Information Retrieval
Ranking and Search for Textual Documents

Aryan Mehra - 2017A7PS0077P
Siddhant Khandelwal - 2017A7PS0127P
Department of Computer Science and Information Sciences
Birla Institute of Technology and Science, Pilani

7 March 2020

Introduction and Aim

This project deals with building a text query based search engine that runs on a sample corpus of HTML Wikipedia pages. The name of the corpus file is megawikicorpus.txt and is provided in the folder as well. The main idea of the project is to make an end to end system that reads and processes the HTML text corpus of Wikipedia pages, and creates an easy to use index on them. This index is to be further used to query the corpus for relevant documents through single or multi-term queries. We even apply multiple heuristics to improve the search results and thus better the performance of the search engine.

Code Status and Features of Search Engine Prototype

We achieve the following with our prototype:

- Document Pre-processing: LNC
- Document Vector Building
- Query Processing: Complete
- Scoring of Documents
- Sample Query Analysis

Phase 1 explanation

In the first phase, we make a pass over the corpus thus making a database of the documents individually and also establishing the vocabulary of words that occur in the entire corpus. This vocabulary is indexed as it enters the "vocabulary" and the reverse mapping of the word to the index is maintained in another small "inverse vocabulary" meta-data file. Thus, the *lnc* processed documents vector, the vocabulary and the inverse vocabulary mapping are all stored as one time synthesised files. We also store a small metadata file that stores the stemmed versions of the vocabulary words, which is later used in one of our heuristics. Another metadata file is the "title" file for every document. We cleverly extract the title from within the doc tags. Thus, all in all the phase 1 is all about processing the documents within the doc tags, extracting the title and, most importantly, creating multiple short metadata files about the database. Since the corpus is small enough to be dealt on personal computer systems, much attention is not diverted to the space and time efficiency of the search model. Rather, the improvement in results and scoring is improved.

Query Analysis

We now test our model on 10 different multi-term queries. We report the relevance of the documents that are fetched by manually reading the documents for content and terms that match the searched terms. The Y/N in the relevance column thus represents a subjective score. These answers are reported upto a relevant decimal place only and are reported without the use of any heuristic.

Query	Top K Documents	Score	Relevance
football competition	FC Den Bosch	0.1192867	Y
	FIFA World Cup	0.0889931	Y
	FA Cup	0.0878264	Y
	Premiere League	0.079861	Y
	Snap (gridiron football)	0.0742975	Y
	Bobby Charlton	0.0727690	Y
	Bob Young (businessman)	0.0712576	N
	Fulham F.C.	0.0705541	Y
	Bristol City F.C.	0.0646919	Y
	British Steel (Historic)	0.0643389	N

Table 1: Query 1 - **football competition**

Query	Top K Documents	Score	Relevance
christian dinomination	Free Methodist Church	0.0754253	Y
	Fundamentalism	0.0651003	Y
	Baptism	0.06233413	Y
	Bethlehem	0.0557221	Y
	Baptists	0.05493393	Y
	Bishop	0.0512365	Y
	Feminist theology	0.04531049	Y
	Book of Mormon	0.0443321	Y
	Bank of England	0.0408356	N
	Demographics of Grenada	0.0388912	N

Table 2: Query 2 - **christian dinomination**

Query	Top K Documents	Score	Relevance
rock band music	Barış Manço	0.1395596	Y
	Liverpool (album)	0.1331696	Y
	Belle and Sebastian	0.129195	Y
	Bar Kokhba (album)	0.1196155	Y
	Boy band	0.1147368	Y
	Bill Haley	0.113276	Y
	Blue Öyster Cult	0.112998	Y
	Bob Wills	0.110501	Y
	Goran Bregović	0.1037685	Y
	Buddy Holly	0.1009581	Y

