Madaap - a tool to locate and extract GIS datasets from web

Abhinav Ghai

Madaap is a tool to search for GIS related datasets across the web and extract entities like Title, Author, Abstract, Download links etc from the relevant webpages thus creating a catalog of freely available GIS datasets.

Madaap contains several components/modules as listed below:

1. Collector - collects webpage URLs and puts them to a queue
2. Extractor - picks a URL from queue,checks if this webpage is GIS-related and extracts entities from that webpage and stores them in our database.
3. Checker - Regularly checks the database to see if the URLs are still valid or not
4. Crawler: takes the seed URL and adds the subpages(to depth 1) to the queue after filtering some useless URLs which end in css,jpg,gif,mp3,pdf etc

**1. Collector:**

It is a generic entity, which collects URL from any place X and adds them to a queue.

Currently, there are two type of collectors.

1. Manual Feeder: It collects URL from a text file named url.txt at regular intervals of time and adds them to queue. After adding them to queue, it sends those URL to a text file named used.txt
2. Twitter collector: It queries twitter at regular intervals of time with a query, which is read from configuration file twitter.xml. From the twitter replies, which are in JSON format, we look for URL entities within the tweet, and add those URL to the queue.

There are also defunct collectors like facebook,google plus and delicious collectors which should be expanded in later stages.

Search interval can be configured through configuration file for manual feeder as well as twitter collector

**2. Extractor:**

It picks up URL, one at a time, from the queue where collectors were adding the URLs. This URL undergoes following steps. This step heavily uses tools provided by GATE toolkit.[gate.ac.uk]

1. A generic GATE document is created using this URL.
2. Document is used to extract entities like Title,Author,Abstract etc.[explained later]
3. If URL has downloadable links, it is inserted into database along with the extracted entities, else it is discarded.
4. The outgoing links from this URL are also considered for extraction. They are added to the queue for extraction process as well.

Step 2 further explained:

Objective: We are provided with an HTML page and we want to extract entities like Title, Abstract,Author,Links etc from it.

We are using GATE toolkit to accomplish this task. This is how it works.

Simply put, GATE takes a webpage and treats it as a document. Then we add annotations on top of this document by processing it using NLP tools like stemmer, tokenizer, gazzeteer, morpher etc. These NLP tools are part of GATE. GATE also adds original HTML tags from the webpage to the document annotation set. These annotations are then used to extract entities from the document. We have devised certain set of rules to identify these entities.[These rules can be changed/improved by further testing]

**Entity extraction rules**

1. Abstract:Find <p> tags in html whose content’s length is greater than a specified minimum length(configurable). Find the <p> tag which is at the top of webpage among these paragraphs.
2. Title: Find content from Title,H1 and H2 HTML tags
3. Author
   1. Entities tagged as Organization by GATE(very loose, will give many results)
   2. Find sentences containing words with word root publish, distribution, produce, provide, create etc. Within these sentences, search for entities tagged as Organization by GATE. These entities are more likely to be our authors.
   3. Find entities tagged as Organization by GATE in Title area of HTML page

Option b and c are more strict versions of option a. Currently, we are using option b and c in code.

1. Download link: Get all the outgoing links from a page and see if they end in a certain set of end markers(configurable) of our GIS datasets
2. Format: Check all the unique end markers in set of downloadable links
3. Spatial extent(XML only): Find a set of 4 elements in XML whose contents are just numbers that lie in the range -180 to 180.

In order to perform these entity extraction rules, we use the annotations done on the webpage by GATE.

Example annotations: HTML tags, Tokens(from tokenizer), Sentence(from Sentence splitter).

**3. Checker:**

URL added to our catalog might not remain active after a certain period of time. So, we run a checker at regular intervals of time to see if the URL are still active or not. We use HTTP response codes to identify inactive URLs. Time interval of checking is configurable.

**4. Crawler:**

Given a URL, we don’t just want to extract entities from that web page only. It might be useful to consider web pages which are at a certain depth from it. So, we take the input URL as a seed and look at web pages till depth 1 currently (for computation purpose as a trade-off). We eliminate certain web pages like css,js,jpg,gif,mp3,pdf etc as potential GIS dataset web pages.

**Understanding configuration files:**

There are two configuration files.

1. **madaap.xml**

GATE:

gate.home: installation folder of GATE(this folder will contain gate.exe, bin,lib folders)

Timer:

timer.ManualFeederInterval: time interval for manual feeder collector. Unit is hours

timer.TwitterInterval: time interval for twitter collector. Unit is minutes

timer. CheckerInterval: time interval for checker. Unit is hours.

Database:

database: url,port,username and password for the database where we want to store the URL and extracted entities

Download:

downloads: end markers for identifying downloadable datasets.

Spatial extent:

SpatialExtentKeyword: keywords used to identify north,south,west and east elements in XML page while extracting spatial extent from XML files.

Abstract:

Abstract.MinimumLength: Minimum number of characters(not words) to eliminate very small paragraphs from being chosen as abstract.

1. **twitter.xml**

query.q: query for twitter, may contain several word and hashtags(#gis #data)

query.rpp: results per page parameter for twitter query(20 seems good)

query.page: number of page parameter for twitter query(5 seems good)

query.lang: language(en for english)

[<https://dev.twitter.com/docs/api/1/get/search>] may help to understand how twitter search API works

**Tools and Softwares used:**

1. GATE (Release 7.0,build 4485)[<http://gate.ac.uk/>]

for annotating web pages and performing NLP on web page for extracting entities

2. Java SE 7[<http://www.oracle.com/technetwork/java/javase/overview/index.html>]

language for implementation

3. MySQL 5.5.24[<http://dev.mysql.com/usingmysql/get_started.html>]

for storing extracted entities and URL in database

4. PHP 5.4.3 [<http://php.net/>]

for designing a website to review the extracted entities from database.

Download from here - [<http://windows.php.net/download/>]

5. Procrun[<http://commons.apache.org/proper/commons-daemon/procrun.html>]

to run java application as a Windows service on server machine

Download from here - [<http://www.apache.org/dist/commons/daemon/binaries/>]

**Source code:**

Source code is publicly available on github at following link

[<https://github.com/abhinav-ghai/madaap>]

**Usage:**

Madaap application is installed as a windows service on Windows Server 2008 R2 machine which can be start or stop using Server manager

It can also be run from command line with following command.

**java -jar madaap.jar**

The folder that contains madaap.jar should also contain two folders named config and input.

config folder must contain madaap.xml and twitter.xml

input folder must contain url.txt and used.txt

Run install.bat in order to install madaap as a windows service[Using procrun]

In order to remove service: **prunsrv.exe //DS//Madaap**