MoreData: A Geospatial Data Enrichment Framework

Leonardo J. A. S. Figueiredo
Federal University of Viçosa
Viçosa, Brazil
leonardo.j.alves@ufv.br

Germano B. dos Santos
Federal University of Viçosa
Florestal, Brazil
germaino.santos@ufv.br

Raissa P. P. M. Souza
Federal University of Viçosa
Viçosa, Brazil
raissa.polyanna@ufv.br

Fabrício A. Silva
Federal University of Viçosa
Florestal, Brazil
fabricio.asilva@ufv.br

Thais R. M. Braga Silva
Federal University of Viçosa
Florestal, Brazil
thais.braga@ufv.br

ABSTRACT

In recent years, we are facing a significant increase in the collection and availability of geospatial data. This type of data is paramount to help decision makers in different contexts, such as smart cities, mobile social networks, and e-commerce. In addition, the behavior of mobile users can be extracted and exploited to improve the quality of the services offered by mobile providers. However, raw location data needs to be semantically enriched to be useful, which is an arduous task because it requires queries on different data sources of different formats and complex joins. To mitigate this difficulty, this work proposes MoreData, a flexible and expandable framework for the semantic enrichment of geospatial data. The framework accepts different input formats, provides connectors to different sources (APIs, Relational Database, Fast Search Engine Tool, and Open Street Map), and is easy to extend to new sources. To evaluate the framework’s benefits, we use it to enrich a real database comprised of thousands of mobile users’ locations with data. We observe that the adoption of the framework leads to less time and effort in the data enrichment process, saving time for the more important tasks, such as building and analyzing models.

CCS CONCEPTS

• Information systems → Mediators and data integration; Location based services.

KEYWORDS

geospatial data, semantic enrichment, framework

ACM Reference Format:

In recent years, there has been a considerable increase in the collection and availability of geospatial data as companies and the scientific community are realizing the importance of this type of data to help on decision making, the demand for solutions that make use of geospatial data has attracted the attention of companies and researchers. Geospatial data is beneficial to different contexts, such as smart cities, this type of data helps on the understanding of urban mobility and in the formulation of public policies. In mobile social networks, geolocated behavior patterns can be explored to offer better advertising. E-commerce companies can learn the interests of their customers according to the characteristics of the region where they live and visit. In general, with geospatial data, it is possible to extract knowledge that leads to a better understanding of mobile users from different scenarios.

However, raw geospatial data, often represented as the tuple $\langle$latitude, longitude$\rangle$, does not add value to the business problem itself. For this reason, it is necessary to enrich the raw data with external sources, usually from different standards and formats, in order to extract the semantic information inherent to a particular location. For example, it is more useful to recognize that a mobile user frequently visits a Shopping Mall than simply pointing out the user locations without semantic knowledge. The enrichment of geospatial data is usually a manual and expensive task, which requires specific expertise of the analysts and depends on the access standard and format of the external sources.

To mitigate this difficulty, we propose a flexible framework, called MoreData, to carry out the enrichment of geospatial data with different sources, besides being easily expandable. The framework aims at joining the original raw location data with external sources with minimum efforts from the analyst. Thus, given a location represented by a tuple (latitude, longitude) or by any identification key such as (ZipCode), the framework matches the original data with data from other sources (e.g., census data or any available API), according to the user’s interests, returning an enriched dataset.

MoreData provides connectors for data coming from APIs, relational databases, the fast search engine ElasticSearch, and the collaborative base OpenStreetMap. Besides, MoreData is easily expandable so the user can easily create new connectors. To assess

1https://github.com/gegen07/more-data

2http://elastic.co

3https://www.openstreetmap.org/
the benefits of the framework, we used real data from 34,428 mobile users, containing their points of interest, which we enriched with the locations’ category using the Open Street Map connector. It was possible to observe that the framework significantly facilitated the enrichment effort, reducing the time spent by analysts on this task. Therefore, analysts can dedicate most of their efforts on more relevant tasks to data-driven projects, such as knowledge extraction.

Until now, we have found no studies to help enriching geospatial data from different sources. We found some frameworks to manipulate data for specific purposes different from geospatial enrichment Hung et al. [3], Li et al. [4]. Other studies found in the literature manipulate different data formats such as KML, XML and RDP Maynard and Bontcheva [5], Sardianos et al. [8], Varlamis et al. [9], which reinforces the need for solutions that deal with different data formats. Finally, many studies, such as Farrahi and Gatica-Perez [2], McInerney et al. [6], proposed solutions that involve the use of enriched geospatial data and would benefit from our framework. Thus, we believe that MoreData will be useful for the research community and companies. Unlike other works, the framework presented is capable of being easily extended to new formats and sources. In addition to the already developed formats, it is simple to add new ones, as it only requires the implementation of the conversion to a single format (JSON).

