TODAY

STRING MATCHING

STRING EDIT DISTANCE

GRAMMARS

REGULAR GRAMMAR = FINITE STATE MACHINE

1. String matching

Problem: Does "which year were you lazy" contain "we"? Where?

Applications: - DNA subsequence matching
- text mining

Example: \( \bar{x}_i = \begin{bmatrix} x_{i1} \\ \vdots \\ x_{iD} \end{bmatrix} \)  \( x_{di} = \) number of occurrences in the \( d \)th word in the document \( i \) of the dictionary

Classify \( \alpha(x) = w_j \) if \( a_j x + b_j > a_k x + b_k + k \)

For \( w_j = "science fiction" \), \( a_j = \begin{bmatrix} 2 \\ 0 \\ 0 \\ 0 \end{bmatrix} \)

rocket
swamp
coven
computer
dragon
love
robot
Algorithm: Boyer-Moore String Matching

\[ x = \text{string} \]
\[ t = \text{text} \]
\[ s = \text{index into } t \]

while \( s < \text{length}(t) - \text{length}(x) + 1 \),

find a matching suffix of length \( n \)

shift forward to line up \( n+1 \) characters

\[ t = \text{theye lowcabwith...} \]

\[ x: \quad abracababa \]

\[ n+1 = 3 \]

shift:

\[ abracababa \]

\[ n+1 = 3 \]

"Good-Suffix Table" — stored in memory

<table>
<thead>
<tr>
<th>( n )</th>
<th>( n+1 ) choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>all</td>
</tr>
<tr>
<td>4</td>
<td>all</td>
</tr>
<tr>
<td>\vdots</td>
<td>\vdots</td>
</tr>
</tbody>
</table>
2. STRING EDIT DISTANCE

Problem: how dissimilar are “store” and “stop”? 

Applications:

1. K-nearest neighbors, e.g., is “glasnost” a German or Russian word?
   A: Find 3 other words that minimize \( D(w, “glasnost”) \)

2. Perzen window: \( p(“glasnost” | \text{Russian word}) \)
   \[ e = \frac{1}{n} \sum_{i=1}^{n} e = D(w, “glasnost”) \]

Method

\( X = [A, g, o, l, d, f, i, s, h, \diamondsuit] \)
\( Y = [A, g, o, u, r, d, \heartsuit] \)
\( \tilde{a} = \text{alignment function}: \text{maps } Y \text{ indices } \rightarrow X \text{ indices} \)

E.g., \( \tilde{a} = \{0, 1, 2, 3, 4, 9\} \) gives

\( \begin{array}{cccccccc}
& G & O & L & D & F & I & S & H & \diamondsuit \\
& G & O & U & D & & & & & \heartsuit \\
\end{array} \)

\( \uparrow R \) \quad "DELETION": element of \( X \) disappears

\( \uparrow "\text{INSERTION}" " \): element of \( Y \) inserted

\( "\text{SUBSTITUTION}" " \): element changes

\( D(X, Y) = \min_{\tilde{a}} \left[ \# \text{Insertions} + \# \text{Deletions} + \# \text{Substitutions} \right] \)
Alignment Matrix:

Let \( C_s = \text{cost of substitution} \)
\( C_d = \text{cost of deletion} \)
\( C_i = \text{cost of insertion} \) \( \) Must be equal if \( D(x, y) \) symmetric

\[
\text{Cost}(i, j) = C_s \left[ x(i) \neq y(j) \right] + \min \left\{ \text{Cost}(i-1, j-1), C_d + \text{Cost}(i-1, j), C_i + \text{Cost}(i, j-1) \right\}
\]

\[
D(x, y) = \text{Cost} \left( \text{length}(x) + 1, \text{length}(y) + 1 \right)
\]

3. Recognizing with Grammars

Problems:

4. Is “cats chase mice” an English sentence?
5. What is \( P(\text{cats chase mice} | \text{English}) \)?

Top-level node: sentence
Non-terminal nodes: noun phrase, verb phrase
Terminal nodes: words

Diagram:

- S
  - NP
    - N: cats
  - VP
    - V: chase
    - N: mice
Sentence length = N

Non-terminals: A, B, C, D, E

Terminals: a, b, c, d, e

Chomsky's Hierarchy of Grammar

<table>
<thead>
<tr>
<th>Level</th>
<th>Type</th>
<th>Alkawate Rules</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Regular</td>
<td>A → aB</td>
<td>VP → &quot;chase&quot; , NP → N , S → &quot;cats&quot; , VP → N , N → &quot;mice&quot;</td>
</tr>
</tbody>
</table>

\[ P(\text{cats chase mice} \mid S) = P_3 P_1 P_2 P_4 = 0.5 \]

Computational Complexity of Recognition = \( \Theta(N^3) \)

Can be re-written as a finite state machine:

\[ \text{cats} / 1.0 \]
\[ \emptyset / 1.0 \]
\[ \text{mice} / 0.5 \]

Level 2

Context-Free Grammar (CFG)

A → BC
A → a

Examples:

\[ S \rightarrow NP, VP \quad P_1 = 1.0 \]
\[ NP \rightarrow N \quad P_2 = 1.0 \]
\[ VP \rightarrow V, N \quad P_3 = 1.0 \]
\[ V \rightarrow \text{chase} \quad P_4 = 1.0 \]
\[ N \rightarrow \text{cats} \quad P_5 = 0.5 \]
\[ N \rightarrow \text{mice} \quad P_6 = 0.5 \]
\[ P(\text{cats chase mice} \mid S) = P_1 P_2 P_3 P_5 P_4 P_6 = 0.25 \]

Computational Complexity of Recognition = \( O(N^{3.5}) \)

**Level 1**

Context-Dependent Grammar

- \( ABC \rightarrow ADC \)
- \( ABC \rightarrow AaC \)

Computational Complexity = \( O(N^{\alpha}) \) for some \( \alpha \)

**Level 0**

Unrestricted Grammar

- \( ABC \rightarrow DEF \)

Computational Complexity = \( O(N^{\beta}) \)

Training a Regular Grammar

Given bracketed training data:

- \( x_1 = \text{cats chase mice} \)
- \( x_2 = \text{dogs chase cats} \)

\[ \hat{P}_{ML} (\text{dogs, VP} \mid S) = \frac{N(S \rightarrow \text{dogs, VP})}{N(S)} = \frac{1}{Z} \]

\[ \hat{P}_{ML} (\text{chase, NP} \mid \text{VP}) = \frac{N(\text{VP} \rightarrow \text{chase, NP} \mid \text{VP})}{N(\text{VP})} = \frac{2}{Z} \]