大数据环境下信息抽取模板自动发现与聚类

计 92 丘骏鹏 2009011282

指导老师：朱小燕 郝宇

提纲

选题背景和动机

工作内容和思路

工作进度安排

参考文献

提纲

背景

- 已经获取到海量的新闻、博客、论坛等网页原始数据，需要从中提取结构化的信息
- 从已有数据中抽取模板，利用模板去抽取相似网页中的信息。
- 新的网页可能采取新的模板，需要自动检测，分类和抽取这些新的模板

动机

- 一个网站可能采用多个模板，如果采用人工标注模板的办法工作量很大
- 前期同学的工作主要基于一个网页检测其中的模板，没有利用大数据的优势
- 充分利用数据的冗余性来进行模板的抽取
- 目标：提取结构化信息，如新闻中的标题和正文，博客的标题和内容等，用于后续的处理。

```
<document>
  <news>
    <title>foobar</title>
    <content>blablabla</content>
  </news>
</document>
```

相关工作

- 计算 HTML 文档结构相似度
  - 基于 DOM Tree 本身的方法：Tree Edit Distance[7]。特点：直接对树结构进行操作，算法复杂度大，不适用于大量数据的处理。
  - 基于 Path 集合：每个节点的路径是根节点到该节点的序列，将 DOM Tree 用路径集合表示，在两个网页的路径集合上计算相似度 [2, 5, 6]。
  - 基于 Tag 集合或序列：直接计算两个网页的 Tag 集合的 Jaccard 相似度，或者利用树的遍历将 DOM Tree 转化为 Tag 序列，然后通过最长公共子序列等方法计算相似度 [2, 5, 8]。
  - 基于傅里叶变换：将文档结构转化为一个序列，将其视为时序序列，用傅里叶变换变换到频域计算幅度差别 [2, 4]。

相关工作

- 网页聚类算法
  - 基于划分的聚类。主要用的是 k-medoids，但需要已知模板个数，对于本问题不太适用。
  - 层次聚类。这种聚类算法不需要预先知道类的个数，速度较快。
- 模板提取算法
  - 无监督方法：利用网页结构和内容的重复出现发现模板，后期需要人工指定语义 [1]。
  - 半监督方法：利用少量标注，进行学习，然后抽取出模板。学习过程可以考虑语义的信息 [3]。

提纲

工作内容（输入）

- 文档集合

```
<html>                    |  <html>
  <body>                  |    <body>
    <h1>Title1</h1>       |      <h1>Title2</h1>
    <p>Content1</p>       |      <p>Content2</p>
  </body>                 |    </body>
</html>                   |  </html>
--------------------------|--------------------------
<html>                    |  <html>
  <body>                  |    <body>
    <div>                 |      <div>
      <div>foo1</div>     |        <div>foo2</div>
      <div>bar1</div>     |        <div>bar2</div>
    </div>                |      </div>
  </body>                 |    </body>
</html>                   |  </html>
```

工作内容（输出）

- 抽取的模板 1

```
<html>
  <body>
    <h1>?</h1>
    <p>?</p>
  </body>
</html>
```

- 抽取的模板 2

```
<html>
  <body>
    <div>
      <div>?</div>
      <div>?</div>
    </div>
  </body>
</html>
```

- 对比 (Python 的 Django 框架)