Table 3: Query 3 - **rock band music**

Query	Top K Documents	Score	Relevance
battery phone	Telecommunications in Greenland	0.0618311	Y
	Battery Park City	0.05649474	N
	Game Boy line	0.04230679	Y
	Communications in Gibraltar	0.0368607	Y
	General Motors	0.0327508	N
	Battleship	0.02790949	N
	Fuel cell	0.0260803	N
	Battle of Waterloo	0.0236865	N
	Battle of Blenheim	0.02277083	N
	Frigate	0.02257449	N

Table 4: Query 4 - **battery phone**

Query	Top K Documents	Score	Relevance
comedy movies	Film genre	0.1147181	Y
	Frontline (Australian TV series)	0.0726993	Y
	Bruce Campbell	0.0668738	N
	Fantasy film	0.06538017	Y
	Bollywood	0.0586906	Y
	Bubblegum Crisis	0.05713859	N
	Frank Capra	0.05205351	N
	Father Ted	0.05147715	N
	Four Weddings and a Funeral	0.0510764	Y
	Brian De Palma	0.0505191	N

Table 5: Query 5 - **comedy movies**

Query	Top K Documents	Score	Relevance
windows computer microsoft	Badtrans	0.1143165	Y
	Graphical user interface	0.1134235	Y
	BASIC	0.1112434	N
	Blitz BASIC	0.1017462	N
	BeOS	0.0955757	Y
	BIOS	0.0776102	Y
	Borland	0.07582191	N
	File manager	0.0666258	Y
	Break key	0.0627472	N
	Buffer overflow	0.06105632	N

Table 6: Query 6 - **windows computer microsoft**

Query	Top K Documents	Score	Relevance
drink tea coffee	Bubble tea	0.16386	Y
	Bombay Sapphire	0.03682670	Y
	Flores	0.0343131	N
	Economy of Guinea	0.0314853	N
	Frasier	0.0254748	N
	Fellatio	0.02477133	N
	Economy of Guatemala	0.02445179	N
	George Orwell	0.02333152	N
	Food writing	0.02264204	Y
	Blood alcohol content	0.02217356	Y

Table 7: Query 7 - **drink tea coffee**

Query	Top K Documents	Score	Relevance
electrons and protons	Elementary particle	0.0817338	Y
	Bohr model	0.0730335	Y
	BCS theory	0.0689133	Y
	Beta decay	0.0663905	Y
	Fermion	0.063128	Y
	Fuel cell	0.057641	N
	Baryon	0.051899	Y
	Bioleaching	0.049208	N
	Big Bang	0.04833084	Y
	Ferromagnetism	0.0441311	Y

Table 8: Query 8 - **electrons and protons**

Query	Top K Documents	Score	Relevance
television show	Frontline (Australian TV series)	0.120598140	Y
	Fred Savage	0.11930272	Y
	Four Feather Falls	0.09788046	Y
	Father Dougal McGuire	0.0968484	Y
	Bill Oddie	0.096056	Y
	Telecommunications in Greece	0.0943406	N
	Father Ted	0.0895852	Y
	Buffy the Vampire Slayer (film)	0.0878782	Y
	BBC Red Button	0.0873083	Y
	Telecommunications in Gabon	0.0868316	N

Table 9: Query 9 - **television show**

Query	Top K Documents	Score	Relevance
genetic disorder	Biotechnology	0.0754359	Y
	Gamete	0.0660261	Y
	Bipolar disorder	0.06444745	Y
	Bacterial conjugation	0.048789	Y
	Base pair	0.03942837	N
	Genetics	0.0386768	Y
	Cell (biology)	0.0375410	N
	Benzodiazepine	0.0348380	Y
	Bongo (antelope)	0.0330083	N
	Bacteriophage	0.0326802	Y

Table 10: Query 10 - **genetic disorder**

Improvements and Heuristic Innovations

We now try to improve the model proposed above on the condition that we keep using *lnc.ltc* and the same data structure that we have saved in the Part 1 of the model code, in the form of the incidence matrix. We use the metadata files to our rescue for the same purpose. We present three simple heuristics to improve the user interaction and result relevance in case of search.

