Geospatial webmining for emergency management

Christian Fritz[1], Christian Kirschner[1], Daniel Reker[1], Andre Wisplinghoff[1], Heiko Paulheim[2], Florian Probst[2]
[1] Technische Universitaet Darmstadt
{c_fritz,c_k,d_reker,andre_w}@rbg.informatik.tu-darmstadt.de
[2] SAP Research
{heiko.paulheim,f.probst}@sap.com

1 Introduction

Emergency management is a domain where information has to be gathered, aggregated, and visualized dynamically and quickly. By providing the right information at the right time, the chaos phase between the occurrence of a disaster and the start of well-organized relief measures can be significantly shortened [1].
The information needed in an emergency scenario can be quite diverse. For example, a person planning an evacuation may need to know about companies that can transport people, and places that can serve as emergency shelters. For the first, bus and taxi companies, logistics companies as well as rental car providers may be taken into account. The latter may include hotels and schools as well as sports arenas and concert venues.
Although all this information is available on the web, it cannot be easily accessed. Since such non-trivial categories such as "buildings that can serve as emergency shelter" are not sharply defined, one cannot simply enter "emergency shelter" into Google and retrieve a list of emergency shelters. Instead, lots of subsequent manual searches have to be performed, and the results have to be aggregated by hand.
While several emergency management tools exist (cf. [5] for a survey), this concern has not been addressed yet. In this paper, we introduce a prototype which allows for crawling the web for relevant information on objects belonging to non-trivial categories and provide the aggregated results as an OGC compliant web feature service.

2 Concept

Our concept consists of five steps (see Figure 1). In a preparation step, a list of sources is manually defined, whereas sources are web catalogues containing a distinct page for each listed object. We identify relevant information by browsing the pages of this predefined list (step 1). The information is extracted (step 2) and processed so that it can be stored in a database (step 3). Finally the information is provided in a standardized format (step 4). We will consider these steps in detail in the following.
We implemented a prototype in Java that mines specific websites and extracts information about buildings, which can be potentially used as emergency accommodations and hospitals in Germany, and stores them in a database.

2.1 Step 0: Preparation

In our approach we do not gather data from the whole web, but limit our search to a set of specific domains. Such strict subsets of the data can provide almost all relevant information for a given purpose with less interference.

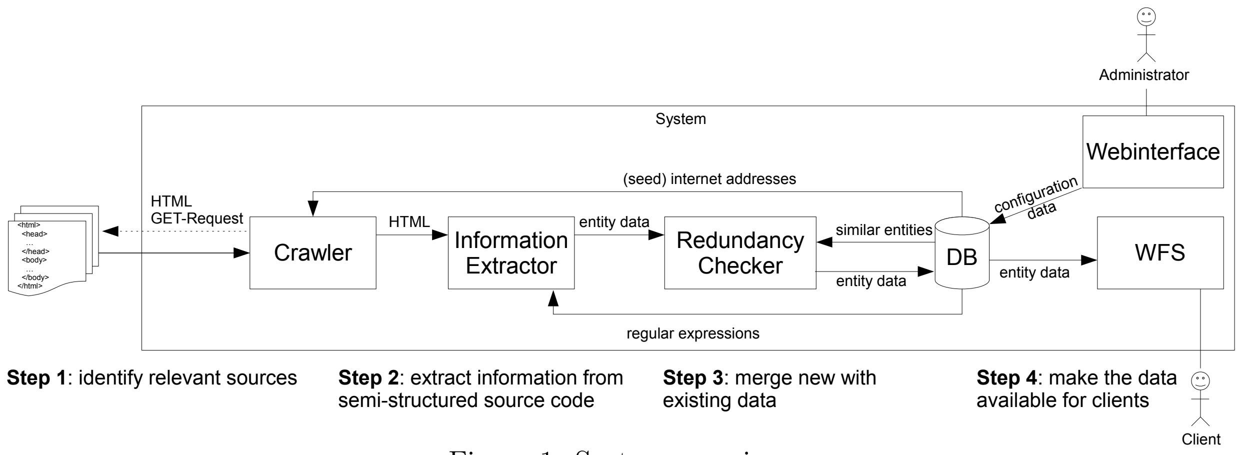
Administrator
System
Webinterface
(seed) internet addresses
HTML GET-Request
HTML
entity data
similar entities
configuration data
Crawler
Information Extractor
Redundancy Checker
DB
WFS
entity data
entity data
regular expressions
Step 1: identify relevant sources
Step 2: extract information from semi-structured source code
Step 3: merge new with existing data
Step 4: make the data available for clients
Client

Figure 1: System overview

We are aware of the impact of this limitation, as a manual selection of data sources is needed, which carries the risk to leave out relevant information. Since we direct our attention on the correctness of mined data, the precision of the data is more important than an outstanding recall. Regarding our prototype, we needed to decide on a set of websites that should be crawled. For hospitals, specific sources are available, but emergency accommodations are not clearly defined, so multiple concepts, each with its own sources, have to be combined.

