Collaborative Project

GeoKnow - Making the Web an Exploratory place for Geospatial Knowledge

Project Number: 318159 **Start Date of Project:** 2012/12/01 **Duration:** 36 months

Deliverable 3.1.1

Development of First Prototype for Spatially Interlinking Data Sets

Dissemination Level	Public
Due Date of Deliverable	Month 9, 31/08/2013
Actual Submission Date	30/09/2013
Work Package	WP 3, Spatial Knowledge Aggregation, Fusing & Quality Assessment
Task	T 3.1
Type	Prototype
Approval Status	Approved
Version	1.00
Number of Pages	19
Filename	

Abstract: This deliverable presents our aproach to the spatial interlinking of datasets. We focus especially on presenting two key developments in this regard. First, we present GeoLift, a framework for the enrichment of RDF datasets with geo-spatial information. Thereafter, we present the geo-spatial extension of the LIMES framework.

Project funded by the European Commission within the Seventh Framework Programme (2007 - 2013)

History

Version	Date	Reason	Revised by
0.1	01/05/2013	Initial version of this deliverable	Axel Ngonga
0.1	10/05/2013	Extended version of this deliverable	Axel Ngonga
0.2	19/08/2013	Extension of the functionality of GeoLift	Mohammed Sherif, Mofeed Hassan
0.3	01/09/2013	GeoLift Manual, Extension of LIMES Manual	Axel Ngonga, Mohamed Sherig, Mofeed Hassan
0.4	19/09/2013	Review of the deliverable	Giorgos GIannopoulos
0.5	02/10/2013	Revised version of this deliverable	Axel Ngonga
0.6	03/10/2013	Second review this deliverable	Giorgos GIannopoulos
1.0	07/10/2013	Final version of this deliverable	Axel Ngonga

Author List

Organization	Name	Contact Information
INFAI	Axel Ngonga	ngonga@informatik.uni-leipzig.de
INFAI	Mohamed Sherif	sherif@informatik.uni-leipzig.de
INFAI	Mofeed Hassan	mounir@informatik.uni-leipzig.de

Time Schedule before Delivery

Next Action	Deadline	Care of
First version	01/08/2013	Axel Ngonga
First Consolidated version	15/08/2013	Axel Ngonga, Mofeed Hassan, Mohamed Sherif
Peer review	22/08/2013	Giorgos Giannopoulos
Submit final restructured version	31/08/2013	Giorgos Giannopoulos

Executive Summary

Manifold RDF datasets contain geo-spatial information. Linking and fusing these datasets promises to further the development of complex semantic applications which require geo-spatial information. Yet, two problems occur when trying to achieve this goal. First, the geo-spatial information contained in RDF datasets is not always explicit. For example, biographies in music datasets contain references to places in their literals. Moreover, current Link Discovery frameworks offer minimal support for geo-spatial link discovery.

In this deliverable, we present two frameworks that allow solving the problems described above. First, we present GeoLift, a framework that allows making geospatial information avalaible in RDF datasets explicit. Thereafter, we present the novel geo-spatial extension of the LIMES framework[1], which allows linking complex resources in geo-spatial data sets (e.g., polygons, linestrings, etc.). We evaluate LIMES on several geo-spatial datasets and show that the approach that we developed outperforms the state of the art by orders of magnitude.

[1]Parts of this section of the deliverable was published in [6].

Table of Contents

1 Introduction

Manifold RDF data contain implicit references to geographic data. For example, music datasets such as Jamendo include references to locations of record labels, places where artists were born or have been, etc. Our approach to addressing the problem of retrieving geo-spatial information in RDF datasets and making it explicit led to two main results: GeoLift and the geo-spatial extension of the Limes framework.

The aim of the spatial mapping component, dubbed GeoLift, is to retrieve this information and make it explicit. To achieve this goal, GeoLift accepts a configuration file as input, which describes the location of the input file as well as the sequence in which its modules are to be used and where the output of the tool is to be written. The framework can deal with an on the RDF serializations supported by the Jena reader (i.e., NT, N3, RDF/XML and Turtle) and give out data in any of these formats. In the following, we begin by presenting the basic assumptions that underlied the development of the first component of GeoLift. Then, we present the technical approach behind GeoLift. A user manual which described how to use the tool can be found at http://github.com/GeoKnow/GeoLift.

The geo-spatial extension of Limes enables users to link geo-spatial entities contained in RDF datasets or SPARQL endpoints by combining the Hausdorff and the symmetric Hausdorff distances (which allow comparing the polygons that describe the said entities) with other metrics such as the string metrics for comparing labels and Minkowski distances for comparing numeric values. It was designed with scalabilty in mind and implements sophisticated algorithms that allow reducing the overall runtime of the approach by up to two orders of magnitude. Limes also consumes an XML configuration which describes the in- and output of the tool as well as the metrics to use to compare resources. A manual encompassing a description of the tool and how to configure it is available within the distribution, which can be found at http://titan.scms.eu/LIMES.0.6.RC3.zip.

2 Assumptions

Mentions of geo-spatial entities can be retrieved in three different ways within the Linked Data paradigm:

1. *Through links*: Several datasets contain links to datasets with explicit geographical information such as DBpedia or LinkedGeoData. For example, in a music dataset, one might find information such as `http://example.org/Leipzig owl:sameAs http://dbpedia.org/resource/Leipzig`. We call this type of reference *explicit*. We can now use the semantics of RDF to fetch geographical information from DBpedia and attach it to the resource in the other ontology as `http://example.org/Leipzig` and `http://dbpedia.org/resource/Leipzig` refer to the same real-world object, i.e., the city of Leipzig in Germany.

