IR Assignment - 1

Akashdeep S
181IT203

Data collection

The dataset consists of 100 Google news articles across 5 categories namely politics, health, entertainment, sports and finance with 20 articles from each category.
This data is collected using the google news RSS API (https://news.google.com/rss/)
This API provides the following information on each article
Headline - headline of the article.
URL - link to the article.
Source - the article source.
Source URL - link to the source.
Category - category of the article.

The content of the article is scraped using a python library 'newspaper3k' which uses the URL of the article to download the content.

Dataset sample -

	headline	**url**	**source**	**source_url**	**category**	**text**
0	Manappuram Finance to consider fund raising op...	https://www.business-standard.com/article/news...	Business Standard	https://www.business-standard.com	finance	Manappuram Finance said the company may consid...
1	PayU aims to provide full-stack financial serv...	https://www.business-standard.com/article/comp...	Business Standard	https://www.business-standard.com	finance	Finance, a digital lending player backed by Pr...

Preprocessing

The initial number of tokens in the corpus is 10,238
The size of the corpus = 3,47,516 characters

Sample corpus text - " \n\nThe company is considering various options for raising funds through borrowings including by the way of issuance of various debt securities in onshore/offshore securities market by Public Issue "

The following preprocessing techniques are applied

Normalization

Punctuation removal:

All the punctuations in the corpus are removed. The new line tokens present in the corpus are replaced by space. There is an increase in the number of terms as the new line tokens are removed.

Example: " Papers.\n\nBased " is converted to " Papers Based "

After removing punctuations -
The number of terms = 10530
The size of the corpus = 336211 characters

Case folding :

All alphabets in the corpus are converted to lowercase.

Example: " Manappuram Finance said the company may consider and approve issuances of Debt Securities during September 2021 "
is converted to
" manappuram finance said the company may consider and approve issuances of debt securities during september 2021 "
After case folding -
The number of terms = 9431
The size of the corpus = 336211 characters

The number of terms in the corpus has reduced as the same terms with different cases are considered as 1 term after case folding. Whereas the number of characters in the corpus has remained the same.

Tokenization

It cuts the character sequence into a word sequence.

Example: " manappuram finance said the company may consider and approve issuances ... "
is converted to
['manappuram', 'finance', 'said', 'the', 'company', 'may', 'consider', 'and', 'approve', 'issuances', …]

After tokenization -
The number of terms = 9431
The size of the corpus = 277687 characters

The number of terms remains the same but the number of characters has reduced since space is not considered after tokenization.

Stopwords removal

The most common words in English with very little semantic value is removed from the corpus.
Examples of stopwords are 'i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're" etc.

Example: ['manappuram', 'finance', 'said', 'the', 'company', 'may', 'consider', 'and', 'approve', 'issuances', …]
is converted to
['manappuram', 'finance', 'said', 'company', 'may', 'consider', 'approve', 'issuances' …]

After stopwords removal -
The number of terms = 9296
The size of the corpus = 219075 characters

Both the number of terms and characters has reduced after removing 135 stopwords from the corpus.

Stemming

All the words in the corpus are reduced to their root form without the knowledge of the context. Porter stemmer is used for stemming.

Example: ['manappuram', 'finance', 'said', 'company', 'may', 'consider', 'approve', 'issuances' …]
is converted to
['manappuram', 'financ', 'said', 'compani', 'may', 'consid', 'approv', 'issuanc', …]

The number of terms and characters in the corpus after applying different types of stemmers

	Porter	**Snowball**	**Lancaster**
No.of terms	7097	7062	6280
No.of characters	189381	189436	165753

Both the number of terms and characters has reduced after stemming all the words to their root form using Porter stemmer.
Snowball Stemmer is an updated version of Porter’s Stemmer with new rules that were introduced modifying some of the existing ones already existing in Porter’s Stemmer. Hence better reduction in the number of terms.
Lancaster stemmer's rules are more aggressive than Porter and Snowball and it is one of the most aggressive stemmers as it tends to over stem a lot of words. Hence a huge reduction in the number of terms and characters.

Lemmatization

Lemmatization reduces the inflectional/variant forms to a base form called a lemma. It is aware of the meaning of the words.

The WordNet lemmatizer is used.
Example: ['manappuram', 'finance', 'said', 'the', 'company', 'may', 'consider', 'and', 'approve', 'issuances', …]
is converted to
['manappuram', 'finance', 'said', 'company', 'may', 'consider', 'approve', 'issuance', ...]

The number of terms and characters in the corpus after applying different types of lemmatizers

	WordNet	**Spacy**
No.of terms	8493	7803
No.of characters	214699	209500

Both the number of terms and characters has reduced after using the WordNet lemmatizer but the number of terms is greater than stemming as wordnet also uses the semantic context of the words while processing.
Spacy determines the part-of-speech tag by default and assigns the corresponding lemma. Hence the number of terms is lesser than that of Wordnet.

Summary

	No.of terms	**No.of characters**
Initial	10238	347516
Normalization	9431	336211
Tokenization	9431	277687
Stopwords removal	9296	219075
Stemming	7097	189381
Lemmatization	8493	214699

Data structure

The following data structures are used for the construction of the inverted index. Considering n as the number of terms.

List

A list is used to store the index terms. A 2D list is used to store the documents corresponding to the terms. The index of the terms in the list is used to map to the corresponding documents containing the terms in the 2D list. Implemented using python list.

