

BBC Sport Document Search Engine

DS8003

Rafik Matta

Contents

Summary	3
Technology Choices + Performance Tuning.....	3
Architecture	5
Code Explanation	6
Build Index	6
Load Index + Query	6
Results.....	7
Appendix A - Code.....	9
Initialize	9
Build Index Function	9
Load Index + Query Functions.....	10
Call the Functions.....	11
Appendix B – Screenshots of System Running in PySpark Console	13
Load Data to HDFS and Start PySpark Console	13
Load Libraries and set shuffle partitions.....	13
Define Build Index Function	14
Define Load Index + Query Function.....	14
Run Build Rugby Index	15
Run Load Index + Dummy Query to Action the Load.....	15
Run the Queries	16
Query1,N=1	16
Query1,N=3	16
Query1,N=5	16
Query2,N=1	17
Query2,N=3	17
Query2,N=5	17
Appendix C – Excerpts of the top Documents	18
Query 1.....	18

Rugby/098.txt	18
Rugby/127.txt	18
Rugby/086.txt	19
Query 2.....	19
Rugby/001.txt	19
Rugby/003.txt	20
Rugby/134.txt	20

Summary

This search system has been developed by building an index around the provided BBC Sport document data. The system was developed using Hortonworks HDP with HDFS as a persistence layer, Spark as the processing layer, and Livy + Jupyter as the interface layer. The system allows a user to choose which of the directories available in the dataset to index and then using Livy+Jupyter allows them to query and review results. The system performs quite well at searching once the data has been loaded. Queries on average take under a second.

Technology Choices + Performance Tuning

Out of the possible technology choices available for this system, PySpark (2.3) on HDP 2.6.5 was the primary technology choice. Spark was chosen for its ease of use, and built in optimizations (especially for querying). MongoDB would have also served well for this data, given its unstructured nature as well as relatively small size. That being said, the main drawback to using Mongo was the overly complex aggregation pipeline mechanism required to actually do anything useful. Spark offers a pretty easy to use SQL like interface over Mongo's JSON based structure, which due to the Catalyst engine is already optimized.

Very few performance tuning tricks needed to be done to achieve almost instant querying as DataFrames in Spark already do much of the query optimization. The slowest operation is loading the index. Therefore, the first query takes a performance penalty due to the lazy load nature of Spark, but every subsequent query thereafter is very rapid.

Here are the specifications of the machine/cluster used to perform the analysis:

- Windows 10 64-bit
- Hortonworks HDP 2.6.5 with Spark v 2.3
- 16 GB of DDR3 RAM
- 4-Core Intel Core i7 (2011 Model) – Hyperthreading Enabled

With this in mind, the following choices were made:

1. Run Spark in Local mode (as it's running on a single laptop), with 8 cores assigned to driver (as it's technically a 4-core multithreaded machine making it virtually 8 cores)
2. Set partition size to 8 (equal to the number of cores) to eliminate shuffles of data (most expensive Spark operation).
3. Make extensive use of DataFrames to leverage Catalyst optimizations.
4. In the Query logic, ensure the query vector was loaded using a Broadcast variable in Spark which ensures a balanced join.

Further to this, provided in Table 1 is a list of the technology choices available for the project and developing the system as well as their pro's and con's.

Technology	Used	Pros	Cons
MapReduce	No	MapReduce is excellent for low level Map/Reduce operations on a Hadoop Cluster. It allows effectively full control over the process and data flow	MapReduce doesn't offer out of the box optimization, which means unless one is well trained in using it, one would be better off using something like Spark.
Hive	No	Hive offers an excellent SQL like interface and with the recent updates made to it, it offers ACID transaction guarantees. With HiveServer2 Interactive, performance is significantly better as well (using Tez).	The major issue with Hive is that there's no really solid interface to interact with it as opposed to something like Spark + Livy. It also still performs relatively worse to Spark.
Pig	No	Pig serves as a step up from MapReduce, and offers some minor out of the box optimizations.	Pig still does not do well with offering really well optimized operations as compared to Spark. Further, the interface to use it and the general syntax leaves much to be desired.
HBase	No	HBase is great for column-based operations. This actually would have made a lot of sense for the project.	Given the optimization capabilities of Spark and the generally marginal performance difference achievable, it was not worth going through the learning curve of HBase.
Mongo	No	As MongoDB is an object store, it would actually have been another excellent fit for this problem. As we had to build our own Index logic, Mongo would have done nothing here except serving as a persistence layer. Also Mongo doesn't require a Hadoop Cluster.	The major con to using Mongo here would be that it's aggregations aren't particularly intuitive and having to use an aggregation pipeline wasn't necessary here.
HDFS	Yes	Distributed File System plus natively supported by Spark	Requires a Hadoop Cluster to be setup (used the HDP Sandbox)
Spark	Yes	Powerful and easy to use distributed processing engine with built in optimizations and a data structure called DataFrame for doing SQL like querying operations	Requires a Hadoop Cluster to be setup (used the HDP Sandbox)

Table 1. Technology Comparison Table

Architecture

The below diagrams show the system design + dataflow.

