# CS839 Project Stage 2 Report

## Member Infomation


- Guangtong Bai ( guangtong.bai@wisc.edu )
- Dan Jiang ( dan.jiang@wisc.edu )
- Changtian Sun ( changtiansun@cs.wisc.edu )


## Web Data Sources

| Data Source | Link | Full Name | Number of Entities | Description |
|---|---|---|---|---|
| IMDb | www.imdb.com | Internet Movie Database | 5.3 Million | IMDb (Internet Movie Database) is an online database of information related to films, television programs, home videos and video games, and internet streams, including cast, production crew and personnel biographies, plot summaries, trivia, and fan reviews and ratings. |
| TMDb | www.themoviedb.org | The Movie Database | 452 Thousand | The Movie Database (TMDb) is a community built movie and TV database. Every piece of data has been added by the community dating back to 2008. TMDb's strong international focus and breadth of data is largely unmatched. |

## Extraction Approaches

First, we go to these two websites. In IMDb, we used the search function to list the movies according to the US box office in descending order. In TMDb, we go to popular movies. This will ensure that they can have a number of overlaps when we try to extract 3500 to 4000 entities from each of them.

The two websites have thousands of pages giving the movie lists. As there are disciplinary changes in the page addresses (such as page number in ascending order), we can access these pages one by one directly. These pages will provide us with links of each movie, which can be extracted by BeautifulSoup.

Then our crawler uses the request tool to open the link for each movie and the open source tool BeautifulSoup is again used here. We first use BeautifulSoup to extract the parse the pages into tree structures. Then we extract our expected information from the tree generated using the APIs that BeautifulSoup provides.

## Type of Entities

### Type of Entities

Movies

### Data Files

- `imdb.csv` : Table A: *4000* movie entities crawled from IMDb
- `tmdb.csv` : Table B: *3894* movie entities crawled from TMDb

### Attributes

| Attributes | Descriptions |
|---|---|
| id | The index assigned to each movie. It is continuous from 1 for each dataset. |
| title | The name of each movie. |
| year | The year in which each movie was on. |
| genres | The genres of each movie. There can be more than one genre. Each genre is separated by a semicolon without space. |
| language | The main language of each movie. |
| runtime | The length of each movie, in minutes. |
| budget | The budget of each movie. |
| revenue | The revenue of each movie. |
| directors | The directing staff of each movie. There can be more than one director. Each director is separated by a semicolon without space. |
| writers | The writing staff of each movie. There can be more than one writer. Each writer is separated by a semicolon without space. |
| actors | The cast of each movie. Each actor/actress is separated by a semicolon without space. |

## Open Source Tools

### requests

- Requests is a Python HTTP library, released under the Apache2 License. The goal of the project is to make HTTP requests simpler and more human-friendly. The current version is 2.21.0.
- In our crawlers, we used this tool to access the pages from IMDb and TMDb that we want to crawl.

### BeautifulSoup

- Beautiful Soup is a Python package for parsing HTML and XML documents (including having malformed markup, i.e. non-closed tags, so named after tag soup). It creates a parse tree for parsed pages that can be used to extract data from HTML, which is useful for web scraping.
- In our crawlers, we used BeautifulSoup to parse the pages that we want to crawl into a parse tree. Then we use the parse trees it produced to extract the information that we want.

### multiprocessing

- Multiprocessing is a package that supports spawning processes using an API similar to the threading module. The multiprocessing package offers both local and remote concurrency, effectively side-stepping the Global Interpreter Lock by using subprocesses instead of threads. Due to this, the multiprocessing module allows the programmer to fully leverage multiple processors on a given machine.
- In our crawlers, we used 8 processes to crawl the two data sources parallely.