Issues and Limitations with previous model

- There is no provision for wrongly spelled queries entered in the engine. The engine simply rejects the query or gives less relevant results.
- In case the user enters words that are not present in the corpus itself, there is no mechanism of giving suggestions of similar rooted words that are there in the corpus.
- The search results are based on a simple search score based on cosine similarity only. There is no means of weighting the documents with relevance based on title or any zone in the document. Clearly, in case of Wikipedia documents, the ones with the same query words in the title are more important that other documents.

Heuristic 1 - Finding similar root words for non vocabulary words

If we can't find a word in the vocabulary, instead of completely skipping it altogether, like a normal vector space implementation, we instead try our best to find the root of the word. Then we lookup at the metadata structure that tells us the word in the vocabulary that occurs with the maximum frequency with that root word (stemmed version) . We then ask the user to identify that as the best possible guess to the word he typed. The example shows how "Two Tribes" was very important before this heuristic only because of the occurrence of lonely. There is no "widower" in this file. Also the "Book of Ruth" that actually does have

"widow" in it, is not there in the list at all. When we apply the heuristic there is a drastic positive effect on the relevance of the documents that are fetched. The "information retrievals" query asks the user to proceed with substituting "retrievals" with the word "retrieved", thus not basing it's search solely on "information" but also on the other word. Some relevant results like the "Gemini 10" mission document about retrieved lunar data are ranked now, along with "Binary Search Tree", which is also about retrieved nodes and information storing data structures. Similarly in the third example of "Computer Clocking" as well we see some more relevant results like "binary prefix" and "FIFO" hardware based articles being ranked among top 10, which was not seen before the heuristic is applied.

Query	Top K Documents	Score
lonely widowers	Gheorghe Zamfir	0.0638761
	Bestiary	0.0284889
	Two Tribes	**0.0251270**
	Baku	0.0237896
	The Beach Boys	0.02007620
	Golem	0.0173950
	Blue Öyster Cult	0.0169219
	Béla Bartók	0.0165081
	George Orwell	0.0157471
	Francis of Assisi	0.01555567

Table 11: Query - lonely widower without heuristic for rooted word search on "widower"

Query	Top K Documents	Score
lonely widowers	Gheorghe Zamfir	0.04866887
	Book of Ruth	**0.0330344**
	Berry Berenson	0.03060701
	Gaudy Night	0.029974901
	Bram Stoker	0.02537709
	George Orwell	0.02219702
	Bestiary	0.02170646
	Book of Lamentations	0.0213263
	Bill Haley	0.01948232
	Two Tribes	**0.0191449**

Table 12: Query - lonely widower with heuristic for rooted word search on "widower"

Query	Top K Documents	Score
information retrievals	Full disclosure (computer security)	0.1143971
	Gregory Chaitin	0.10796625
	Factoid	0.10323326
	List of fictional guidebooks	0.095812
	FileMan	0.09395816
	Foresight Institute	0.0932866
	Telecommunications in Guinea	0.0891554
	FBI Most Wanted Terrorists	0.0827613
	Burnt-in timecode	0.0783396
	Fourth-generation programming language	0.07764508

Table 13: Query - "Information Retrievals" without root word heuristic

Query	Top K Documents	Score
information retrievals	Gemini 10	0.0493142
	Full disclosure (computer security)	0.03331633
	Binary search tree	0.0317582
	Gregory Chaitin	0.0314434
	Factoid	0.03006501
	List of fictional guidebooks	0.0279038
	FileMan	0.02736379
	Foresight Institute	0.02716823
	Telecommunications in Guinea	0.02596507
	FBI Most Wanted Terrorists	0.02410288

