Towards a Scalable Geoparsing Approach for the Web

Sheikh Mastura Farzana[1,†], Tobias Hecking[1,†]

[1]German Aerospace Center (DLR), Institute for Software Technology, Linder Höhe, 51147 Cologne, Germany

Abstract

The ongoing surge in web data generation and storage, coupled with embedded geographic information, holds immense potential for enhancing search applications across diverse domains. However, extracting geographic information for further enhancement of web search remains inadequately explored. This paper addresses a critical gap in the realm of geographic information extraction from web data, emphasizing the absence of unified pipelines for processing such information. In response to this void, we present a pipeline specifically tailored for web data. Furthermore, our contribution extends beyond the development of the pipeline itself to include a comparative analysis of various gazetteer-based geotagging methods in terms of accuracy and scalability along with a sizable corpora of location annotated web documents.

Keywords

Geoparsing, Geographic Information Extraction, Geotagging, Geocoding

1. Introduction

Accurate geographic information extraction from web resources is a building block of a geo-enriched search index, which enables a wide range of location aware search applications. One way of achieving this is to apply geoparsing on web data. Geoparsing is a standard way of extracting locations from text (geotagging) and mapping them to their respective coordinates (geocoding) [1].

However, apart from the challenges of the geoparsing process itself [2], scaling up geoparsers to the web comes with additional issues regarding scalability and efficient preprocessing of websites [3].

Currently there is a lack of large scale location annotated web data to test the scalability of available geoparsers. Existing public datasets are heavily domain specific and contain particular content types (eg: social media data), making them unsuitable for evaluating web geoparsers. End-to-end geoparsing pipelines that can efficiently extract geo-coordinates from websites in a single pass are still under development.

This paper outlines a pipeline that can streamline processing web data for geographic information extraction. Additionally, presents a large corpora of location annotated web contents.

GeoExT 2024: Second International Workshop on Geographic Information Extraction from Texts at ECIR 2024, March 24, 2024, Glasgow, Scotland

*Corresponding author.

†These authors contributed equally.

Sheikh.Farzana@dlr.de (S. M. Farzana); Tobias.Hecking@dlr.de (T. Hecking)

0000-0003-4242-2458 (S. M. Farzana); 0000-0003-0833-7989 (T. Hecking)

Furthermore, the paper delves into the advantages and limitations of various gazetteer-based geoparsers, providing insights for future advancements in the field.

2. Related Work

There are many off-the shelf geoparsers available with different capabilities. However, these parsers were not intended for web scale geoparsing and display individual limitations. The Edinburgh Geoparser, designed for standalone use, poses challenges in seamless integration with diverse data processing frameworks. Its computational efficiency per document remains ambiguous [1]. CLAVIN, an open-source geotagging and geoparsing tool, excels in context-based resolution but demands specialized expertise in Java which may be difficult to integrate in an existing web data processing pipeline [2]. Geoparsepy, utilizing OpenStreetMap, requires a dedicated PostgreSQL database integration and returns coordinate polygons instead of individual coordinates which is helpful when projecting locations on a map but less so when precise coordinates are expected [3]. The newly available spatial clustering-based voting approach that combines different parsing approaches shows remarkable results for location disambiguation, however has an incredibly high time consumption rate unsuitable for web scale geoparsing [4].

here are different types of geoparsers than can be explored to find the appropriate solutions. Hu et al. explains in his paper several types of geoparsers [1]. Our previous experiments have shown that learning based geoparsers are significantly slower than simpler parsers.[3] Moreover, since crawling and indexing web data requires complex framework of it's own, a geoparsing component must be capable of seamlessly integrating into these frameworks which is difficult to achieve with existing geoparsers as they were not intended for such use cases.

Our studies have shown that among the many approaches of geoparsing, although learning based and hybrid solutions perform better, gazetteer based solutions are the easiest to integrate into existing pipelines, have acceptable performance, can be scalable and extend to multiple languages. Therefore, in this paper we have implemented several possible gazetteer based geographic information extractors, all of them are capable of integrating into existing web data processing frameworks. Their performance has been analyzed on different metrics on a large location annotated web data corpus created by us as well as on another pre-existing location annotated corpora.

3. Geographic information extraction pipeline

Web content analysis serves as a valuable tool for extracting various types of information, ranging from content category, language, various meta-information etc. The extraction of these components relies on different modules, many of which are pre-existing. As a result, any module for extracting geographic information must be capable of merging with the rest of the analysis pipeline without added complexity. 1 shows different parts of our proposed web content analysis pipeline.