Legend:

- – Spark Step/Process
- – HDFS Step/Process
- – Livy Step/Process

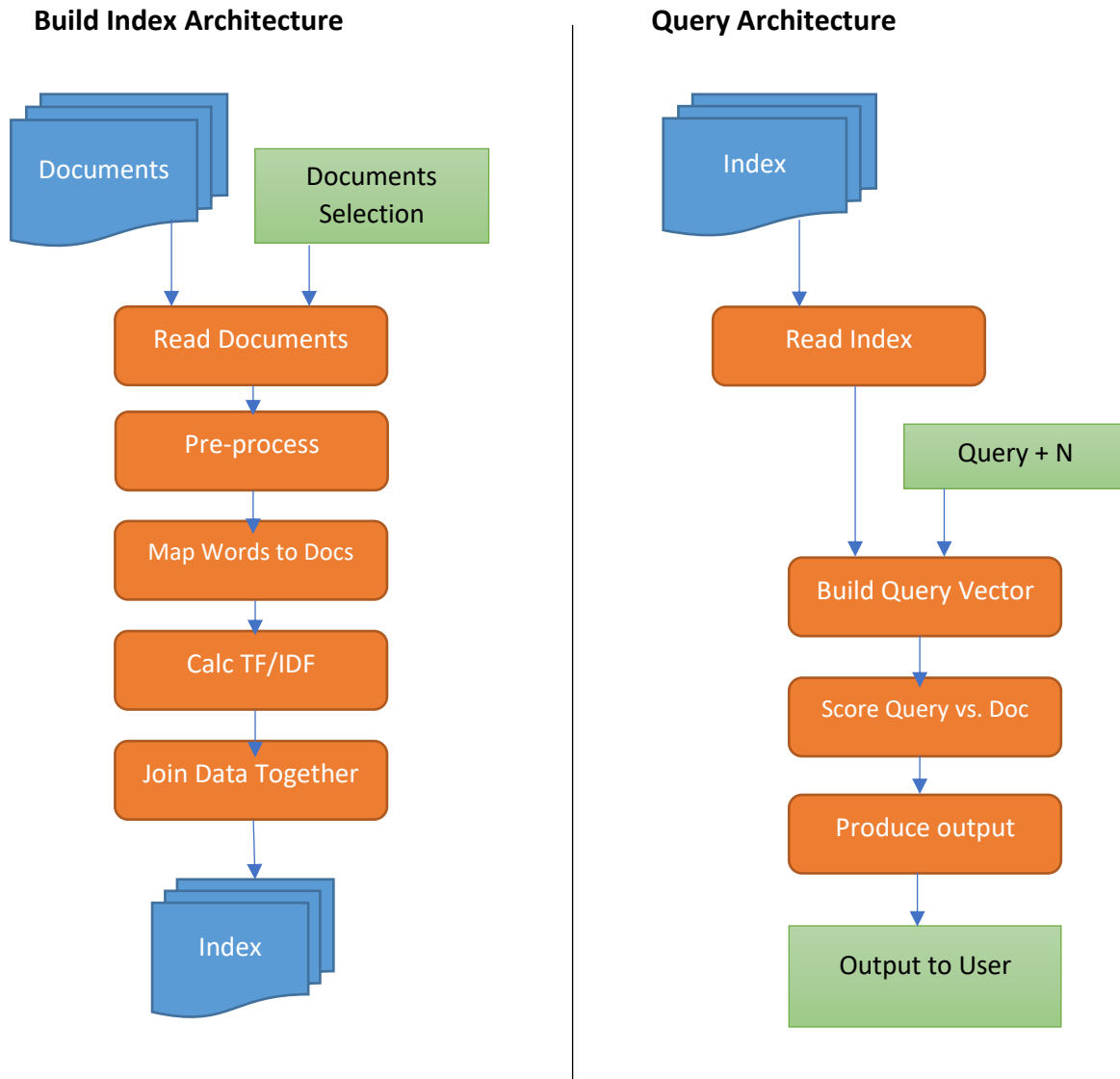


Figure 1. System Architecture

Code Explanation

This explanation coincides with the System Architecture Diagram in Figure 1, as well as the code in Appendix A. All line numbers can be found in the corresponding section in Appendix A.

Build Index

1. Set the partitions to the appropriate amount for the cluster setup. (Line 9)
2. Load the text files belonging to a specific sport into Spark as an RDD using the wholeTextFiles function and count the total number of documents. (Line 12-13)
3. Pre-process the text. This includes removing punctuation, end-of-line characters, lower casing all the words to produce one large string per document. The end result is an RDD mapping where the key is the document name/reference and the values are the pre-processed document content. (Line 16-20)
4. Flatten the resulting string into words by separating based on whitespaces and convert to a DataFrame. From here, we stop using RDD's and begin leveraging the power of DataFrames. (Line 22- 23)
5. Calculate the Term Frequency and the Inverse Document Frequency based on the following formulas (Line 25-34):

$$TF = \frac{t}{d}$$

where t = # of times a term occurs, and d = total words in a document

$$IDF = \log\left(\frac{N}{DF_t}\right)$$

where N = total documents, DF_t = # of times a terms occurs in corpus

6. Calculate TF-IDF by joining the resulting TF DataFrame and IDF Dataframe and using the follow formula (Line 36):

$$TFIDF = TF * IDF$$

7. Write resulting TF-IDF index to HDFS using the ORC format. This format was chosen due to its excellent compression characteristics (Line 38).

