Extraction, Integration and Exploration of Crowdsourced Geospatial Content from Multiple Web Sources

George Lamprianidis
Institute for the Management of Information Systems
R.C. ATHENA, Greece
glampr@imis.athena-innovation.gr

Dimitrios Skoutas
Institute for the Management of Information Systems
R.C. ATHENA, Greece
dskoutas@imis.athena-innovation.gr

George Papatheodorou
Institute for the Management of Information Systems
R.C. ATHENA, Greece
papatheodorou@imis.athena-innovation.gr

Dieter Pfoser
Department of Geography and GeoInformation Science
George Mason University, U.S.A.
dpfoser@gmu.edu

ABSTRACT

Our work focuses around a Web application that retrieves user-generated geospatial content from multiple popular Web sources, and applies schema mapping and entity matching techniques to obtain an integrated dataset. Moreover, density-based clustering of the obtained data is performed to reveal areas of interest for various data categories. An analysis and overview of the underlying data are also provided by computing various statistics that are visualized in a series of charts. Further data exploration and navigation is enabled via keyword search and faceted browsing. This demonstration covers all the steps of the process, from selecting an area and the sources for data collection, to visualizing and navigating the integrated results.

Categories and Subject Descriptors

H.2.8 [**DATABASE MANAGEMENT**]: Database Applications—*Data mining*

General Terms

Algorithms, Experimentation, Performance

Keywords

points of interest, crowdsourced geospatial data, data extraction and integration

SIGSPATIAL '14, Nov 04-07 2014, Dallas/Fort Worth, TX, USA
ACM 978-1-4503-3131-9/14/11.
http://dx.doi.org/10.1145/2666310.2666367

1. INTRODUCTION

The amount of user-generated geospatial content on the Web is constantly increasing, making it a valuable source for enriching and enhancing the content of geospatial services and applications. However, as the content increases, so do the numerous available data providers, making it also increasingly difficult to create and maintain a comprehensive and consolidated dataset comprising such content. Furthermore, the raw content created by users on the Web and social networks is highly heterogeneous and varies significantly in quality and accuracy. Extracting, integrating, and mining this crowdsourced geospatial content is far from trivial.

For instance, Wikipedia includes, among others, information about places and Points of Interest (POIs). Inspired by its success, OpenStreetMap and Wikimapia have also emerged as collaborative mapping projects, both having a user base that exceeds a million users, resulting in a large pool of crowdsourced geospatial data. An additional abundance of geospatial entities can be obtained from various other services (e.g. Google Places), from user check-ins in social networks (e.g. Foursquare), from Web sites providing information about events (e.g. Eventful), etc. This new wealth of sources and content opens up new opportunities for improving, enriching and enhancing applications and services in the geospatial domain, such as location-based services or trip planning.

The system we built is so far able to collect four different types of user-generated content from a total of ten sources:

- Venues (POI information) from DBpedia, Wikimapia, OpenStreetMap, Foursquare and Google Places.
- Photos from Flickr and Panoramio, as well as photos of venues from the above sources, when available.
- Events, and their venues, from LastFM and Eventful.
- Text messages from Twitter.

Once the data have been collected, as described in Section 2, we process it in a variety of ways. Processing focuses mainly on the venue type data, since it is more heterogeneous than the rest. In particular, we map venue categories

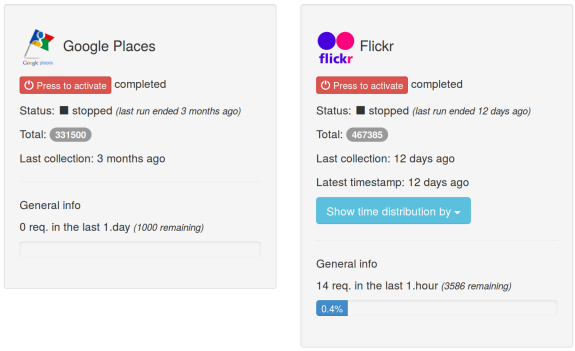

Figure 1: Data collection panel.

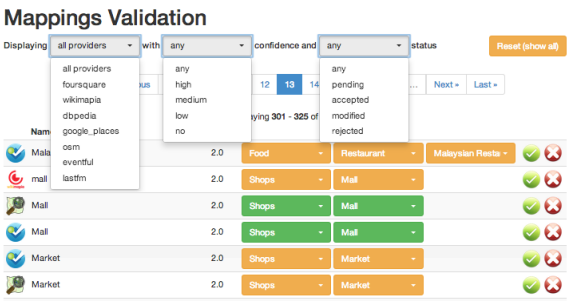

Figure 2: Category mapping panel.

assigned by each respective source to a common schema, as described in Section 3, and we perform entity matching to identify duplicate POIs across sources, as described in Section 4. All types of available data undertake also a clustering process to detect regions of interest, as outlined in Section 5. Finally, to make the entire dataset available to users and other applications, we provide search endpoints via a Web page and an API, as shown in Section 6.

