Pattern Based Approaches to Pre-processing Structured Text: A Newsfeed Example

Paul Bogg

University of Technology Sydney
plbogg@it.uts.edu.au

Abstract. Text documents presenting a structured format allow the reader the ability to quickly run their eye over the page and read information relevant to them. By presenting information in this manner, the author allows ease of information extraction by the reader. If the structure used throughout the document involves a pattern or set of patterns to describe the text, then if text pre-processing methods can identify the patterns involved, those methods can also extract the same text as that of the naked eye. This extraction of meaningful text can then be used for further text mining applications. This paper describes a text pre-processing program that identifies text patterns and extracts the appropriate text.

1 Introduction

Text mining is the process of identifying and extracting information and/or statistics from text documents that are of interest [2]. Before the mining activity begins on a document, the text is prepared so that the raw text presented to the text mining algorithm is precisely the interesting information that the mining process is seeking (and none that might skew the information incorrectly). This step is called the *pre-processing* stage [3].

The pre-processing stage is typically the most complex stage in a text (or data) mining exercise. Structured text documents contain identifiable patterns of text that may be extracted by a program for the purpose of pre-processing. If these patterns can be identified correctly, and consequently so too the interesting text, then we can speed up the whole text mining process significantly [5]. A technique to achieve this is described.

2 Process

To demonstrate the theory behind the pattern identification approach to pre-processing, we will use email newsfeeds as our base type of text document. An email newsfeed is a purely text based news document sent to a user to brief him or her on daily events around the world (ie, news). These newsfeeds are emailed to a user on a regular basis informing them of potentially useful information. Generally, most

internet news paper sites provide a free email newsfeed service. For example <http://www.usatoday.com> and <http://www.smh.com.au>.

An example of a newsfeed is shown below in Courier New format:
(extract take from the New York Times email newsfeed received on the 10th September 2001).

```
NATIONAL
=====
Black Families Resist Mississippi Land Push

Mississippi's right to eminent domain is being
challenged by
black property owners whose claims date back to
Reconstruction.
http://www.nytimes.com/2001/09/10/national/10LAND.html?to
daysheadlines

-----

Sacramento Shooting Suspects Dies in Gun Battle

Joseph Ferguson, sought in the killing of 5 people in
Sacramento,
reportedly shot himself after a violent gun battle
with police officers early today.
http://www.nytimes.com/aponline/national/AP-
Sacramento-Shootings.html?todaysheadlines

-----
```

Here, 2 articles of news content are described in the form of a heading, body and corresponding URL. We adopt the hypothesis that the pattern that is to be identified is: heading, body and URL. If this hypothesis is valid then we should be able to identify the common structure involved and so too extract the text information based on that pattern.

2.1 Classification

The first step in identifying patterns within a document is to classify certain sections of text. This is the primary initialisation step that any text mining program should perform if it is to extract information from a given document [4]. Classification of parts of the document may require some initial prior knowledge of possible document layout (depending on the type of document), but generally, for most purposes, domain experts are not needed.

Each unique part of the document which can be identified is called an 'Identifier'.

Given a sample chunk of text, we identify two base types of text identifiers: Single line text, and Multiple line text. Short of identifying words and concepts within the text (which belongs more to the mining stage – remember all we want is to pre-process), this is the highest level of abstraction that is necessary to consider. *Single line text* is a solitary line within the document neither preceded nor followed by another line of text (for example, a heading). *Multiple line text* is a block of 2 or more lines of consecutive text.

Using our sample newsfeed above, an example of single line text is:

Black Families Resist Mississippi Land Push

And an example of multiple line text is:

Mississippi's right to eminent domain is being
challenged by
black property owners whose claims date back to
reconstruction.

Furthermore, in line with our hypothesis that is specific to the domain of newsfeeds, we identify the third type of text that we require: the URL. This chunk of text will be a line with a URL pointing to a co-related news article that may be in a different format (for example HTML). In our sample newsfeed above, an example of a URL is:

[http://www.nytimes.com/2001/09/10/national/10LAND.html?
todaysheadlines](http://www.nytimes.com/2001/09/10/national/10LAND.html?todaysheadlines)

So now we have all three primary identifiers that can be used to classify patterns within the documents, and to identify the sections that we would like to keep (ie, patterns containing any or all of these identifiers).

To supplement these three primary identifiers, we introduce identifiers that describe the surrounding document sub sections. There are three of these secondary identifiers:

- Space – for lines within the document which contain only space and newline characters
- Potential Boundary – for lines which contain significant percentage of non-alphanumeric characters, they have the potential to be a boundary
- Unknown – for lines which there is can be no classification given

Together these six identifiers are sufficient to identify patterns in newsfeed documents as is shown below.

