SS ZG537 IR Assignment

Aravind Metku

2019HT12094

**Introduction**

IR system implementation using TFIDF mechanism for ranking. The system specializes in clinical trials, Specifically covid19 clinical trials. <https://clinicaltrials.gov/> is a very famous database for clinical trials. There are about 3095 clinical trials available at the time for writing this.

Since the clinical trials submitted are in specific XML format we can extract all the relavant tags. For this implementation I chose to capture following tags. There are different mechanics performed based on the type of the tag. Which gives more rich results.

XML file is always in format specified here <https://clinicaltrials.gov/ct2/html/images/info/public.xsd>

Tags selected for index construction

1. keyword
2. condition
3. brief\_title
4. official\_title
5. brief\_summary
6. detailed\_description
7. location\_countries

**Details**

**Prepossessing**

Each tag in XML is processed differently to capture as much information as possible. Every tag is normalized, and tokenized for only words. Following are the details of preprocessing steps are each tag.

Used ntlk lemmatization to stem the words but only for user input free text tags like title, description.

1. brief\_title
   1. Removes stop words
   2. Performs lemmatization
2. official\_title
   1. Removes stop words
   2. Performs lemmatization
3. brief\_summary
   1. Removes stop words
   2. Performs lemmatization
4. location\_countries/country
   1. *No pre-processing*
5. detailed\_description
   1. Removes stop words
   2. Performs lemmatization
6. Phase
   1. *No pre-processing*
7. intervention/description
   1. Removes stop words
   2. Performs lemmatization
8. intervention/intervention\_type
   1. *No pre-processing*
9. Condition
   1. *No pre-processing*
10. Keyword
    1. *No pre-processing*

**Inverted index**

Following is the format of the inverted index constructed

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Term 1 | Document count | Doc 1 + occurrence count of term in doc 1 | Doc 2 + occurrence count of term in doc 2 | Doc 3 + occurrence count of term in doc 3 |
| Term 2 | Document count | Doc 3 + occurrence count of term in doc 3 | Doc 4+ occurrence count of term in doc 4 |  |
| Term 3 | Document count | Doc 3 + occurrence count of term in doc 3 |  |  |

**Doc id to Doc name index**

Following the format of the index. **Doc id is calculated as the hash of the doc name.**

|  |  |
| --- | --- |
| Doc id 1 | Doc name 1 |
| Doc id 2 | Doc name 2 |
| Doc id 3 | Doc name 3 |

**TFIDF**

TFIDF score is calculated from above inverted and doc id to name index. Following the structure of the tfidf.

Here there are 5 documents with vocabulary of 2 {term1, term2}. *Note that the values are just for example*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Doc 1 | Doc 2 | Doc 3 | Doc 4 | Doc 5 |
| Term 1 | 1.34 | 0.0 | 0.5 | 3.4 | 0.2 |
| Term 2 | 0.34 | 1.0 | 1.5 | 3.3 | 0.5 |

**Examples**

Results are from the medium size data set submitted in the code. *The original data set contains 3095 files and complete index cache size is around 1.1gb*

**Print result format – [similarity score] [filename] – brief\_title**

1. Query – “ARDS”
   1. Result

Text

Description automatically generated

1. Query – “Phase 2 ARDS”
   1. Result

Text

Description automatically generated

1. Query – “remdesivir phase ARDS”

Text

Description automatically generated

1. Query – “lfjdslfjdsofi hdslfojdsakfl”

Text

Description automatically generated

1. Query – “Hydroxychloroquine adults”

A close up of some grass

Description automatically generated

1. Query – “Hydroxychloroquine India”

Text

Description automatically generated