Apache Tika, OpenNLP, and Lucene in Geocoding approach

Joonpyo Jun (joonpyo3@illinois.edu)

CS 410: Text Information Systems

Dr. ChengXiang Zhai

2 November 2021

Apache Tika, OpenNLP, and Lucene in Geocoding approach

As the 4th industrial revolution is ongoing in various fields in technology, the Natural Language Processing (NLP) became one of the most important fields of study. Good NLP provides a better quality of media, resources, and other technology-related services to people who need them, especially for the data analysts and text processing software providers. Among various NLP sets of tools, I was intrigued by the Apache OpenNLP toolkit that provides several core NLP tools such as Tokenization, POS Tagging, Parsing, Natural Language Generation, Feedback Analysis, and more (OpenNLP – Overview, 2015). Because the text processing service is most likely to be introduced in the industry sooner or later, I started to search for the project or research that utilized the Apache OpenNLP. From IEEE, the *Automatic Approach for Discovering and Geocoding Locations in Domain-Specific Web Data* attracted me. So, I decided to dive into this research paper and explain how the core functionalities of Apache OpenNLP can be implemented in the industry and potential research along with other useful Apache software – Apache Tika and Lucene.

Briefly, the Apache OpenNLP is an open-source Java library to process natural language text, providing services to handle raw text efficiently. The API and CLI are available for free via its website, http://opennlp.apache.org/, while individual models are also available for downloads (The Apache Software Foundation, 2021). For example, you can specifically download *Tokenization (as a .bin file)* without other functionalities. Specifically, this project (*Geocoding Locations*) only utilized the location entity extraction model (*en-ner-location.bin* from the most recent manual). I could conclude that each OpenNLP model is perfectly individual for a specific use without implementing the entire API. So, what does the location entity extraction model do? This model provides a class called *NameFinder* that takes the input stream (tokens) and trained location model and identifies the locations in the

input. After that, these identified locations are taken into general NLP steps, Tokenization, stop-word removal, and stemming (Mattmann, 2016, p. 88-89).

The structure of algorithm used in the entire project is constituted of three stages. The first part is when Apache Tika extracts location-related text from visible text and from metadata in the files that are identified by the Internet Assigned Numbers Authority (IANA) (Mattmann, 2016, p. 88). The Apache OpenNLP then takes this extracted text and automatically detects if certain location is mentioned, as explained above with *NameFinder*. Finally, the location mentions are passed to Apache Lucene which utilizes *Geonames* dataset to allow geocoding and geolocation from location names (Apache Lucene, n. d.) The *Geonames* dataset provides useful geographic features, including, but not limited to, *Name, Alternate names, Latitude & Longitude, Feature class & code, Population,* and more (Mattmann, 2016, p. 90). The Apache Lucene library – *Lucene Geo Gazetteer* then indexes these searchable features. Moreover, a search functionality and a specialized ranking algorithm are implemented as well from Lucene index. As a result of the entire process, the *best* location, a set of *alternate* locations, and latitude and longitude coordinates are returned from the data available on the web and more specifically, in the domain specific web (Mattmann, 2016, p. 90).

Now that we have all these location-related data, the next step is to test the efficiency of this approach on real data from the DARPA MEMEX and NSF Polar CyberInfrastructure programs. It resulted in 94% of accuracy in location identification. Because people continuously update the geolocations on *Geonames.org*, it is inevitable to automate newly updated popular place names with geospatial coordinate data beyond points. For example, "bounding boxes and other shapes to allow for more meaningful and spatially directed queries, when combined with location textual data" (Mattmann, 2016, p. 92-93).

References

The Apache Software Foundation. (2021). *Apache OpenNLP Developer Documentation*. Apache opennlp developer documentation. Retrieved November 3, 2021, from http://opennlp.apache.org/docs/1.9.3/manual/opennlp.html.

Apache tika. Apache Tika. (n.d.). Retrieved November 3, 2021, from https://tika.apache.org/.

Mattmann, C. A., & Sharan, M. (2016). An automatic approach for discovering and geocoding locations in domain-specific web data (application paper). *2016 IEEE 17th International Conference on Information Reuse and Integration (IRI)*. https://doi.org/10.1109/iri.2016.19

Mohanan, M., & Samuel, P. (2016). *Open NLP based refinement of software requirements*. Mirlab. Retrieved November 3, 2021, from http://www.mirlabs.org/ijcisim/regular_papers_2016/IJCISIM_30.pdf.

OpenNLP - Overview. OpenNLP - overview. (2015). Retrieved November 3, 2021, from https://www.tutorialspoint.com/opennlp/opennlp_overview.htm.

Welcome to Apache Lucene. Apache Lucene. (n.d.). Retrieved November 3, 2021, from https://lucene.apache.org/.