- The storage required for storing the index terms is in the order of the number of terms present. Hence space complexity is O(n).
- The time complexity for insertion of 1 term is O(n) because while inserting a term, we first have to check if the term is present in the list or not which takes the order of O(n) and then append the term to list if it's not already present which takes O(1) time.
- The time complexity for retrieval of 1 term is O(n) because we have to go through the entire list which is in the order of O(n).
- The time complexity for deletion of 1 term is O(n) because we first have to find the term in the list which takes O(n) time and then remove the term from list if it's present which takes O(n) time.

Hashmap

The index terms and the posting lists are stored as key-value pairs with terms as keys. The python dictionary is used as hashmap.

- The storage required for storing the index terms is in the order of the number terms present. Hence space complexity is O(n) where n is the number of terms.
- The time complexity for insertion of 1 term is O(1) in the average case. In the worst case its O(n) because we're producing the hash of the index term which takes O(1) and in the worst case all terms have same hash and we would have to iterate through the keys with same hash, to check if current term is present or not which takes O(n). But this happens rarely, so the amortized time complexity of insertion is O(1).
- The time complexity for retrieval of 1 term is O(1) in the average case. In the worst-case, it's O(n) if all terms have the same hash and we would have to iterate through the keys with the same hash, to check if the current term is present or not which takes O(n). But this happens rarely, so the amortized time complexity of retrieval is O(1).
- The time complexity for deletion of 1 term is O(1) in the average case. In the worst case, it's O(n) due to the same reason explained above.

Binary search trees

The index terms and the posting lists are stored as key and value attributes of nodes in a balanced binary search tree. The python sorted dictionary module implements a balanced binary search tree to store key value pairs and is used here.

- The storage required for storing the index terms is in the order of the number terms present. Hence space complexity is O(n) where n is the number of terms.
- The time complexity for insertion of 1 term is O(logn) because traversing the tree to insert the node takes O(logn) time.
- The time complexity for retrieval of 1 term is O(logn) because traversing the tree takes O(logn) time.
- The time complexity for deletion of 1 term is O(logn) because traversing the tree to find the node takes O(logn) time and applying various rotations to maintain tree balance takes O(1) time.

Comparison

The amortized time complexities of the data structures.

	List	Hashmap	BST
Insertion	O(n)	O(1)	O(logn)
Retrieval	O(n)	O(1)	O(logn)
Deletion	O(n)	O(1)	O(logn)

	List	Hashmap	BST
Space complexity	O(n)	O(n)	O(n)

The time required to complete the 3 operations on the corpus containing around 8000 terms.

	List	Hashmap	BST
Insertion	2.932521s	0.020566s	0.087725s
Retrieval	2.870371s	0.012416s	0.013058s
Deletion	1.339360s	0.009608s	0.051292s

Hashmap is the **most optimal data structure** for the construction of the inverted index because of the superior time complexity.

Query

The queries are performed using hashmap data structure for storing index terms and the corpus is processed using the Porter stemmer.

term1 AND term2 AND term3

Time complexity -
Considering M as the number of index terms and
N as the maximum document frequency of the 3 terms (or size of posting list)

Considering hashmap is used to store the index
The time complexity for retrieving the posting lists for the 3 terms - O(3*1) = O(1)

Since the documents are sorted, the intersection of 2 posting lists can be found in linear time. Hence,

The time complexity for the AND operation i.e intersection of the posting lists of term1 and term2 - O(N).

The time complexity for the AND operation i.e intersection of the posting lists of (term1 AND term2) and term3 - O(N).

Hence the total time complexity is O(1) + O(N) + O(N) = **O(N).**

Examples:
Query - "cricket AND stuart AND binny"

Results - "Stuart Binny announces retirement from first class and International cricket - The Indian Express
Stuart Binny India allrounder who owns the record of the best bowling figures ..."

Query - "juice AND good AND health"

Results - "Want to improve your eyesight? Try these delicious and healthy dips - The Indian Express
Caring for your eyes is of paramount importance Most people have been working from home since the pandemic began and …"

Query - "movie AND actor AND debut"

Results - "Entertainment News Roundup: Actor Ed Asner, star of 'Mary Tyler Moore,' 'Lou Grant' dies at age 91; 'Candyman' Towers Over Box Office With Impressive $22 Million Debut and more - Devdiscourse …"

term1 OR term2 AND NOT term3

Time complexity -
Considering N as the number of index terms and
M as the maximum document frequency of the 3 terms

Considering hashmap is used to store the index
The time complexity for retrieving the posting lists for the 3 terms - O(3*1) = O(1)

Since the posting list is sorted a single pass is enough to get the complement of the posting list of a term. Hence, Time complexity for NOT operation, i.e finding the complement of the posting list of term3 (NOT term3) - O(N)

Intersection and union operations can be performed in linear time if the lists are sorted. Hence,

Time complexity for the AND operation i.e intersection of the posting lists of term1 and term2 - O(N)

The time complexity for the OR operation i.e union of the posting list of (term1 AND term2) and (NOT term3) - O(N)

Hence the total time complexity is O(1) + O(N) + O(N) + O(N) = **O(N).**

Examples:
Query - "MP OR politics AND NOT money"

Results -
"Has the first-past-the-post system polarised Indian politics? - The Hindu
Politicising social divides and failings of the parliamentary system have led to this situation India’s parliamentary democracy is going through a phase of intense confrontation between the dominant ruling..."

Query - 'rs OR rupees AND NOT finance'

Results - "COVID-19 occupancies very less; prepared for any eventuality, says Narayana Health - CNBCTV18
Viren Shetty ED Group COO Narayana Health said becuase of COVID their ..."

Query - "fever OR cold AND NOT covid"

Results - "UP should ensure health facilities to control fever in Mathura: Priyanka - Business Standard
After 29 cases of miteborne rickettsiosis known as scrub typhus were detected in Mathura General Secretary ..."