My grades for **Midterm exam**

Q1

3 / 3

In IR, what are similarity measures for? Name and discuss 4 of them that we studied.

We use similarity to compare **how similar** two documents or a document and a query are.

1. **Cosine similarity**: First we use TF-IDF to calculate the frequency of every term in two documents, then we **measure the cosine of the angle between two vectors**. We don't care about the length of the vectors, we only care about the angle.
2. **Jaccard similarity**: First we use n-grams/shingling to divide each of the two documents to be many shingles. We will get a term/shingle set from each document. Then we **measure the ratio of the intersection to the union of two sets**.
3. **Spearman correlation**: First we represent each one of the two

documents as two ranked lists, the elements in the list should be terms and sorted by their rank. Then we **measure the non-linear relationship between two sets**.

4. **Weight similarity**: First we use a vector to represent all terms in each one of the two documents with weights(e.g. the frequency of the term). Then we **measure the sum of the dot products of the weights of the matched terms**.

Q2

2 / 2

How is 'speedup' achieved by a search engine, in serving queries, and in crawling the web, in terms of these four aspects: data structures, computational machinery, disk space, bandwidth? In other words, provide four different ways/techniques, one related to each aspect.

---

1. Data structures: we use **tire** data structure to compress data to reduce its size and improve access time; We also use **inverted index** to store data so that we can retrieve them quickly by using key-value structure.
2. Computational machinery: We use **distributed computing** to divide the workload among multiple machines or processors. We use **MapReduce** to divide big data into many chunks to achieve the speed-up of big data in distributed computing.
3. Disk space: We use **deduplication** methods to reduce the amount of disk space needed to store data.
4. Bandwidth: We use **content delivery networks**(CDN) to

distribute traffic among multiple servers or locations. For example, if a user in Japan wants to access a video on YouTube, the CDN will first try to find the nearest server to find that video. If the server has the video, it will directly deliver it to the user; if not, it will request that video from the host server.

Q3

2 / 2

Absolutely incorrect method

How is a web browser and a web crawler similar? How are they different?

Similar: Web browsers and web crawlers both **request and receive web pages from servers**. Different: Web browsers **display web pages for human users**, while web crawlers only **index web pages for search engines**. Therefore, a web crawler can also visit many more web pages than a typical user would with a browser.

Q4

3 / 3

Broadly speaking, pages in a site typically point to other pages in the site (eg. think CNN, Wikipedia, OfferUp, YouTube, usc.edu, etc).

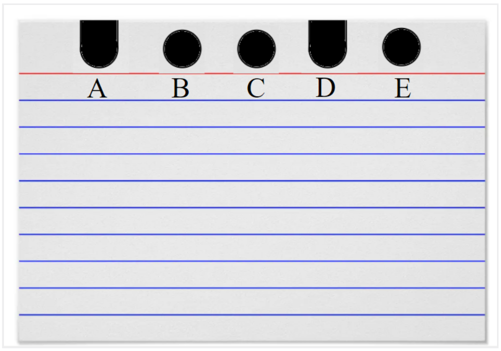
A web crawler can be written, to hold future (as of yet unvisited) URLs in a queue data structure, or, in a stack data structure. Which of these would site administrators prefer (if they had a say in the crawler architecture), and why?

A **queue data structure should be preferred** by site administrators because it refers **BFS**, which means the web crawler will access all the **direct links** in the current web page first. It's better because we usually believe that the relevance or PageRank of a web page will gradually decrease with the number of links jumps. That is to say, other web pages that can be directly clicked and accessed in a web page are some web pages most related to the current web page. The stack structure refers to the DFS algorithm, that is, we will first click on a certain link until there are no sub-pages. And this access method will quickly access the webpage that is far away from the current webpage, which is meaningless to the website administrator.