[1] https://www.ltg.ed.ac.uk/software/geoparser/
[2] https://github.com/bigconnect/clavin
[3] https://github.com/stuartemiddleton/geoparsepy

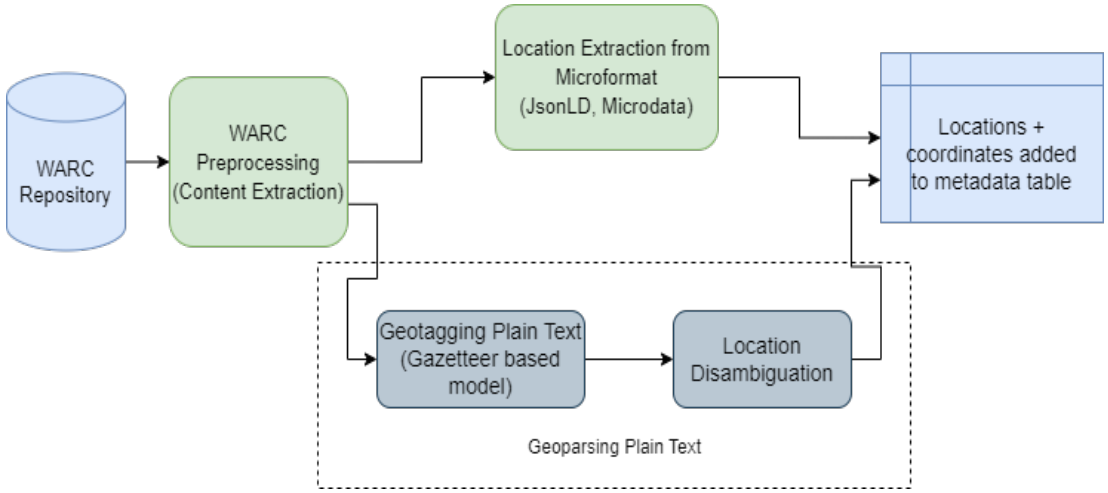

Figure 1: Web data processing pipeline.

- **WARC Repository:** Web crawlers[4] are used to crawl the web and the crawled data is stored as WARC (Web ARChive) files, a specialized format for archiving web content[5]. Crawling can be seen as a process that accesses webpages and downloads their content. Therefore, WARC files contain metadata such as the URL, timestamp, and content type, along with the actual content.
- **WARC Preprocessing:** Each WARC file can contain information of several webpages. These files are required to be processed in order to retrieve individual data of each web page, mainly plain text and various forms of metadata including microformats, language, HTML headers etc. Robust libraries, such as Resiliparse,[6] can be employed for efficient processing.
- **Geoparsing Component:** It is the central element of the web data processing pipeline for our case, data extracted in the previous step are fed into specialized modules for location information extraction. This component comprises of two key processes:
  - **Location Extraction from Microformat:** Specific locations and relevant information related to a web article are often embedded within HTML code using microformats. This special formatting, supported by conventions within HTML, is easily extractable using existing libraries such as extruct[7]. We focus on popular microformats like microdata [8] and json-ld [9] to extract locations without the need for additional disambiguation as the location extracted from here are added by the webpage publishers and definitely link to the content of the webpage.

[4]https://stormcrawler.net/
[5]https://iipc.github.io/warc-specifications/
[6]https://resiliparse.chatnoir.eu/en/stable/
[7]https://pypi.org/project/extruct/
[8]https://developer.mozilla.org/en-US/docs/Web/HTML/Microdata
[9]https://schema.org/docs/schemas.html

  - **Geoparsing Plain Text:** The second approach involves running the extracted plain text through first a geotagging module, in our case, a gazetteer based model. From the geotagging step we extract possible place names from text and associate them with both locations and geographic coordinates using a gazetteer (4.1). Section 4.3 describes at length the geoparsing models we have employed in this paper.

- **Metadata Table:** Extracted geographic information, including locations and coordinates, as well as other information relevant to the particular web resource (not relevant for this paper) is incorporated into a large table. One possible way of storing such large tables is using Apache Parquet file format[10]. These files play a crucial role in enriching web indexes and enhancing web search results.

