EXPLORING NLP FUZZY MATCHING ALGORITHMS

Madhurima Nath

Data Scientist, Slalom

WHAT IS FUZZY MATCHING?

FUZZY/APPROXIMATE STRING MATCHING

- Method to find strings which match a pattern approximately.
- Identifies the likelihood/probability that two records are true match based on some parameters.

ID	Name
01	Microsoft
02	Microsoft Co.
03	Microsoft Corporation
04	Mcrosoft Corp
05	Microosoftt

→ Microsoft Corporation

APPLICATIONS OF FUZZY MATCHING

- Spell checker
- Deduplication of records
- Master data management
- Plagiarism detection
- Bioinformatics and DNA sequencing
- Spam filtering
- Content search

HOW DOES FUZZY MATCHING WORK?

FUZZY MATCHING ALGORITHMS

- Edit distance metric - quantifies how dissimilar two strings are by counting the minimum number of operations required to transform one string into the other
  - Levenshtein distance
  - Damerau–Levenshtein distance
- Bitap algorithm - tells whether a given text contains a substring which is "approximately equal" (defined in terms of Levenshtein distance) to a given pattern
- n-gram - predicts next item in a sequence of text (in form of a Markov model)

EDIT DISTANCE METRICS

measures the number of edits needed to transform one word into another

Levenshtein Distance

- most common metric
- techniques for editing:
  - insertion
  - deletion
  - substitution/replacement
- e.g.: LD(irks → risk) = 3

Demerau-Levenshtein Distance

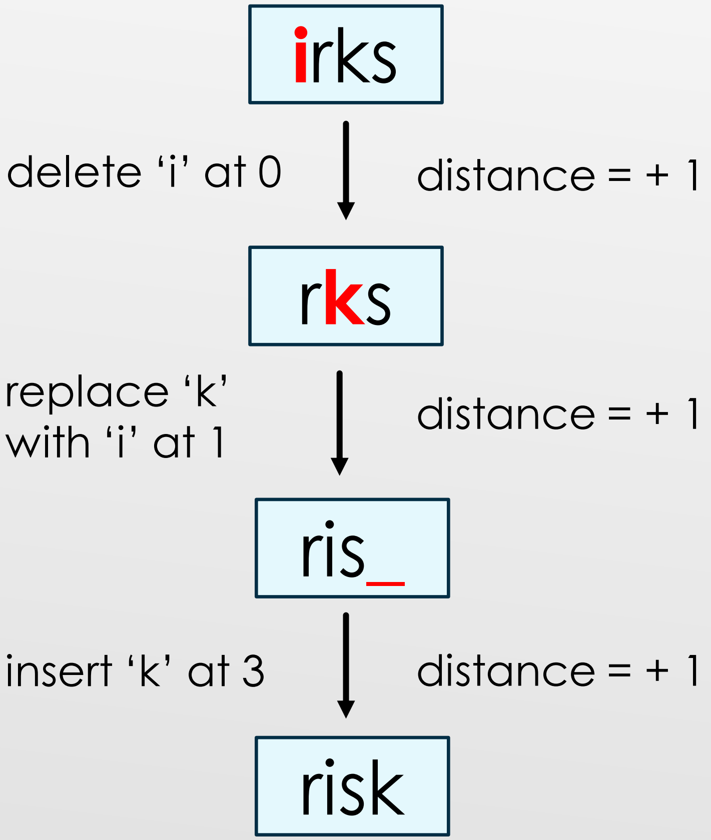
- similar to Levenshtein distance
- techniques for editing:
  - insertion
  - deletion
  - substitution/replacement
  - transposition
- e.g.: DLD(irks → risk) = 2

Levenshtein, Vladimir I. "Binary codes capable of correcting deletions, insertions, and reversals", 1966.
Damerau, Fred J. "A technique for computer detection and correction of spelling errors", 1964.

