Dynamic Programming in Speech

CSC401/2511 2024 A3 Tutorial 02. Presenter: Ken Shi

Word Error Rate

For ASR Evaluation

Levenshtein Distance

In its origin:

The Levenshtein distance between two strings a, b (of length $|a|$ and $|b|$ respectively) is given by $\operatorname{lev}(a, b)$ where

$$\operatorname{lev}(a,b) = \begin{cases} |a| & \text{if } |b| = 0, \\ |b| & \text{if } |a| = 0, \\ \operatorname{lev}\left(\operatorname{tail}(a), \operatorname{tail}(b)\right) & \text{if } \operatorname{head}(a) = \operatorname{head}(b), \\ 1 + \min \begin{cases} \operatorname{lev}\left(\operatorname{tail}(a), b\right) \\ \operatorname{lev}\left(a, \operatorname{tail}(b)\right) \\ \operatorname{lev}\left(\operatorname{tail}(a), \operatorname{tail}(b)\right) \end{cases} & \text{otherwise} \end{cases}$$

UNIVERSITY OF TORONTO

Levenshtein in Word Error Rate (WER)

- Two strings: Reference Sentences (Groundtruth) and Hypothesis Sentences (Transcript)
- Four states describing a pair of words, one from each:
  - Match: Two words are the same
  - Insertion: A new hypothesis word added to the sequence
  - Deletion: An existing reference word missing
  - Substitution: A complete replacement of the word
- WER formula:

$$WER = \frac{\#\text{Insertions} + \#\text{Deletions} + \#\text{Substitutions}}{\#\text{ReferenceWords}}$$

UNIVERSITY OF TORONTO

WER Example

Consider this sentence pair:

- Reference: how to recognize speech
- Hypothesis: how to wreck a nice beach

		How	To	Wreck	A	Nice	Beach
How							
To							
Recognize							
Speech							

UNIVERSITY OF TORONTO

WER Example

Substitution / Match	Deletion
Insertion	Destination

Consider this sentence pair:

- Reference: how to recognize speech
- Hypothesis: how to wreck a nice beach

		How	To	Wreck	A	Nice	Beach
	0	1	2	3	4	5	6
How	1	0	1	2	3	4	5
To	2	1	0	1	2	3	4
Recognize	3	2	1	1	2	3	4
Speech	4	3	2	2	2	3	4

S = 2
I = 2
D = 0
WER = (2 + 2 + 0) / 4
= 100%!

UNIVERSITY OF TORONTO

Recall: A Monotonic Forward Algorithm

Remember this from the lecture?

Function monotonic_forward
Inputs $a = a_1, a_2, \ldots, a_U$ **and** $b = b_1, b_2, \ldots, b_T$
1: **Define** $table[0 \ldots U, 0 \ldots T]$
2: initialize($table[0 \ldots U, 0]$, $table[0,1 \ldots T]$)
3: **For each** u **in** $1 \ldots U$:
4: **For each** t **in** $1 \ldots T$:
5: $table[u,t] = \text{step}(a_u, b_t, table[u-1,t-1], table[u-1,t], table[u,t-1])$
6: **Return** finalize($table[0 \ldots U, 0 \ldots T]$)

Your Job

- Implement initialize, step and finalize
  - Initialize: defines the table, initializes the helper row/column
  - Step: forward calculate values of the table
  - Finalize: backtrace which operation has been done through the iteration and computes WER

UNIVERSITY OF TORONTO

Practical Tip: Priority Operations

- What happen when there is a tie?
- Convention: Match > Substitution > Insertion > Deletion
- It is really a choice, but for the purpose of the assignment, we want you to follow this particular convention.

UNIVERSITY OF TORONTO

Your Task

- WER is used as a measure to evaluate different ASR system (in this case, Kaldi and Google)
- You shouldn't have to modify the main function to see how it is used in action (provided that your implementation is correct)
- Please follow the instruction in the assignment sheet carefully, as our test may look into every component of your function to evaluate your operations

- Interesting thing to think about: Does the cost has to be one across the board? What if they don't?

UNIVERSITY OF TORONTO

Dynamic Time Warping

In Speaker Verification

UNIVERSITY OF TORONTO

Task: Speaker Verification

- Given: Audio samples of somebody saying a specific line/word
- Goal: Identify if the samples are coming from the same speaker.

- Approach: Dynamic Time Warping!

UNIVERSITY OF TORONTO

Dynamic Time Warping (DTW)

- Given: two temporal sequence X and Y
- Goal: find the **optimal warping path** that minimizes the cost that the path take

- Approach: Use DP to maintain an Accumulative Cost Matrix

- Three Conditions (Constraints):
  - **Boundary condition**: start by pairing the first elements from both, end by pairing the last elements from both.
  - **Monotonicity condition**: every step must push forward
  - **Step size condition**: increment in forward should be bounded by certain range

UNIVERSITY OF TORONTO

DTW in Speaker Verification

- Follows the exact recipe like WER, but instead of dealing with strings, we deal with MFCCs.
- The above should automatically maintain Boundary condition, Monotonicity condition
- The above should also make the step size 1 (upper, left or upper left)
- In terms of cost, we will use the **euclidean distance** between each MFCC frame
- For the specific usage, see example from Lecture Slides!

UNIVERSITY OF TORONTO

A few tips

- The main function is written for you. It chops a few voice segments from the dataset and runs the algorithm. You should be able to tell what is the correct answer just by looking at how the segments are loaded.
- The default test given to you should give you an intuitive result if it does work.
- You are welcomed to investigate further by playing around with different segments you find or listen to segments of the specified time stamp

- Obviously this is not a perfect algorithm. Based on lecture material, can you think of why it could sometimes underperform? Does it have to do with how this task is set up?

UNIVERSITY OF TORONTO

Small Examples

Levenshtein and DTW

UNIVERSITY OF TORONTO

Running Example - Levenshtein

- A: [1, 3, 4]
- B: [1, 2, 3]
- Levenshtein:
  - Solutiin 1:
    - Replace 3 with 2, A is now [1, 2, 4]
    - Replace 4 with 3, A is now [1, 2, 3]
  - Solution 2:
    - Insert 2, A is now [1, 2, 3, 4]
    - Delete 4, A is now [1, 2, 3]

Running Example - DTW

- A: [1, 3, 4]
- B: [1, 2.1, 3]
- DTW:
  - First match: 1 to 1, cost 0
  - Second match: 3 in A to [2, 3] in B, cost is (3 – 2.1) ** 2 = 0.81
  - Third match: 4 in A to 3 in B, cost is (4 – 3) ** 2 = 1

UNIVERSITY OF TORONTO