PROJECT DOCUMENTATION

Deepak Muddebihal

id: #23256716

Building Watson

2015

**Project Documentation**

**Introduction:**

The aim of this project is to build IBM’s model of Watson which was on the show “Jeopardy”. Here, given that a user asks a particular question, the most suitable answer should be given.

Eg: “Who played Sherlock Holmes in the film of the same name in 2009?”

Answer: Robert Downey Jr.

In the same way, a series of questions are asked, which are expecting suitable answers.

Here the questions are solely answered based on the content of the Wikipedia pages.

The task here is to index the Title of the Wikipedia pages along with its contents in such a way, that given a query, it should be compared with the contents of the Wikipedia pages and the corresponding Document Title which is most suitable with respect to answering that query should be returned.

A number of methodologies can be used to index the Wikipedia page such as stemming, lemmatization, positional indexing of terms etc.

In order to get the “best” results, based on the answers given along with the jeopardy questions, various scoring functions can be implemented such as the tf-idf model, Okapi BM25 etc. as will be shown.

**Indexing and retrieval:**

The basic technique which I used to index the Wikipedia documents is stemming.

I discarded the contents of the Wikipedia pages which was under “See also”, “External links”, “Further reading” and “References”, as they were not making any difference to the results I obtained (seen later).Along with the above I removed all the lines of the pages which were of the type ”==…..==”, as it was junk data which did not contribute in answering the question.

I took out all the special characters and punctuations which were occurring in the Wikipedia pages such as “[](),!-;:“ etc.

I partitioned in the set of questions into a Training data set and a Testing data set.

**Training data set:**

The first 20 questions of the question list were taken as the training data set. New techniques were first run on this data set.

**Testing data set:**

The remaining 80 questions of the question list were taken as the testing data set on which if the method performed well on the training data set it was then ran on the testing data set.

I took all the words from the clue as the query EXCLUDING the category section of the question.

**Measuring performance:**

Using this method of stemming, and the entire clue as the query, I used the metrics Precision at 1, Precision at 10 and Mean Reciprocal Rank to display the credibility of the results obtained.

Precision at 1 and 10 are used so as to give a good heuristic in terms of how good the method used is to obtain results and MRR is very useful when you care about the position of the top answer.

**Training data set results:**

|  |  |  |
| --- | --- | --- |
| Precision at 1 | Precision at 10 | Mean Reciprocal Rank |
| 3/20=0.15 | 6/20=0.3 | 3.83/20=0.1916 |

Run with :

java ".;jar1.jar;jar2.jar;jar3.jar;jar4.jar" Project wiki 1 0 0 training\_set.txt (Windows)

java ".:jar1.jar:jar2.jar:jar3.jar:jar4.jar" Project wiki 1 0 0 training\_set.txt (Linux)

When this method was run on the Testing data set, the results obtained were:

**Testing data set results:**

|  |  |  |
| --- | --- | --- |
| Precision at 1 | Precision at 10 | Mean Reciprocal Rank |
| 13/80=0.1625 | 41/80=0.5125 | 21.81/80=0.2726 |

Run with :

java ".;jar1.jar;jar2.jar;jar3.jar;jar4.jar" Project wiki 1 0 0 testing\_set.txt (Windows)

java ".:jar1.jar:jar2.jar:jar3.jar:jar4.jar"Project wiki 1 0 0 testing\_set.txt (Linux)

The Precision at 1, Precision at 10, and Mean Reciprocal Rank are shown above for the above system.

**Changing the scoring function:**

In reference to the same queries and documents above the BM25 model was executed.

The BM25 with default values k=1.2 & b=0.75, executed on the training set gave the following results:

Training data set values:

|  |  |  |
| --- | --- | --- |
| Precision at 1 | Precision at 10 | Mean Reciprocal Rank |
| 1/20=0.05 | 5/20=0.25 | 2.33/20=0.1165 |

Run with :

java ".;jar1.jar;jar2.jar;jar3.jar;jar4.jar" Project wiki 1 1 0 training\_set.txt 1.2 0.75 (Windows)

java ".:jar1.jar:jar2.jar:jar3.jar:jar4.jar" Project wiki 1 1 0 training\_set.txt 1.2 0.75

(Linux)

In order to obtain better results I tuned, the values of ‘k’ and ‘b’ and ran the new parameter values on the training data set

The best results I obtained were using the values of k=5.0 and b=0.0 which were:

Training data set values:

|  |  |  |
| --- | --- | --- |
| Precision at 1 | Precision at 10 | Mean Reciprocal Rank |
| 7/20=0.35 | 8/20=0.4 | 7.5/20=0.375 |

