An Approach of XML-ifying the Crude Corpus in the Field of Opinion Mining

Debnath Bhattacharyya[1], Kheyali Mitra[1], Minkyu Choi[2], Rosslin J.Robles[2], and Debashis Ganguly[1]

*[1]Computer Science and Engineering Department
Heritage Institute of Technology
Kolkata-700107, India
{debnathb,kheyalimitra,DebashisGanguly}@gmail.com*

*[2]Hannam University, Daejeon – 306791, Korea
freeant07@naver.com, rosslin_john@yahoo.com*

Abstract

This paper is meant for an easy approach for XML ifying of crude corpus in the field of Opinion Mining. The XMLification is done based on regular expressions. Corpus is the plural form of 'corpora'. It is nothing but the collection of linguistic data. In this proposed work, the corpus is reviews posted on web sites; more specifically some product reviews. The reviews or the opinions are in the html files which are collected from sites like Cnet.com, Epinions.com, Amazon.com, ebay.com etc. After getting the crude corpus of html files, it is polished further to get only the required part of review details from that web page and thus removes the rest. This corpus is processed again and yields ultimate output in the form of XML files which contains only the important parts of the review details from raw html page. These XML files are ready to be used for further steps of Opinion Mining like parts of Speech(POS) tagging or any kind of language processes for machine learning process..

Keywords: *Crude corpus, language processing, regular expression, XML, parts of speech tagging.*

1. Introduction

Using of collections of text in language study is not a new idea. The work began on making lists of all the words in a certain text which is called concordance. In most of cases, word frequencies are counted from single texts or from collections of texts and produced lists of the most frequent words. The term 'corpus linguistics' might be unknown in past but most of the works were quite similar to the kind of activities based on corpus. The main difference is the automation which is done today using computers. The versatility in technological development, along with the translations available in different languages has lead to use of this corpus for specific machine learning mechanism as well as various automatic translation applications. But the prime objective of researchers as well as the naive users is to give a fast developing technique of machine learning systems which must be both exact and effective. To create exact dataset for a particular purpose is a very tedious job due to the crisis of exact corpus regarding respective research work. Opinion mining and sentiment analysis deal with the computational treatment of human opinions, sentiment, and subjectivity in text. And for this, accumulating proper corpus from normal web based search engines are not so easy to get. In our case too, we did the tedious job for having raw html files having opinions of reviewers. And after that we have refined them according to our need. To make others job easy we have proposed our algorithm so that the same task can be done automatically in future. It will create a polished corpus in the form of XML files which simplifies the POS tagging activities or others language processing activities.

2. Terminology

The terms those are used in this paper, are explained in the following.

2.1. Corpus

Corpus is a collection of texts in electronic form (in the case of the spoken language - a transcription of speech), used for linguistic research. A special search engine facilitates work with this corpus. It will aid users in finding words and collocations in context and determine their frequency in the corpus and their original text source .It also enables further processing of the found data (alphabetical classification, etc.). Some corpora can be searched also according to parts of speech.

2.2. Natural Language Processing

Natural language processing (NLP) is a subfield of artificial intelligence and computational linguistics. It studies the problems of automated generation and understanding of natural human languages.

Natural-language-generation systems convert information from computer databases into normal-sounding human language. Natural-language-understanding systems convert samples of human language into more formal representations that are easier for computer programs to manipulate.

2.3. Linguistics

Linguistics is the scientific study of language. Linguists focus on describing and explaining language and are not concerned with the prescriptive rules of the language. Linguists are not required to know many languages and linguists are not interpreters.

The linguist is there to identification of the common elements of all languages. The linguist then tries to place these elements in a theoretical framework that will describe all languages and also predict what cannot occur in a language.

2.4. Regular Expression

In computing, regular expressions provide a concise and flexible means for identifying strings of text of interest, such as particular characters, words, or patterns of characters. Regular expressions (abbreviated as regex or regexp, with plural forms regexes, regexps, or regexen) are written in a formal language that can be interpreted by a regular expression processor, a program that either serves as a parser generator or examines text and identifies parts that match the provided specification.

