# Ranking by Set Similarity

CS5154/6054

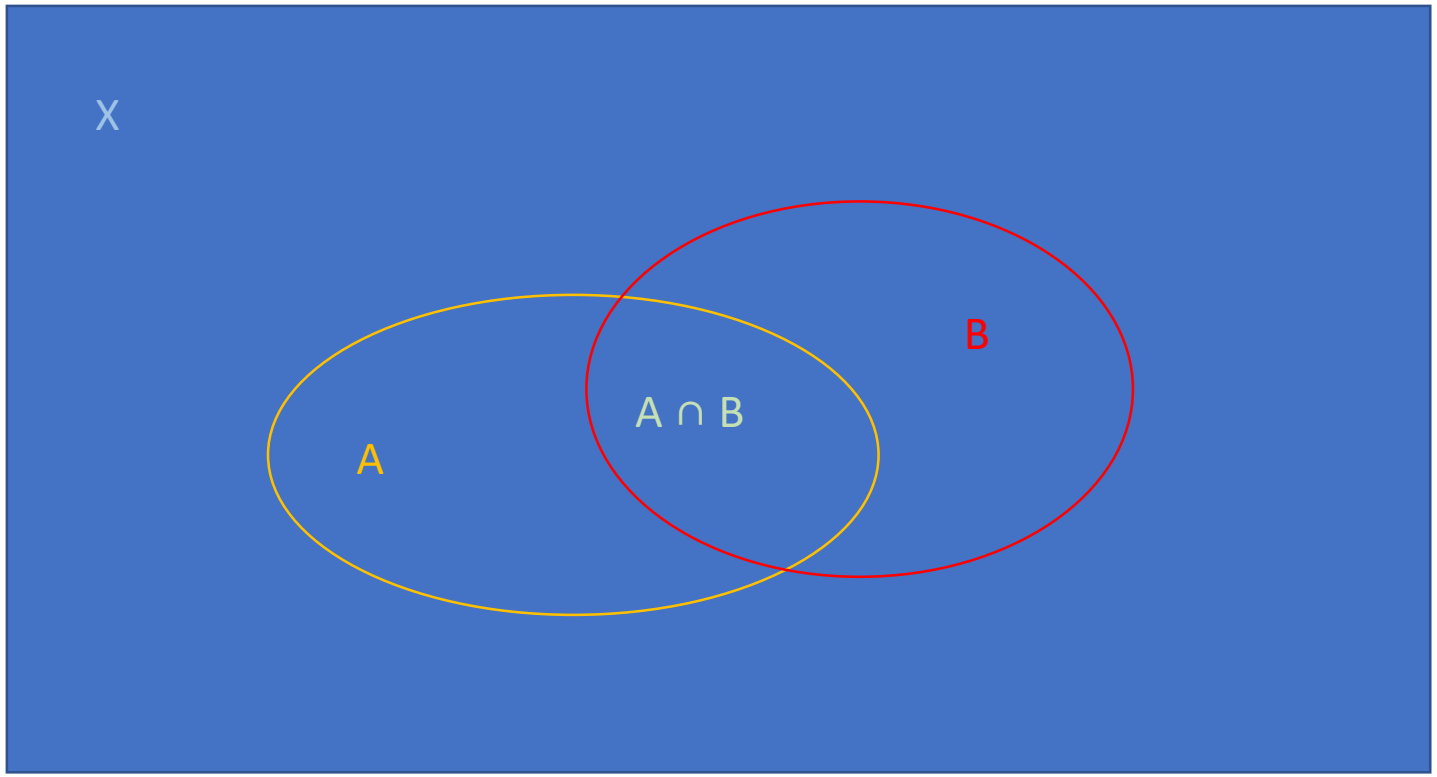
Yizong Cheng

8/30/2022

# Set Representation of Documents

- A document is a set of words.
  - Actually, a sequence of words.  But more on this later.
- A query is a set of words.
  - Let’s soften the Boolean AND and OR between words in the query.
- Retrieve documents in a collection that rank high in set similarity to the query.
- Do we have to go through all documents?
  - Maybe only those sets (documents) with non-empty intersections to the query.
  - If that is the case, the inverted index helps.

