CSC 583 Homework 2

Sina Ehsani

February 17, 2019

1 Problem 1

Suggest what tokenization and normalized form(s) should be used for these words (including the word itself as a possibility). Justify your decision.

- **'Cos:** token: 'cos, normal: because (NLTK: 'co, Google Search: Cos)
- **Shi'ite** token: shiite, normal: shia (NLTK: shi'it, Google Search: Shiite)
- **cont'd:** token: cont'd, normal: continue (NLTK: cont'd, Google Search: No suggestion)
- **Hawai'i:** token: hawai Normal: hawaii (NLTK: hawai'i, Google Search: Hawaii)
- **O'Rourke:** token: o'rourke normal: orourke (NLTK: o'rourke, Google search: O'Rourke)
- **ain't:** token: ain't normal: beneg (NLTK: ain't, Google search: ain't)
- **me@privacy.net:** token & normal: 'me','@privacy.net' (NLTK: 'me', '@', 'privacy.net')
- $< /htmel >$ **Some text** $< /htmel >$**:** token: Some text, normal: Some text (save the exact text.)

2 Problem 2

Assume a biword index. Give an example of a document (could be a made up paragraph) which will be returned for a query of "New York University" but is actually a false positive which should not be returned.

In the city of New York, University of Colombia is located, which is a Ivy League research university.

3 Problem 3

Shown below in problem 3 is a portion of a positional index in a defined format Which document(s) if any match each of the following queries, where each expression within quotes is a phrase query?

1. "fools rush in"1
document 2 position 1
document 4 position 8
document 7 position 3 and 13

2. "fools rush in" AND "angels fear to tread"
document 4 position 8 starts "fools rush in" and position 12 "angels fear to tread"

4 Problem 4

Write down the entries in the permuterm index dictionary that are generated by the term "hope".

hope$, ope$h, peho, ehop, $hope

5 Problem 5

Compute the edit distance between "paris" and "arid". What are the N (rows) and M (columns) dimensions of the edit distance matrix? Write down the $N \times M$ array of distances between all prefixes as computed by the edit distance algorithm in Figure 3.5 in IIR. For each cell in the matrix, use the four-number representation to keep track of your intermediate results.

The dimensions as you can see from Figure 1, the matrix is 4×5. The matrix can also be flipped (from paris to arid) which make is 5×4 dimensions, but same distance 2

		p	a	r	i	s
	0	1 1	2 2	3 3	4 4	5 5
a	1	1 2	1 3	3 4	4 5	5 6
	1	2 1	2 1	2 2	3 3	4 4
r	2	2 2	2 2	1 3	3 4	4 5
	2	3 2	3 2	3 1	2 2	3 3
i	3	3 3	3 3	3 2	1 3	3 4
	3	4 3	4 3	4 2	3 1	2 2
d	4	4 4	4 4	4 3	3 2	2 3
	4	4 4	5 4	5 3	4 2	3 2

Figure 1: The yellow path shows the minimum edit distance path

6 Problem 6

Consider the fragment of a positional index shown in homework problem 6.
The /k operator, word1 /k word2 finds occurrences of word1 within k words of word2 (on either side), where k is a positive integer argument. Thus k = 1 demands that word1 be adjacent to word2.

1. Describe the set of documents that satisfy the query Gates /2 Microsoft.

Document 1: (Gates 3, Microsoft 1)
Document 3: (Gates 2, Microsoft 3)

2. Describe each set of values for k for which the query Gates /k Microsoft returns a different set of documents as the answer.

If we give /k the value of 1 and 5;
for /k = 1: Only document 3 will be returned.
for /k = 5: Documents, 1,2, and 3 will be returned.

7 Problem 7: Project

7.1 Part 1

Construct a positional index and add support for Boolean proximity queries using the /k operator. That is, word1 /k word2 finds occurrences of word1 within k words of word2 (on either side), where k is a positive integer argument. Hint: use the algorithm from Figure 2.12 in the IIR textbook.