We describe the framework in Section 2. We present a real case study of the framework in Section 3. Finally, we draw conclusions and future works in Section 4.

2 MOREDATA

The objective of the proposed framework is to help data analysts to enrich geospatial data with external sources. Given a raw location, the objective of an enrichment process is to join the raw data with an external source, such as a list of geospatial geometries, adding new attributes to it. For example, given a $\langle\text{latitude}=19.9746, \text{longitude}=-43.9438\rangle$, an enriched process would return $\langle\text{latitude}=19.9746, \text{longitude}=-43.9438, \text{category}=\text{Shopping Mall}\rangle$. During the enrichment step, it is possible to add a suffix to each new attribute to indicate the source of the data. In addition, it is easy to add a timestamp attribute. It is necessary to update the data source and enrich the entire database again to update a Point of Interest since re-enrichment is more efficient than the cost of checking changes.

The MoreData comprises two main modules: the Enricher and the Converter. It was developed using Python language and libraries like geopandas and overpass. Figure 1 presents an overview of its components.

2.1 Enricher

The Enricher is the core component of the framework. It comprises two major modules: Connector, responsible for connecting with an external data source; and Constructor, responsible for performing the enrichment itself. Each connector must implement the interface IEnricherConnector, as shown in Figure 2. The use of the Strategy design pattern allows high extensibility of the framework. So, it is possible to create other connectors that implement an interface and use them in the Enricher class, thus creating new concomitant data sources according to the user’s need. In other words, new connectors can be created by inheriting from the IEnricherConnector interface and implementing its abstract methods.

The current version of MoreData has four connectors implemented and tested, as described in the following:

2.1.1 ElasticSearch Connector. ElasticSearch is a fast-search base widely adopted in the last years. The connector to this source uses two JavaScript Object Notation (JSON) configuration structures: pipeline, responsible for defining which attribute of the index will be enriched and which will be used as key to the relationship; and policy, which defines the use of a field to enrich an Elastic Search index.
During the definition of the pipeline and policy structures, it is possible to indicate which relationship between the indexes will make the enrichment, whether a geospatial query or an ordinary one. The first case offers the possibility of using four operations, provided by ElasticSearch, between geospatial geometries: intersection, disjoint, contains, and within; the second case allows a query that joins the original and the ElasticSearch data with conditions to match specific keys indicated by the user.

After defining the fields, relationships, and which type of query for the enrichment, the framework performs the re-indexing process using the pipeline created.

2.1.2 API Connector. This connector executes Hyper Text Transfer Protocol (HTTP) requests from a Uniform Resource Identifier (URI) defined by the user to collect necessary information for the association of the API data with the original data. A cache is used to store the returned values, speeding up the enrichment process in future requests.

The user must implement the method to process the returned data by the API, allowing different treatments on what to enrich and how to enrich, in which fields to store the results.

2.1.3 OSM Connector. The OSM connector collects data from the Open Street Map API. There is also the possibility to use the framework only to enrich data from previously collected data from OSM, without the need to download new data at the moment of the enrichment. This is a good choice when many queries are expected to run assuming data from the same geospatial region. After obtaining the data from OSM, the connector locally indexes the elements returned in a R-Tree to improve the time of the geospatial search.

The local search can be performed in two ways: from a radius provided by the user or by the pertinence relationship between a location and the geospatial geometries of OSM elements. In the first mode, it makes a projection change, from the World Geodetic System to the Azimuth Projection, for a better representation of the resulting circle and use of the meter as a unit of length. After changing the projection and transforming the point into a circle, the connector enriches the original data with the geospatial geometry intersected by the circle and its data.

In the second option, the framework makes a check of the pertinence relationship between the polygon collected from the OSM and a location in the original data. Then, the connector enriches the original data with the corresponding geospatial geometry and its attributes.

2.1.4 Relational Database Connector. This connector connects to a relational database, offering some possibilities of use through the SQLAlchemy\(^4\) library. Therefore, it allows the use of different systems such as MySQL, PostgreSQL, SQLite, among others that are supported by SQLAlchemy.

After connecting to the database, a query written by the user is executed, returning the result set that is used to enrich the original data. To perform this query, MoreData uses a column belonging to the user-defined table and an attribute from the source to join both datasets. After the query, the connector creates new attributes with the names defined by the user in the original data.

