longer change, or the change is negligible, between iterations.

However, with string-based representations of data objects, it is impossible to compute centroids of clusters. This is easily possible with numeric data, but with nominal data it is impossible. It is possible to use K-means with tokenized inputs, but for the edit-based dissimilarity functions this would not work. Instead, we use the concept of a medoid. A medoid is simply a data object within a cluster that is close to the centroid, and is used in place of a centroid.

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a density based clustering algorithm [35]. DBSCAN is fundamentally different from K-medoids because it can capture irregular or unusual geometric clusters. The two input parameters are minimum number of points and a value that defines a radius, often referred to as epsilon. DBSCAN works by going through all data objects, and if a given data object has enough other data objects in its neighbourhood, defined by the input parameters, then those data objects form a cluster. Next, the cluster is expanded by joining nearby data objects. This algorithm treats all data objects that do not belong to a cluster as noise. However, since all PLs are valid data layers, they cannot be considered noise, so the input parameter for the minimum number of data objects to belong to a cluster is one.

Hierarchical Agglomerative Clustering (HAC) is a clustering algorithm [32] which works by either iteratively splitting one large cluster or combining individual clusters, starting with each data object as a cluster. The latter, a bottom-up approach, was implemented. This means a cluster is created for every data object, and clusters are merged one at a time. To determine which two clusters should be merged, an intra-cluster distance metric is needed. For this project, we used the notion of complete linkage.

The complete linkage of two clusters is defined as the maximum distance of all the possible object distances from one the objects in one cluster to all the objects in the other cluster.

$$linkage = (\max (distance(p_i, q_j)) \mid p_i \in C_1, q_j \in C_2) \quad (4)$$

The lowest complete linkage of two clusters is calculated for all cluster pairs. If the complete linkage value is less than some threshold value between two clusters, then the two clusters are merged and the process is repeated.

4. Evaluation and Results

This chapter evaluates the various methods discussed in the methodology for automatically grouping Property Layers (PLs). First, the evaluation metrics used for the evaluation are introduced. Next, we introduce the notion of testing data and how it was collected. The last three sections of this chapter are the evaluation of dissimilarity functions, clustering, and matching, respectively.

4.1. Testing Data

The goal of this methodology is to identify groups of related sensor data layers, based on their observed properties. To test the effectiveness of the methodology, the groups of PLs must be analysed to see if PLs in the same group really are similar. To carry out this evaluation, some measure of the “true” similarity between PLs is needed. To do this, four human operators were asked to rate the relationships between PLs. Each person had a list of PL pairs, for example, “precipitation” and “groundwater”. Then, they were asked to give a score for that relationship, ranking it as similar or not similar.

The human ranked relationships between PLs can be compared with the machine computed relationships between PLs. The human ranked relationships are used to look at what relationships are captured by the methodology, and what relationships are missed or incorrectly classified.

Since there are 212 PLs, that means there are 22,366 distinct PL-PL relationships. It is too time consuming to have a person rank each relationship. Therefore, we select 8 distinct PLs, and classify the relationships between each target PL and every other PL. These PLs are shown in Table 8. So for example, a PL from Table 8, such as “urn:ogc:def:property:noaa:ndbc:Dew Point” is ranked against every other PL, giving 211 PL-PL pair relationships. Each human operator was asked to perform the same task on the same data set, and the average similarity was used in the evaluation. This redundancy allows us to negate the impact of differences in the testing data.

Table 8. Ground truth property layers for testing.

Observed Property URL
urn:ogc:def:property:noaa:ndbc:Dew Point
urn:ogc:def:property:ucberkeley:odm:Rainfall mm
urn:ogc:def:property:GeoCENS:kenaston_soil_mesonet:SoilMoisture
urn:ogc:def:property:ucberkeley:odm:Solar Radiation Total kW/m ²
urn:ogc:def:property:geocens:rocky_view_groundwater:groundwater
urn:ogc:def:property:universityofsaskatchewan:ip3:airtemperature
urn:ogc:def:property:noaa:ndbc:Wind Direction