2.2 Step 1: Fetching relevant documents

Following [4], we use breadth-first search for crawling the sources. Multiple domains can be crawled at once to speed up the process.

2.3 Step 2: Extracting Information

We assume that within one heterogeneous part of a source, entities that contain the same type of information are saved in the same, semi-structured way. Provided that there are textual characteristics, which identify the required information, we have chosen regular expressions [2] as our means for information extraction. They represent an efficient way to find predefined patterns in text. By this we can concentrate on recurring patterns and ignore irrelevant parts.
While using regular expressions is a simple and straightforward approach, patterns cannot be detected flexibly and dynamically. However, since the data sources are limited in our approach, the one-time effort of writing the expressions is feasible. For each attribute and source one regular expression is needed.
It is possible to use additional services to complete the data, mainly if we can deduce new information from already found data. If our prototype finds no geographic coordinate in a source, the Google Geocoding API is used to compute the missing information using the address. Missing phone numbers are added as well, using a German online telephone directory service (http://dasoertliche.de).

2.4 Step 3: Merging retrieved data

In our approach we extract information from different sources. Consequently, we have to ensure that no duplicates are inserted into our database, and that different incomplete information on one entry are merged to a more complete one. Therefore, a strategy is needed to detect whether two entities are equal. If we find an object and identify it as equal to another object found before, we add the missing attributes and replace attribute values by those of the newer object.
Besides increasing the completeness of the data, we can also include more attributes per object by merging the information of different sources.

In our implementation, we developed heuristics to find equal objects/entries using certain key attributes and normalization algorithms.
For hospital objects we additionally store the corresponding wards (e.g. internal medicine). As different terms are used for describing the same ward, we need to identify those ambiguities in order to maintain consistency in our database. Therefore, we use a synonym table developed by experts which maps the extracted wards to a standard denotation.

2.5 Step 4: Publishing the data

The retrieved and revised data can be published in a standardized format. There are several possibilities to output the data, for example as RDF or with an OGC-standard web feature service. The latter has the advantage of using geographic components of the data and can be directly included in GIS software like Udig or Gaia. In our prototype, we implemented a basic web feature service (WFS).

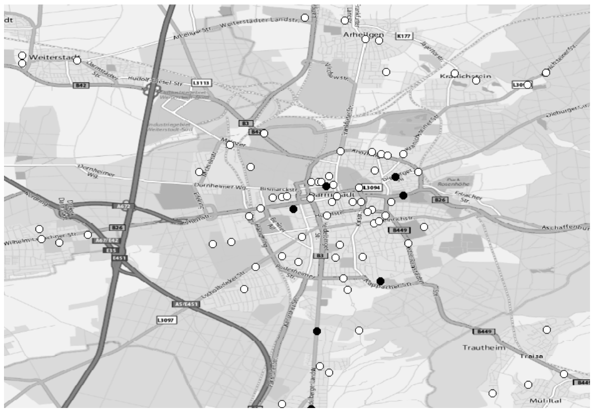

Figure 2: screenshot of results from our prototype visualized by Gaia, black points are hospitals, white points are emergency accommodations

3 Evaluation

For our evaluation, we chose four websites which provide hospital data and seven websites which contain information about possible emergency accommodations. These are for example sites about schools, venues, hotels, etc. We will evaluate single sources first and then compare them to the state-of-the-art.

3.1 Evaluation of single sources

We wanted to verify the need for more than one resource and compared the results of a crawler run making use of all sources with other crawler runs limited to single sources. Therefore we created several independent databases for the crawler runs. We evaluated them based on a manually created gold standard for the counties “Darmstadt-Dieburg” and ”Darmstadt”.