2.5. Parsing

In computer science and linguistics, parsing, or, more formally, syntactic analysis, is the process of analyzing a sequence of tokens to determine their grammatical structure with respect to a given more or less formal grammar.

Parsing is also an earlier term for the diagramming of sentences of natural languages, and is still used for the diagramming of inflected languages, such as the Romance languages or Latin. The term parsing comes from Latin pars, meaning part of speech

2.6 Opinion Mining

Opinion is nothing but what other thinks, their views on a certain thing. Opinion Mining is a recent discipline at the crossroads of information retrieval and computational linguistics which is concerned not with the topic a document is about, but with the opinion it expresses [1][2].

.

3. Related Works

The amount and diversity of corpus related research projects and groups are great and it has been started a couple of decades ago. In this section, some of the previous works in the field of corpus refinement and opinion mining are depicted. We have considered all these works at the time of our research.

David Sankoff and Henrietta Cedegren developed a program called varbrul (1979).It performed multivariate analysis on linguistic data. Data was encoded for that system using “coding strings” and the program was developed to analyze phonological data.

In the early 80’s Anthony Kroch and Don Hindle started using varbrul to analyze syntactic data. Their program manipulated coding strings to make varbrul more effective for syntactic analysis. Basically, a corpus of selected texts was assembled and then coding strings were added to each sentence by someone reading through the sentences and writing the coding strings by hand.

In 1991, Anthony Kroch and Ann Taylor started developing a syntactically annotated corpus of Middle English. This resulted in the publication of the first phase of the Penn-Helsinki Parsed Corpus of Middle English (PPCME1) in 1994. To build a corpus, linguists need parsers to describe sentence structure and labelers or “taggers” to label parts of speech [3][4]. At the time, there were no automatic parsers or labelers available that were robust enough to handle Middle English, with its spelling and word-order variations.

The second phase of the corpus, PPCME2, began in 1995. Eric Brill had written a trainable tagger. To use this tagger, the linguist would first write a training set of correctly labeled example sentences. From the training set the tagger develops a lexicon and a set of rules for assigning tags. This made it possible to include part-of-speech tags in the second version of the corpus. Once the part-of-speech tags were included, it was then possible to do automatic parsing which is based on the part-of-speech tags. The first parser the linguists used is called “fidditch”, written by Don Hindle, working on a grant Tony Kroch had to study speech and writing. The Penn Treebank Project, a Modern English corpus project under the direction of Mitch Marcus, wound up using the fidditch parser, and also developed friendly interfaces for correcting the tagged and parsed output.

Mike Collins wrote a trainable automatic parser that was extremely good for Modern English. Again, the key was that it was trainable. He was interested in seeing the training feature in action, so he did a lot of work to make the parser handle Middle English correctly. This provided parsing that was so much more accurate than fidditch that it cut the correcting time by a magnitude of 4.

Today, corpus linguistics is closely connected to the use of computers; so closely, actually, that the term 'Corpus Linguistics' for many scholars today means 'the use of collections of COMPUTER-READABLE text for language study'. The variety of corpus related research projects and groups are vast. “Automatic Mapping among Lexico-Grammatical Annotation”

(AMALGAM) is an attempt to create a set of mapping algorithms to map between the main tag sets and phrase structure grammar schemes. The AMALGAM project has developed a set of resources for qualitative comparisons between the main Part-of-Speech tag sets and phrase structure grammar schemes used in English corpus linguistics [3][4][5]. Software has been developed to tag text with up to 8 different PoS-tag schemes.

In 1993/1994, Textcorpora and Erschliessungswerkzeuge collected textual material for German, French and Italian, developed a representation for texts and markups, along with a query language and a corpus access system for linguistic exploration of the text material. Texts and analysis results are kept separate from each other, for reasons of flexibility and extensibility of the system; this is possible because of a particular approach for storage and representation.

The International Corpus of English (ICE) is an ongoing project for corpus preparation. Twenty centers around the world, compilers are engaged collecting material for the ICE corpora. Each ICE corpus will contain 1 million written and spoken words of a national variety of English. The first ICE corpus to be completed is the British component, ICE-GB. As a whole, the 20 corpora will be useful for variation studies of various kinds.

