Hongmin (Steven) Kim
Alejandro Macias
Amara Tariq

##How to run and other build specifications

Before attempting to compile program, please go to the following Google Drive folder and download the folder and place within the ‘P1’ directory:
https://drive.google.com/file/d/18raxfHiFXEZJceGJt2U0Tnx5xEeByFxN/view?usp=sharing

These are preprocessed files that our search engine uses in order to retrieve results. These were created with efficiency in mind.

After downloading and placing this folder in the P1 directory, run the following command to be able to compile and run our program:

``` python mainDriver.py ```

Initial bootup includes processing in the files, but it should only take around a minute and a half.

##Text Processing

Details on implementation decisions pertaining to text processing (which tokenizer you used, any modifications make, stemmer/lemmatizer used, etc.).

For our text processing, we first make sure to lowercase the entire document and remove any punctuation. We chose to remove all punctuation. The two pieces of punctuation that were a bit harder to decide to remove were dashes (-) and apostrophes (‘). One of the homework assignments had “old-school” in the documents, and this is where we questioned whether it was good to remove the dash. Although removing the dash could alter the meaning of “old-school,” we were conscious of other situations where we would really lose a chance of relevancy if we kept the dashes in. Ultimately we did think it was more worth removing. Apostrophes were a bit easier to decide, especially due to the custom stopword list we came up with, which would check for single letters (ie ‘a’ and ‘s’). This made taking possessive apostrophes out easier, and we felt like, similar to the dashes, the few instances where they would be useful wouldn’t lose a whole lot of meaning if we were to remove the apostrophe.

For stopword lists, we had initially decided to use the NLTK default stopword list. We had never had experience with many stopword lists, and since we had decided to code in python, NLTK seemed to be the most popular stopword list. After using it for some of the homeworks, one potential issue arose, we found that the list didn’t take out as many words as would be useful for a larger dataset like we are using here. Due to this, we decided to join two stopwords lists together to ultimately process out more stopwords, while making the tokens we gained afterward more accurate to getting a document's meaning and relevance. The stopword list we joined with NLTK was from ‘countwordsfree.com/stopwords,’ which we decided to use due to including more formal, “textbook” based stopwords. We thought this would help due to processing wiki pages, which tends to use more “textbook” language.

For stemming and lemmatization, we decided to stem our text because since we are working with a large data set, we decided that reducing morphological variations would reduce the complexity of the text processor itself. Thus, our text resulted in a more common stem variation by removing the suffixes. In order to accomplish the stemming task, we utilized the nltk.stem.porter to reduce the morphological variation of the words. The reason we chose this package was because our concern was to leverage our retrieval performance as well as our search engine efficiency. During the testing phase of our implementation, we discovered that the resulting stemmed word was sufficient to map to words with a more generic or common root word. For example, if the word didn’t result in a valid root word such as the word ‘happiness’ stemmed to an uncommon root word ‘happi.’ With the help of the algorithm nltk.stem.porter, the algorithm identified synonyms to attempt to match the stemmed word ‘happi’ which in this case mapped to the word ‘happy.’ As a result, we successfully implemented a solution to efficiently expand our information resources that are relevant to the query.

##Statistics on Document Collection

Statistics on DC (number of documents, index size pre/post stemming/lemmatization, stop word removal, etc.)

The total number of documents we have is 1,286,263 documents.

##Query Suggestions

Details on implementation decisions pertaining to query suggestions (which data structures did you used to ease the retrieval of suggestions, etc.).

We decided to preprocess and create some additional csv files for data such as the query logs. This was done to not only have a location with all the data formatted in a way we could use later, but to also save time and be more efficient when the next user runs the program. The other file we created and saved was an index for the query logs. This overall helped ease the retrieval of the suggestions.

Throughout our code, one of the other methods we used to gain efficiency was to avoid using the traditional loops (ie for, while, etc). Instead we used pandas built in function to be able to instantaneously combine the columns in our dataframe. Another major reason we decided to do data frames throughout the project.

