LECTURE FOUR

SPELLING CORRECTION

4.1 Introduction

- Two principal uses
  - Correcting document(s) being indexed
  - Correcting user queries to retrieve "right" answers
- Two main flavors:
  - Isolated word
    - Check each word on its own for misspelling
    - Will not catch typos resulting in correctly spelled words
    - e.g., ***from*** → ***form***
  - Context-sensitive
    - Look at surrounding words,
    - e.g., ***I flew form Heathrow to Narita.***

We begin by examining two techniques for addressing isolated-term correction: edit distance, and k-gram overlap. We then proceed to context-sensitive correction.

4.2 Edit distance

Given two character strings s1 and s2, the edit distance between them is the minimum number of edit operations required to transform s1 into s2.

The edit operations allowed for this purpose are:

(i) insert a character into a string;

(ii) delete a character from a string

(iii) replace a character of a string by another character;

Edit distance is sometimes known as *Levenshtein distance*. For example, the edit distance between cat and dog is 3.

For example, that we wanted to compute the edit distance between “Zeil” and “trials”. in converting “Zeil” to “trials”, we start by forming a table of the cost (edit distance) to convert “” to “”, “t”, “tr”, “tri”, etc.:

		t	**r**	**i**	**a**	**l**	**s**
	0	1	2	3	4	5	6

In other words, we need 0 steps to convert “” to “”, 1 to convert “” to “t”, 2 to convert “” to “tr”, and so on.

Next, we add a row to describe the cost of converting “Z” to “”, “t”, “tr”, …, “trials”:

		t	**r**	**i**	**a**	**l**	**s**
	0	1	2	3	4	5	6
Z	1	1	2	3	4	5	6

OK, clearly we need 1 step to convert “Z” to “”. How are the other entries in this row computed? Let's back up just a bit:

		t	**r**	**i**	**a**	**l**	**s**
	0	1	2	3	4	5	6
Z	1	?					

What's the minimum cost to convert “Z” to “t”? It's the smallest of the three values computed as

Add

1 plus the cost of converting “Z” to “” (we get this cost by looking to the left one position).

Remove

1 plus the cost of converting “” to “t”, giving “tZ” (we get this cost by looking up one position).

Change

1 (because “Z” and “t” are different characters) plus the cost of converting “” to “” (we get this cost by looking diagonally up and to the left one position). The last of these yields the minimal distance: 1.

		t	**r**	**i**	**a**	**l**	**s**
	0	1	2	3	4	5	6
Z	1	1	?				

What's the minimum cost to convert “Z” to “tr”? It's the smallest of the three values computed as

Add

1 plus the cost of converting “Z” to “t” (we get this cost by looking to the left one position).

Remove

1 plus the cost of converting “” to “tr”, giving “trZ” (we get this cost by looking up one position).

Change

1 (because “Z” and “t” are different characters) plus the cost of converting “” to “t” (we get this cost by looking diagonally up and to the left one position). The last of these yields the minimal distance: 2.

		t	**r**	**i**	**a**	**l**	**s**
	0	1	2	3	4	5	6
Z	1	1	2	3	4	5	6

And we add the next row, using the same technique:

		t	**r**	**i**	**a**	**l**	**s**
	0	1	2	3	4	5	6
Z	1	1	2	3	4	5	6
e	2	2	2	3	4	5	6

The row after that becomes a bit more interesting. When we get this far:

		t	**r**	**i**	**a**	**l**	**s**
	0	1	2	3	4	5	6
Z	1	1	2	3	4	5	6
e	2	2	2	3	4	5	6
i	3	3	3	?			

We are looking at the cost of converting “Zei” to “tri”. It's the smallest of the three values computed as

Add

1 plus the cost of converting “Zei” to “tr” (we get this cost by looking to the left one position).

Remove

1 plus the cost of converting “Ze” to “tri”, giving “trii” (we get this cost by looking up one position).