2.2 Identifying Patterns

If we look at the sample newsfeed, we intuitively see that there is some kind of pattern occurring, even if we cannot spot it. As the first step in the identification of the pattern, we illustrate the classification of the newsfeed in terms of the identifiers discussed in the previous section. Thus, we can now re-write the document in terms of these identifiers – the result of this operation is:

```

FREE_TEXT
POTENTIAL_BOUNDARY
FREE_TEXT
SPACE
MULTIPLE_TEXT
URL
SPACE
POTENTIAL_BOUNDARY
SPACE
FREE_TEXT
SPACE
MULTIPLE_TEXT
URL
SPACE
POTENTIAL_BOUNDARY

```

From this representation, the patterns are more obvious. Instinctively, we see that the main pattern we can use to extract the correct text is FREE_TEXT – SPACE – MULTIPLE_TEXT – URL. However, there is another less obvious case where by we can use the POTENTIAL_BOUNDARY as a kind of tag to identify when an extractable pattern may exist, and consequently attempt to parse the text between each one. We will discuss this later, but for now, we follow the initial approach.

2.3 Initial Approach

We start identifying main patterns through a process of first grouping sequential identifiers together into 2-tuples, and then removing any 2-tuple that does not occur more than once in the document. To further reduce the number of 2-tuples, we finally remove any 2-tuple that does not contain an identifier that has been flagged (we are looking for either FREE_TEXT or MULTIPLE_TEXT. URLs are supplementary). Thus, after the first run through our classified newsfeed the following 2-tuples remain:

```

FREE_TEXT – SPACE (occurs 2 times)
SPACE - MULTIPLE_TEXT (occurs 2 times)
MULTIPLE_TEXT – URL (occurs 2 times)

```

Once again, we can start to see the main patterns beginning to emerge. Using these new 2-tuples, the next step is to attempt to further ‘drill-down’ the pattern to come to a list of 3-tuples. Once this new list has been constructed, to improve the chance of

reaching a final dominant pattern, any 3-tuple that overlaps another 3-tuple is removed, including itself (with any overlaps resolved by the keeping the 3-tuple which occurs generally before the other. If a pattern begins to overlap itself, then we cease ‘drilling-down’ for that pattern). The following is an example of this phenomenon.

FREE_TEXT – SPACE – MULTIPLE_TEXT (occurs 2 times)

SPACE – MULTIPLE_TEXT – URL (occurs 2 times)

MULTIPLE_TEXT – URL – SPACE (occurs 2 times)

Is the list of 3-tuples that is obtained by ‘drilling-down’ the 2-tuple patterns. To identify potential patterns, any overlapping patterns are removed. This results in a more refined list:

FREE_TEXT – SPACE – MULTIPLE_TEXT (occurs 2 times)

MULTIPLE_TEXT – URL – SPACE (occurs 2 times)

If we continue this distillation of the patterns within the document, further drilling-down and refining, we will come to a point where we can identify a dominant pattern (or set of patterns) which we can use to extract meaningful text. A pattern is *generally dominant* when it either is repeated a number of times after being drilled-down, or is the solitary pattern remaining after refinement. Using our example above, post-refinement, the following pattern dominant is derived:

FREE_TEXT – SPACE – MULTIPLE_TEXT – URL – SPACE (occurs 2 times)

As you can see, this pattern mirrors (or very similarly resembles) the same one you instinctively viewed earlier on from the original email newsgroup. Using this pattern, the meaningful information (or “articles”) is extracted and then passed to the text mining procedure.

2.4 Alternative Approach

As we mentioned in Section 2.2, there is an alternative approach to extracting information from the structured text. In our initial approach (described in Section 2.3), we attempted to deduce a primary pattern (or set of patterns) that contained the text we wanted to extract and pass to the text mining procedure. If that approach found a pattern, then we could use it as the basis for identifying an “article” of information.

As an alternative to that approach, we now utilise the properties of “boundaries” as flags for identifying where meaningful information occurs. Within a structured text, boundaries may exist between articles which identify when a new article commences and finishes. If we look at our original email newsgroup, we can identify several potential boundaries:

NATIONAL

===== ← *Boundary*
 Black Families Resist Mississippi Land Push

Mississippi's right to eminent domain is being
 challenged
 by black property owners whose claims date back to
 Reconstruction.

[http://www.nytimes.com/2001/09/10/national/10LAND.html?to
 daysheadlines](http://www.nytimes.com/2001/09/10/national/10LAND.html?to

 daysheadlines)

----- ← *Boundary*

Sacramento Shooting Suspects Dies in Gun Battle

Joseph Ferguson, sought in the killing of 5 people in
 Sacramento, reportedly shot himself after a violent
 gun
 battle with police officers early today.
[http://www.nytimes.com/aponline/national/AP-
 Sacramento-Shootings.html?todaysheadlines](http://www.nytimes.com/aponline/national/AP-

 Sacramento-Shootings.html?todaysheadlines)

----- ← *Boundary*

The boundaries are shown above with the italicised label “← *Boundary*”. If we identify the main boundaries within the document, then by analysing the text between the boundaries for common patterns, we are also able to extract meaningful text.

3 Application and Results

By incorporating the two approaches (in section 2.3 and section 2.4), we have constructed a program that takes a structured text document, parses it, attempts to identify dominant patterns that capture meaningful text, and consequently parses the text into “articles” ready for use by the text mining procedure.