4.2. Property Layer Mapping Evaluation

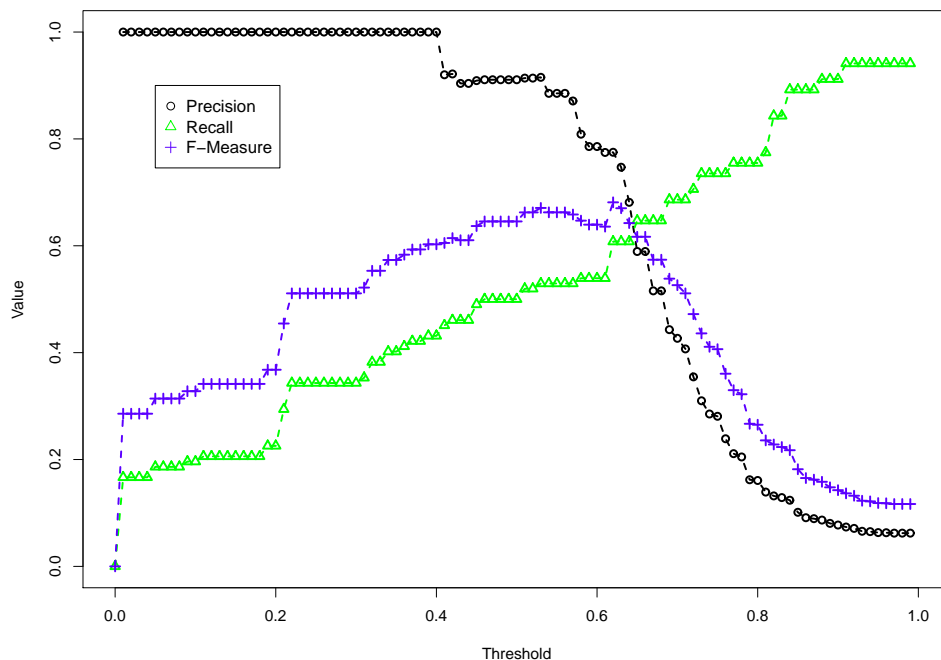
The evaluation of the PL-PL mapping is discussed. Every PL pair from the testing data section is tested against our methodology. To have the system classify a PL pair as similar or not similar, it requires two inputs, a dissimilarity function, and a threshold. The first input is one of the dissimilarity functions described in the previous section. This will produce a value from zero to one for a PL pair, by using the observed properties of the two PLs as input. Next, a threshold value is needed. This is the

cutoff value; everything below this value is classified as similar, and everything above is classified as not similar.

As a simple example, consider a PL pair with observed properties as “windspeed” and “windgust”. Note that these observed properties have been normalized, and are input for an edit-based function. We choose the LALD as the dissimilarity function, and compute the dissimilarity between them, which is 0.56. We need a threshold to decide whether 0.56 should be classified as similar or not. If we choose a threshold of 0.60, then this PL-pair would be classified as similar.

The threshold is a completely arbitrary number, and selecting the right threshold depends entirely on the dissimilarity function used. PL pairs are evaluated with a range of threshold values, to help find the ideal threshold that minimizes False Positives and False Negatives. Consider Figure 2, which shows how the LALD dissimilarity function performs at different threshold values. In this figure, every distinct threshold value is a unique PL mapping. A low threshold value will only map between PLs that have very low dissimilarity values, or for the LALD algorithm, strings that are very similar. That is why the precision is so high, only observed properties with the same name are mapped. However, the recall is very low with a low threshold, because similar observed properties with minor character differences are classified as not similar. When the threshold is high, the filter is not restrictive enough, and maps are created between PLs that have absolutely no relation. This results in a very low precision, but a higher recall.

Figure 2. Length adjusted Levenshtein dissimilarity performance.



To interpret these figures, we look at the threshold that produces the highest F-Measure. Since the F-Measure is a balance of precision and recall, we generally see a rise and fall as the threshold is increased. Stopping at the highest F-Measure, we have selected the ideal threshold for that dissimilarity function.

Figure 3 shows how the LALD and Jaccard dissimilarity functions compare with one another. It is interesting to note that the Jaccard dissimilarity function performs the best with the maximum threshold value. That is because if two observed properties do not have any matching tokens, they have a dissimilarity of 1.0. Since the threshold never exceeds 1.0, all observed properties with no matching tokens will never match. Note that the Jaccard dissimilarity function is step-wise, that is because the dissimilarity is based on the number of matching tokens and the number of non-matching tokens. Since each observed property only contains a maximum of several tokens, there are only a small number of possible values that the function will return.

The Jaccard dissimilarity performs much better than the LALD, so we will use that as a baseline for evaluating the semantic dissimilarity function. We will evaluate different word pair similarity scores generated from WordNet.

We refer to Figures 4–7. The many different WordNet measures are implemented. Figure 4 presents word similarity measures based on information content. They simply do not perform as well as the baseline Jaccard measure, because the F-Measure is consistently lower, no matter what the threshold is. Figure 5 are path lengths between WordNet words, and as well do not perform very well. This group of word pair algorithms perform the worst because they define very high similarities between words that are not related. Figure 6 uses three relatedness measures, and these perform very close to the Jaccard dissimilarity value. This is because the word pair similarity scores are generally quite low. The *lesk* measure is the only one that performs as well as the baseline. Finally, Figure 7 shows the proposed *kno* word pair similarity score, which slightly outperforms the baseline. In this situation, a very simple word pair algorithm is effective for defining similar relationships.

Figure 3. Comparison of dissimilarity functions.