# Set Intersection A ∩ B



# Four Intersections and Their Counts

| | B | not B | |
|---|---|---|---|
| A | $\|A \cap B\|$ | $\|A\| - \|A \cap B\|$ | $\|A\|$ |
| not A | $\|B\| - \|A \cap B\|$ | $\|X\| - \|A\| - \|B\| + \|A \cap B\|$ | $\|X\| - \|A\|$ |
| | $\|B\|$ | $\|X\| - \|B\|$ | $\|X\|$ |

# Expected Intersection Size(s)

- Suppose sets A and B are unrelated.
- Randomly spread |A| points and then |B| points from X.
  - Or, randomly attach ‘A’ to points in X with probability |A|/|X|.
  - And do the same with the ‘B’ with probability |B|/|X|.
- See how many are common.
  - How many X points are  expected to have both ‘A’ and ‘B’ attached?
  - Unrelated events A and B have joint probability that is the product of the individual probabilities.
  - This probability is (|A|/|X|)(|B|/|X|) for each X point to belong to both A and B and the expected intersection size is this multiplied by |X|, or
  - |A||B|/|X|.

# Four Expected Counts

| | B | not B | |
|---|---|---|---|
| A | \|A\|\|B\|/\|X\| | \|A\|(\|X\| -\|B\|)/\|X\| | \|A\| |
| not A | (\|X\|-\|A\|)\|B\|/\|X\| | (\|X\|-\|A\|)(\|X\|-\|B\|)/\|X\| | (\|X\|-\|A\|)/\|X\| |
| | \|B\| | (\|X\|-\|B\|)/\|X\| | \|X\| |

# Intersection or Common Neighbors (CN)

- When |A| and |B| are very small compared to |X|, as in the case of documents and queries compared to vocabulary, the expected intersection count is very close to zero.
- In this case, any intersection is significant.
  - The larger the intersection, the more relevant A and B are to each other.
  - May simply rank documents by the intersection of their term sets to the query term set.
  - Imaging the term-document matrix being a bipartite graph, this intersection is the size of the common neighbor of two nodes in the same part.

# Jaccard Coefficient

- Jaccard coefficient, defined in 3.3.4 of IIR, is a similarity measure between sets.
- It is the ratio of the size of the intersection of the two sets and the size of the union of the two sets.
- When two sets are the same, the intersection and union are the same set and Jaccard coefficient is 1.
- When two sets are disjoint, the intersection is empty, Jaccard coefficient is 0.
- Otherwise Jaccard coefficient is a number in [0, 1].

# Computing Intersection and Union Sizes

- $|A|$ is the size of the set A (the number of elements in A).
- A U B is the union of sets A and B and the size is $|A \cup B|$.
- $A \cap B$ is the intersection of A and B and has the size $|A \cap B|$.
- Knowing one of the union and intersection, the other can be easily computed.
- $|A| + |B| = |A \cup B| + |A \cap B|$.
- Jaccard coefficient (A, B) = $|A \cap B| / ( |A| + |B| - |A \cap B|)$.

# Other Normalized Intersection Measures

- Dice coefficient: Dice (A, B) = $2|A \cap B|/(|A| + |B|)$
  - Size of intersection is no greater than either $|A|$ or $|B|$, or their average.
  - Dice coefficient is between 0 and 1.
  - Exercise 8.7 of IIR
  - Also called balanced F-measure, or F1.
- Hamming distance:
  - Symmetric difference: $D(A, B) = |A| + |B| - 2|A \cap B|$
  - The smaller the Hamming distance is, the more similar are A and B.

