Content Summarization of a Book using its Index

Pankaj Bhambhani
College of Information and Computer Sciences
University of Massachusetts Amherst
pbhambhani@cs.umass.edu

Abstract

Generating coherent summaries of long documents such as books is a relatively untouched field, with most of the research work being focused on short documents. In this Project we attempt to look at summarization of books. In particular we look at books which have a back-index and see if it can provide some useful information in generating good summaries. We provide a new dataset for the evaluation of such summarization systems. We propose our method and evaluate our generated summaries using two different metrics. We then attempt to analyze our results and suggest future improvements.

1 Introduction

Humanity has created a vast ocean of knowledge in the form of books. Books represent one of the oldest forms of written communication and have been used since thousands of years ago as a means to store and transmit information. These standards though, are changing with the evolution of technology. With the globalization of internet in todays world, we have an explosion of digital information all over the globe. A lot more people are now interested and curious in getting knowledge and information about various fields than previously, thanks to openly available content (for e.g. via Massive Open Online Courses - MOOCs as they call them). Though books are have always been the key source of crucial information (and continue to be), these new generation of explorers prefer newer methods of learning, such as online blog posts and YouTube video tutorials and they find books pretty long to read (no wonder they've invented a new term for long explanation texts - TL;DR Too Long; Didnt Read). Also, in general the increasing social media trend is reducing our concentration time period, so a lot of people end up starting books enthusiastically but never finishing them.

In this Project, we look at content summarization of books as an attempt to bridge this gap between knowledge and enthusiasm. We view text summarization of books as a field that has much scope - in the recent years, there have been large number of books published in electronic format, on portals such as Gutenberg (http://www.gutenberg.org/), DailyLit (https://dailylit.com/), WikiBooks (https://en.wikibooks.org/wiki/Main_Page), Google Book Search (https://books.google.com), etc. Equivalently, more and more producers of print books are making a digital version available for purchase through libraries. There has been previous work on *text summarization*, but most of it has been concerned with the summarization of *short* documents, like news stories, articles, blog comments, etc. There has been limited work done for summarization of long documents equivalent to the size of the book (more on this in Section 2), despite there being much scope for it.

We attempt to use a novel technique to summarize a book - using its Bibliographical Index, generally found at the back of a book. We understand that not all books have this index as its tedious to generate and can sometimes be not worth the effort. As such, our summarization technique is limited to books which have this index. The emphasis of our technique, as has been the case for typical long-document summarization methods, is more on covering the breadth of the document (book) with a limited set of sentences than on the sentences themselves being heavily correlated to each other. We introduce a new dataset for this purpose, one has the content of a book along with its index, and can be used for evaluation of systems generating book summaries using similar techniques. We also analyze our result using standard evaluation metrics, and see how index-based summarization can be a better choice than normal book summarization. We also discuss some limitations of our methods and suggest improvements and future work that may help generate better summaries for books that have a bibliographic index.

2 Related Work

Automatic text summarization is one of the actively-researched fields in natural language processing. The work in summarization can be dated back to early approaches to automatic abstraction (Luhn 1958; Edmundson 1969). The literature typically devises two types of text summarization (Carenini and Cheung 2008) - *extractive summarization*, which involves extracting important information from the text, as in identifying the sentences most relevant to the topic; and *abstractive summarization* which involves an additional step of generating new sentences to add fluency to the extracted text. Most of the advances in text summarization have been in the domain of extractive summarization, and even though abstractive summarization is generating interest (Rush et al. 2015) on shorter summaries (such as news headlines), we believe it still has a long way to go to match extractive summarization on longer documents. Hence, in this project, we focus solely on extractive summarization.

Even though there has been a lot of successful work on summarization of short documents with lots of evaluation datasets available as well (typically based on news articles), there has been very limited work concentrated on content summarization of books. In particular, we could only find one previous work (Ceylan and Mihalcea 2007), which best matched our goals for this project. This work attempts to explore different summarization techniques for books and equivalent-size documents. Among other things, this work shows that the existing systems developed for the summarization of short documents do not fare well when applied to very long documents such as books. It also presents a dataset for the evaluation of book summarization systems. However, our approach of using Index of a book while generating summaries is different than the one used in this paper, and requires a dataset of books that have this index, and none of the books in the above dataset matched our requirement. We therefore created our own dataset for the purpose of this work.

