Conference Paper

An Approach to Geotag a Web Sized Corpus of Documents with Addresses in Randstad, Netherlands

Author(s):
Czech, Alexander

Publication Date:
2018-01-15

Permanent Link:
https://doi.org/10.3929/ethz-b-000225615

Rights / License:
Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International
An Approach to Geotag a Web Sized Corpus of Documents with Addresses in Randstad, Netherlands

Alexander Czech (alexander.czech@tuwien.ac.at)

Center of Regional Science at TU Wien (www.srf.tuwien.ac.at)

Abstract. This paper describes a cluster compute workflow about how a web sized corpus of documents \((3.6 \times 10^9\) documents, 260 TiB of data) can be geotagged and how semantic similarities of documents geotagged to the same address could be used to verify these tags.

Keywords. Geotagging, Data Science, Data Mining, Natural Language Processing

1. Introduction

The web can be used to query spatial information. An example of this is to look for a certain restaurant in the city or a still open grocery store. Related to that a study that compared search trends of pedestrian traffic at semantically related places found that they correlate with each other (Kostakos et al. 2013).

The intent is to produce a workflow to geotag a web sized corpus of HTML documents and test the validity of the geotag-document-relation. This incorporates tools ranging from NLP (Natural Language Processing) to cluster computing. The geotag is created by matching post address strings with a document corpus. The addresses belong to the region of Randstad in the Netherlands. It is foundational work for further research into this region and its web-spatial relations.
2. Datasets

Four different datasets are used. An address dataset, a dataset of raw crawled data, a Wikipedia based text corpus and a web directory.

The address dataset is a derivate of OSM (Open Street Map)\(^1\) data. OSM was queried via overpass XML API\(^2\) for all elements that contain address tags. These were then transformed into an address point dataset containing 3.65 million unique addresses.

Common Crawl\(^3\) is a non-profit organization that provides raw web crawling data on a monthly basis. Their archives contain over 3.16 billion URLs with over 260 TiB of uncompressed content.

Wikipedia dumps\(^4\) for Dutch, English, French and German where downloaded. The open source wikiextractor\(^5\) tool was used to extract the plain text. Even though the corpora of the individual languages are not equal in size, they are of semantically comparable content.

A web directory sorts pages into categories. The since April 2017 no longer functional DMOZ was a multilingual and community maintained directory. A static version of the April 2017 state is still available\(^6\). The web directories for Dutch, English, French and German where extracted and processed, yielding 1.3 million URLs each one of which is sorted into a topic group system which is depending on the language up to 12 categories deep\(^7\).

3. Paragraph Vectors (doc2vec)

Document similarity measurements are a common task in information retrieval and NLP. A usual approach is transforming documents into a vector representation and comparing the similarity of these (Manning et al. 2009).

\(^1\) https://www.openstreetmap.org/
\(^2\) http://overpass-api.de/index.html
\(^3\) https://commoncrawl.org/
\(^4\) https://dumps.wikimedia.org/
\(^5\) https://github.com/attardi/wikiextractor
\(^6\) https://dmoztools.net/
\(^7\) For example grass.osgeo.org is categorized in: Science->Social_Sciences->Geography->Geographic_Information_Systems->Software
Paragraph Vectors (doc2vec) are an extension to word2vec, making it possible to learn DVs (Document Vector). In a word2vec model, word vectors are learned from \( n \) surrounding words with a flat neural net. The learned vectors can be used to either predict a word by those surrounding it, or predict the surrounding words for a given word. Semantically similar words produce similar vectors (Mikolov et al. 2013a, Mikolov et al. 2013b).

Similarly a DV in doc2vec is learned from all word-vectors in a document. The model can be implemented in two ways, the \textit{distributed memory model} where the order in which words appear is taking into account or \textit{distributed bag of words} where the word order is ignored. As with word2vec semantically similar documents produce similar DVs (Le & Mikolov 2014).

Paragraph Vectors models have been shown to perform robustly in similarity tasks, when they were not previously trained on the same corpus. Even though being the simpler approach the \textit{distributed bag of word model} performed better than the \textit{distributed memory model} in an off-the-shelf application (Lau & Baldwin 2016).

4. Model Training and Verification

For each language (Dutch, English, French, and German) a doc2vec model is trained using the genism library\(^8\), on the plain text Wikipedia dumps with each paragraph defined as a document\(^9\).

Using Beautiful Soup\(^{10}\) a plain text web scrap was attempted for every URL in the DMOZ dataset. If successful the text was checked if it was longer than 6 words, 32 characters\(^{11}\), and with langid\(^{12}\)(Lui & Baldwin 2012), if it was in the expected language. If all criteria where met it was transformed into a DV with the trained models. The number of vectors for each language as well as the number of unique topic groups is shown in Table 1.