2. *Through linking*: It is known that the Web of Data contains an insufficient number of links. The latest approximations suggest that the Linked Open Data Cloud alone consists of 31+ billion triples but only contains approximately 0.5 billion links (i.e., less than 2% of the triples are links between knowledge bases). The second intuition behind our approach is thus to use link discovery to map resources in an input knowledge base to resources in a knowledge that contains explicit geographical information. For example, given a resource `http://example.org/Athen`, GeoLift should aim to find a resource such as `http://dbpedia.org/resource/Athen` to map it with. Once having established the link between the two resources, GeoLift can then resolve to the approach defined above.

3. *Through Natural Language Processing*: In some cases, the geographic information is hidden in the objects of data type properties. For example, some datasets contain biographies, textual abstracts describing resources, comments from users, etc. One could for example imagine a dataset containing the description "This building is located in Leipzig" for a resource `http://example.org/ThomasChurch` which stands for the Thomas Church in Leipzig. The idea here is to use this information by extracting Named Entities and keywords using automated Information Extraction techniques. In our example, this would for example mean extracting that the location "Leipzig" is included in the description of the resource. Semantic Web Frameworks such as FOX[2] have the main advantage of providing URIs (in our example `http://dbpedia.org/resource/Leipzig`) for the keywords and entities that they detect. These URIs can finally be linked with the resources to which the datatype properties were attached, e.g., by adding the triple `http://example.org/ThomasChurch http://example.org/relatedTo http://dbpedia.org/resource/Leipzig` to the input knowledge base. Finally, the geographical information can be dereferenced and attached to the resources whose datatype properties were analyzed.

The idea behind GeoLift is to provide a generic architecture that contains means to exploit these three characteristics of Linked Data. In the following, we present the technical approach underlying GeoLift.

[2] http://fox.aksw.org

3 Technical Approach

3.1 Architecture

GeoLift was designed to be a modular tool which can be easily extended and re-purposed. In its first version, it provides two main types of artifacts:

1. *Modules*: These artifacts are in charge of generating geographical data based on RDF data. To this aim, they implement the three intuitions presented above. The input for such a module is an RDF dataset (in Java, a *Jena Model*). The output is also an RDF dataset enriched with geographical information (in Java, an enriched *Jena Model*). Formally, a module can thus be regarded as a function $\mu : \mathcal{R} \rightarrow \mathcal{R}$, where $\mathcal{R}$ is the set of all RDF datasets.
2. *Operators*: The idea behind operators is to enable users to define a workflow for processing their input dataset. Thus, in case a user knows the type of enrichment that is to be carried out (using linking and then links for example), he can define the sequence of modules that must be used to process his dataset. Note that the format of the input and output of modules is identical. Thus, the user is empowered to create workflows of arbitrary complexity by simply connecting modules. Formally, an operator can be regarded as a function $\varphi : \mathcal{R} \cup \mathcal{R}^2 \rightarrow \mathcal{R} \cup \mathcal{R}^2$.

The corresponding architecture is shown in Figure 1. The input layer allows reading RDF in different serializations. The enrichment modules are in the second layer and allow adding geographical information to RDF datasets by different means. The operators (which will be implemented in the future version of GeoLift) will combine the enrichment modules and allow defining a workflow for processing information. The output layer serializes the results in different format. The enrichment procedure will be monitored by implementing a controller, which will be added in the future version of GeoLift.

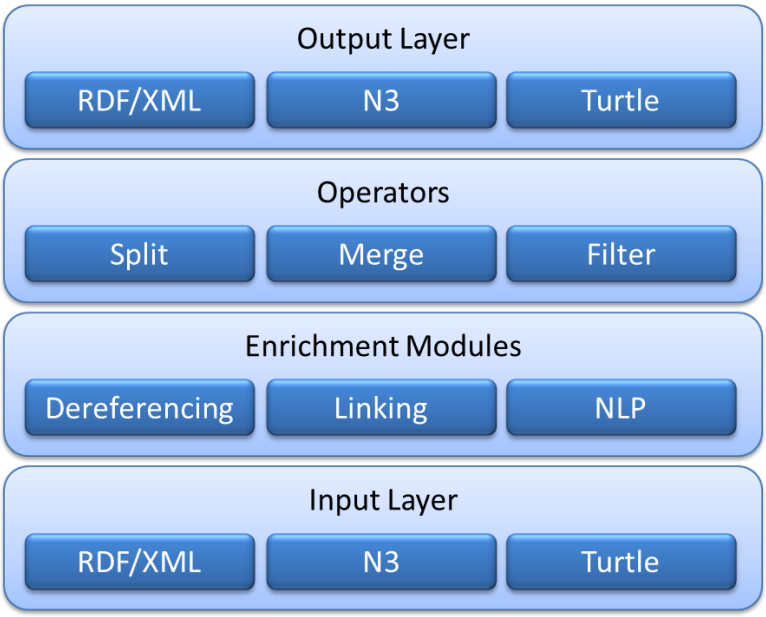


Figure 1: Architecture of GeoLift.

In the following, we present the implementation of the three intuitions presented above in GeoLift.

