**The University of Texas at Dallas**

**CS 6322**

**Information Retrieval**

**Spring 2016**

**Section : 001**

**Class Project Report**

**Project TITLE: Search engine for Book**

TEAM: 02

Students:

**Swastik Sabat**

**Kunal krishna**

**Sarvotam Pal Singh**

**Janmejaya Sahoo**

**Saquib Khan**

1. The Problem: Generate a search engine for Books.

Our primary focus was to create a book related search engine. System workflow is as follows:

****

Major deliverables by each team member are as mentioned below :

* Crawling: Swastik Shekhar Sabat
* Indexing and relevance: Kunal Krishna
* User interface and comparisons with Google and Bing: Janmejaya Sahoo
* Clustering: Sarvotam Pal Singh
* Query expansion and relevance feedback: **Saquib Khan**

Learning –

* Crawling – We gained knowledge of how crawlers work. We also learned the functioning of Carwler4J library.
* Indexing and Relevance – We learned the practical aspects of building indexes and relevance models.
* User Interface – We learned the use of Html,CSS,JQuery,Java Script and Bootstrap. It was a good experience of creating modular and flexible UI that can collaborate with many different modules.
* Clustering- We learned how clustering is effective in clubbing similar documents together.

•Query Expansion- Learned the pros and cons of relevance pseudo feedback and how metric clustering works in the real world search engine.

Difficulties-

• Variety and vastness of data – Since Book websites host a wide variety of data, each site in it’s own format, we had difficulties crawling and extracting meaningful data.

• Incomplete Data – Not all websites contain all our required data. Some sites don’t have album information, some don’t have genre and so on. We had problems with the quality of our data due to this.

* Permissions – Not all sites permitted us to crawl them.
* Politeness – As we had to limit our maximum crawling speeding, it took a long time for us to crawl.

**Crawling:**

**Swastik Shekhar Sabat(sss150430)**

Design –

A Web crawler basically collects all the data that is required for a search engine. It is quite simple at its core:

1. Select a seed URL to crawl

2. Fetch and parse page

3. Save the important content

4. Extract URLs from page

5. Add URLs to queue

6. Repeat

The crawler starts with a list of URLs to visit, called the seed URLs. As the crawler visits these URLs, it identifies all the hyperlinks in the page and adds them to the list of URLs to visit, called the crawl frontier.

URLs from the frontier are recursively visited according to a set of policies. If the crawler is performing archiving of websites it copies and saves the information as it goes. The archives are stored in a way that they can be viewed, read and navigated in the form of html files.

A URL frontier is the collection of URLs that the crawler intends to fetch and process in the future.URL frontiers generally work in one of two ways - a batch crawl or a continuous crawl.

A batch crawl’s frontier contains only new URLs, whereas a continuous crawl’s can contain a seed set of URLs but new ones may be added (or existing ones removed) during the crawl.

Regardless of the type of crawl, the data store used to manage the frontier needs to be fast because it will slow down crawling otherwise.

**Swastik Shekhar Sabat(sss150430)**

I have used open source java library crawler 4j to crawl important websites related to Book.

In order to collect book data, I crawled up to 100 websiteseed URLS which resulted into 600000 web pages.

In order to avoid duplicates in the crawl, the crawler normalizes new URLs to remove duplicates.

For example, many pages are expressed relative to the main domain. Such that a link might be /about.html, but would need to be transformed into: http://www.goodreads.com/about.html. And since many URLs in the wild have query parameters (such as those used for analytics or session information), it is easy to come across many different versions of a URL that reference the same web page. These extraneous parameters typically have to be removed or normalized so that the only URL that is crawled is the canonical version of the page.

Once the crawler is configured the last piece of the puzzle was extracting and structuring the data from different web pages.

The most important part of parsing web pages is that the parser must be able to deal with messy markup and will be resilient to errors. I have written different parsers for each websites that removes the page chrome (like navigation, headers/footers, search boxes, advertisements, etc.) so that only the relevant data is returned.