3.1.1 Hospitals

We first analyzed the websites and the available attributes of the hospitals (table 3.1.1). The different sources do not provide data for all attributes, but by merging them a higher coverage can be achieved.

	kranken-haus.de	krankenhaus-experte.de	weisse-liste.de	hospital-abc.de
address	X	X	X	X
phone	X	X	X	X
fax	X	X	X	X
mail	X	X	X	X
bedcount	X	X	X	
wards			X	X
coordinates			X	

Table 1: comparison of sources (quantity and attributes) *= only German hospitals

As we can see in table 3.1.1, which compares recall and precision of the addresses of hospitals, almost all sources have already a good precision, so the erroneous entries of one source would be discarded in a majority voting.

source	**recall**	**prec.**
krankenhaus.de	0.5714	1
krankenhaus-experte.de	0.5714	1
hospital-abc.de	0.8571	0.9286
weisse-liste.de	0.5714	1
prototype	0.8571	1

Table 2: comparison of hospital sources (address)

By merging information of different sources, we hoped to achieve a higher recall over single sources. Except in the case of one source, which is nearly complete, the combination of multiple information repositories yields higher recall for addresses than single sources, but as we have seen in table 3.1.1, the outstanding source does not provide all attributes.

3.1.2 Emergency accommodations

We repeated the process described above for sources containing data about possible emergency accommodations. The difficulty in gaining information about emergency accommodations is the diversity of the eligible buildings. Since there is no source which contains all entities, we combine different sources to a new type. As you can see in table 3.1.2, a satisfying recall is only possible by the aggregation of sources.

source	**type**	**recall**	**prec.**
schulweb.de	schools	0.152	0.866
gezielt.net	venues	0.071	0.714
places.falk.de	venues, schools, hotels, museums	0.263	1
hotel.de	hotels	0.152	0.867
stadionwelt.de	stadiums	0	-
hotelguide.de	hotels	0.061	1
schulradar.de	schools	0.384	0.947
prototype	all	0.687	0.955

Table 3: comparison of evacuation accommodation sources (address) - based on all evacuation accommodations of the gold standard (independent of source type)

3.2 Evaluation against state-of-the-art

We evaluated the results of our prototype and data taken from the Open Street Maps project and Google Maps by calculating its correctness and coverage with the help of our gold standard (table 3.2).
While the evaluation of hospitals results in placing our approach on the top of the evaluated approaches, Google and OSM do not allow to search for emergency accommodations without a disproportional effort.

approach	**recall**	**prec.**
LinkedGeoData - hospitals	0.714	1
Google Maps - hospitals	0.8	0.857
prototype - hospitals	0.857	1

Table 4: comparison of different approaches

4 Conclusion & Future Work

We presented a concept for fetching and aggregating specific spatial data from semi-organized sources like selected websites. Important parts of a system capable of executing this task are a webcrawler, an information extraction module, a persistent store and an output subsystem.
We have shown a sample implementation which is able to find hospitals and emergency accommodations. We evaluated our prototype against state-of-the-art approaches and discovered that we outperform them concerning recall and precision by merging information of different sources.
Because of the encouraging results, further research on this topic is worthwhile.
We explained that our system needs to identify relevant websites and describe the structure of these sites using regular expressions. So far, we provided this information through experts. We want to try different techniques from machine learning, namely automated classification for website identification and pattern learning for structure description, to reduce the amount of needed user interaction and make the setup of our service available even for untrained user. As we use very simple regular expressions, a knowledge-lean approach could be used to learn these patterns [3]. Further evaluation will have to be done as we expect

the automatization to impair our results.
To improve data quality, a ranking algorithm could be implemented to deal with the fact, that the websites will have varying correctness and up-to-dateness. Data retrieved from higher-rated sources would be preferred.
It is easy to integrate other data sources like already existing databases or ontologies in our service. This allows to combine the efforts of different approaches to possibly further improve results and the aggregation of more information that is not found in the internet like the counts of free beds in a hospital.

References

[1] Sebastian Doeweling, Florian Probst, Thomas Ziegert, and Knut Manske. Soknos - an interactive visual emergency management framework. In *NATO Science for Peace and Security Series C: Environmental Security*, 2009.

[2] Jeffrey Friedl. *Mastering Regular Expressions.* O'Reilly Media, Inc., 2006.

[3] I. Muslea. Extraction patterns for information extraction tasks: A survey. In *The AAAI-99 Workshop on Machine Learning for Information Extraction*, 1999.

[4] Marc Najork and Janet L. Wiener. Breadth-first search crawling yields high-quality pages. 2001.

[5] Heiko Paulheim, Sebastian Döweling, Karen Tso-Sutter, Florian Probst, and Thomas Ziegert. Improving Usability of Integrated Emergency Response Systems: The SoKNOS Approach. In *Proceedings "39. Jahrestagung der Gesellschaft für Informatik e.V. (GI) - Informatik 2009"*, volume 154 of *LNI*, pages 1435–1449, 2009.