Masters in Informatics Engineering
Dissertation
Final Report

POI Mining and Generation

Filipe Rodrigues
fmpr@student.dei.uc.pt

Coordinators:
Francisco Pereira
Ana Alves

Date: July 14th, 2010

FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Abstract

With the current broad use of location aware devices, the activity of geo-tagging is becoming normal. The most atomic unit of this activity is the Point Of Interest (or landmark) which consists of a pair of Latitude/Longitude coordinates and a tag, normally the name of the place, a word that unambiguously identifies it and, possibly, some extra information such as the category of the POI. We present a solution to automatically extract POIs from various sources on the Web, such as Yahoo, Manta and Yellow Pages, aggregating them in a large database that can be used in navigation, characterization of a place, land use analysis and geo-reference of texts. This solution is also able to detect equivalent POIs between the multiple sources and to automatically classify them to a widespread taxonomy like the North American Industry Classification System (NAICS). We also propose a solution to automatically infer places of interest based on geo-referenced content available on the Internet like geo-tagged photos, blog posts and news feeds, thus giving a different perspective of the city and reducing the dependency on large POI directories.

Acknowledgements

I want to thank Professor Francisco Pereira and Ana Alves for the precious guidance and valuable knowledge they gave me during the last year. I also want to thank Shan Jiang and Professor Joseph Ferreira from the Massachusetts Institute of Technology for their collaboration in this and other related works. Finally, I want to thank the other researchers at AmILab for all the support and for the great working environment they provide.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the increasing number of mobile devices and social networks in the latest years, the amount of geo-referenced information available on the Web is growing at an astonishing rate. Capture devices such as camera-phones and GPS-enabled cameras can automatically associate geographic data with images, which is significantly increasing the number of geo-referenced photos available online. Social networks also have an important role. They are a great medium where users can share information they collect with their mobile devices. As a consequence, the amount of online descriptive information about places has reached reasonable dimensions for many cities in the world.

A point of interest, or POI for short, is a specific point location that someone may find useful or interesting. POIs can be used in navigation, characterization of a place, sociological studies, city dynamics analysis, geo-reference of texts, etc. Such a simple information structure can be used and enriched such that context-aware systems behave more intelligently.

In spite of their importance, the production of POIs is scattered across a myriad of different websites, systems and devices, thus making it extremely difficult to obtain an exhaustive database of such a wealthy information. There are hundreds, if not thousands, of POI directories in the Web, with POIs for places all over the World. These are in fact great sources of information. However, each one uses its own format to represent the POIs and its own taxonomy to classify them. Also, the Web servers that provide POI information (e.g., Yahoo, Manta, Yellow Pages, CitySearch, Upcoming) are mere repositories, and therefore, they do not take advantage of the full potential of such information. In this project, we explore a number of techniques to overcome these problems.

Our work is primarily focused on the extraction, analysis, manipulation and classification of these POIs. Besides that, we are also interested in the automatic inference of places of interest based on geo-referenced information available on the Web, in social networks (such as Flickr and Facebook), blogs, etc. This way, we could complement the other POI sources or, ultimately, stop needing them at all. How good would it be, if we did not need users and enterprise employees submitting their GPS locations to these large POI data sources? Also, since the POIs change over time, their databases can become obsolete unless a constant maintenance is performed.

In this document, we present a multi-threaded approach for the automatic extraction of POIs from Web sources along with an analysis of the collected data. We also propose an algorithm for POI matching that makes use of a string comparison library to identify similarities between POIs from different sources, and various approaches for POI classification. The objective of the latter approaches is for us to be able to classify POIs from different sources to a common and more widespread taxonomy like NAICS (United States, Canada and Mexico) and ISIC (United Nations). Doing so is essential in order to perform a proper analysis of the extracted POIs like, for instance, a land use analysis, which is crucial for urban planning. If the POIs are not mapped to a common taxonomy we wont be able to determine, for instance, how many POIs of restaurants exist in a given area because a POI source may classify them as "Restaurants" and the other as "Eating place". Finally, we present an approach to automatically determine places of interest based on geo-referenced information from Flickr[1], a huge online community mostly dedicated to sharing photographs. This approach is mainly based on clustering algorithms and on a proposed modified version of the common TF-IDF weighting algorithm.

The remainder of this document is organized as follows: chapter 2 provides a summary of the state of the art; chapter 3 describes the different approaches used for the different tasks; chapter 4 explains how we perform validation of our approaches; chapter 5 shows the different results we obtained; chapter 6 presents some final conclusions of the developed work; and finally, chapter 7 presents some final thoughts on the research work carried out during the last year.

[1] http://www.flickr.com

Chapter 2

State of the art

In this chapter we discuss the state of the art that is important to understand some of the developed approaches and their contextualization.

2.1 POI extraction

In this section we contextualize our POI Extraction software with other existing alternatives.

There are no specialized tools for POI extraction that we know about. However, there are web scraping tools that allow users to visually select parts of webpages they want to extract, define repetition patters and other options, and extract the selected information directly into a user-defined file or database. Some examples of such tools are:

- Visual Web Ripper (http://www.visualwebripper.com)
- Web Content Extractor (http://www.newprosoft.com)
- Web Scraper Plus+ (http://www.velocityscape.com)

The prices of these tools range from $99 to $749, but a more professional tool can go up to $2499[1]. Besides the high price, this kind of tools are not very flexible, and therefore they cannot be applied in all situations. They do not deal with problems like hidden information in the HTML or in the JavaScript code, request limitations imposed by the server, information spread along multiple pages, poorly formatted web

[1] http://www.screen-scraper.com

pages, authentication requirements, URL manipulation, etc. In summary, these tools are appropriate to small simple scraping jobs. For larger, highly customized tasks it is better and cheaper to develop our own scraper tool. Furthermore, the development time is not that long if we use a framework that automatically performs the connection handling, database management, and other required tasks.

2.2 Land Use and Spatial Analysis

This section gives some insight about land use and spatial analysis, which are very important applications of POI data. These kind of studies allow us to understand some key aspects of the city and the land, that are a crucial input for urban planners. However, in order for the POI data to be properly used, it is necessary that it correctly classified in a adequate and mature taxonomy like the NAICS, which is widely used for this kind of studies. Without the use of a good taxonomy, POI data is pretty much chaotic and very hard to make use of. We also present two studies that could benefit from well-classified POI data. These studies help the reader further understand the importance and wide range of applications of the POI data.

Land use refers to how the land is generally used, whether it is residential, commercial, industrial, open space, etc. Land use planning can be defined as: a systematic attempt to minimize the adverse effects land changes have on society and environments and to maximize human benefits. Another definition states that land use planning is defined as the process of protecting and improving the environments in a city through the proper use and development of land. Therefore, land use planning is a process for determining how land is used, both now and in the future [1].

At a more mundane level, interaction occurs between everyday behavior and future land use patterns: existing land use arrangements in part determine where people live, where they work and how and when they travel there, where they shop, where they play, etc., while such behavior in turn helps to shape future land use patterns [2].

The most common method for land use analyses is surveys. However, performing surveys in large areas is very expensive and takes a lot of time. Furthermore, a survey only lets us analyze the land use in a given time. Keeping a good history of the land uses requires frequent surveys. Thus, other methods for gathering land use information are required, like for instance, POI-based approaches. For example, the company GroundSure[2] uses its own POI data to provide an up-to-date picture of land use and identify

[2]http://www.groundsure.com

current potential sources of contamination and pollution, producing reports that are very useful to urban planners.

Spatial analysis has long been a topic of interest of researchers, who seek a comprehensive understanding on how the city behaves in different perspectives and its impact in the economy. Methods for analyzing spatial (and space-time) data have already been well developed by statisticians [3] and econometricians [4].

On the econometrics perspective, Currid et al [5] try to understand the importance of agglomeration economies as a backbone to urban and regional growth, by identifying clusters of several "advanced" service sectors (professional, management, media, finance, art and culture, engineering and high technology) and comparing them in the top ten populous metropolitan areas in the U.S. They concluded that there are three spatial typologies of growth in the advanced services within U.S. urban regions. These typologies allowed them to understand qualities of place in general and of places specifically that drive the agglomeration of advanced services.

On a particular case study of the biotech industry in the U.S., Sambidi and Harrison[6] also analyze factors affecting site-selection of industries, testing the hypothesis of spatial agglomeration economies in that industry and confirm it using spatial econometrics. In the same topic, Arbia[7] classifies the spatial processes of individual firms into a birth process (new firms) and a growth process (existing firms), and proposes a model of economic activities on a continuous space also with the purpose of studying the geographical concentration of economic activities and analyze the economic behavior of individual firms.

2.3 Ontology matching

In this section we talk about ontologies and ontology matching techniques which are the basis of one of the POI Classification approaches later presented in this document.

An ontology is a formal representation of a set of concepts within a domain and the relationships between these concepts. It is used to reason about the properties of that domain, and may be used to define the domain.

Ontologies are commonly described in Web Ontology Language (OWL). OWL is intended to be used when the information contained in documents needs to be processed by applications, as opposed to situations where the content only needs to be presented to humans. OWL can be used to explicitly represent the meaning of terms in vocabularies

and the relationships between these terms. OWL has more support for expressing meaning and semantics than XML, RDF, and RDF-S, so OWL goes beyond these languages in its ability to represent machine interpretable content on the Web[8].

As the number of ontologies that are made publicly available and accessible on the Web increases steadily, so does the need for applications to use them. A single ontology is no longer enough to support the tasks envisaged by a distributed environment like the Semantic Web. Multiple ontologies need to be accessed from several applications. Ontology mapping could provide a common layer from which several ontologies could be accessed and hence could exchange information in semantically sound manners. Developing such mappings has been the focus of a variety of works originating from diverse communities over a number of years [9].

Even though there is a lot of research and development in the ontology mapping area, only a small set of tools is publicly available for general use. We went through some research projects like X-SOM [10], COMA/COMA++ [11], MAFRA [12], PROMPT [13] and GLUE [14], in order to select the most appropriate for our goals.

COMA++ is one of the most popular tools for Ontology Matching that are available to the community. COMA++ uses a composite approach to combine different match algorithms like string matchers, reuse-oriented matchers and statistics. Using a flexible infrastructure for combining and refining matcher results, the match processing is supported as a workflow of several match steps. They implemented specific matching strategies (i.e. workflows) for context-dependent, fragment-based, and reuse-oriented matching, respectively:

- Context-dependent Matching: This kind of matching takes into account the path to the node instead of just the node for matching the ontologies. Context-dependent matching is necessary for schemas with shared elements, because the latter exhibit multiple contexts which should be differentiated for a correct matching.
- Fragment-based Matching: To cope with large schemas, COMA++ implements a fragment-based match processing approach. Following the divide-and-conquer idea, it decomposes a large match problem into smaller subproblems by matching at the level of schema fragments. With the reduced problem size, they aim not only at better execution time but also at better match quality compared to schema-level matching.
- Reuse-oriented Matching: With this strategy they pursue the reuse of previously determined match results.

X-SOM also appeared to be a very interesting tool. However, it was not available to the public. X-SOM has a modular and extensible architecture that automatically combines several matching techniques by means of a neural network. It uses many interesting matching modules such as:

- Jaro module (syntactical): finds the similarity of two terms using an algorithm based on Jaro String Similarity.
- Levenshtein module (syntactical): computes the Levenshtein distance between two terms and evaluates their degree of similarity
- WordNetSimilaritymodule(syntactical): uses the WordNet thesaurus, builds a path of hypernym relations between two terms, and calculates their similarity with respect to this path.
- Googleapi module (syntactical): queries the Google search engine and, based on the number of results, computes the similarity of two terms.

Using a three-layer neural network, the framework then automatically learns, on a domain-independent training set, how to weigh each matching algorithm in order to maximize overall precision and recall. The framework also allows the weights of each matching module to be set manually, by a human expert. However, this solution implies that the operator knows in advance how reliable the various techniques are, in order to assign appropriate weights.

For further information about these and other ontology matching tools/projects like MAFRA, PROMPT and GLUE, the interested reader can take a look at [9].

2.4 Machine Learning applications in Space Analysis

This section describes some applications of Machine Learning algorithms mainly in the field of Space Analysis.

The applications of machine learning algorithms in classification tasks are vast and cover diverse areas such as Medicine, Transportation and Urban Planning. In the latter, land-use/land-cover information has long been recognized as a very important material [15]. However, as Fresco [16] claimed, accurate data on actual land-use cannot be easily found at both global/continental and national/regional scales.

In order to cope with these problems, automatic approaches to classify land use are being developed using distinct techniques.

A common approach to infer land-use/land-cover, is to use satellite imagery. However, while these approaches have already proven to get good results, they are more suited to land-cover inference, which is considered somehow different from land-use by many authors. Campbell [17], for example, considers land-cover to be concrete whereas land-use is abstract. That is, land-cover can be mapped directly from images, while land-use requires land-cover and additional information on how the land is used. Danoedoro [18] tries to improve land-use classification via satellite imagery by combining spectral classification, image segmentation and visual interpretation. Although he showed that satellite imagery could be used for generating socio-economic function of land-use at 83.63% accuracy, he is the first to recognize that applying such techniques to highly populated areas would be problematic.

Li et al. [19] also use data mining techniques to discover knowledge from GIS databases and remote sensing image data that could be used for land use classification. In the field of remote sensing, Bayes classification (or maximum likelihood classification) is most widely used and, for most multi-spectral remote sensing data, the Bayes method classifies the coarse classes correctly, such as water, residential area, green patches, etc. But usually more detailed classification is required in land use classification. In order to subdivide some of the classes, Li et al. propose the use of inductive learning techniques, particularly the C5.0 algorithm. By using these techniques they were able to get an overall accuracy of 89%. Comparing their final result with the result produced only by Bayes classification, the overall accuracy increased 11%.

An alternative to satellite imagery is the POI data. Using a large commercial POI database, Santos and Moreira [20] create and classify location contexts using decision trees. They start by identifying clusters by means of a density-based clustering algorithm (Shared Nearest Neighbor algorithm). Using the identified clusters they then define areas (or regions) that represent them through the application of a concave hull algorithm they developed. Finally, making use of the C5.0 algorithm, they classify a given location according to such characteristics as the number of POIs in a cluster, the size of the area of the cluster and the categories of the POIs within the cluster.

In order to use POI data for the classification of places and land-use analysis, POI classification is an essential task. Griffin et al. [21] use decision trees to classify GPS-derived POIs. However, they refer to POIs as “personal” locations to a given individual (i.e. home, work, restaurant, etc.). The main goal of their approach is then to automatically classify trips. In their approach, they start by determining clusters of trip-stops (i.e. stops that took more than 5 minutes) using a density-based clustering algorithm (Dbscan). Then, they make use of the C4.5 algorithm to classify the generated clusters as being “home”, “work”, “restaurant”, etc., based on the time of the day and the

length of the stay. However, to our best knowledge, no previous approaches have been made to classify POIs to a classification system like NAICS. The latter is widely used for industry classification and has already been used, for instance, to classify Web Sites through machine learning techniques [22].

2.5 POI generation

The availability of map interfaces and location-aware devices leads to a growing amount of unstructured geo-referenced information available on the Web. Due to the underlying potential of this kind of information, many researchers are trying to find a way to make sense out of it. One potential use of this information is the identification of new meaningful places in a fully automated way.

Ahern et al. [23] proposed a mechanism that uses unstructured text labels (i.e., tags) from geo-referenced photos available on Flickr, a very popular photo-sharing website, with the objective of generating aggregate knowledge in the form of representative tags for arbitrary areas in the world. In their research they used a dataset consisting of 6 million public geo-referenced (or "geo-tagged") photographs on Flickr. Using this large dataset, they create clusters to represent a group of photos. In order to do that, they use the k-Means clustering algorithm, based on the photos latitude and longitude. The stopping condition used for the k-Means algorithm is when each clusters centroid movement drops below 50 meters. Then, if any two cluster centroids are too close to each other they merge the clusters.

Once the clusters have been determined, the system scores the clusters tags to see if it is possible to extract representative tags for each cluster. One of the factors they use for scoring is TF-IDF (term frequency $\times$ inverse document frequency) which assigns a score to each tag in a cluster according to how specific it is to that cluster, i.e., how well it represents that cluster. TF-IDF is often used in information retrieval and text mining. The TF-IDF weight is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus. Therefore, their approach assigns a higher score to tags that have a larger frequency within a cluster compared to the rest of the area under consideration. The assumption they make is that the more unique a tag is for a specific cluster, the more representative the tag is for that cluster. Besides TF-IDF, they also consider the user information to help them scoring the photos. Here the assumption is that a user is likely to assign to most of the photos he takes the same tag(s).

They faced some issues when analyzing the Flickr data, namely noise (e.g. photos with tags that are not relevant to the location) and errors (e.g., photos that are geo-tagged incorrectly).

Regarding visualization, they developed a tool, World Explorer, which helps expose the content of the data, using a map interface to display the derived tags and the original photo items.

In order to evaluate their results Ahern et al. [23] focused themselves on collecting qualitative feedback regarding the visualization and the effectiveness of the extracted tag data. In particular, they were interested in examining whether recruited participants found the application useful and for which purpose. The results they obtained were very promising, with some participants nostalgically remembering places they visited in the past.

Twaroch et al. [24] suggested a way to decide whether a given particular named entity is in fact a place, by using spatial prepositions (e.g., in, inside, within, at, near, etc) and document counts provided by web search engines such as the Yahoo BOSS API, in order to measure the cognitive significance of landmarks. For their research, they used 2500 distinct location entries mined from a social web source (Gumtree[3]) for the region of Cardiff, UK. The data represented location tags freely entered by people to sell/buy items or make social contacts. They started by flittering out tags that contained addresses or numbers using regular expressions. Then, they defined a trigger phrases list using special proposition like in, inside, within, at, near, etc. Finally, they counted the number of documents returned by web queries using Yahoo's BOSS API for each of the candidate names. In this way, they were able to evaluate their relevance, thus measuring the cognitive significance of landmarks.

Mummidi et al. [25] use collections of user-defined pushpins in a map to discover new POIs. On their website, they allow users to create collections of geographically anchored pushpins which they can tag and classify without a predefined taxonomy of categories. Their data mining solution then extracts, from each pushpin, candidate POI phrases from its title and notes/tags, representing them as n-grams. Then it finds geometric clusters of these pushpins using dendrograms, which manifests a hierarchical agglomerative clustering technique. Once the clusters are determined, they examine the pushpins n-grams and other features for likely POI names using TF-IDF along with other parameters they consider relevant. TF-IDF is used as follows: if a given text phrase is mentioned frequently in a cluster, but infrequently elsewhere, that increases the confidence that the phrase name is a POI.

[3]http://www.gumtree.com

One negative aspect is that they do not deal with misspelling errors and abbreviations, which are very common in this kind of volunteered geographic information (VGI). Their validation is based in human contribution through a survey, which is somewhat imprecise. Regarding the results, the inquired users did not identify a good percentage of the POIs they generated. However, most of the POIs they did recognized were deemed correct.

Jaffe et al. [26] proposed a framework for automatically selecting a summary set of photos from a large collection of geo-referenced photographs. They start by applying a modified version of the Hungarian clustering algorithm to their collection of photographs. Then they score the different clusters depending on several factors like tag-distinguishability (determined using TF-IDF), photographer-distinguishability (also determined using TF-IDF) and density of the cluster. In order to choose a photo to represent each cluster, their algorithm picks a photo with the tags that best match the top tags for that cluster according to the tag-distinguishability parameter.

An initial evaluation of their implementation in a set of geo-referenced photos showed that their algorithm and visualization perform well, producing summaries and views that are highly rated by users.

Chapter 3

Approaches

3.1 POI extraction

Extracting POIs from Web resources is essentially a Web scraping task. There are multiple ways to scrape a Web page, being the most commonly used: XPath, string manipulation and regular expressions. Our approach uses regular expressions because we found it to be the most robust and flexible. Other approaches are more error prone and require more frequent patching and updating. The use of regular expressions also makes the source code a lot clearer. A complete comparison between approaches can be found on appendix A.

3.1.1 Challenges

During the implementation of the POI Extractor application we came across many different challenges. This section describes the most relevant ones.

Since POI data is scattered across various sources in different formats, it was necessary to create a relational database schema that could accommodate POIs with different levels of information. In appendix B the reader can see the E-R model of our database. We were also careful while defining the POI identifiers. We preferably try to use the POI id used by the POI source, but this is not enough to guarantee uniqueness. Therefore, we use a composite id, by concatenating the source, coordinates, and internal id (or name alternatively) of the POI as follows:
<POI_source>_<latitude>_<longitude>_<internal_id_or_name>
(e.g., yahoo_42.358529_-71.054036_58351151,
manta_42.359730_-71.101450_dnbcompany_7d1wjd,
yp_42.4144264_-71.1260849_Charles A Jones).

In many cases, POI websites provide a developers API, using REST or SOAP Web services, which facilitates programmers' work. However, most of the websites still do not have one, besides, some of these websites have poorly formatted HTML, which makes the scraping task extremely hard.

Two interesting examples are the cases of Yahoo and Manta. Yahoo provides an API for their POIs. However, the API does not show the complete information it has about each POI, so, we also have to go to the Yahoo's page about that POI to retrieve the missing information in the API. Furthermore, when we search the Yahoo's API for POIs in a given zip code, it only returns a maximum of 250 results, and considering that a zip code covers a wide area, 250 is far from the total number of POIs that Yahoo really has for that zip code. In order to extract a more complete set of POIs for a given zip code, we use their taxonomy to refine the search, thus, instead of one request for a zip code, we make more than a thousand, each one for a different category of the taxonomy. In this way, we increase the maximum possible number of extracted POIs from 250 to more than 250000. Because Manta doesn't provide an API it doesn't raise these problems; however, its search engine is extremely difficult to use and understand. We extract POIs from Manta by parsing the HTML of the city we were interested in, in order to get the URLs for the Manta Web pages containing the POIs for the different categories.

Due to its nature, POI extraction applications are generally I/O bound, i.e., the time it takes to complete a computation is determined mainly by the period of time spent waiting for input/output operations to be completed. In order to cope with this, we developed a multi-threaded application based on the master-worker model, using threads to make multiple simultaneous requests to the Web server so that when some threads are, for example, analyzing the HTML response or inserting information in the database the others can be making new requests to the server, thus maximizing the throughput.

Unfortunately, this leads us to our next issue: most of the Web sources have request limits, which often present themselves in two different ways (based on the request rate or based on a fixed request limit per day). For example, Yahoo has a limit of 5000 requests per IP address per day and Yelp has a limit of just 100 requests. Furthermore, they have a kind of DoS (Denial of Service attack) defense mechanism that blocks IP addresses making lots of requests per second. The user must be aware of the legal implications of over-requesting because many websites have restricted policies against that. In our case, we were only interested in the POI data just for our study areas/cities, mainly Boston and New York. Our use of this data is legal but due to project time constraints, we needed to speed up considerably the data acquisition. We did not perform any kind of massive attack to the POI data servers, but we did have to use a parallelized method

of data collection. In order to deal with that, we made use of some of the thousands of HTTP proxies, also known as Web proxies or anonymizers, available throughout the Internet. These proxies generally attempt to anonymize web surfing. The server receives requests from the anonymizing proxy server, which works as an intermediary, and thus the server we are connecting to does not receive information about the end user's address. Typically, proxy users are merely interested in anonymity for added security, hiding their identities from potentially malicious websites, for instance, or on principle, to facilitate constitutional human rights of freedom of speech. Therefore, proxies can be very useful to our POI Extractor. The ideia is quite simple: since Web servers have request limits per IP address and, since each of these HTTP proxies has its own IP address, we can use them to overcome the request limitations presented by the Web servers. Another interesting application of HTTP proxies is to overcome country-based website restrictions. For instance, the american version of the Yellow Pages website is not accessible to portuguese IP addresses for some reason. Therefore, the only way to retrieve POI data from their website was to use proxies, particularly, american proxies.