This data is then stored in the form of text files as a book object which mainly contains Book Name, Book Description, Author Name, ISBN Number,Book URL. This way I provided hyperlink information and other relevant information to my teammate Kunal who was responsible for indexing

**Name:Kunal Krishna**

**Net Id: kxk155230**

**Indexing and Relevance:**

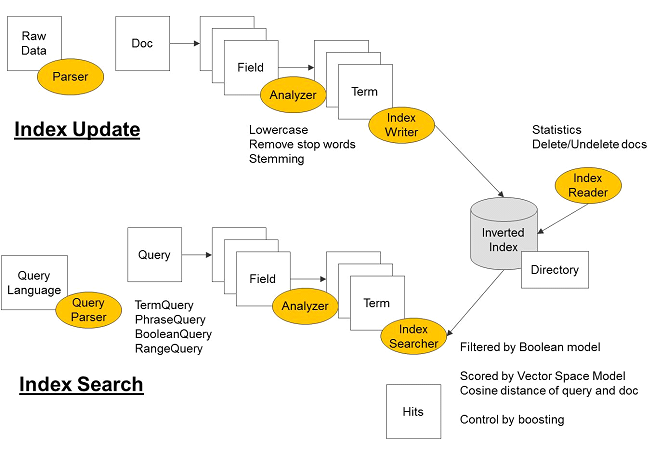
1. Describe how you assembled the index – make a picture and included it in your description. Also elaborate on how your web graph was created –

**Design -** The index was created from the crawled data of about 100000 documents. My teammate swastik provided me with the parsed data in text format. I was given 100000 text files, each containing varying amounts of data, and my task was to build an index by extracting relevant terms from the text file. In order to assemble the entire index, I followed the below steps:

* Parse the raw data
* Analyze the data(Lowercase removal, stop words removal, stemming)
* Used Standard NLP Lemmatizer and porter Stemmer java files for generating lemma ,stemming and finally used w1 method to build an indexs.
* Used Tree Map, Hash Map, and Hash Set data structure to build the inverted index.
* Used Gamma, Delta encoding scheme to compress the index.
* Incremental indexing as fast as batch indexing
* Ranked searching, and fast retrieval of results under seconds used Web Graph to get the page rank and page hits.
* Handled query types-phrase, wildcard, proximity, range
* Sorting on any of the fields stored in the index
* Simultaneous update and searching

**Name:Kunal Krishna**

**Net Id: kxk155230**



**Web Graph –**

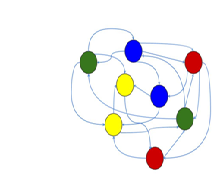
We can view the web consisting of static HTML pages together with the hyperlinks between them as a directed graph in which each web page is a node and each hyperlink a directed edge. That is in a web graph each vertex is a URL and the outgoing edges of a vertex are the hypertext links contained in the corresponding page. For creating the web graph for our music search engine, I have used the JUNG (Java Universal Network/Graph) Framework. JUNG is a software library that provides a common and extendible language for the modeling, analysis, and visualization of data that can be represented as a graph or network. The JUNG architecture is designed to support a variety of representations of entities and their relations, such as directed and undirected graphs, multi-modal graphs, graphs with parallel edges, and hypergraphs, Dense Sparse graph. I constructed the web graph by providing relevant data to the JUNG API. For each URL I have stored the number of outgoing and incoming links. A vertex was created out of each

**Name:Kunal Krishna**

**Net Id: kxk155230**

, and the directed edges going out from that vertex represent the outgoing links from that URL. So the input data given to the JUNG Framework was the URL of the crawled website and the incoming and outgoing links. My teammate Swastik who was responsible for crawling provided me with this data. We have used the HashMap Data structure for mapping of the URL and its incoming and outgoing links.

1. **Give statistics – e.g. the number of nodes, number of links, the largest number of ingoing links or outgoing links for the graph you have generated.**

****

Sample Web Graph

Above figure gives a sample representation of how the web graph is constructed. Each vertex (URL) has links going to other vertices as well as internal links which direct on same URL but different webpage. Each vertex also has incoming links from other vertices. We have gathered the following statistics from our generated web graph.