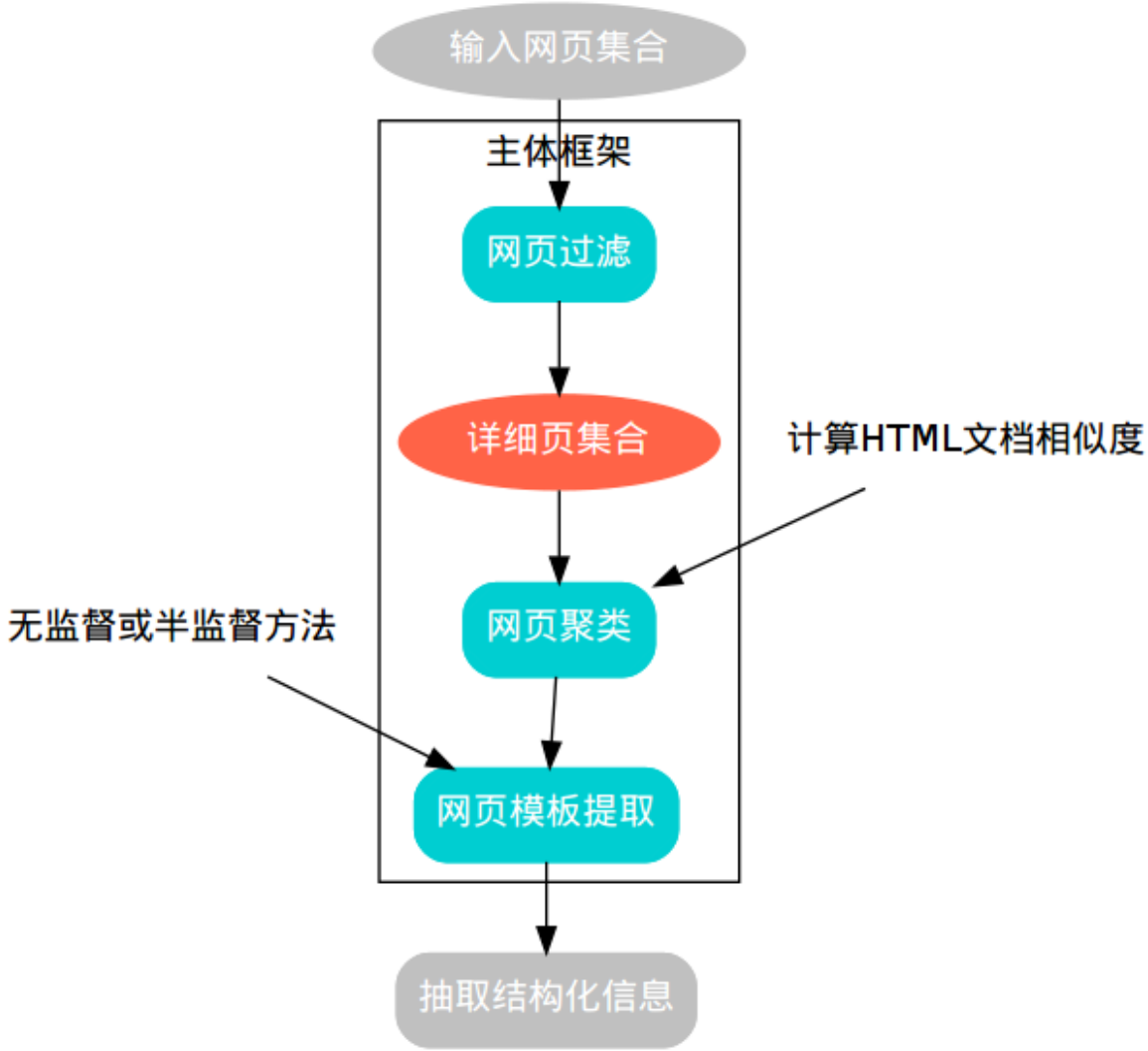
```
<html>
  <body>
    <h1>{{ news.title }}</h1>
    <p>{{ news.content }}</p>
  </body>
</html>
```

整体思路

- 实验的数据量决定了设计的算法不能太复杂，但同时数据的冗余也可以弥补算法的粗糙性，从而也能获得较好的效果。
- 每一步可以先采用一些复杂度较低的算法，评价其运行效果和时间，然后再进行改进
- 从前面可以看到，模板和静态 HTML 文档类似，不同的是模板有不变部分和变化部分，但两者可以用统一的方法进行表示
- 同时需要考虑可以动态增加模板和进行并行化处理

框架

整体框架示意图

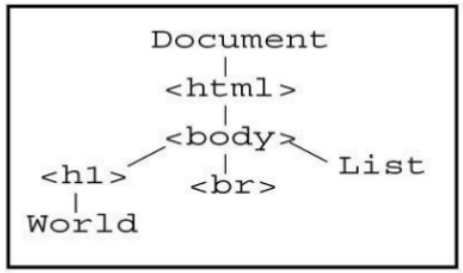

输入网页集合
主体框架
网页过滤
详细页集合
计算HTML文档相似度
无监督或半监督方法
网页聚类
网页模板提取
抽取结构化信息

模块设计

- 网页过滤
  - 网页中含有目录页和详细页，需要过滤掉目录页
  - 要能快速进行计算，不需要过于复杂的机器学习算法
  - 可以采用 URL 特征结合一些文本特征，比如目录页一般是一些超链接列表，且一般都是短文本
- 聚类
  - 计算相似度可以采用基于 Path 集合或者 Tag 集合的方法，并可以利用一些已有的集合相似度的近似算法，降低计算复杂度
  - 采用层次聚类方法进行模板的自动聚类
  - 初步计划参照 [2, 5, 6] 的方法进行实现