4. Geoparsing Module

In order to correctly identify locations, inference of each word in a text is important, followed by precise disambiguation of words that can mean both location or something else, eg: Paris, can be a person's name or a location in France. This step is followed by mapping each extracted location to it's geographic coordinates.

For the sake of simplicity and reduction of processing time of huge web corpora, in this work we do not apply any location disambiguation. Instead, each identified location is already annotated with all coordinates along with country codes. Disambiguation can then be performed on demand in a post-processing step. However, since place name disambiguation and one-to-one mapping of places to specific coordinates is an essential step of geoparsing, henceforth we will address our proposed geoparsers as geographic information extractors instead.

For testing and refining the extraction approaches, we have created a dataset of annotated web data. The subsequent sections provide detailed explanations of the location gazetteer, the process of location extraction from microformats, various gazetteer-based geoparsers, and performance evaluation.

4.1. Location Gazetteer:

A pre-existing database of locations is considered as a location gazetteer.[11][12] We have analyzed GeoNames and OpenStreetMap(OSM) data, both publicly available. However, OSM database structure is unsuitable for our task as it returns polygon for area instead of single coordinate per location. On the other hand, GeoNames is comprehensive, easy to transform into different formats and contains necessary coordinate information. Hence, we have chosen GeoNames data for crafting our location gazetteer; we use the 'allCountries.txt' file provided for free public use by GeoNames.

[10] https://parquet.apache.org/
[11] https://www.geonames.org/
[12] https://www.openstreetmap.org/

4.2. Location Extraction from Microformats

Microformats are formatting conventions intricately integrated into HTML code of web-pages, commonly applied for structuring data such as contact information, events, and geographic details, enhancing accessibility and understanding of web content. In this paper, we have only included JsonLD microformat and we plan to integrate microdata format in future. JsonLD is embedded as dictionary which can contain many nested dictionaries. During extraction process, we look for entities which has the key 'address'. If found we process it to gather all the locations included inside the tag and search for a match in GeoNames gazetteer 4.1.

4.3. Gazetteer based Place Name Extraction Models

The gazetteer matching mechanism entails cross-referencing location information in text data with an established location database, in our case the gazetteer mentioned in 4.1. While this method is generally successful in precisely identifying location names, its efficacy highly depends on the quality of the gazetteer and the parsing approaches used. Based on the limitations of existing gazetteer based geoparsers (eg: Geoparsepy) we chose to instead implement different types of place name extractors namely string matching, GateNLP [13], Grammar based approach and Named Entity Recognition (NER).

4.3.1. String matching

The easiest solution for looking for a location in a document is to match each word in that document against a location gazetteer. Due the simplicity of the approach, it can be incredibly fast. In our implementation, we first match pairs of consecutive words with the gazetteer to be able to identify locations such as *San Francisco*, if found we remove them from the main text and proceed to look for a match with single worded locations. It is necessary to mention that we implemented this solution with first four consecutive words look-up (eg: San José del Cabo) followed by three consecutive words (eg: Andorra La Vella) and then word pair and single worded cities. However, this solution decreases the recall and precision values with an increase in time consumption.

4.3.2. GateNLP

GateNLP is an open-source framework for natural language processing and text mining. We take advantage of this library to extract locations from plain text. GateNLP accepts a specially formatted gazette and has a look-up function that can match exact parts of strings to the provided gazette. In our case the gazette is a modified version of the location gazetteer 4.1. GateNLP is able to retrieve locations that comprises of multiple words. Looking up each location from gazetteer in the text without this library would be very time consuming and redundant.

[13] https://gatenlp.github.io/python-gatenlp/

4.3.3. Grammar based approach

We use language specific grammar rules to identify locations in a text. Often proper nouns in a text are locations. We use NLTK[14] library to construct a tagged tree from all words of a text and retrieve the ones tagged as proper noun (NNP). It is possible to add different grammars from different languages for similar extractions. Here is the grammar equation we have used for English.

$$\text{NP} : \{< \text{NNP} > +\}$$

The extracted proper nouns are then matched with the gazetteer to identify locations, corresponding coordinates and country codes.

4.3.4. NER

Named entity recognition is a popular method of identifying the entity types of a text. We use SpaCy[15] to identify location (LOC, GPE) and match them with the gazetteer. A multilingual SpaCy model (xx-ent-wiki-sm) is used to be able to process documents is several languages.