Q5

0 / 2

We can use 'index cards' to index a small collection of books (THAT is why they are called that - duh!). We use a card for each book that we want to index. Shown below is a sample index card, for a book that belongs in categories B, C, E (intact holes), but not categories A,D (cut-out holes). Given a collection of such cards (eg. 1000 of them) where each card is for a book that could be in any of A,B,C,D,E categories (at least one, but can be 2, 3, 4 or all 5), how would you pick out all the books that are in **B and C** categories? Using the same 500 cards again, how would you select books that can be in **A or D** categories? You have access to several long rods (eg. knitting needles) that can pass through the holes.

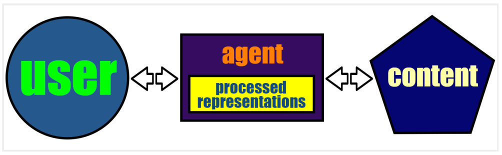


I would use KNN to classify all the cards. Because knn is better on polymorphic problems.

Q6

3 / 3

The following diagram is on the front page of our site:



Discuss the role of the agent with respect to the user, and, with respect to the content? In other words, what goes on in the left pair of arrows, and, in right pair?

What is different about the agent's behavior, when the agent is based on a large pretrained language model, eg when it is ChatGPT integrated with Bing, or Google's Bard?

The pair of arrows on the left and the pair of arrows on the back are **two independent processes**. For the pair of arrows on the left, that is, the communication process between the user and the agent: the user will first send a request/query to the agent to request some content. At this time, the agent will first use the **inverted index** and find out the

You were to name this site

raw content of the content that the user wants to access, and then use the **information retrieval system** to find out some top contents that are most relevant to the user's query. At the same time, on the right part, the agent will regularly **crawl** new raw content or some raw content updates through crawlers to ensure that its own data is up-to-date. After the crawler gets the raw content, the agent will use the **inverted index** to store the raw content in the index format. (*the work of the left and right parts can be distributed through MapReduce to greatly speed up*).

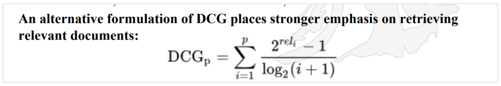
Q7

3 / 3

What is the purpose of (reason for) discounting cumulative gains, when we rank search engines?

We typically use 1/log2(rank+1), for discounting (because 1/rank is too 'harsh'). What other rank-based function can you think of (which is also less harsh than 1/rank)?

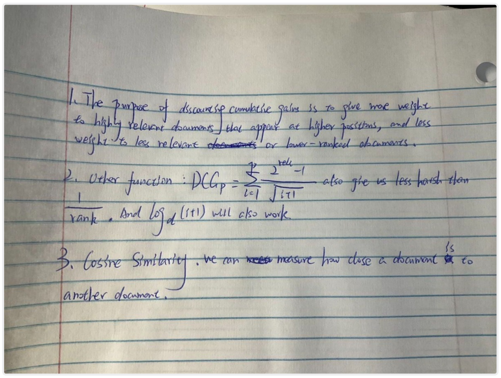
To boost relevance, use this alternative formula:



An alternative formulation of DCG places stronger emphasis on retrieving relevant documents:

$$\mathrm{DCG_p} = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{\log_2(i+1)}$$

What other function (of relevance) can we use for this purpose?



1. The purpose of discounting cumulative gains is to give more weight to highly relevant documents that appear at higher positions, and less weight to less relevant ~~documents~~ or lower-ranked documents.

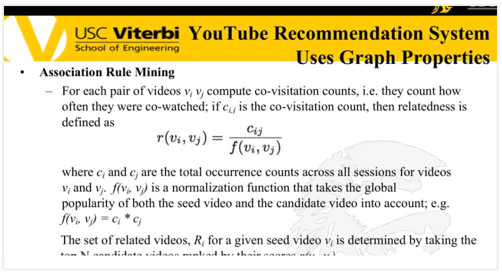
2. Other function : $DCG_p = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{\sqrt{i+1}}$ also give us less harsh than $\frac{1}{rank}$. And $\log_d(i+1)$ will also work.