Run with :

java ".;jar1.jar;jar2.jar;jar3.jar;jar4.jar"Project wiki 1 1 0 training\_set.txt 5.0 0.0 (Windows)

java ".:jar1.jar:jar2.jar:jar3.jar:jar4.jar" Project wiki 1 1 0 training\_set.txt 5.0 0.0 (Linux)

This variance in results can be explained with respect to the formula given in class:

RSVd= ∑log(n/df).((k+1)tf/k((1-b)+bx(Ld/Lave))+tf)

Based on this formula, if the value of b=0 and k has a value > 0, the formula reduces to (k+1)tf/k+tf, thus if the value of k is very large with respect to term frequency, the answer will vary solely based on term frequency as that dominates, it is for this reason that this gives us very high results.

Now these newly tuned parameters of k and b are run on the testing set and the results obtained are as follows:

Testing data set results:

|  |  |  |
| --- | --- | --- |
| Precision at 1 | Precision at 10 | Mean Reciprocal Rank |
| 23/80=0.2875 | 51/80=0.6375 | 31.68/80=0.396 |

Run with :

java ".;jar1.jar;jar2.jar;jar3.jar;jar4.jar" Project wiki 1 1 0 testing\_set.txt 5.0 0.0 (Windows)

java ".:jar1.jar:jar2.jar:jar3.jar:jar4.jar" Project wiki 1 1 0 testing\_set.txt 5.0 0.0 (Linux)

As observed after the tuning of the parameters on the training set( 20 questions) improved the results, when the same parameters were run on the testing set the results obtained were much better than the results obtained with tf-idf model on the testing data set.

**Error analysis & Improving retrieval:**

By using my best method of stemming currently and using all the words in the query ONLY, I obtained

**Training data set results:**

|  |  |  |
| --- | --- | --- |
| Precision at 1 | Precision at 10 | Mean Reciprocal Rank |
| 3/20=0.15 | 6/20=0.3 | 3.83/20=0.1915 |

When this method was run on the Testing data set, the results obtained were:

**Testing data set results:**

|  |  |  |
| --- | --- | --- |
| Precision at 1 | Precision at 10 | Mean Reciprocal Rank |
| 13/80=0.1625 | 41/80=0.5125 | 21.81/80=0.2726 |

Observations:

After examining the results obtained and comparing it to the jeopardy questions a few observations are:

**1) Category missing in the query:**

The categories which are present in the jeopardy set were not included in the query which I used to run against the indexed documents.

**Method to improve retrieval**

After including the category in the query, this was the result I obtained on the training data set (20 questions)

Training data set results:

|  |  |  |
| --- | --- | --- |
| Precision at 1 | Precision at 10 | Mean Reciprocal Rank |
| 3/20=0.15 | 9/20=0.45 | 4.48/20=0.2242 |

Run with :

java ".;jar1.jar;jar2.jar;jar3.jar;jar4.jar" Project wiki 1 0 1 training\_set.txt (Windows)

java ".:jar1.jar:jar2.jar:jar3.jar:jar4.jar" Project wiki 1 0 1 training\_set.txt (Linux)

The 3 jeopardy questions which are being answered in the **new** method are:

a)UCLA CELEBRITY ALUMNI

This woman who won consecutive heptathlons at the Olympics went to UCLA on a basketball scholarship

Jackie Joyner-Kersee

b)SERVICE ORGANIZATIONS

Originally this club's emblem was a wagon wheel; now it's a gearwheel with 24 cogs & 6 spokes

Rotary International

c)'80s NO.1 HITMAKERS

1988: "Father Figure"

George Michael

As it can be observed the CATEGORY in each case ,

in the last case the document of George Michael has a number of his No 1 hits between the years “1981-1986” as well as “1987-1989”, and the CATEGORY was '80s NO.1 HITMAKERS which has contributed significantly in obtaining the answer to this jeopardy question in the final result set.

The same can be noticed in Jackie-Joyner Kersee,which has the CATEGORY: UCLA CELEBRITY ALUMNI where the fact that it was mentioned that she was a UCLA celebrity alumni is present in the document text which causes her to be returned in the results.

When this new method of appending the category to the query was run across the testing data, the results were shown as follows:

Testing data set results:

|  |  |  |
| --- | --- | --- |
| Precision at 1 | Precision at 10 | Mean Reciprocal Rank |
| 19/80=0.2375 | 45/80=0.5625 | 27.48/80=0.3435 |

Run with :

java ".;jar1.jar;jar2.jar;jar3.jar;jar4.jar"Project wiki 1 0 1 testing\_set.txt (Windows)

java ".:jar1.jar:jar2.jar:jar3.jar:jar4.jar"Project wiki 1 0 1 testing\_set.txt (Linux)

which is noticed to be significantly higher than my previous result obtained without appending the CATEGORIES to the query.