3.2 Using Links

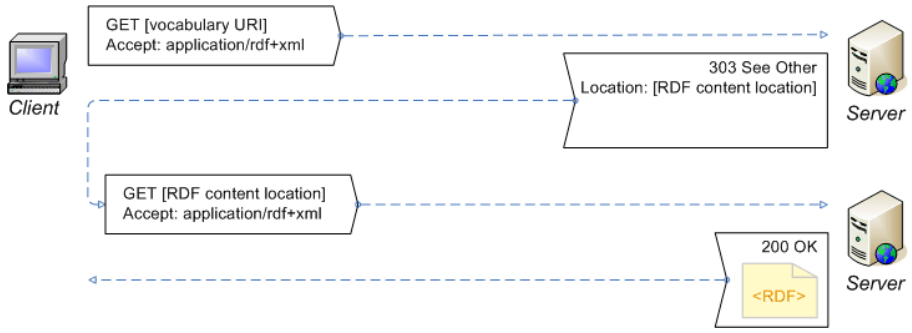


Figure 2: Content Negotiation as used by GeoLift (courtesy of W3C)

For datasets which contain `owl:sameAs` links, we deference all links from the dataset to other datasets by using a content negotiation on HTTP as shown in Figure 2. This returns a set of triples that needs to be filtered for relevant geographical information. Here, we use a predefined list of attributes that links to geographical information. Amongst others, we look for `geo:lat`, `geo:long`, `geo:lat_long`, `geo:line` and `geo:polygon`. The list of retrieved property values can be configured.

3.3 Using Linking

As pointed out before, links to geographical resources are missing in several knowledge bases. Here, we rely on the metrics implemented in the Limes framework[3] [7, 5, 9] to link the resources in the input dataset with geographical datasets. Limes, the **Li**nk Discovery Framework for **Me**tric **S**paces, is a framework for discovering links between entities contained in Linked Data sources. Limes is a hybrid framework [5] that combines the mathematical characteristics of metric spaces as well prefix-, suffix- and position filtering to compute pessimistic approximations of the similarity of instances. These approximations are then used to filter out a large amount of those instance pairs that do not suffice the mapping conditions. By these means, Limes can reduce the number of comparisons needed during the mapping process by several orders of magnitude and complexity without loosing a single link. The architecture of Limes is shown in Figure 3

Linking using Limes [5, 4] can be achieved in three ways:

1. *Manually*, by the means of a link specification [5], which is an XML-description of (1) the resources in the input and target datasets that are to be linked and (2) of the similarity measure that is to employed to link these datasets.
2. *Semi-automatically* based on active learning [10, 11, 12]. Here, the idea is that if the user is not an expert and thus unable to create a link specification, he can simply provide the framework with positive and negative examples iteratively. Based on these examples, Limes can compute links for mapping resources with high accuracy.
3. *Automatically* based on unsupervised machine learning. Here, the user can simply specify the sets of resources that are to be linked with each other. Limes implements both a deterministic and non-deterministic machine-learning approaches that optimize a pseudo-F-measure to create a one-to-one mapping.

The techniques implemented by Limes can be accessed via the SAIM user interface[4], of which a screenshot

[3] http://limes.sf.net
[4] http://saim.aksw.org

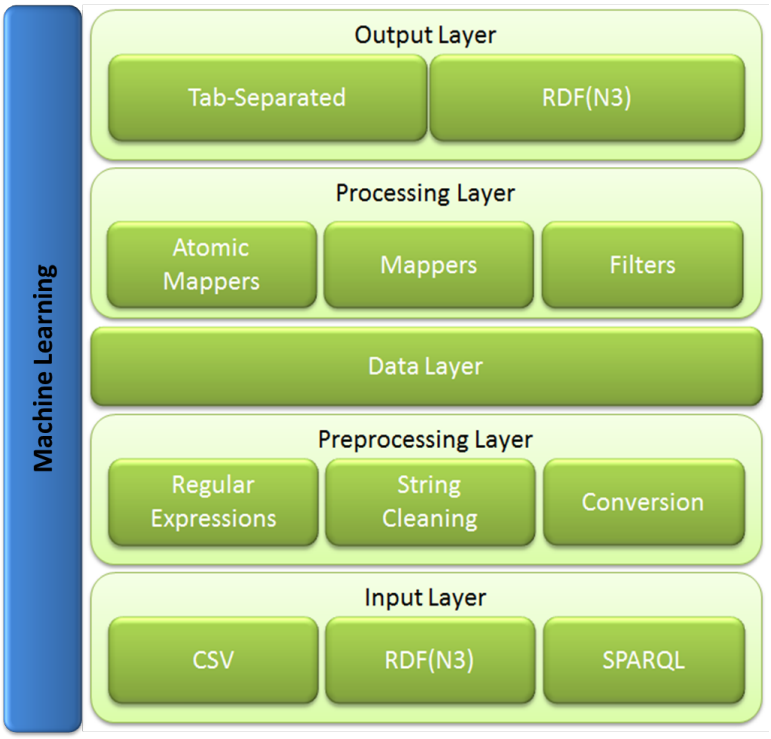


Figure 3: Architecture of LIMES

is shown in Figure 4.