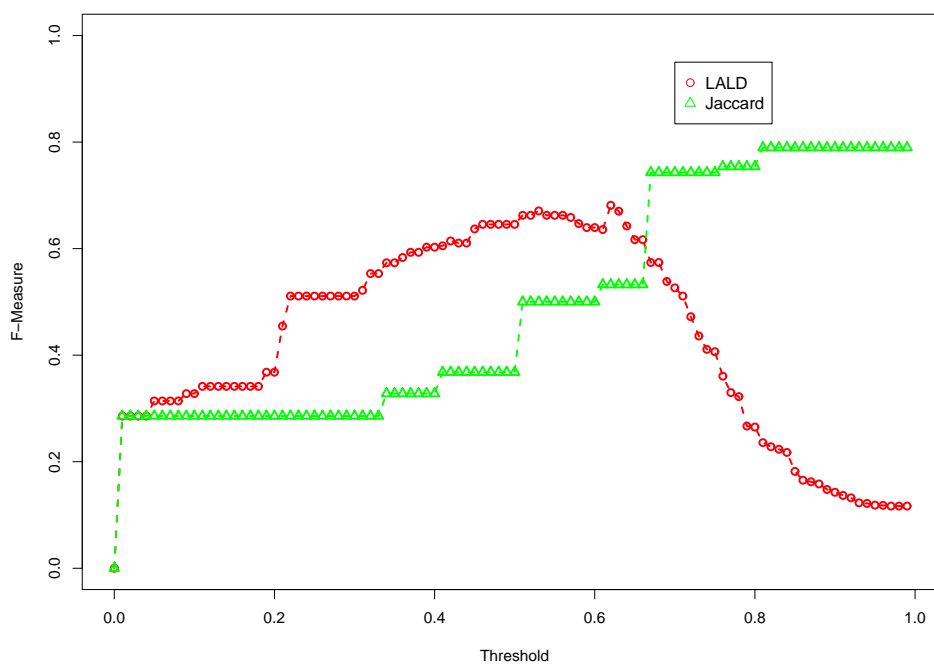


Figure 4. Semantic dissimilarity evaluation using word similarities derived from WordNet, based on an information content approach.

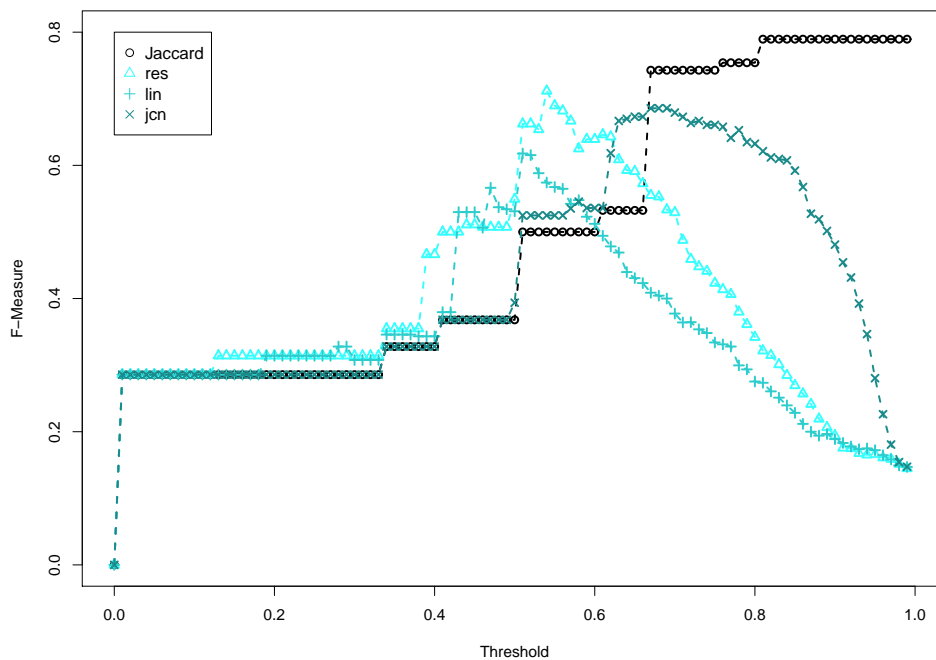
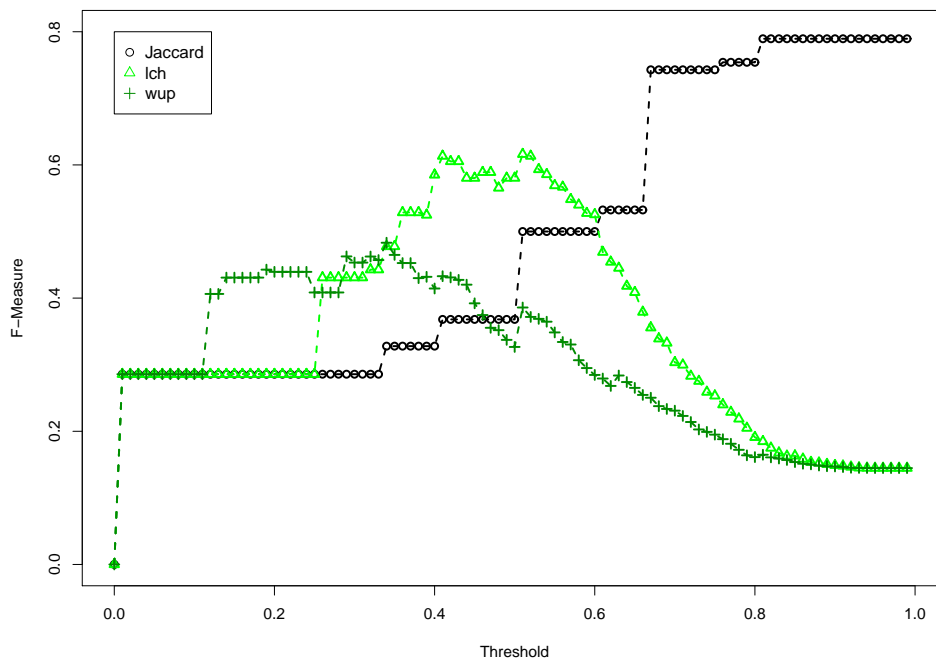


