605.744: Information Retrieval
Problem Set (Module 2)

Sabbir Ahmed

September 7, 2022

1. (10%) What advantage(s) do balanced binary trees have over hashtables when used to store our dictionary of indexing terms?

   Answer:

2. (30%) Character n-gram overlap is used for both automated spelling correction and name matching (i.e., deciding whether two names might be the same, a common database problem known as "record linkage"). Using a character 3-gram representation, how many distinct n-grams do "CHEONGSONG" and "CHEONMACHONG" have in common? What is the Dice-coefficient score for these two strings using 3-grams? What is the Dice score using 4-grams instead? Which score is higher? Note: although there is nothing conceptually wrong in doing so, for this problem, do not use "padded" n-grams (e.g., '$' or '_' symbols marking the beginning and end of the strings).

   Answer:

   The following are the 3-grams of:

   "CHEONGSONG":

   $$X = \{CHE, HEO, EON, ONG, NGS, GSO, SON\}, |X| = 7$$

   "CHEONMACHONG":

   $$Y = \{CHE, HEO, EON, ONM, NMA, MAC, ACH, CHO, HON, ONG\}, |Y| = 10$$

   Both of the words:

   $$X \cap Y = \{CHE, HEO, EON, ONG\}, |X \cap Y| = 4$$

   The Dice-coefficient is computed with the formula: $2|X \cap Y|/(|X| + |Y|)$. Then,

   $$2(4)/(7 + 10) = 8/17 = 0.47$$

   The following are the 4-grams of:

   "CHEONGSONG":

   $$X = \{CHEO, HEON, EONG, ONGS, NGSO, GSON, SONG\}, |X| = 7$$

”CHEONMACHONG”:

$Y = \{CHEO, HEON, EONM, ONMA, NMAC, MACH, ACHO, CHON, HONG\}, |Y| = 9$

Both of the words:

$$X \cap Y = \{CHEO, HEON\}, |X \cap Y| = 2$$

The Dice-coefficient is: $2|X \cap Y|/(|X| + |Y|)$. Then,

$$2(2)/(7 + 9) = 4/16 = 0.25$$

The Dice-coefficient for 3-grams is higher.

3. (30%) Compute the edit distance (or Levenshtein distance) for these two pairs of strings: (a) ”CHEBYSHEV” and ”TSCHEBYSCHEF”; and (b) ”LEVINSTINE” and ”LEVENSHTEIN”. Then report a sequence of transformations for that cost that converts one string into the other. You should use unit costs for each operation: insertion, deletion, or substitution; that is, each step has a cost of 1. Note, you do not need to write a program or produce any code for this problem - these examples can be easily determined by pencil and paper - you do not need to construct a table as the example in the textbook.

   Trivia: Computing edit distance is a classic dynamic programming problem with an $O(N^2)$ complexity. However, the decision problem ”Do strings x and y (say each of length N) have an edit distance $<= k$ can be solved in $O(kN)$, which is subquadratic. See: Esko Ukkonen's paper 'Algorithms for approximate string matching' (1985).

   Answer:

   (a) ”CHEBYSHEV” and ”TSCHEBYSCHEF”

      i. Insert T: ”_CHEBYSHEV” -> ”TCHEBYSHEV”
      ii. Insert S: ”T_CHEBYSHEV” -> ”TSCHEBYSHEV”
      iii. Insert C: ”TSCHEBYS_HEV” -> ”TSCHEBYSCHEV”
      iv. Substitute F with V: ”TSCHEBYSCHE$\underline{\text{V}}$” -> ”TSCHEBYSCHEF”

      The edit distance is 4.

   (b) ”LEVINSTINE” and ”LEVENSHTEIN”

      i. Substitute I with E: ”LEV$\underline{\text{I}}$NSTINE” -> ”LEVENSTINE”
      ii. Insert H: ”LEVENS_TINE” -> ”LEVENSHTINE”
      iii. Insert E: ”LEVENSHT_INE” -> ”LEVENSHTEINE”
      iv. Delete E: ”LEVENSHTEIN$\underline{\text{E}}$” -> ”LEVENSHTEIN”

      The edit distance is 4.

4. (30%) Following the method described in the textbook (or lecture materials), calculate Soundex codes for the strings: (a) ”Stanford” and (b) ”Georgetown”? Show intermediate steps to produce the final code.

   Answer:

   (a) ”Stanford”

i. Retain the first letter of the name and change all other occurrences of a, e, i, o, u, y, h, w to 0: Stanford -> St0nf0rd
ii. Replace consonants with their corresponding digits: St0nf0rd -> S3051063
iii. Remove all zeros: S3051063 -> S35163
iv. Return the first four positions: S35163 -> S351

(b) ”Georgetown”

i. Retain the first letter of the name and change all other occurrences of a, e, i, o, u, y, h, w to 0: Georgetown -> G00rg0t00n
ii. Replace consonants with their corresponding digits: G00rg0t00n -> G006203005
iii. Remove all zeros: G006203005 -> G6235
iv. Return the first four positions: G6235 -> G623