Table 14: Query - "Information Retrievals" with root word heuristic

Query	Top K Documents	Score
computer clocking	Bill Atkinson	0.12846533
	Fred Brooks	0.12663561
	Bill Schelter	0.1246882
	BQP	0.11866102
	Bastard Operator From Hell	0.1176504
	Brian Kernighan	0.1097326
	Burroughs Corporation	0.1054794
	Badtrans	0.1005509
	Backplane	0.0935088
	Bob Young (businessman)	0.09118767

Table 15: Query - "Computer Clocking" without the rooted heuristic

Query	Top K Documents	Score
computer clocking	Bill Atkinson	0.0599711
	Fred Brooks	0.0591170
	Binary prefix	0.0588375
	Bill Schelter	0.0582079
	BQP	0.055394
	Bastard Operator From Hell	0.0549224
	Global Positioning System	0.0512778
	Brian Kernighan	0.0512262
	FIFO (computing and electronics)	0.0502513
	Bill Haley	0.0502376

Table 16: Query - "Computer Clocking" with the rooted heuristic

Heuristic 2 - Post score title weighting

The reason this heuristic will work very well is that we are working on HTML pages of web (Wikipedia) corpus. We use the title metadata to give more weight to documents that have the query words in the title itself. It is intuitive that the title has more importance especially given that we are dealing with Wikipedia corpus. Hence instead of applying zone weighting from the very beginning, we increase the overall score of the documents in which a query term occurs in the title by a factor of 0.1 for every query term. In the example shown below, clearly the user wants articles on a particular "George". But the occurrence of the proper noun in other documents gives several less relevant results that may have George within the text, but are not about a said "George". Similarly the second result in "United Kingdom" becomes much more relevant than before.

Query	Top K Documents	Score
George	George Pappas	0.11732033
	Benjamin Franklin-class submarine	0.0907473
	Fred Reed	0.0892120
	Branch Davidians	0.0852340
	George Whipple	0.0823836
	History of Grenada	0.06243350
	Frederick William, Elector of Brandenburg	0.06190684
	Liberal Party (UK)	0.056414096
	FileMan	0.0554932
	Batman amp Robin (film)	0.054744036

Table 17: Query - "George" without the title weighting heuristic

Query	Top K Documents	Score
George	George Pappas	0.217320331
	George Whipple	0.18238366
	George Lucas	0.15004012
	George H. W. Bush	0.1486493
	George Berkeley	0.14711856
	George R. R. Martin	0.1457683
	George Washington	0.14346174
	George Orwell	0.13618248
	Georges Braque	0.10000
	Benjamin Franklin-class submarine	0.090747311

Table 18: Query - "George" with the title weighting heuristic

Query	Top K Documents	Score
United Kingdom	List of political scandals in the United Kingdo	0.2215823
	Balfour Declaration of 1926	0.1402482
	Foreign relations of Georgia	0.11444124
	Foreign relations of Grenada	0.1135159
	Fusion cuisine	0.1106803
	Furlong	0.0955718
	Baron Aberdare	0.094950618
	Politics of Gibraltar	0.0931745
	Foreign relations of Guinea	0.086883407
	Boxing Day	0.086192162

Table 19: Query - United Kingdom (without title weight heuristic)

Query	Top K Documents	Score
United Kingdom	List of political scandals in the United Kingdom	0.4215823
	United Kingdom general election, 2001	0.2564782
	Federal jurisdiction (United States)	0.1509960
	Flag of the United States	0.14543221
	Gulf Coast of the United States	0.14484035
	Balfour Declaration of 1926	0.14024827
	United States Foreign Intelligence Surveillance Court	0.12560322
	Foreign relations of Georgia	0.11444124
	Foreign relations of Grenada	0.1135159
	Fusion cuisine	0.1106803050

Table 20: Query - United Kingdom (with title weighting)

Query	Top K Documents	Score
New York	Bob Frankston	0.1754979
	Big Apple	0.1569208
	Brooklyn Historic Railway Association	0.119393
	Futurians	0.11439808
	Futurama (New York World's Fair)	0.09458171
	Fiorello H. La Guardia	0.0943277
	Fantasy Games Unlimited	0.0942019
	William M. Tweed	0.093652939
	Battery Park City	0.09113768
	Buffalo Bills	0.0895386