The literature also identifies two main types of methods for generation of text summaries - *supervised* methods, which involve machine-learning algorithms trained using existing, typically human-written summaries as a reference; and unsupervised techniques, which operate on features derived from the input document.

Among methods involving supervised learning, there has been a number of successful attempts at generating text summaries for short documents. Given a set of input documents and their corresponding reference summaries, these methods identify different properties of the reference summary, such as the presence of named entities, location of key phrases and the positional scores. Unsupervised learning methods also have had success. Typically these methods take into account the weights of words in sentences, as well as the position of sentences in the document. These techniques have been successfully implemented in the centroid approach (Erkan and Radev 2004a), which extends the idea of tf.idf weighting (Salton and Buckley 1988) by introducing word centroids, as well as integrating other features such as position, first-sentence overlap and sentence length. More recently, graph-based methods that rely on sentence connectivity have also been found successful, using algorithms such as node-degree based centrality (Salton et al. 1997) and eigenvector centrality (LexRank - Erkan and Radev 2004b). A review of some of these summarization techniques can be found in (Nenkova and McKeown 2012) . For our project, due to limited time and limited number of books in our dataset,we decide to use LexRank in our experiment. So to summarize, this project focuses on *extractive* summarization using eigenvector centrality based unsupervised learning algorithm *LexRank* and using books with a back-of-the-book *index*.

3 Model - LexRank (Erkan and Radev 2004b)

Here we give a brief overview of the LexRank algorithm used for generating summaries in this project. LexRank is from a general family of stochastic graph-based algorithms used for calculating the relative importance of textual units (sentences in this case). Some of the work prior to this was based on *tf-idf* weighing sentences, where *tf* is the term frequency of a word in a document and *idf* is the inverse document frequency, i.e. inverse of the frequency of a word in all documents (the whole corpus). This is done by identifying a set of words as being *central* to the document who have a tf-idf score above a certain threshold - the set being called the *centroid* of the document. A sentence that contains more words from the centroid is considered more *central* to the document topic. The summarizer then picks the *most central* sentences to generate a summary.

The newer methods instead work on the idea of *sentence based centrality*, where a document is treated as a network of sentences that are related to each other. A sentence that is related (similar) to many other sentences in the network is considered more *central* to the topic of the document. Using a bag-of-words model and a word-vector representation, each sentence can represented as a vector of size N, where N is the size of the vocabulary, with the value at the vector position for each word being equal to the *tf*, the number of occurrences of the word in a sentence times the *idf* of the word in the document. The similarity of two sentences is then just the cosine similarity between the two corresponding vectors (called the *idf-modified* cosine similarity).

$$idf - modified - cosine(x, y) = \frac{\sum_{w \in x,y} tf_{w,x} \quad tf_{w,y} \quad (idf_w)^2}{\sqrt{\sum_{x_i \in x}(tf_{x_i,x} idf_{x_i})^2} \quad \sqrt{\sum_{x_i \in x}(tf_{x_i,x} idf_{x_i})^2}} \tag{1}$$

where $tf_{w,s}$ is the term-frequency of the word w in the sentences s. A new graph similar to the centroid-based method can then be constructed from the above metric, with the nodes being sentences and two nodes being connected if the similarity is above a certain threshold. A common way to then define the centrality of a sentence is by the number of other sentences connected to it in the graph, i.e. the degree of the sentence; hence, this kind of centrality is called *degree-based centrality* (Salton et al. 1997). However, a problem with this approach is that centrality here depends solely on the degree, in other words if each edge is considered as a *vote* for that node, than the centrality depends on how many other nodes *vote* for that node. There could be some set of sentences which are off-topic but have high correlation among each other and hence can get a higher *vote*.

