Outline

- Announcements
- Distance Metrics
Announcements

- Project preliminary report is due 10/2
- First quiz will be released on Canvas this week
- Homework assignment 1 announcement
Homework Assignment 1

• Educational objectives

• Issues in optimizing/modifying solution based on auto-grader score

• Approaches to checking/testing your solution before submitting

• Importance of documentation
Homework Assignment 1

**Educational objectives**

- Develop an understanding of the complexities and subtleties of natural language
- Develop an understanding of the connection between Natural Language “Processing” and Natural Language “Understanding”
- Identify a range of tools and related tasks in NLP
- Learn about the limitations of rule-based systems (more data -> more special cases) and the need for data-driven ML techniques
Homework Assignment 1

- Issues in optimizing/modifying solution based on auto-grader score
  - Type of output/feedback provided by autograder
  - A single numeric is not informative enough to know the reason(s) behind deviation from expected solution esp. when the task involves many cases and conditions to test for
  - A trial and error approach can lead to improved score on submission k but lowered score on submission k+1
  - If score is found to be “high enough,” may not invest in improving solution, thinking about the problem, etc.
  - What is a more principled way to test your solution?
Homework Assignment 1

• Approaches to testing your solution
  • Construct a small file containing a number of test sentences where you manually make changes according to the requirements and compare this file with your program output
  • Sample from provided input files to construct a new test file and compare gold replacements (manual replacements) with your program output
  • Compare your output to output obtained by colleagues who have a similar understanding of the problem requirements
• Importance of documentation!
Distance Metrics

• How similar are two strings?
• Hamming distance
• Levenshtein distance
• Min Edit distance
• Other metrics
Hamming distance

• e.g., how similar are the two strings 01101010 and 11011011?

• Hamming distance = number of positions where the two string differ

• What is the hamming distance between flaw and lawn?

• What is the time complexity for computing hamming distance?

• What is the lower/upper bound on the hamming distance between two strings of lengths $n$ and $m$?
Comparing Documents

• How to compute the distance between two documents?
Distance between documents

• How to compute the distance between two documents?
  • Construct a “common” vocabulary space
  • Represent each document as a bag-of-words
  • Apply a distance metric to the two doc vectors
• How to construct the vocabulary space?
  • Types vs. Tokens
  • Stemming
  • IR approach
Hamming distance

• What is the hamming distance between flaw and lawn?

• But are “flaw” and “lawn” 100% different?

• Limitations of hamming distance

• Levenshtein distance

  • minimum number of single-character edits (insertions, deletions or substitutions) required to change one word into the other

  • e.g., flaw —> lawn (Levenshtein distance = 2)

• What is the lower bound on Levenshtein distance?
Distance Metrics

• How similar are two strings?

  • spell correction, word/document similarity, machine translation
Edit Distance

- The minimum edit distance between two strings is the minimum number of editing operations needed to transform one string into the other.

- Editing operations include:
  - Insertion
  - Deletion
  - Substitution (in some references, the cost for the substitution operation is 2 x cost of insertion or deletion)
String edit distance

• How many letter changes to map A to B
• Substitutions
  – EXAMP E L
  – EXAMP LE 2 substitutions
• Insertions
  – EXA P LE
  – EXAMP LE 1 insertion
• Deletions
  – EXAMMP LE
  – EXA _ MPLE 1 deletion
Edit Distance

• Note about alignment

• How to find min edit distance?

• For two strings X, Y of length n, m, respectively
  • D(i,j) the edit distance between X[1..i] and Y[1..j]
  • D(n,m) is the edit distance between X and Y
Min Edit Distance Computation

- Dynamic programming
- Solving a problem by combining solutions to subproblems
- Bottom-up approach:
  - For small $i, i$, compute $D(i,j)$
  - Combine solutions to compute $D(i,j)$ for larger $i, j$
Min Edit Distance Computation

- initialization: \( D(i,0) = i, \ D(0,j) = j \)

<table>
<thead>
<tr>
<th></th>
<th>9</th>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>4</th>
<th>6</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>E</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>P</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>M</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>M</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>A</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>X</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>E</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>#</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>
String Edit Distance

function MIN-EDIT-DISTANCE(target, source) returns min-distance

\[ n \leftarrow \text{LENGTH}(target) \]
\[ m \leftarrow \text{LENGTH}(source) \]
Create a distance matrix distance\[n+1,m+1]\]
\[ \text{distance}[0,0] \leftarrow 0 \]
for each column \( i \) from 0 to \( n \) do
   for each row \( j \) from 0 to \( m \) do
      distance\[i,j]\leftarrow\text{MIN}(\)
      distance\[i−1,j]\+\text{ins-cost}(target\_j),
      distance\[i−1,j−1]\+\text{subst-cost}(source\_j, target\_i),
      distance\[i,j−1]\+\text{ins-cost}(source\_j))

Figure 5.5 The minimum edit distance algorithm, an example of the class
of dynamic programming algorithms.
Other metrics?

- **String** distance metric
  - Hamming distance
  - Levenshtein distance
  - Min Edit distance
  - similarity based metrics (cosine similarity)
- **numeric** similarity metrics (in vector space, e.g., cosine similarity)
Three Spelling Problems

1. Detecting isolated non-words
   "graffe" "exampel"

2. Fixing isolated non-words
   "graffe" $\Rightarrow$ "giraffe" "exampel" $\Rightarrow$ "example"

3. Fixing errors in context
   "I ate desert" $\Rightarrow$ "I ate dessert"
   "It was written be me" $\Rightarrow$ "It was written by me"

Next Class
Probability model

• Most likely word given observation
  – Argmax \( P(W \mid O) \)

• By Bayes Rule is equivalent to
  – Argmax \( \frac{P(W)P(O \mid W)}{P(O)} \)

• Which is equivalent to
  – Argmax \( P(W)P(O \mid W) \) (denom is constant)

• \( P(O \mid W) \) calculated from edit distance

• \( P(W) \) calculated from language model