4. Our Work

Before going into the details of the algorithm proposed here for crude corpus refinement, it is better to mention about the selection of the raw text, i.e. crude corpus which is to be refined. Generally, the raw texts in this case, are in the form of html files containing users' reviews on products [6] [7]. Here we have chosen Cnet.com to collect the product reviews; the product is also specific named digital camera and cell phone. The reason behind this is only to enhance the volume of the corpus which can be done by selecting the most reviewed brands as well as products. There are individual folders for individual product reviews. Especially a tree structure is maintained for each product and even for each brand under a single product. Here in this paper, the algorithm is basically implemented over normal html files supporting css. This selected html files are stored into a folder and here, it is referred as INPUT_FOLDER and the patterns which are to be matched are termed as REGULAR_EXPRESSION.

After having the input files in hand the algorithm will do two things, the html file is to be refined first based on the given patterns. Thereafter, the refined file will be converted to xml file for the ease of storing the required information in a precise and simple manner. After proper refinement and xml file generation, according to the algorithm specified in section D, all these xml files will be saved according to the product name followed by the user name with the extension of xml. This name will be the unique identifying key for each file. And those files will be stored in a folder of the product name.

It is better to be confessed that the method for corpus refinement can be personalized, i.e., can be selected according to the user ends. But, the authors specifically suggest this specialized scheme, proposed in this paper, as because here the information are extracted and restored in a very simple way, without making any redundancy by using the unique identifying key for each file. Not only that, the xml format checking is also done during the html to xml conversion. The xml file format is chosen because it is the best way to restore the information in a very precise manner from one file and to retrieve it from it. The algorithm is given hereunder:

4.1. AEAXTCCTFOM_CORPUSREFINEMENT_MAIN ()

This is the main function in our algorithm. This function will be used to refine the html files and will call other modules of our algorithm like HTML_Refinement, XML_FileGenerator and FileMaintain.

a. Select the folder bearing the user review as INPUT_FOLDER.
b. Call AEAXTCCTFOM_HTML_REFINEMENT (INPUT_FOLDER, REGULAR _EXPRESSION) and obtain the return value as the refined html file REFINED_FILE containing only the required portion.
c. Next call AEAXTCCTFOM_XMLFILEGENERATOR (REFINED_FILE, XML_TAG) and obtaining the desired XML_FILE.
d. Call AEAXTCCTFOM_FILEMAINTAIN (INPUT_FOLDER, XML_FILE) to store those xml files within the given folder.

4.2. AEAXTCCTFOM_HTML_REFINEMENT (INPUT_FOLDER, REGULAR_EXPRESSION)

This function will take the Input_Folder as folder and the given regular expression as Regular_Expression and it will extract the required part from html file based on given regular expression.

a. Store only the files having extension html or htm of the Input_Folder.
b. Read each file.
c. Compare each line of it with the all given Regular_Expression.
d. If any match is found, write that line into a new file Refined_File. The refined part must maintain certain sequence.
e. Return the file with same name and extension.

4.3. AEAXTCCTFOM_XMLFILEGENERATOR (REFINED_FILE, XML_TAG)

This function will be used to generate the final xml file which is the final output of this algorithm. This function will take Refined_File and XML_Tag as argument and finally it will output the XML_File.

a. Store only the files having extension html or htm of the INPUT_FOLDER.
b. Select a REFINED_FILE from the set of html files available in the collection of INPUT_FOLDER.
c. Read the file.
d. For each line in the file replace the html tag by the XML_TAG if the index value of XML_TAG matches with the line number.
e. Give the name of the xml file as the Product name followed by reviewer's name.
f. Return XML_FILE.