\(^8\) https://radimrehurek.com/gensim/models/doc2vec.html

\(^9\) Model hyper parameters: dm=0; size=100; min_count=200; iter=5; negative=5; workers=4; sample=1e-5

\(^{10}\) https://www.crummy.com/software/BeautifulSoup/

\(^{11}\) 20th percentile of all Wikipedia datasets

\(^{12}\) https://github.com/saffsd/langid.py
For a statistical comparison, all vectors of every language were also divided into a comparable number of randomly generated groups, see Table 1.

<table>
<thead>
<tr>
<th>Language</th>
<th># Vectors</th>
<th># Topic Groups</th>
<th># Random Groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>English</td>
<td>851,458</td>
<td>127,748</td>
<td>92,243</td>
</tr>
<tr>
<td>German</td>
<td>199,758</td>
<td>22,336</td>
<td>22,683</td>
</tr>
<tr>
<td>French</td>
<td>71,494</td>
<td>10,236</td>
<td>10,940</td>
</tr>
<tr>
<td>Dutch</td>
<td>40,557</td>
<td>4,394</td>
<td>4,815</td>
</tr>
<tr>
<td>Total</td>
<td>1,163,267</td>
<td>164,714</td>
<td>118,222</td>
</tr>
</tbody>
</table>

Table 1 Number of vectors and topic groups and random groups per language

For all DVs belonging to the same topic group or the same random group the cosine similarity was calculated within each group. For both sets the mean cosine similarity and standard deviation where determined and a Z-test performed, see Table 2.

<table>
<thead>
<tr>
<th>Topic Groups</th>
<th># Cosine Similarities</th>
<th>English</th>
<th>German</th>
<th>French</th>
<th>Dutch</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>96,629,658</td>
<td>7,621,236</td>
<td>880,003</td>
<td>843,438</td>
</tr>
<tr>
<td></td>
<td>mean</td>
<td>0.526</td>
<td>0.529</td>
<td>0.490</td>
<td>0.622</td>
</tr>
<tr>
<td></td>
<td>SD</td>
<td>0.140</td>
<td>0.161</td>
<td>0.166</td>
<td>0.147</td>
</tr>
<tr>
<td>Random Groups</td>
<td># Cosine Similarities</td>
<td>4,928,545</td>
<td>989,291</td>
<td>246,439</td>
<td>184,789</td>
</tr>
<tr>
<td></td>
<td>mean</td>
<td>0.282</td>
<td>0.382</td>
<td>0.313</td>
<td>0.518</td>
</tr>
<tr>
<td></td>
<td>SD</td>
<td>0.172</td>
<td>0.158</td>
<td>0.161</td>
<td>0.144</td>
</tr>
<tr>
<td>Z-test</td>
<td>p-value</td>
<td>&lt;0.0001</td>
<td>&lt;0.0001</td>
<td>&lt;0.0001</td>
<td>&lt;0.0001</td>
</tr>
</tbody>
</table>

Table 2 Cosine Similarity distributions and Z-Test

The p-values for all languages are highly significant, allowing the conclusion that both sets of cosine similarity values are not identical and the models are able to detect semantic similarities, because the topic groups are on average more similar to each other than the random groups.

5. Matching Addresses and Geotag Validation

Because of the amount of data involved in matching addresses of Randstad to documents in the Common Crawl dataset, the whole computation is done on spark compute cluster13 in pyspark14. The documents are preprocessed similarly to the Wikipedia and DMOZ datasets. Every document is matched against all city names in the address dataset. If a city matches with a document, all street names of this city are also matched against it. For every street name match, all addresses belonging to the street are matched with a regular expression15 against the document. If in this last step one or more matches are made, the document gets geotagged with the matches.

13 https://spark.apache.org/
14 https://spark.apache.org/docs/2.1.1/api/python/
15 https://docs.python.org/3.5/library/re.html
The language for all documents that are matched to one or more addresses is identified with langid. If a doc2vec model for the language exists, a DV is inferred.

The validation of the geotag is based upon the assumption that documents linked to the same address are semantically more similar to each other than a random set, analogue to the model validation in the previous chapter.

The current state of the project is to finalize this last step and implement the task on a bigger compute cluster.

6. **Outlook**

The aim of this work is to provide an access to a corpus of web based data on a spatial basis on which the relation space-web/place-web can be analyzed. The focus in future work is to look into which places people are likely to find when they search for them through the web.

**References**