##Relevance Ranking

Details on implementation decisions pertaining to relevance ranking (data structures that eased the process, libraries used, etc.).

This section was a bit more straightforward in terms of creating functions for computation. Each numerator/denominator of the equation has its own function. We didn't use any specific libraries for this computation, everything was based on our data frames. Anything new we computed that we knew we would need in the future would become a new column added to the data frame. Not only was this again very efficient, but it made it easier to also view the output and see what is going on.

##Snippet Generation

Details on implementation decisions pertaining to snippet generation (data structures that eased the process, libraries used, etc.).

This was one of the hardest sections for our group. We just really struggled to understand the equation and apply it to individual sentences efficiently. We originally kept trying to see if we could use some of the functions we had developed for relevance ranking above, but in the end we decided it would be best to just take some time and really make specific code for processing individual sentences.

One major issue and decision we made was to not return all the relevant documents found. This was mainly due to our main test query of "morocco saudi," which returned over 4,000 documents. We found that creating snippets of all of these documents wasn't going to be helpful as a user wouldn't really look past 100 documents. In order to help limit the processing time and also make the output better for a user, we decided to limit the process to the top 50 relevant documents.

##Result Analysis

For the discussion of the generated results, you should include your 5 test-queries, a sentence discussing the intent (i.e., expected information), outcomes of your searches using your SE (as specified on Tables 1 and 2) and Wikipedia search module (as specified on Table 3), your thoughts on the results retrieved by your SE, and an overall performance analysis based on the resources retrieved (or lack thereof) by Wikipedia with respect to your SE.

For the "Frozen foods" and "Frozen characters" queries. We are not quite sure why our snippets are not functioning. These are one of the few words that we noticed that the snippets don't work properly. According to the errors, this could be due to our logic in the computation for the terms or this can also be due to the version from .pynb files to .py files. We have tried our best to resolve this issue, but we are taking note that this will be a good starting point and opportunity to enhance this specific feature for project 2.

TestSet Queries
Frozen Characters
Morocco Saudi
describe how to throw a ball

how to throw a ball
Frozen foods

Table 1 - Test Set Queries - 5 queries chosen by group members

Query 1 - Frozen Characters

This query was chosen out of curiosity for how our search engine would process and return documents for a word that could mean and return slightly different resources since “frozen” can refer to both a physical state and the Disney movie franchise.

Query: Frozen Characters			
Ranking	*Document ID*	*Snippet*	*RankingScore*
1			
2			
3			

Table 1 - top three snippets

Query: Frozen Characters		
Ranking	*Candidate suggestion*	*Score*
1	Frozen Pool	9.510939
2	Freezer centre	9.510939
3	Frozen Television	9.510939

Table 2 - top three candidate suggestions

From this table above, you can see that based on our text processing, our search engine couldn’t really tell the difference between frozen for something like frozen food, and the Disney Frozen. The results could probably be more accurate if lemmatizing was used, as that helps differentiate words that may sound and spell the same but mean different things.

Query: Frozen Characters	
Ranking	*Document Title*
1	Frozen (franchise)
2	Frozen (2013 film)
3	Olaf (Frozen)
4	Anna (Frozen)
5	Elsa (Frozen)

Table 3 - Top 5 results in Wiki searches for “Frozen Characters”

Query 2 - Morocco Saudi

When we were testing our system, this was one of the few terms that kept catching our eyes. One of the reasons we decided to include this was due to our initial discussion on what punctuation to remove. When referring to two countries, especially where there are documents discussing wars, there is usually a dash between them. This was something we had decided to remove, so we were curious to see if that had been a good decision.

Query: Morocco Saudi			
Ranking	*Document ID*	*Snippet*	*RankingScore*
1		Moroccan–Saudi Arabian relations refers to the current and historical relations between Morocco and Saudi Arabia. Morocco and Saudi Arabia have together taken steps to curb Iranian influence in the Arab world, although Morocco has a moderate approach to Iran while Saudi Arabia is more cautious and hostile of Iran.	