Load Index + Query

1. Load the Index that will be used for searching (Line 41-46, 73)
2. Select Query + result amount (Line 48, 81-82).
3. Take the query string and split it into words (using whitespace as a delimiter). (Line 50)
4. Calculate the TF and the IDF of the Query String and calculate the TF-IDF using the same logic and formulas given previously. (Line 52-58)
5. Compare the resulting Query vector against all of the documents and produce the score using the following formula (Line 60):

$$Score(Q, Doc) = \frac{|Q \cap Doc|}{|Q|} \sum_{q \in Q} TFIDF(q, Doc)$$

6. Output the result (Line 61).

Results

The following analysis was run on the system:

Query1 = "England claim Dubai Sevens glory"

Query2 = "Yachvili slotted over four penalties"

Index Used = "rugby"

Query1, N = 1

file	score
rugby/098.txt	0.12769736588535455

Query1, N = 3

file	score
rugby/098.txt	0.12769736588535455
rugby/127.txt	0.06749421879927163
rugby/086.txt	0.05140068228743627

Query1, N = 5

file	score
rugby/098.txt	0.12769736588535455
rugby/127.txt	0.06749421879927163
rugby/086.txt	0.05140068228743627
rugby/092.txt	0.012255797343391425
rugby/060.txt	0.007656956682990293

Query2, N = 1

file	score
rugby/001.txt	0.043268433667157026

Query2, N = 3

file	score
rugby/001.txt	0.043268433667157026
rugby/003.txt	0.02242535737241099
rugby/134.txt	0.020408392192625024

Query2, N = 5

file	score
rugby/001.txt	0.043268433667157026
rugby/003.txt	0.02242535737241099
rugby/134.txt	0.020408392192625024
rugby/141.txt	0.01865745104894276
rugby/097.txt	0.016309428405249717

Appendix A - Code

Initialize

```
01 from pyspark.sql.functions import *
02 from pyspark.conf import SparkConf
03 from pyspark.context import SparkContext
04 from pyspark.sql.types import *
05 from pyspark.sql.functions import *
06 import re, datetime
07
08 #set the correct number of partitions
09 sqlContext.setConf("spark.sql.shuffle.partitions", u"8")
```

Build Index Function

```
11 def build_index(doc_folder):
12     text_files = sc.wholeTextFiles("/user/root/bbcsport/" + doc_folder)
13     file_count = text_files.count()
14     # pre-process stage
15     # 1. remove prefix from file names
16     hdfs_precursor = "hdfs://sandbox-
hdp.hortonworks.com:8020/user/root/bbcsport/"
17     files = text_files.map(lambda file: (file[0].replace(hdfs_precursor,
""), file[1]))
18     # 2. clean text, make lower case,
19     lines = files.map(lambda lines: (lines[0], re.sub('\n+', '\n',
lines[1]).replace('\n', ' ')))
20     lines = lines.map(lambda line: (line[0], re.sub('[^\w\s-]', '',
line[1].lower().strip())))
21     # Map words to docs
22     words = lines.flatMapValues(lambda word: word.split(" "))
23     words_df = words.toDF(["file", "word"])
24
25     # calculate TF/IDF
26     # 1. counts words per doc
27     file_words_count =
words_df.groupBy("file").agg(count("word").alias("word_count"))
28     # 2. count words overall
29     words_count = words_df.groupBy("word", "file").count()
30     # 3.calculate tf
31     tf = words_count.join(file_words_count, file_words_count.file ==
words_count.file, 'left').withColumn("tf", col("count") /
col("word_count")).select(words_count.word, words_count.file, "tf")
32     # 4.calculate IDF
33     doc_freq =
words_df.groupBy("word").agg(countDistinct("file").alias("df"))
34     idf = doc_freq.groupBy("word", "df").agg(log(file_count /
column("df")).alias("idf"))
35     # join data and calculate tf/idf
36     tfidf = tf.join(idf, tf.word == idf.word, 'left').withColumn("tf_idf",
col("tf") * col("idf")).select(tf.word,tf.file,idf.idf,"tf_idf")
37     # write data to file
38     tfidf.write.orc(doc_folder + '.orc')
39     return
```

Load Index + Query Functions

```
41 def load_index(index_name):
42     #read the index file
43     tfidf = spark.read.orc(index_name + '.orc')
44     #cache it for use amongst all the queries so it's not reloaded but
kept for the whole session
45     tfidf.persist()
46     return(tfidf)
47
48 def query_call(query, N,index):
49     #split the query string into words
50     query_words = query.lower().split(" ")
51     #count the total number of words
52     total_words = len(query_words)
53     #create the dataframe for the query vec
54     query_df = spark.createDataFrame(query_words, StringType())
55     #count the number of times a term occurs in the query string
56     query_df =
query_df.groupBy("value").count().select(col("value").alias("word"),
col("count").alias("tf"))
57     #calculate the IDF of the query and join with the Index to calculate
scores
58     query_idf = query_df.join(broadcast(index), index.word ==
query_df.word, 'left').select(index.file, query_df.word,
query_df.tf,index.idf, index.tf_idf)
59     #calculate score and order by highest to lowest
60     results = query_idf.groupBy("file").agg((sum("tf_idf") *
(count("word") / total_words)).alias("score")).orderBy(desc("score"))
61     results.show(N)
62     return
```