4.4. AEAXTCCTFOM_FILEMAINTAIN (INPUT_FOLDER, XML_FILE)

This function will finally manage the xml files in a folder whose name is same as input folder name. This function will take Input_Folder, XML_File as argument and finally it will output the folder containing all xml files which are following the xml syntax.

a. Store the name of Input_Folder and create a new folder with that name.
b. Read each XML_FILE.

c. Check the syntax as well as the character set supported by xml format. If any discrepancy found, modify it.
d. Finally store all the XML_FILEs within the given folder.
e. Exit.

5. Result and Discussion

The result obtained after the execution of our proposed algorithm is shown in figure 2. This file is obtained after the execution of AEAXTCCTFOM_XMLFileGenerator function. The input file which is used here is shown in figure 1. It is the output of AEAXTCCTFOM_HTML_Refinement function where the required portion from the original html file is extracted.

5.1. Complexity analysis of the stated algorithm

For AEAXTCCTFOM_HTML_Refinement (Section 4.2): In step 1, it will check all the files in that folder whose extension is .html or htm and thus yields the time complexity = O (n).

In step 2, each line of individual file is read. So the time complexity is O (m). In step 3, for each line there will be a checking of all regular expressions. So the time complexity = O (n). So, overall we get the total time complexity is O (n*m).

For AEAXTCCTFOM_XMLFileGenerator (Section 4.3): In step 2, each html or htm file is read which yields time complexity of O (n). Again in step 3, to replace the html tags by xmls O (m) has incurred. So, overall time complexity becomes O (n*m).

For AEAXTCCTFOM_FileMaintain (Section 4.4): For step 2, each file has to be checked. So, each one requires time of O (n).

5.2. Test Result

Figure 1 is the original HTML web page containing review on digital camera (product). These types of page are our crude corpus and the regular expressions based on which the refinements are done, are selected from the source code of each pages. To make the expressions consistent, a rigorous study is done on the available source codes. And the result of this is to get the most unique words o sometimes html tags which rarely change with the design or some modification in the web pages. A few examples are:

<h1 class="title">: This is used to retrieve the heading of the page where the product name is stored.

<span class='rgr'>Written: This regular expression is used to get the user name/ reviewer name.

<b>Pros: </b>: This helps to get the con tent of Pros.

<b>Cons: </b>: This extracts the Cons.

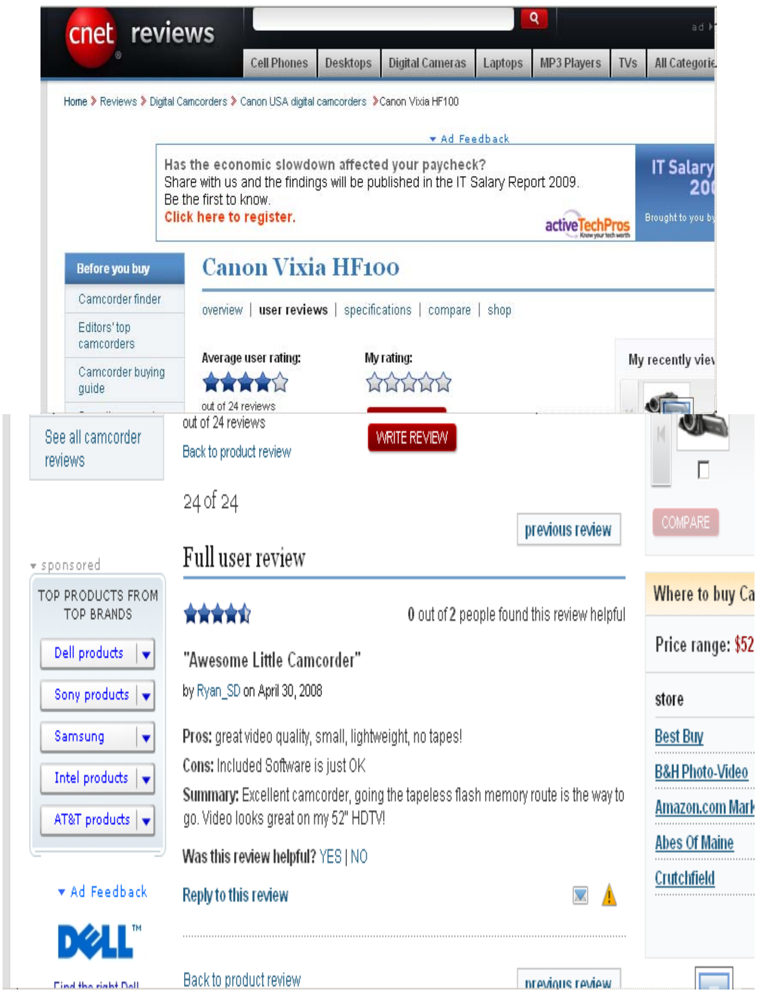
<b>The Bottom Line: </b>: This is to refer to get the bottom line.