Using HTTP proxies allowed us to reduce the POI extraction time to a few weeks, but remember that using this solution requires an up-to-date proxy list which is difficult to get because they are constantly updating their Web address and their state (i.e., online or offline).

Still concerning the request limitations, our POI Extractor allows users to define a random interval between requests to the server. Some Web servers are so sensitive to multiple simultaneous requests that even if one makes one request every ten seconds for a long period of time, it will be blocked. Therefore, we implemented a sleep mechanism that forces the Extractor to wait a given period of time before it makes another request. If the application still gets blocked, it automatically increases this interval. Of course the proxies alternative might also address this problem, but the user's IP address might get blocked by the proxy instead and, since for a common user extraction time is not crucial, interleaving the requests is recommended instead of using proxies.

Again, besides not having violated any principle of the websites, we did not put in risk any of the used systems or raised any particular concern. Globally(e.g. for Yahoo), the amount of data extracted is irrelevant.

3.1.2 Software architecture

As mentioned before in section 3.1.1 our POI Extractor is a multi-threaded application based on the master-worker model. Figure 3.1 depicts the global architecture of the

application.

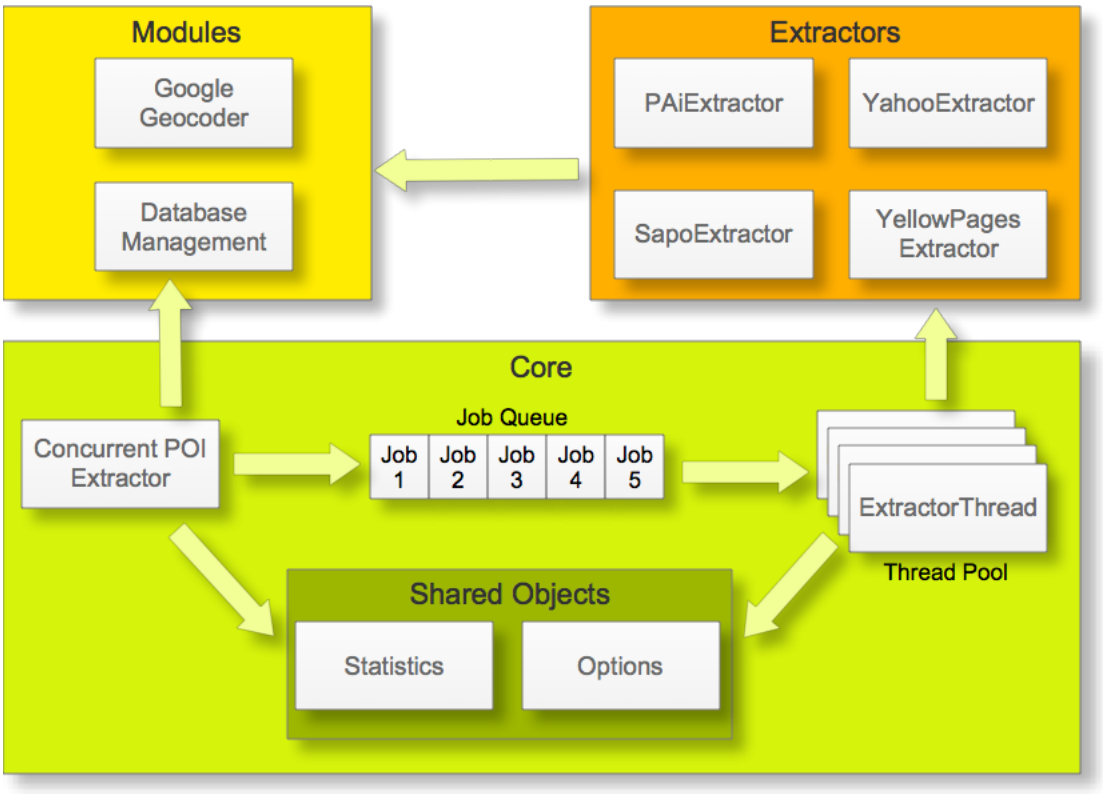

FIGURE 3.1: Architecture of the POI Extractor

The architecture of the application was specified in order to be easily extended. In this way, if one wants to extract POIs from a different source than the ones provided, she can plug-in her own extractor and make use of the multiple resources that our framework provides (HTTP status code checking, proxies, geocoding, reverse geocoding, etc), significantly decreasing the development time, since the developers job is constrained to the definition of regular expressions to get the information she wants. The rest is almost all managed by the framework.

Our POI Extractor is currently able to extract POIs from Yahoo[1], Manta[2], City Search[3], Yellow Pages[4], Boston Globe[5], Upcoming[6], Yelp[7], Sapo[8] and Páginas Amarelas[9].

Figure 3.2 depicts the main user interface of the POI Extractor application.

[1] http://local.yahoo.com
[2] http://www.manta.com
[3] http://www.citysearch.com
[4] http://www.yellowpages.com
[5] http://www.boston.com/bostonglobe/
[6] http://upcoming.yahoo.com
[7] http://www.yelp.com
[8] http://www.sapo.pt
[9] http://www.pai.pt

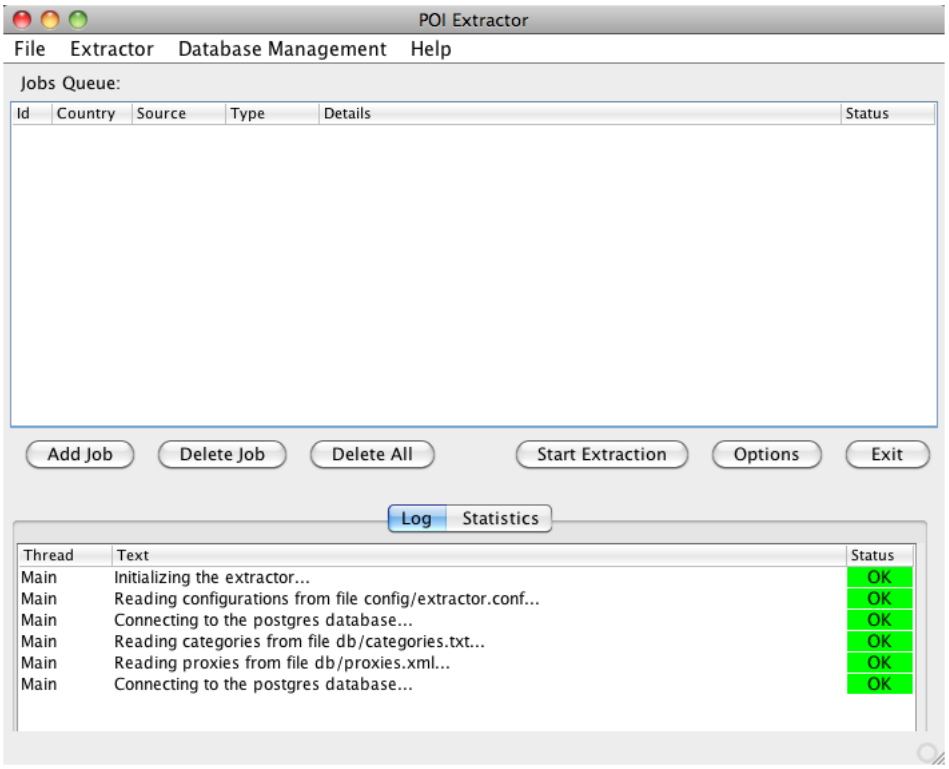

FIGURE 3.2: User Interface of the POI Extractor

Another feature worth mention is the ability of the framework to identify similar POIs, i.e., POIs from multiple Web sources that refer to the same place. The next section explains the POI matching algorithm in detail.

3.2 POI matching

When we are extracting POIs from multiple sources, it is important to have a way to identify similar POIs, so that we do not end up with redundant information and also to collect as much information as possible about a given place. This requires a way to identify similarities based, not only in proximity, but also in name likeness. Since most of the POIs sources are dependent on user contribution, submitting and updating information about places, it is not feasible to rely only on a naïve string comparison between POI names. Also, in POI matching, it is crucial to have good precision and recall results. Precision can be seen as a measure of exactness or fidelity, whereas recall is a measure of completeness. Precision is defined as the number of true positives (i.e. the number of items correctly labeled as belonging to the positive class) divided by the total number of elements labeled as belonging to the positive class, while recall is defined as the number of true positives divided by the total number of elements that actually

belong to the positive class (i.e. the sum of true positives and false negatives). Our ultimate objective is then to get a good recall while keeping the precision close to 100%.

Our approach makes use of the JaroWinklerTF-IDF class from the SecondString[10] project [27] to identify close names, ignoring misspelling errors and some abbreviations. SecondString is an open-source Java-based package of approximate string-matching techniques developed by researchers at Carnegie Mellon University.

We also make use of the URL of the POIs official website, when available, to identify both matches that are not recognized by using only the proximity and the name similarity and also some mismatches that would be considered matches otherwise.

It's important to note that we do not really score the similarity between POIs. We just define thresholds that let us decide if they refer to the same place or not.

Taking this into account, two POIs will be considered similar by our algorithm if they fit into one of the following groups:

- The distance between the two POIs is less than 80 meters, the name similarity is above 0.70 and one or both POIs do not have website information.
- The distance between the two POIs is less than 80 meters, the name similarity is above 0.70 and the website similarity is higher than 0.60.
- The distance between the two POIs is less than 80 meters, the name similarity is above 0.60 and the website similarity is higher than 0.95.

The various thresholds presented were obtained through an analysis of results for a first experiment we made using low thresholds followed by a iterative process of threshold tuning, experimentation and evaluation of the results obtained. The tuning process was not completely blind since, besides comparing the changes made by the previous tuning, we also had the similarity values obtained for the different metrics (i.e. name and website likeness and euclidian distance of the location). Therefore, in each iteration we had a closer idea where the threshold should be.

We also considered using the POI categories to further improve our algorithm. However, we soon realized that doing so, although it might improve recall, it would certainly reduce the precision due to the lack of coherency between the taxonomies of the different POI sources. The only possibility would be to use a common taxonomy to further refine the matches. Unfortunately, that would require both POIs to be classified in the same taxonomy. In the next section, we describe some of the approaches we developed to classify POIs using a common taxonomy.

[10] http://secondstring.sourceforge.net

3.3 POI classification

In this section, we describe the different approaches we tested to classify POIs to a common taxonomy, which would allow, for instance, many interesting land use studies.

In Urban Planning, the classification of space or *land use* has been traditionally a burdensome and manual task. Until recently, the pace of change in available data was still slow enough to support those approaches, but currently the massive quantity and update rate of geo-referenced locations (POIs) available in the web demands for an automatic approach.

Category information from POI sources can allow us to classify POIs to a given taxonomy. Since in many POI sources each POI is assigned more than one category, the number of possible combination can be huge. However, finding mappings between the source taxonomy and the target taxonomy is not always a trivial task. Consider the following mappings:
"Newspaper Publishers" -> "Newspaper Publishers"
"Newspapers Printing" -> "Newspaper Publishers"
"Laboratories" -> "Research & Development in Biotechnology"

Even though the first mapping is obvious, the other two are not, specially the last one. This is mainly due to the different levels of granularity and the names of the categories. The fact that the POIs can belong to multiple categories can help differentiate target categories, thus fixing some granularity issues. However, sometimes this is not enough, and in order to determine such mappings it would be necessary additional information about the POIs, to find the correct mapping. At this point, semantic annotations could help us disambiguate POIs with similar category sets.

We are particularly interested in classifying POIs to more widespread taxonomies like NAICS[11] (U.S., Canada and Mexico), ISIC[12] (United Nations) or CAE[13] (Portugal). All the responsible entities of these classification systems provide a complete listing of the categories online.

On our approaches, we tried to classify POIs to NAICS mainly because most of the data we have is from North America, particularly from Boston and New York.

The North American Industry Classification System (NAICS) is the standard used by Federal statistical agencies in classifying business establishments for the purpose of collecting, analyzing, and publishing statistical data related to the U.S. business economy

[11] http://www.naics.com
[12] http://unstats.un.org/unsd/cr/registry/isic-4.asp
[13] http://www.ine.pt/ine_novidades/semin/cae/CAE_REV_3.pdf

[28]. NAICS was developed under the auspices of the Office of Management and Budget (OMB), and was adopted in 1997 to replace the old Standard Industrial Classification (SIC) system.

NAICS is a two through six-digit hierarchical classification code system, offering five levels of detail. Each digit in the code is part of a series of progressively narrower categories, and the more digits in the code signify greater classification detail. The first two digits designate the economic sector, the third digit designates the sub-sector, the fourth digit designates the industry group, the fifth digit designates the NAICS industry, and the sixth digit designates the national industry. A complete and valid NAICS code contains six digits [29].

Figure 3.3 shows part of the NAICS hierarchy.

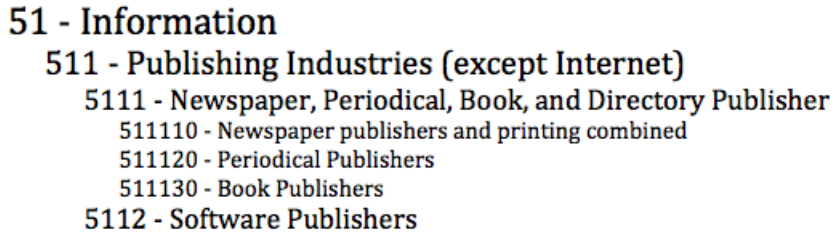

FIGURE 3.3: Example of the NAICS hierarchy

For further information about the NAICS categories the interested reader should go to appendix E.

Albeit we are interested in classifying POIs to NAICS, it would be possible to apply our approaches to ISIC or CAE, since the methodology would be analogous.

In our experiments we classify POIs for different NAICS levels (i.e. NAICS categories with different granularities), particularly two, four and six-digit NAICS codes. This choice is typical in Urban Planning depending on the study at hand (e.g. level 2 allows to analyze economic sectors, while level 6 goes to the level of the establishment specificities).

After comparing several classification approaches with and without enriched information obtained from semantic annotations, we apply the results to the urban modeling task of estimating employment size at a disaggregated level. This task is traditionally made at a coarser level (Traffic Analysis Zone, Census Tract or Block Group level) than what could be now possible. This part of the work is done in collaboration with Shan Jiang (shanjang@mit.edu) and Professor Joseph Ferreira (jf@mit.edu) at MIT. We provide a brief summary of this work in appendix D.

The following subsections present three different approaches we used in order to address the classification issue.

3.3.1 Ontology Matching approach

The idea in this approach was to use ontology mapping to determinate the mappings between two taxonomies. So, the first step was to get the complete source and target taxonomies. As said before, NAICS provides the complete taxonomy online. However, for some POI sources on the Web, it was necessary to gather the taxonomies in a different way. Although some websites like Yelp provide the complete hierarchy of their categories, most of them do not. Instead, they require us to scrape throughout a series of pages, usually searching the main categories pages for their subcategories and so forth. That was the case of the Yahoo and the Yellow Pages taxonomies. On the other hand, getting the Manta taxonomy was much harder, since the Manta website does not provide a Web page with all the top-level categories of their taxonomy. In order to infer the Manta taxonomy, we had to analyze the Manta webpages for the POIs from that source, to get the list of the higher level categories of a POI. For example, if we browse the Manta webpage for the POI "Adam Young Inc" we get the following category information: "Advertising & Marketing > Radio, Television, and Publishers' Advertising Representatives > Television and Radio Time Sales". This indicates that the bottom level category "Television and Radio Time Sales" is part of the "Radio, Television, and Publishers' Advertising Representatives" category, which in its turn, is part of the "Advertising & Marketing" top-level category. By processing this information for all Manta POIs in our database we were able to approximate the taxonomy Manta uses.

Once we had both the source and target taxonomies, we used a conversion tool we developed to convert them to ontologies, using OWL Lite to describe them.

OWL Lite uses only some of the OWL language features and has more limitations on the use of the features than OWL DL or OWL Full. For example, in OWL Lite, classes can only be defined in terms of named superclasses (superclasses cannot be arbitrary expressions), and only certain types of class restrictions can be used. Equivalence between classes and subclass relationships between classes are also only allowed between named classes, and not between arbitrary class expressions. Similarly, restrictions in OWL Lite use only named classes. OWL Lite also has a limited notion of cardinality - the only cardinalities allowed to be explicitly stated are 0 or 1 [8]. Although its limitations, OWL Lite is more than enough to describe a taxonomy, which is an example of a very simple ontology, where the only relationship between classes is the "subClassOf" relationship.

Using the two generated ontologies, we used COMA++ [11], an ontology mapping tool, to find the mappings between them. Using the mappings that COMA++ generated, we then developed a tool to classify the POIs. This tool, checks the mappings for each

of the categories of a POI, and classifies the POI according to the mapping with the highest score according to COMA++.

3.3.2 WordNet approach

In this approach, we use WordNet[14] to find synonyms that will help us determine the closest match in the target taxonomy based on a string comparison library.

WordNet is a large lexical database of English, developed under the direction of George A. Miller, where nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Using Wordnet, we are able to generate different category descriptions with the same meaning, but that might be more similar to the category we are trying to match. This way, if we are trying to classify a POI with the category "Automobile Dealers" to NAICS, we can use WordNet to get the synsets for the word "Automobile" and therefore get the category "Car Dealers", which can then be easily matched to the NAICS code "441110 - Car Dealers" using a string comparison library.

Using WordNet is important to generate multiple equivalent definitions of a given category, but the NAICS itself provides multiple definitions for its own categories, which increases even further the probability of finding a match.

With the multiple category definitions generated with WordNet for each of the categories of a POI, we search the NAICS category list in order to find the closest match. We make use of the string comparison library mentioned in section 3.2 to find the NAICS definition that is more similar to the one generated. The NAICS code correspondent to the highest scored match for all the original categories is then assigned to the POI.

The classification procedure can be summarized as follows:

1. Get the categories from the POI we are trying to classify;
2. Generate multiple equivalent definitions for each category using WordNet;
3. Search the NAICS database for the closest global match using a string similarity library.

3.3.3 Machine Learning approaches

In this approach we use machine learning techniques to estimate the best NAICS code of a POI.

[14] http://wordnet.princeton.edu

3.3.3.1 POI Sources

Our data for this approach consists of a large set of POIs extracted from Yahoo! through their public API, another set provided by Dun & Bradstreet (D&B) [30], a consultancy company that specializes in commercial information and insight for businesses, and a third one from InfoUSA[15] provided by the Harvard Center for Geographic Analysis (ESRI Business Analyst Data). In the first data set (from Yahoo!), the database is essentially built from user contributions. In the other two the data acquisition process is semi-automatic and involves integration of official and corporate databases, statistical analysis and manual evaluation [30]. The POIs from D&B and InfoUSA have a NAICS code assigned (2007 version), but the ones from Yahoo! do not. However, each POI from Yahoo! is assigned, in average, roughly two categories from the Yahoo! category taxonomy.

We have 156364 POIs from Yahoo!, 29402 from D&B and 196612 from InfoUSA for the area of Boston, Massachusetts. We also used 331118 POIs from Yahoo! and 16852 from D&B for the New York city area to see how our previously trained model would perform in a different city. We estimate that the Yahoo's categories taxonomy has more than 1300 distinct categories distributed along a 3-level hierarchy.

Given its nature, the growth of the Yahoo! database (or any other user content platform) is considerably faster than D&B and InfoUSA, and the POI categorization follows less strict guidelines, which in some cases may become subjective. Our hypothesis is that there is considerable coherence between Yahoo categories and NAICS codes, such that a model can be learned that automatically classifies incoming Yahoo! POIs.

There is, however, a major hurdle that needs to be overcome: the same POI (name, address, latitude, longitude) often does not have the same representation in both databases. This demands for a careful *POI Matching* operation.

3.3.3.2 POI Matching and Data Preparation

As mentioned before, our matching algorithm compares POIs according to their name, Web Site and distance. It makes use of the JaroWinklerTF-IDF class from the SecondString project [31] to identify close names, ignoring misspelling errors and some abbreviations (see section 3.2 for more details).

After matching Yahoo! POIs to D&B and InfoUSA, we build two different databases, where each POI contains a set of categories from Yahoo! and a NAICS classification

[15] www.infousa.com

[h!]

	Dataset A	Dataset B
NAICS source	Manta	InfoUSA
Total POIs	7289	44634
Distinct NAICS	504	689
Distinct categories	802	1109
Distinct category combinations	569	1002
Category combinations that appear only once	136	92
Categories that appear only once	181	107
NAICS that appear only once	115	96

TABLE 3.1: Some statistics of datasets A and B for Boston

provided by D&B and InfoUSA respectively. We started by using only the D&B database to retrieve the NAICS code for the Yahoo! POIs. However, we later realized that a larger set could provide a different perspective on the results. By using the POIs from InfoUSA for Boston we were able to get a dataset six times larger than the initial one, mostly because the InfoUSA has a better coverage of the Boston Metropolitan Area than D&B does (at least according to the data in our database). From this point on, we shall refer to the initial dataset, which results from POI matches between Yahoo! and D&B, as dataset A, and to the dataset resultant from the POI matching between Yahoo! and InfoUSA as dataset B. Table 3.1 shows some statistic details of both datasets used.

The dataset A contains 7289 POIs for Boston and Cambridge and 2415 for New York. In comparison with the original databases, these are much smaller sets due to a very conservative matching approach (string similarity of at least 80%, max distance of 80 meters). However the POI quantities are high enough to build statistically valid models. We performed a detailed analysis of this data and identified 569 different category combinations which included only 802 distinct categories from the full set (of over 1300). From D&B, our data covers 504 distinct six-digit NAICS codes. However, the 2007 NAICS taxonomy has a total of 1175 six-level categories, meaning that our sample data only covers some of the most common NAICS codes, which only represents about 43% of the total number of NAICS categories.

Figure 3.4 shows the distribution of POIs along the different NAICS codes for dataset A. As we can see in the chart, the distribution is far from being uniform.

