Wikipedia Mapping

Danilo Zocco
11-510-013

Gian-Reto Bonadurer
17-605-205

Nick Stoeckl
13-605-506

Patrick Waelchli
17-601-873

Software Engineering for Economists
Philipp Zahn
University of St. Gallen

January 5, 2018

Contents

List of Figures

1 Introduction

The amount of accessible data increases everyday. Naturally this places ever greater importance on data science as it is a proven method of processing such vast numbers of available data points. [Provost and Fawcett, 2013] Gathering and processing information automatically from the internet is becoming ever more crucial as we delve deeper into the age of digitization. Whether it is in order to gain a competitive advantage, to conduct extensive market research, or to simply gather information the power of data science can be implemented as a fundamental advantage to replace manual labour with algorithms.
This project illustrates, in a simplified manner, one way data science has the ability to tackle problems like these. The build crawler gathers the available links within a Wikipedia article and subsequently visualizes its findings through a network-map. Although various methods exist for achieving the aforementioned goal, the next sections outlines this team's approach along with their reasoning behind the decisions.

2 Tools

2.1 Language

The general-purpose programming language Python is widely used for data science worldwide, the reason being that the language allows for a broad variety of application forms. [Summerfield, 2007] With regards to this project, Python is by no means the only language that could be used to build a crawler but it seemed to be the most suitable as the syntax of Python and the prebuilt modules allow an uncomplicated use. The availability of these prebuilt modules enabled the team to choose the most suitable one for the project and thus allowed them to place more focus on the customisation of the crawler itself.
Finally, because the team members expect to use Python in the future, the choice for the language's lastest version (Python 3) was clear. While, at first the team thought the limited availability of forum articles might lead to an unreasonable amount of effort to solve prevailing problems, it later became apparent that available online resources were more than sufficient to solve the task.

2.2 Modules

- **Beautiful Soup**
  Beautiful Soup (bs4) is a Python library designed for quick turnaround projects like screen-scraping. It allows for an easy way to navigate, search, and modify a parse tree, making it easy to extract links from the Wikipedia page without using much code. [Crummy, 2018]

- **Requests**
  The Requests module enables communication with HTTP. As a result, the request module in Python does not need any manual input and automatically adds query strings to the URL. [Request, 2018]

- **NetworkX**
  "NetworkX is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks." [Networkx, 2018] This module is used to create the network, which is subsequently visualised as a map.

- **Matplotlib**
  The generated network with the networkx module is plotted and visualised with the Matplot library. This tools allows for the creation of different plot types and networks with a high degree of customizability, if needed. The default version is used for plotting as it consistently produces the most readable network graph.

- **NumPy**
  NumPy provides tools for scientific computing. It makes the use of linear algebra easier and allows to manipulate multidimensional arrays, among other capabilities. It is used, in our case, to clean our dataset before building the proper network.

- **itertools**
  Python's itertools module provides tools to build fast and memory efficient iterators. Used in conjuction with NumPy, itertools allows to manipulate large datasets in a quick way.

3 Program

The program is subdivided into two main parts: a crawler that gets links from Wikipedia and a builder that takes those links, builds a network and finally generates a chart of it.

3.1 Crawler

The function of a crawler is to access a webpage and to retrieve the hyperlinks listed on it. It then accesses those new links and repeat the process for each and every page it visits. This is not to be confused with a scrapper that proceeds almost in the same way but will usually gather other pieces of information such as pictures or text.
We faced one main issue while building the crawler. In order not to end up crawling the entire Wikipedia website, we set a limit to the number of items crawled. As a result, depending on the starting article we choose, the crawler hits this limit after only a few pages. To fix this, we can either increase the maximum items we want to crawl (which also increase the running time) or choose to crawl by depth instead of breadth. If we choose to crawl by depth first, instead of crawling every link on a page and then moving to the next page, the crawler will always take the first internal link found in an article and directly start crawling the corresponding new article. This allows for more pages to be crawled before reaching the maximum items limit and for a less clutterd network to be built and graphed.

To illustrate the difference between "breadth-first" and "depth-first" search, we included one graph for each. They are both built with "University of St. Gallen" as

starting article and have a limit of 15 and 20 items, respectively. Figure 1 illustrates breadth-first search while Figure 2 illustrates depth-first search.