Number of nodes: 500385

Number of Links: 764769

Largest number of ingoing links: 542

Largest number of outgoing links: 147

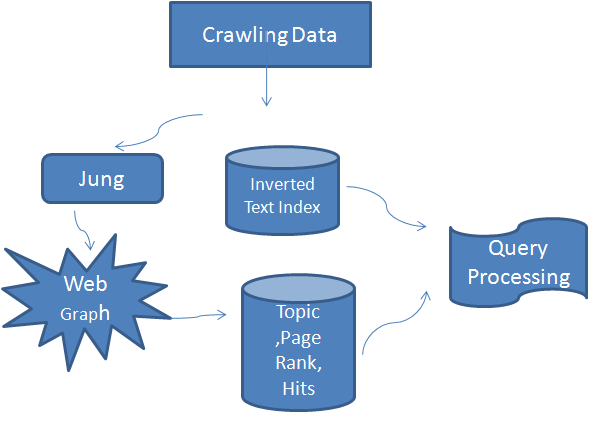
1. **Show how you connected the information from the graph to the index.**

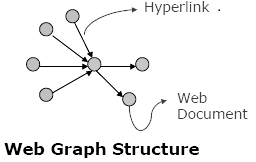
The web graph provides us with the information of each document. In order to build an index I extracted the data from each vertex of the graph. The indexing the documents containing

**Name:Kunal Krishna**

**Net Id: kxk155230**

book title, author name, ISBN, year and publisher. This way each vertex of the web graph was mapped in the Inverted index.





1. **Describe in detail the two relevance models that you created and provide the weighting schemes that you have used.**

I have used two relevance model using Page rank and Page Hit. Once we get the list of document we use to sort them using pagerank and page hit.

Ranking algorithms assign values to each vertex (or edge) according to a set of criteria that reflect structural properties of the network. These criteria are generally intended to measure the “influence”, “authority”, or “centrality” of a given vertex/edge. Several of these algorithms interpret their input as a Markov network: a directed weighted graph in which the vertices represent states, the edges represent possible state transitions, and the edge weights represent transition probabilities. (A directed graph with non-negative weights can be converted into a Markov network by normalizing the weights on the outgoing edges from each vertex to sum to 1.) Ranking algorithms that JUNG provides include BetweennessCentrality , which labels each vertex and edge in a graph with a value that is derived from the number of shortest paths that pass through it; PageRank which ranks each vertex in a modified Markov network according to

**Name:Kunal Krishna**

**Net Id: kxk155230**

its stationary probability; PageRankWithPriors a generalization of PageRank whose ranks are calculated relative to a specified set of root vertices;

HITS ,which ranks each vertex in a graph according to the “hubs-and-authorities” importance measures; KStepMarkov, which ranks each vertex according to a fast approximation of the PageRankWithPriors algorithm; and VoltageRanker ,which treats the network as an electrical circuit (with specified voltage sources and sinks) and assigns each vertex a value corresponding to its residual voltage in this circuit.

5)      Give an example of the topic based page ranks you have computed – showing which web pages of your search engine have the highest page ranks.

The topic based Page Rank of each vertex from the web graph is calculated using the JUNG framework.

Page Rank class in JUNG assigns scores to each vertex according to the Page Rank algorithm. ThePage Rankclass assigns weights to edges. The weights on the outgoing edges for a given vertex represent transition probabilities, that is, they sum to 1.After incorporating Page Rank in the search engine following web pages were retrieved with highest pages rank,

* http://www.goodreads.com/
* https://www.librarything.com/
* http://www.shelfari.com/

6)      Similarly, discuss the HITS scores and show which web pages have obtained the largest scores.

**HITS** algorithm is in the same spirit as **Page-Rank**. They both make use of the link structure of the Web graph in order to decide the relevance of the pages. The difference is that unlike the **Page Rank** algorithm, **HITS** only operates on a small sub-graph (the seed *S*Q) from the web graph. This sub--graph is query dependent; whenever we search with a different query phrase, the seed changes as well. **HITS** rank the seed nodes according to their authority and hub weights. The highest ranking pages are displayed to the user by the query engine.

To calculate the hits of the page first we need to create the object of the hits class and with the help of this object we need to create the sub-graph from the web graph. I set maximum iteration to 10 using setMaxIterations() method. After with the help of evaluate() function we will get the page hits of the all the nodes.

Below are the web-pages has highest HITS scores.

* http://www.goodreads.com/
* https://www.librarything.com/
* http://www.shelfari.com/

7)      Elaborate on how you have collaborated with the student responsible for the user interface in generating queries to test the relevance models and to display the results of your search engine.

For this module, I provided my teammate **Janmejaya**, an interface in JSP by which it was easy to perform searches on the generated index. This interface was used to fetch the search results. We have tested the search engine for 100 queries, and evaluated the query results based on the two relevance models that we have implemented. We generated a random set of queries from a book database. The generated queries consisted of book title, author, and headline.