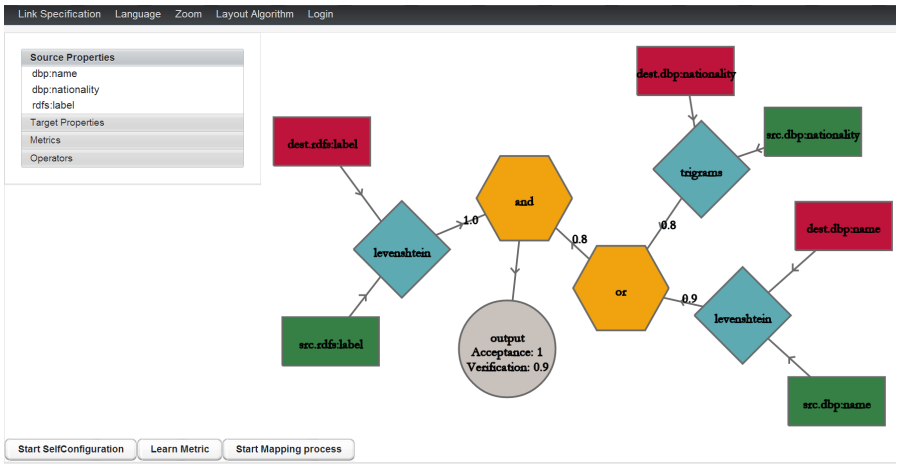


Figure 4: Screenshot of SAIM

3.4 Using Named Entity Recognition

The geographical information hidden in datatype properties is retrieved by using Named Entity Recognition. In the first version of GEOLIFT, we rely on the FOX framework. The FOX framework is a stateless and extensible framework that encompasses keyword extraction and named entity recognition. Its architecture consists of three layers as shown in Figure 5.

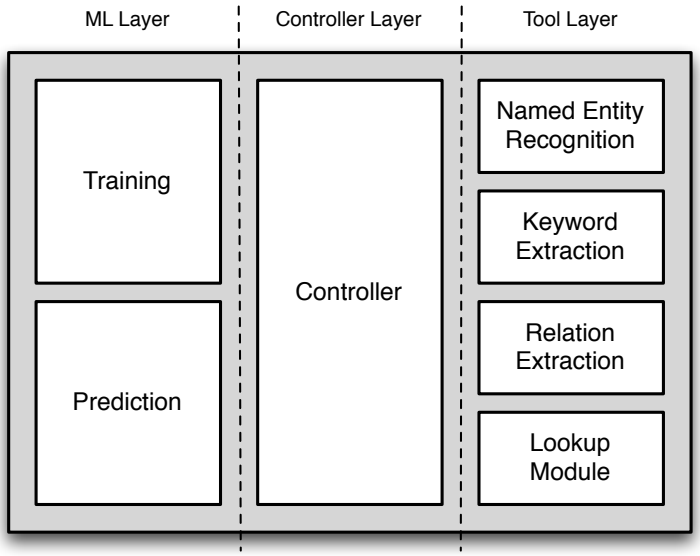


Figure 5: Architecture of the FOX framework.

FOX takes text or HTML as input. Here we use the objects of datatype properties, i.e., plain text. This data is sent to the *controller layer*, which implements the functionality necessary to clean the data, i.e., remove HTML and XML tags as well as further noise. Once the data has been cleaned, the controller layer begins with the orchestration of the tools in the *tool layer*. Each of the tools is assigned a thread from a thread pool, so as to maximize usage of multi-core CPUs. Every thread runs its tool and generates an event once it has completed its computation. In the event that a tool does not complete after a set time, the corresponding thread is terminated. So far, FOX integrates tools for KE, NER and RE. The KE is realized by tools such as KEA[5] and the Yahoo Term Extraction service[6]. In addition, FOX integrates the Stanford Named Entity Recognizer[7] [3], the Illinois Named Entity Tagger[8] [14] and Alchemy[9] for NER.

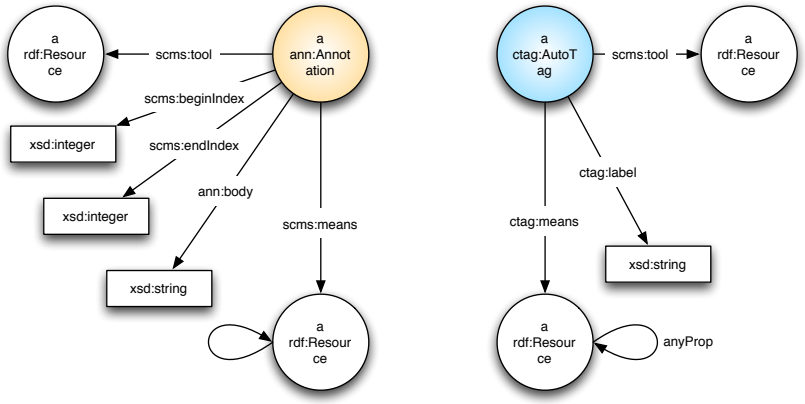
The results from the tool layer are forwarded to the *prediction module* of the *machine-learning layer*. The role of the prediction module is to generate FOX's output based on the output the tools in FOX's backend. For this purpose, it implements several ensemble learning techniques [2] with which it can combine the output of several tools. Currently, the prediction module carries out this combination by using a feed-forward neural network. The neural network inserted in FOX was trained by using 117 news articles. It reached 89.21% F-Score in an evaluation based on a ten-fold-cross-validation on NER, therewith outperforming even commercial systems such as Alchemy.