# Measures between A and B Using X

- Accuracy
- $\chi 2$(IIR 13.5.2)
- Mutual information(IIR13.5.1)

# How to Rank Sets in a Collection

- For each set A in the collection, compute Jaccard(A, Q) where Q is the query as a set.
  - This requires computing $|A \cap Q|$, $|A|$, and $|Q|$
- Sort the collection according to Jaccard(A, Q) and display the top K.
- Computation problem: compute the intersections between Q and all documents A
- Time needed: proportional to collection size N (number of documents) times size of universe X (size of dictionary).
- With inverted index, can be done in time proportional to $|Q|$.

# How to Rank via the Inverted Index

- For each term t in the query set, find the postings list for t.
- Increment the intersection count for each document in the postings list.
  - Need to initialize the counts to zero before the for loop.
- Only documents in the union of the postings lists for words in the query will be ranked, based on the now available intersection size, and Jaccard coefficient.
  - Need sizes of the sets of words representing documents, which should be pre-computed.
- Much more efficient with the inverted index.

```
# IR3A.py CS5154/6054 cheng 2022
# read lines from a text file as documents
# tokenize each into a set
# randomly name one doc as query
# compute Jaccard coefficient between query and all docs
# print docs with Jaccard coefficient > 0.1
# Usage: python IR3A.py

import re
import random

f = open("bible.txt", "r")
docs = f.readlines()
f.close()
N = len(docs)
sets = list(map(lambda s: set(re.findall('\w+', s)), docs))
query = random.randint(0, N)
print(query)
print(docs[query])
A = sets[query]
for i in range(N):
    B = sets[i]
    C = A & B
    if len(C) == 0:
        continue
    D = A | B
    jaccard = len(C) / len(D)
    if jaccard > 0.1:
        print(i, docs[i], jaccard, C)
```

19941
Ye have multiplied your slain in this city, and ye have filled the streets thereof with the slain.

20 And God blessed them, saying, Be fruitful, and multiply, and fill the waters in the seas, and let fowl multiply in the earth.
0.10344827586206896 {'in', 'the', 'and'}
25 So God created man in his own image, in the image of God created he him; male and female created he them.
0.10714285714285714 {'in', 'the', 'and'}
46 But of the tree of the knowledge of good and evil, thou shalt not eat of it: for in the day that thou eatest thereof thou shalt surely die.
0.125 {'thereof', 'in', 'the', 'and'}
57 But of the fruit of the tree which is in the midst of the garden, God hath said, Ye shall not eat of it, neither shall ye touch it, lest ye die.
0.11764705882352941 {'in', 'ye', 'the', 'Ye'}

```
# IR3B.py CS5154/6054 cheng 2022
# read lines from a text file as docs
# tokenize each as a set of words
# make the inverted index
# name those words with postings list longer
than 1000 stopwords
# remove stopwords from the sets representing
docs
# randomly select a doc as the query
# compute Jaccard coefficients by set
intersection and union
# list docs with Jaccard coefficient > 0.2
# Usage: python IR3B.py
```

```
invertedIndex = {}
for i in range(len(docs)):
    for s in set(re.findall('\w+', docs[i])):
        if invertedIndex.get(s) == None:
            invertedIndex.update({s : {i}})
        else:
            invertedIndex.get(s).add(i)

stopwords = set()
for k, v in invertedIndex.items():
    if len(v) > 1000:
        stopwords.add(k)

N = len(docs)
sets = list(map(lambda s: set(re.findall('\w+', s)) - stopwords, docs))
```

