Information Retrieval - Search Engines
Prior-Art Search Assignment
Part I

Simon Platiotis, Undergraduate Student, IHU

***Abstract*—This document is the first of two parts of the mandatory assignment on 'Information Retrieval – Search Engines' course of IEE department at IHU. It's goal is the familiarization of the student with the process of index creation.**

***Keywords*—information retrieval, search engines, terrier, trec, index, text preprocessing, stemming, stopwords**

I. Introduction

For the sake of this assignment we were requested to generate five different indices based on a collection of approximately 50.000 TREC formated patent documents [1] using a search engine of our choice. The difference between each indixe would be derived from our choice of text preprocessing(stopwords, stemming, etc). For this specific case the search engine used is Terrier and the preprocessing stage consisted of different combinations of stemmers and the default stopwords list of Terrier.

II. Setup

A. Terrier Installation

The search engine of choice is Terrier which is open source. The installation can be completed easily by dowloading Terrier through the official webpage. The chosen version for the assingment is 5.2 mainly due to stability concerns [2]. Inside the directory of Terrier all executable bash scripts are located in bin. Executing trec_setup with target document collection for indexing will initialize the index, generate the properties files inside etc directory and is the mandatory first step.

B. Parameterization

Terrier Parameterization is achieved either through terrier.properties file inside etc directory or during the execution of a bash script. During the indexing process the important parameters are the **TrecDocTags**, specificly TrecDocTags.doctag, TrecDocTags.idtag, TrecDocTags.skip which specify the tags for the processing of the document and **termpipelines** which determines the processing stages each term goes through and the availables options are the following. Stopwords, using a stopwords list checks if a term should be skipped. PorterStemer which stems each term using the Porter Stemer Algorithm. TRv2PorterStemmer, is the an older implimentation of Porter Stemmer Algorithm on older versions of Terrier(<2). WeakPorterStemer, stems each term using a weak version of PorterStemer [3]. The strength of the stemmer is defined by the rate of over-stemmin and under-stemming. In general strong stemmers increase the recall of the results but with a cost in precision and weak stemmers increase the precision of the results while reducing the recall [4].

III. Indices

A. First Index, unchanged terms

The first index had no text preprocessing which resulted in an unmodified indexing of each term. To create an empty pipeline the stempipelines parameter must be empty. This is achieved through the terrier.properties file, by setting stempipelines equal to none. Since the stempipeline is set and the TrecDocTags are already correctly specified because the document collection is using default tag naming, the index can be created. The terrier script is located inside the bin directory. Executing the terrier script with the option **bi**, sorthand for batchindexing. During this step the terrier script allows extra parameterization by setting a system property using the -D option. -Dstempipelines=, overwrites the stempipelines value set in terrier.properties. Upon completion the index data ara located in directory var of terrier.

B. Second Index, stopwords only

The second index's text preprocessing filters out stopwords using a stop word list(the default list is stopword-list.txt located in terrier's share directory). This is achieved through the terrier.properties file by setting stempipelines equal to stopwords. Since the stempipeline is set and the TrecDocTags are already correctly specified because the document collection is using default tag naming, the index can be created. The terrier script is located inside the bin directory. Executing the terrier script with the option **bi**, sorthand for batchindexing. During this step the terrier script allows extra parameterization by setting a system property using the -D option. -Dstempipelines=stopwords, overwrites the stempipelines value set in terrier.properties. Upon completion the index data ara located in directory var of terrier.

C. Third Index, Weak Porter Stemmer

The third index's text preprocessing filters out stopwords using a stop word list(the default list is stopword-list.txt located in terrier's share directory) and stems the filtered terms using a weak version of the Porter Stemmer Algorithm. This is achieved through the terrier.properties file by setting stempipelines equal to stopwords and weak porter stemmer. Since the stempipeline is set and the TrecDocTags are already correctly specified because the document collection is using default tag naming, the index can be created. The terrier script is located inside the bin directory. Executing the terrier script with the option **bi**, sorthand for batchindexing. During this step the terrier script allows extra parameterization by setting a system property using the -D option. -Dstempipelines=stopwords,WeakPorterStemmer, overwrites the stempipelines value set in terrier.properties. Upon completion the index data ara located in directory var of terrier.

D. Fourth Index, TRv2 Porter Stemmer

The fouth index's text preprocessing filters out stopwords using a stop word list(the default list is stopword-list.txt located in terrier's share directory) and stems the filtered terms using the original implementation of the Porter Stemmer Algorithm in Terrier(The original Porter Stemmer was updated to the latest version on Terrier v2.0). This is achieved through the terrier.properties file by setting stempipelines equal to stopwords and weak porter stemmer. Since the stempipeline is set and the TrecDocTags are already correctly specified because the document collection is using default tag naming, the index can be created. The terrier script is located inside the bin directory. Executing the terrier script with the option **bi**, sorthand for batchindexing. During this step the terrier script allows extra parameterization by setting a system property using the -D option. -Dstempipelines=stopwords,TRv2PorterStemmer, overwrites the stempipelines value set in terrier.properties. Upon completion the index data ara located in directory var of terrier.

E. Fifth Index, Porter Stemmer

The Fifth index's text preprocessing filters out stopwords using a stop word list(the default list is stopword-list.txt located in terrier's share directory) and stems the filtered terms using the latest version of Porter Stemmer Algorithm. This is achieved through the terrier.properties file by setting stempipelines equal to stopwords and weak porter stemmer. Since the stempipeline is set and the TrecDocTags are already correctly specified because the document collection is using default tag naming, the index can be created. The terrier script is located inside the bin directory. Executing the terrier script with the option **bi**, sorthand for batchindexing. During this step the terrier script allows extra parameterization by setting a system property using the -D option. -Dstempipelines=stopwords,PorterStemmer, overwrites the stempipelines value set in terrier.properties. Upon completion the index data ara located in directory var of terrier.

IV. Conclusion

Summing up, Terrier index parameterization depends on the TercDocTag for managing the Terc formatted document collection and TermPipeline as far as preprocessing is concerned. Bellow is a table with condensed information about the indices.

TABLE I.

Index	***Description***	***Pipeline***
1	Terms unchanged.	Null
2	Filter out stopwords using default stopword-list.txt	Stopwords Only
3	Check if the stem is not a stopword and stem it using the weak Porter Stemmer	Weak Stemmer
4	Check if the stem is not a stopword and stem it using the dated Porter Stemmer	Dated Stemmer
5	Check if the stem is not a stopword and stem it using the latest Porter Stemmer	Latest Stemmer

Fig. 1. Indices-Pipelines Table

V. References

[1] M. Salabasis "Patent Document Collection".

[2] S. Vasileios, "*Terrier Presentation* to undergraduate students of IHU." , November 2020.

[3] Terrier Documentation, "Terrier.properties.sample".

[4] Cristian Moral, Angélica de Antonio, Ricardo Imbert and Jaime Ramírez, 'A survey of stemming algorithms in information retrieval', Escuela Técnica Superior de Ingenieros Informáticos, Universidad Politécnica de Madrid, Spain, March 2014.

[5] L. Mounia "Introduction to Information Retrieval,", Yahoo! Research, June 2011.