3. Cosine Similarity . we can ~~need~~ measure how close a document ~~[illegible]~~ is to another document.

Q8

2 / 2

YouTube came up with a relatively simple way to create recommended videos:



USC **Viterbi** School of Engineering **YouTube Recommendation System Uses Graph Properties**

- **Association Rule Mining**
  - For each pair of videos v_i v_j compute co-visitation counts, i.e. they count how often they were co-watched; if $c_{i,j}$ is the co-visitation count, then relatedness is defined as

$$r(v_i, v_j) = \frac{c_{ij}}{f(v_i, v_j)}$$

where c_i and c_j are the total occurrence counts across all sessions for videos v_i and v_j. $f(v_i, v_j)$ is a normalization function that takes the global popularity of both the seed video and the candidate video into account; e.g. $f(v_i, v_j) = c_i * c_j$

The set of related videos, R_i for a given seed video v_i is determined by taking the

Rather than such a co-visitation count approach, it's also possible to group videos based on their content - what are two very different ways to do this?

---

1. We can relate to the idea of YouTube's **ContentID**. We represent each video with its contentID, because the idea of contentID has the idea of **SimHash**, that is, two videos with more similar content will get closer hash values. Therefore, we can group videos according to their contentID, and recommend the topK videos with the most similar hash values to users.
2. We can also use the idea of **classification** to group videos, such as KNN and Rocchio algorithm. I prefer using KNN

algorithm because the video here is a **polymorphic individual**, and KNN, an algorithm that pays more attention to individual characteristics, will pay more attention to the average of categories than Rocchio. feature algorithm to achieve better performance.

Q9

3 / 3

For 'power searching' Google, we use search modifiers such as :site, :filetype, :intext, etc. These help narrow down the search

What three additional keywords (search modifiers like the above) can you think of (that would be useful to people if Google added them), to narrow down searches to specific categories: "when I enter a search term/phrase X, I want it to be narrowed down to Y category"? Think broadly!

---

1. :info. It will narrow the search results of the term X down to **information** category. For example, if someone wants to know how to make an apple pie, they could search for ":info applepie" to get the recipe.
2. :language. It will search for content in a **specific language**. For example, if someone wants to find information about a topic in Chinese, they could use the search modifier ":language=cn" to narrow down the search results to pages written in Chinese.

3. :source. It will search for content from a **particular source or website**. For example, if someone wants to find information about a specific topic from a reliable source such as a university website, they could use ":source.edu" to narrow down the search results.

Q10

2 / 2

Sites such as OfferUp, uspto.gov, eBay, RateMyProfessor etc offer specific services (eg we don't search for an ML pdf on OfferUp). Given that, how would (or do) the four sites mentioned above, make it easy to search for what they offer? In other words, what might each one index (so that we can search those indexes)?

1. OfferUp may use the **position** as the key of the inverted index and all **companies** as posting lists.
2. uspto.gov may use the **category of the patent** as the key in the inverted index and all **related patents** as the posting lists.
3. eBay may use the **item** as the key in the inverted index, and use the **different seller websites** corresponding to each item as posting lists.
4. RateMyProfessor may use the **professors** as the key in the inverted index and all **reviews** as posting lists.

Q11

3 / 3

Characterize Rocchio, kNN and nearest-neighbor techniques for document classification (where we assign an incoming document, one of 'n' existing classes), in terms of

- aggregation (averaged or not)
- geometry (point, line, area...)
- robustness

In other words, for each technique, talk about the three aspects listed above.

Rocchio: aggregation: averaged, geometry: area, robustness: good because it uses the average value. kNN: aggregation: average of a small group of individuals, geometry: points/vectors, robustness: good in polymorphic problems. nearest-neighbor: aggregation: not, geometry: point, robustness: not good, only focus on one point.

Q12

1 / 2

Names two sites that you use, where 'approximate matches' are performed and results displayed, when a user enters a query (searches for something)?

Google; Similar Site Search