Class 21

October 31, 2022

Contents

Template Matching 1
String Matching 1

Template Matching

String Matching

- Use `Hamming Distance` metric to measure similarity between strings.

Take example: `PUNJAB` and `APUNJAB`. There's a huge hamming distance between these two (but they are actually very similar).

So we should use a better distance metric. Instead we check how many steps are needed to convert one word to another:

For example for converting `PUNJAB` to `APUNJAB` we only need 1 operation (1 insertion).

This new distance is known as **Edit Distance**.

We want to find an efficient way of doing this.

- Allowed operations for Edit Distance
  - Add, Remove, Replace Characters
  - Cost of these operations are the hyperparameters for the algorithm
  - Assume add and remove have cost 1
  - replace has cost 0 (if both characters same) and 1 otherwise
- Find the optimal set of operations

There's an algorithm for this known as **Dynamic Time Warping**.

Assume the two strings to be $p_1p_2p_3 \dots p_k$ and $q_1q_2q_3 \dots q_r$.

We know that cross matching in characters is not allowed. So if we have a matching in the two strings we know that the substrings left of the matching (will match together) and substrings right of this matching (will map together).

So we can create an optimal algorithm.