Call the Functions

```
64 #example usage
65 #build index for rugby
66 a = datetime.datetime.now()
67 build_index("rugby")
68 b = datetime.datetime.now()
69 print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints
time taken
70
71 #load rugby index
72 a = datetime.datetime.now()
73 index = load_index("rugby")
74 b = datetime.datetime.now()
75 print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints
time taken
76
77 ##### Run Initial Dummy Query to Actually Load Index #####
78 a = datetime.datetime.now()
79 N = 1 # top N results
80 query = "test"
81 query_call(query, N,index)
82 b = datetime.datetime.now()
83 print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints
time taken
84
85 ##### This is the code that should be repeated. Once the initial load is
done #####
86 ##### The query should run in the same session multiple times #####
87 a = datetime.datetime.now()
88 N = 1 # top N results
89 query = "England claim Dubai Sevens glory"
90 query_call(query, N,index)
91 b = datetime.datetime.now()
92 print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints
time taken
93
94 a = datetime.datetime.now()
95 N = 3 # top N results
96 query = "England claim Dubai Sevens glory"
97 query_call(query, N,index)
98 b = datetime.datetime.now()
99 print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints
time taken
100
101 a = datetime.datetime.now()
102 N = 5 # top N results
103 query = "England claim Dubai Sevens glory"
104 query_call(query, N,index)
105 b = datetime.datetime.now()
106 print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints
time taken
107
108 a = datetime.datetime.now()
109 N = 1 # top N results
110 query = "Yachvili slotted over four penalties"
111 query_call(query, N,index)
112 b = datetime.datetime.now()
```

```

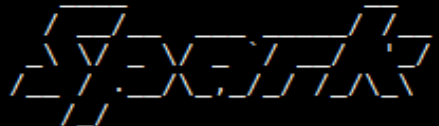
113 print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints
time taken
114
115 a = datetime.datetime.now()
116 N = 3 # top N results
117 query = "Yachvili slotted over four penalties"
118 query_call(query, N,index)
119 b = datetime.datetime.now()
120 print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints
time taken
121
122 a = datetime.datetime.now()
123 N = 5 # top N results
124 query = "Yachvili slotted over four penalties"
125 query_call(query, N,index)
126 b = datetime.datetime.now()
127 print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints
time taken

```

Appendix B – Screenshots of System Running in PySpark Console

Load Data to HDFS and Start PySpark Console

```
[root@sandbox-hdp ~]# hadoop fs -put bbc sport/
[root@sandbox-hdp ~]# pyspark --master local[8]
SPARK_MAJOR_VERSION is set to 2, using Spark2
Python 2.7.5 (default, Jul 13 2018, 13:06:57)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-28)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
Welcome to
```

The PySpark logo is a stylized representation of the word 'PySpark' using a grid of lines to form the letters. It is displayed in a light blue color on a black background.

version 2.3.0.2.6.5.0-292

```
Using Python version 2.7.5 (default, Jul 13 2018 13:06:57)
SparkSession available as 'spark'.
>>> █
```

Load Libraries and set shuffle partitions

```
>>> from pyspark.sql.functions import *
from pyspark.conf import SparkConf
from pyspark.context import SparkContext
from pyspark.sql.types import *
from pyspark.sql.functions import *
import re, datetime

#set the correct number of partitions
sqlContext.setConf("spark.sql.shuffle.partitions", u"8")>>> from pyspark.conf import SparkConf
>>> from pyspark.context import SparkContext
>>> from pyspark.sql.types import *
>>> from pyspark.sql.functions import *
>>> import re, datetime
>>>
>>> #set the correct number of partitions
... sqlContext.setConf("spark.sql.shuffle.partitions", u"8") █
```

Define Build Index Function

```
>>> def build_index(doc_folder):
...     text_files = sc.wholeTextFiles("/user/root/bbcsport/" + doc_folder)
...     file_count = text_files.count()
...     # pre-process stage
...     # 1. remove prefix from file names
...     hdfs_precursor = "hdfs://sandbox-hdp.hortonworks.com:8020/user/root/bbcsport/"
...     files = text_files.map(lambda file: (file[0].replace(hdfs_precursor, ""), file[1]))
...     # 2. clean text, make lower case,
...     lines = files.map(lambda lines: (lines[0], re.sub('\n+', '\n', lines[1]).replace('\n', ' ')))
...     lines = lines.map(lambda line: (line[0], re.sub('[^\w\s-]', '', line[1].lower().strip())))
...     # Map words to docs
...     words = lines.flatMapValues(lambda word: word.split(" "))
...     words_df = words.toDF(["file", "word"])
...
...     # calculate TF/IDF
...     # 1. counts words per doc
...     file_words_count = words_df.groupBy("file").agg(count("word").alias("word_count"))
...     # 2. count words overall
...     words_count = words_df.groupBy("word", "file").count()
...     # 3. calculate tf
...     tf = words_count.join(file_words_count, file_words_count.file == words_count.file, 'left').withColumn("tf", col("count") / col("word_count")).select(words_count.word, words_count.file, "tf")
...     # 4. calculate IDF
...     doc_freq = words_df.groupBy("word").agg(countDistinct("file").alias("df"))
...     idf = doc_freq.groupBy("word", "df").agg(log(file_count / column("df")).alias("idf"))
...     # join data and calculate tf/idf
...     tfidf = tf.join(idf, tf.word == idf.word, 'left').withColumn("tf_idf", col("tf") * col("idf")).select(tf.word, tf.file, idf.idf, "tf_idf")
...     # write data to file
...     tfidf.write.orc(doc_folder + ".orc")
...     return
...
>>> |
```