5. Evaluation

5.1. Dataset

In order to create a large location annotated corpora we required web data. Instead of crawling random web pages, we applied a reverse parsing technique. We first created a set of queries using the 'cities500' table of the GeoNames database. Further filtering the data by removing all locations that are outside Germany in order to keep the number of queries to a manageable amount. For each location of the table we formulate the queries in the format *city, state, country* (e.g., Bonn, North-Rhine-Westphalia, Germany), resulting in approximately 11, 500 queries. The assumption here is that, inclusion of state and country information along with city names will provide certain levels of disambiguation to the search engine. For example, to provide a distinction between Frankfurt, Hesse, Germany and Frankfurt, Brandenburg, Germany etc.

Next, we executed the queries using the ChatNoir search engine [5], on the CommonCrawl2017[16] and ClueWeb2022[17] data indices downloading maximum of top 10 search results. From each resulting URI we the downloaded plain text using the Resiliparse library and corresponding microformat data using the Extruct library.

Afterwards, we annotated the plain texts with only the queried locations it is associated with. This annotation process resulted in a test data corpus comprising 21, 834 web documents and a total of 60, 070 annotated locations, average length of the texts are 26, 482 characters.

Furthermore, we have used the 'LGL corpora' as referenced by Gritta et al. in his paper [6]. The dataset was created in 2010 by Lieberman et al.[7] This dataset comprises 557 news articles, each meticulously annotated with location information, including geo-coordinates,

[14] https://www.nltk.org/
[15] https://spacy.io/
[16] https://commoncrawl.org/search?query=2017
[17] https://lemurproject.org/clueweb22.php/

by experts. Evaluating the performance of the geoparsers on this dataset alongside our larger corpora ensures the consistency of our results across both datasets.

The LGL corpora does not include microformat information, subsequently we did not attempt microformat location extraction during analysis on this dataset.

It is to be mentioned that both dataset contain only English text.

Considering the substantial size of our web corpora, we sampled 10, 000 instances to efficiently analyze time consumption. We had a total of 4140 JsonLD microformat in our 10k sample. 1580 instances of the web corpora returned locations extracted from microformat. For cases where no location is found from microformat, we proceed to apply different geoparsing approaches, the configurations of which are explained in Section 4.3, covering a spectrum from naive to sophisticated implementations.

5.2. Evaluation Metrics

In our case, the most important evaluation metrics are recall rate and time consumption. *Recall* shows us the percentage of locations that are successfully retrieved compared to actual annotations. Since our location gazetteer includes all countries but the Web Corpora 10k exclusively features German city annotations, during performance analysis, if we apply *Precision and F1-score* metrics, although important indicators of performance, the results will be very inconsistent and incomparable, hence we refrain from applying these metrics on the Web Corpora 10k dataset.

However, *precision, recall, f1-score, and runtime*, all 4 metrics are applied on the LGL corpora for analyzing performance. We have computed precision, recall, and F1-score (denoted as P, R and F) over total number of annotated locations across all documents against total number of retrieved locations across all documents. Runtime, measured in seconds, it is the total amount of time that was required to process all texts from each dataset, it excludes the time needed to load datasets or models that have an one-time requirement for the entire process. Our tasks were executed on an Intel(R) Xeon(R) Platinum 8380 CPU @ 2.30GHz machine.

5.3. Results

Table 1
Performance Metrics for Different Geoparsing Approaches

	LGL Corpora				Web Corpora 10k	
	P_{total}	R_{total}	F_{total}	$Runtime_{total}$	R_{total}	$Runtime_{total}$
StringMatching	0.07	0.57	0.12	**1.47**	0.84	**231.32**
$StringMatching_{sr}$	0.09	0.57	0.15	16.03	0.84	3218.34
GateNLP	0.07	**0.77**	0.12	5.33	**0.98**	1349.28
$GateNLP_{sr}$	0.08	**0.77**	0.14	18.81	0.96	4083.0
GrammarRules	0.27	0.62	0.38	8.06	0.84	2651.78
NER	**0.57**	0.40	0.47	20.01	0.77	1624.31
NER_{sr}	**0.57**	0.56	**0.56**	7.93	0.67	4239.55

Table 1 shows very interesting results for each approach. Here, *sr* denotes instances where stop words were removed. It was an attempt on our part to reduce the amount of text to be analyzed hoping it will result in reduced time consumption. However, as we can determine from the table that inclusion of a stop word removal function increases the time consumption substantially but fails to improve any of the metrics significantly.

