Delft University of Technology

Bachelor Graduation Project

Initial Research Report

UrbanSearch

Authors:
Tom Brunnik
Marko Malis
Gijs Reichert
Piet van Agtmaal

Supervisor:
Claudia Hauff
Clients:
Evert Meijers
Antoine Peris

April 26, 2017

TU Delft

Abstract

It is yet to be discovered how the importance of cities in the global network can be elucidated. In this paper, we develop a methodology to be able to reveal an answer to this matter. ...
Keywords: urban, city, data mining, document analysis, filtering

1 Introduction

Common belief is that agglomeration benefits are key to economic growth [4]. However it may be that this econmic growth's primary cause is the increase in (inter)national network embeddedness.

The huge amount of textual data generated online or the numeric historic archives are great sources of information on social and economic behaviours and constitute the bulk of information flowing among each other. Advanced text mining on newspapers and websites containing city names would allow to better understand the role of information in shaping urban systems.
Similar to research efforts in other domains such as financial trade [5] and sales forecasting [8], the idea is to develop search queries that capture urban-urban interactions as they can be found on the web through the co-occurence of geographical names on websites e.g. "Zeeuws-Vlaanderen + Amsterdam " OR "Amsterdam + Zeeuws-Vlaanderen".

We will start by looking at related work that has been done on data from search queries, discussing the similarities to our research. Next we will further analyse the problem in its four mian subsections: the extraction of data from the internet; the filtering and categorizing of this data; the development of search-able queries and last the visualization of the found data. From this problem analysis we will draw the requirement analysis. The requirement analysis is then used to decide upon the framework and tools we are going to use. And last we will give a short conclusion.

2 Requirements

2.1 Must haves

1. General
   (a) Adding city names
   (b) Grouping relations and "zooming" on these relations
2. Search Engine
   (a) Filter results
   (b) Data mining
3. Filtering
   (a) Logic Filters
   (b) Relations Filters
4. Machine Learning
   (a) Types of relations
5. Visualization
   (a) Statistics of relations? Query relations
   (b) Strength of relations
   (c) Types: ML CBS defined

2.2 Should haves

1. General
   (a) Pluggable datasets
2. Machine Learning
   (a) Generalising relations, grouping relations

2.3 Could haves

1. General
   (a) International city names
2. Visualization
   (a) Front end for the app

2.4 Would haves

3 Frameworks and Tools

3.1 Extraction

3.1.1 Common Crawl

Common Crawl [2] is a freely accessible corpus of the pages across the web. Their data is updated and released on a monthly basis. Many researchers have used the data for varying purposes [7] [3] [6]. Since the UrbanSearch project requires us to crawl the web (see section FIXME), the corpus is a very suitable candidate for us to work with.

The data of Common Crawl comes in three formats[1]:

WARC This is the default and most verbose format. It stores the HTTP-response, information about the request and meta-data on the crawl process itself. The content is stored as HTML-content.

WAT Files of this type contain important meta-data, such as link addresses, about the WARC-records. This meta-data is computed for each of the three types of records (meta-data, request, and response). The textual content of the page is not present in this format.

WET This format only contains extracted plain text. No HTML tags are present in this text. For our purposes, this is the most useful format.

For extracting data from Common Crawl, many open-source libraries are available. Common Crawls' official website refers to `cdx-index-client`[2] as a command line interface to their data. It allows for, among others, specifying which index to use, supports multiple output formats (plain text, gzip or JSON) and can run in parallel.

To be able to use the data on our own server, an estimation must be made on the required resources. A simple query on the latest index using the online interface[3] yields 1676 pages of 15000 entries each, which are roughly 25 million entries in total. However, there are over 5.5 million registered domain names with top level domain `.nl`[4]. One would expect many more pages to exist with that number of domains. There are several explanations for this, including:

[1] `https://gist.github.com/Smerity/e750f0ef0ab9aa366558`
[2] `https://github.com/ikreymer/cdx-index-client`
[3] `http://index.commoncrawl.org/CC-MAIN-2017-13-index?url=*.nl&output=json&showNumPages=true`
[4] `https://www.sidn.nl/a/knowledge-and-development/statistics?language_id=2`

- Common large websites, such as `www.google.nl` and `www.wikipedia.nl` have not been fully indexed by Common Crawl, because their "parents", `www.google.com` and `www.wikipedia.org` have already been indexed almost entirely.
- Not every website allows their pages to be crawled. According to Common Crawls' official website, their bots can be blocked via the common `robots.txt`. Additionally, they honor so-called `no-follow` attributes that prevents the crawler to follow embedded links. Sites that use these features are therefore partially or not at all included in the indices of Common Crawl.