2		List of shopping malls in Saudi Arabia This is a list of shopping malls in Saudi Arabia. Riyadh Jeddah Khobar Dammam	
3		Abdullah Al-Sadhan () (born 13 May 1958 in Shaqraa) is a Saudi Arabian actor. He is mostly for his roles in the Saudi comedy Tash ma Tash, along with fellow actor Nasir Al-Gasabi.	

Table 1 - Top three snippets

Query: Morocco Saudi		
Ranking	*Candidate suggestion*	*Score*
1	Morocco-Saudi Arabia relations	18.121403
2	List of shopping malls in Saudi Arabia	9.11609
3	Abdullah Al Sadhan	9.11609

Table 2 - Top three candidate suggestions

Table 2 compared to below (wiki results) had the same document for rank 1. It was interesting to see that shopping malls ended up being rank 2 for us, but it makes sense if the document is giving addresses of each shopping mall which would include the country at the very end of the address. Wikipedia probably processes addresses in a way so that it doesn't cloud the results and ranking with the repetition addresses can have.

Query: Morocco Saudi	
Ranking	*Document Title*
1	Morocco–Saudi Arabia relations
2	Saudi Arabia national football team
3	Saudi Arabian-led intervention in Yemen
4	Morocco national football team
5	Brioche Dorée

Table 3 - Top 5 results in Wiki searches for "Morocco Saudi"

When searching in Wiki search, we were happy to still see a document with "Morocco-Saudi" (with a dash). What was interesting to us was the amount of football articles that immediately popped up, as we expected the top articles to be politically related when searching for two countries. The most surprising article was number 5, as it's a French bakery chain, with the only connection being that they have locations in these countries. The article didn't even mention either country more than once.

Query 3 - describe how to throw a ball

This query was chosen to emulate how some users will search full questions in their search engine.

Query: describe how to throw a ball			
Ranking	*Document ID*	*Snippet*	*RankingScore*
1		Bang Bang Ball is an arcade game released by Banpresto in 1996. Skateboarding mice must throw coloured balls against clusters of moving balls, while avoiding being hit by any of the balls.	
2		Throw distance is extended by ball spin, but the curve is otherwise unnoticeable. The grooves in the racquet along with the striations on the ball provide a great deal of spin which allows players to throw deep curveballs with little effort.	
3		"Juggle two balls in one hand, but instead of throwing and catching the third ball with the other hand, you simply hold it in that hand, moving the ball up and down." Another common variation is to carry one of the two outside throws up and down rather than throwing it, which may be called the Dummy pattern, or "fake columns" (siteswap: (4,2)),	

		which may be interpreted as adding a carry to the "lone" hand during the two-in-one.	

Table 1

Query: describe how to throw a ball		
Ranking	*Candidate suggestion*	*Score*
1	Bang Bang Ball	9.653202
2	Trac Ball	9.213528
3	Columns Juggling	9.194703

Table 2 - Top three candidate suggestions

As you can see above, the types of suggestions we generated weren't very specific to throwing balls. We aren't quite sure what caused such discrepancies.

Query: describe how to throw a ball	
Ranking	*Document Title*
1	Curveball (redirect from Curve ball)
2	Glossary of baseball terms (section throw a clothesline)
3	Siteswap (section Connections to abstract algebra)
4	Rules of basketball (redirect from How to play basketball)
5	Back-pass rule (section Tricks to circumvent the rule)

Table 3 - Top 5 results in Wiki searches for "describe how to throw a ball"

One interesting thing immediately for the wiki searches was how most of the returned references referred to baseball, where we expected football to show up. Though this search is more accurate to the query than what our search engine was able to come up with. Accuracy in terms of the resources seems more relevant.