Table 21: Query - "New York" without title weighing

Query	Top K Documents	Score
New York	**Futurama (New York World's Fair)**	0.294581710
	Buffalo, New York	0.2773849
	Bob Frankston	0.175497976
	Big Apple	0.15692089
	Brooklyn Historic Railway Association	0.11939342
	Futurians	0.11439808
	BBC News (TV channel)	0.11385650
	Fiorello H. La Guardia	0.094327715
	Fantasy Games Unlimited	0.09420196
	William M. Tweed	0.09365293

Table 22: Query - "New York" with title weighing

Spelling Correction - (Bonus) Heuristic 3

We apply a spelling corrector as a bonus heuristic to improve the results. (This method is also suggested in the assignment itself). The user has the flexibility to get relevant results even when the query is misspelled. The first two tables show the difference between common nouns that are misspelled whereas the next table shows results for "Gautam Budha" spell corrected to "Gautama Buddha". Many relevant results are achieved through this heuristic.

Query	Top K Documents	Score
asian **lamgauges** (without spellcheck)	Fusion cuisine	0.1093170
	Balalaika	0.0551926
	Fairmount, Indiana	0.05079570
	Body mass index	0.05008624
	Fu Manchu	0.0423702
	Man Booker Prize	0.0416458
	Federated States of Micronesia	0.0397719
	Bali	0.03869606
	Bodhidharma	0.0379339
	Bollywood	0.03619452

Table 23: Misspelled query - asian lamgauges

Query	Top K Documents	Score
asian **lamgauges** (with spellcheck)	Baltic languages	0.088237
	Fusion cuisine	0.0854153
	Fricative consonant	0.0796636
	Fatherland	0.0738888
	Persian language	0.0737351
	Bantu languages	0.0713508
	Far East	0.0663319
	Fourth-generation programming language	0.0649245
	Demographics of Guinea	0.0625610
	Frisians	0.0611574

Table 24: Misspelled query with heuristics - asian lamgauges corrected to asian languages

Query and Description	Top K Documents	Score
no documents are fetched for "gautam budha" but the spell correction heuristic version gives results for "Gautama Buddha"	Four Noble Truths	0.078432
	Bodhidharma	0.0510125
	Buddhist philosophy	0.037520
	Bronze	0.0130262
	Bahá'í Faith	0.0091130
	Bertrand Russell	0.008623

Table 25: The misspelled query "Gautam Budha" yields no results but the spell corrected version yields very relevant results for "Gautama Buddha"

Although several other examples can be provided for good spell correction done by the search engine simulation presented here, more examples are left for the reader to explore through the code. The example of "Gautam Budha" above is unique in the fact that it presents the query in local dialect (Hindi) and converts it to customize itself to the English version of the both the words to fetch the relevant results.

Limiting case found (Exception)

One limiting case found to the use of all heuristics together was that since the spell check that we use is non contextual and based on a python library, some regional words that may be very relevant to the user are misinterpreted by the library to be closely related misspelled words. Example: the word "bollywood" is spell corrected to "hollywood" because the word although exists in the corpus, but is detected as a spelling error by the python "pyspellcheck" library. The result of this is that the HTML document titled "Bollywood" is definitely fetched, but it is at position (rank) 4 with a score of 0.0739.

About the authors, future Scope and open source contribution

The authors - Aryan Mehra and Siddhant Khandelwal - are pursuing their Bachelors in Engineering in Computer Science from BITS Pilani, India. Both of them have prior experience with Machine Learning and Vision based enhancement. This is one of their first endeavors in information retrieval based models for text extraction and search. Further the authors want to extend this project to build an automated web crawler for web based extraction on institutional data and corpora. The attached python notebooks have a detailed explanation of the above implementation and has been or will be made open source on the authors github page(s).