Using these expressions, the 1st stage of refinement is done and hence produces the refined html file given in Figue2 which is the output of AEAXTCCTFOM_HTML_Refinement function. After that, this file is fed to the AEAXTCCTFOM_XMLFileGenerator function to generate corresponding xml file as shown in figure 2. Another output file is shown in Figure 3 which is taken from Epinions.com and Figure 4 shows the final xml output file.

In the first figure, the name of the product, user ratings, user name, date, pros, cons, summary are the important part of html file. So, when the file is converted to corresponding xml file the individual part is stored with in the name of the regular expression itself. This is only for making the further process easier in the actual research work. So this is our final corpus which we can use for polarity detection of opinions.

The result obtained after the execution of our proposed algorithm is shown in figure 2 and figure 4.These results are obtained after the execution of NARC_XMLFileGenerator function. The input file which is used here is shown in figure 1, figure 3. It is the output of NARC_HTML_Refinement function where the required portion from the original html file is extracted.

The AEAXTCCTFOM_HTML_Refinement and AEAXTCCTFOM_XMLFileGenerator functions are performing the tasks keeping the lopping counters same, i.e., time complexity remaining constant i.e., maximum of O (n*m) or O (n2) which in turn depends on the number of patterns that will be matched and the number of input files.



Has the economic slowdown affected your paycheck?
Share with us and the findings will be published in the IT Salary Report 2009.
Be the first to know.
Click here to register.

Canon Vixia HF100

Average user rating: out of 24 reviews

My rating:

out of 24 reviews
Back to product review

24 of 24

Full user review

0 out of 2 people found this review helpful

"Awesome Little Camcorder"
by Ryan_SD on April 30, 2008

Pros: great video quality, small, lightweight, no tapes!
Cons: Included Software is just OK
Summary: Excellent camcorder, going the tapeless flash memory route is the way to go. Video looks great on my 52" HDTV!

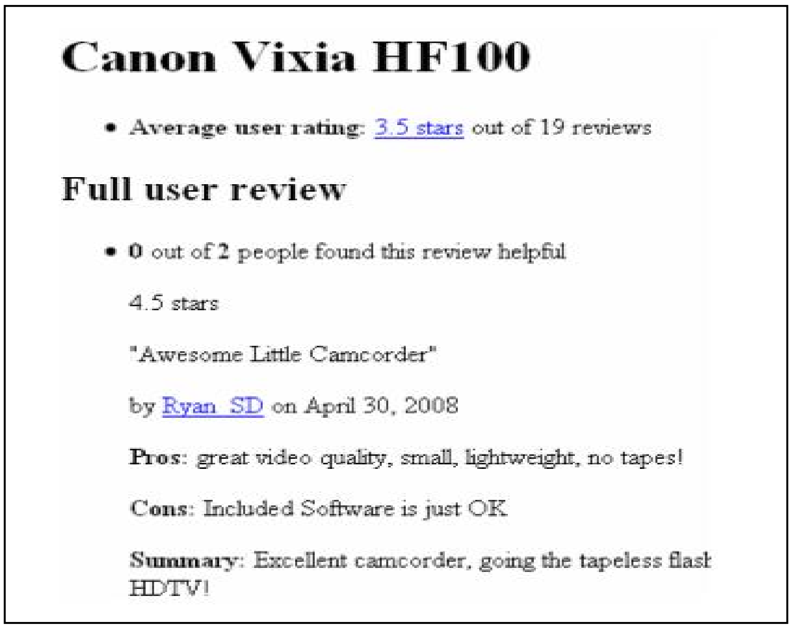
Was this review helpful? YES | NO

Reply to this review

Back to product review

Price range: $52

Figure 1. Original HTML web page containing review (cnet.com)