Once the neural network has combined the output of the tool and generated a better prediction of the named entities, the output of FOX is generated by using the vocabularies shown in Figure 6. These vocabularies extend the two broadly used vocabularies Annotea[10] and Autotag [11]. In particular, we added the constructs explicated in the following:

- `scms:beginIndex` denotes the index in a literal value string at which a particular annotation or keyphrase begins;

[5] http://www.nzdl.org/Kea/
[6] http://developer.yahoo.com/search/content/V1/termExtraction.html
[7] http://nlp.stanford.edu/software/CRF-NER.shtml
[8] http://cogcomp.cs.illinois.edu/page/software_view/4
[9] http://www.alchemyapi.com
[10] http://www.w3.org/2000/10/annotation-ns#
[11] http://commontag.org/ns#

- `scms:endIndex` stands for the index in a literal value string at which a particular annotation or keyphrase ends;
- `scms:means` marks the URI assigned to a named entity identified for an annotation;
- `scms:source` denotes the provenance of the annotation, i. e., the URI of the tool which computed the annotation or even the system ID of the person who curated or created the annotation and
- `scmsann` is the namespace for the annotation classes, i.e, location, person, organization and miscellaneous.



(a) Named Entity Annotation (b) Keyword Annotation

Figure 6: Vocabularies used by FOX for representing named entities (a) and keywords (b)

4 Geo-Spatial Link Discovery

Geo-spatial resources are often described as (ordered) sets of points or as polygons. This way of describing resources differs considerably from the approach followed for most Linked Data resources, which are commonly easiest identified by the means of a label. Consequently, such descriptions have not yet been payed much attention to in the field of link discovery (LD). We addressed this gap by developing ORCHID, a reduction-ratio-optimal approach for LD. ORCHID assumes the LD problem as being formulated in the following way: Given a set S of source instances, a set R of target instances and a distance threshold θ, find the set of triples $(s, t, \delta(s, t)) \in S \times T \times \mathbb{R}^+$ such that $\delta(s, t) \leq \theta$. Given this assumption, the idea behind ORCHID is to address the LD problem on geographic data described as (ordered) sets of points by two means. First, ORCHID implements time-efficient algorithms for computing whether the distance between two polygons s and t is less or equal to a given distance threshold θ. These algorithms rely on the distance between s and t being bound by θ to discard unnecessary distance computations. So far, ORCHID implements the Hausdorff distance for measuring the distance between polygons. Moreover, ORCHID implements a space tiling algorithm for orthodromic spaces which allows discarding yet another large number of unnecessary computations. This algorithm is based on $\mathcal{HR}^3$ [4].

4.1 Preliminaries

While there are several approaches for computing the distance between two polygons [1], a common approach is the use of the Hausdorff distance [13] hd:

$$hd(s, t) = \max_{s_i \in s}\{\min_{t_j \in t}\{\delta(s_i, t_j)\}\}, \tag{1}$$

where δ is the metric associated to the affine space within which the polygons are defined. We assume that the earth is a perfect ball with radius $R = 6378\,km$. It is important to notice that the Hausdorff distance is a metric in the mathematical sense of the term in any metric space. Moreover, the orthodromic distance is also known to be a metric, leading to the problem formulated above being expressed in a metric space.

4.2 ORCHID

Several approaches have addressed the time-efficient computation of Hausdorff distances throughout literature (see [13] for a good overview). Yet, so far, these approaches have not been concerned with the problem of only finding those triples $(s, t, hd(s, t))$ with $hd(s, t) \leq \theta$. In the following, we present several approaches for achieving this goal.

4.2.1 Naive approach

The naive approach for computing $hd(s, t)$ would compare all elements of the polygon $s \in S$ with all elements of the polygons $t \in T$ by computing the orthodromic distance between all $s_i \in s$ and $t_j \in t$. Let $\bar{S}$ be the average size of the polygons in S and $\bar{T}$ be the average size of the polygons in T. The best- and worst-case runtime complexities of the naive approach are then $O(|S||T|\bar{S}\bar{T})$.

4.2.2 Bound approach

A first idea to make use of the bound $hd(s, t) \leq \theta$ on distances lies in the observation that

$$\exists s_i \in S : \min_{t_j \in t}\{od(s_i, t_j)\} > \theta \rightarrow hd(s, t) > \theta \tag{2}$$

This insight allows terminating computations that would not lead to pairs for which $hd(s,t) \leq \theta$ by terminating the computation as soon as a s_i is found that fulfills Eq. (2). In the best case, only one point of each $s \in S$ is compared to all points of $t \in T$ before the computation of $hd(s,t)$ is terminated. Thus, the best-case complexity of the approach is $O(|S||T|\bar{T})$. In the worst case (i.e., in the case that the set of mappings returned is exactly $S \times T$), the complexity of the bound approach is the same as that of the naive approach, i.e., $O(|S||T|\bar{S}\bar{T})$.

4.2.3 Indexed Approach

The indexed approach combines the intuition behind the bound approach with geometrical characteristics of the Hausdorff distance by using two intuitions. The first intuition is that if the distance between any point of s and any point of t is larger than θ, then $hd(s,t) > \theta$ must hold. Our second intuition makes use of the triangle inequality to approximate the distances $od(s_i, t_k)$. In the following, we present these two intuitions formally. We dub the indexed approach which relies on the second intuition alone CS while we call the indexed approach that relies on both intuitions $BC + CS$.