Define Load Index + Query Function

```
>>> def load_index(index_name):
...     #read the index file
...     tfidf = spark.read.orc(index_name + ".orc")
...     #cache it for use amongst all the queries so it's not reloaded but kept for the whole session
...     tfidf.persist()
...     return(tfidf)
...
>>> def query_call(query, N, index):
...     #split the query string into words
...     query_words = query.lower().split(" ")
...     #count the total number of words
...     total_words = len(query_words)
...     #create the dataframe for the query vec
...     query_df = spark.createDataFrame(query_words, StringType())
...     #count the number of times a term occurs in the query string
...     query_df = query_df.groupBy("value").count().select(col("value").alias("word"), col("count").alias("tf"))
...     #calculate the IDF of the query and join with the Index to calculate scores
...     query_idf = query_df.join(broadcast(index), index.word == query_df.word, 'left').select(index.file, query_df.word, query_df.tf, index.idf, index.tf_idf)
...     #calculate score and order by highest to lowest
...     results = query_idf.groupBy("file").agg((sum("tf_idf") * (count("word") / total_words)).alias("score")).orderBy(desc("score"))
...     results.show(N)
...     return
```

Run Build Rugby Index

```
>>> #example usage
... #build index for rugby
... a = datetime.datetime.now()
>>> build_index("rugby")
18/12/04 03:28:04 WARN Client: interrupted waiting to send rpc request to server
java.lang.InterruptedException
    at java.util.concurrent.FutureTask.awaitDone(FutureTask.java:404)
    at java.util.concurrent.FutureTask.get(FutureTask.java:191)
    at org.apache.hadoop.ipc.Client$Connection.sendRpcRequest(Client.java:1094)
    at org.apache.hadoop.ipc.Client.call(Client.java:1457)
    at org.apache.hadoop.ipc.Client.call(Client.java:1398)
    at org.apache.hadoop.ipc.ProtobufRpcEngine$Invoker.invoke(ProtobufRpcEngine.java:233)
    at com.sun.proxy.$Proxy13.getBlockLocations(Unknown Source)
    at org.apache.hadoop.hdfs.protocolPB.ClientNameNodeProtocolTranslatorPB.getBlockLocations(ClientNameNodeProtocolTranslatorPB.java:272)
    at sun.reflect.GeneratedMethodAccessor8.invoke(Unknown Source)
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    at java.lang.reflect.Method.invoke(Method.java:498)
    at org.apache.hadoop.io.retry.RetryInvocationHandler.invokeMethod(RetryInvocationHandler.java:290)
    at org.apache.hadoop.io.retry.RetryInvocationHandler.invoke(RetryInvocationHandler.java:202)
    at org.apache.hadoop.io.retry.RetryInvocationHandler.invoke(RetryInvocationHandler.java:184)
    at com.sun.proxy.$Proxy14.getBlockLocations(Unknown Source)
    at org.apache.hadoop.hdfs.DFSClient.callGetBlockLocations(DFSClient.java:1249)
    at org.apache.hadoop.hdfs.DFSClient.getLocatedBlocks(DFSClient.java:1236)
    at org.apache.hadoop.hdfs.DFSClient.getLocatedBlocks(DFSClient.java:1224)
    at org.apache.hadoop.hdfs.DFSInputStream.fetchLocatedBlocksAndGetLastBlockLength(DFSInputStream.java:309)
    at org.apache.hadoop.hdfs.DFSInputStream.openInfo(DFSInputStream.java:274)
    at org.apache.hadoop.hdfs.DFSInputStream.<init>(DFSInputStream.java:266)
    at org.apache.hadoop.hdfs.DFSClient.open(DFSClient.java:1549)
    at org.apache.hadoop.hdfs.DistributedFileSystem$4.doCall(DistributedFileSystem.java:332)
    at org.apache.hadoop.hdfs.DistributedFileSystem$4.doCall(DistributedFileSystem.java:327)
    at org.apache.hadoop.fs.FileSystemLinkResolver.resolve(FileSystemLinkResolver.java:81)
    at org.apache.hadoop.hdfs.DistributedFileSystem.open(DistributedFileSystem.java:340)
    at org.apache.hadoop.fs.FileSystem.open(FileSystem.java:787)
    at org.apache.spark.input.WholeTextFileRecordReader.nextKeyValue(WholeTextFileRecordReader.scala:75)
    at org.apache.hadoop.mapreduce.lib.input.CombineFileRecordReader.nextKeyValue(CombineFileRecordReader.java:65)
    at org.apache.spark.rdd.NewHadoopRDD$anon$1.hasNext(NewHadoopRDD.scala:214)
    at org.apache.spark.InterruptibleIterator.hasNext(InterruptibleIterator.scala:37)
    at scala.collection.Iterators$anon$11.hasNext(Iterator.scala:408)
    at scala.collection.Iterators$class.foreach(Iterator.scala:893)
    at scala.collection.AbstractIterator.foreach(Iterator.scala:1336)
    at org.apache.spark.api.python.PythonRDD$.writeIteratorToStream(PythonRDD.scala:204)
    at org.apache.spark.api.python.PythonRunners$anon$2.writeIteratorToStream(PythonRunner.scala:407)
    at org.apache.spark.api.python.BasePythonRunner$WriterThread$$anonfun$run$1.apply(PythonRunner.scala:215)
    at org.apache.spark.util.Utils$.logUncaughtExceptions(Utils.scala:1988)
    at org.apache.spark.api.python.BasePythonRunner$WriterThread.run(PythonRunner.scala:170)
18/12/04 03:28:06 WARN Column: Constructing trivially true equals predicate, 'file#0 = file#0'. Perhaps you need to use aliases.
18/12/04 03:28:06 WARN Column: Constructing trivially true equals predicate, 'word#1 = word#1'. Perhaps you need to use aliases.
>>> b = datetime.datetime.now()
>>> print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints time taken
Time taken: 13.703067 seconds
>>>
```