HTML 文档的表示（模板类似）

- DOM Tree



- Path 集合

```
| Document\<html>                    |
| Document\<html>\<body>             |
| Document\<html>\<body>\<h1>        |
| Document\<html>\<body>\<h1>\World  |
| Document\<html>\<body>\<br>        |
| Document\<html>\<body>\List        |
```

- Tag 序列

```
Document,<html>,<body>,<h1>,World,<br>,<List>
```

相似度计算

- 基于 Path 集合
  - Common Path

$$d_{CP}(D_1, D_2) = 1 - \frac{|p(D_1) \cap p(D_2)|}{\max(|p(D_1)|, |p(D_2)|)}$$

  - Common Path Shingles

$$d_{CPS}(D_1, D_2) = 1 - \frac{|S(D_1, w) \cap S(D_2, w)|}{\max(|S(D_1, w)|, |S(D_2, w)|)}$$

  - MDL[6]
    利用路径 -文档矩阵进行计算，较为复杂，根据实验效果决定是否使用

相似度计算

- 基于 Tag 序列
  - Tag Vector

$$d_{TV}(D_1, D_2) = \sqrt{\sum_{i=1}^{N}(v_i(D_1) - v_i(D_2))^2}$$

  - Longest Common Tag Subsequence

$$d_{LCTS}(D_1, D_2) = 1 - \frac{|lcts(D_1, D_2)|}{\max(|D_1|, |D_2|)}$$

模块设计

- 模板抽取
  - 可以考虑直接利用聚类的结果（例如，一个类中的文档之间的共同路径集合），利用路径集合可以直接表示文档的模板结构
  - 或者采取一些标注，结合一些已有的模板检测算法进行模板的抽取
- 优化和改进
  - 在计算相似度时加入其他特征，和结构特征共同考虑。内容特征，比如文本的行间分布，tag 的属性值，比如某些 div 的 id，CSS 类名
  - 考虑算法的并行性，部署到 Hadoop 集群上，加快算法运行速度

提纲

日程安排

- 1-4 周：任务分析，文献阅读，研究算法
- 5-8 周：网页过滤，网页聚类和网页模板提取模块初步实现
- 9-12 周：算法修正，系统改进，结果分析
  - 是否需要改进相似度算法（包括加入新的特征，改变计算方法）
  - 如何设计并行化计算
  - 如何有效评价系统的功能
- 13-15 周：论文撰写

提纲

参考文献 I

[1] Arvind Arasu and Hector Garcia-Molina.
Extracting structured data from web pages.
In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 337–348. ACM, 2003.

[2] David Buttler.
A short survey of document structure similarity algorithms.
United States. Department of Energy, 2004.

[3] Andrew Carlson and Charles Schafer.
Bootstrapping information extraction from semi-structured web pages.
Machine Learning and Knowledge Discovery in Databases, pages 195–210, 2008.

[4] Sergio Flesca, Giuseppe Manco, Elio Masciari, Luigi Pontieri, and Andrea Pugliese.
Detecting structural similarities between xml documents.
In *Proc. of the Inter. ACM SIGMOD Workshop on The Web and Databases (WebDB)*, pages 55–60, 2002.

[5] Thomas Gottron.
Clustering template based web documents.
Advances in Information Retrieval, pages 40–51, 2008.

[6] Chulyun Kim and Kyuseok Shim.
Text: Automatic template extraction from heterogeneous web pages.
Knowledge and Data Engineering, IEEE Transactions on, 23(4):612–626, 2011.

参考文献 II

[7] Davi De Castro Reis, Paulo B Golgher, ASd Silva, and AF Laender.
Automatic web news extraction using tree edit distance.
In *Proceedings of the 13th international conference on World Wide Web*, pages 502–511. ACM, 2004.

[8] 潘有能.
XML 挖掘：聚类、分类与信息提取.
浙江大学出版社, 2012.