To solve this issue, we can look at a similar concept of *prestige* in social networks (Erkan and Radev 2004a). In such networks, the prestige of a person depends not only the number of relations/friends he has, but also on *who* his friends are. Using this, we can define a centrality $p(u)$ of a node u as a combination of the centrality of all its adjacent nodes, normalized by the degree of that node.

$$p(u) = \sum_{v \in adj[u]} \frac{p(v)}{deg(v)} \tag{2}$$

Representing a matrix similar to adjacency matrix but with values calculated from the above equation, we can then do a random walk over the matrix to get centrality values that converge for each node. However, if the graph has disconnected components, then the random walk will not reach all the nodes in the graph, so we must allow the random walker to ”escape” this component by jumping randomly to any of the N nodes in the graph with a probability d. The modified centrality looks like:

$$p(u) = \frac{d}{N} + (1 - d) \sum_{v \in adj[u]} \frac{p(v)}{deg(v)} \tag{3}$$

where the probability d is called the *damping factor*. This algorithm now is essentially a variant of PageRank (Page et al. 1999) and is called *LexRank*. A more powerful variant of LexRank doesn't consider similarities based on thresholds, instead it defines weights of edges based on the *tf-idf modified* cosine similarity, and uses that to calculate node centralities.

$$p(u) = \frac{d}{N} + (1-d) \sum_{v \in adj[u]} \frac{idf - modified - cosine(u, v)}{\sum_{z \in adj[v]} idf - modified - cosine(z, v)} p(v) \tag{4}$$

4 Dataset

One of the first hurdles that we encountered while working on this project was the lack of a proper dataset to work on. As we mentioned in Section 2, the only real related work we found was Ceylan and Mihalcea 2007, but their dataset didn't track index of the books. We tried to search places like Kaggle (https://www.kaggle.com/) and also looked at some of the datasets released as part of the annual Document Understanding Conferences (DUC). Even though we did find a few datasets related to text summarization, they were all short documents and we couldn't find a publicly available distribuion of the specific type of dataset that we were looking for. The lack of a general dataset of book summarization is understandable as metnioned in the paper since it's not easy even for humans to summarize long books and it takes quite a bit of time and effort to write down coherent summaries. To add to that, most of the books having an index are coursework related books requiring deep knowledge and comprehension to generate good summaries. Some of the books like Gray's Anatomy (Gray 2009) for example , would be very difficult for even experts to summarize, despite the fact that it has been read by many medical students all over the world.

We decided to create a dataset of books that were small to intermediate in length and whose content as well as summaries were digitally available on the internet. Taking into consideration the time and resources available for this project, we decided to limit our dataset to 10 books. It took some time for us to find books that are available (or have old editions available) in a digital format, that have a book index and also have concisely written summaries publicly available. Our source of books was Project Gutenberg and the ebooks available in UMass Amherst Libraries. The reference summaries were taken from WikiSummaries (http://www.wikisummaries.org).

For e.g., following is a sample extract of a summary of *Getting Things Done* by David Allen taken from WikiSummaries.

```
".... Allen states that a person is the
    most productive when the mind is
    clear, free of what he calls "open
    loops" -- the things people commit
    to do which remain undone and become
     a drag on the unconscious mind. He
    uses the analogy of RAM on a
    personal computer, with the idea
    that too much "stuff" stored in a
    person's short-term memory can blow
    a fuse. His idea is that the
    conscious mind is a focusing tool,
    not a storage place....."
```

4.1 Pre-processing

We did some pre-processing for the book content as well as the index. Some of that was necessary and some was done to speed-up the experiment. For the index, the pre-processing depended on how the author specified the index. For example, names of persons are usually specified differently in the index than the way they may have been used in the book (for instance the name *Brendan O'Connor* would be written in index as *O'Connor, Brendan*, i.e last name followed by first name). A single word could have multiple mentions in different contexts (for instance, in a book about the history of a country, the word *economy* might have different usages during different periods of time (ancient, medieval, modern) and each of these usages might have separate mentions in the index) in which case we just treated them as different words, or one word could be referenced in the index of other words (for instance, an index mention of *painting*, could also have a reference to the index mentions of the word *art*) which we treated as one mention (removed references). This pre-processing required some effort, and required combination of scripts and manual filtering. Also, for this project, we only the consider the words in the index, and not the page-numbers associated with this. This could be vital information for future improvements in the process and is discussed in Section 8. For convenience, we sort this list of index words alphabetically and remove duplicate mentions of words.