Run Load Index + Dummy Query to Action the Load

```
>>> #load rugby index
... a = datetime.datetime.now()
>>> index = load_index("rugby")
>>> b = datetime.datetime.now()
>>> print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints time taken
Time taken: 0.353459 seconds
>>>
```

```
>>> ##### Run Initial Dummy Query to Actually Load Index #####
... a = datetime.datetime.now()
>>> N = 1 # top N results
>>> query = "test"
>>> query_call(query, N, index)
+-----+-----+
|      file|      score|
+-----+-----+
|rugby/062.txt|0.010909395240812712|
+-----+-----+
only showing top 1 row

>>> b = datetime.datetime.now()
>>> print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints time taken
Time taken: 1.662721 seconds
>>>
```

Run the Queries

Query1,N=1

```
>>> ##### This is the code that should be repeated. Once the initial load is done #####
... ##### The query should run in the same session multiple times #####
... a = datetime.datetime.now()
>>> N = 1 # top N results
>>> query = "England claim Dubai Sevens glory"
>>> query_call(query, N,index)
+-----+
|      file|      score|
+-----+
|rugby/098.txt|0.12769736588535455|
+-----+
only showing top 1 row

>>> b = datetime.datetime.now()
>>> print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints time taken
Time taken: 0.656322 seconds
```

Query1,N=3

```
>>> a = datetime.datetime.now()
>>> N = 3 # top N results
>>> query = "England claim Dubai Sevens glory"
>>> query_call(query, N,index)
+-----+
|      file|      score|
+-----+
|rugby/098.txt|0.12769736588535455|
|rugby/127.txt|0.06749421879927163|
|rugby/086.txt|0.05140068228743627|
+-----+
only showing top 3 rows

>>> b = datetime.datetime.now()
>>> print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints time taken
Time taken: 0.515966 seconds
```

Query1,N=5

```
Time taken: 0.515966 seconds
>>> a = datetime.datetime.now()
>>> N = 5 # top N results
>>> query = "England claim Dubai Sevens glory"
>>> query_call(query, N,index)
+-----+
|      file|      score|
+-----+
|rugby/098.txt| 0.12769736588535455|
|rugby/127.txt| 0.06749421879927163|
|rugby/086.txt| 0.05140068228743627|
|rugby/092.txt|0.012255797343391425|
|rugby/060.txt|0.007656956682990293|
+-----+
only showing top 5 rows

>>> b = datetime.datetime.now()
>>> print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints time taken
Time taken: 0.525471 seconds
```


Query2,N=1

```
>>> a = datetime.datetime.now()
>>> N = 1 # top N results
>>> query = "Yachvili slotted over four penalties"
>>> query_call(query, N,index)
+-----+-----+
|      file|      score|
+-----+-----+
|rugby/001.txt|0.043268433667157026|
+-----+-----+
only showing top 1 row

>>> b = datetime.datetime.now()
>>> print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints time taken
Time taken: 0.570697 seconds
```

Query2,N=3

```
>>> a = datetime.datetime.now()
>>> N = 3 # top N results
>>> query = "Yachvili slotted over four penalties"
>>> query_call(query, N,index)
+-----+-----+
|      file|      score|
+-----+-----+
|rugby/001.txt|0.043268433667157026|
|rugby/003.txt| 0.02242535737241099|
|rugby/134.txt|0.020408392192625024|
+-----+-----+
only showing top 3 rows

>>> b = datetime.datetime.now()
>>> print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints time taken
Time taken: 0.50925 seconds
```