Query 4 - how to throw a ball

This query came up from the previous one, as we wanted to see how different it would be to blatantly ask a "how to" question versus searching for a description of how to.

Query: how to throw a ball			
Ranking	*Document ID*	*Snippet*	*RankingScore*
1		Bang Bang Ball is an arcade game released by Banpresto in 1996. Skateboarding mice must throw coloured balls against clusters of moving balls, while avoiding being hit by any of the	

		balls.	
2		Throw distance is extended by ball spin, but the curve is otherwise unnoticeable. The grooves in the racquet along with the striations on the ball provide a great deal of spin which allows players to throw deep curveballs with little effort.	
3		"Juggle two balls in one hand, but instead of throwing and catching the third ball with the other hand, you simply hold it in that hand, moving the ball up and down." Another common variation is to carry one of the two outside throws up and down rather than throwing it, which may be called the Dummy pattern, or "fake columns" (siteswap: (4,2)), which may be interpreted as adding a carry to the "lone" hand during the two-in-one.	

Table 1 - Top three snippets

Query: how to throw a ball		
Ranking	*Candidate suggestion*	*Score*
1	Bang Bang Ball	9.653202
2	Trac Ball	9.213528
3	Columns juggling	9.194703

Table 2 - top three candidate suggestions

From above, you can see that we returned essentially the same information compared to “describe how to throw a ball.” Our stopwords were able to get both queries to equal “throw ball.” Though it seems like a good approach, as you can see in table 3 for each query, Wikipedia was able to somehow still differentiate a difference.

Query: how to throw a ball	
Ranking	*Document Title*
1	Throwball (redirect from Throw ball)
2	Curveball (redirect from Curve ball)
3	Dodgeball (redirect from Dodge ball)
4	Sinker (baseball) (section Throwing mechanics)
5	Rules of basketball (redirect from How to play basketball)

Table 3 - Top 5 results in Wiki searches for “how to throw a ball”

One major difference based off the snippets we observed in this wiki was that these queries actually had step by step guides on how to throw a ball, whereas the last query was more descriptive in the types of throwing.

Query 5 - Frozen foods

We chose this query to try and test a similar search to “Frozen Characters” and see if we can find different resources as “frozen” in both mean different things.

Query: Frozen foods			
Ranking	*Document ID*	*Snippet*	*RankingScore*
1			
2			
3			

Table 1 - Top three snippets

Query: Frozen Foods		
Ranking	*Candidate suggestion*	*Score*
1	Freezer centre	11.49549
2	Frozen television	9.510939
3	Frozen Head	9.510939

Table 2 - Top three candidate suggestions

Compared to the first test query, this one still gave results that don’t differentiate between the meaning of “frozen” in both cases. Compared to the wiki results below, they were actually able to see a clear difference in their returned documents.

Query: Frozen foods	
Ranking	*Document Title*
1	Frozen food
2	List of frozen food brands
3	Morton Frozen Foods
4	Clarence Birdseye (redirect from Birdseye Frozen Foods)
5	Ajinomoto (redirect from Ajinomoto Frozen Foods)

Table 3 - Top 5 results in Wiki searches for “how to throw a ball”

<u>Insights and Lessons Learned</u>

For your overall discussion, you should describe insights on the project (challenges and solutions).

One of the most challenging parts of this was to just simply wrap our brains around how to efficiently read and manipulate the data. It was hard to understand overall what some sections were supposed to be doing, such as the suggested queries and cosine similarity. These were sections that we literally had to break down to baby steps in order to really understand what needed to be done. Group collaboration was a must, because together we were able to understand.

Another major challenge for the two undergrads was just keeping up with the experience of our graduate team member. The undergrads had never really worked with any large scale data or text processing, where for the graduate student this was their field of interest, so they knew a lot. Undergrads were able to learn a lot, and pair programming really helped gain some confidence in being able to not only code in python, but also be able to efficiently use dataframes and pandas.