EDIT DISTANCE METRICS

measures the number of edits needed to transform one word into another

Levenshtein Distance



irks

delete ‘i’ at 0

distance = + 1

rks

replace ‘k’ with ‘i’ at 1

distance = + 1

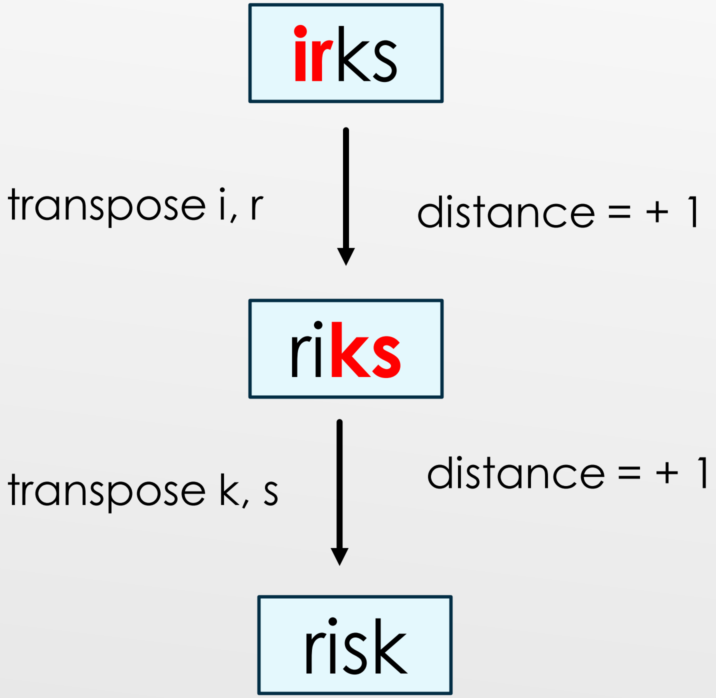
ris_

insert ‘k’ at 3

distance = + 1

risk

Demerau-Levenshtein Distance



irks

transpose i, r

distance = + 1

riks

transpose k, s

distance = + 1

risk

BITAP ALGORITHM

states whether a given text contains a substring which is "approximately equal" (defined by Levenshtein distance) to a given pattern

also known as the shift-or, shift-and or Baeza-Yates–Gonnet algorithm

Input:
Text: womenwhocode
Pattern: code

Output: Pattern found at index: 8

Input:
Text: youareawesome
Pattern: youareamazing

Output: No Match.

https://www.geeksforgeeks.org/java-program-to-implement-bitap-algorithm-for-string-matching/

BITAP ALGORITHM

- Input text and pattern as string
- If length(pattern) = 0 or exceeds length(text),
  return ‘No match’

- Initialize a bit array
- Start a for loop:

N-GRAM ALGORITHM

predicts next item in a sequence of text (in form of a Markov model)

- n-gram: set of values generated from a string by pairing sequentially occurring ‘n’ characters/words
- **goal**: compute probability of a sequence of characters/words or sentence

LET’S DO SOME PROBABILITY

N-GRAM ALGORITHM

predicts next item in a sequence of text (in form of a Markov model)

Conditional probability with 2 variables/words:

$$p(w_1 \cap w_2) = p(w_1)p(w_2|w_1)$$

Conditional probability with 4 variables:

$$p(w_1 \cap w_2 \cap w_3 \cap w_4) = p(w_1)p(w_2|w_1)p(w_3|w_1 \cap w_2)p(w_4|w_1 \cap w_2 \cap w_3)$$

Compute joint probability using chain rule:

$$p(w_1, w_2, \ldots, w_n) = \prod_i p(w_i|w_1 w_2 \ldots\ w_{n-1})$$

N-GRAM ALGORITHM

predicts next item in a sequence of text (in form of a Markov model)

Compute joint probability using chain rule:

$$p(w_1, w_2, \dots, w_n) = \prod_i p(w_i | w_1 w_2 \dots w_{n-1})$$

TOO MANY POSSIBLE COMBINATIONS!!

SOLUTION: **MARKOV APPROACH**

Approximate each component of the product by maximum likelihood estimate

MORE EXAMPLES ON JUPYTER NOTEBOOK

OTHER FUZZY MATCHING ALGORITHMS

- Edit distance
  - Longest common subsequence
  - Hamming distance
  - Jaro distance
- Needleman–Wunsch algorithm
- Smith–Waterman algorithm
- BK Tree metric
- Soundex or Metaphone – phonetic algorithms