**2)Short length of the query:**

Another feature which can be observed is the length of the query.

In the original method (without appending CATEGORIES to query) used by me, for those queries (clues) which have a length of more than 10-12 words on average return results and those queries which have queries of length less than 10 words usually do not have a precision in the top 10 itself, a few examples can be observed:

The queries which had results returned are:

a)

The practice of pre-authorizing presidential use of force dates to a 1955 resolution re: this island near mainland China

Taiwan

b)

Daniel Hertzberg & James B. Stewart of this paper shared a 1988 Pulitzer for their stories about insider trading

The Wall Street Journal

c)

After the fall of France in 1940, this general told his country, "France has lost a battle. But France has not lost the war"

Charles de Gaulle|de Gaulle

As can be seen, the length of the query in these cases is an average of 12-13 words.

Some of the queries which did not even have a precision in the top 10 are:

a)

The Naples Museum of Art

Florida

b)

In 2009: Joker on film

Heath Ledger

c)

1988: "Father Figure"

George Michael

d)

This Italian painter depicted the "Adoration of the Golden Calf"

Tintoretto

From the above it can be noticed that these queries have an average length of 6-7 words.

The query (HINT) does not have a sufficient number of words with which it can compare to the corresponding texts of the documents stored in the disk directory in order to obtain a good result.

Thus it can be concluded that a minimum number of words in the query are required in order to get a good and satisfactory result.

**Impact of stemming and lemmatization:**

a)no stemming or lemmatization:

The results I obtained without performing any stemming or lemmatization (used the WhiteSpace Analyzer) on the data are as follows

Testing Data Set Results:

|  |  |  |
| --- | --- | --- |
| Precision at 1 | Precision at 10 | Mean Reciprocal Rank |
| 3/80=0.0375 | 15/80=0.1875 | 5.67/80=0.0708 |

Run with :

java ".;jar1.jar;jar2.jar;jar3.jar;jar4.jar"Project wiki 0 0 0 testing\_set.txt (Windows)

java ".:jar1.jar:jar2.jar:jar3.jar:jar4.jar"Project wiki 0 0 0 testing\_set.txt (Linux)

b)With stemming (used the Standard Analyzer):

Testing Data Set Results:

|  |  |  |
| --- | --- | --- |
| Precision at 1 | Precision at 10 | Mean Reciprocal Rank |
| 13/80=0.1625 | 41/80=0.5125 | 21.81/80=0.2726 |

java ".;jar1.jar;jar2.jar;jar3.jar;jar4.jar"Project wiki 1 0 0 testing\_set.txt (Windows)

java ".:jar1.jar:jar2.jar:jar3.jar:jar4.jar"Project wiki 1 0 0 testing\_set.txt (Linux)

c)With lemmatization:

Initially I used The lemmatizer on the Stanford Core NLP website, however it took very long to create the dictionary hence I have not added the commands, but the results recorded by me were as follows:

Testing Data Set results:

|  |  |  |
| --- | --- | --- |
| Precision at 1 | Precision at 10 | Mean Reciprocal Rank |
| 8/80=0.1 | 33/80=0.4125 | 14.37/80=0.1796™™ |

My best result was the one I obtained through stemming as that had given me the best result of all, reason being that words were chopped off and many words could be matched in a better and more accurate way as compared to that of neither stemming or lemmatization or just lemmatization.

Neither stemming nor lemmatization did not work well as each word, be it a past, present or future tense of another word is indexed differently, which results in a very large dictionary as well as less accurate results.

Using lemmatization, the words even after being processed/normalized still resulted in being different to that in the query, which also did not match the words being indexed in the document.

**Description of the code:**

In the beginning, all the variables which are being used in the program are initialized as well as all the arguments given.

Lines Start-95:

Initializing and declaring the variables, analyzers as well as the arguments which are used in the program.

Lines 97-268:

The dictionary is created.

It consists of indexing document titles as well as its contents. During the file parsing, using Regex all the unnecessary details (such as contents of References, External Links, See also & lines which have == ==) are all discarded and not indexed as the contents of the document.

Lines 272-350:

The queries are formed from the chosen files (training document or testing document), by using the HINT and CATEGORIES (improved method) from the jeopardy questions.

Lines 350-End:

The queries formed are processed with the contents of documents and the if the result is present, the document title along with its top 10 hits are displayed.

The precision at 1, precision at 10 and Mean Reciprocal Rank are displayed for the respective data set used.