2.2 Constructor

The Constructor component implements the builder design pattern to use all available connectors created through the Enricher class and compose the data enrichment using the EnricherBuilder class, as shown in Figure 3. With the use of this design pattern, it is possible to enrich a raw data with different connectors, enabling a combination while using these structures.

![Figure 3: Implementation of the Construction Module](image)

2.3 Converter

MoreData uses the JSON format throughout the enrichment as a key to maintainability. However, from this module, it is possible to provide files from other formats, such as CSV and Parquet, as input to the framework. Also, this module offers output options that can be of CSV, JSON, and Parquet types. It is worth noting that the converter is also easily expandable to new formats of interest to the user. Furthermore, the converter also has the option of separating the output into multiple smaller files to reduce the problems faced when working with large files.

3 DEMONSTRATION

We tested MoreData using a dataset composed of location records of 34,428 real mobile users of the Android operating system, got under an non-disclosure agreement from a mobile services provider company. The data contains 178,877 distinct points of interest (POIs) of these users. A PoI is represented by its center location \(\text{PoI}_i = (\text{lat}_i, \text{lng}_i)\). The objective of this demonstration is to come up with the category \(\text{cat}_i\) of \(\text{PoI}_i\) using the OSM enricher, such as \(\text{PoI}_i = (\text{lat}_i, \text{lng}_i, \text{cat}_i)\).

Initially, we used MoreData to convert the original base from CSV format to JSON, and then enrich it using the OSM connector. Since geographically small business places, such as pubs and bakeries, are affected by locations with low accuracy, we choose only four

\(^4\)https://www.sqlalchemy.org/
different categories of OSM places: Aerodrome, Hospital, Shopping Mall, and University. In total, 7,540 places were collected to cover the entire original PoI locations. The OSM connector was very useful and the enrichment was configured and executed very easily. As results, we found out that many users have PoIs located at one of these places.

```python
from moredata import Data,
    Enricher,
    OSMPlacesConnector,
    parse_document

import moredata.utils.util as util

users = Data(data_file=USER_DATA,
    parser_func=parse_document,
    data_type="json")

osm_enricher = Enricher(
    connector=OSMPlacesConnector(
        key="shop",
        value="mall",
        radius=100,
        dict_keys=["points_of_interest"],
        geometry_intersected=True)
)

user_enriched = osm_enricher.enrich(users)

util.write_json_generator_to_json(
    OUTPUT_PATH,
    user_enriched,
    100000)
```

Listing 1: MoreData code example

The listing above is the use of MoreData to enrich a PoI with Shopping Mall polygons within 100 meters as a radius. Thereby it demonstrates the simple way to enrich location using the developed tool. On line 9, the data is set. The USER_DATA constant is the data directory location. Besides that, the file is assumed to be a json file already, thus, without the need to use the converter module. On line 12, it is specified what and how it will enrich. For that enrichment it used a specific OSMConnector as showed in figure 2. This connector returns the geometry of a Shopping Mall which intersects the PoI with radius. On lines 21 and 23, it is executed the enrichment and saved the json file as a result of enrichment, respectively. The figure 4 represents the result of enrichment of a user.

Figure 4: PoI location enriched with Shopping Mall

Usually, data-driven projects are known to fail and to require long periods of time to prepare and enrich the original dataset. Therefore, a framework such as MoreData is very useful to reduce this time and, consequently, increase the chance of success of the project. We could observe this in the case study, once the time to enrich the original location data records with their corresponding categories from OSM was straightforward and fast. In addition to this particular case, we have also tested all other connectors under different scenarios. All these scenarios will be covered in the demonstration presentation.

4 CONCLUSIONS AND FUTURE WORKS

We proposed a framework capable of assisting the semantic enrichment of geospatial data. MoreData was essential to streamline and simplify the enrichment process in the chosen use case. Also, during the use of the framework, we perceived its adaptation to lead with the gathering of different data. With this, it is possible to verify the easy applicability of the framework in different contexts to enrich raw data according to the needs of the users.

This is an ongoing work with many open opportunities. First, we plan to parallelize the enrichment process to improve the framework’s performance. Next, we will implement more advanced enrichment methods, such as detecting the functional area of a region, identify the users’ activities in particular regions, and compute mobility characteristics of trajectory data.

ACKNOWLEDGMENTS

We thank CAPES, CNPq and Fapemig for their financial support.

REFERENCES