Information Retrieval Project Report

Class Project

presented by
Michael Dell(1359533), Alexander Müller(1376818), Maxim TODO

University Mannheim

July 2014

1 Introduction and Problem Statement

Maxim

2 Architecture Overview

Michi

3 Extraction Pipeline

4 HTML Extraction

Michi

4.1 Person Contact Information Extraction

TODO Michi

After we exploited the structure of the website, the need of having additional checks whether the heuristic really extracted a person or not came up. Therefore we decided to make use of Name Entity Recognition (NER). NER is defined as the identification of proper names in text or collection of text and assign them to a category person, location and organization. Due to our problem domain we only considered Named Entities (NE) of the type Person.

A well-known library for Natural Lange Processing (NLP) tasks is Stanford Core NLP [2], which also includes in their current version a NER Component to recognize English names. Of course this is not optimal and therefore we combined the current version of Stanford Core NLP with an older where a German NER Component exists [1].

In the evaluation Chapter we will investigate how this two different libraries performed.

Nevertheless for each for both approaches the NER Recognition usage is the same. As illustrated above we exploit the html structure of a webpage as doing so in this case. We iterate over the DOM Structure and only consider headers. If in one heading a NE of type Person was found, the algorithm tries to extract contact information, otherwise it continues.

The result of this step of our pipeline is a set of person entity candidates that still can contain corrupt or duplicate data. Therefore the next step is to clean the set.

4.2 Candidate Pruning

The person entities extracted in the previous step of the extraction pipeline can still contain a lot of corrupt data. Some examples are extractions that relate to a NE in the heading, but no contact data in the related parts like: “Publications of Prof. Dir. Simone Paolo Ponzetto”. Furthermore there will be a lot of duplicates in the resulting set, because lots of university employees do have their contact data on duplicate webpages.

To solve the first case we once more analyze the heading that lead to the extraction of contact data and count how many of the words are on a name word list. If this is more than one third we consider them this as a correctly extracted Person entity.

Second to eliminate duplicate entries we developed a light-weight heuristic to identify them. For the ease of use we define that two people are duplicate entities when first name, last name and the email address are the same. Now the question arose which one to keep? For further improving the performance of our system we decided to keep the entity with most not null values in its fields.

After this step the set of candidate entities was pruned in order to assure a high data quality.

4.3 Enable Search with Apache Solr

In the previous chapters the extraction of person entities associated with their contact data was presented. In order to make this information accessible to end-users, to fulfill their information need an information retrieval system is necessary. A special requirement is fulfilling the need of field queries for the purpose of inverse searches. For instance if a user got a call from a telephone number he doesn’t know, he want to retrieve all people that have the corresponding number.

An ideal system to fulfill those user requirements is Apache Solr, which was developed as all-in-one information retrieval/search engine. It fulfills the tasks of index creation, both for free-text queries and field search, as well as the handling and parsing of user queries. Since our problem the search problem in our case is not an typical search on unstructured or semi-structured data residing in textual documents we need to change the configuration of Apache Solr.

The entities that were extracted are stored in a MySQL database with the representation as shown in Figure 1. In order to allow Solr to access the database we needed to change some of the configuration parameters. First of all necessary java classed needed to be included to allow Solr to establish a JDBC connection to the database the entities reside in. Now we enabled the creation of inverted indices both, for field queries and free text queries considering all fields when the user

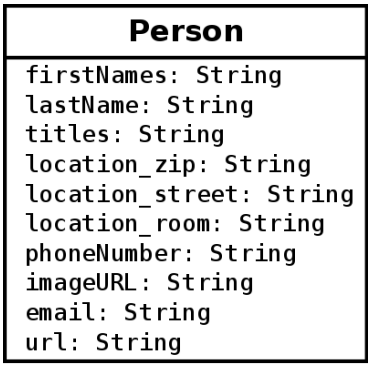

Figure 1: Datastructure of a Person Entity

enters his or her query.

This query interface is in Apache Solr by default exposed via webservices, which we used to communicate to our custom build User Interface which will be described in the following section.

5 Project Evaluation

Maxim

6 Summary

Maxim

References

[1] Manaal Faruqui and Sebastian Padó. Training and evaluating a german named entity recognizer with semantic generalization. In *Proceedings of KONVENS 2010*, Saarbrücken, Germany, 2010.

[2] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, 2014.