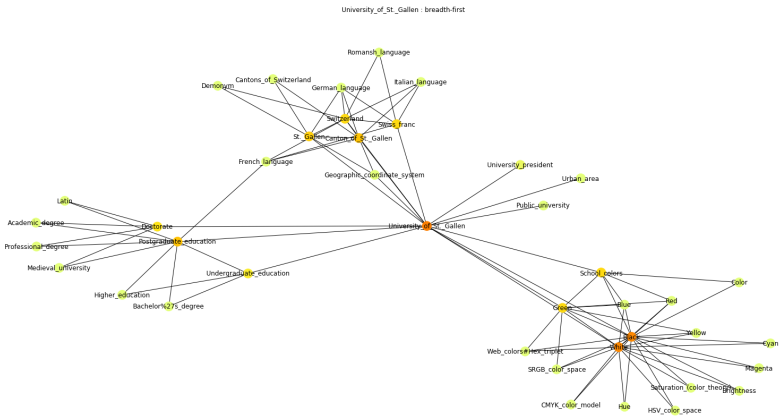

Figure 1: Breadth-first

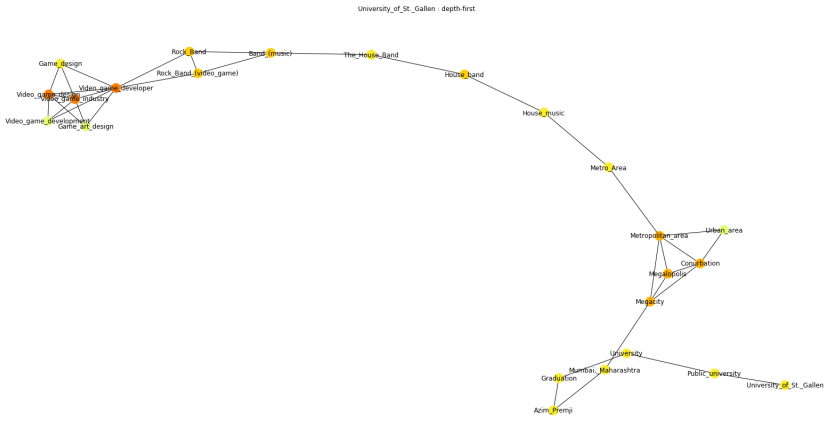

Figure 2: Depth-first

3.2 Building and drawing the network

The input to this part is a dictionary of Wikipedia articles and their corresponding internal links. Before building the network object with the NetworkX module, we use NumPy and itertools to suppress the pages that would become end nodes (i.e articles linked to one and only one other article). These end nodes do not add much information to the network and are often hardly readable on the graph produced. See Figure 3 for an example. This is obtained with "Foobar" as starting article and a limit to just 30 items with breadth-first search. Notice how a small increase in maximum items can render the graph unreadable when keeping the end nodes. End nodes can be useful when using depth-first search, but it also tends to make the graph unreadable for higher items limits.
Once this is done, we use the NetworkX module to build a networkx.Graph object. These are composed of nodes (articles) and edges (links between articles). Finally, we are able to plot the network with Matplotlib.

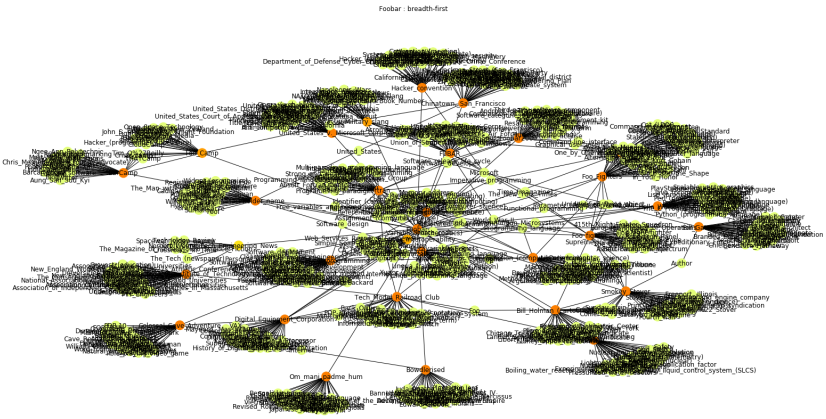

Figure 3: Keeping end nodes

4 Conclusion

In conclusion, this project has showed the team the benefits of applying data science to certain tasks in order to automate otherwise labor-intensive processes. The Python-based code uses five separate modules and two main steps to crawl a user-defined Wikipedia article, to identify links to other articles, and to subsequently visualize the interconnections of those links in a map. One issue that quickly became evident, was that of the depth versus breadth of the search. While it is not a problem per se, it often lead to confusing visualizations depending on the starting article. An understanding of the issue at hand, however, allows the user to redefined the limits in order to produce more attractive visualizations. Although the wiki crawler has no implementable economic application, it does serve as an example of how data science and data manipulation has the ability to provide otherwise unreachable insights.

After extensive testing, the team has found only a few minor limitations to the program. The largest of those is the fact that the program has difficulty when crawling Wikipedia articles in languages other than English. While not certain, the team concluded that this is likely due to the "disambiguation" part of the parser, as other words were used in place of disambiguation in other languages. Furthermore, this program is only applicable to Wikipedia articles, which was the teams original goal but limits the application scope of the project.

Overall, this project represents only the first step in a network analysis process. Putting the drawing feature aside, the networks built by our program could be used for further and more detailed network and relationship analysis.

References

[Crummy, 2018] Crummy (2018). Crummy. https://www.crummy.com/software/BeautifulSoup/.

[Networkx, 2018] Networkx (2018). Networkx. https://networkx.github.io.

[Provost and Fawcett, 2013] Provost, F. and Fawcett, T. (2013). *Data Science for Business: What you need to know about data mining and data-analytic thinking.* O'Reilly Media, Inc.

[Request, 2018] Request, P. (2018). Request. http://docs.python-requests.org/en/master/.

[Summerfield, 2007] Summerfield, M. (2007). *Rapid GUI Programming with Python and Qt: the definitive guide to PyQt programming.* Pearson Education.