Query2,N=5

```
>>> a = datetime.datetime.now()
>>> N = 5 # top N results
>>> query = "Yachvili slotted over four penalties"
>>> query_call(query, N,index)
+-----+-----+
|      file|      score|
+-----+-----+
|rugby/001.txt|0.043268433667157026|
|rugby/003.txt| 0.02242535737241099|
|rugby/134.txt|0.020408392192625024|
|rugby/141.txt| 0.01865745104894276|
|rugby/097.txt|0.016309428405249717|
+-----+-----+
only showing top 5 rows

>>> b = datetime.datetime.now()
>>> print("Time taken: " + str((b-a).total_seconds()) + " seconds") #prints time taken
Time taken: 0.466992 seconds
```

Appendix C – Excerpts of the top Documents

Query 1

Rugby/098.txt

England claim Dubai Sevens glory

England beat Fiji 26-21 in a dramatic final in Dubai to win the first IRB Sevens event of the season.

Having beaten Australia and South Africa to reach the final, England fell behind to an early try against Fiji. They then took charge with scores from Pat Sanderson, Kai Horstman, Mathew Tait and Rob Thirlby, but Fiji rallied to force a tense finale. Scotland were beaten 33-15 by Samoa in the plate semi-final and Ireland lost 17-5 to Tunisia in the shield final. Mike Friday's England side matched their opponents for pace, power and skill in the final and led 19-7 at half-time. But Neumi Nanuku and Marika Vakacegu touched down for Fiji, only for a needless trip by Tuidriva Bainivalu on Geoff Appleford to allow England to run down the clock. "To be honest, England have wanted to win in Dubai for a very long time now, and the people here have wanted us to win for just as long," said Friday.

"We didn't want to put pressure on ourselves but we are thankful we have achieved that and brought through some young talent at the same time that can hopefully play for the England '15s' in a few years." Portugal confirmed their impressive progress in Sevens rugby by recording a sudden-death win over France in the bowl final. Samoa won the plate title by edging out Argentina 21-19.

Rugby/127.txt

England given tough Sevens draw

England will have to negotiate their way through a tough draw if they are to win the Rugby World Cup Sevens in Hong Kong next month.

The second seeds have been drawn against Samoa, France, Italy, Georgia and Chinese Taipei. The top two sides in each pool qualify but England could face 2001 winners New Zealand in the quarter-finals if they stumble against Samoa. Scotland and Ireland are in Pool A together with the All Blacks. England won the first event of the International Rugby Board World Sevens series in Dubai but have slipped to fourth in the table after failing to build on that victory.

However, they beat Samoa in the recent Los Angeles Sevens before losing to Argentina in the semi-finals. "England have the ability and determination to win this World Cup and create sporting history by being the only nation to hold both the 15s and Sevens World Cups at the same time," said England sevens coach Mike Friday. "England have a fantastic record in Hong Kong and have won there the last three years, but the World Cup is on a different level. "Every pool contains teams who have caused upsets before and we will have to work hard to ensure we progress from our group. "We have not performed consistently to our true potential so far in the IRB Sevens which has been disappointing - but we can only look forward." England won the first Rugby World Cup Sevens in 1993 with a side that included the likes of Lawrence Dallaglio and Matt Dawson. In 1997 and 2001, England lost in the quarter-finals.

(seeds in brackets)

New Zealand (1), Scotland (8), Tonga, Ireland, Korea, USA.

England (2), Samoa (7), France, Italy, Georgia, Chinese Taipei.

Fiji (3), Australia (6), Canada, Portugal, Japan, Hong Kong.

Argentina (4), South Africa (5), Kenya, Tunisia, Russia, Uruguay.

Rugby/086.txt

South Africa sweep top awards

South Africa's Schalk Burger was named player of the year as the Tri-Nations champions swept the top honours at the International Rugby Board's awards.

The flanker topped a list which included Ireland star Gordon D'Arcy and Australian sensation Matt Giteau. Jake White claimed the coaching award while his side held off Grand Slam winners France to take the team award. England player Simon Amor beat team-mate Ben Gollings and Argentine Lucio Lopez Fleming to win the sevens award. Burger's award came just a week after he won the equivalent prize from his fellow international players and White, who also coached Burger at under-21 level, paid tribute to him. "Schalk's emergence as a major force has meant a lot to South African rugby, but has also influenced world rugby," said White. "He's become to South African rugby what Jonty Rhodes was to South African cricket. It's amazing what he has achieved in such a short time so far in his international career." Amor, who will captain England in this season's opening IRB Sevens tournament, the Dubai Sevens, which start on Thursday, was delighted with his award. "There are so many great sevens players on the circuit at the moment that this is a genuine honour," said the Gloucester fly-half.

Query 2

Rugby/001.txt

Hodgson shoulders England blame

Fly-half Charlie Hodgson admitted his wayward kicking played a big part in England's 18-17 defeat to France.