We sampled our program across 300 email newsfeeds from a variety of different sources from the USA, UK and Australia.

The results are shown in Table 1 below.

An *article* is anything that appears repetitively within a pattern and is extracted from the document by the pre-processing procedure. In this case, for email newsfeeds, an article consists primarily of the identifiers FREE_TEXT, MULTIPLE_TEXT, and/or URL. For example, for the sample emails received from boston.com, the information within each article extracted was typically FREE_TEXT and URL.

Table 1. Results obtained by the pre-processing program.

Newsfeed Subscription	Total Articles	Articles Identified	% Bad Articles	% missed	Articles
boston.com	482	464	4.310344828	7.883817427	
guardianunlim ited.co.uk	269	240	7.083333333	17.10037175	
usatoday.com	252	273	7.692307692		0
– tech					
usatoday.com	230	230	0		0
– regular					
age.com.au	816	827	5.07859734	3.799019608	
afr.com.au	370	671	44.56035768		0
smh.com.au	205	251	21.9123506	4.390243902	
ltnews_daily (smh.com.au)	218	285	23.50877193		0
nytimes.com. au	395	423	9.692671395	3.291139241	
			Average Bad Articles	Average Articles Missed	
	3237	3664	13.75985942	4.051621325	

Total Articles – total number of proper articles identified manually from emails sent by this provider

Articles Identified – total number of articles identified by the program

%Bad Articles – unwanted articles which were extracted by the program

%Articles missed – wanted articles which were not extracted by the program

As the results in Table 1 show, the program was overall quite accurate in the number of articles extracted. Of all the articles extracted, only about 4% of the articles viewed as being wanted (in terms of meaningful text) were not identified by the program. One of the goals of the exercise was to explore the use of quite general patterns, giving leniency towards more generic patterns that picked up wanted articles rather than specific, refined patterns that were more accurate in removing bad articles. This is reflected in the Average Bad Articles statistic of nearly 14% - this means almost 14% of all articles found were not wanted.

In a closer inspection of these two statistics, much of the inaccuracy of this program could be attributed to a small handful of documents without proper structure. Occasionally, the odd newsfeed would not have a proper structure from which to extract a reasonable pattern and consequently the entire batch of articles extracted would be unwanted. Likewise, many of the bad articles found by the program were infact advertisements for differing products and were not placed in a distinguishable location within the email.

It was interesting to note that the accuracy of the program was closely related to the consistency and rigidity of the structure involved in representing the email text newsfeed. The more structured emails (such as the ones from *usatoday.com*) presented an easier task of extraction for the program. Meanwhile, newsfeeds with a more ad-hoc text structure (such as *smh.com.au*) presented a more guess-like approach to pattern identification.

4 Future and Direction

Based on the results, the primary improvements to future attempts in utilising patterns for pre-processing is to reduce percentage of bad articles found and to improve the percentage of articles missed.

One way to screen articles for meaningful text or not would be to involve some kind of light weight Natural Language Processing [6]. As mentioned earlier, in our example for newsfeeds, many of the bad articles found were actually advertisements placed within the text. We could possibly screen for advertisements[1] in this case by analysing patterns of words, searching for the articles' 'audience'. In the newsfeeds case, the majority (if not all) articles found were aimed at the general reader, and did not refer to them specifically. This is unlike an advertisement where the language is slightly catered for a more personal feel.

5 Conclusion

Pre-processing is an important step in the overall data mining process. The ability to extract the appropriate information is vital for subsequent algorithms and mining activities. Domain experts are excellent for writing programs for specific domains, though this is very cumbersome.

Two different approaches to identifying patterns have been described. Each with its own strength. These two approaches have been combined in a program that identifies patterns from newsfeed data. Despite the simplicity of these approaches the program performs rather well. Structure text presents an opportunity for programs to utilise the patterned approach in setting out text to extract meaningful information. If we can identify the dominant patterns, and can extract the text according to them, then perhaps we can unburden the load placed on creating data and text mining applications.

References

1. Domingos, P. and Richardson, M. Mining the Network Value of Customers, In: Proc. 7th Int. Conf. on Knowledge Discovery and Data Mining, San Francisco, CA, 2001, ACM Press, pp. 57–66
2. Han, J. and Kamber, M. Data Mining: Concepts and Techniques. Morgan Kaufmann, San Francisco, CA.
3. Hand, D., Mannila, H. and Smyth, P. Principles of Data Mining. The MIT Press, Cambridge, MA, 2001.
4. Hilderman, R. J. and Hamilton, H. J. Knowledge discovery and interestingness measures: A survey. Technical Report CS 99-04, Department of Computer Science, University of Regina, October 1999.
5. Leake, D. and Kolodner, J. Learning through case analysis. In: Encyclopedia of Cognitive Science. Macmillan, London 2001.
6. Lin, D. and Pantel, P. DIRT – Discovery of Inference Rules from Text. In Proc. 7th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining KDD 2001, pp. 323–328.