Figure 5. Semantic dissimilarity evaluation using word similarities derived from WordNet, based on a path length approach.



The evaluation of the semantic dissimilarity functions shows that the semantic dissimilarity function performs consistently behind the basic Jaccard dissimilarity function. This is because the WordNet word pair similarity scores identify relationships between words that are not intuitively related.

Figure 6. Semantic dissimilarity evaluation using word similarities derived from WordNet, based on word relatedness measures.

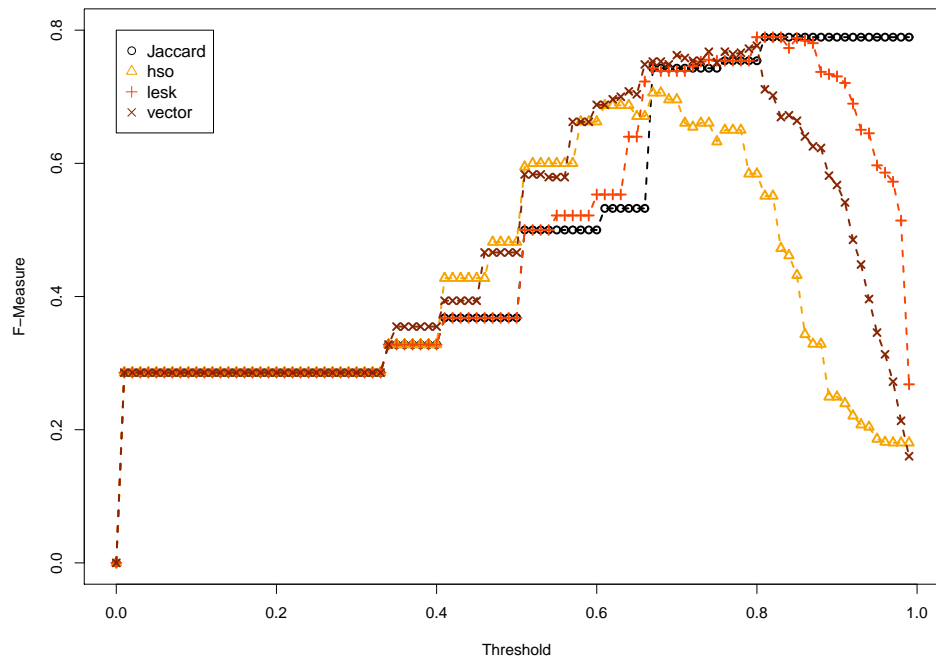
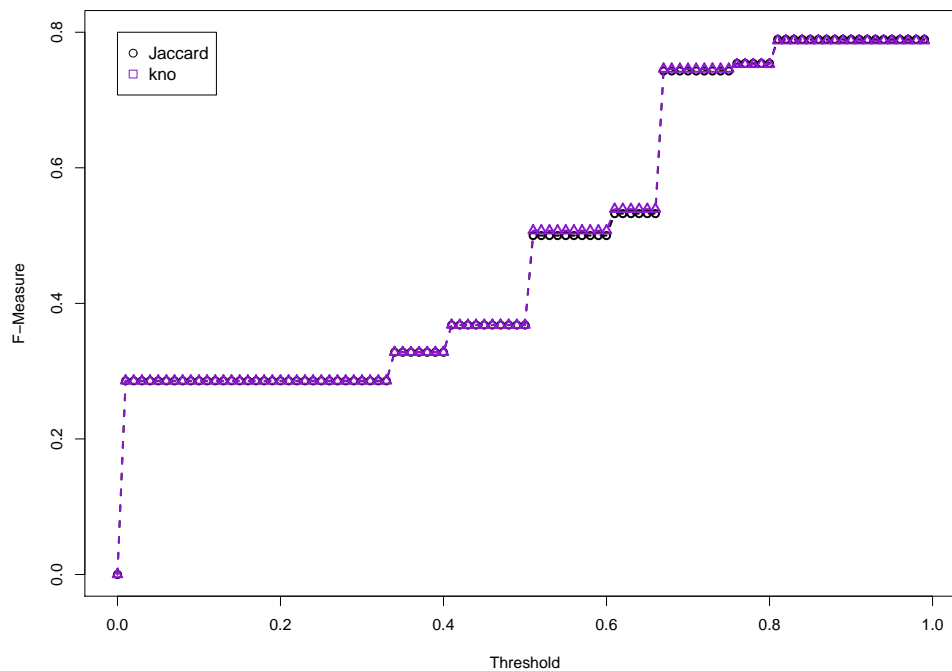