For the book content, since we didn't consider the page numbers associated with index words, we decided to treat the entire book as one single text instead of diving it in pages. We tokenized the text into sentences, using the NLTK tokenizer for this, with the standard Punkt English tokenizer vocab. We also removed non-ascii characters for ease of operation as those would generally not be present in the index words. We report the following statistic about the books and the summaries - the number of words in the book content range 20,000 to 2,50,000 with the average number of words being 88,750. Similarly the number of words in the summaries range from 300 to 4,000 with the average number of words being 2450. Figure 1 shows a plot of book lengths vs summary lengths.

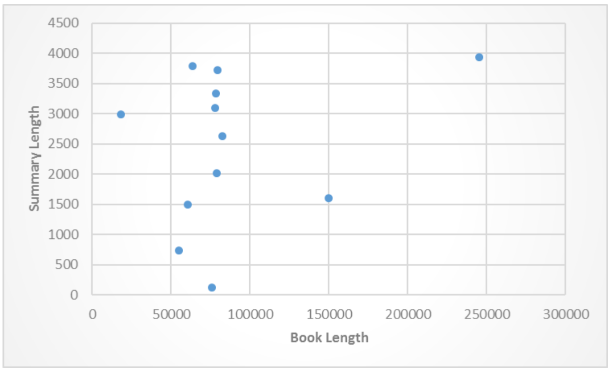

Figure 1: Plot of Book length vs Summary length

5 Evaluation Metrics

For the evaluation of our text summaries, we used the ROUGE and BLEU metrics. ROUGE is a method which takes into account Ngram statistics and is found to be highly correlated with human evaluations (Lin and Hovy, 2003). The version of ROUGE we used was ROUGE 2.0 which can be found online at http://www.rxnlp.com/rouge-2-0/. The evaluations in Section 7 are reported using the ROUGE-1 setting, which seeks unigram matches between the generated and the reference summaries. BLEU is another metric which is used for machine translation tasks. It uses modified n-gram precision measure to rank sentences by not taking into account an n-gram once it is used/exhausted. It also penalizes sentences for brevity as compared to reference sentence lengths.

As in the case of Ceylan and Mihalcea 2007, we have to deal with pre-existing summaries, with large summary-length variations across the 10 books and across the reference summaries, which is unlike many of the previous evaluations, where the length of the reference and generated summaries was pre-defined and fixed. To address this, we decided to compare the summaries generated by using the book index with two other types of summaries - one in which the entire content of the book was passed to the summarizer (without doing index-based extraction as explained in Section 6) to generate a summary of equivalent length; and a baseline summary created by taking the first n sentences of the book, where n is the length of the corresponding reference summary. The evaluations for these summaries are also presented in Section 7.

6 Experiment/Algorithm

As mentioned in Section 4.1, we first pre-process the book by treating the entire content of the book as a single text instead of dividing it into pages, and then tokenize it into sentences. We also pre-process the index and extract the words, sorting alphabetically and removing duplicates. Once the pre-processing is complete the experiment itself is quite simple. For the simple case (used in this project), every sentence in the text is assigned a binary score 1 or 0, based on the presence or absence of index words in the sentence respectively. In a general case, each index word could be assigned a weight w that takes into account its tf-idf frequency (which can be derived by searching over the page-numbers associated with the index) and a sentence could be assigned different weights based on how many ”heavy-weight” index words are present in the sentence. The matching sentences are then added to a resulting list of sentences (if not already present). This list contains sentences that are in some sense *more relevant* than the rest, and is then passed to a LexRank based content summarizer to produce a final summary. The length of the summary is equivalent to the length of the reference summary. The working of LexRank is explained in Section 3. The version of LexRank summarizer used for this project is taken from an open-source implementation of the same for Python - *Sumy* and is available at https://github.com/miso-belica/sumy/blob/dev/sumy/summarizers/lex_rank.py. We also tried a slight variant of LexRank called *TextRank*, which takes weights of edges into account while calculating sentence centrality (as explained in that section). The codebase and dataset for our project is available at GitHub - https://github.com/pankajb64/bookesis