After the results were displayed we analyzed the two relevance models that we have implemented. Comparing the results of the two relevance models with the search results from Google and bing we came to the conclusion that the Page rank gave better relevant results as compared to the Page Hit model for our search engine. For the Page rankmodel the first four URL’s which were retrieved were relevant, and the most non relevant URL was displayed as the last result. On the other hand the Page Hitrelevance model gave the first two links as relevant, but rest of the URL’s was irrelevant. The queries which we have tested for judging our relevance model are:

The queries which we have tested for judging our relevance model are:

|  |  |
| --- | --- |
|  |  |
|  |  |  |  |  |
| **Query** | **URL** | **Relevant/Non-** | **Google** | **Bing** |
|  |  | **Relevant** |  |  |
|  |  |  |  |  |
| **Harry Potter** | http://www.goodreads.com/ | Relevant |  |  |
|  |  |  |  |  |
|  | https://www.librarything.com/ | Relevant |  |  |
|  | http://www.goodreads.com/ | Relevant |  |  |
|  | http://www.goodreads.com/ | Relevant |  |  |
|  |  |  |  |  |
| **Narnia story** | https://www.librarything.com/ | Somewhat relevant |  |  |
|  | http://www.goodreads.com/ | Relevant |  |  |
|  | https://www.librarything.com/show/31304-a | Relevant |  |  |
|  |  |  |  |  |
| **Hotels of North** | https://www.librarything.com/ | Relevant |  |  |
| **America** |  |  |  |  |
|  | http://amazon.com/sales/books | Not Relevant |  |  |
|  |  |  |  |  |

8)      State clearly how many queries you have used, how you have generated them and how you have judged the results of your relevance models.

I have used 20 queries to test the model, I have generated based on the

Author of the book, example JK Rowling, David Mitchell,Rick Moody

Title of the book, Harry Potter, Narnia, Hotels of North America

9)      Elaborate on how you have collaborated with the student responsible for the clustering for improving your relevance models.

After using my relevance models for the query I passed my result to clustering model, Sarvotam who is responsible for clustering used java ML library to implement the clustering. He has created the cluster and based on my result, we tested our result to filter out the non relevant document, link and got more improved version of results. He passed the entire document to me which belongs to the query cluster (it is decided based on query).

**Clustering**

**Net ID: SXS155032**

**Name: Sarvotam Pal Singh**

**Describe how you have designed the flat clustering – how many predefined clusters did you select, and why?**

Clustering has been performed on the extracted information that were used to build the index, by using the K-means algorithm. The most important parameter in the K-means algorithm along with the distance function is the k, which represents the number of clusters to be created. For the purpose of this project and without having any prior knowledge of the data, I used hit and trial method for estimation of k, with various valued like 10, 20, 25 & 30. But finally deciding with value of K = 15 because better convergence results. The distance metric used was jaccard distance.

This is a very rough estimate that can be later adjusted by using well-known techniques that are based on the analysis of variation. The two types of variation that these methods optimize are the intra-cluster and inter-cluster variation. The optimization comes by minimizing the variation within the clusters and maximizing the variation across clusters.

**What did you do with the results of clustering – did you incorporate them in the relevance models – and did you provided to the user interface results that were obtained when clustering is used?**

After the clustering is performed, we have our 15 clusters. So, now when the query is entered the distance of the query with the 15 cluster centroids is calculated and we choose the top three clusters with whose centroid the distance of query was minimum. We choose the top three clusters because we do not want to miss the relevant documents which may be missed if we choose just the top cluster. Now, using the relevance model the weight if the query with different documents is calculated and then we display the top 20 results from all the three clusters combined.

The clustering results were used to provide to the user the option of search for relevant pages. For this part I collaborated with Janmejaya Sahoo (GUI Designer) in order to present the clustering results to our front end, using a separate tab. After a user submits a query and receives the most relevant results from our search engine, each of the URLs is being checked against the clustering the clustering module, to find the relative documents and presented to the user. We have only performed kmeans clustering algorithm to improve our results.