4.2.3.1 Intuition 1: Bounding Circles

Formally, the first intuition can be expressed as follows:

$$\min_{s_i \in s, t_j \in t} \{od(s_i, t_j)\} > \theta \rightarrow hd(s,t) > \theta. \tag{3}$$

Finding the two points s_i and t_j which minimize the value of $od(s_i, t_j)$ requires $O(|s||t|)$ computations of od, i.e., $O(|S||T|\bar{S}\bar{T})$ overall. However, a lower bound for this minimum for all pairs $(s,t) \in S \times T$ can be computed efficiently by using encompassing circles: Let $\mathcal{C}(s)$ resp. $\mathcal{C}(t)$ be the smallest circles that fully encompass s resp. t. Moreover, let $r(s)$ resp. $r(t)$ be the radius of these circles and $\zeta(s)$ resp. $\zeta(t)$ be the centers of the circles $C(s)$ resp. $C(t)$. Then,

$$\min_{s_i \in s, t_j \in t} \{od(s_i, t_j)\} > od(\zeta(s), \zeta(t)) - (r(s) + r(t)) = \mu(s,t). \tag{4}$$

Figure 7 displays the intuition behind this approximation graphically. Note that this equation also holds when the circles overlap (in which case $od(\zeta(s), \zeta(t)) - (r(s) + r(t)) < 0$ as $od(\zeta(s), \zeta(t)) < (r(s) + r(t))$.

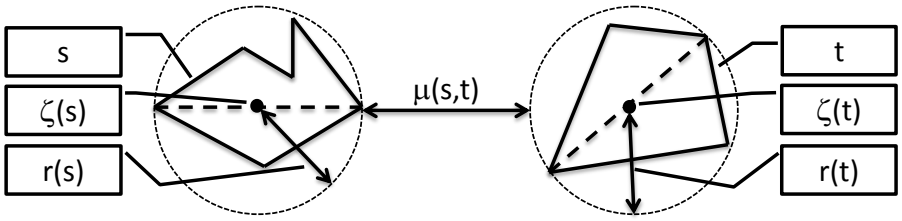


Figure 7: Lower bound of Hausdorff distances based on circles

4.2.3.2 Intuition 2: Distance Approximation using the Cauchy-Schwarz Inequality

Now given that we have computed all distances between all pairs $(t_j, t_k) \in t^2$, we can reuse this information to approximate distances from any s_i to any t_k by relying on the Cauchy-Schwarz inequality in a fashion similar

to the LIMES algorithm presented in [8]. The idea here is that we can compute an upper and a lower bound for the distance $od(s_i, t_k)$ by using the distance $od(s_i, t_j)$ previously computed as follows:

$$|od(s_i, t_j) - od(t_j, t_k)| \leq od(s_i, t_k) \leq od(s_i, t_j) + od(t_j, t_k). \tag{5}$$

For each s_i, exploiting these pre-computed distances can be carried out as follows: For all t_k for which $od(s_i, t_k)$ is unknown, we approximate the distance from s_i to t_k by finding a point t_j for which

$$t_j = \arg\min_{t_x \in t'} od(t_x, t_k) \tag{6}$$

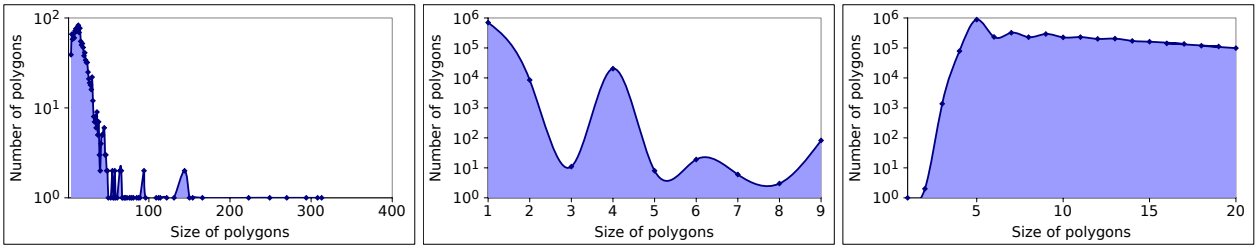
holds, where $t' \subseteq t$ is the set of points t_x of t for which $od(s_i, t_x)$ is known. We call the point t_j an *exemplar* for t_k. The idea behind using one of points closest to t_k is that it gives us the best possible lower bound $|od(s_i, t_j) - od(t_j, t_k)|$ for the distance $od(s_i, t_k)$. Now if $|od(s_i, t_j) - od(t_j, t_k)| > \theta$, then we can discard the computation of the distance $od(s_i, t_k)$ and simply assign it any value $\Theta > \theta$. Moreover, if $|od(s_i, t_j) - od(t_j, t_k)|$ is larger than the current known minimal distance between s_i and points in t, then we can also discard the computation of $od(s_i, t_k)$. If such an exemplar does not exist or if our approximations fail to discard the computation, then only do we compute the real value of the distance $od(s_i, t_k)$.

