ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 1, January 2014

Flat Back Index Generation for E-Books: A Tri-gram Approach

Saket Soni[1], Raj Kumar Singh[2], Sarang Pitale[3]

Assistant Professor, Dept. Of Computer Science & Engineering, Chhatrapati Shivaji Institute of Technology, Durg, India[1]

Assistant Professor, Dept. Of Information Technology, Bhilai Institute of Technology, Durg, India[2]

Assistant Professor, Dept. Of Information Technology, Bhilai Institute of Technology, Durg, India[3]

ABSTRACT: Comprehending a book is a frequent activity which each one of us does in our existence. A universal strategy to find a page for reading is to use front index and back index. A front index generally contains the sections and subsections matter with their corresponding page numbers. A back index contains various seed words of books with corresponding page numbers in the sorted alphabetical order. To spot a topic, the page numbers are identified using these indexes. The back index is of two types flat and hierarchical. The professional indexers find their job tedious when they make a back index for a book. These all jobs are manual and require background knowledge of subject. At present various automatic tools are available which generates the back-of-book indexes. Various top book authors does not offer back indexing only due to its complexity and labour-intensive manual modifications. The present paper demonstrates one such method which generates flat back-of-book index efficiently.

Keywords: Typed Dependency Parser, Noun Phrases, Noun Phrase Extraction, Bi-grams, Tri-grams, Flat Back Index

I. INTRODUCTION

The proposed approach will automate the generation of back-of-book indexes. This approach converts the portable Document Format files to text format. PDF [1] has various advantages over other formats. Security and object wrapping are some of such advantages. But these advantages are the road blockers for back index generation process. This advantage restricts the information extraction process. The current scheme uses iText [2] library for extracting information from PDF format e-books. It splits the PDF file and converts into text file format. The text files are then arranged according to the page numbers. In next step these text pages are passed through Stanford Parser [3]. The intermediate outputs are refined by using string matching techniques. At last the final output contains individual or combination of noun phrases can be called as nouns bi-gram and tri-gram that will be arranged in alphabetical order.

In any standard book the back index component includes, Noun Phrases with or without sub-headings, References (i.e. page numbers). The *Noun Phrase (NP)*, if a phrase as a noun [4] (or indefinite pronoun) as its head word, or which performs the same grammatical function as such a phrase then the phrase is called Noun Phrase. Noun phrases are very common cross-linguistically, and they may be the most frequently occurring phrase type. Noun phrases can be embedded inside each other; for instance, the noun phrase some of his constituents contains the shorter noun phrase his constituents. In some modern theories of grammar, noun phrases with determiners are analysed as having the determiner rather than the noun as their head; they are then referred to as determiner phrases.

II. RELATED WORK

Several works has been done in this field. In [5] authors extract front index using string matching technique. They have developed a Frame Work for e-books, known as MCFE which uses the front index extraction process and takes the extracted front index as Meta information. In [6] authors identify the page numbers to spot a topic using front index and back index. Authors have presented a Meta content framework to generate back indexes for e-books which uses part of speech tagging. In [7] authors generated a set of model answers from two or more books in HTML format, so that it can be easily rendered in web browser, using data mining approach.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 1, January 2014

III. METHODOLOGY

The overall process of back-of-book index generation can be represented by a block diagram as shown in fig. 1 below.

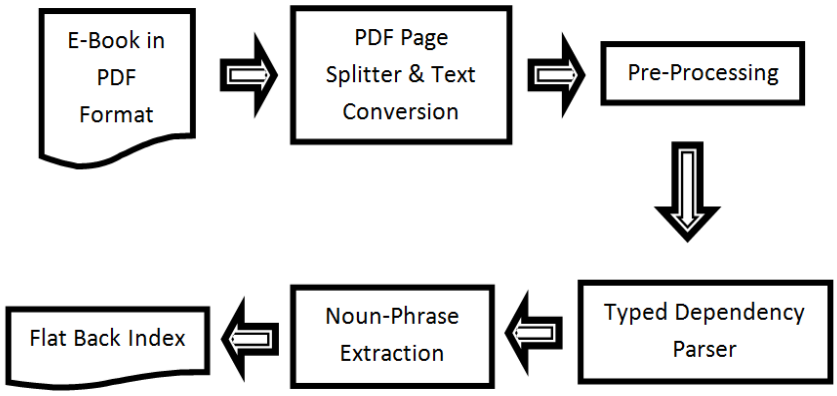

Figure 1: Process of flat back of the book index generation

A. PDF PAGE SPLITTING AND TEXT CONVERSION

This module splits the PDF e-books into same number of text pages the book contains. PDF to text conversion can be done by various tools [8]. A java [9] based library iText is used which splits the PDF file into number of text pages. The naming conversion or the pages is as "PageNumber.txt". The split process is important because it provides a unique identification of the page and its respective content.

