Curated scrapes: Designing a web crawler for ethnographic research data

Kim Pham (Librarian), Kirsta Stapelfeldt (Librarian), Alejandro Paz (Assistant Professor, Anthropology) - University of Toronto Scarborough

Background
This pilot project brings together anthropologist Alejandro Paz, faculty and students from the Department of Computer & Mathematical Science, and the Library’s Digital Scholarship Unit to investigate the increasingly central role that Israeli online journalism in English plays in the global dissemination of news stories about the Middle East. The stories in English, from online newspapers circulate widely, and inform the vigorous public debate about Israel and Palestine. Online digital media are now recognized as a central, if poorly understood, aspect of globalization. In order to understand how this digital news is being produced, disseminated, and used across a global digital newscape, we formed a team to build MediaCAT (Media Crawler & Archive Technology), a web crawler and archive application suite. As product manager, the Digital Scholarship Unit plays crucial role in the development process to ensure the application is designed in a responsible, sustainable manner to support future web archiving services provided through the library.

Technology
MediaCAT is an open-source Python based application that uses the Django framework. Django was originally developed for a news environment and is designed for database-driven sites that require search and indexing of large volumes of content. Django provides a customizable front-end and authentication system that allows the creation of multiple users to access the site. The web crawler is integrated in the framework and uses the Newspaper API to perform a scrape of a target list of sites. It finds a positive match for a given set of sites and/or keywords.

By utilizing the Newspaper API, the application extracts content, unlike most other open-source web crawlers. The Newspaper API uses natural language processing (via NLTK) to identify and parse content that is written into the database which is then made searchable. Crawling is also more efficient by only capturing sites that match given queries. These URLs are then captured using wpull and PhantomJS to store WARCs for in-depth technical analysis. The continuously updating data will be used to investigate the process of producing news for a global public sphere. A second function is to monitor a set of Twitter handles for the same set of sources and keywords. The result is a list of individual URLs or tweets with references, either mentions or hyperlinks, to one of the sources.

Baseline Crawler
- Stores state on MySQL DB
- Stores state on SQLite DB
- Stores state on MySQL DB
- Stores state on SQLite DB

Weekly Crawler
- Gets tweets
- Matches tweets
- Article links
- Gets RSS feeds
- Extracts article links
- Using Newspaper API
- Article links sent to ArticleExplorer to determine if they should be stored

We are in the process of migrating the application to a stable server from Digital Ocean. In anticipation of this move, a product roadmap was created as tool to manage the strategic direction of application. Crawler performance is considered to be a major priority and constant area for improvement. Recently, in-memory paging was swapped out and tested with a default Django SQLite Django database without success. Afterwards, a manual MySQL database was implemented to manage visited site information which has been working but still requires further testing. Multithreading across many domains is now possible, but not within a single domain. Additional hardware upgrades (cores and RAM) will be necessary to scale up. We anticipate running our first baseline crawl on three to four domains for articles since 2004 by Spring 2016. We hope to incorporate external tools as part of our workflow to analyse our content and have been investigating warcbase to explore our WARC files and JaVaScRiPT libraries such as d3 and p5 (processing) to visualize networked information.