KECE470 Pattern Recognition

Chapter 8. Template Matching

Chang-Su Kim

Many slides are modified from Serigos Theodoridis's own notes.

Template Matching

Given a set of reference patterns, known as templates, determine which one matches a test pattern best.

Note: each class is represented by a single typical pattern

Ex) aplke (test pattern)

Distance between Sequences

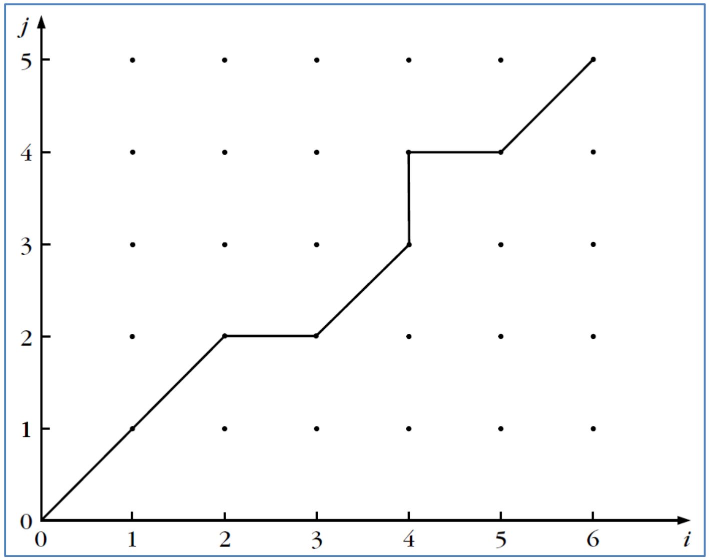
- Reference pattern: $\boldsymbol{r}(i), i = 1, \dots, I$
- Test pattern: $\boldsymbol{t}(j), j = 1, \dots, J$
- Path:

$$(0,0) = (i_0, j_0), (i_1, j_1), \dots, (i_f, j_f) = (I, J)$$

- Path cost

$$D = \sum_{k=1}^{f} d(i_k, j_k | i_{k-1}, j_{k-1})$$

- The distance is defined as $D_{\min}$ over all path



Bellman's Optimality Principle

- Optimal Path

$$(i_0, j_0) \xrightarrow{\text{opt}} (i_f, j_f)$$

- Optimal path constrained to pass through (i, j)