**How many queries did you experiment with – such that clustering could be used to improve the results of your search engine. State clearly how many queries you have used to test the impact of the results of each clustering method, how you have generated them and how you have judged the results of your relevance models?**

I have used 20 queries to test the impact of clustering algorithm on the results of our search engine. The queries were decided on the basis of data we have crawled and the queries chosen are versatile in nature

**Net ID: SXS155032**

**Name: Sarvotam Pal Singh**

varying from various authors, famous characters and books. The results were judged on the basis of comparison between the results by relevance model and the results after the clustering. The results shown after clustering were more relevant when opened and verified then the results that were shown after the query was run just on the relevance model

**Provide three examples of the queries and the results produced by your search engine and the clusters that you have created?**

**Query = Agatha Christie**

### Your query is: stephen hawking

### Our Search Engine Results on Cluster

|  |  |
| --- | --- |
| Book Name | ISBN |
| [My Brief History](https://www.goodreads.com/book/show/18209305-my-brief-history) | 0345535286 |
| [Stephen Hawking's Universe: The Cosmos Explained](https://www.goodreads.com/book/show/2099.Stephen_Hawking_s_Universe) | 0465081983 |
| [The Universe in a Nutshell](https://www.goodreads.com/book/show/2095.The_Universe_in_a_Nutshell) | 055380202X |
| [What's Out There: Images from Here to the Edge of the Universe](https://www.goodreads.com/book/show/2106.What_s_Out_There) | 1844831906 |
| [God Created The Integers](https://www.goodreads.com/book/show/2096.God_Created_The_Integers) | 0762419229 |
| [The Black Hole War: My Battle with Stephen Hawking to Make the World Safe for Quantum Mechanics](https://www.goodreads.com/book/show/2161966.The_Black_Hole_War) | 0316016403 |
| [The Nature of Space and Time](https://www.goodreads.com/book/show/2103.The_Nature_of_Space_and_Time) | 0691050848 |
| [The Illustrated A Brief History of Time/The Universe in a Nutshell](https://www.goodreads.com/book/show/2141233.The_Illustrated_A_Brief_History_of_Time_The_Universe_in_a_Nutshell) | 0307291227 |
| [The World Treasury of Physics, Astronomy & Mathematics from Albert Einstein to Stephen W. Hawking & from Annie Dillard to John Updike](https://www.goodreads.com/book/show/2109.The_World_Treasury_of_Physics_Astronomy_Mathematics_from_Albert_Einstein_to_Stephen_W_Hawking_from_Annie_Dillard_to_John_Updike) | 0316281336 |
| [The Dreams That Stuff Is Made Of: The Most Astounding Papers of Quantum Physics--and How They Shook the Scientific World](https://www.goodreads.com/book/show/8534601-the-dreams-that-stuff-is-made-of) | 0762434341 |

### Your query is: katniss

### Our Search Engine Results on Cluster

|  |  |
| --- | --- |
| Book Name | ISBN |
| [Mockingjay (The Hunger Games #3)](https://www.goodreads.com/book/show/7260188-mockingjay) | 0439023513 |
| [The Girl Who Was on Fire: Your Favorite Authors on Suzanne Collins' Hunger Games Trilogy (The Hunger Games Companions)](https://www.goodreads.com/book/show/8046680-the-girl-who-was-on-fire) | 1935618040 |
| [The Hunger Games (The Hunger Games #1)](https://www.goodreads.com/book/show/2767052-the-hunger-games) | 0439023483 |
| [The Hunger Games and Philosophy: A Critique of Pure Treason (Blackwell Philosophy and Pop Culture #28)](https://www.goodreads.com/book/show/10776887-the-hunger-games-and-philosophy) | 1118065077 |
| [The World of the Hunger Games (The Hunger Games Companions)](https://www.goodreads.com/book/show/11873368-the-world-of-the-hunger-games) | 0545425123 |
| [Artemis: The Indomitable Spirit in Everywoman](https://www.goodreads.com/book/show/21543946-artemis) | 1573245917 |
| [The Hunger Games Trilogy Boxset (The Hunger Games #1-3)](https://www.goodreads.com/book/show/7938275-the-hunger-games-trilogy-boxset) | 0545265355 |
| [Texts from Jane Eyre: And Other Conversations with Your Favorite Literary Characters](https://www.goodreads.com/book/show/20695981-texts-from-jane-eyre) | 1627791833 |