Hodgson failed to convert three penalties and also missed a relatively easy drop goal attempt which would have given England a late win. "I'm very disappointed with the result and with myself," Hodgson said. "It is very hard to take but it's something I will have to get through and come back stronger. My training's been good but it just didn't happen." Hodgson revealed that Olly Barkley had taken three penalties because they were "out of my range" but the centre could not convert his opportunities either, particularly the drop goal late on. "It wasn't a good strike," he added. "I felt as soon as it hit my boot it had missed. It's very disappointing, but I must recover." Andy Robinson said he would "keep working on the kicking" with his squad. However, the England coach added that he would take some positives from the defeat.

"We went out to play and played some very good rugby and what have France done?" he said. "They won the game from kicking penalties from our 10m line. "It's very frustrating. The lads showed a lot of ambition in the first half, they went out to sustain it in the second but couldn't build on it. "We took the ball into contact, and you know when you do that it is a lottery whether the referee is going to give the penalty to your side or the other side. "We have lost a game we should have won. There is a fine line between winning and losing, and for the second week we've been on the wrong side of that line and it hurts."

England went in at half-time with a 17-6 lead but they failed to score in the second half and Dimitri Yachvili slotted over four penalties as France overhauled the deficit. England skipper Jason Robinson admitted his side failed to cope with France's improved second-half display.

"We controlled the game in the first half but we knew that they would come out and try everything after half-time," he said. "We made a lot of mistakes in the second half and they punished us. They took their chances when they came. "It's very disappointing. Last week we lost by two points, now one point."

Rugby/003.txt

Yachvili savours France comeback

France scrum-half Dimitri Yachvili praised his team after they fought back to beat England 18-17 in the Six Nations clash at Twickenham.

Yachvili kicked all of France's points as they staged a second-half revival. "We didn't play last week against Scotland and we didn't play in the first half against England," he said. "But we're very proud to beat England at Twickenham. We were just defending in the first half and we said we had to put them under pressure. We did well." Yachvili admitted erratic kicking from England's Charlie Hodgson and Olly Barkley, who missed six penalties and a drop goal chance between them, had been decisive. "I know what it's like with kicking. When you miss some it's very hard mentally, but it went well for us," he said. France captain Fabien Pelous insisted his side never doubted they could secure their first win against England at Twickenham since 1997. France were 17-6 down at half-time, but Pelous said: "No-one was down at half-time, we were still confident. "We said we only had 11 points against us, which was not much. "The plan was to keep hold of possession and pressure England to losing their composure." France coach Bernard Laporte accepted his side had not played well. "We know we have to play better to defend the title," he said. "I'm not happy we didn't score a try but we're happy because we won."

Rugby/134.txt

Scotland 18-10 Italy

Six Chris Paterson penalties gave Scotland victory in a dour but clinical encounter against Italy at Murrayfield.

Coach Matt Williams' side were outmuscled and outplayed in a tense first half but led 6-3 at the break. Paterson slotted four more second-half penalties and Scotland were denied a try when wing Sean Lamont's touchdown was ruled out for a forward pass. A late Andrea Masi try was small consolation for Italy, chasing their first away win in the Six Nations. Scotland came out on top of the early exchanges and took a quick 3-0 lead through the boot of full-back Paterson. But the more powerful Azzurri pack eventually rumbled into life.

A series of drives into Scottish territory set up a penalty attempt, missed by full-back Roland De Marigny, and a wayward drop-goal effort from Luciano Orquera. Scotland defended the initial thrusts but on 20 minutes Italy, after coming up yards short of the line, equalised through a De Marigny penalty. Italy were offered another penalty when Scottish flanker Simon Taylor was offside but the left-footed De Marigny pushed his kick wide. Scotland finally made the most of a rare foray into the Italian half and snatched three points from Paterson when an Italian forward handled the ball in a ruck.

As the half wore on, both sides squandered promising spells of momentum with sloppy penalties, and the period fizzled out with Scotland numerically, if not psychologically, on top. Italy's De Marigny narrowly missed a chance to level the scores again shortly after the break but his long-range kick shaved the right upright. And Scotland capitalised with a third Paterson penalty on 50 minutes. Williams' side seemed to have found a spark from somewhere and, after a couple of probing attacks, Paterson was able to slot another three points to widen the gap.

With the pendulum of possession swinging towards Scotland, Lamont thought he had wriggled over in the left corner after 65 minutes but play was recalled for a marginal forward pass from Paterson. Another Paterson penalty on 70 minutes kept the pressure on the wilting visitors. But John Kirwan's men had the last laugh when Gordon Ross' attempted clearance was charged down and Masi pounced for the try, converted by De Marigny.

: C Paterson; S Webster, A Craig, H Southwell, S Lamont; D Parks, C Cusiter; T Smith, G Bulloch (capt), G Kerr; S Grimes, S Murray; S Taylor, J Petrie, A Hogg.

R Russell, B Douglas, N Hines, J Dunbar, M Blair, G Ross, B Hinshelwood.

R de Marigny; Mirco Bergamasco, C Stoica, A Masi, L Nitoglia; L Orquera, A Troncon; A Lo Cicero, F Ongaro, M Castrogiovanni; S Dellape, M Bortolami (capt); A Persico, D Dal Maso, S Parisse.

G Intoppa, S Perugini, CA del Fava, S Orlando, P Griffen, R Pedrazzi, KP Robertson.