$$(i_0, j_0) \xrightarrow{\text{opt s.t } (i,j)} (i_f, j_f)$$

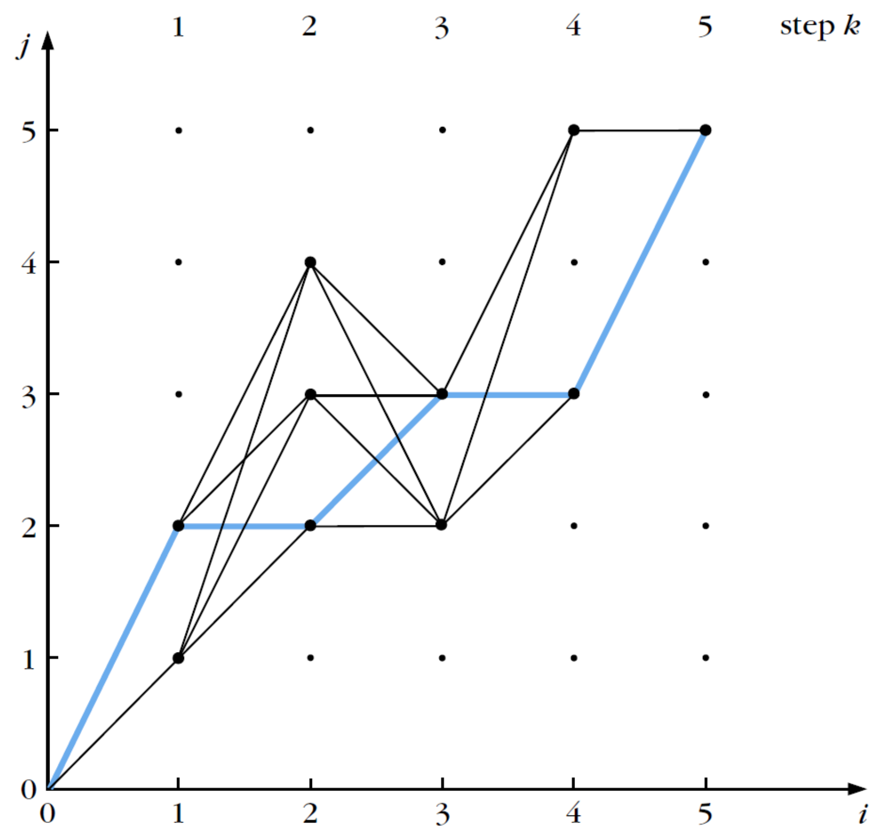
- Bellman's principle

$$(i_0, j_0) \xrightarrow{\text{opt s.t } (i,j)} \left(i_f, j_f\right) = (i_0, j_0) \xrightarrow{\text{opt}} (i, j) \oplus (i, j) \xrightarrow{\text{opt}} \left(i_f, j_f\right)$$

Dynamic Programming

$$D_{\min}(i_k, j_k) = \min_{i_{k-1}, j_{k-1}} D_{\min}(i_{k-1}, j_{k-1}) + d(i_k, j_k | i_{k-1}, j_{k-1})$$

- Local constraints: there is a set of allowed predecessors.



Dynamic Programming

Example 8.1

Figure 8.3 shows the optimal paths (black lines) to reach the nodes at step $k=3$ starting from the nodes at step $k=0$. The grid contains three nodes per step. Only the optimal paths, up to step $k=3$, have been drawn. The goal of this example is to extend the previous paths to the next step and compute the optimal paths terminating at the three nodes at step $k=4$. Bellman's principle will be employed. Assume that the accumulated costs of the optimal paths $D_{\min}(3, j_3), j_3 = 0, 1, 2$ at the respective nodes are:

$$D_{\min}(3,0) = 0.8, D_{\min}(3,1) = 1.2, D_{\min}(3,2) = 1.0 \tag{8.2}$$

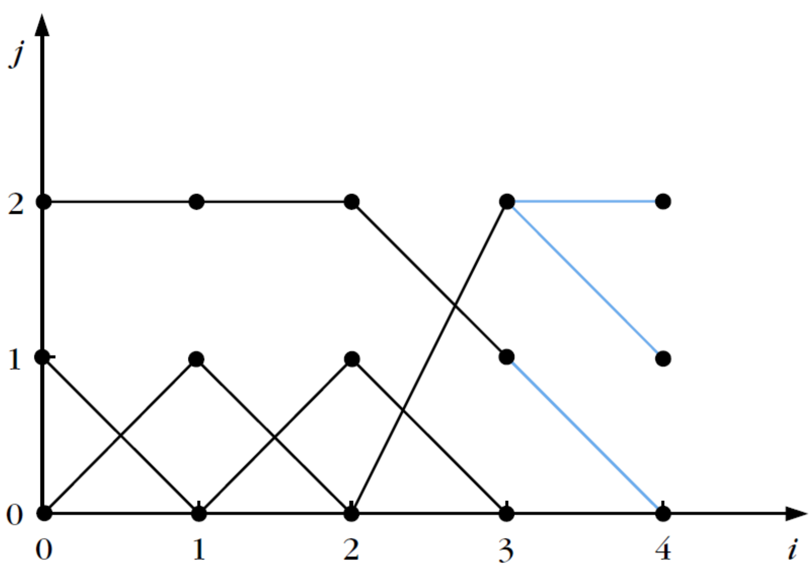


Table 8.1 Transition Costs Between Nodes for the Example 8.1

Nodes	(4, 0)	(4, 1)	(4, 2)
(3, 0)	0.8	0.6	0.8
(3, 1)	0.2	0.3	0.2
(3, 2)	0.7	0.2	0.3

Edit Distance

- Typing errors for "beauty"
  - Wrongly identified symbol (e.g. "befuty")
  - Insertion error (e.g. "bearuty")
  - Deletion error (e.g. "beuty")

- Edit distance between two strings A and B

$$D(A,B) = \min_{j}[C(j) + I(j) + R(j)]$$

  - Minimum total number of changes, insertions, and deletions
  - e.g.
    - beuty $\xrightarrow{I}$ beauty
    - beuty $\xrightarrow{C}$ beaty $\xrightarrow{I}$beauty
    - beuty $\xrightarrow{D}$ bety $\xrightarrow{I}$ beuty $\xrightarrow{D}$ bety $\xrightarrow{I}$ beuty $\xrightarrow{D}$ bety $\xrightarrow{I}$ beuty $\xrightarrow{I}$ beauty