2. DATA COLLECTION

To initiate the data collection process, the user first specifies a targeted geographical area. This can be done visually, by drawing a rectangle on an interactive online map. Then, a panel is presented, where the user can control and monitor the collection process for each of the available sources (Fig. 1). This panel displays the status and progress of each task, the amount of data collected so far, and, where applicable, the temporal and spatial distribution of the collected data. The latter is presented, respectively, either via a time plot, with configurable time granularity (day, month, week or quarter) or via a map where the user can choose the spatial granularity (based on geohash length and map zoom level). The available usage quota for each of the data providers is also displayed to allow for more effective scheduling of the collection.

The collection process is handled by a group of download clients, each specifically designed to handle each source's particularities (i.e., different APIs, quotas on number of requests per time period and number of results per request), but all adhering to a similar workflow. This workflow promotes parallelization, as any area selected by the user can be dissected into smaller parts, each one assignable to a separate process.

Initially, the given area is split into four quadrants. Each quadrant is inserted into a queue, to be processed by available worker processes. When a worker dequeues a bounding box b, if $b.max\_edge > edge_{MAX}$, b is again split into four quadrants which are re-inserted into the queue; otherwise, the corresponding download client is triggered to fetch data from the respective source. The amount of data gathered ($b.data\_count$) is then compared to the maximum number the provider can return ($data_{MAX}$) for that type of query. If $b.data\_count < data_{MAX}$ the bounding box is marked as "completed"; otherwise, the splitting process is once more repeated. To avoid endlessly dissecting bounding boxes into extremely small quadrants, the split stops if $b.max\_edge$ drops below a specified threshold $edge_{MIN}$. The collection process completes when the queue is empty.

3. CATEGORY CONSOLIDATION

The first step towards integrating the data collected from multiple sources is to map categories (i.e., POI types) specified by the original sources to a common classification. For this purpose, we have defined an internal category hierarchy, similar to that of Foursquare, since this was found to have a broader scope and higher variety of POI types compared to the other sources. To keep this categorization more comprehensive, we also restricted it to up to three levels of depth. The top level categories include the following POI types: *Entertainment* (e.g. music venue, nightlife spot, movie theater), *Culture* (e.g. opera house, art gallery, museum), *Education* (e.g. college, university, library), *Food* (e.g. restaurant, coffee shop, breakfast spot), *Places* (e.g. beach, forest, lake), *Shops* (e.g. souvenirs, bookstore, clothing store), *Services* (e.g. postbox, ATM, bike rental) *Professional* (e.g. company, office, convention center), *Travel and Transport* (e.g. airport, subway station, hotel), *Athletics and Sports* (e.g. gym, stadium, tennis court), and *Religion* (e.g. temple, shrine, church).

Finding category mappings is done semi-automatically. First, we use a state-of-the-art tool, the S-Match semantic matching framework [4], to automatically compute candidate mappings between the source and target categories. S-Match is a Java library that implements several semantic matching algorithms. Given two graph-like structures (e.g., taxonomies, database/XML schemas, ontologies), it identifies nodes in the two structures that semantically correspond to each other. This is done by applying various matching rules, and exploiting external resources, such as WordNet[1]. Next, we apply a post-process filtering step to assign a score to each of the candidate mappings for each source category, in order to eventually select the best one. These scores are computed by string similarity measures comparing the two categories in a candidate mapping. Specifically, we compute the Levenshtein Distance of the category names, using the implementation provided in the SimMetrics library[2]. Thus, the steps of the category mapping process can be summarized as follows: (1) transform source and target categories to a common XML format, and provide as input to S-Match; (2) compute candidate mappings with S-Match; (3) compute scores to filter candidate mappings; (4) output the mapping with the highest score for each source category.

[1] `http://wordnet.princeton.edu/`
[2] `http://www.aktors.org/technologies/simmetrics/`

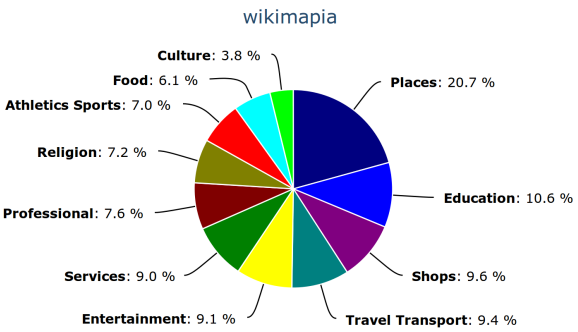

Figure 3: Category distribution chart.