### Your query is: agatha christie

### Our Search Engine Results on Cluster

|  |  |
| --- | --- |
| Book Name | ISBN |
| [Endless Night](https://www.goodreads.com/book/show/16366.Endless_Night) | 0007151675 |
| [The Christie Curse (A Book Collector Mystery #1)](https://www.goodreads.com/book/show/15808728-the-christie-curse) | 042525528X |
| [The Adventure of the Christmas Pudding (Hercule Poirot #33)](https://www.goodreads.com/book/show/690146.The_Adventure_of_the_Christmas_Pudding) | 0007121083 |
| [Death Comes As the End](https://www.goodreads.com/book/show/121645.Death_Comes_As_the_End) | 0312981619 |
| [Sparkling Cyanide (Colonel Race #4)](https://www.goodreads.com/book/show/212622.Sparkling_Cyanide) | 0007136854 |
| [Come, Tell Me How You Live](https://www.goodreads.com/book/show/111516.Come_Tell_Me_How_You_Live) | 0671432826 |
| [The Sittaford Mystery](https://www.goodreads.com/book/show/16339.The_Sittaford_Mystery) | 0007136846 |
| [Lord Edgware Dies (Hercule Poirot #9)](https://www.goodreads.com/book/show/215492.Lord_Edgware_Dies) | 0002314576 |
| [A is for Arsenic: The Poisons of Agatha Christie](https://www.goodreads.com/book/show/23848320-a-is-for-arsenic) | 147291130X |
| [Postern of Fate (Tommy and Tuppence #5)](https://www.goodreads.com/book/show/102311.Postern_of_Fate) | 0007111487 |

**Query Expansion sak120230 SAQUIB AZIZ KHAN**

I implemented Associative Clustering and Metric clustering for Query Expansion. To do this, I got the document set for the first 50 documents from the regular query. Using this local document set, I created a local vocabulary with its frequency distributions and used those for calculation of two metrics(associative and metric). For every word in query, I got 2 unique words using clustering and ran the expanded query to get results that were more in line with expected results for a particular query. Below are certain examples:

**QUERY : HARRY**

1. Associative Clustering :
   1. Output for query without expansion: (The highlighted records are irrelevant)

|  |
| --- |
| **Book Name** |
| Harry, a History: The True Story of a Boy Wizard, His Fans, and Life Inside the Harry Potter Phenomenon |
| The Psychology of Harry Potter: An Unauthorized Examination Of The Boy Who Lived |
| Harry Potter and the Chamber of Secrets (Harry Potter #2) |
| Harry Potter and the Half-Blood Prince (Harry Potter #6) |
| Harry Potter and the Cursed Child (Harry Potter) |
| What's a Christian to Do with Harry Potter? |
| Harry and the Dinosaurs Go To School (Harry and the Dinosaurs) |
| The Heaven Tree Trilogy (Heaven Tree #1-3) |
| The Sorcerer's Companion: A Guide to the Magical World of Harry Potter |
| The Overlook (Harry Bosch #13) |

* 1. Local Document Set (Actual used were 50 documents. Mentioned here are only 6 documents as each has a lot of content):

The detailed documents are given in a separate file at the end of this report.

* 1. Local Vocabulary &Correlation values:

|  |  |
| --- | --- |
| potter | 0.272079331 |
| book | 0.106910809 |
| brown | 0.100715137 |
| away | 0.099879663 |
| prince | 0.092896175 |
| whole | 0.092896175 |
| muggles | 0.092896175 |
| month | 0.080097087 |
| favourite | 0.075560803 |
| chamber | 0.074303406 |
| examine | 0.071647275 |
| three | 0.061557789 |
| bear | 0.05830721 |
| affair | 0.05830721 |
| stoker | 0.049938348 |
| rubeus | 0.049938348 |
| inside | 0.044665012 |
| azkaban | 0.04425914 |