Figure 7. Semantic dissimilarity evaluation using word similarities derived from WordNet, using our proposed approach.



The results of the clustering are similar, as we see in Figures 8 and 9.

Figure 8. Clustering results using length adjusted Levenshtein dissimilarity.

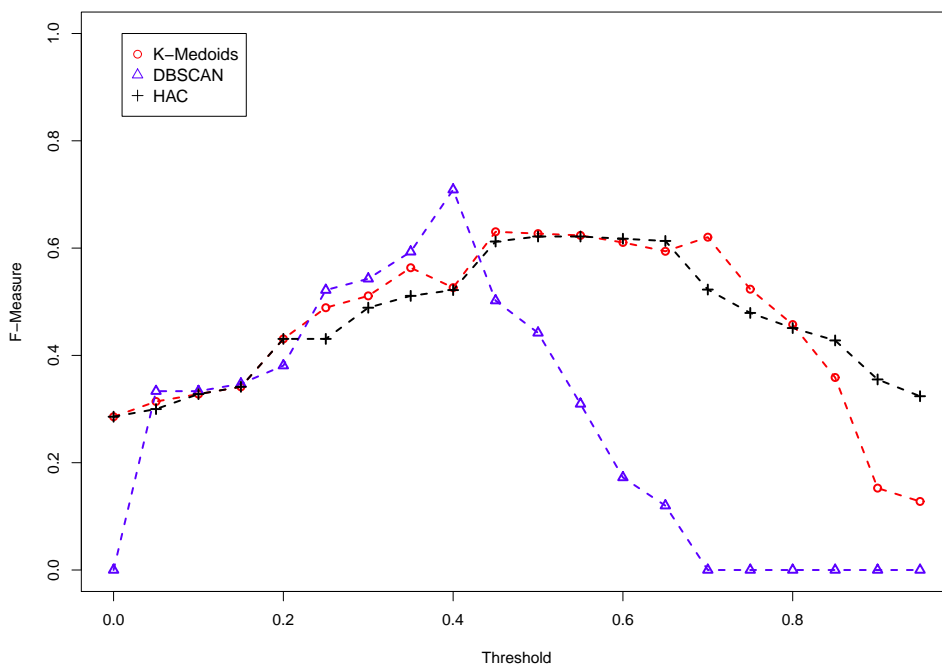
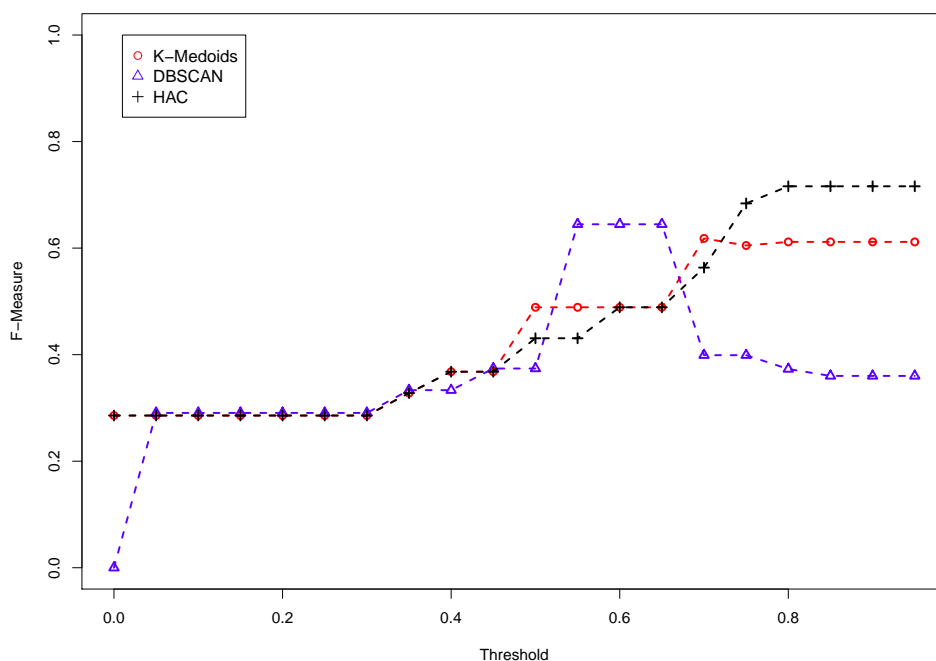


Figure 9. Clustering results using Jaccard dissimilarity.