The best-case complexity of this step alone would be $O(|S||T|\bar{S})$ while in the worst case, we would need to carry out $O(|S||T|\bar{S}\bar{T})$ computations of od. The overall complexity of the indexed approach is $O(|S|\bar{S}^2 + |T|\bar{T}^2 + |S||T|)$ (i.e., that of the bounding circles filter) in the best case and $O(|S|\bar{S}^2 + |T|\bar{T}^2 + |S||T| + |S||T|\bar{S}\bar{T})$ in the worst case.

5 Evaluation

5.1 Experimental Setup

The goal of our experiments was to show that our implementation of the Hausdorff distance is time-efficient w.r.t. and can scale to even large datasets. We thus selected three publicly available datasets of different sizes for our experiments. The first dataset, *Nuts*, contains a detailed description of 1,461 specific European regions.[12] The second dataset, *DBpedia*, contains all 731,922 entries from DBpedia that possess a geometry entry.[13] Finally, the third dataset, LGD, contains all 3,836,119 geo-spatial objects from LinkedGeoData that are instances of the class `Way`.[14] An overview of the distribution of the polygon sizes in these datasets is given in Figure 8. All experiments were deduplication experiments, i.e., we linked each dataset with itself.



(a) Nuts (b) DBpedia (c) LGD

Figure 8: Distribution of polygon sizes

All experiments were carried out on a 32-core server running JDK 1.7 on Linux 10.04. The processors were 8 quadcore AMD Opteron 6128 clocked at 2.0 GHz. Unless stated otherwise, each experiment was assigned 10GB of memory and was ran 5 times. The time-out for experiments was set to 3 hours per iteration. The granularity parameter α of Orchid was set to 1. In the following, we present the minimal runtime of each of the experiments.

5.2 Results

In our first evaluation of Orchid, we measured the runtimes achieved by the three different implementations of the Hausdorff distances on random samples of the Nuts, DBpedia and LGD data sets. We used three different thresholds for our experiments, i.e., $100\,m$, $0.5\,km$ and $1\,km$. In Figure 9, we present the results achieved with a threshold of $100\,m$. The results of the same experiments for $0.5\,km$ and $1\,km$ did not provide us with significantly different insights. As expected the runtime of all three approaches increases quadratically with the size of the sample. There is only a slight variation in the number of comparisons (see Figure 9) carried by the three approaches on the DBpedia dataset. This is simply due to most polygons in the dataset having only a small number of nodes as shown in Figure 8. With respect to runtime, there is no significant difference between the different approaches on DBpedia. This is an important result as it suggests that we can always use the CS or $BC + CS$ approaches even when the complexity of the polygons in the datasets is unknown.

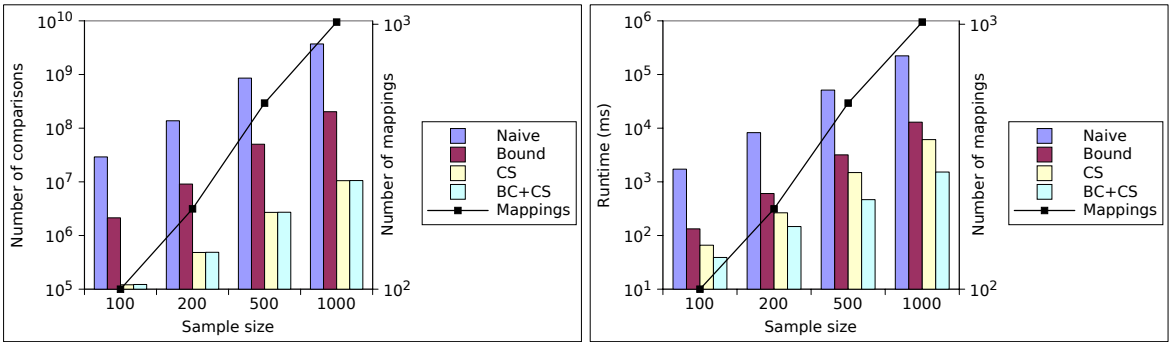
On the two other datasets, the difference between the approaches with respect to both the number of comparisons and the runtime can be seen clearly. Here, the bound implementation requires an order of

[12] We used version 0.9.1 as available at http://nuts.geovocab.org/data/.

[13] We used version 3.8 as available at http://dbpedia.org/Datasets.

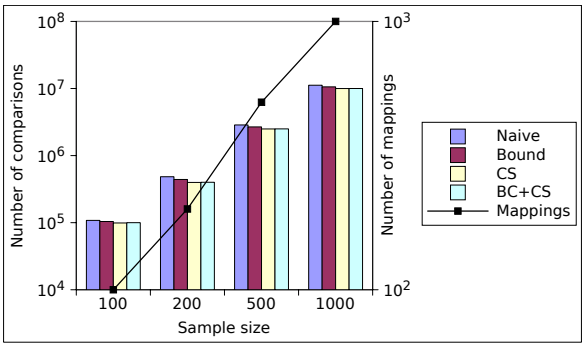
[14] We used the RelevantWays dataset (version of April 26th, 2011) of LinkedGeoData as available at http://linkedgeodata.org/Datasets.

magnitude less comparisons than the naive approach while the indexed implementations need two orders of magnitude less comparisons. The runtimes achieved by the approaches reflect the observations achieved on the comparisons. In particular, the bound approach is an order of magnitude faster than the naive approach. Moreover, the $BC + CS$ approach outperforms the bound approach by approximately one further order of magnitude. Note that up to approximately 1.07% of the comparisons carried out by $BC + CS$ are the result of the indexing step.