3.1.2 Eurostat

Eurostat is the statistical office of the European Union situated in Luxembourg. Its mission is to provide high quality statistics for Europe [1]. We identified Eurostat as a source that is not useful for this particular problem. Although Eurostat contains a lot of statistics on European cities, there is not enough useful information which contributes to giving more insight into the network connectivity of cities. Therefore, we did not include Eurostat as an information source.

3.2 Filtering and Categorizing

3.2.1 Clustering

3.2.2 Filtering

3.2.3 Machine Learning

Machine learning for website classification works by using the raw text from a website. Because the text from websites is often unstructured the 'bag of words' model is used. This model counts how often each word is used. Afterwards the machine learning works in 4 steps:

1. **Creating a feature extractor**
   Given text from a website, returns the "features" from this text. Features are the words that occur in the text and the number of occurences. Before this data is extracted stop words (the, is, at etc) are removed and the rest of the words are stemmed meaning all words will be changed to their root-forms (features - feature, controlled - control).
2. **Manually labelling**
   The first set of examples (which for each class can be just a few of websites), will be manually labeled.
3. **Generating a classifier** TODO
   The labelled examples are fed to a learning algorithms. This will generate a classifier. Several algorithms are:
   (a) SVM - LibSVM ($O(n^3)$?)/*Liblinear*
   (b) Naive Bayes
   (c) Neural Networks
   (d) Descision trees (usually C4.5)
   (e) scikit-learn
4. **Entering new examples**
   When a new (unlabeled) example (website) comes - extract the features and feed it to your classifier - it will tell you what it thinks it is (and usually - what is the probability the classifier is correct). Afterwards the classifier is updated to include new features extracted from the example.

There are 2 programs should be looked at: TODO
Weka: feature extractor and classifier (http://www.cs.waikato.ac.nz/ml/weka/)
Scikit-learn: ? (http://scikit-learn.org/stable/)

3.2.4 TF-IDF

basic idea: 1. using training data to assign values on words - filter meaningless words - assign words with highest value as categories? 2. Do the same on training data for each category (choose a few documents manually per category) and then check for websites for which categories has the highest value.

3.3 Search Queries

3.3.1 Enter Queries

3.3.2 Get Results

3.3.3 Specifications

3.4 Visualisation

3.4.1 neo4j?

3.4.2 Connection between cities

3.4.3 The Strength of these connections

References

[1] Eurostat: About. `http://ec.europa.eu/eurostat/about/overview`. Accessed: 2017-04-26.

[2] Common Crawl. Common crawl. `https://commoncrawl.org/`, 2017. [Online; accessed 25-April-2017].

[3] Hannes Mühleisen and Christian Bizer. Web data commons-extracting structured data from two large web corpora. *LDOW*, 937:133–145, 2012.

[4] Michael E Porter. Location, competition, and economic development: Local clusters in a global economy. *Economic development quarterly*, 14(1):15–34, 2000.

[5] Tobias Preis, Helen Susannah Moat, and H Eugene Stanley. Quantifying trading behavior in financial markets using google trends. *Nature: scientific reports*, 2013.

[6] Sameer Singh, Amarnag Subramanya, Fernando Pereira, and Andrew McCallum. Wikilinks: A large-scale cross-document coreference corpus labeled via links to wikipedia. *University of Massachusetts, Amherst, Tech. Rep. UM-CS-2012-015*, 2012.

[7] Jason R Smith, Herve Saint-Amand, Magdalena Plamada, Philipp Koehn, Chris Callison-Burch, and Adam Lopez. Dirt cheap web-scale parallel text from the common crawl. In *ACL (1)*, pages 1374–1383, 2013.

[8] Lynn Wu and Erik Brynjolfsson. The future of prediction: How google searches foreshadow housing prices and sales. In *Economic analysis of the digital economy*, pages 89–118. University of Chicago Press, 2014.