B. PRE-PROCESSING OF E-BOOK

This step takes the PDF e-book as input. E-books are then pre-processed for the further operations. Pre-processing includes:

1) *Page number association:* After conversion of PDF to text each page is allotted with a page number. Page number allocation is important because the page numbers generated after splitting and conversion of pdf file and actual page number which start the chapter contents are different. For this purpose we have to identify actual page number from the chapter contents are starting and then put it manually in our program.

2) *Removal of special characters from text file:* This part of task is done only to make the text file unified. Presence of special characters including hyphens, question marks, and extra dots excluding full stop can affect the sentence identification process while post processing.

C. PARSING

In this step the system inputs each chapter contents to the Stanford Parser. Stanford parser then generates the type dependencies for the text. Parser from Stanford is a java program and is integrated with the system easily. This approach uses Stanford Typed Dependency Parser [10]. The Typed Dependency representation was designed to provide a simple description of the grammatical relationships in a sentence that can easily be understood and electively used by people without linguistic expertise who want to extract textual relations[11]. In particular, rather than the phrase structure representations that have long dominated in the computational linguistic community, it represents all sentence relationships uniformly as typed dependency relations.

Here is an example sentence:

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 1, January 2014

"Bell, based in Los Angeles, makes and distributes electronic, computer and building products."

For this sentence, the Stanford Dependencies (SD) representation is:

nsubj(makes-8, Bell-1)
nsubj(distributes-10, Bell-1)
partmod(Bell-1, based-3)
nn(Angeles-6, Los-5)
prep in(based-3, Angeles-6)
root(ROOT-0, makes-8)
conj and(makes-8, distributes-10)
amod(products-16, electronic-11)
conjand(electronic-1, computer-13)
amod(products-16, computer-13)
conjand(electronic-11, building-15)
amod(products-16, building-15)
dobj(makes-8, products-16)
dobj(distributes-10, products-16)

Each file is parsed individually and a typed dependency for each sentences are generated for all the pages. The process generates different file containing typed dependency with name "Page-Number.txt" at different folder.

D. NOUN PHRASE EXTRACTION

This process provides relationship between the words for each sentence of the generated files. The output of Typed Dependency Parser is used to extract the noun phrases that can give seed words for back index generation. The typed dependency form contains relationship between two words i.e. *nn(secondword-pos, firstword-pos)*. These relations can generate either bi-grams or tri-grams. Here *pos* specify the position of the word within the sentence.

1) *Seed Words:* Seed word is a first occurrence of a noun from which noun phrase chain can be constructed. For example in given relationship nn(Angles-6, Los-5), the word Angles can be treated as seed word. Seed word can play an important role in tri-gram and bi-gram generation.

2) *Tri-Grams Generation:* Tri-grams are those sentences which contains three seed words. This step uses noun phrases generated by typed dependency parser. If *secondword-pos* of first line and the *seconword-pos* of second line for two consecutive words are identical then we can declare it as tri-gram and rearrange them. Here rearranging process involves taking all the four terms of two consecutive nn's delete the repeating term and finally placing them according to *pos* values. After rearranging process this approach deletes the entry from dependency and removes all *pos* values. This step also deletes the both the lines from dependency relation.

3) *Bi*-Grams Generation*:* Bi-grams are those sentences which contains three seed words. The remaining content inside bracket with nn's after above step can give bi-grams. Here obviously *secondword-pos* of first line and the *seconword-pos* of second line for two consecutive words will not be identical then we can declare it as bi-gram and rearrange them. Here rearranging process involves taking the two terms of current line and putting them according to *pos* values. After rearranging process this approach deletes the entry from dependency and removes all *pos* values. This process will run while the file is not empty.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 1, January 2014

E. ALPHABETICAL ORDER SORTING

This last step uses simple sorting technique for arranging all bi-grams and tri-grams in alphabetical order. In this process all the bi-gram and tri-grams are merged at one location, rearranged and storing them into a different location. The all arranged terms are generated in the form of back indexed with page numbers separated by comma.

F. ALGORITHM

FBI_Algo

Input: E-Book in PDF
Output: Flat Back index

Step 1: For each pdf page
Step 2: convert pdf into text files
Step 3: End loop
Step 4: For each text files
Step 5: convert typed dependency of each sentences
Step 6: store all pages with corresponding page number as file name
Step 7: End loop
Step 8: For each pages of typed dependency
Step 9: extract only nn(...)
Step 10: if two consecutive first word inside two consecutive nn(...) are same
Step 11: Then
Step 12: select both words inside nn's and generate a tri-gram
Step 13: attach a page number separated by comma
Step 14: else
Step 15: select nn and generate a bi-gram.
Step 16: attach a page number separated by comma
Step 17: End if
Step 18: Rearrange in order
Step 19: End

IV. OBSERVATIONS AND RESULTS

A number of experiments were run to process e-books and it is observed that the time needed to process an input file depends upon the number of pages present in the book. The experimental setup was implemented upon text corpora from varied course ware (e-books in the present setup) and also with the different page counts and content sizes. All the observations clearly indicate that the predominantly prevailing factor that contributes to excessive execution time is the varying document page size.