Canon Vixia HF100

- Average user rating: 3.5 stars out of 19 reviews

Full user review

- 0 out of 2 people found this review helpful

4.5 stars

"Awesome Little Camcorder"

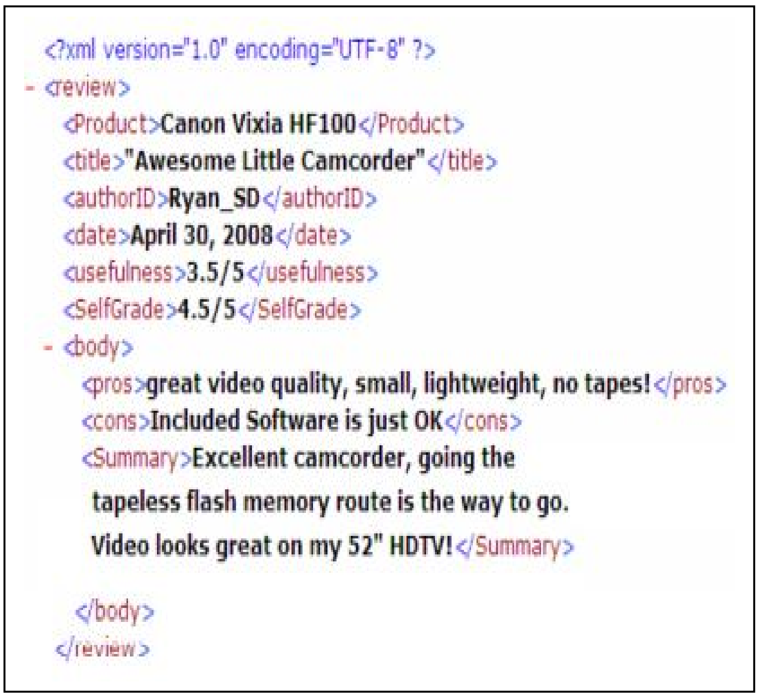
by Ryan_SD on April 30, 2008

Pros: great video quality, small, lightweight, no tapes!

Cons: Included Software is just OK

Summary: Excellent camcorder, going the tapeless flash
HDTV!

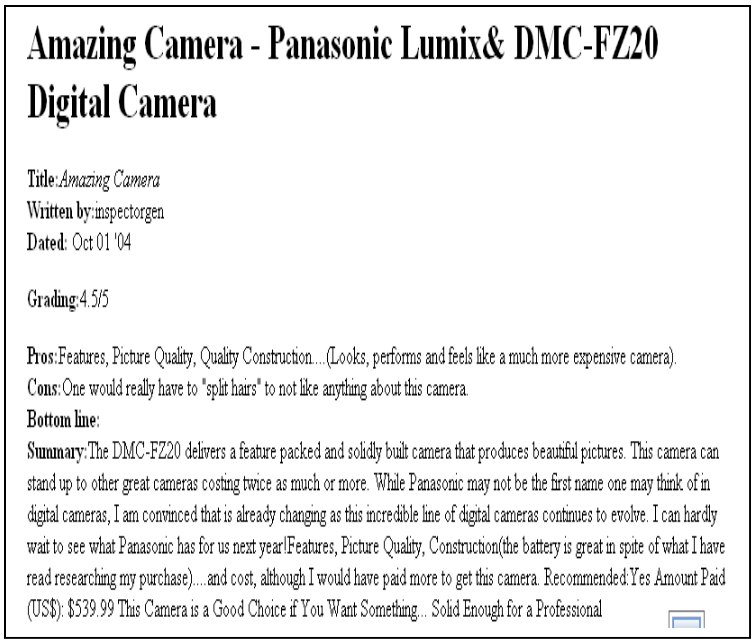
Figure 2. Output of AEAXTCCTFOM_HTML_Refinement function (Cnet.com)



```
<?xml version="1.0" encoding="UTF-8" ?>
- <review>
   <Product>Canon Vixia HF100</Product>
   <title>"Awesome Little Camcorder"</title>
   <authorID>Ryan_SD</authorID>
   <date>April 30, 2008</date>
   <usefulness>3.5/5</usefulness>
   <SelfGrade>4.5/5</SelfGrade>
 - <body>
     <pros>great video quality, small, lightweight, no tapes!</pros>
     <cons>Included Software is just OK</cons>
     <Summary>Excellent camcorder, going the
      tapeless flash memory route is the way to go.
      Video looks great on my 52" HDTV!</Summary>

    </body>
  </review>
```