Further analysis on the coherence between NAICS and Yahoo! shows that only in 80,2% of the POIs in dataset A the correspondent NAICS was consistent with the most common one for that given set of categories, which means that about one fifth of the POIs are incoherent with the rest of the sample. For different NAICS levels, particularly for two-digit and four-digit NAICS, the same analysis showed, as expected,

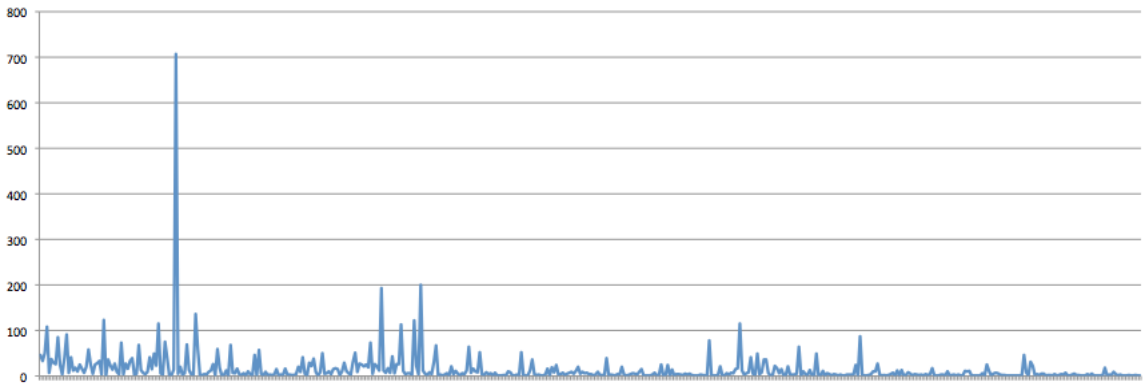

FIGURE 3.4: Distribution of the POIs in dataset A along the different NAICS code

NAICS code	Description	Occurrences
423730	Warm Air Heating and Air-Conditioning Equipment and Supplies Merchant Wholesalers	707
446130	Optical Goods Stores	200
314999	All Other Miscellaneous Textile Product Mills	193
493120	Refrigerated Warehousing and Storage	136
332997	Industrial Pattern Manufacturing	123

TABLE 3.2: Most common NAICS in the dataset A

Yahoo! category	Occurrences
Salons	157
All Law Firms	129
Government	116
Trade Organizations	115
Architecture	86

TABLE 3.3: Most common Yahoo! categories in the dataset A

a higher level of coherency. For the two and four-digit NAICS, 87,1% and 83,4% of the POIs, respectively. Therefore, by having the same set of Yahoo! categories mapping to different NAICS codes in different occasions, it is not expectable that we obtain a perfect model that classifies correctly all test cases. In order to understand the impact of these inconsistencies in the results, we also modified the POI dataset so that the NAICS code of a given POI would match the NAICS codes of the other POIs with the same category set, assigning to each POI the most common NAICS code for that given category set in the dataset. The results of this experiment are also presented in section 5.4.

Tables 3.2 and 3.3 show, respectively, the five most common NAICS and Yahoo! categories we identified in dataset A.

Regarding dataset B, we identified 689 distinct NAICS codes and 1109 distinct categories of the more than 1300 that we estimate Yahoo! has. The latter are much higher than the ones from dataset A (only 802) and therefore provides a better coverage of the source taxonomy. Then number of distinct category combinations almost doubled when compared to dataset A, which leads to more diversity in the training data and hopefully more accurate classifiers.

Figure 3.5 shows the distribution of POIs along the different NAICS codes for dataset B. Similarly to the distribution for dataset A, it is a irregular distribution.

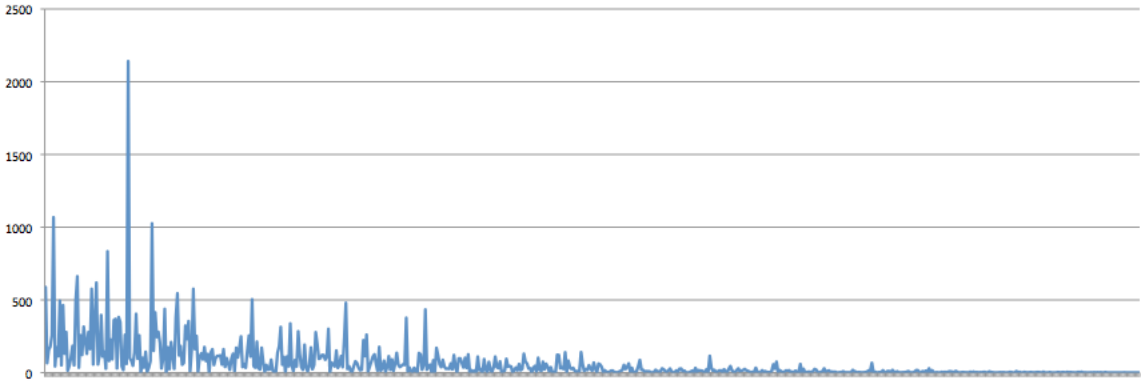

FIGURE 3.5: Distribution of the POIs in dataset B along the different NAICS code

Another possibility to generate a training set would be to manually classify a small set of Yahoo! POIs to the NAICS. Even though this would be a terribly painful and time-consuming task, it might generate a more consistent training set, since the NAICS classifications provided by D&B and InfoUSA result from the contribution of multiple users/sources, which makes them somehow subjective (remember that the NAICS codes of businesses are not always trivial to identify). However, we are interested in automating the classification process as much as possible, therefore we opted for the previously described approach through POI Matching. Also, manually producing training sets with the dimensions of the ones we use would not be feasible.

3.3.3.3 Flat Classification

The "flat classification" task corresponds to directly assigning a NAICS code to a POI given its set of Yahoo! categories. It is "flat" because the inherent hierarchy of NAICS is not taken into account in the classification model. Each NAICS code is simply seen as an isolated string "tag" that is assigned to a POI.

We experimented various machine learning algorithms for this particular classification task. Table 3.4 provides a brief description of the algorithms we tested. It is not the

Implementation	Description
DecisionStump	Does regression (based on mean-squared error) or classification (based on entropy).
FT	Classifier for building 'Functional trees', which are classification trees that could have logistic regression functions at the inner nodes and/or leaves.
ID3	Unpruned decision tree based on the ID3 algorithm.
J48	Pruned or unpruned C4.5 decision tree.
J48graft	Grafted (pruned or unpruned) C4.5 decision tree.
RandomForest	Forest of random trees.
RandomTree	Tree that considers K randomly chosen attributes at each node. Performs no pruning. Also has an option to allow estimation of class probabilities based on a hold-out set (backfitting).
DecisionTable	Simple decision table majority classifier.
JRip	Propositional rule learner, Repeated Incremental Pruning to Produce Error Reduction (RIPPER), which was proposed by William W. Cohen as an optimized version of IREP.
IBk	K-nearest neighbors classifier. Can select appropriate value of K based on cross-validation. Can also do distance weighting.
IB1	1 - nearest-neighbor classifier. Simplification of IBk.
K*	K* is an instance-based classifier, that is the class of a test instance is based upon the class of those training instances similar to it. It differs from other instance-based learners in that it uses an entropy-based distance function.
BayesNet	Bayesian Network
NaiveBayes	Naive Bayes model
MultilayerPerceptron	A classifier that uses backpropagation to classify instances.
ZeroR	Predicts the mean (for a numeric class) or the mode (for a nominal class).
OneR	Uses the minimum-error attribute for prediction, discretizing numeric attributes.

TABLE 3.4: Brief description of each Weka algorithm tested

scope of this paper to describe any of the algorithms in detail. The interested reader is redirected to dedicated literature ([34, 35]).

In our experiments we built classifiers for different NAICS levels (i.e. NAICS categories with different granularities), particularly two, four and six-digit NAICS codes. This choice is typical in Urban Planning depending on the study at hand (e.g. level 2 allows to analyze economic sectors, while level 6 goes to the level of the establishment specificities).

POI Name	Yahoo Categories	Tags
Willett Institute of Finance	Financial Planning, Investment Services	income, shares, Security Analysis, plan, Cash
W3 Edge LLC	Computer Communications, Web Services	network, software system, devices, servers, wireless technologies
Curis Incorporated	Doctors & Clinics, Laboratories, Medical Laboratories	hospital, General practice, chemistry, clinic, Clinical laboratory

TABLE 3.5: Some tags produced by Kusco.

3.3.3.4 Extension with Semantic Annotations

Describing POIs only by one concept or two (their categories) may be not sufficiently diverse to help data mining algorithms to classify more precisely. We propose the use of semantically enriched [36] information about places to incorporate more attributes in order to refine POI details. This enrichment is made by the KUSCO system, which basically consists of gathering textual descriptions available on the Wikipedia and applying Text Mining techniques on them, such as Part-Of-Speech tagging, Noun-Phrase Chunking and Named Entity Recognition. As one POI may belong to more than one category, all its categories descriptions are processed. By computing TF_IDF, the most relevant tags become the index of a place. These tags are also semantically contextualized in WordNet [37] in order to aggregate synonym tags in just one entry. Table 3.5 shows some examples of POI indexes produced.

A sample of 150 Tag Indexes about Boston POIs were manually validated by 22 volunteers who know the city in study, answering the question, for each word, whether it is related to the POI or not. We obtained a precision of 56% ($\sigma = 0.2$) considering all unanswered tags as invalid. In some cases even the volunteers disagree, reflecting the subjective nature of this information. To further validate this data for our purposes, we analyzed whether words clustered around categories and the accuracy reaches 97% (using the K-Means algorithm to cluster and classify).

These new tags were associated to each POI and the process of section3.3.3.3 was repeated.

Please note that all the POI semantic enrichment work is part of Ana Alves [36] PhD thesis. It is described here only to help the reader understand the source of the semantic tags and thus help her comprehend the obtained results.

3.3.3.5 Hierarchical Classification

In this approach we take advantage of the hierarchical structure of the NAICS and build a hierarchy of classifiers. In this hierarchy each classifier decides what classifier to use next, narrowing down the NAICS code possibilities on each step, until a final 6-digit code (or 4-digit code, depending on the goal) is achieved. Figure 3.6 depicts one possible hierarchy.

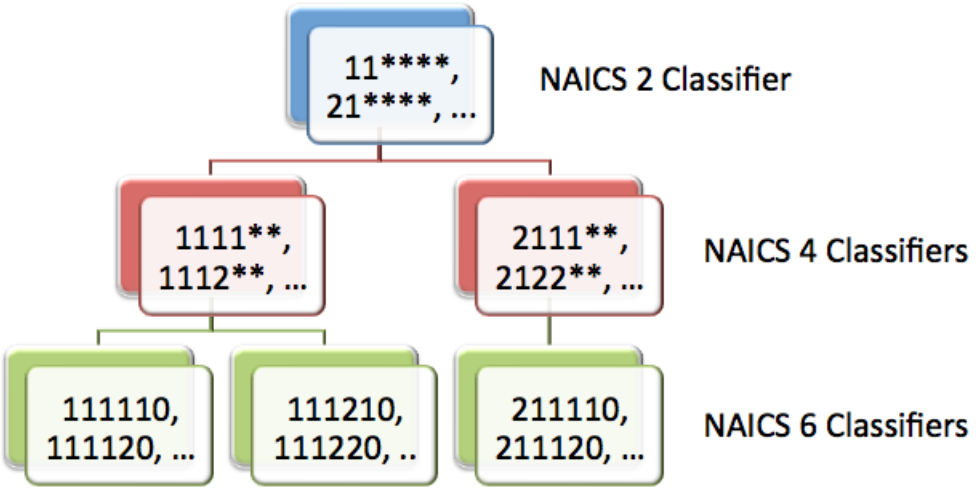

FIGURE 3.6: A possible hierarchy of classifiers

By looking at this hierarchy, we can see that it has 3 levels (2, 4 and 6-digit NAICS). The first level always consists of a single classifier that decides which NAICS sector (2-digit code) the POI belongs to. Taking the sector into account, the algorithm then decides which level 2 classifier to use next. After that, the same process repeats itself for all the levels until a leaf node is achieved in the tree structure of the hierarchy. To provide an example consider a POI that has the following NAICS code: 111110. According to figure 3.6 the top-level classifier will decide that it belong to sector 11 ("Agriculture, Forestry, Fishing and Hunting") and the left-most level 2 classifier will be used next. Then, this classifier will determine that the 4-digit NAICS code of the POI is 1111 ("Oilseed and Grain Farming") and, based on this decision, the left-most classifier in the third level of the figure will be used, and will supposedly classify the POI with the NAICS code 111110 ("Soybean Farming"). Of course along the top-down course a mistake can be made by one classifier. In this case, the error would propagate downwards and there would be no way to recover from it, and hence the final NAICS code would be wrong.

Our hypotheses is that by using a hierarchy of classifiers, each classifier will have less possible outputs and hence the classification scheme can be less complex and more accurate. If we consider, for example, the ID3 algorithm, the entropy values for the

different features will be computed according to a smaller class subset, and therefore the selection of the next feature to use (which is based on the entropy calculation) will be different and the resulting tree will also be different. Hopefully, the generated classifier can be more suited to that particular classification (like deciding for a POI that belongs to NAICS sector 53 if it belongs to the subcategory 531, 532, etc).

In our experiments we use four different hierarchies of classifiers, two with 2 levels:

- NAICS 2 and NAICS 4
- NAICS 2 and NAICS 6

and other two with 3 levels:

- NAICS 2, NAICS 3 and NAICS 4
- NAICS 2, NAICS 4 and NAICS 6

In order to implement the hierarchical classifiers we took advantage of the open-source feature of Weka and its very well documented Java API, and developed our own Java code using the already implemented algorithms and classes that Weka provides.

As one would expect the computational complexity of the classifiers increased many times, depending on the base machine learning algorithm used. As a result, it was not possible for us to test some of the more computational intensive algorithms. However, like we did for the flat classification, we tried to test different types of machine learning algorithms like: bayesian networks, tree-based learners, instance-based learners, rule-based learners. Neural networks were not possible to test due to their computational demands, both in processing power and memory.

3.4 POI generation

By using multiple sources of geo-referenced information, we want to be able to determine places of interest that reflect the current interests of the population. There are certain challenges regarding this, like temporality constrains. The interest that a place raises in a person varies along the time, according to many factors, which can be economical, social, cultural or even political. Also, naming a place accordingly is another great challenge. It isn't that hard to determine places that generate a lot of "buzz", the problem is to find the ones that are really relevant, properly name them and distinguish them from other neighboring places of interest.

Regarding the sources of information, Flickr is an obvious choice due to its dimension and hence we used it in our research. However, it would be interesting to find other sources that are not restricted to photos, and also include news, blogs and twits. Flickr supplies a map interface through which users can drag their photos to the map locations where the photos were taken. In addition, many photos are accurately geo-tagged using GPS logs or location-aware devices.

Once we decided to use Flickr as our data source we started gathering as much information as possible from it. Since most of data is for the Boston Metropolitan Area we decided to also focus this research in that area. The extraction phase took almost two months, and allowed us to collect a total of 255212 photos and 75986 distinct tags. Each photo has an average of 7 tags associated with it.

Using the data we collected from Flickr, we applied two different clustering algorithms: K-means and DBScan. K-means clustering is a well known method of cluster analysis which aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, while DBScan is a density-based algorithm for discovering clusters in large spatial databases with noise. The latter, in contrast with the former, has the advantage of not requiring the user to explicitly define the number of clusters we want the algorithm to generate. Instead, it requires us to define the epsilon (ϵ) and the minimum number of points required to form a cluster (minPts). It starts with an arbitrary starting point that has not been visited. The ϵ-neighborhood for this point is retrieved, and if it contains sufficiently many points, a cluster is started. Otherwise, the point is labeled as noise. Note that this point might later be found in a sufficiently sized ϵ-environment of a different point and hence be made part of a cluster. If a point is found to be part of a cluster, its ϵ-neighborhood is also part of that cluster. Hence, all points that are found within the ϵ-neighborhood are added, as is their own ϵ-neighborhood. This process continues until the cluster is completely found. Then, a new unvisited point is retrieved and processed, leading to the discovery of a further cluster or noise.

For both algorithms we used different parameter combinations in order to find the one that best suits our goals.

Each of the centroids of the identified clusters was then considered to be a place of interest. Since the pairs latitude/longitude are pretty much meaningless by themselves, we tried to identify the tags that are more representative of the place/cluster using TF-IDF (term frequency, inverse document frequency), i.e., tags that appear frequently inside a cluster but infrequently elsewhere. TF-IDF is often used in information retrieval and text mining. The TF-IDF weight is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The importance increases

proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus. We adapted the standard TF-IDF in the following way: a document is represented by a cluster and the terms correspond to the tags. Therefore the TF-IDF can be computed as

$$tf_{t,c} = \frac{n_{t,c}}{\sum_{k \in T} n_{k,c}} \tag{3.1}$$

$$idf_t = \log \frac{|C|}{|\{c : c \in C \wedge t \in c\}|} \tag{3.2}$$

$$tfidf_{t,c} = tf_{t,c} * idf_t \tag{3.3}$$

where t is a tag and c a cluster and therefore:

- $n_{t,c}$ denotes the number of occurrences of tag t in cluster c
- $|C|$ is the total number of clusters and
- $|\{c : c \in C \wedge t \in c\}|$ is number of documents where the tag t appears

In order to further improve the TF-IDF results we start by applying some basic filters of tags we know beforehand that we do not want to consider. With these filter we remove:

- tags with less than 3 consecutive characters
- tags that match the regular expression: 'geo:lat=[0-9. -]*'
- tags that match the regular expression: 'geo:long?=[0-9. -]*'
- tags that match the regular expression: 'meta:exif='

Since the tags associated with each photo on Flickr result from user typing them in, it is fairly common to find errors, misspells and different ways writing names of places like "Boston_Common" and "BostonCommon". The fact that tags in some system can have spaces while in others do not, further potentiates the risk of finding different ways of referring to the same place. Hence, it might not be the best approach to use the standard TF-IDF measurement. Therefore, we made some changes to the TF-IDF computation in order to take tag similarity into account. Our similarity-based TF-IDF can be calculated as follows:

$$tf_{t,c} = \frac{\sum_{k \in S_t} (n_{k,c} * Sim_{k,t})}{\sum_{k \in T} n_{k,c}} \tag{3.4}$$

$$idf_t = \log \frac{|C|}{\sum_{k \in S_t} (|\{c : c \in C \wedge k \in c\}| * Sim_{k,t})} \tag{3.5}$$

$$tfidf_{t,c} = tf_{t,c} * idf_t \tag{3.6}$$

where $Sim_{k,t}$ denotes the similarity score between tags k and t according to the JaroWinklerTF-IDF algorithm from the SecondString project also mentioned in section 3.2, and S_t is the set of tags that are similar to tag t (i.e. similarity score is above a given threshold).

Making these changes to the TF-IDF computation significantly increased the CPU, and hence the time the algorithm took to run increased immensely. In order to cope with that increase we parallelized the TF-IDF algorithm to benefit from the multiple cores the CPU's of our machines had. To further reduce the computational time, we calculated all the similarity scores between tags only once and saved them in memory so that we didn't have re-calculate them each time we needed them. To avoid an excessive memory usage, we made an optimization which consisted in pre-computing all tags that are similar to each tag, maintaining only that information in memory instead of all similarity scores between all tags.

Using our modified version of TF-IDF, we display for each cluster the top-5 tags with higher TF-IDF scores. We opted to display 5 tags instead of a single one mainly because, most of the times, a single tag is not good enough to unambiguously represent a place. Instead, by showing 5 tags we can help the user to better contextualize the place and know further aspects about it.

Chapter 4

Validation

In this chapter we describe how we perform validation of the different approaches developed.

Validating the quality of our approaches is a complicated issue. Even though the validation of the POI classification is relatively simple to do, validating the POI matching or the POI generation algorithms is not so easy, specially in an automated way. Due to their nature, these approaches rely only on human evaluation, making the validation process a lot more time consuming and less reliable. Therefore, whenever possible, we try to use an automatic validation scheme.

4.1 POI matching

In order to validate our POI matching approach, we made an analysis of the precision and the recall of our algorithm. To measure the precision, we generated 5 csv files with 200 random mappings between POIs from Yahoo and Manta, and distributed them between five volunteers that belong our working group. Each of those 200 entries in the csv files contained almost all the information one needed to check if the two POIs were the same. This information included: names, categories, distance between the two POIs and the correspondent websites, so that, in a last case scenario, one could navigate to the websites for confirmation.

Regarding the recall estimation, we also generated 5 csv files, but this time, with 50 random entries each, representing POIs that were not considered matches but that were very close (the distance was a bit off, or the name similarity was a little below the defined threshold).

The results we obtained were very good. We obtained a precision of 98% and a estimated high value for recall.

4.2 POI classification

A NAICS code is assigned to a business according to the activity that generates more revenue. Therefore, sometimes is not obvious, even for an experienced person, to find out what the NAICS of a business is. Besides, there is no central government agency with the role of assigning, monitoring, or approving NAICS codes for establishments. Individual establishments are assigned NAICS codes by various agencies for various purposes using a variety of methods. Due to this ambiguity, some websites often assign more than one NAICS to a business. Even worse, these websites most of the times rely on user contribution, thus the NAICS they show might not be correct at all. All this makes the validation of our approach quite difficult.

The easiest way to perform validation is to use a set POIs that we already know the NAICS code through other sources (like from D&B or InfoUSA using POI Matching). This way, for the Wordnet and the Ontology Matching approaches, we can take the POIs from that set, classify them with NAICS using our approaches and then compare the results with NAICS provided by D&B or InfoUSA for that POI. Similarly, for the machine learning approaches we use ten-fold cross-validation, a technique for estimating the performance of a predictive model. In ten-fold cross-validation, the original sample is randomly partitioned into ten subsamples. Of the ten subsamples, a single subsample is retained as the validation data for testing the model, and the remaining nine subsamples are used as training data. The cross-validation process is then repeated ten times (the folds), with each of the ten subsamples used exactly once as the validation data. The ten results from the folds are then averaged to produce a single estimation. The advantage of this method over repeated random sub-sampling is that all observations are used for both training and validation, and each observation is used for validation exactly once.

For the hierarchical approaches we also perform ten-fold cross-validation, but the data division for training/testing is slightly more complicated than for standard flat classification. Like in "normal" ten-fold cross-validation, we also start by leaving 10% of the data out for test and use the remaining 90% for training, repeating this process ten times. However, each classifier in a given level only receives the part of those 90% of training data that respects him. For instance, a level two classifier for deciding which sub-category of NAICS sector 53 a given POI belongs to would only be trained with POIs that belong to that NAICS sector. Hence, the only classifier that receives all the training data (90%) would be the top-level classifier (i.e. the one that decides which

NAICS sector a POI belong to). After the training phase, the hierarchy is tested with the 10% of the data left out. This process is repeated ten times, and the average accuracy over the ten iterations is determined.

It is however possible that the NAICS determined by our approach is better, i.e. makes more sense, at least from the user point of view, than the one provided by D&B or InfoUSA. If we think about the machine learning approach, two POIs from Yahoo that present the same categories will be classified to the same NAICS, which makes sense. However, that NAICS may not match the one from D&B or InfoUSA. In fact, if from the training process resulted that most of the POIs from Yahoo! with a given set of categories have a certain NAICS, why shouldn't this one? Of course this raises, once again, the question of the consistency and quality of the POI data. If, for instance, the Yahoo! POIs are not correctly categorized, then this will not apply.

4.3 POI generation

Once again, in order to validate the generated POIs we rely essentially on human validation. However, in this case, this kind of validation is especially difficult to do, because it requires some knowledge from the test subjects about the study area, in order to be able to recognize the places and do a proper confirmation that a place is in fact a point of interest or not.

Due to these kinds of issues, it was not possible for us to perform validation of these approaches. Instead, we present some results obtained and study cases that help us understands the strengths and the weaknesses of this approach.