After successful execution of the code it will generate a editable flat back index with tri-grams and bi-grams. It has been made editable because indexer can get flexibility towards modification of generated index if needed. The sample output is shown below in fig. 2 that is generated from only three pages of e-book.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 1, January 2014

Figure 2: Generated Flat Back Index with bi-grams and tri-grams

V. CONCLUSION

Machine-generated Back index can act as implicitly available back-ground knowledge (ontology). Such an implicit ontology plays an important role in:

- NLP Machine translation tasks
- Automatic topic spotting
- Content Summarization
- Topic relevancy computation and ranking,

Without the use of explicit ontology, the publish-cum-printing time is drastically reduced, index appearance: totally precise and complete. Expense-overhead is reduced against equivalent manual effort required for task completion and more comprehensive than flat back indexes having only unigrams.

VI. ACKNOWLEDGMENT

The author sincerely thanks Prof. Arpana Rawal, Prof. Ani Thomas for their timely, invaluable and indispensable guidance and consequently encouragement shown towards the work group.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 1, January 2014

REFERENCES

1. Adobe Systems Incorporated, http://www.adobe.com/pdf/
2. iText - Free / Open Source PDF Library for Java and C# , http://www.itextpdf.com/
3. Extraction of Grammatical relations between text fragments: Typed dependency parser (courtesy: The Stanford Natural Language Processing Group). http://nlp.stanford. edu:8080/parser/index.jsp
4. Loos, Eugene E., et al. Glossary of linguistic terms: What is a noun? 2003.
5. Tripti Sharma, Sarang Pitale, 'Front Index extraction from research documents using Meta-Content Framework', Indian Journal of Education and Information Management (IJE&IM), Vol. 1 No.7, pp.301-305, 2012.
6. Tripti Sharma, Sarang Pitale , 'Meta-Content framework for back index generation', International Journal on Computer Science and Engineering (IJCSE), Vol. 4 No. 04, pp. 627-633, 2012.
7. Tripti Sharma, Sarang Pitale, 'Chapter extraction from research documents using Meta-Content Framework', International Journal of Engineering and Advanced Technology (IJEAT), Vol. 1, Issue 5, pp. 337-339, 2012.
8. Sarang Pitale and Tripti Sharma, 'Information Extraction Tools for Portable Document Format', International Journal of Computer Technology And Applications (IJCTA), Vol. 2 (6), pp. 2047-2051, 2011.
9. Oracle Corporation, http://www.java.com
10. Marie-Catherine de Marneffe and Christopher D. Manning, 'Stanford Typed Dependencies Manual', The Stanford Natural Language Processing Group, 2008.
11. Prof. Ani Thomas, Dr. M.K. Kowar, Dr. Sanjay Sharma (2011), 'Extracting Noun Phrases in Subject and Object Roles for Exploring Text Semantics', International Journal on Computer Science and Engineering (IJCSE), Vol. 3 No. 1, pp. 1-7, 2011.

BIOGRAPHY

Saket Soni is Assistant Professor in Department of Computer Science and Engineering in Chhatrapati Shivaji Institute of Technology, Durg, India. He pursued his Bachelor of Engineering in Computer Science and Engineering from Guru Ghasidas University, Bilaspur, Chhattisgarh, India and Master of Technology in Computer Science and Engineering from Indian Statistical Institute, Kolkata, West Bengal. His primary research interests focus on network security and natural language processing.

Raj Kumar Singh is Assistant Professor in the Department of Information Technology in Bhilai Institute of Technology, Durg (C.G.), India. He pursued his Bachelor of Engineering in Information Technology from Chhattisgarh Swami Vivekanand Technical University, Bhilai, Chhattisgarh. Currently pursuing Master of Technology in Computer Science & Engineering from Chhattisgarh Swami Vivekanand Technical University, Bhilai, Chhattsigarh. His primary research interests focus on natural language processing and data mining.

Sarang Pitale is Assistant Professor in the Department of Information Technology in Bhilai Institute of Technology, Durg (C.G.), India. He pursued his Master of Technology in Computer Science and Engineering form Chhattisgarh Swami Vivekanand Technical Unversity, Bhilai and Bachelor of Engineering from Pt. Ravishankar Shukla University Raipur, Chhattisgarh. Sarang Pitale's primary research interests focus on Data Mining techniques. He is a member of Computer Society of India, International Association of Computer Science & Information Technology, International Association of Engineers.