Data mining: Role of TEX files

Manjusha Susheel Joshi
Bhaskaracharya Institute of Mathematics
Pune 411004, India
manjusha dot joshi (at) gmail dot com
http://www.bprim.org

1 Background

In a recent data mining project, I was trying to understand how to extract important words from a document. I realised that while writing a document, an author usually emphasizes important words by using italics, bold face, underlining, or quotes. This is a general observation for electronic documents.

While working on the project to try to find out appropriate information from the document set, I was looking for a better file format. In this regard, the TEX file format attracted my attention.

Nowadays most research journals accept TEX files from authors. Journals then put pdf files of the articles or abstracts on the web site, or submit them for printing.

$$\text{Authors} \overset{\text{TEX}}{\Longrightarrow} \text{research journals} \overset{\text{PDF}}{\Longrightarrow} \text{User}$$

At present, the research journals typically classify papers by the year of publication or volume of the issue in which a given paper is published.

Almost always, users have access only to PDF files, and the journals do not publish the TEX sources of the articles in any way.

2 Situation

Journals mostly have their own style files that take care of their abstract formatting, section heading style, headers, footers, and so on. They generally support keywords, citation, index, content, etc. All these features make the TEX file a very special document — special in the sense that one can extract 'feature words' from the document relatively easily.

Specifically, words in the index, abstract, section headings, and emphasized words in the document body are words which we can call *feature words* of the document. So for such TEX files, these feature words can be extracted, and submitted along with the TEX file to the journal's website.

Now, suppose we have available a collection of such TEX files for a year. Then from all the feature words associated with the TEX files, a program can collect feature words, understand which word is from which group and make groups of these documents based on clustering techniques (`http://en.wikipedia.org/wiki/Data_clustering`).

When users access the particular year of the journal, they can also see the overall topics easily, and a large set of keywords to help navigate through the articles. Even though authors provide keywords now, they usually merely highlight the topic or main theme of the article. Here we are considering more feature words from the document.

If a user asks for some particular keyword, since all the articles are already grouped according to their topics, a search program can show the user corresponding articles, by looking at the feature word data. Thus, searches can be faster and better when TEX documents are available.

$$\text{User} \overset{\text{Query}}{\Longrightarrow} \text{Journal website} \Longrightarrow \text{Specific paper}$$

3 Why TEX files and not PDF in general?

1. PDF files are rather heavy in size, while TEX files are light.
2. One can collect feature words when the file is in TEX format. TEX files are plain text, so rules to process them are fairly easy to design. For instance, we can find boldface words in the TEX file with the rule 'Search for the pattern '`{\bf `' and save words until the matching `}`. Another example: 'Ignore words starting with `\`'. Ultimately we can collect the actual content of the document. Once collection is over, we do not need to process the TEX file again.
3. Text extracted from PDF files often doesn't understand 'fi' or 'ff' ligatures properly; moreover, Greek letters $\alpha, \beta, \ldots$ are not understood by present text extractors.
4. For figures, text extractors typically find ASCII codes instead of text; many times we have observed garbage when a figure is present in the text file.
5. Submission of the feature word file along with the PDF file is possible. In fact, each PDF file can be represented by the collection of feature words for that file. When a user makes a query, instead of searching in the entire PDF file, searching can be done in only these keyword collections, which would be considerably faster and produce more relevant results.

4 Why TEX files and not other markup?

Consider HTML files: for line breaks, paragraphs, even for extra space, explicit commands are required, which makes the source full of commands which do not hold any information with respect to the content of the article.

On the other hand, if we look at a typical TEX file, it uses fewer formatting commands in comparison to an HTML file. For example, paragraphs are indicated simply by blank lines. Thus, there is less disturbance when extracting text from the source.

5 User wishes

Suppose a user wants to search for a general concept in a repository. This would usually require a full text search, which is time-consuming. To speed up the search, what if the documents were already clustered by subject or concepts?

If we can classify documents beforehand, the search process could be more like this:

- Do we have the query results already saved? If so, return them.
- If not, the query is made against the 'cluster representatives', described below, to return the appropriate documents.

6 Challenges

This leads to the question of how to form such concepts or clusters beforehand, without knowing the query. How should such clusters be represented, and how do we find the representatives?

In a document source file, the author highlights words with additional, perhaps invisible, markup. These words presumably help to represent the document. How do we capture these representatives?

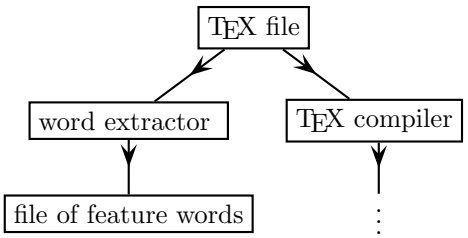
From a PDF file, text extraction is not simple, as we have seen. Another common format, RTF (rich text format), is even less straightforward.

HTML format is somewhat better, although the additional commands mentioned above complicate the job.

There are many commands with which the TEX compiler collects important information from the TEX file to highlight in the output file. Some well-known examples:

- For emphasis, LATEX uses commands such as `{\bf ...}`, `\textbf{...}`. So we can identify emphasized words fairly easily.
- To add a word to an index: `\index{word}`.
- For section headings: `\section{...}`.

So we are assuming that the (LA)TEX compiler understands that some words are to be highlighted for some reason. We are interested in these words given some special importance by the author. Thus, the general picture looks like this:



Suppose there is a file of 2000 words, out of which we found say 300 words that are special words. Clearly we can increase speed of searches.

7 Looking ahead

We can cluster documents based on their feature words. For example, we might try this rule:

> *If documents have more than 70% of the feature words in common, group them together.*

Now clusters will be defined according to their common feature words, 70% in this case. The remaining 30% of words for each document in the cluster can be a secondary representative of the cluster.

To handle a query, we can search in the primary representation of the clusters and if not found, search in secondary representations. This will make our search even faster.

Journals can provide this facility to their users, and it would be useful for other areas where TEX files are in use.