Figure 3. Output of AEAXTCCTFOM_XMLFilegenerator function



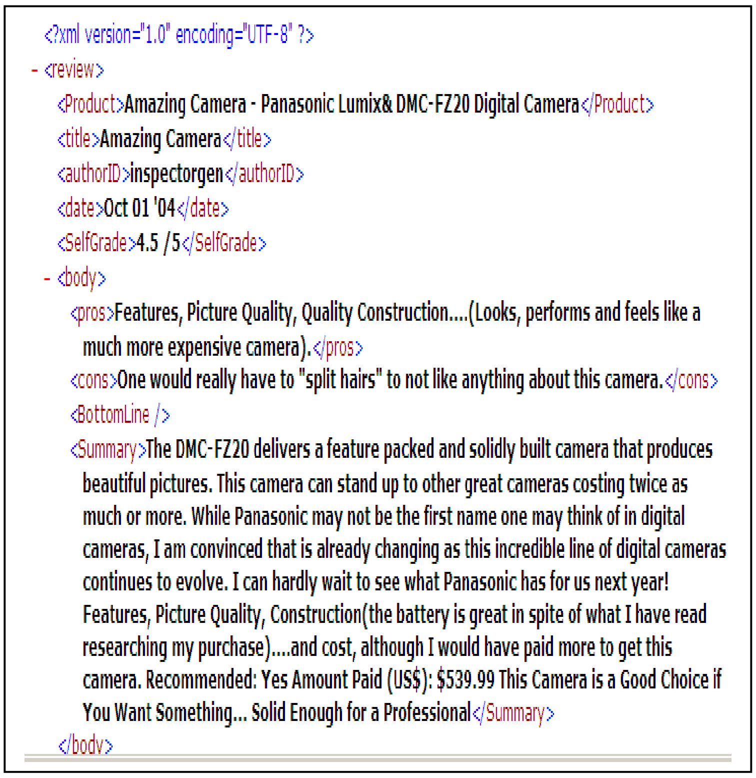
Amazing Camera - Panasonic Lumix& DMC-FZ20 Digital Camera

Title:*Amazing Camera*
Written by:inspectorgen
Dated: Oct 01 '04

Grading:4.5/5

Pros:Features, Picture Quality, Quality Construction....(Looks, performs and feels like a much more expensive camera).
Cons:One would really have to "split hairs" to not like anything about this camera.
Bottom line:
Summary:The DMC-FZ20 delivers a feature packed and solidly built camera that produces beautiful pictures. This camera can stand up to other great cameras costing twice as much or more. While Panasonic may not be the first name one may think of in digital cameras, I am convinced that is already changing as this incredible line of digital cameras continues to evolve. I can hardly wait to see what Panasonic has for us next year!Features, Picture Quality, Construction(the battery is great in spite of what I have read researching my purchase)....and cost, although I would have paid more to get this camera. Recommended:Yes Amount Paid (US$): $539.99 This Camera is a Good Choice if You Want Something... Solid Enough for a Professional