Chapter 5

Results

5.1 POI extraction

In order to be able to see the POIs in a map along with their details and actually interact with them by clicking on them, we developed a website using the Google Maps API and a REST Web service that serves as an interface to our database. A complete list of the methods made available by the REST Web service, along with the correspondent description can be seen in Appendix C.The developed website makes use of MarkerClusterer[1], a JavaScript library developed by Xiaoxi Wu and altered by us in order to meet our needs. MarkerClusterer allows us to group the markers displayed in a map in clusters, so that it is possible to view a large number of POIs simultaneously in a map. Figure 5.1 shows a screenshot of this website. However, the reader can take a look at the website by himself and interact with it in `http://greenhomes.dei.uc.pt:8080/Maps/index.html`.

We also developed a simple Android application to visualize the POIs of our database, that allows the users to select the POI source and the NAICS industry sector he is interested in. Figure 5.2 shows three screenshots of this application.

Currently our POI database has a total of 981956 POIs, mainly from the areas of Boston, New York, San Francisco and Lisbon. Table 5.1 shows the number of POIs according to the area and the POI source.

We made a simple analysis of the coverage of the different POI sources by representing all the POIs for the a given area in a map, using mapping tools like qGIS and uDig. Figures 5.3 and 5.4 show maps of the POIs for the Boston Metropolitan Area from InfoUSA and Yahoo respectively.

[1] http://gmaps-utility-library.googlecode.com/svn/trunk/markerclusterer/1.0/docs/reference.html

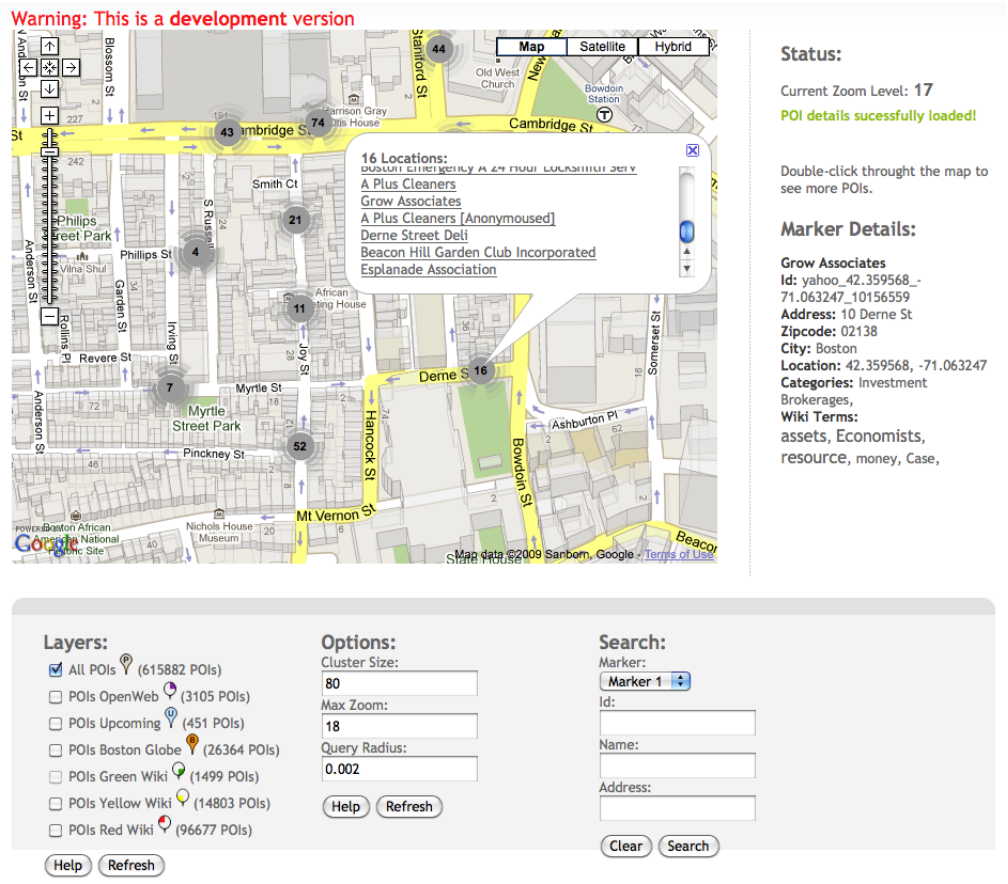

FIGURE 5.1: Screenshot of the visualization platform

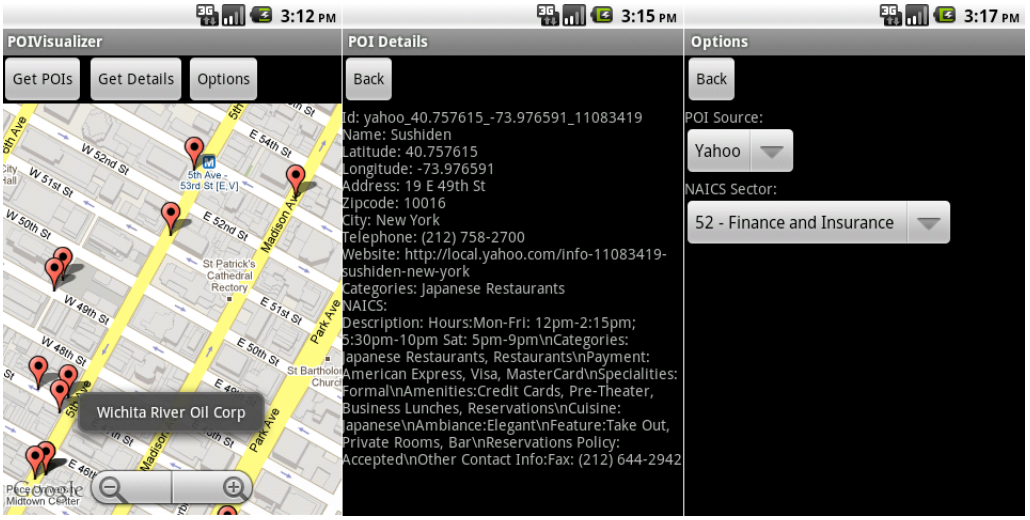

FIGURE 5.2: Screenshots of the Android visualization application

5.2 POI analysis

In order for us to further understand some aspects of the collected POI data, we made a few analyses that we will talk about in this section. It is important to note that is

	New York	Boston	S. Francisco	Lisbon	Total
InfoUSA	-	196612	-	-	196612
Yelp	5179	10900	0	-	16498
Yellow Pages	7694	13302	0	-	21774
Upcoming	96	1000	13	-	2098
City Search	377	7070	0	-	8367
Yahoo!	183147	156364	89331	-	433375
Manta	16496	29402	0	-	46188
Sapo	-	-	-	37465	37465
Pág.Amarelas	-	-	-	153737	153737
Total	374962	417517	181546	191202	981956

TABLE 5.1: Number of POIs according to the area and the POI source

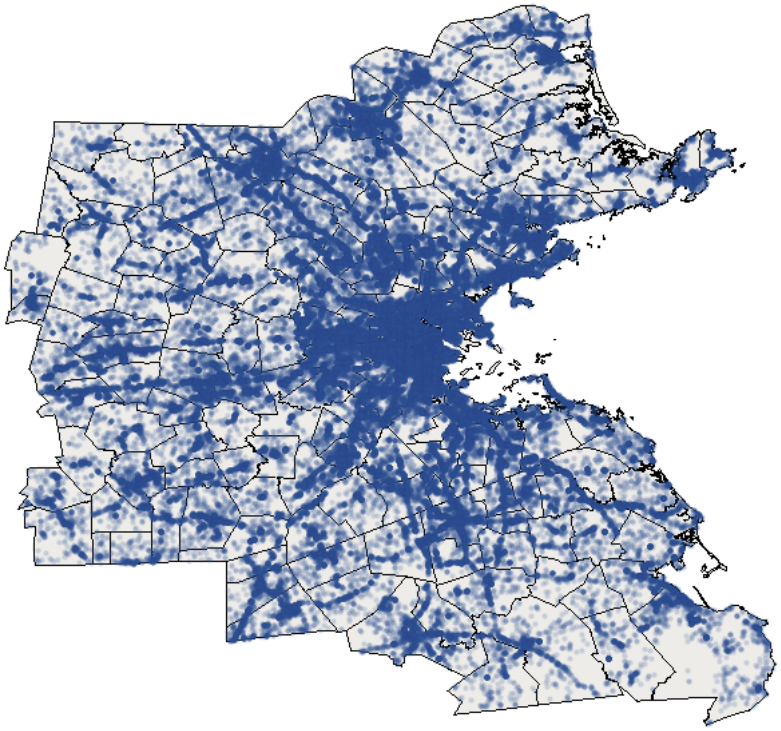

FIGURE 5.3: POIs from InfoUSA for the Boston Metropolitan Area

not our purpose to make a exhaustive analysis of data. Instead, we are interested in displaying some interesting analyses than can be performed using classified POI data.

The analysis performed is based on the POI data from Manta and from InfoUSA for the Boston Metropolitan Area. We chose this data because it is classified with NAICS codes. Using these codes, we defined vectors to represent areas using the ID of that area and the number of POIs for each NAICS sector. In order for the analysis to be fair, we normalized the data by dividing the number of POIs for the size of the area, hence each vector has the following format: [area ID, density of the POIs from NAICS sector 11, density of the POIs from NAICS sector 21, density of the POIs from NAICS sector

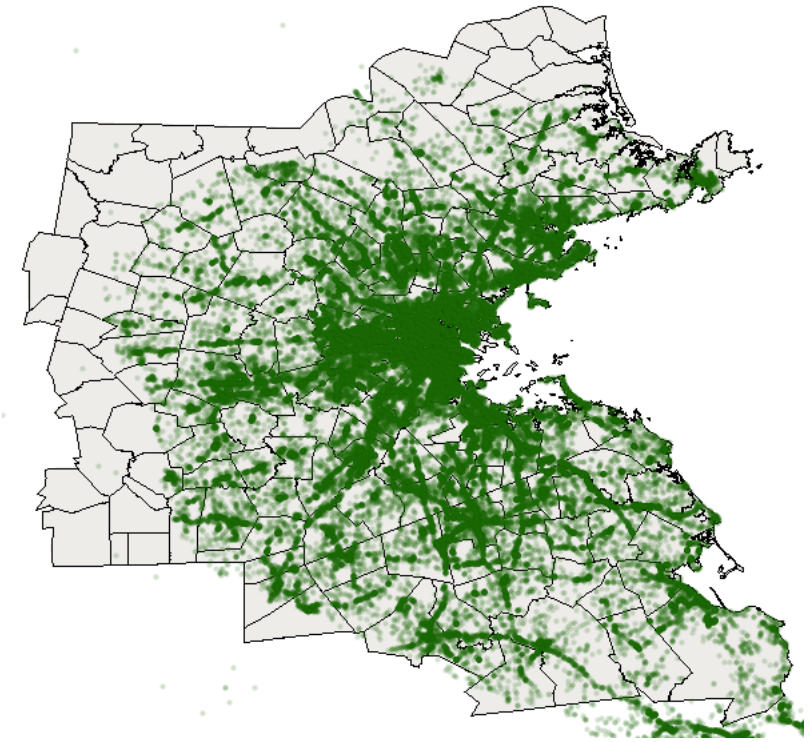

FIGURE 5.4: POIs from Yahoo for the Boston Metropolitan Area

22, ...], where the density is expressed in number of POIs by area unit. By grouping the POIs in area and by NAICS sectors, we were able to compare different areas of the Boston Metropolitan Area.

We started by comparing the two POI sources. Figure 5.5 shows a chart with the distribution of the POIs for the different NAICS sector for both POI sources. As we can see in this chart, the sources are quite different, being the InfoUSA the most complete one for almost all of the sectors.

Similarly to that analysis, we compared five different towns according to their POI densities distributions along the different NAICS sectors. Figure 5.6 shows the results obtained. These results, like all the remaining, are only based on POI data from InfoUSA.

As we can see in figure 5.6 the POI densities and distribution are quite different from town to town. For instance, Winthrop, which is a small town in surroundings of Boston, has much smaller POI densities than Boston and Cambridge, which are towns with high populations.

Inspired by the results in figure 5.6 we made a clustering analysis of the vectors defined using Expectation Maximization (EM) algorithm. This algorithm is implemented by Weka in a way that it does not require us to define the number of clusters we want to

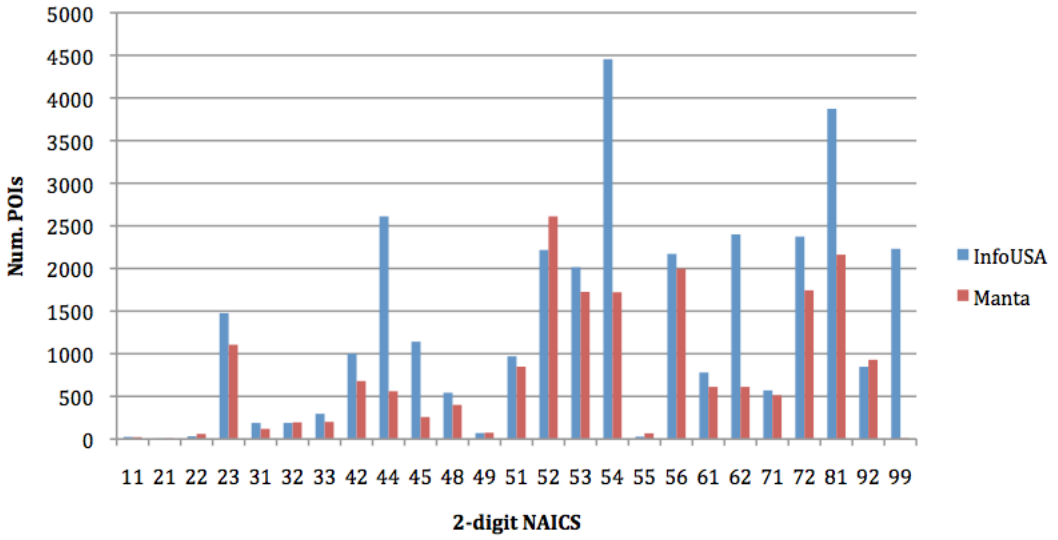

FIGURE 5.5: Distribution of the POIs for the different NAICS sector

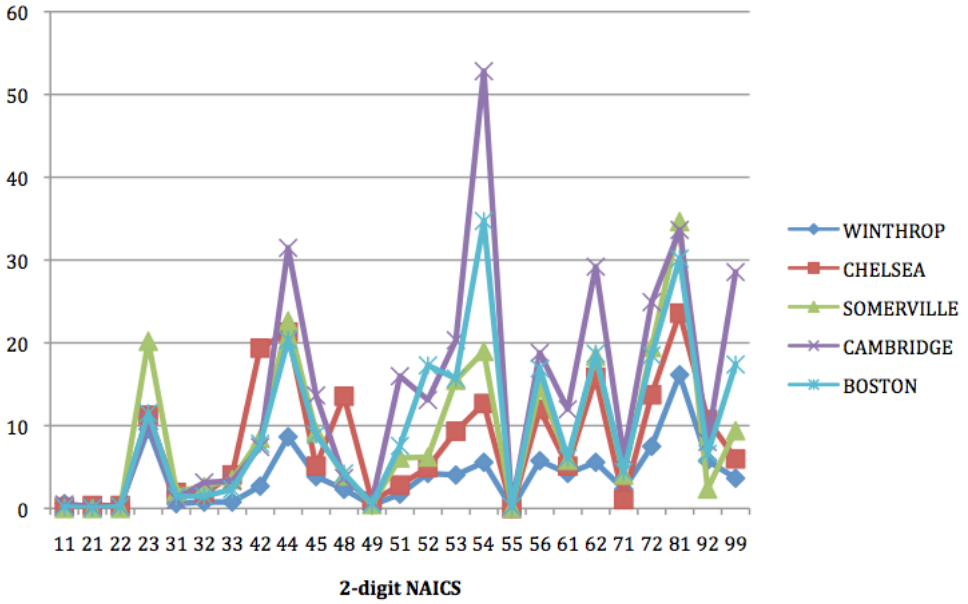

FIGURE 5.6: Comparison between five different towns according to their POI distributions along the different NAICS sectors

generate as parameter. Instead, it uses cross-validation for determine that number. The process follows the following steps:

1. the number of clusters is set to 1
2. the training set is split randomly into 10 folds.
3. EM is performed 10 times using the 10 folds the usual CV way.
4. the loglikelihood is averaged over all 10 results.

5. if loglikelihood has increased the number of clusters is increased by 1 and the algorithm continues at step 2.

The results obtained can be seen in figures 5.7 (cluster assignments) and 5.8 (cluster centroids).

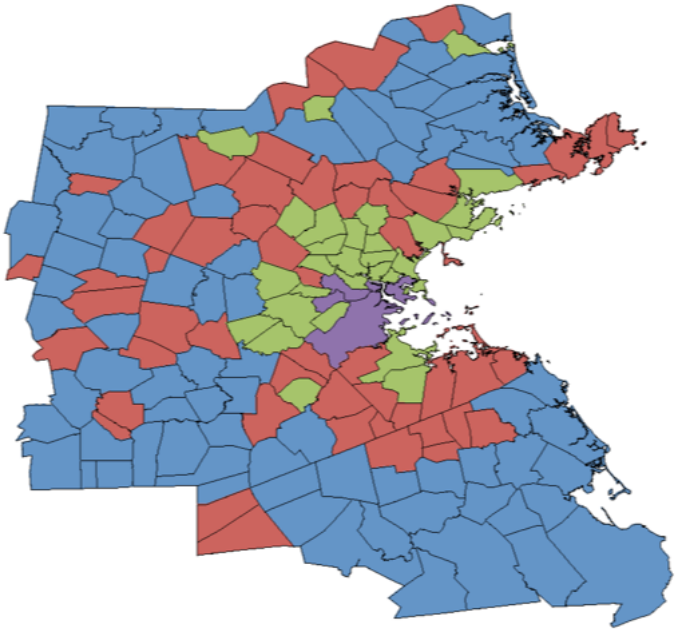

FIGURE 5.7: EM cluster assignments

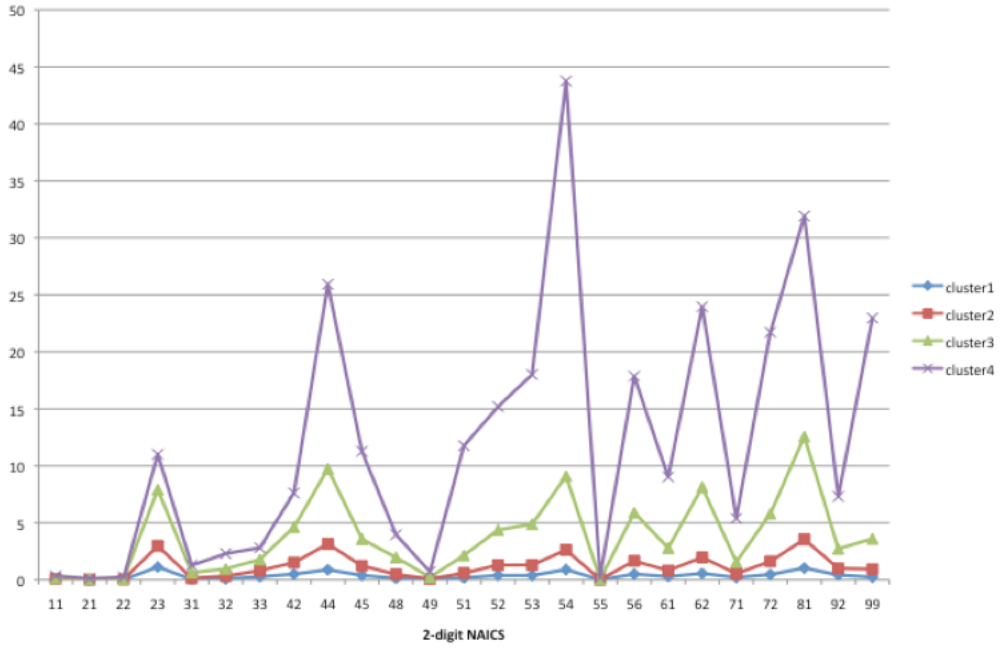

FIGURE 5.8: EM cluster centroids

Based in the results from the clustering, many conclusions can be drawn. For example, we can see that there are two towns (Boston and Cambridge) that distance

themselves from the rest of the towns and are somehow similar to each other. We can also see that the centroid of the cluster they belong to (see figure 5.8, in purple) represents areas of high POI densities.

We also investigated possible correlations between the different NAICS sectors. The correlation matrix is presented in figure 5.9. It is important to note that this analysis was made at the block group level and not at the town level like all the others.