# Stopwords Defined by Postings List Size

{'all', 'unto', 'he', 'saith', 'thou', 'do', 'it', 'no', 'Israel', 'one', 'king', 'came', 'made', 'hath', 'will', 'LORD', 'come', 'be', 'day', 'even', 'up', 'And', 'which', 'and', 'But', 'thy', 'Lord', 'at', 'from', 'for', 'go', 'shalt', 'my', 'but', 'on', 'their', 'that', 'in', 'things', 'are', 'For', 'men', 'is', 'I', 'God', 'an', 'your', 'the', 'people', 'to', 'by', 'were', 'man', 'land', 'him', 'of', 'children', 'you', 'into', 'thee', 'ye', 'when', 'son', 'also', 'us', 'not', 'Then', 'was', 'have', 'hand', 'before', 'there', 'shall', 's', 'upon', 'had', 'The', 'her', 'went', 'they', 'a', 'his', 'them', 'with', 'said', 'saying', 'this', 'as', 'after', 'against', 'me', 'house', 'out', 'we'}

18395
O daughter of my people, gird thee with sackcloth, and wallow thyself in ashes: make thee mourning, as for an only son, most bitter lamentation: for the spoiler shall suddenly come upon us.

18395 O daughter of my people, gird thee with sackcloth, and wallow thyself in ashes: make thee mourning, as for an only son, most bitter lamentation: for the spoiler shall suddenly come upon us.
1.0 {'ashes', 'suddenly', 'sackcloth', 'most', 'O', 'daughter', 'gird', 'spoiler', 'make', 'thyself', 'bitter', 'only', 'mourning', 'lamentation', 'wallow'}
21771 And I will turn your feasts into mourning, and all your songs into lamentation; and I will bring up sackcloth upon all loins, and baldness upon every head; and I will make it as the mourning of an only son, and the end thereof as a bitter day.
0.24 {'sackcloth', 'make', 'bitter', 'only', 'mourning', 'lamentation'}

```
# IR3C.py CS5154/6054 cheng 2022
# read lines from a text file as docs
# tokenize each as a set of words
# make the inverted index
# name those words with postings list longer than
1000 stopwords
# remove stopwords from the sets representing
docs
# randomly select a doc as the query with at least
8 words
# using the inverted index
# retrieve sets of docs containing each of the word
in query
# update a dictionary called intersection
# list docs with intersection to query > 3
# Usage: python IR3C.py

import re
import random
```

```
sets = list(map(lambda s: set(re.findall('\w+', s)) -
stopwords, docs))
for iter in range(12):
    query = random.randint(0, N)
    if len(sets[query]) > 8:
        break
print(query)
print(docs[query])
A = sets[query]
print(A)
intersections = {}
for t in A:
    for d in invertedIndex.get(t):
        if intersections.get(d) == None:
            intersections.update({d : 1})
        else:
            x = intersections.get(d) + 1
            intersections.update({d : x})

for k, v in intersections.items():
    if v > 3:
        print(v, k, docs[k])
```

28659
What then? notwithstanding, every way, whether in pretence, or in truth, Christ is preached; and I therein do rejoice, yea, and will rejoice.

{'truth', 'rejoice', 'or', 'every', 'Christ', 'What', 'preached', 'pretence', 'way', 'whether', 'then', 'notwithstanding', 'therein', 'yea'}
14 28659 What then? notwithstanding, every way, whether in pretence, or in truth, Christ is preached; and I therein do rejoice, yea, and will rejoice.

4 6566 That through them I may prove Israel, whether they will keep the way of the LORD to walk therein, as their fathers did keep it, or not.

4 3249 And every soul that eateth that which died of itself, or that which was torn with beasts, whether it be one of your own country, or a stranger, he shall both wash his clothes, and bathe himself in water, and be unclean until the even: then shall he be clean.

4 28167 For we must all appear before the judgment seat of Christ; that every one may receive the things done in his body, according to that he hath done, whether it be good or bad.

# Recall Counter in Assignment 2

- Counter in IR2A.py does exactly what intersection does in IR3C.py
- Replace the loop with for t in A and the argument to update with invertedIndex.get(t)

```
f = open(sys.argv[1], 'r')

counter = Counter()
for t in f:
    counter.update(re.findall('\w+', t))
```