Eindhoven University of Technology

Title goes here

Stefan Patelski

November 9, 2013

1 Introduction

1.1 Crawler : S.R. Patra

1.1.1 Motivation

The very first task was to collect data for our project. We decided to implement a focused crawler for Rotten tomatoes website to collect information about different movies. Large collection of movies with credible reviews made Rotten Tomatoes a suitable candidate for us to get our data from. The goal was to start with a seed URL and extract all the URLS on that HTML page along with the metadata about the movie such as movie name, director, producers,reviews etc. The process needs to be repeated for all the collected links.

1.1.2 Approach

For each page, the method *CollectData* in the crawler read the HTML content line by line, only collecting the relevant information and disregarding all the other content that were not relevant to us. This was done with the help of an HTML parser which used *Jsoup* library functions. Whenever relevant data were encountered, they were appended to an XML document. To find the movie URLs , Java's pattern matching functionality was being used which looks for a particular pattern of anchor texts which are specific to movie pages.

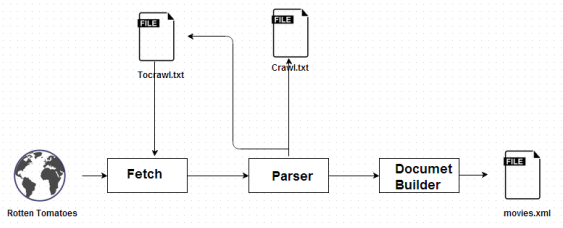

Figure 1: Crawler Architecture

1.1.3 Challenges

The main challenge was to encounter the non uniformity of the content in Rotten Tomatoes web pages. Some of the web pages contained all the information about the movies while in others, a few of the information fields were missing. The crawler was designed to scrap the HTML data in a way such that maximum amount of information can be retrieved. Another challenge was to make sure that the crawler is not blocked by the site. This was handled by building the crawler in a flexible way so that, during an iteration, the crawler will only crawl through a specific number of pages

specified to it as a parameter. After each iteration, the URLs which are already crawlled and the URLs those are yet to be crawlled are written back to *Crawl.txt* and *Tocrawl.txt* respectively which will be used in the next iterations.

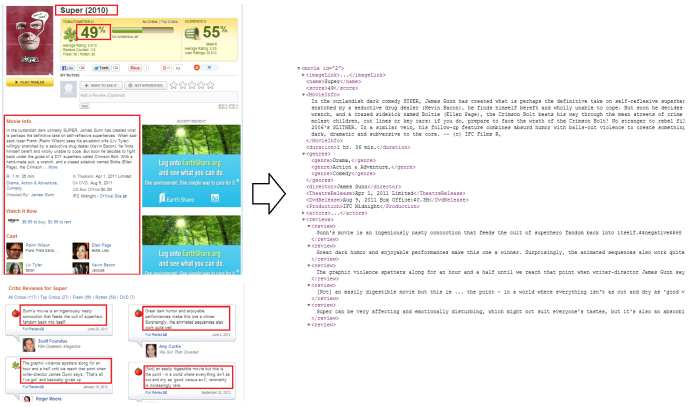

Figure 2: Collecting data from Rotten Tomatoes

1.1.4 Evaluation & Summary

After several iterations, we collected 6227 movies each containing movie metadata and the reviews. We collected only top critics reviews because we feel that those are the most representative of all the reviews. As focused crawler parse all the content in documents and filter only the relevant content for the application, crawling took more time than a normal crawler would take. But the flexibility in the design of the crawler allowed us to run it in iterations so we did not face any issue regarding the time for crawling. We took 20 random samples from the file and compared the contents with the content in the site. In all the cases, we were able to capture all the relevant data needed for the search and sentiment analysis functionality.