	11	21	22	23	31	32	33	42	44	45	48	49	51	52	53	54	55	56	61	62	71	72	81	92	99
11	1	0.08	0.11	-0.1	0.09	0.08	0.11	0.02	-0.1	0.03	0.07	0.13	0.1	0.03	-0.1	-0	0.06	-0.1	-0	-0	0.08	-0	-0.1	0.08	0
21	0.08	1	0.06	0.06	0.04	0.07	0.09	0.09	0.06	0.07	0.1	0.06	0.08	0.08	0.03	0.05	0.04	0.06	0.03	0.03	0.06	0.04	0.03	0.05	0.07
22	0.11	0.06	1	0.02	0.06	0.13	0.14	0.08	0.05	0.1	0.09	0.13	0.1	0.11	0.05	0.06	0.11	0.03	0.07	0.05	0.1	0.07	0.01	0.2	0.05
23	-0.1	0.06	0.02	1	0.18	0.2	0.25	0.4	0.46	0.35	0.26	0.09	0.25	0.34	0.39	0.41	0.06	0.43	0.28	0.36	0.23	0.39	0.51	0.2	0.31
31	0.09	0.04	0.06	0.18	1	0.22	0.26	0.27	0.27	0.25	0.17	0.17	0.23	0.22	0.22	0.24	0.09	0.21	0.16	0.2	0.19	0.26	0.23	0.18	0.2
32	0.08	0.07	0.13	0.2	0.22	1	0.38	0.3	0.25	0.28	0.26	0.23	0.26	0.26	0.24	0.24	0.13	0.21	0.19	0.18	0.23	0.24	0.19	0.22	0.22
33	0.11	0.09	0.14	0.25	0.26	0.38	1	0.36	0.27	0.28	0.27	0.21	0.27	0.26	0.21	0.25	0.08	0.2	0.18	0.17	0.24	0.24	0.19	0.21	0.22
42	0.02	0.09	0.08	0.4	0.27	0.3	0.36	1	0.47	0.39	0.29	0.18	0.33	0.4	0.36	0.39	0.12	0.39	0.28	0.33	0.27	0.41	0.41	0.22	0.35
44	-0.1	0.06	0.05	0.46	0.27	0.25	0.27	0.47	1	0.54	0.27	0.14	0.37	0.51	0.57	0.5	0.1	0.57	0.36	0.51	0.31	0.67	0.7	0.23	0.44
45	0.03	0.07	0.1	0.35	0.25	0.28	0.28	0.39	0.54	1	0.25	0.17	0.41	0.47	0.44	0.45	0.11	0.45	0.34	0.42	0.33	0.54	0.52	0.28	0.38
48	0.07	0.1	0.09	0.26	0.17	0.26	0.27	0.29	0.27	0.25	1	0.18	0.23	0.24	0.22	0.23	0.08	0.26	0.17	0.19	0.23	0.26	0.24	0.22	0.22
49	0.13	0.06	0.13	0.09	0.17	0.23	0.21	0.18	0.14	0.17	0.18	1	0.19	0.21	0.12	0.15	0.11	0.12	0.13	0.11	0.18	0.16	0.09	0.21	0.14
51	0.1	0.08	0.1	0.25	0.23	0.26	0.27	0.33	0.37	0.41	0.23	0.19	1	0.43	0.38	0.47	0.13	0.36	0.32	0.39	0.33	0.38	0.35	0.27	0.39
52	0.03	0.08	0.11	0.34	0.22	0.26	0.26	0.4	0.51	0.47	0.24	0.21	0.43	1	0.48	0.5	0.15	0.45	0.32	0.46	0.32	0.51	0.48	0.29	0.39
53	-0.1	0.03	0.05	0.39	0.22	0.24	0.21	0.36	0.57	0.44	0.22	0.12	0.38	0.48	1	0.53	0.1	0.53	0.36	0.51	0.3	0.54	0.59	0.23	0.46
54	-0	0.05	0.06	0.41	0.24	0.24	0.25	0.39	0.5	0.45	0.23	0.15	0.47	0.5	0.53	1	0.12	0.5	0.39	0.54	0.35	0.47	0.53	0.24	0.44
55	0.06	0.04	0.11	0.06	0.09	0.13	0.08	0.12	0.1	0.11	0.08	0.11	0.13	0.15	0.1	0.12	1	0.08	0.1	0.1	0.1	0.11	0.07	0.1	0.09
56	-0.1	0.06	0.03	0.43	0.21	0.21	0.2	0.39	0.57	0.45	0.26	0.12	0.36	0.45	0.53	0.5	0.08	1	0.34	0.46	0.28	0.53	0.59	0.21	0.43
61	-0	0.03	0.07	0.28	0.16	0.19	0.18	0.28	0.36	0.34	0.17	0.13	0.32	0.32	0.36	0.39	0.1	0.34	1	0.4	0.27	0.35	0.42	0.26	0.31
62	-0	0.03	0.05	0.36	0.2	0.18	0.17	0.33	0.51	0.42	0.19	0.11	0.39	0.46	0.51	0.54	0.1	0.46	0.4	1	0.29	0.49	0.58	0.25	0.41
71	0.08	0.06	0.1	0.23	0.19	0.23	0.24	0.27	0.31	0.33	0.23	0.18	0.33	0.32	0.3	0.35	0.1	0.28	0.27	0.29	1	0.32	0.25	0.28	0.29
72	-0	0.04	0.07	0.39	0.26	0.24	0.24	0.41	0.67	0.54	0.26	0.16	0.38	0.51	0.54	0.47	0.11	0.53	0.35	0.49	0.32	1	0.64	0.25	0.41
81	-0.1	0.03	0.01	0.51	0.23	0.19	0.19	0.41	0.7	0.52	0.24	0.09	0.35	0.48	0.59	0.53	0.07	0.59	0.42	0.58	0.25	0.64	1	0.21	0.46
92	0.08	0.05	0.2	0.2	0.18	0.22	0.21	0.22	0.23	0.28	0.22	0.21	0.27	0.29	0.23	0.24	0.1	0.21	0.26	0.25	0.28	0.25	0.21	1	0.2
99	0	0.07	0.05	0.31	0.2	0.22	0.22	0.35	0.44	0.38	0.22	0.14	0.39	0.39	0.46	0.44	0.09	0.43	0.31	0.41	0.29	0.41	0.46	0.2	1

FIGURE 5.9: Correlation matrix of the different NAICS sectors

By analyzing the correlation matrix, we can see that the strongest correlations occur between the following NAICS sectors:

- NAICS sector 44 (Retail Trade) and sector 81 (Other Services (except Public Administration)
- NAICS sector 44 (Retail Trade) and sector 72 (Accommodation and Food Services)
- NAICS sector 72 (Accommodation and Food Services) and sector 81 (Other Services (except Public Administration))

These correlations suggest that, for instance, when there is a high density of POI from sector 44 (Retail Trade) in a given block group, there also is a high density of POIs from sector 72 (Accommodation and Food Services), which somehow goes along with common sense.

Finally, we performed a Principal Component Analysis (PCA) using the same vectors described before. PCA involves a mathematical procedure that transforms a number of possibly correlated variables into a smaller number of uncorrelated variables called principal components. The eigenvalues and the proportions of the determined eigenvectors can be seen in figure 5.10. As we can see, only the first eigenvector accounts for almost

75% of the variability in the data, and eight eigenvectors account for more than 96% of the variability, which also points to a strong correlation.

eigenvector	eigenvalue	proportion	cumulative
V1	18.65212	0.74608	0.74608
V2	2.00849	0.08034	0.82642
V3	0.97009	0.0388	0.86523
V4	0.72766	0.02911	0.89433
V5	0.52938	0.02118	0.91551
V6	0.45871	0.01835	0.93386
V7	0.36126	0.01445	0.94831
V8	0.30783	0.01231	0.96062

FIGURE 5.10: Principal Component Analysis (PCA) results

5.3 POI matching

The following list shows five examples of POI matches that can help understand not only how the approach works, but also some of the issues we faced when developing it.

1. **Name1:** Copy Cop Llc
   Name2: Copy Cop
   Website1: http://www.copycop.com
   Website2: http://copycop.com/
   Categories1: Photocopying and duplicating services in Boston, Quick Printing
   Categories2: Commercial Printers, B2B Printing Facilities
   Name Similarity: 0.816
   Distance: <5 meters

2. **Name1:** American Plumbing & Heating
   Name2: American Plumbing & Heating
   Website1: -
   Website2: -
   Categories1: Plumbing Contractors in Boston, Plumbing & Hvac Contrs
   Categories2: Plumbing, B2B Contractors
   Name Similarity: 1.000
   Distance: >65 meters

3. **Name1:** Ultimate Parking Inc
   Name2: Ultimate Parking LLC
   Website1: http://www.ultimateparking.com
   Website2: http://ultimateparking.com/
   Categories1: Automobile parking in Boston, Automobile Parking, Parking Lots and Garages
   Categories2: Parking Services

Name Similarity: 0.667
Distance: <15 meters

4. **Name1:** Boston Public Schools
Name2: Boston Public School Admin Lib
Website1: http://www.boston.k12.ma.us
Website2:
Categories1: School custodian, contract basis in Boston, Maintenance Department For Schools, Janitorial Services
Categories2: Libraries
Name Similarity: 0.767
Distance: <15 meters

5. **Name1:** Summit Logistics Llc
Name2: Summit Logistics
Website1:
Website2:
Categories1: Silk screen design in Boston, Warehousing & Shipping Broker, Graphic Design Services
Categories2: Industrial Importers
Name Similarity: 0.816
Distance: <15 meters

By analyzing the results, we can see that matches 1 and 2 are correct even thought the distance between the two POIs in match number 2 is slightly high. Match number 3 is a good example where the use of the website URL allowed to find a match that would be discarded otherwise due to the low name similarity. Match number 4 is a example of a mismatch that was considered a match by the algorithm. One could argue that this mismatch could be discovered if we used the category information, because they are way different. However, doing so, could also discard matches like match number 5, where the names are similar and the distance is short but the categories are quite different or, at least, difficult to compare.

Using the thresholds defined in section 3.2, by manually validating a random subset of the POI matches identified (6 sets of 50 random POIs assigned to 6 volunteers), we concluded that the percentage of correct similarities identified was above 98% ($\sigma = 1.79$). Differently to validations mentioned in this document, this is an extremely objective one, not demanding external participants or a very large sample [2].

[2]Using the central limit theorem, the standard error of the mean should be near 0.73. Assuming an underestimation bias for n=6 of 5% (according to [32]), accuracy keeps very high, being the 95% confidence interval [96.5%, 98.7%]

Approach	NAICS2	NAICS4	NAICS6
Coma++ w/context	85.92	50.89	33.43
Coma++ nodes only	69.23	39.37	20.96
Wordnet+Levenstein	52.66	29.19	20.18
Wordnet+JaroWinkler v1	61.81	40.08	31.30
Wordnet+JaroWinkler v2	75.32	58.63	42.23
Wordnet+JaroWinkler v3	79.31	59.91	43.08

TABLE 5.2: Accuracies for the Ontology Matching (COMA++) and WordNet approaches

5.4 POI classification

In this section we present the results we obtained for the different approaches we used for POI classification.

Table 5.2 and figure 5.11 shows the results we obtained using the Ontology Matching and the WordNet approaches for different NAICS levels, i.e., two, four and six-digit NAICS codes. The "w/context" and the "nodes only" in the COMA++ results indicate the type of ontology mapping performed: using the context of the class to determine the mappings or only the nodes. The four WordNet approaches represented in the graph refer to multiple implementations. The first one uses the Levenstein string comparison algorithm to compare the category names, while the others use the already mentioned JaroWinklerTFIDF library from the SecondString project. The multiple versions (v1, v2 and v3) refer to different implementation for the match weighting and result merging algorithms.

If we analyze the results, we can see that all the algorithms perform better for the two-digit NAICS, which is expected since the four and six-digit NAICS are much narrower than the two-digit and thus more difficult to determine. We can also notice that the WordNet approach performs better than Ontology Mapping approaches using COMA++, except for the two-digit NAICS where one of the COMA++ approaches outperformed the others. However, it is important to note that COMA++ is not able to find mappings for all the classes in the ontology (i.e. not all the categories have a mapping), therefore the approach is only able to classify a small part of the dataset, and the results depicted in Figure 5.11 are only referent to the POIs the algorithms were able to classify. The same problem happens with the WordNet approach because we defined a threshold to discard matches with less than 0.70 of similarity in the category names, thus the final NAICS code is determined according to the best match in the set of matches with a similarity above 0.70. Nevertheless, the percentage of the POIs

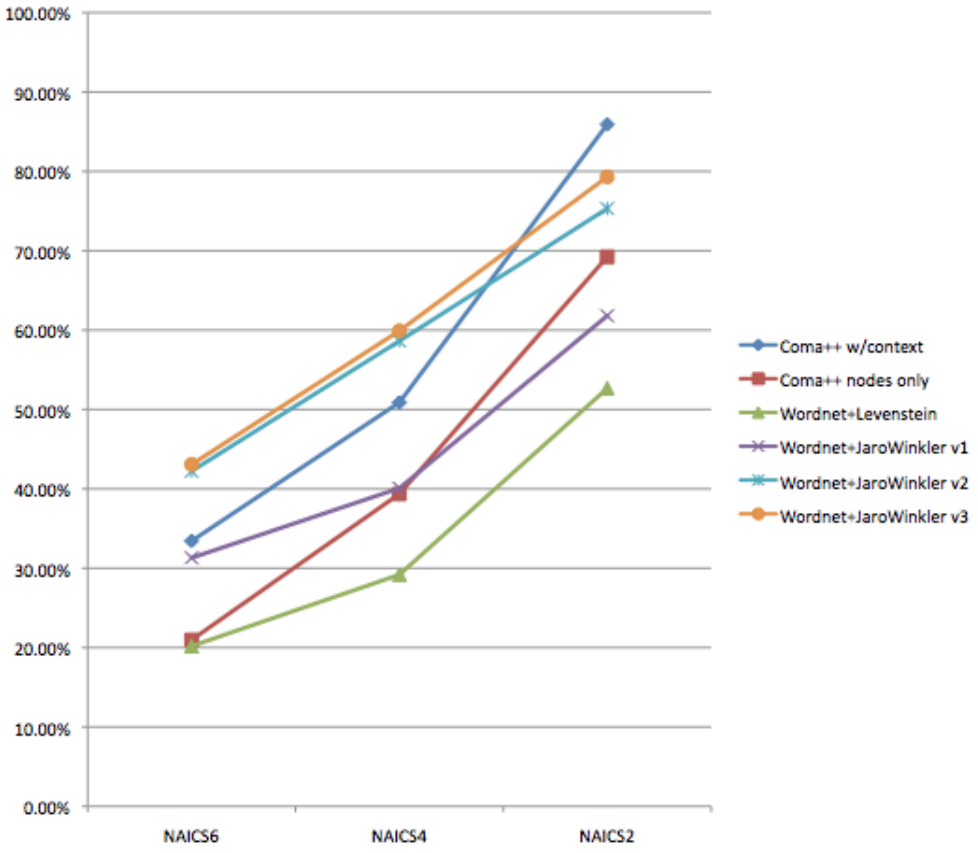

FIGURE 5.11: Accuracies for the Ontology Matching (COMA++) and WordNet approaches

that the WordNet approach is not able to classify is below 15% while for the COMA++ approaches this value can go up to 50%.

Table 5.3 shows the accuracies obtained using different machine learning algorithms for different NAICS levels (two, four and six-digit codes) using dataset A. There are some missing results in the table because the algorithm took over 72 hours to run.

Before we start analyzing the machine learning results, it is important to mention that the ZeroR and OneR algorithms, because of the way they work, were not applied to "compete" for the best results against the other algorithms. Instead, they merely serve as baselines for the other algorithms.

As expected, we obtained better results classifying POIs to the two-level NAICS than for the six-level NAICS, since the eventual noise due to ambiguous classifications in the POI datasets is smaller.

We can see that the tree-based (e.g. ID3, RandomForest) and instance-based learning approaches (e.g. IBk, K*) are the ones that perform better in this classification

Algorithm	NAICS2	NAICS4	NAICS6
DecisionStump	22.271	10.137	-
FT	85.759	-	-
ID3	84.248	75.837	72.119
J48	83.397	75.755	71.282
J48graft	83.823	76.358	71.776
RandomForest	84.879	77.099	72.983
RandomTree	84.207	75.906	72.379
DecisionTable	77.840	71.256	-
JRip	79.624	72.187	67.838
IB1	80.736	70.952	65.299
IBk	84.989	76.811	73.052
K*	84.893	77.566	73.408
BayesNet	80.681	56.394	42.440
NaiveBayes	74.547	40.354	28.444
MultilayerPerceptron	5.762	-	-
ZeroR	14.586	9.701	9.701
OneR	21.858	12.349	12.088

TABLE 5.3: Results obtained for the different machine learning algorithms with POIs from dataset A for the Boston area

task, especially the latter. Notice that only 80,2% of data is classified in a totally non-ambiguous way. The most successful algorithm is IBk (with k=1), which essentially finds the similar test case and assigns the same NAICS code. The difference in accuracy between tree-based and instance based approaches is very small to make strong conclusions, however we could expect that instance based models bring better results since the distribution of the different Yahoo! categories is relatively even among examples of the same NAICS code (implying no clear "dominance" of some categories over others). Understandably, the Naive Bayes algorithm performs badly because the assumption that different Yahoo! categories for the same NAICS classification are independently distributed is obviously false (e.g. "Doctors & Clinics, Laboratories, Medical Laboratories" are correlated). Such assumption is not fully necessary in Bayesian Networks, which actually brings better results. Unfortunately, we could not find a model search algorithm that performs in acceptable time (less than 72 hours) and produces a more accurate model. We used Simulated Annealing and Hill Climbing.

In table 5.4 we can see the results obtained by modifying the POI dataset, so that the NAICS codes of POIs where ambiguities arise are grouped together in the same "super-category", eliminating the inconsistencies. This way, if 95% of the POIs from Yahoo! with both the categories "Book Printing" and "Book Publishing" have the NAICS code "323117", we will change the NAICS code of the remaining 5% of the POIs with those two categories to "323117".

Algorithm	NAICS2	NAICS4	NAICS6
ID3	92.975	89.728	88.680
RandomForest	93.609	90.805	89.846
IBk	94.170	91.189	89.979

TABLE 5.4: Results obtained for the different machine learning algorithms using a re-classified dataset

Algorithm	NAICS2	NAICS4	NAICS6
ID3	83.061	73.586	69.209
RandomForest	83.488	74.867	70.318
IBk	83.360	74.909	70.276

TABLE 5.5: Results obtained for the different machine learning algorithms using POI data from Boston for training and POI data from New York for testing

By comparing the results in table 5.4 with the results in table 5.3, we realize that the NAICS labeling inconsistencies in the POI data have a major negative effect in the performance of the machine learning algorithms, reducing the accuracy in more than 16% in some cases for the six-level NAICS codes.

It would be expectable to obtain accuracies more close to 100% for the results in table 5.4. However, that does not happen due to the fact that 115 of the 514 NAICS codes covered by our dataset A only occur once. Therefore, when we split the dataset to perform the ten-fold cross-validation, a significative number of the test cases will have NAICS codes that the algorithm was not trained for, causing it to incorrectly classify them.

Table 5.5 shows the results we obtained by training the machine learning approaches with dataset A from Boston and Cambridge and testing them with New York POI data. As we can see in the results, if we apply the generated model to a different city, it still performs well, even though the accuracy drops a little in some cases. This is perfectly understandable since even the Yahoo! taxonomy differs slightly from city to city.

Table 5.6 shows the results obtained for the different machine learning algorithms using dataset B.

By analyzing the results from table 5.6 we can see that the results have significantly improved over dataset A, which shows the importance of the training data in the performance of the machine learning algorithms.

Table 5.7 shows the results obtained using both the categories from Yahoo! and the semantic annotations. Please remember that the presented results are based on semantic enriched POIs from dataset A.

Algorithm	NAICS2	NAICS4	NAICS6
ID3	90.567	85.459	82.091
J48	90.113	85.085	81.831
RandomForest	90.758	85.710	82.436
RandomTree	90.500	85.275	81.818
JRip	85.748	80.998	78.495
IB1	87.224	81.495	76.826
IBk	91.024	85.974	82.553
K*	90.227	85.849	82.522
BayesNet	88.961	77.964	67.877
NaiveBayes	87.910	70.250	56.052
ZeroR	16.580	4.994	4.797
OneR	23.013	9.480	7.107

TABLE 5.6: Results obtained for the different machine learning algorithms with POIs from dataset B for the Boston area

Algorithm	NAICS2	NAICS4	NAICS6
ID3	83.967	75.986	72.087
J48	82.935	75.504	71.559
RandomForest	86.467	78.876	75.848
RandomTree	81.467	73.417	69.289
JRip	81.307	73.509	69.289
IB1	85.435	76.582	72.706
IBk	86.903	79.059	75.619
K*	86.834	79.541	76.261
BayesNet	80.183	56.467	40.137
NaiveBayes	74.541	30.688	20.091

TABLE 5.7: Results obtained for the different machine learning algorithms with POI data from the Boston area using semantic annotations

By comparing the results in table 5.7 (with semantics) with the ones presented in table 5.3 (without semantics), we can see that there was some improvement in some algorithms (like IBk). This is somehow understandable if we consider the way IBk (k-nearest neighbor) works. Since it measures the euclidean distances from the test case to all training examples, having more information about the POI (in this case semantic annotations) would supposably help, thus increasing the accuracy. On the other hand the performance of other algorithms such as ID3 decreased when compared to the results from table 5.3. This fact suggests that having semantic information about the POIs might be difficulting the choice of the next feature to use in the decision tree by messing with the entropy and gain computation.

Finally tables 5.8 to 5.11 show the results obtained using the different hierarchical classification schemes for various types of machine learning algorithms. Like in previous

Algorithm	Flat classification Accuracy	Hierarchical classification Level1 acc.	Level2 acc.
ID3	85.459	90.659	85.620
J48	85.085	90.172	84.9009
RandomForest	85.710	90.959	85.969
RandomTree	85.275	90.509	85.315
JRip	80.998	85.806	80.440
IB1	81.495	87.637	81.126
IBk	85.974	91.080	86.097
K*	85.849	90.305	85.244
BayesNet	77.964	88.002	74.243
NaiveBayes	70.250	30.688	20.091

TABLE 5.8: Comparison between the results for dataset B using flat classification (4-digit NAICS) and hierarchical classification with 2 levels (NAICS 2 and 4)

Algorithm	Flat classification Accuracy	Hierarchical classification Level1 acc.	Level2 acc.	Level3 acc.
ID3	85.459	90.659	88.766	85.575
J48	85.085	-	-	-
RandomForest	85.710	-	-	-
RandomTree	85.275	90.509	88.490	85.226
JRip	80.998	-	-	-
IB1	81.495	87.637	85.336	81.126
IBk	85.974	91.080	89.219	86.097
K*	85.849	-	-	-
BayesNet	77.964	-	-	-
NaiveBayes	70.250	-	-	-

TABLE 5.9: Comparison between the results for dataset B using flat classification (4-digit NAICS) and hierarchical classification with 3 levels (NAICS 2, 3 and 4)

tables, there are some missing values because the algorithms took over 72 hours to run.

Intuitively, we thought that performing classification in a hierarchical way would always produce better results than using standard flat classification, however it was not the case. Instead, for some algorithms the results improved while for others they didn't. Therefore, we cannot say that using hierarchical classification for classifying POIs to NAICS is always a better solution. In fact, as shown before by comparing the datasets A and B, the quality and the dimensions of the dataset seems to have a much bigger impact on the results than whether we apply hierarchical or flat classification.

Another interesting fact in the results from the hierarchical classification is that the accuracies vary considerably with the hierarchy type used. For instance, when classifying POIs with 6-digit NAICS codes, we can see that using a two-level hierarchy

Algorithm	Flat classification Accuracy	Hierarchical classification Level1 acc.	Level2 acc.
ID3	82.091	90.659	82.100
J48	81.831	90.173	81.484
RandomForest	82.436	90.959	82.477
RandomTree	81.818	90.509	81.654
JRip	78.495	85.806	76.398
IB1	76.826	87.637	76.826
IBk	82.553	91.080	82.551
K*	82.522	90.305	81.661
BayesNet	67.877	89.059	69.336
NaiveBayes	56.052	88.002	59.885

TABLE 5.10: Comparison between the results for dataset B using flat classification (6-digit NAICS) and hierarchical classification with 2 levels (NAICS 2 and 6)

Algorithm	Flat classification Accuracy	Hierarchical classification Level1 acc.	Level2 acc.	Level3 acc.
ID3	82.091	90.659	85.620	82.111
J48	81.831	90.172	84.901	81.341
RandomForest	82.436	90.959	85.969	82.398
RandomTree	81.818	90.509	85.315	81.694
JRip	78.495	85.806	80.440	76.889
IB1	76.826	87.637	81.126	76.826
IBk	82.553	91.080	86.097	82.539
K*	82.522	90.305	85.244	81.486
BayesNet	67.877	-	-	-
NaiveBayes	56.052	-	-	-

TABLE 5.11: Comparison between the results for dataset B using flat classification (6-digit NAICS) and hierarchical classification with 3 levels (NAICS 2, 4 and 6)

the RandomForest algorithm improved over the flat classification, while using a three-level hierarchy it actually became worse (although the differences in accuracy are small).

5.5 POI generation

Since we didn't perform validation of the POI generation approach, for reasons we already explained before, in this section we present and discuss a few case studies of some generated places of interest.