* 1. Expanded Query:  
     **harry potter favourite**
  2. Output: ( No irrelevant records)

|  |
| --- |
| Book Name |
| [Harry Potter and the Chamber of Secrets (Harry Potter #2)](https://www.goodreads.com/book/show/15881.Harry_Potter_and_the_Chamber_of_Secrets) |
| [What's a Christian to Do with Harry Potter?](https://www.goodreads.com/book/show/124979.What_s_a_Christian_to_Do_with_Harry_Potter_) |
| [Harry Potter Boxset (Harry Potter #1-7)](https://www.goodreads.com/book/show/862041.Harry_Potter_Boxset) |
| Harry Potter and the Order of the Phoenix (Harry Potter #5) |
| Harry Potter Hardcover Boxed Set, Books 1-6 (Harry Potter #1-6) |
| The Sorcerer's Companion: A Guide to the Magical World of Harry Potter |
| [Harry, a History: The True Story of a Boy Wizard, His Fans, and Life Inside the Harry Potter Phenomenon](https://www.goodreads.com/book/show/3130430-harry-a-history) |
| [Harry Potter and the Half-Blood Prince (Harry Potter #6)](https://www.goodreads.com/book/show/1.Harry_Potter_and_the_Half_Blood_Prince) |
| [Fact, Fiction, and Folklore in Harry Potter's World: An Unofficial Guide](https://www.goodreads.com/book/show/15873.Fact_Fiction_and_Folklore_in_Harry_Potter_s_World) |
| [Harry Potter and the Cursed Child (Harry Potter)](https://www.goodreads.com/book/show/29056083-harry-potter-and-the-cursed-child) |

**Query Expansion sak120230 SAQUIB AZIZ KHAN**

**Query Expansion sak120230 SAQUIB AZIZ KHAN**

1. Metric Clustering
   1. Query Output without expansion: (Irrelevant records are highlighted)

|  |
| --- |
| Harry, a History: The True Story of a Boy Wizard, His Fans, and Life Inside the Harry Potter Phenomenon |
| The Psychology of Harry Potter: An Unauthorized Examination Of The Boy Who Lived |
| Harry Potter and the Chamber of Secrets (Harry Potter #2) |
| Harry Potter and the Half-Blood Prince (Harry Potter #6) |
| Harry Potter and the Cursed Child (Harry Potter) |
| What's a Christian to Do with Harry Potter? |
| Harry and the Dinosaurs Go To School (Harry and the Dinosaurs) |
| The Heaven Tree Trilogy (Heaven Tree #1-3) |
| The Sorcerer's Companion: A Guide to the Magical World of Harry Potter |
| The Overlook (Harry Bosch #13) |
| Harry, a History: The True Story of a Boy Wizard, His Fans, and Life Inside the Harry Potter Phenomenon |

* 1. Local Document Set: This is same as the one for Associative Cluster
  2. Local Vocabulary & Correlation values:  
     Top two are selected because they have the highest values of s-term.

|  |  |
| --- | --- |
| S TERM VALUES | TERM |
| 0.334587631 | potter |
| 0.147235906 | book |
| 0.136513158 | world |
| 0.117164616 | edition |
| 0.106910809 | first |
| 0.100715137 | series |
| 0.099879663 | ron |
| 0.096676737 | language |
| 0.092896175 | title |
| 0.092896175 | isbn |
| 0.092896175 | english |
| 0.086165049 | azkaban |
| 0.081807082 | one |
| 0.080097087 | wizard |
| 0.075560803 | bosch |
| 0.074303406 | character |
| 0.073484385 | story |
| 0.071647275 | hogwart |

* 1. Expanded Query:  
      harry potter book
  2. Output for Expanded Query:

|  |
| --- |
| **Book Name** |
| Harry Potter Boxset (Harry Potter #1-7) |
| Harry Potter and the Chamber of Secrets (Harry Potter #2) |
| Harry Potter and the Half-Blood Prince (Harry Potter #6) |
| Harry Potter and the Order of the Phoenix (Harry Potter #5) |
| What's a Christian to Do with Harry Potter? |
| Harry Potter Hardcover Boxed Set, Books 1-6 (Harry Potter #1-6) |
| Harry Potter and the Cursed Child (Harry Potter) |
| The Sorcerer's Companion: A Guide to the Magical World of Harry Potter |
| Harry, a History: The True Story of a Boy Wizard, His Fans, and Life Inside the Harry Potter Phenomenon |
| Fact, Fiction, and Folklore in Harry Potter's World: An Unofficial Guide |

If you notice closely at the highlighted books in regular query results and their lack in output for expanded queries, it becomes clear that there is a clear advantage in using query expansion in our relevance model.  
  
However, certain queries that are generic in nature. Example: “classics about 19th century literature” find expansions which are not very useful. I think what we need for generic queries is topic based clustering. Simple K-means clustering, that we have implemented in our project, gives us more diverging results that ultimately do not cluster well for such generic queries.