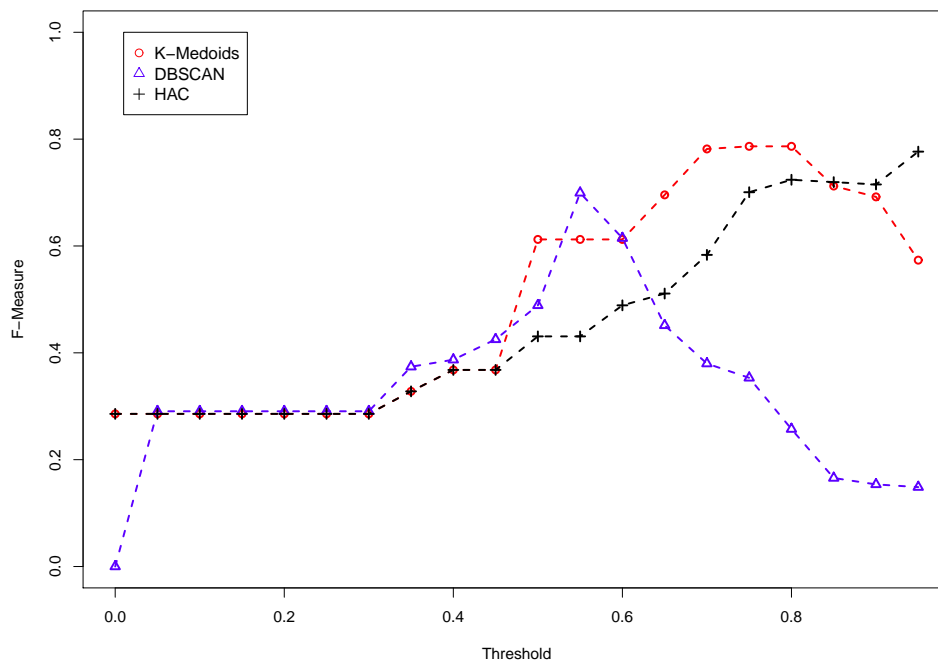
According to Table 9, the word pair algorithm with the highest F-Measure is *vector*, which we show in Figure 10. For these figures, the threshold depends on the clustering algorithm. For K-Medoids, it is used to identify an outlying medoid in a cluster, which creates a new cluster. For DBSCAN, the threshold is used as the epsilon input. For HAC, the threshold defines the maximum linkage, which will combine two clusters if the maximum dissimilarity between all objects is greater than the threshold. We vary the threshold, as in the PL-PL mapping, to see which set of clusters produces the highest F-Measure. Therefore, the *vector* word pair similarity score with K-Medoids or HAC clustering method performed

the best. This is because *vector* word pair similarity score is a conservative word pair similarity score, so it relies strongly on the syntax of words, but still identifies the necessary relationships between concepts.

Table 9. Highest F-measures for clustering.

Dissimilarity Function	Best F-Measure
LALD	0.71
Jaccard	0.72
<i>res</i>	0.66
<i>lin</i>	0.56
<i>jcn</i>	0.64
<i>lch</i>	0.64
<i>wup</i>	0.52
<i>hso</i>	0.69
<i>lesk</i>	0.66
<i>vector</i>	0.79
<i>kno</i>	0.72

Figure 10. Clustering results using semantic dissimilarity, using *vector* word similarity.



5. Conclusions

We start this paper by comparing the rise of the World Wide Web (WWW) in the early 1990s to today’s rising Sensor Web. The Sensor Web runs on top of the Internet through the use of open standards, but even with the use of open standards for sharing information, there are still interoperability issues that arise. Many SDIs rely on accessing data from multiple sources and integrating them seamlessly into a

single, logical presentation for the user. Our focus is on the task of grouping semantically similar sensor data layers. This will increase the usability of SDIs by saving users the time of manually sorting through data layers.

However, there are many problems associated with this task. The sheer number of unique sensors will necessitate an automatic approach, as manual categorization will not be feasible with the rising number of sensors. As well, the heterogeneous naming of sensor data layers makes it difficult to perform exact string matching. One well researched solution to this problem has been the semantic catalogue. These catalogues automatically groups together heterogeneous data sources. However, this approach requires the creation and maintenance of ontologies, which is a very time intensive process. As well, real world data providers often do not provide ontologies with their data, which makes it extremely difficult to use that data in a semantic catalogue.