Figures 5.12 to 5.16 show some of the identified places of interest using the DBScan clustering algorithm and our similarity based TF-IDF algorithm. By analyzing the figures presented we can have an idea of the quality of the approaches developed and identify some weaknesses.

FIGURE 5.12: Screenshot 1 of places of interest identified by our POI generation approach

Figure 5.12 shows a perfect example of well identified place of interest. If the user goes to google maps, for instance, and searches for the "Boston Museum of Science" she can see that it is in fact in the place identified by the marker in the figure. In other words, the centroid of the cluster corresponds exactly were the POI is, and the name identified using our TF-IDF algorithm over the Flickr tags unambiguously identifies the place.

Figure 5.13 depicts a weakness of our approach, which is events. Events typically occur recurrently in the same place, and also usually generate lots of photos that attendees upload to Flickr. Therefore it is very common to find lots of pictures of events in Flickr. Since these events are often restricted to a small area, the clustering algorithm will generate a cluster for them, and since the tags associated with it appear very frequently in that cluster and very infrequently elsewhere, the TF-IDF algorithm will attribute a high score to those tags. A possible solution to overcome this problem would be to make an analysis of the photos over time. If the photos in a cluster are all from a single short time period, or if the photos appear to have a ciclic pattern over time (monthly or yearly, for instance), then maybe we should reconsider the TF-IDF score that we assign to the tags associated with those photos.

Figure 5.14 shows another correctly identified place of interest which is the "Massachusetts General Hospital". This screenshot is important to understand that a place can be referred by different names. In this case, the "Massachusetts General Hospital"

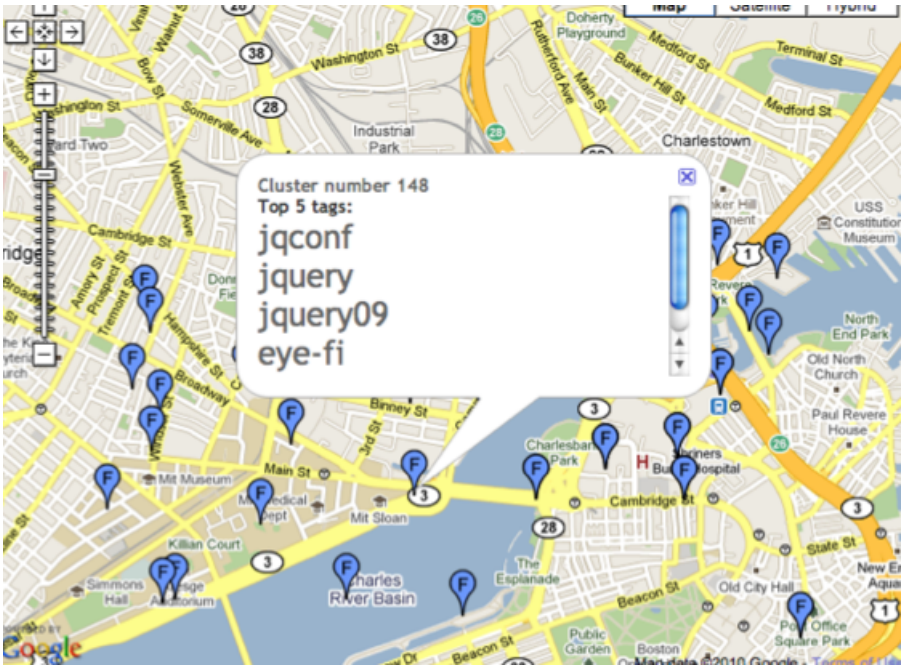

FIGURE 5.13: Screenshot 2 of places of interest identified by our POI generation approach

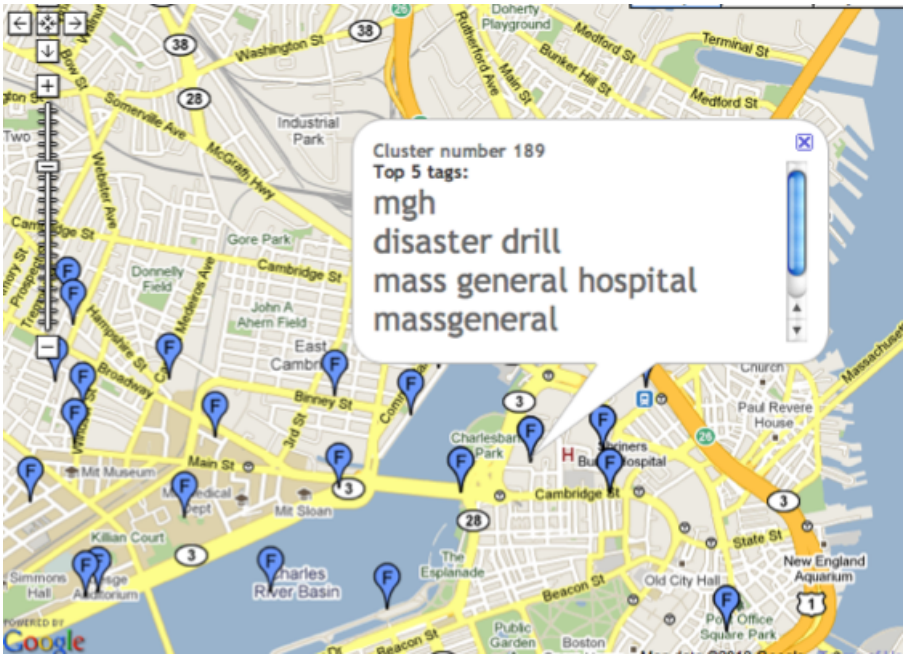

FIGURE 5.14: Screenshot 3 of places of interest identified by our POI generation approach

appears referred to as "mgh", "mass general hospital" and "massgeneral". This represents another major issue in naming clusters. The fact that we use the similarity-based TF-IDF approach helps us deal with some of these issues, but even that solution is incapable of determining that "mgh" and "mass general hospital" are tags that refer to the same thing.

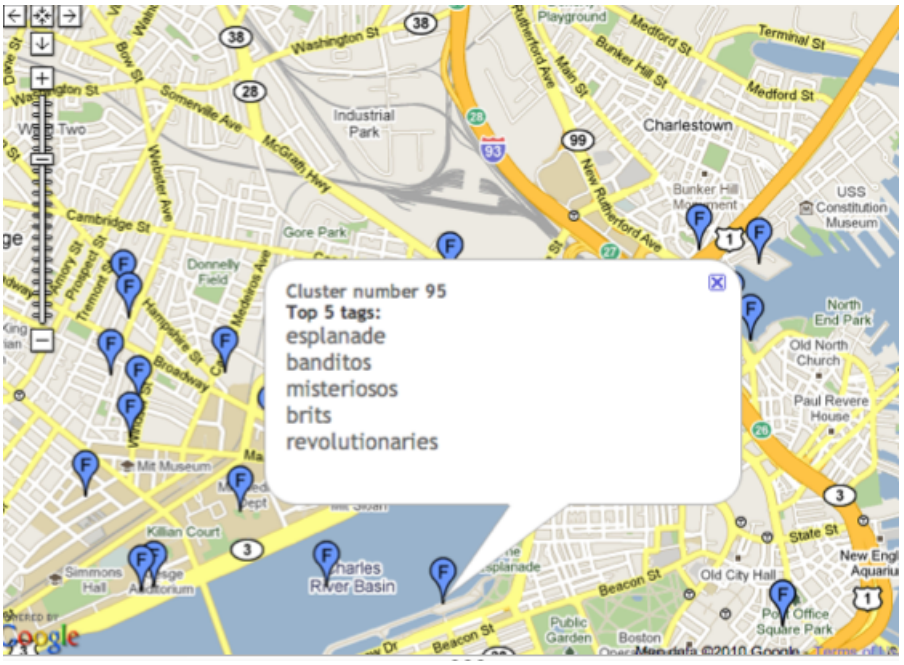

FIGURE 5.15: Screenshot 4 of places of interest identified by our POI generation approach

Figure 5.15 shows another bad case. In this particular case, the cluster includes many different places of interest and therefore the top-5 tags associated with it do not uniquely identify a single place. This raises other problems which are cluster size and clustering parameters. These problems will be further discussed later in this section.

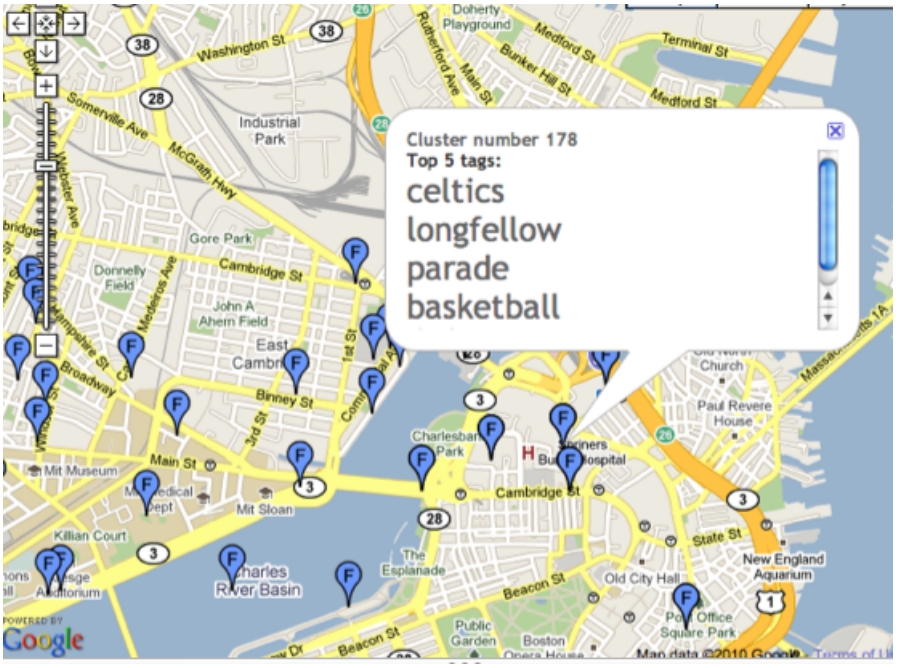

FIGURE 5.16: Screenshot 5 of places of interest identified by our POI generation approach

Finally, figure 5.16 shows a example where the cluster centroid corresponds well to the place it represents ("Boston Celtics Stadium"), however the top-5 tags are not good enough to unambiguously represent it, specially for someone who is not familiarized with the city of Boston.

An important result that we observed, however it was somehow expected, was the enormous impact that the clustering parameters, and hence the cluster size, have on the results. Not to mention, the problem of identifying which is the best clustering algorithm. Intuitively, one would thing that a density-based clustering method would be more suited for this kind of task. However, choosing the correct parameters can be very difficult for this type of clustering, and therefore, other types of clustering methods like K-means are not such bad options.

Figures 5.17 and 5.18 show the different patterns of the generated clusters using K-means and DBScan respectively. As we can see in these figures, the distributions of the generated clusters in space are a lot different, which once again highlights the impact of the clustering algorithm used.

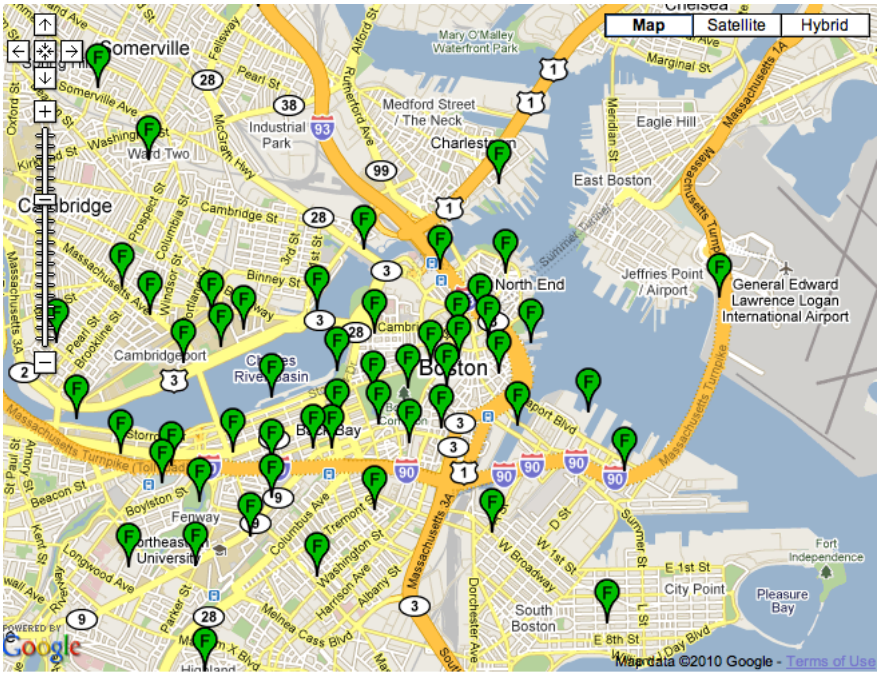

FIGURE 5.17: Screenshot of the generated clusters using K-means (k=100)

Figure 5.19 shows the clusters obtained using DBScan with the following parameters: eps=0.005, minPts=50. By comparing this figure with figure 5.18 we can see that a minor change in parameters of the clustering algorithm (eps=0.005 instead of 0.001) can have a huge impact on the results. The impact of these parameters in the results is also an important factor that complicates the validation of this approach.

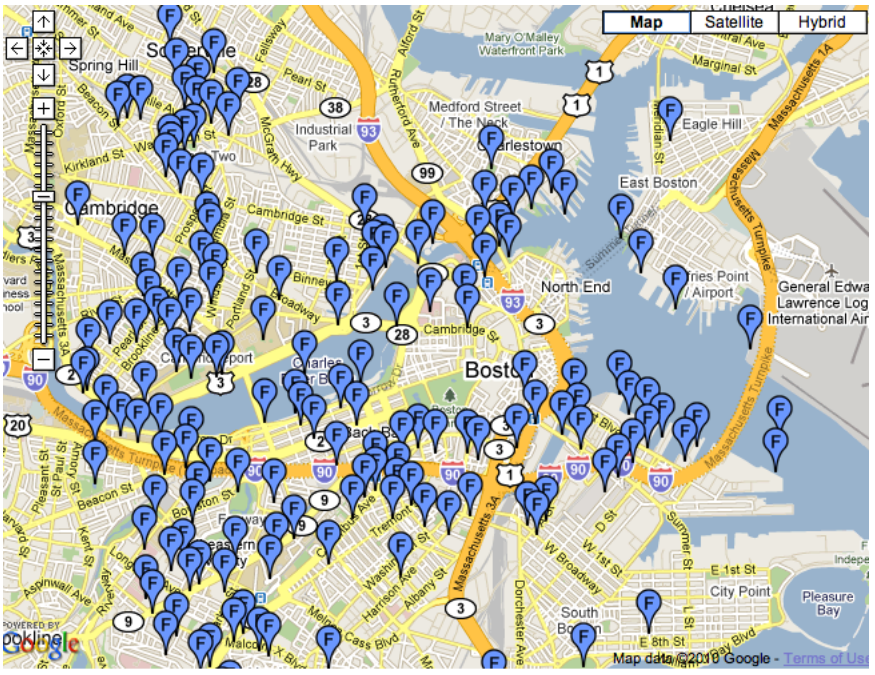

FIGURE 5.18: Screenshot of the generated clusters using DBScan (eps=0.001, minPts=50)

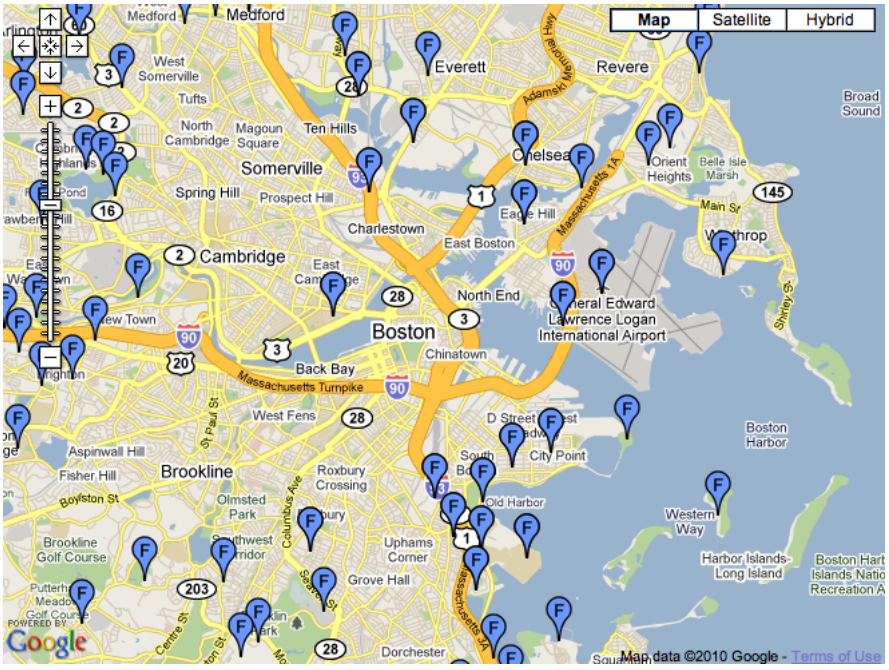

FIGURE 5.19: Screenshot of the generated clusters using DBScan (eps=0.005, minPts=50)

By making an overall analysis, the results we obtained are in fact good (although we showed some bad examples here). If the reader wants to analyze them personally, the results can be seen at the following web address: http://www.greenhomes.dei.uc.pt/Maps/index.html. In this webpage you can choose between the two different clustering algorithms used and change some of the clustering parameters. Besides that, you can

also choose between the results of our similarity-based TF-IDF and standard TF-IDF (the similarity-based TF-IDF is mentioned in the webpage as the "v2").

Chapter 6

Conclusion

We have presented a multi-threaded application that allows the automated extraction of Points of Interest (POI) for different areas in the world and from some of the thousands of directories available throughout the Internet such as Yahoo, Yellow Pages, Manta, CitySearch, etc. This application allowed us to collect a total of 981956 POIs, mainly for the areas of Boston, New York, San Francisco and Lisbon. Using this wealthy information we developed a POI Matching algorithm that allows us to determine similar POIs among different sources with a precision of 98%, hence reducing the amount of redundant information in our database and collecting as much information as possible about a POI in a single database entry. We also made a few simple analyses over the collected POI data mostly for the Boston Metropolitan Area.

Based on the results from the POI Matching algorithm, we proposed three classification methods in order to classify POIs from their source taxonomy to a more convenient one, particularly to the North American Industry Classification System (NAICS). These methods are based on different approaches using Ontology Matching (see Section 3.3.1), WordNet (see Section 3.3.2) and Machine Learning algorithms (see Section 3.3.3). Using Machine Learning algorithms we experimented different solutions:

- flat and hierarchical classification schemes
- with and without semantic annotations
- with training sets with different sizes and from different sources (namely D&B and InfoUSA)

Based on the obtained results we can conclude that the Machine Learning approaches are the ones that provide better accuracies. Among the different algorithms tested it is hard to name one that over-performs all the others, however we can say with good

confidence that the instance-based learners (lazy), like the k-nearest neighbor (IBk), and the tree-based learners, like RandomForests and ID3, are the ones more suited for this particular classification task.

Regarding our experiments with hierarchical classification schemes and with semantic annotations we got some improvements in some cases, however the results were not good enough to make strong conclusions about their quality over the base approach (flat and without semantics). As opposed to this, the results using NAICS codes from InfoUSA (dataset B) proved that the dimension and the quality of the training set has a major impact on the accuracies of the classifiers.

In conclusion, we proved that is possible to classify POIs from web-based sources (in our case Yahoo!) with 2-digit and 6-digit NAICS codes with accuracies of roughly 90% and 82% respectively. These accuracies are more than acceptable for the application of the classified POI data in Urban Planning studies like the one presented in appendix D. In this study, the classified POIs were applied to the urban modeling task of employment size and location disaggregation from Block Group level to Block level and the results show encouraging quality. By using our approaches, the researches in the field of Urban Planning can less dependent on proprietary POI databases that quickly become obsolete.

Finally, we presented an approach to infer places of interest based on geo-referenced photos from the very popular photo sharing website Flickr. In this approach we use a combination of clustering and TF-IDF, where the clustering algorithms are used to identify clusters of photos in space and the TF-IDF rating is used to identify the top-5 tags associated with each cluster, which allow the user to identify the place.

We also presented a modified version of TF-IDF that can improve the scoring of terms that can contain errors, misspells or have multiple ways of being spelled. In our particular case, we saw some improvement of using this similarity-based TF-IDF over standard TF-IDF. However we cannot make any strong conclusion about that because we didn't perform a proper validation. Instead, we presented some case studies that demonstrated the strengths and the weaknesses of the approach.

Despite the good potential demonstrated by our POI generation approach (like the many others presented in section 2.5), it is still too soon to migrate to fully automated POI generation solutions. Therefore, for the next years, we will probably still rely on user-contributed and proprietary POI databases, with particular focus on the former. It is however clear that these kinds of areas will certainly be the focus of many research works and will have many contributions in the future.

Chapter 7

Final thoughts on the research

In this chapter I discuss some aspects of my research during the last year.

The research work was not totally new for me, neither was the workgroup I was part of. I started working with Professor Francisco Pereira two years ago, but during the first year my work was only part-time and the focus of that work was different from my current work. Therefore, by working full-time, this last year gave me a whole new perspective on research, which I found to be very interesting and motivating. I also had the opportunity to work closely with other researchers like Ana Alves (who was also my co-supervisor during last year) and João Oliveirinha. I was part of the AmILab workgroup which I consider to have a great working environment and a good companionship.

Regarding the initial objectives of my research, I think they were all achieved. My work was supposed to have two main components: POI Mining and POI Generation. It ended up having a third one, POI Classification, which was not considered at first, but we later realized its huge importance and we decided we should focus on that question as well. The POI Classification work opened up a new collaboration opportunity with researchers Shan Jiang and Professor Joshep Ferreira at the Massachusetts Institute of Technology. This collaboration was great because not only I had the opportunity to collaborate with researchers from a area different than informatics, but it also gave me the pleasure of working with foreign researches from a world famous university.

Figure 7.1 shows a Gantt chart with the planning of research work. In this chart green cells represent free time (i.e. the task took less time than expected) and red cells represent time that was not supposed to be spent on that task (i.e. the task was overdue). I tried to be as honest and accurate as possible when filling that chart.