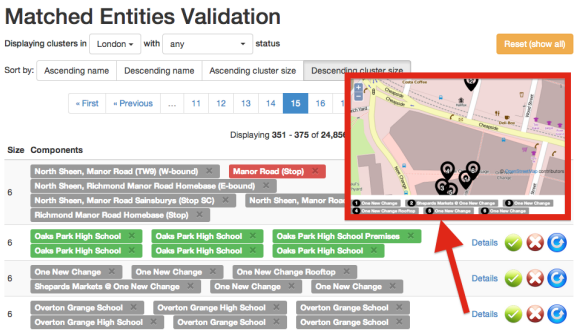

Figure 4: Entity matching panel.

Once this automatic process is completed, the results are displayed on the category mapping panel (Fig. 2) to allow for manual validation by the user. This panel allows the user to navigate through the automatically computed mappings, by sorting and filtering the results in various ways, e.g. according to the original source or according to the matching score. The displayed mappings are color-coded according to their status (pending validation, accepted, modified, rejected) to further ease navigation. Thus, the user can decide to accept, reject or modify the proposed mappings. The results are then updated in the system accordingly.

Furthermore, the tool also reports useful statistics about the mapping process results. One is the percentage of each source's content (throughout all downloaded areas) that was mapped with a certain score; for example, 77.62% of data from Foursquare were mapped with "high" confidence to the common classification, while 34.28% of data from OpenStreetMap were mapped with "medium" confidence. Another one is the distribution of categories per provider and geographical area. For example, in London, from the content collected from Wikimapia, 9.4% was classified under *travel and transport*, while 9.1% was classified under *entertainment*, as shown in Fig. 3.

4. DUPLICATE DETECTION

The next step in the integration process is to match entities collected from the different sources in order to identify (and subsequently remove or fuse) duplicates. This problem arises from the fact that often the same entities, especially popular POIs, appear in many sources, perhaps with different, complementary or sometimes even conflicting representations. Similar to the problem of category mappings addressed in the previous section, this is again an important, recurring and widely studied problem in the area of information integration, appearing in the literature under various terms, such as entity matching, record linkage, data deduplication, etc. [2].

Finding matching entities across sources, or even within the same source, is important for various reasons. First, different sources may often provide complementary information for the same entity; thus identifying matching entities allows to build a richer and more complete entity profile (e.g., finding photos from one source, comments and ratings from another, etc.). Second, it can help to detect, and possibly even correct, errors, in the case that conflicting information for the same entity is found in various sources. Automatic correction can be achieved when fusion rules are available, e.g. based on an assessment of the quality and trustworthiness of each source, such as "always prefer values obtained from Wikipedia over other sources", or based on a voting scheme, e.g. keep the value provided by most sources (if found in more than two). Third, having identified duplicates is useful when users search and browse the available content, since allowing duplicates in the search results can easily deteriorate the user experience and cause frustration.

The challenge here, as in the case of reconciling different category hierarchies, arises from the fact that there are no unique, common identifiers for the entities used by all sources. Moreover, the name of an entity may appear with slight variations, or even misspellings, in different sources. Thus, the typical approach is to define some measure of similarity between entities, and then consider as matches the cases where this similarity exceeds a specified threshold.

In our case, since we are dealing with geospatial entities, we take both their spatial and semantic information into account to identify matches. More specifically, we assume that two collected POIs P_i and P_j correspond to the same real-world POI if the following conditions hold: (a) the coordinates of P_i and P_j do not defer by more than a specified threshold ε; (b) the name similarity of P_i and P_j is above a specified threshold σ; and (c) P_i and P_j belong to the same top level category. The tool applies these conditions to identify matching entities. For efficiency, it splits the whole area into cells and only compares nearby entities according to the parameter ε. In the future, we plan to further optimize this step based on the algorithms presented in [1] for spatio-textual similarity joins.

Finally, as in the case of category mappings, the tool presents the discovered duplicates to the user for validation. The entity matching panel (Fig. 4) displays the names of the POIs that have been matched according to the above conditions, showing also their locations on the map. Subsequently, the user can choose to accept or reject entities marked as potential duplicates.

5. REGIONS OF INTEREST

A further analysis task is to identify emerging *regions of interest*, i.e. geographical areas with high density of POIs of certain categories. This has various applications. For example, travel guides can use this information to identify areas with many museums, sights or other attractions; marketing firms can use it to better target their audience in specific areas; entrepreneurs and businesses can get an overview of which areas are well served and which suffer, to decide where

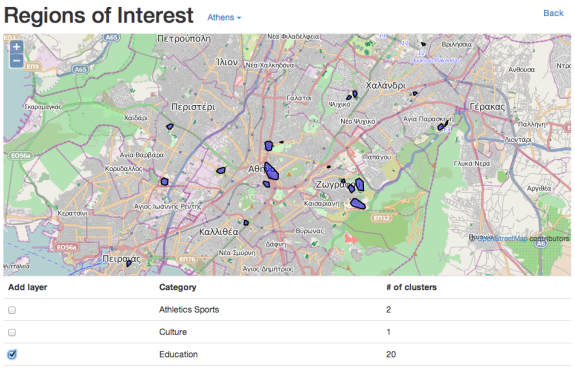

Figure 5: Regions of Interest panel.