Edit Distance

- Delete

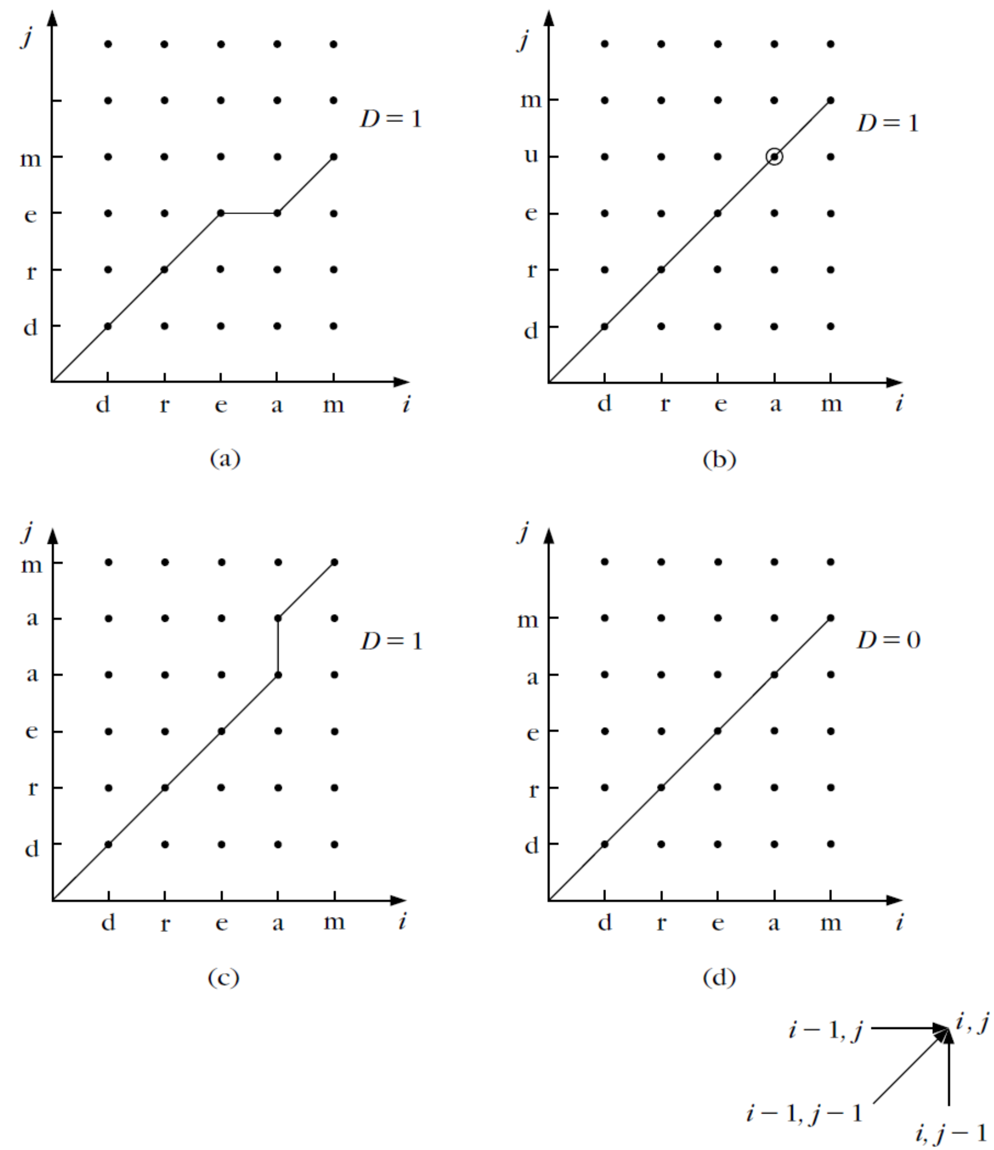
$$d(i,j|i,j-1) = 1$$

- Insertion

$$d(i,j|i-1,j) = 1$$

- Diagonal transition

$$d(i,j|i-1,j-1) = \begin{cases} 0 & r(i) = t(j) \\ 1 & r(i) \neq t(j) \end{cases}$$



Edit Distance (Algorithm)

- Initialization
  - $D(0,0) = D(0,*) = D(*,0) = 0$
- For $i = 1$ to I
  - For $j = 1$ to J

$$D(i,j) = \min \begin{Bmatrix} D(i-1,j-1) + d(i,j|i-1,j-1) \\ D(i-1,j) + 1 \\ D(i,j-1) + 1 \end{Bmatrix}$$

  - End For
- End For
- Output $D(I,J)$