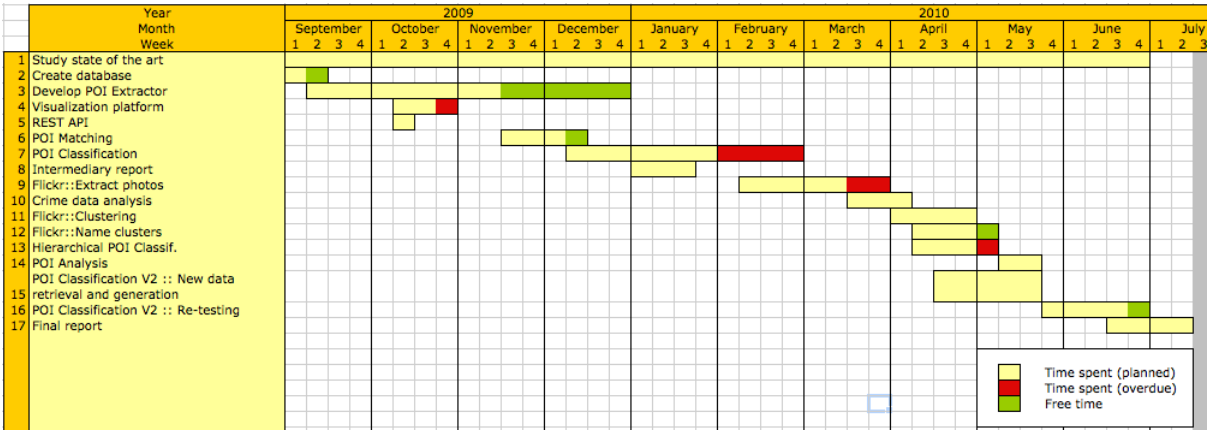

FIGURE 7.1: Gantt chart of planning of my research

As we can see in the Gantt chart, all tasks took more or less the expected time. We can also see that I didn't assigned specific time slots to the study of the state of the art. I made this choice because this study took place during all the year in parallel with the other tasks, even thought it was more intensive in certain times.

During last year, the following three papers were submitted to different conferences:

- Towards an activity-based approach for estimating travel destinations, Shan, J., Rodrigues, F., Alves, A., Pereira, F., Ferreira, J. - WCTR (Accepted)
- Automatic classification of Points-of-Interest for land-use analysis, Rodrigues, F., Pereira, F., Alves, A., Shan, J., Ferreira, J. - SIGSPATIAL (Submitted)
- Place in perspective: Extracting online information about Points of Interest, Alves, A., Pereira, F., Rodrigues, F., Oliveirinha, J. - AmI (Submitted)

In conclusion, I think my research went well, fulfilled the objectives proposed and was a very valuable experience for me.

Appendix A

Web Scraping techniques

There are many techniques to extract information from a webpage. This section describes the five most commonly used.

A.1 Human copy-paste

As odd as it may sound, copy-paste is considered an example of Web scraping by many people. Although it is very easy to do, it takes a huge amount of time and patience, making it almost impossible to use for large numbers of pages. Besides, if you're hosting a web service to serve as a wrapper and deliver the scraped information, you need to make the extraction process automated. However, for a very small number of pages, copy-paste works just fine.

Advantages of this technique

- You don't need to have any programming skills.
- No time spent developing a scraper.

Disadvantages of this technique

- It is not scalable. It would be almost impossible to use for a large number of pages.
- It is a very monotonous, time consuming and boring technique.

A.2 String manipulation

This is probably the most used technique because it involves no needs for the programers to learn about regular expressions or any other technique. This technique is also considered the most "ad-hoc" one. It consists in using a complex set of substring(), indexOf() and split() method calls over the string object containing the HTML source code. Considering this, it is not hard to imagine the amount of tests you will have to make before you get your scraper working.

Advantages of this technique

- You don't need understand regular expressions.
- No need for any third-party library. You use only built-in functionalities of the programming language.
- Easy to do. You don't have to know a lot of programming.

Disadvantages of this technique

- Working with strings like this is very error-prone and takes a lot experiences until you get the right result.
- The resulting code tends to be very messy.
- If the web page changes its structure, even only by a little bit, it is very likely that your whole scraper will become unusable. You won't be able to patch it up because you probably won't understand the code you wrote a few time ago.

A.3 Regular expressions

Regular expressions are a very, very powerful tool. When you mastered the use of them, you will ask yourself why you have not learn them before. Regular expressions, or "regexes" for short, are very useful when it comes to string manipulation. Take a look at following example to understand the difference.

Imagine you need to validate e-mail addresses. The verbal directions for doing so might be something along the lines of "Make sure the e-mail address contains an at (@) symbol." You could probably handle this task with a single line of Java code:

```
If (email.indexOf("@") > 0) {
   return true;
}
```

So far, so good. Suppose additional requirements creep in, though, as they invariably do. Now you also need to make sure that all e-mail addresses end with the .org extension. So you amend your code as follows:

```
If ((email.indexOf("@") > 0) && (email.endsWith(".org"))){
   return true;
}
```

But the requirements continue to creep. You now need all e-mail addresses to be of the form firstname_lastname, so you use the StringTokenizer to tokenize the e-mail address, extract the part before the @, look for the underscore (_) character, tokenize the strings around that, and so on. Pretty soon, you have some convoluted code for what should be a fairly straightforward operation. The use of regular expressions can greatly simplify and condense this process. With regular expressions, you could write the following:

```
String regex = "[A-Za-z]+_[A-Za-z]+@[A-Za-z]+\\.org";
if (email.matches(regex)) return true;
```

Advantages of this technique

- The scraper code will be smaller and much clearer (although the regular expressions will probably need some explanatory information).
- Higher development speed.
- Regular expressions make the scraper much easier to patch/update when a change is made in web page structure.

Disadvantages of this technique

- You will have to learn regular expressions and some developers find them hard to use.

A.4 DOM

DOM (short for Document Object Model) is a cross-platform and language-independent convention for representing and interacting with objects in HTML, XHTML and XML documents. Aspects of the DOM (such as its "Elements") may be addressed and manipulated within the syntax of the programming language in use. The public interface of a DOM are specified in its Application Programming Interface (API).

Using DOM you can load an entire web page to an object in memory, and then iterate trough its element and navigate through their children, parents and siblings.

Advantages of this technique

- It is very easy to understand how the DOM works.

Disadvantages of this technique

- Your code will probably become very confusing and hard to understand.
- The web page needs to be very well formatted (which most of the times does not happen).
- You will probably need to use some kind of third-party library for this (most of the times the built-in DOM library is not very easy to use, and third-party libraries often have increased functionality).
- Increased memory usage. The complete DOM tree needs to be loaded to memory in the object-oriented paradigm.

A.5 XPath

XPath is used to navigate through elements and attributes in an XML document. XPath is a major element in W3C's XSLT standard - and XQuery and XPointer are both built on XPath expressions.

Using XPath you can easily select any elements in HTML. For that you can use a expressions like this: id('divxpto')/ul/li[2]/a. This expression will navigate to the element with id='divxpto', then go to the "ul" child, then to second "li" element and finally, it will get the "a" element.

Advantages of this technique

- The code is very easy to maintain, patch or update.
- Higher development speed.
- There are lots of great tools available to test your XPath expressions in web pages.

Disadvantages of this technique

- You will have to make updates very frequently to the XPath expressions.
- The web page needs to be very well formatted (which most of the times does not happen).
- You will need to learn how to use XPath expressions.
- Depending on the programming language, you might need to use some kind of third-party library for this.
- Increased memory usage. The complete DOM tree also needs to be loaded into memory in the object-oriented paradigm.

Appendix B

The database

Figure B.1 depict the E-R model of our database.

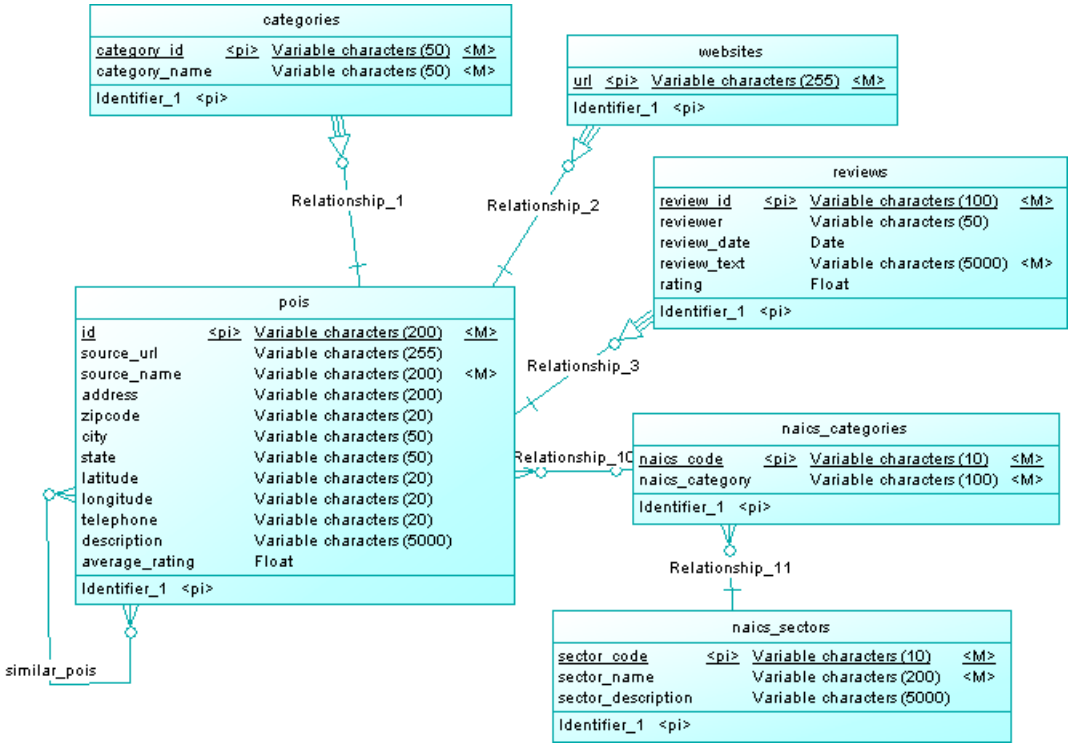

FIGURE B.1: E-R model of the database

Appendix C

The REST Web service

This section describes the methods available through our REST Web service, along with the parameters each one takes.

The Web service URI is:
http://greenhomes.dei.uc.pt:8080/POIsREST/resources/

The following list describes the resources that are available through our Web service and the correspondent input parameters:

POIsREST/resources/pois

Provides a list of POIs inside a given circular area defined a center and a radius. Input parameters:

- lat: latitude of the center of the circle.
- lng: longitude of the center of the circle.
- radius: radius of the circle.
- ne (optional): northeast coordinates of top-left corner of the map (for map statistic information only).
- sw (optional): southwest coordinates of bottom-right corner of the map (for map statistic information only).
- perspective: perspective were retreiving POIs from; can be: openweb, upcoming, boston_globe, wiki or all.
- output: the format of the response; can be either JSON or XML.

Example:
http://greenhomes.dei.uc.pt:8080/POIsREST/resources/pois?

lat=42.35854391749705&lng=-71.06042861938477&radius=0.002&
perspective=all&ne=(42.42269621215634, -70.95588684082031)&
sw=(42.29584977392906, -71.18247985839844)&output=json

POIsREST/resources/semantics

Provides details and semantic information about a given POI. Input parameters:

- id: the ID of the POI.
- semantics: whether of not the service should return semantic information about the POI along with the details of it; can be either true or false.
- output: the format of the response; can be either JSON or XML.

Example:
http://greenhomes.dei.uc.pt:8080/POIsREST/resources/semantics?
id=yahoo_42.360618_-71.060838_10152836&semantics=true&output=json

POIsREST/resources/search

Allows the search of POIs by ID, name or address. Only one of the search parameters is required. If more than one search parameter is specified a AND operation is performed. Input parameters:

- id: the ID of the POI.
- name: the name of the POI.
- address: the address of the POI.
- output: the format of the response; can be either JSON or XML.

Example:
http://greenhomes.dei.uc.pt:8080/POIsREST/resources/search?
name=apple&output=json

POIsREST/resources/counts

Provides statistic information about the number of POIs in the database. Input parameters:

- output: the format of the response; can be either JSON or XML.

Example:
http://greenhomes.dei.uc.pt:8080/POIsREST/resources/counts?
output=json

POIsREST/resources/event

Provides details about a given event. Events are associated with some POIs. Input parameters:

- id: the ID of the event.
- output: the format of the response; can be either JSON or XML.

Example:
http://greenhomes.dei.uc.pt:8080/POIsREST/resources/event?
id=89404555&output=json

Appendix D

An application in Urban Planning

In this appendix we describe a practical application of Yahoo! POIs classified to NAICS using a non-hierarchical approach with the k-nearest neighbor classifier (IBk) without semantic annotations (see section 3.3.3.3 for more details).

In the field of Urban Planning, urban simulation models have evolved significantly in the past several decades. For instance, the travel demand modeling approach has been in the transition from the traditional Four-Step Model (FSM) to the Activity-Based Model (ABM) [38]. Consequently, requirements for disaggregated data increase greatly, ranging from population data, employment data, to travel survey data. The employment data (on the travel destination side) is usually obtained from proprietary sources, which adds another layer of barriers to widely applying Activity-Based Modeling approach, let alone the expensive travel-survey data acquisition. In order to study this issue, researchers are trying to develop new methods of estimating disaggregated employment size and location by category.

In our case, we intend to develop a set of new methods and demonstrate their applications for estimating activities, incorporating them into travel demand and urban simulation models. This will be beneficial for cities that lack detailed survey data for building Activity-Based Models but wish to test the sensitivity of travel behavior to policy changes such as Intelligent Transportation Systems (ITS) implementations that are likely to alter activity patterns. An important step to achieve these goals is to obtain a disaggregated employment distribution by POIs of an area. For the case of Cambridge, MA, we have official data at the Block Group (BG) level (obtained from the U.S. Census Transportation Planning Package 2000), which essentially describes the total size of employees by economic sector at that spacial resolution. We need to distribute these totals into Block or Parcel level.

For demonstration purposes we only use POIs from the "Retail Trade" sector of the NAICS taxonomy, i.e. categories whose code starts by 44 or 45. Figures D.1 and D.2 show the aggregated retail employment density at the Block Group level and distribution of our POI data from Yahoo! at the Census Block level for Cambridge, respectively.

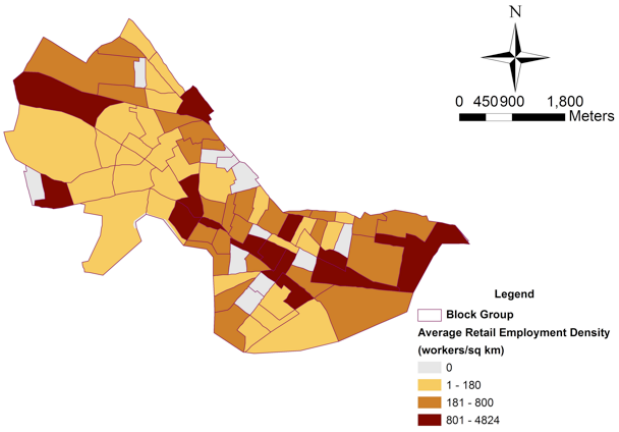

FIGURE D.1: Aggregated retail employment density at the Block Group level (pl/sq km= employed people per square kilometer).

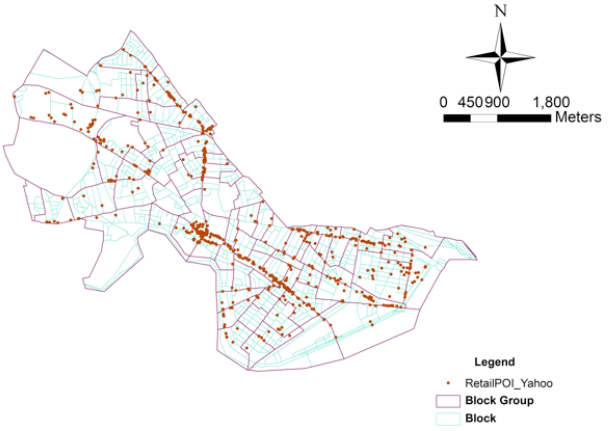

FIGURE D.2: Cambridge retail POI distributions from Yahoo!

By using the business establishment survey data (from InfoUSA, 2007) which is believed to be close to the population, we are able to obtain a benchmark estimate of employment size by category at the Census Block level for the study areas. This will function as a ground truth to test our algorithm. Notice however that the dates for each of the databases are quite distinct (2000 for Census, 2007 for InfoUSA and 2010 for Yahoo!) therefore some error is expected to happen.

We employ local maximum likelihood estimation (MLE) method as described below to derive the disaggregated destination estimation at Block level.

1. We calculate the total number of POIs (destinations) by category c in each Block b.

2. We assume that the employment size at destination d in Block Group g of category c is proportional to some function f of its associated block area $a_{d,c,g}$, which means the effective area of the destination d in Block Group g of category c. The form of function f will be explored based on the empirical data, and we also allow the possibility that $f(a_{d,c,g}) = a_{d,c,g}$ which is the natural benchmark case. Mathematically, assume that for employment category c, there are $n_{c,g}$ destinations at Block Group g. For $d = 1, 2, \ldots, n_{c,g}$, let random variable $e_{d,c,g}$ be the employment size of category c at destination d in Block Group g.

3. We assume that $e_{d,c,g}(d = 1, 2, \ldots, n_{c,g})$ are i.i.d.$(f(a_{d,c,g}) \cdot \alpha_{c,g}, \sigma^2_{c,g})$, where $\alpha_{c,g}$ is the employment size of category c per unit of effective area at Block Group g; $\alpha_{c,g}$ and $\sigma_{c,g}$ are positive constants independent of d. $E(e_{d,c,g}) = f(a_{d,c,g}) \cdot \alpha_{c,g}$ and $Var(e_{d,c,g}) = \sigma^2_{c,g}$. We then estimate $\alpha_{c,g}$ by employing the maximum likelihood method locally at Block Group g for employment category c. Thus we obtain an estimate of employment size $e_{d,c,g}$ of category c at destination d in Block Group g.

4. Finally, we sum up the employment size in category c in Census Block b in Census Block Group g.

By employing the same local maximum likelihood method described above and using the business establishment survey data (e.g., ESRI Business Analysis package) which is believed to be close to the population POIs, we obtain a benchmark estimate of employment size by category at the Block level for the study area, $E^*_{b,c,g}$. By using the derived POI information (obtained from the machine learning algorithm), we obtain an estimate of employment size by category c at Block b for the study area, $\hat{E}_{b,c,g}$.

Then the mean squared error (MSE), weighted mean squared error (WMSE), and the relative weighted mean squared error (RWMSE) can be calculated to evaluate the goodness of fit of the model (see Equations 1, 2, 3, and 4).

$$MSE(\hat{E}_{b,c,g}, E^*_{b,c,g}) = \sum_{b,c,g} (\hat{E}_{b,c,g} - E^*_{b,c,g})^2 \tag{D.1}$$

$$WMSE(\hat{E}_{b,c,g}, E^*_{b,c,g}) = \sum_{b,c,g} w_{b,c,g} (\hat{E}_{b,c,g} - E^*_{b,c,g})^2 \tag{D.2}$$

$$RWMSE(\hat{E}_{b,c,g}, E^*_{b,c,g}) = \frac{\sum_{b,c,g} w_{b,c,g} (\hat{E}_{b,c,g} - E^*_{b,c,g})^2}{\sum_{b,c,g} w_{b,c,g} (\bar{E}_{b,c,g} - E^*_{b,c,g})^2} \tag{D.3}$$

$$\bar{E}_{b,c,g} = \frac{w'_{b,g} \sum_q E^*_{q,c,g}}{\sum_q w'_{q,g}} \tag{D.4}$$

In Equation 2, when we take the weight $w_{b,c,g} = 1$ for any subscripts b, c, and g, the corresponding WMSE becomes MSE. In Equation 4, $w'_{b,g}$ = area of Block b in Block Group g, and $\bar{E}_{b,c,g}$ is the estimated employment size in Block b of category c, using

the traditional disaggregation approach, assuming that the employment is uniformly distributed across blocks in each Block Group g.

If RWMSE is less than 1, it means that the quality of the derived POIs is reliable, so is the new method; the smaller the RWMSE, the more accurate is the method. If WMSE or RWMSE equals to 0, it means that the derived POIs from the Internet match exactly with the trusted proprietary POIs (treated as the population POIs). However, if RWMSE is greater than 1, it means that the derived POIs cannot well reflect the distribution of the population POIs.

Figures D.3 and D.4 show the estimation results of the disaggregated retail employment density at Block level in Cambridge, MA, by using POIs from infoUSA and Yahoo! respectively. By comparing the estimation results, we find that the disaggregated employment estimations by using the POIs captured from the Internet using Yahoo! and those obtained from the proprietary source (infoUSA 2007) are very close.

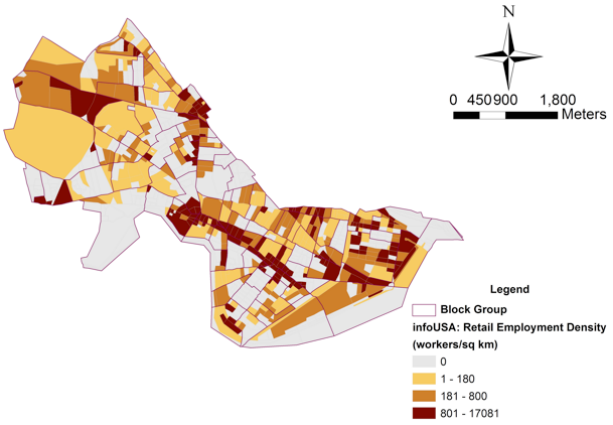

FIGURE D.3: Disaggregated retail employment densities at the Block level, in Cambridge, MA, by using POIs from infoUSA

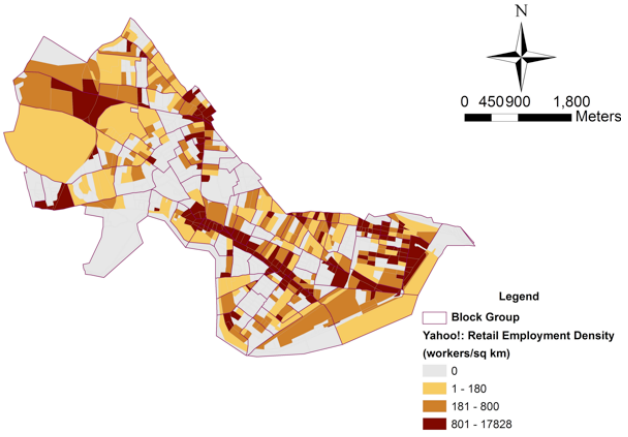

FIGURE D.4: Disaggregated retail employment densities at the Block level, in Cambridge, MA, by using POIs from Yahoo!

Employing Equation 3, the disaggregated employment estimation at the Block level using Yahoo! POI gives RMSE = 0.312. The RMSE is significantly smaller than 1, which means that using the extracted Yahoo! online POIs to estimate the disaggregated employment sizes at the Block level has reduced the mean squared error by around 69% compared to the traditional average disaggregation approach.

Appendix E

North American Industry Classification System (NAICS)

This appendix shows further details about the North American Industry Classification System (NAICS) categories.

Figure E.1 shows the NAICS industry sectors, and figures E.2 and E.3 shows the most common six-digit NAICS codes.