Figure 4. Output of AEAXTCCTFOM_HTML_Refinement function(Epinions.com)



```
  <?xml version="1.0" encoding="UTF-8" ?>
- <review>
    <Product>Amazing Camera - Panasonic Lumix& DMC-FZ20 Digital Camera</Product>
    <title>Amazing Camera</title>
    <authorID>inspectorgen</authorID>
    <date>Oct 01 '04</date>
    <SelfGrade>4.5 /5</SelfGrade>
  - <body>
      <pros>Features, Picture Quality, Quality Construction....(Looks, performs and feels like a
        much more expensive camera).</pros>
      <cons>One would really have to "split hairs" to not like anything about this camera.</cons>
      <BottomLine />
      <Summary>The DMC-FZ20 delivers a feature packed and solidly built camera that produces
        beautiful pictures. This camera can stand up to other great cameras costing twice as
        much or more. While Panasonic may not be the first name one may think of in digital
        cameras, I am convinced that is already changing as this incredible line of digital cameras
        continues to evolve. I can hardly wait to see what Panasonic has for us next year!
        Features, Picture Quality, Construction(the battery is great in spite of what I have read
        researching my purchase)....and cost, although I would have paid more to get this
        camera. Recommended: Yes Amount Paid (US$): $539.99 This Camera is a Good Choice if
        You Want Something... Solid Enough for a Professional</Summary>
    </body>
```

Figure 5. Output of AEAXTCCTFOM_XMLFilegenerator function.

The purpose of these algorithms is to achieve an xml file containing all the required part corresponding to an html file. Our aim is to use xml files as a corpus for the purpose of natural language processing so that the polarity and the sentiment of the reviews can be detected.

6. Conclusion

In this paper the major emphasis is given on the required portion of html file extraction which in terms will be converted into corresponding xml file and this is highly required in the field of machine learning mechanism in opinion mining especially for PoS tagging (Parts of Speech) and then sentiment analysis. Here, selection of each pattern for matching the regular expression is done in a very simple manner which reduces the code complexity.

The basic algorithm is for corpus refinement and the supported file formats are generally html, htm. And we have taken corpus of the volume of around 2000 files stored in tree like folder structure.

Through this algorithm, we have succeeded to generate corpus for the further work in the field of opinion mining especially for the technique of POS tagging. A huge Product review Database is used for testing. Only one such example is shown here. The regular expressions of those html files are used for text extraction from the input files. In this case regular expressions are the tentative tags supported by html where required review information is stored. Though the expressions are not same for all kind of review pages, it depends on the website i.e. the domain from where we are collection the data. So we have to study those expressions prior to using this algorithm on any html page containing review. To generalize our approach, the only thing which has to be modified in the algorithm is to store all possible patterns i.e., the regular expressions related to the user reviews into a separate file and while reading each line of an input file, we need to consult that file containing the patterns by reading each line of it. And thus it will work for all patterns irrespective of the domain from where those are collected.

Acknowledgement

This work was supported by the Security Engineering Research Center, granted by the Korea Ministry of Knowledge Economy.

References

[1] Bo Pang and Lillian Lee, “Opinion mining and sentiment analysis”, Foundations and Trends in Information Retrieval 2 (1-2), pp. 1-135, 2008.
http://www.cs.cornell.edu/home/llee/opinion-mining-sentiment-analysis-survey.html.

[2] A. Agarwal and P. Bhattacharyya, “Sentiment analysis: A new approach for effective use of linguistic knowledge and exploiting similarities in a set of documents to be classified,” in Proceedings of the International Conference on Natural Language Processing (ICON), 2005

[3] E. Breck, Y. Choi, and C. Cardie, “Identifying expressions of opinion in context”, IJCAI-07, pages 2683–2688. http://www.cs.cornell.edu/home/cardie/papers/ijcai-2007.pdf.

[4] Hatzivassiloglou and McKeown, 1997] Vasileios Hatzivassiloglou and Kathleen R. McKeown. Predicting the semantic orientation of adjectives. In ACL97, 1997.

[5] C. O. Alm, D. Roth, and R. Sproat, “Emotions from text: Machine learning for text-based emotion prediction,” in Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP), 2005.

[6] http://reviews.cnet.com/?tag=hdr%3bsnav as visited on 23/08/2009

[7] http://www.epinions.com/ as visited on 23/08/2009