7 Results

As mentioned in Section 5, we used two different metrics to evaluate our summaries generated using the LexRank and the TextRank methods with the reference summaries, namely ROUGE-1 and BLUE. Unlike most of the previous summarization systems which were designed for short documents and were supposed to produce summaries of predefined lengths, we had to consider the length of the reference summary into account and generate a summary of equivalent length. So really we can only compare individual generated sum-

	ROUGE - 1			**BLEU**
Summary Type	**Avg. R**	**Avg. P**	**Avg. F**	**Avg. BLEU Score**
LexRank(Full Text)	**69**	27	38	10.2
LexRank(Index-Extracted Text)	59	**34**	**41**	**17.4**
TextRank(Index-Extracted Text)	**67**	28	39	11.5
BaseLine	53	**41**	**44**	**12.1**

Table 1: Evaluation Results for LexRank and TextRank with and without index-based extraction

maries against their corresponding reference summaries and cannot compare summaries of two different books against each other. To counter this, we designed to create two more summaries in addition to the two generated summaries. One summary was created by passing the full content text *as is* to the LexRank summarizer (instead of passing it through our index-based extraction method). The other summary is a baseline summary generated by taking the first n lines of the text, where n is the length of the corresponding reference summary. All the 4 summaries thus have length equal to the reference summary.

Table 1 shows the results of evaluations for the four different types of summaries using the two metrics. These results were calculated for the 10 books and then averaged. The numbers highlighted in bold are the first and second best options for each case. As is evident from the table, the performance of TextRank is equivalent to the performance of the LexRank with no index-based extraction, with high average values of Recall and low average values of Precision and F-Measure. A high value of Recall would mean the summary had a large fraction of n-gram overlaps with the reference summary, while a low Precision would be it also had a much larger number of n-grams that weren't in the reference summary. For generating concise summary equivalent to the reference summaries, a high value of F-Measure is desirable. The LexRank algorithm applied on index-based extracted text from the book is able to get a high value of F-Measure than the LexRank with full text as input or the TextRank algorithm. However, it does not do quite as well as the baseline summary in the ROUGE-1 setting. This is because as seen in a few DUC conferences on text summarization (wherein the algorithms generating the baseline summaries were winners), it is tough for summarization algorithms to beat the baseline summaries, since it is in those initial lines that the author generally tends to introduce the text and layout a plan for the remainder of the text. In those initial lines, there generally is a much higher mention of the significant words that would eventually appear in the index, only the sentence structure itself could be different. To counter this, we resorted to the BLEU metric which as mentioned in Section 5 uses modified n-gram precision measure to ignore repeating n-grams and punishes very short sentences. The results shown in the Table show that our LexRank method has a significant number of different relevant n-grams than the remaining three-summaries which led to it getting a higher score. The baseline summary still proved to be a good measure as it did better than the other two summaries on the BLEU score as well.

It is worth noting that in comparision to the full text content, the index-based extracted text was significantly smaller in size, with an average reduction in the number of sentences close to 47%. As a result the LexRank summarization was also efficient - on an average the summarization with the full text as an input took ~ 600 seconds to generate a summary, while the summarization on the index-based extracted text took only ~ 180 seconds. The time complexity of the extraction algorithm is $\mathcal{O}(n * m)$, where n is the number of sentences in the text and m is the number of unique index words. Generally m is only about ~ 200 words, so this algorithm is pretty fast. The overall process thus is also pretty efficient than the raw case of passing the full text as input.