11	Agriculture, Forestry, Fishing & Hunting
21	Mining
22	Utilities
23	Construction
31-33	Manufacturing
42	Wholesale Trade
44-45	Retail Trade
48-49	Transportation & Warehousing
51	Information
52	Finance & Insurance
53	Real Estate & Rental & Leasing
54	Professional, Scientific, & Technical Services
55	Management of Companies & Enterprises
56	Administrative & Support & Waste Management & Remediation Services
61	Educational Services
62	Health Care & Social Assistance
71	Arts, Entertainment, & Recreation
72	Accommodation & Food Services
81	Other Services (except Public Administration)
92	Public Administration

FIGURE E.1: NAICS Industry Sectors

Accommodations
Bed and Breakfast Inns 721191
Boarding Houses 721310
Casino Hotels 721120
Hotels & Motels 721110
Other Lodging 721199
RV (Recreational Vehicle) Parks/Camps 721210

Administrative Support & Waste Management & Remediation Services
Business Service Centers 561430
Call Centers 561420
Carpet/Upholstery Cleaning 561740
Collection Agencies 561440
Credit Bureaus 561450
Document Prep Services 561410
Employment Services 561300
Exterminating Services 561710
Facilities Support Services 561210
Investigation and Security Services 561600
Janitorial Services 561720
Landscaping Services 561730
Office Administrative Services 561110
Other Buildings/Dwellings Services 561790
Other Business Support Services 561490
Other Support Services 561900
Travel Arrangement Services 561500
Waste Management Services 562000

Agriculture, Forestry, Fishing/Hunting
Animal Aquaculture 112510
Animal Production Support 115210
Cattle Feedlots 112112
Cattle Ranching 112111
Crop Production Support 115110
Dairy Cattle / Milk Production 112120
Fishing 114110
Forest Nurseries 113210
Forestry Support 115310
Fruit/Nut Farming 111300
Greenhouses 111400
Hunting/Trapping 114210
Logging 113310
Oilseed/Grain Farming 111100
Other Animal Production 112900
Other Crop Farming 111900
Pig Farming 112210
Poultry/Egg Production 112300
Sheep/Goat Farming 112400
Timber Tract Operations 113310
Vegetable/Melon Farming 111210

Arts, Entertainment and Recreation
Agents/Managers for Athletes/Entertainers 711410
Amusement Parks & Arcades 713100
Gambling Industries 713200
Independent Artists/Writers/Performers 711510
Museums/Historical/Similar Institutions 712100
Other Amusement/Recreation Industries (Gyms) 713900
Performing Arts Companies 711100
Promoters of Arts/Sports/Similar Events 711300
Spectator Sports 711210

Construction
Building Finishing Contractors 238300
Commercial Bldg Construction 236200
Electrical Contractors 238210
Foundation/Structure/Bldg Exterior Contractors 238100
Land Subdivision 237210
Other Building Equipment Contractors 238290
Other Heavy/Civil Engineering Construction . 237990
Other Specialty Trade Contractors 238900
Plumbing/Heating/AC Contractors 238220
Residential Bldg Construction 236110
Road/ Bridge Construction 237310
Utility System Construction 237100

Educational Services
Educational Services 611000

Finance and Insurance
Commercial Banking 522110
Commodity Contracts Brokerage 523140
Commodity Contracts Dealing 523130
Consumer Lending 522291
Credit Card Issuing 522210
Credit Intermediation Activities 522300
Credit Unions 522130
Insurance Agencies & Brokerages 524210
Insurance/Employee Benefit Funds 525100
Insurance/Reinsurance Carriers 524150
International Trade Financing 522293
Investment Banking/Securities Dealing 523110
Life/Health/Medical Insurance/ Reinsurance Carriers 524140
Other Depository Credit Intermediation 522190
Open-End Investment Funds 525910
Other Financial Investment Activities 523900
Other Financial Vehicles 525990
Other Insurance Related Activities 524290
Other Nondepository Credit Intermediation 522298
Real Estate Credit 522292
Real Estate Investment Trusts 525930
Sales Financing 522220
Savings Institutions 522120
Secondary Market Financing 522294
Securities Brokerage 523120
Securities/Commodity Exchanges 523210
Trusts/Estates/Agency Accounts 525920

Health Care and Social Assistance
Child Day Care Services 624410
Chiropractors 621310
Community Food/Housing/Emergency/ Other Relief Services 624200
Dentists 621210
Family Planning Centers 621410
Freestanding Emergency Centers 621493
HMO Medical Centers 621491
Home Health Care Services 621610
Hospitals 622000
Individual/Family Services 624100
Kidney Dialysis Centers 621492
Medical/Diagnostic Labs 621510
Mental Health Practitioners 621330
Mental Health Specialists 621112
Nursing/Residential Care Facilities 623000
Optometrists 621320
Other Health Care Services 621900
Other Misc Health Practitioners 621399
Other Outpatient Care Centers 621498
Outpatient Mental Health/ Substance Abuse Ctrs 621420
Physical/Occupational Therapists 621340
Physicians 621111
Podiatrists 621391
Vocational Rehabilitation Services 624310

Holding Companies
Bank Holding Companies 551111
Other Holding Companies 551112

Information
Book Publishers 511130
Cable/Other Subscription Programming 515210
Data Processing/Hosting/Related Services ... 518210
Directory/Mailing List Publishers 511140
Internet Publishing/Broadcasting 516110
Internet Service Providers 518111
Motion Picture/Video Industries 512100
Newspaper Publishers 511110
Other Information Services 519100
Other Publishers 511190
Periodical Publishers 511120
Radio/Television Broadcasting 515100
Software Publishers 511210
Sound Recording Industries 512200
Telecommunications 517000
Web Search Portals 518112

Manufacturing
Apparel/Footwear/ Leather/Textiles
Apparel Accessories/Other Apparel Mfg 315990
Apparel Knitting Mills 315100
Cut/Sew Apparel Contractors 315210
Footwear Mfg 316210
Leather/Hide Tanning Finishing 316110
Men's & Boys' Apparel Mfg 315220
Other Apparel Mfg 315290
Other Leather/Allied Product Mfg 316990
Textile Mills 313000
Textile Product Mills 314000
Women's & Girls' Apparel Mfg 315230
Beverage & Tobacco Product
Breweries 312120
Distilleries 312140
Soft Drink/Ice Mfg 312110
Tobacco Manufacturing 312200
Wineries 312130
Chemical
Basic Chemical Mfg 325100
Cleaning Materials Mfg 325600
Fertilizer/Other Agricultural Chemical Mfg 325300
Other Chemical Product/Preparation Mfg 325900
Paint, Coating, & Adhesive Mfg 325500
Pharmaceutical Mfg 325410
Resin/Synthetic Rubber/Fibers Mfg 325200
Computer and Electronic Product
Audio/Video Equipment Mfg 334310
Communications Equipment Mfg 334200
Computer/Peripheral Equipment Mfg 334110
Magnetic/Optical Media Mfg 334610
Navigational/Measuring/Electromedical/ Control Instruments Mfg 334500
Other Electronic Component Mfg 334410
Electrical Equipment/Appliance/ Component
Electrical Equipment Mfg 335310
Electric Lighting Equipment Mfg 335100
Household Appliance Mfg 335200
Other Electrical Equipment Mfg 335900
Fabricated Metal Product
Architectural/Structural Metals Mfg 332300
Boiler/Tank/Shipping 332400
Coating/Engraving/Heat Treating 332810
Container Mfg 332510
Cutlery/Handtool Mfg 332210
Forging/Stamping 332110
Hardware Mfg 332610
Other Fabricated Metal Product Mfg 332900
Spring/Wire Product Mfg & Machine Shops 332700
Food
Animal Food Mfg 311110
Animal Processing 311610
Bakeries/Tortilla Mfg 311800
Dairy Product Mfg 311500
Fruit/Vegetable Preserving Mfg 311400
Grain/Oilseed Milling 311200
Other Food Mfg 311900
Seafood Product Prep 311710
Sugar Product Mfg 311300
Machinery
Agriculture/Construct/Mining Machinery Mfg .. 333100
Commercial/Service Industry MachinerY Mfg 333310
Engine/Power Transmission Equip Mfg 333610
HVAC & Comm Refrigeration Equip Mfg 333410
Industrial Machinery Mfg 333200
Metalworking Machinery Mfg 333510
Other General Purpose Machinery Mfg 333900
Miscellaneous
Furniture/Related Product Mfg 337000
Medical Equipment/Supplies Mfg 339110
Other Misc Mfg 339900
Printing/Related Support Activities 323100
Nonmetallic Mineral Product
Cement/Concrete Product Mfg 327300
Clay Product/Refractory Mfg 327100
Glass/Glass Product Mfg 327210
Lime/Gypsum Product Mfg 327400

FIGURE E.2: NAICS codes

Other Nonmetallic Mineral Product Mfg 327900
Paper/Wood Product
Converted Paper Product Mfg 322200
Paper/Pulp Mills 322100
Sawmills/Wood Preservation 321110
Veneer/Plywood Product Mfg 321210
Other Wood Product Mfg 321900
Petroleum/Coal Products
Other Petroleum/Coal Products Mfg 324190
Paving/Roofing Materials Mfg 324120
Petroleum Refineries 324110
Plastics/Rubber
Plastics Product Mfg 326100
Rubber Product Mfg 326200
Primary Metal Product
Aluminum Production/Processing 331310
Foundries 331500
Iron/Steel Mills/Ferroalloy Mfg 331110
Nonferrous Metal Production/Processing 331400
Steel Product from Purchased Steel 331200
Transportation Equipment
Aerospace Product/Parts 336410
Motor Vehicle Body/Trailer Mfg 336210
Motor Vehicle Mfg 336100
Motor Vehicle Parts Mfg 336300
Railroad Rolling Stock Mfg 336510
Ship/Boat Building 336610
Other Transportation Equipment Mfg 336990

Mining
Coal Mining 212110
Metal Ore Mining 212200
Mining Support Activities 213110
Oil/Gas Extraction 211110
Other Mining/Quarrying 212390
Sand/Gravel/Clay Mining/Quarrying 212320
Stone Mining/Quarrying 212310

Other Services
Personal/Laundry Services
Barber Shops 812111
Beauty Salons 812112
Cemeteries/Crematories 812220
Civic/Religious/Similar Organizations 813000
Coin-Operated Laundries/Drycleaners 812310
Dry cleaning/Laundry Services 812320
Funeral Homes/Funeral Services 812210
Linen/Uniform Supply 812330
Nail Salons 812113
Other Personal Care Services 812190
Other Personal Goods Repair 811490
Parking Lots/Garages 812930
Pet Care Services 812910
Photofinishing 812920
Repair/Maintenance
Auto Body/Paint/Interior/Glass Repair 811120
Auto Mechanical & Electrical Repairs 811110
Electronic/Precision Equipment Repair 811210
Equipment Repair of Commercial Machinery .. 811310
Footwear/Leather Goods Repair 811430
Home/Garden Equipment Repair 811410
Other Automotive Repair (washing/waxing) 811190
Other Personal Services 812990
Reupholstery/Furniture Repair 811420

Professional, Scientific and Technical Services
Advertising/Related Services 541800
Architectural Services 541310
Building Inspection Services 541350
Certified Public Accountants 541211
Computer Facilities Mgmt Services 541513
Computer Systems Design Services 541512
Custom Computer Programming Services 541511
Drafting Services 541340
Engineering Services 541330
Geophysical Surveying/Mapping Services 541360
Landscape Architecture Services 541320
Lawyers 541110
Mgmt/Scientific/TechnicaL Consulting Services 541600
Marketing Research/Public Opinion Polling 541910
Other Accounting Services 541219
Other Computer Related Services 541519
Other Legal Services 541190
Other Professional/Scientific/ Technical Services 541990
Payroll Services 541214
Photographic Services 541920
Scientific Research/Development Services .. 541700
Specialized Design Services 541400
Surveying/Mapping Services 541370
Tax Preparation Services 541213
Testing Laboratories 541380
Translation/Interpretation Services 541930
Veterinary Services 541940

Real Estate
Cooperative Housing 531114
Real Estate Agents/Brokers 531210
Real Estate Appraisers 531320
Other Activities Related to Real Estate 531390
Real Estate Property Managers 531310

Rental and Leasing
Auto/Truck/RV Equipment Rental/Leasing 532100
Commercial Equipment Rental (Farm & Medical) 532400
Consumer Electronics/Appliances Rental 532210
Formal Wear/Costume Rental 532220
General Rental Centers 532310
Mini-warehouses & Self-Storag Units Lessors . 531130
Nonfinancial Intangible Assets Lessors 533110
Nonresidential Buildings Lessors 531120
Other Consumer Goods Rental 532290
Other Real Estate Property Lessors 531190
Residential Bldgs/Dwellings Lessors 531110
Video Rental 532230

Retail Trade
Building Material/Garden Equipment
Hardware Stores 444130
Home Centers 444110
Lawn/Garden Equipment Stores 444200
Other Building Material Dealers 444190
Paint/Wallpaper Stores 444120
Clothing/Accessories
Children's Clothing Stores 448130
Clothing Accessories Stores 448150
Family Clothing Stores 448140
Jewelry Stores 448310
Luggage/Leather Goods Stores 448320
Men's Clothing Stores 448110
Other Clothing Stores 448190
Shoe Stores 448210
Women's Clothing Stores 448120
Electronics/Appliance Stores
Camera/Photographic Supplies Stores 443130
Computer/Software Stores 443120
Electronics Stores 443112
Household Appliance Stores 443111
Food/Beverage Stores
Baked Goods Stores 445291
Confectionery/Nut Stores 445292
Convenience Stores 445120
Drinking Places (Alcoholic Beverages) 722410
Full-Service Restaurants/Cafeterias 722110
Grocery Stores 445110
Limited-Service Eating Places 722210
Liquor Stores 445310
Meat Markets 445210
Other Specialty Food Stores 445299
Produce Markets 445230
Seafood Markets 445220
Special Food Services (Caterers) 722300
Fuel
Gas Stations 447100
Heating Oil Dealers 454311
Liquefied Petroleum Gas Dealers 454312
Other Fuel Dealers 454319
Furniture/Home Furnishings
Floor Covering Stores 442210
Furniture Stores 442110
Other Home Furnishings Stores 442299
Window Treatment Stores 442291
Antique Furniture 453310
General Merchandise
Department Stores 452110
Other General Merchandise Stores 452900
Health/Personal Care Stores
Cosmetics/Beauty Supplies Stores 446120
Pharmacies/Drug Stores 446110
Optical Goods Stores 446130
Other Health/Personal Care Stores 446190
Miscellaneous Store Retailers
Art Dealers 453920
Florists 453110
Gift Stores 453220
Office Supplies Stores 453210
Other Miscellaneous Store Retailers 453990
Pet/Pet Supplies Stores 453910
Used Merchandise Stores 453310
Motor Vehicle/Parts Dealers
Auto Parts & Tire Stores 441300
Boat Dealers 441222
Mobile Home Dealers 453930
Motorcycle Dealers 441221
New Car Dealers 441110
Other Motor Vehicle Dealers 441229
Recreational Vehicle Dealers 441210
Used Car Dealers 441120
Nonstore Retailers
E-Shopping/Mail Order Houses 454110
Other Direct Selling Establishments 454390
Vending Machine Operators 454210
Sporting Goods/Hobbies/Book/Music Stores
Book Stores 451211
Hobby/Toy/Game Stores 451120
Musical Instrument/Supplies Stores 451140
News Dealers/Newsstands 451212
Record Stores 451220
Sewing Goods Stores 451130
Sporting Goods Stores 451110

Transportation and Warehousing
Air Transportation 481000
Air Transportation Support 488100
Charter Bus Industry 485510
Couriers 492110
Freight Transportation Arrangement 488510
Local Trucking 484110
Long-distance Trucking 484120
Interurban/Rural Bus Transportation 485210
Limousine Service 485320
Local Messengers/Delivery 492210
Motor Vehicle Towing 488410
Other Passenger Transportation 485990
Other Road Transportation Support 488490
Other Transportation Support 488990
Pipeline Transportation 486000
Rail Transportation 482110
Rail Transportation Support 488210
Scenic/Sightseeing Transportation 487000
School/Employee Bus Transportation 485410
Specialized Freight Trucking 484200
Taxi Service 485310
Urban Transit Systems 485110
Warehousing/Storage 493100
Water Transportation Support 488300
Water Transportation 483000

Utilities
Electric Generation/Transmission/ Distribution 221100
Natural Gas Distribution 221210
Steam/Water/Sewage Systems 221300

Wholesale Trade
Alcoholic Beverages 424800
Apparel/Notions 424300
Business to Business Electronic Markets 425110
Chemical Products 424600
Commercial Equipment/Supplies 423400
Druggists' Sundries 424210
Electronic Goods 423600
Farm Product Raw Materials 424500
Farm Supplies 424910
Furniture/Home Furnishings 423200
Grocery/Related Products 424400
Hardware/Plumbing/Heating Equipment 423700
Jewelry/Watch/Precious Stone/Metals 423940
Lumber/Other Construction Materials 423300
Machinery/Equipment/Supplies 423800
Metal/Mineral (except Petroleum) 423500
Motor Vehicle & Parts/Supplies 423100
Nursery/Florists' Supplies 424930
Other Misc Durable Goods 423990
Other Misc Nondurable Goods 424990
Paint and Supplies 424950
Paper/Paper Products 424100
Petroleum/Petroleum Products 424700
Printed Materials 424920
Recyclable Materials 423930
Sporting/Recreational Goods 423910
Tobacco/Tobacco Products 424940
Toy/Hobby Goods 423920
Wholesale Trade Agents/Brokers 425120

FIGURE E.3: NAICS codes

Bibliography

[1] City of austin: Neighborhood planning: Introduction to land use. February 2010. www.ci.austin.tx.us/zoning/downloads/land_use_guide1.pdf.

[2] D. Rhind and R. Hudson. Land use. *Methuen & Co, New York, USA*, 1980.

[3] R. P. Haining. *Spatial data analysis in the social and environmental sciences*. Cambridge University Press, Cambridge, 1990.

[4] L. Anselin and R. Florax. *New directions in spatial econometrics*. Springer, New York, 1995.

[5] E. Currid and J. Connolly. Patterns of knowledge: The geography of advanced services and the case of art and culture. *Annals of the Association of American Geographers*, 98:414–434, 2008.

[6] P. Sambidi and W. Harrison. Spatial clustering of the u.s. biotech industry. 2006 Annual meeting, July 23-26, Long Beach, CA 21360, American Agricultural Economics Association (New Name 2008: Agricultural and Applied Economics Association), 2006. URL http://ideas.repec.org/p/ags/aaea06/21360.html.

[7] G. Arbia. Modelling the geography of economic activities on a continuous space. *Papers in Regional Science*, 80(4):411–424, 2001. URL http://ideas.repec.org/a/spr/presci/v80y2001i4p411-424.html.

[8] Owl web ontology language: Overview. February 2010. http://www.w3.org/TR/2004/REC-owl-features-20040210/#s1.2.

[9] Y. Kalfoglou and M. Schorlemmer. Ontology mapping: the state of the art. *The Knowledge Engineering Review 18*, pages 1–31, 2003.

[10] Orsi G. Curino, C. and L Tanca. X-som: Ontology mapping and inconsistency resolution. *4th European Semantic Web Conference, Tyrol, Austria*, 2007.

[11] H.-H. Do and E Rahm. Matching large schemas: Approaches and evaluation. *Information Systems*, 32:857–885, 2007.

[12] Mafra-toolkit, February 2010. `http://mafra-toolkit.sourceforge.net`.

[13] N.F. Noy and M.A. Musen. Prompt: Algorithm and tool for automated ontology merging and alignment. *Seventeenth National Conference on Artificial Intelligence (AAAI-2000), Austin, TX*, 2000.

[14] P. Domingos A. Doan, J. Madhavan and A. Halevy. Learning to map between ontologies on the semantic web. *In The Eleventh International WWW Conference, Hawaii, US*, 2002.

[15] D. T. Lindgren. *Land-use Planning and Remote Sensing*. Martinus-Nijhoff, Boston, MA, 1985.

[16] L. O. Fresco. *The Future of the Land – Mobilizing and Integrating Knowledge for Land-use Options*. John Wiley & Sons, Chichester, 1997.

[17] J. B. Campbell. *Mapping the Land – Aerial Imagery for Land use Information*. Association of American Geographers, Washington, D.C., 1983.

[18] P. Danoedoro. Extracting land-use information related to socio-economic function from quickbird imagery: A case study of semarang area, indonesia. *Map Asia 2006*, 2006.

[19] D. Li, K. Di, and D. Li. Land use classification of remote sensing image with gis data based on spatial data mining techniques. *Geo-Spatial Information Science*, 3: 30–35, 2000.

[20] M. Santos and A. Moreira. Automatic classification of location contexts with decision trees. *CSMU-2006 : Proceedings of the Conference on Mobile and Ubiquitous Systems, Guimares, Portugal*, 2006.

[21] T. Griffin, T. Huang, and R. Halverson. Computerized trip classification of gps data. *Proceedings of 3rd International Conference on Cybernetics and Information Technologies, Systems and Applications (CITSA 2006)*, 2006.

[22] J. Pierre. On the automated classification of web sites. *Linkoping Electronic Articles in Computer and Information Science*, 6, 2001.

[23] Naaman M. Nair R. Yang J. Ahern, S. World explorer: Visualizing aggregate data from unstructured text in geo-referenced collections. *International Conference on Digital Libraries, Vancouver, BC, Canada*, 2007.

[24] Smart P. Twaroch, F. and C. Jones. Mining the web to detect place names. *In Workshop On Geographic Information Retrieval, Napa Valley, California, USA*, 2008.

[25] L. Mummidi and Krumm J. Discovering points of interest from users map annotations. *GeoJournal, Volume 72, Numbers 3-4*, 72:215–227, 2008.

[26] Naaman M. Tassa T. Jaffe, A. and M. Davis. Generating summaries and visualization for large collections of geo-referenced photographs. *International Multimedia Conference, Santa Barbara, California, USA*, pages 89–98.

[27] Mexico Proceedings of the IJCAI-2003 Workshop on Information Integration on the Web (IIWeb-03), Acapulco. A comparison of string distance metrics for name-matching tasks. *Communications Of The ACM*, 2003.

[28] North american industry classification system (naics): Introduction. February 2010. `http://www.census.gov/eos/www/naics/`.

[29] Naics association: Frequently asked questions. February 2010. `http://www.naics.com/faq.htm`.

[30] Dun & Bradstreet. D & b website, February 2010. `http://www.dnb.com/`.

[31] W. Cohen, P. Ravikumar, and S. Fienberg. A comparison of string distance metrics for name-matching tasks. *Proceedings of the IJCAI-2003 Workshop on Information Integration on the Web (IIWeb-03), Acapulco, Mexico*, 2003.

[32] J. Gurland and R.C. Tripathi. A simple approximation for unbiased estimation of the standard deviation. *American Statistician*, 25(4):30?32, 1971.

[33] Geoffrey Holmes Bernhard Pfahringer Peter Reutemann Ian H. Witten Mark Hall, Eibe Frank. The weka data mining software: An update. *SIGKDD Explorations*, 11, 2009.

[34] Ian H. Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques, 2nd Edition*. Morgan Kaufmann, 2005.

[35] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

[36] A. Alves, F. C. Pereira, A. Biderman, and C. Ratti. Place enrichment by mining the web. In *Proceedings of the Third European Conference on Ambient Intelligence*, 2009.

[37] George A. Miller. Wordnet: A lexical database for english. *Communications Of The ACM*, 38:39–41, 1995.

[38] M.G. McNally and C.R. Rindt. *The Activity-Based Approach*. Handbook of Transportation Modeling. Elsevier, Amsterdam, London, 2008.