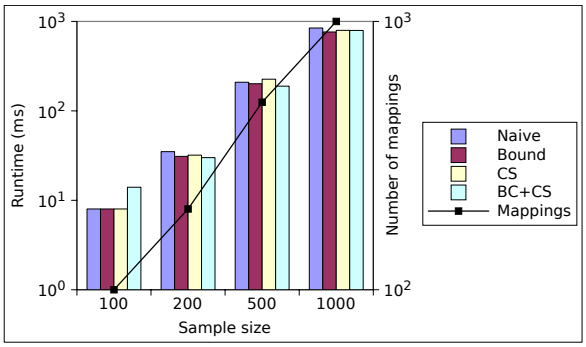


(a) Comparisons on Nuts

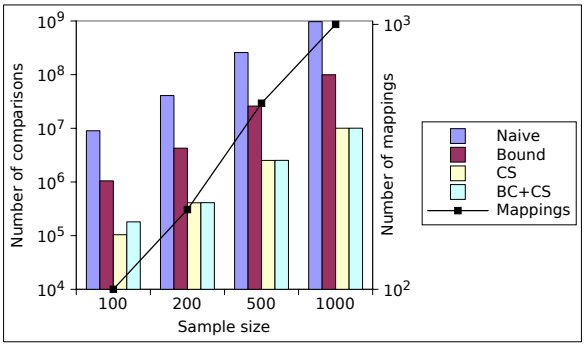
(b) Runtime on Nuts



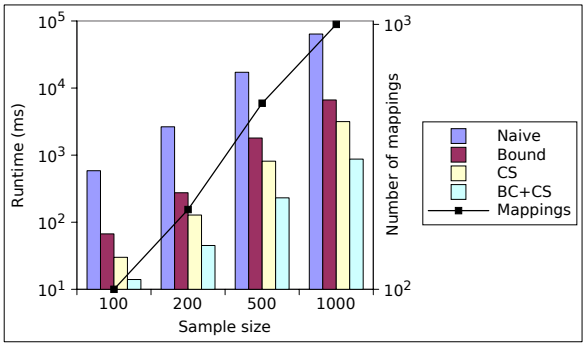
(c) Comparisons on DBpedia



(d) Runtime on DBpedia



(e) Comparisons on LGD



(f) Runtime on LGD

Figure 9: Number of comparisons and runtimes on samples of the datasets.

6 Conclusions

In this document, we presented the GeoLift component for enriching RDF datasets with geo-spatial data. In future work, we aim to implement a graphical user interface on top of GeoLift to enable users to specify their workflows graphically. Moreover, we aim to implement workflow checking functionality. We also presented Orchid, an approach for the time-efficient linking of geo-spatial data. In the next version of the framework, we aim to develop more measures for polygon similarity and evaluate them w.r.t. their robustness so as to provide guidelines for linking geo-special datasets.

References

[1] Mikhail J. Atallah, Celso C. Ribeiro, and Sérgio Lifschitz. Computing some distance functions between polygons. *Pattern Recognition*, 24(8):775–781, 1991.

[2] Thomas G. Dietterich. Ensemble methods in machine learning. In *MCS*, pages 1–15, London, UK, 2000. Springer-Verlag.

[3] J. Finkel, T. Grenager, and C. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *ACL*, pages 363–370, 2005.

[4] Axel-Cyrille Ngonga Ngomo. Link discovery with guaranteed reduction ratio in affine spaces with minkowski measures. In *Proceedings of ISWC*, 2012.

[5] Axel-Cyrille Ngonga Ngomo. On link discovery using a hybrid approach. *Journal on Data Semantics*, 1:203 – 217, December 2012.

[6] Axel-Cyrille Ngonga Ngomo. Orchid – redution-ratio-optimal computation of geo-spatial distances for link discovery. In *Proceedings of ISWC*, 2013.

[7] Axel-Cyrille Ngonga Ngomo and Sören Auer. Limes - a time-efficient approach for large-scale link discovery on the web of data. In *Proceedings of IJCAI*, 2011.

[8] Axel-Cyrille Ngonga Ngomo and Sören Auer. LIMES - A Time-Efficient Approach for Large-Scale Link Discovery on the Web of Data. In *IJCAI*, pages 2312–2317, 2011.

[9] Axel-Cyrille Ngonga Ngomo, Lars Kolb, Norman Heino, Michael Hartung, Sören Auer, and Erhard Rahm. When to reach for the cloud: Using parallel hardware for link discovery. In *Proceedings of ESCW*, 2013.

[10] Axel-Cyrille Ngonga Ngomo, Jens Lehmann, Sören Auer, and Konrad Höffner. Raven – active learning of link specifications. In *Proceedings of OM@ISWC*, 2011.

[11] Axel-Cyrille Ngonga Ngomo and Klaus Lyko. Eagle: Efficient active learning of link specifications using genetic programming. In *Proceedings of ESWC*, 2012.

[12] Axel-Cyrille Ngonga Ngomo, Klaus Lyko, and Victor Christen. Coala – correlation-aware active learning of link specifications. In *Proceedings of ESWC*, 2013.

[13] Sarana Nutanong, Edwin H. Jacox, and Hanan Samet. An incremental hausdorff distance calculation algorithm. *Proc. VLDB Endow.*, 4(8):506–517, May 2011.

[14] Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In *CONLL*, pages 147–155, 2009.