8 Future Scope

There are certainly many things we couldn't do due to time limitations in this project, and they can be looked at points of improvement in future work in this area. The most significant of them is the use of page-numbers associated with index words. The page numbers can serve as weights or ranks for the index words and help distinguish more important words from less important, resulting in generation of better summaries. They can also help differentiate between the importance of different mentions of the same index word. For e.g. if the author mentions the word *economy* as index word with corresponding page numbers being 30-40, then any mentions of economy outside this range of pages should be given lesser importance (ideally ignored). To achieve this, the text itself would have to be segregated into pages, which would require better mining tools, since the books are generally available only in *PDF*, *DVI* or *PUB* format.

It would also be useful to preserve some context in which a sentence was extracted based on the index word. This could be achieved by also extracting the sentences that immediately precede or succeed the chosen sentence. Such a context would probably produce more coherent summaries, thereby increasing Precision. This sentence based extraction could also be extended to Paragraph based extraction, with the paragraph itself being extracted to keep the context. This could also be useful for generating individual summaries pertaining to a specific index word, very similar to query-based summarization. For e.g. an answer to the question *What are the author's views on economy ?*, could be answered by extracting the content from the pages where *economy* appears (obtained from the index) and generating a summary (extractive or abstractive) based on the extracted content.

Finally, better pre-processing can also lead to generating better summaries with specific type of words being handled, for e.g. a word appearing in different phrases may be treated as same or different depending on the context and also depending on its use as a noun or a verb (POS tags might be useful here). Similarly references to a word could be treated as additional mentions of the word, thereby increasing its importance.

9 Conclusion

In this Project, we looked at the task of summarizing the content of a book based on its back index. Although there has been much work done in the domain of text summarization, much of the research is related to summarization of short documents. There is limited work when it comes to summarizing long documents such as books and there is none which tries to exploit the back index available on certain books to generate good summaries; in this project we have tried to do just that. We also introduced a new dataset which can be used to evaluate summarization systems for books with such an index. While analyzing the evaluations, we also saw how it is tough in general for summarization algorithms to beat baseline summaries. The task of text summarization is as much an art as it is a science. There are many subjective opinions, with different authors choosing different parts of the text as relevant to the summary. The evaluation itself is still in very early stage. However, with the ever-increasing number of digital e-books and with a general increasing interest in reading descriptive summaries instead of the books themselves (as mentioned in the Introduction), there is expected to be more research in this field in the near future. We hope that some of this research will look at book indexes as a viable option for generating coherent and descriptive summaries.

References

Giuseppe Carenini and Jackie Chi Kit Cheung. Extractive vs. nlg-based abstractive summarization of evaluative text: The effect of corpus controversiality. In *Proceedings of the Fifth International Natural Language Generation Conference*, pages 33–41. Association for Computational Linguistics, 2008.

Hakan Ceylan and Rada Mihalcea. Explorations in automatic book summarization. Association for Computational Linguistics, 2007.

Harold P Edmundson. New methods in automatic extracting. *Journal of the ACM (JACM)*, 16(2): 264–285, 1969.

G nes Erkan and Dragomir R Radev. Lexpagerank: Prestige in multi-document text summarization. In *EMNLP*, volume 4, pages 365–371, 2004a.

G nes Erkan and Dragomir R Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479, 2004b.

Henry Gray. *Gray's Anatomy: With original illustrations by Henry Carter*. Arcturus Publishing, 2009.

Hans Peter Luhn. The automatic creation of literature abstracts. *IBM Journal of research and development*, 2(2):159–165, 1958.

Ani Nenkova and Kathleen McKeown. A survey of text summarization techniques. In *Mining text data*, pages 43–76. Springer, 2012.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: bringing order to the web. 1999.

Alexander M Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*, 2015.

Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

Gerard Salton, Amit Singhal, Mandar Mitra, and Chris Buckley. Automatic text structuring and summarization. *Information Processing & Management*, 33(2):193–207, 1997.