The following code was used (python):

```
def positionalintersect(p1,p2,k,invertedindex,tokenized):
    '''This function is for proximity intersection of postings lists p1 and p2. The function
     finds places where the two terms appear within k words of each other and returns a list
     of triples giving docID and the term position in p1 and p2.'''
    answer=list()
    shareddoc = set(invertedindex[p1]) & set(invertedindex[p2]) #Find the shared documents
    set
    for i in shareddoc :
        p1indices = [i for i, x in enumerate(tokenized[i-1]) if x == p1]
        p2indices = [i for i, x in enumerate(tokenized[i-1]) if x == p2]
        l=list()
        for pp1 in p1indices:
            for pp2 in p2indices:
                while True:
                    if abs(pp1-pp2) <= k:
                        l.append(pp2)
                    elif pp2 > pp1:
                        break
                    while len(l)!=0 and abs(l[-1] - pp1) > k:
                        l.pop()
                    for s in l:
                        answer.append((i, pp1, s))
                    break
    return(answer)
```

What does your code return for the file above and the query: schizophrenia /2 drug? How about schizophrenia /4 drug?
schizophrenia /2 drug:
$[(1,3,1),(2,1,2)]$
This means we have two matches:
a) In the document 1 (Doc1), schizophrenia is in position 3 and drug is in position 1.
b) In the document 2 (Doc2), schizophrenia is in position 1 and drug is in position 2
schizophrenia /4 drug:
$[(1,3,1),(2,1,2),(3,5,1)]$
This means we have three matches:
a) In the document 1 (Doc1), schizophrenia is in position 3 and drug is in position 1.
b) In the document 2 (Doc2), schizophrenia is in position 1 and drug is in position 2
c) In the document 3 (Doc3), schizophrenia is in position 5 and drug is in position 1

7.2 part 2

Modify the above algorithm to be directional. That is, the query word1 /k word2 must return occurrences of word1 strictly before word2, within k words.
The following code was used (python):

```
def positionalintersect2(p1,p2,k,invertedindex,tokenized):
    '''This function is modified version of the positionalintersect function.
    which means given the query word1 /k word2 is will return occurrences of word1 strictly
    before word2, within k words.'''

    answer=list()
```

```
    shareddoc = set(invertedindex[p1]) & set(invertedindex[p2]) #Find the shared documents
    set
    for i in shareddoc :
    #     p1index=tokenized[i-1].index(p1)
    #     p2index=tokenized[i-1].index(p2)
        p1indices = [i for i, x in enumerate(tokenized[i-1]) if x == p1]
        p2indices = [i for i, x in enumerate(tokenized[i-1]) if x == p2]
        l=list()
        for pp1 in p1indices:
            for pp2 in p2indices:
                while True:
                    if 0 < pp2-pp1 <= k:
                        l.append(pp2)
                    elif pp1 > pp2:
                        break
                    elif pp2 > pp1:
                        break
                    while len(l)!=0 and abs(l[-1] - pp1) > k:
                        l.pop()
                    for s in l:
                        answer.append((i, pp1, s))
                    break
    return(answer)
```

What does your code return for the file above and the query: schizophrenia /2 drug:
$[(2, 1, 2)]$
This means we only have one match:
In the document 2 (Doc2), schizophrenia is in position 1 and drug is in position 2

8 Problem 8

Artificial intelligence (AI) and automation in general clearly improve the quality of our lives (think Google Scholar). However, in many cases, they also eliminate jobs (e.g., the self-driving car impacts the livelihood of taxi drivers). Most often, these negative side effects impact the people least prepared to recover. If you were a policy maker, how would you address this problem? Please describe your solution and explain one of its advantages, and one drawback.

First of all, it should be mentioned that in addition to the improvement in the quality of life that automation brings, laborer jobs (blue-color worker) will also be replaced with more professional jobs. This means the importance of education will increase as well, and having more educated people in a society will be very beneficial in the long-term.

Regarding the policymaking, I believe AI and automation will increase the country's GDP (gross domestic product). Having a wealthier country, we can help people that have been more affected to recover. By either helping them to get educated (if young) or even handing them salary without asking them to work (for older people). One advantage that this method has is that it will help people to get more educated. In the other hand, the disadvantages of this method are that it might increase laziness among some of the people.