The automatic grouping of sensor data layers presents two primary challenges in the form of differences between names. These are syntactic and semantic differences. Syntactic differences are resolved to some extent by open standards, but the same name can be represented using different characters. The best example of syntactic differences is the use of uppercase and lowercase to represent the same name. Semantic differences are more difficult to resolve, and refer to two different names to represent the same real world concept. Our paper contributes a new and useful methodological framework to the GIS community. We present here an evaluation of syntactic and semantic string matching algorithms for the purposes of automatically grouping similar sensor data layers. We investigate the SOS standard in detail, and apply clustering and classification to a new data set, sensor data layers.

Our methodology is a solid bottom-up approach. We first collect data from different OGC SOS services. We then divide it into atomic data layers known as Property Layers (PLs). The text from PLs that convey information about the phenomenon they measure is processed via normalization and tokenization. Next, we introduce WordNet as a lexical database to create word pair similarity scores. Many dissimilarity functions are introduced, based on approximate string matching. Using these dissimilarity functions, we perform PL-PL mapping and PL clustering.

We present an evaluation of how these dissimilarity functions performed in grouping similar sensor data layers. Overall, we see comparable results using edit-based and set-based dissimilarity functions. The semantic dissimilarity function did not perform as expected, and often did not perform very well. The best semantic dissimilarity function was one that only considered very direct and simple relationships between tokens.

6. Future Work

This research shows how sensor data layers may be grouped or related by their observed properties. These groups of PLs generated from the methodology could be incorporated in a Sensor Web SDI. This has been done previously by our research group through the VirtualSOS prototype [11]. PL to PL mapping could be used for a recommendation system, for example, as a user downloads a single observed property, we could retrieve all related PLs that the user might be interested in. In VirtualSOS,

classes are treated like virtual sensor data layers, and when the user selects a class, all related PLs are retrieved.

We could integrate this approach with a top-down approach, and use a bottom-up approach when there is missing or incomplete ontological information.

As well, one very important future work would be to continue investigating algorithms or techniques for generating a semantic similarity. Word pair similarity scores are limited in utility and assume all semantic information is self-contained in each word. Besides, there may be other semantic information in the SOS standard that could be used to infer the sensor data's observed property, such as the Unit of Measure of the observations or the sensor description.

Acknowledgments

The authors would like to thank CANARIE, Cybera, Alberta Innovates Technology Futures, and Microsoft Research for their supports on this project.

References

1. Delin, K.A.; Jackson, S.P. Sensor web: A new instrument concept. *Proc. SPIE* **2001**, *4282*, 1–9.
2. Liang, S.H.; Croitoru, A.; Tao, C.V. A distributed geospatial infrastructure for Sensor Web. *Comput. Geosci.* **2005**, *31*, 221–231.
3. Botts, M.; Percivall, G.; Reed, C.; Davidson, J. OGC[®] sensor web enablement: Overview and high level architecture. *GeoSensor Networks* **2008**, *4540*, 175–190.
4. Na, A.; Priest, M. *Sensor Observation Service Version 1.0.0.*; DocNr. OGC 06-009r6; Open Geospatial Consortium Inc.: Wayland, MA, USA, 2007.
5. Cox, S. *Geographic Information: Observations and Measurements*; Document OGC 10-004r3/ISO 19156; Open Geospatial Consortium Inc.: Wayland, MA, USA, 2010.
6. Nebert, D.D. *Developing Spatial Data Infrastructures: The SDI Cookbook, 2004*. Available online: <http://www.gsdi.org/gsdicookbookindex> (accessed on 25 October 2012).
7. Coleman, D.J.; Nebert, D.D. Building a North American spatial data infrastructure. *Cartogr. Geogr. Inf. Sci.* **1998**, *25*, 151–160.
8. Nogueras-Iso, J.; Zarazaga-Soria, F.J.; Muro-Medrano, P.R. *Geographic Information Metadata for Spatial Data Infrastructures*; Springer: Berlin, Germany, 2005.
9. Bishr, Y. Overcoming the semantic and other barriers to GIS interoperability. *Int. J. Geogr. Infor. Sci.* **1998**, *12*, 299–314.
10. Kuhn, W. Semantic reference systems. *Int. J. Geogr. Infor. Sci.* **2003**, *17*, 405–409.
11. Knoechel, B.; Huang, C.Y.; Liang, S. Design and Implementation of a System for the Improved Searching and Accessing of Real-World SOS Services. In *Proceedings of International Workshop on Sensor Web Enablement 2011*, Banff, AB, Canada, 6–7 October 2011.
12. Jirka, S.; Broering, A.; Foerster, T. Handling the Semantics of Sensor Observables within SWE Discovery Solutions. In *Proceedings of 2010 International Symposium on Collaborative Technologies and Systems (CTS)*, Chicago, IL, USA, 17–21 May 2010; pp. 322–329.