Change

Zero (because “i” and “i” are the same character) plus the cost of converting “Ze” to “tr” (we get this cost by looking diagonally up and to the left one position).

The last of these yields the minimal cost of 2.

		t	**r**	**i**	**a**	**l**	**s**
	0	1	2	3	4	5	6
Z	1	1	2	3	4	5	6
e	2	2	2	3	4	5	6
i	3	3	3	2	?		

And then we can fill out the rest of the row:

		t	**r**	**i**	**a**	**l**	**s**
	0	1	2	3	4	5	6
Z	1	1	2	3	4	5	6
e	2	2	2	3	4	5	6
i	3	3	3	2	3	4	5

And finally, the last row of the table:

		t	**r**	**i**	**a**	**l**	**s**
	0	1	2	3	4	5	6
Z	1	1	2	3	4	5	6
e	2	2	2	3	4	5	6
i	3	3	3	2	3	4	5
l	4	4	4	3	3	3	4

Note that this last row, again, has a situation where the cost of a change is zero plus the subproblem cost, because the two characters involved are the same (“l”).

From the lower right hand corner, then, we read out the edit distance between “Zeil” and “trials” as 4.

4.3 k-gram indexes for spelling correction

We will use the k-gram index to retrieve vocabulary terms that have many k-grams in common with the query. Suppose the text is ***november***

- Trigrams are *nov, ove, vem, emb, mbe, ber*.

- The query is ***december***
  - Trigrams are *dec*, *ece*, *cem*, *emb*, *mbe*, *ber*.

So 3 trigrams overlap (of 6 in each term). How can we turn this into a normalized measure of overlap? We require more nuanced measures of the overlap in k-grams between a vocabulary term and q. The linear scan intersection can be adapted when the measure of overlap is the *Jaccard coefficient* for measuring the overlap between two sets A and B, defined to be $|A \cap B|/|A \cup B|$.

Jaccard coefficient measures similarity between sample sets i and j:

$$JC(i,j) = c/(a + b + c)$$

Where:

- c is the number of common elements between i and j
- a is the number of elements exclusive of i
- b is the number of elements exclusive of j
- If i and j share all the elements, JC(i,j) = 1
- If i and j do not share any element, JC(i,j) = 0.

Example 1:

Feature of Fruit	Sphere shape	Sweet	Sour	Crunchy
Object =Apple	Yes	Yes	Yes	Yes
Object =Banana	No	Yes	No	No

The coordinate of Apple is (1,1,1,1) and coordinate of Banana is (0,1,0,0). Because each object is represented by 4

variables, we say that these objects has 4 dimensions. c=1, a=3 and b=0. Jaccard's coefficient between Apple and Banana is 1/4. Jaccard's distance between Apple and Banana is 3/4.

Example 2

Suppose we have two sets A{7, 3, 2, 4, 1}and B{4, 1, 9, 7, 5}. Then the union is and the intersection between two sets is

$$A \cup B = \{1, 2, 3, 4, 5, 7, 9\}$$

And the intersection between two sets is

$$A \cap B = \{1, 4, 7\}$$

Jaccard's coefficient can be computed based on the number of elements in the intersection set divided by the number of elements in the union set.

$$s_{AB} = \frac{|A \cap B|}{|A \cup B|} = \frac{3}{7} = 0.429$$

4.4 Context-sensitive spell correction

Isolated-term correction would fail to correct typographical errors such as *flew form Heathrow*, where all three query terms are correctly spelled. When a phrase such as this retrieves few documents, a search engine may like to offer the corrected query *flew from Heathrow*.

The simplest way to do this is to enumerate corrections of each of the three query terms even though each query term is correctly spelled, then try substitutions of each correction in the phrase.

For the example *flew form Heathrow*, we enumerate such phrases as:

- ***flew from heathrow***
- ***fled form heathrow***
- ***flea form heathrow***

For each such substitute phrase, the search engine runs the query and determines the number of matching results.