Our fastest model for both dataset is expectedly *string matching*, it has a runtime requirement far less than any of the other models. Regardless, this naive approach of extracting locations from text fails to have acceptable precision values resulting in low F1 Scores as well (LGL Corpora). We can observe similar performance from *GateNLP*, it has incredibly high recall compared to precision in retrieving locations. The precision score is low due to the fact that the parser finds match in the gazetteer for a lot of regular words such as *work, home, road* etc.

Noticing this behavior, we delved into analyzing if our location gazetteer is up to the mark. What we found is that, there are many locations across the world that are also regular words such as *work, tuesday, move, home, road* and many more. We have searched for such odd locations on a publicly available map and was surprised to see these locations exist bringing us to the conclusion that the location gazetteer is quite extensive. At the same time, very naive approaches of extracting geographic locations that do not use any other information such as sentence structure or context are bound to underperform. Additionally, reflecting on the necessity of inference and disambiguation techniques while extracting place names from plain text.

For the case of Web Corpora 10k, the runtime and recall values of different approaches show comparable values to the LGL dataset, indicating consistent behavior. We see that *GateNLP* has a recall of 0.98, however looking at the precision and f1-score of this parser on the LGL dataset we can conclude that *GateNLP* extracts high numbers of non-location words as locations.

We had expected the *grammar rule* based approach to be faster than it is in reality, this approach also suffers from lack of additional context information on extracted proper nouns, as many proper nouns can be place names as well as other things. The *NER* has much more acceptable recall values (LGL Corpora) due to the fact that SpaCy models are trained to be able to distinguish between different types of entities, providing automatic disambiguation between place names and other entities. It can be observed *NER* approach show acceptable recall values with not very low precision hence The *NER* based approach is overall the most hopeful one amongst the four models we have tried based on the metrics.

6. Discussion

Importance of geographic information extraction is undeniable for information linking and the challenge of geoparsing web content is far from resolved. In this paper we have shown valuable insights into the scalability and capabilities of gazetteer based parsing solution for web content with one solution having acceptable performance. However, our current approach has limitations in-terms of place name disambiguation and one-to-one mapping of places to specific coordinates. Regardless, more exploratory work is needed in this area in order to achieve an well performing end-to-end web scale geoparser that has similar accuracy as existing geoparsers for small texts as well as high robustness when integrated into web data processing pipelines.

Acknowledgment

This work has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No 101070014 (OpenWebSearch.EU, https://-doi.org/10.3030/101070014).

References

[1] X. Hu, Z. Zhou, Y. Sun, J. Kersten, F. Klan, H. Fan, M. Wiegmann, Gazpne2: A general place name extractor for microblogs fusing gazetteers and pretrained transformer models, IEEE Internet of Things Journal 9 (2022) 16259–16271.

[2] M. Gritta, M. T. Pilehvar, N. Collier, Which Melbourne? Augmenting Geocoding with Maps, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Melbourne, Australia, 2018, pp. 1285–1296. URL: http://aclweb.org/anthology/P18-1119. doi:10.18653/v1/P18-1119.

[3] S. M. Farzana, T. Hecking, Geoparsing at web-scale-challenges and opportunities (2023).

[4] X. Hu, Y. Sun, J. Kersten, Z. Zhou, F. Klan, H. Fan, How can voting mechanisms improve the robustness and generalizability of toponym disambiguation?, International Journal of Applied Earth Observation and Geoinformation 117 (2023) 103191.

[5] J. Bevendorff, B. Stein, M. Hagen, M. Potthast, Elastic ChatNoir: Search Engine for the ClueWeb and the Common Crawl, in: L. Azzopardi, A. Hanbury, G. Pasi, B. Piwowarski (Eds.), Advances in Information Retrieval. 40th European Conference on IR Research (ECIR 2018), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2018.

[6] M. Gritta, M. T. Pilehvar, N. Collier, A pragmatic guide to geoparsing evaluation, arXiv preprint arXiv:1810.12368 (2018).

[7] M. D. Lieberman, H. Samet, J. Sankaranarayanan, Geotagging with local lexicons to build indexes for textually-specified spatial data, in: 2010 IEEE 26th international conference on data engineering (ICDE 2010), IEEE, 2010, pp. 201–212.