13. Broering, A.; Maue, P.; Janowicz, K.; Nuest, D.; Malewski, C. Semantically-enabled sensor plug & play for the sensor web. *Sensors* **2011**, *11*, 7568–7605.
14. Bermudez, L. *OGC Ocean Science Interoperability Experiment Phase 1 Report (08-124r1)*; Open Geospatial Consortium, 2011.
15. Rezel, R.; Liang, S. A Folksonomy-Based Recommendation System for the Sensor Web. In *Proceedings of 10th International Symposium on Web and Wireless Geographical Information Systems (W2GIS 2011)*, Kyoto, Japan, 3–4 March 2011; pp. 64–67.
16. Manning, C.D.; Raghavan, P.; Schtze, H. *Introduction to Information Retrieval*; Cambridge University Press: New York, NY, USA, 2008.
17. Shehata, S.; Karray, F.; Kamel, M. An efficient concept-based mining model for enhancing text clustering. *IEEE Trans. Knowl. Data Eng.* **2010**, *22*, 1360–1371.
18. Cohen, W.W.; Ravikumar, P.; Fienberg, S.E. A Comparison of String Distance Metrics for Name-Matching Tasks. In *Proceedings of the IJCAI-2003 Workshop on Information Integration on the Web (IIWeb-03)*, Acapulco, Mexico, 9–10 August 2003; pp. 73–78.
19. Cruz, I.; Pal, F.; Antonelli, R.; Stroe, C. Efficient Selection of Mappings and Automatic Quality-Driven Combination of Matching Methods. In *Proceedings of ISWC International Workshop on Ontology Matching (OM 2009)*, Mexico City, Mexico, 3–4 December 2009.
20. Fellbaum, C. *WordNet: An Electronic Lexical Database*; Bradford Books: Bradford, MA, USA, 1998.
21. Budanitsky, A.; Hirst, G. Evaluating wordnet-based measures of lexical semantic relatedness. *J. Comput. Ling.* **2006**, *32*, 13–47.
22. Pedersen, T.; Patwardhan, S.; Michelizzi, J. WordNet::Similarity: Measuring the Relatedness of Concepts. In *HLT-NAACL—Demonstrations '04 Demonstration Papers at HLT-NAACL 2004*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2004; pp. 38–41.
23. Resnik, P. Using Information Content to Evaluate Semantic Similarity in a Taxonomy. In *Proceeding IJCAI'95 Proceedings of the 14th International Joint Conference on Artificial Intelligence*; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 1995; Volume 1, 448–453.
24. Lin, D. An Information-Theoretic Definition of Similarity. In *Proceedings of the 15th International Conference on Machine Learning*, Madison, WI, USA, 24–27 July 1998; Volume 1, pp. 296–304.
25. Jiang, J.J.; Conrath, D.W. *Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy*; The Computing Research Repository, cmp-lg/9709008, 1997.
26. Hirst, G.; St-Onge, E. Lexical Chains as Representations of Context For the Detection and Correction of Malapropisms. In *WordNet: An Electronic Lexical Database*; Fellbaum, C., Ed.; The MIT Press: Cambridge, MA, USA, 1995; pp. 305–332.
27. Banerjee, S.; Pedersen, T. Extended Gloss Overlaps as a Measure of Semantic Relatedness. In *Proceeding IJCAI'03 Proceedings of the 18th International Joint Conference on Artificial Intelligence*; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 2003; Volume 18, pp. 805–810.

28. Henson, C.; Pschorr, J.; Sheth, A.; Thirunarayan, K. SemSOS: Semantic Sensor Observation Service. In *Proceedings of International Symposium on Collaborative Technologies and Systems (CTS'09)*, Baltimore, MD, USA, 18–22 May 2009; pp. 44–53.
29. Janowicz, K.; Raubal, M.; Kuhn, W. The semantics of similarity in geographic information retrieval. *J. Spat. Inf. Sci.* **2011**, *2*, 29–57.
30. Lutz, M.; Sprado, J.; Klien, E.; Schubert, C.; Christ, I. Overcoming semantic heterogeneity in spatial data infrastructures. *Comput. Geosci.* **2009**, *35*, 739–752.
31. Chen, S.; Liang, S. A Hybrid Peer-to-Peer Architecture for Global Geospatial Web Service Discovery. In *Proceedings of Spatial Knowledge and Information-Canada*, Fernie, BC, Canada, 20–22 February 2011.
32. Tan, P.N.; Michael, S.; Vipin, K. *Introduction to Data Mining*; Pearson Education, Inc.: Boston, MA, USA, 2006.
33. Levenshtein, V. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics-Doklady* **1966**, *10*, 707–710.
34. Kaufman, L.; Rousseeuw, P. Clustering by Means of medoids. In *Statistical Data Analysis Based on the L1 Norm*; Dodge, Y., Ed.; North Holland: Amsterdam, The Netherlands, 1987; pp. 405–416.
35. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining KDD-96*, Portland, OR, USA, 2–4 August 1996; pp. 226–231.

© 2013 by the authors; licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution license (<http://creativecommons.org/licenses/by/3.0/>).