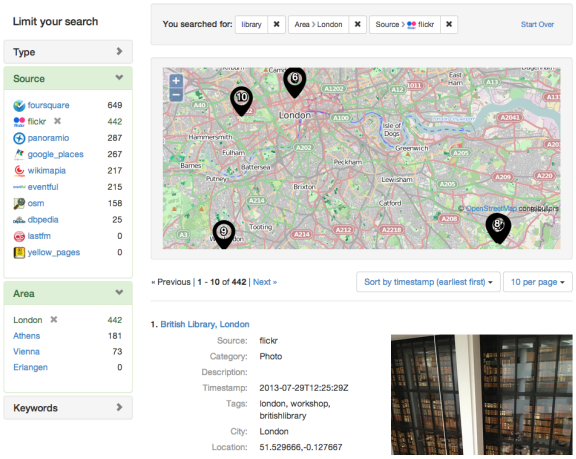

Figure 6: Search and browsing page.

to expand their type of business next.

Figure 5 shows a page where the user can select a category (e.g. education) and display on an interactive map the regions of interest found by the system for this category. These regions are computed via clustering and their shape is defined as the convex hull of the points in the cluster.

Since the main concept for a region of interest is having high density, the underlying clustering process is based on DBSCAN [3], one of the most known and used density-based clustering methods. Computation of clusters is controlled by two parameters, namely ε, which specifies a distance threshold for determining the neighborhood of a point, and $minPts$, which specifies a threshold for the number of neighbors needed to characterize the neighborhood of a point as dense. Based on these, it is defined whether one point is *density reachable* by another point. Clusters of points are thus formed based on this notion of density reachability between points. In our case, we employ an additional constraint that only points belonging to the category specified by the user should be considered.

Similarly to the case of entity matching, to avoid a pairwise comparison of all the POIs in a large area, the system splits the input area into cells, according to the parameter ε, and only considers neighboring cells when retrieving neighbors and calculating densities. Moreover, cells that do not contain more than $minPts$ points, need not be considered as starting points for the formation of clusters, and can be skipped.

6. SEARCH AND BROWSING

The system finally employs a page where the user can search and navigate the data collection via keyword search and faceted browsing. The search is powered by the open source Apache Solr platform[3], which offers powerful full-text search with facet support. The integration with the Web application is managed by the Blacklight Rails engine[4].

During collection and integration, the data are stored in a PostGIS database, and then get imported into Solr and indexed. In addition to tokenization and stemming for full-text search, a spatial index on point coordinates is also built, to allow for efficient evaluation of spatial queries. Thus, Solr is used for all searches, which provides faster response times. Other advantages include assigning different weights to various fields, configuring the criteria for ranking the search results, as well as defining which fields should be used as facets when browsing.

Through the search page (Fig. 6) the user can issue a keyword query to retrieve relevant results. The search can be done on all fields of a resource (label, description, tags) or can be restricted to specific ones. The user can also choose to search only for POIs, photos or events. The results can be further filtered via the available facets, which include the source of origin, the geographical area, the tags, and the category. The search results are displayed as a list, showing some basic information for each one, as well as on a map that shows their locations. Clicking on a specific result, navigates to a page displaying its full information, including also a link to the original source from which this result was retrieved. Thus, the origin of the data can be traced back to the original sources.

We are currently working on more advanced functions to rank and summarize search results, as well as on advanced search and navigation for regions of interest.

[3] `http://lucene.apache.org/solr/`
[4] `http://projectblacklight.org/`

Acknowledgements

This work was partially supported by the EU FP7 Project GEOSTREAM (FP7-SME-2012-315631). Dieter Pfoser's work was partially supported by NGA NURI grant HM02101410004.

7. REFERENCES

[1] P. Bouros, S. Ge, and N. Mamoulis. Spatio-textual similarity joins. *PVLDB*, 6(1):1–12, 2012.

[2] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios. Duplicate record detection: A survey. *IEEE Trans. Knowl. Data Eng.*, 19(1):1–16, 2007.

[3] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, pages 226–231, 1996.

[4] P. Shvaiko, F. Giunchiglia, and M. Yatskevich. Semantic matching with s-match. In *Semantic Web Information Management*, pages 183–202. 2009.