CS 4248
Natural Language Processing

Professor NG Hwee Tou

Department of Computer Science
School of Computing
National University of Singapore
nght@comp.nus.edu.sg

Spelling Errors

- Three increasingly broader problems:
  - Non-word error detection
    - E.g., detecting *graffe* (misspelling of *giraffe*)
  - Isolated-word error correction
    - Consider a word in isolation
    - E.g., correcting *graffe* to *giraffe*
  - Context-sensitive error detection and correction
    - Use of context to detect and correct spelling errors
    - Real-word errors
    - *there* for *three*, *dessert* for *desert*, *piece* for *peace*

Spelling Error Patterns

- Most misspelled words in human typewritten text are single-error misspellings
- Single-error misspellings:
  - Insertion: mistyping *the* as *ther*
  - Deletion: mistyping *the* as *th*
  - Substitution: mistyping *the* as *thw*
  - Transposition: mistyping *the* as *hte*

Spelling Error Correction

- Assume:
  - Detect and correct non-word spelling errors
  - No use of context
- Two steps:
  - Propose candidates of correctly spelled words
  - Score the candidates

Spelling Error Correction

- Proposing candidates of correctly spelled words:
  - Need a large dictionary of words (correctly spelled)
  - Assume single-error misspellings (one insertion, deletion, substitution, or transposition)

Spelling Error Correction

Example:

Error	Correction	Type (w.r.t. correction)
acress	actress	deletion
acress	cress	insertion
acress	caress	transposition
acress	access	substitution
acress	across	substitution
acress	acres	insertion
acress	acres	insertion

Spelling Error Correction

- Example:
  - Given a mis-spelled word *acress* (observation o)
  - Determine whether the correct word is *actress*, *cress*, or *acres* (class c_1, c_2, c_3)
  - Use Bayesian classification

Bayesian Classification

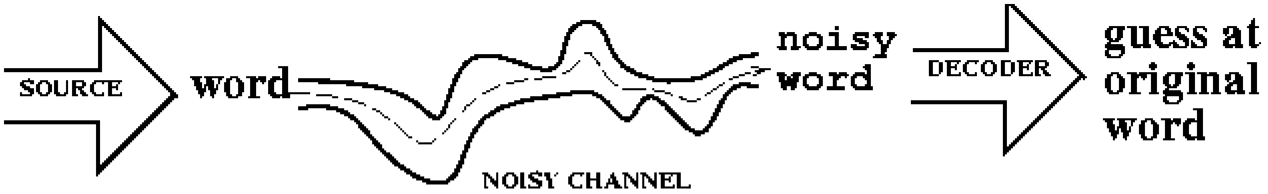
- Problem: Given an observation *o* and a set of classes *C*, infer the class $c \in C$ that *o* belongs to
- Choose the class *c* which is the most probable given the observation *o*

$$\hat{c} = \underset{c \in C}{\arg\max}\, P(c \mid o)$$

Bayesian Classification

$$
\begin{aligned}
\hat{c} &= \arg\max_{c \in C} P(c \mid o) \\
&= \arg\max_{c \in C} \left\{ \frac{P(o \mid c) \cdot P(c)}{P(o)} \right\} \qquad \text{Bayes' Theorem} \\
&= \arg\max_{c \in C} \left\{ \underbrace{P(o \mid c)}_{\text{likelihood}} \cdot \underbrace{P(c)}_{\text{prior}} \right\}
\end{aligned}
$$

Noisy Channel Model



Spelling Error Correction

- Scoring the candidates:
  - Need a corpus of annotated text where the misspelled words are identified and labeled with the correctly spelled words
  - Supervised machine learning
  - Gather the probability estimates from the annotated corpus

$$\hat{c} = \arg\max_{c \in C} \left\{ P(o \mid c) \cdot P(c) \right\}$$

Estimating the Prior *P(c)*

$$P(c) = \frac{C(c)}{N}$$

Maximum Likelihood Estimate (MLE)

c	C(c)
actress	1343
cress	0
caress	4
access	2280
across	8436
acres	2879

Smoothing

c	C(c)	C(c) + 1
actress	1343	1344
cress	0	1
caress	4	5
access	2280	2281
across	8436	8437
acres	2879	2880

Smoothing

$$P(c) = \frac{C(c) + \lambda}{N + B \cdot \lambda}$$

λ: small positive number

C(c): Frequency of c in training corpus

N: Total num of c's

B: Num of bins the c's are divided into (i.e., num of distinct c's)

$$N = C(c_1) + C(c_2) + \cdots + C(c_B)$$

$$N + B \cdot \lambda = \{C(c_1) + \lambda\} + \{C(c_2) + \lambda\} + \cdots + \{C(c_B) + \lambda\}$$

$$1 = \frac{C(c_1) + \lambda}{N + B \cdot \lambda} + \frac{C(c_2) + \lambda}{N + B \cdot \lambda} + \cdots + \frac{C(c_B) + \lambda}{N + B \cdot \lambda}$$

Smoothing

λ = 1 Laplace's law
(add-one smoothing)

$$P(c) = \frac{C(c) + 1}{N + B}$$

λ = 0.5 Jeffreys-Perks law

$$P(c) = \frac{C(c) + 0.5}{N + B \cdot 0.5}$$

Estimating the Likelihood *P(o|c)*

- Use letter frequencies in annotated training corpus to approximate *P(o|c)*
- sub[m, l]: # times the correct letter l was typed as m
- trans[k, l]: # times the correct letter sequence kl was typed as lk
- ins[l, m]: # times the extraneous letter m was inserted after l
- del[k, l]: # times the letter l was deleted from the correct letter sequence kl

Estimating the Likelihood *P(o|c)*

$$P(o \mid c) = \begin{cases} \dfrac{sub[m,l]}{C(l)} & \text{if substitution} \\ \dfrac{trans[k,l]}{C(kl)} & \text{if transposition} \\ \dfrac{ins[l,m]}{C(l)} & \text{if insertion} \\ \dfrac{del[k,l]}{C(kl)} & \text{if deletion} \end{cases}$$

Minimum Edit Distance

- Multiple-error misspellings
- Minimum edit distance between 2 strings:

  The minimum number of editing operations (insertion, deletion, substitution) needed to transform one string into another

Minimum Edit Distance

Alignment	o	f	ε	ε
	ε	f	o	r

Operations

o f

delete o →

f

keep f →

f

insert o →

f o

insert r →

f o r

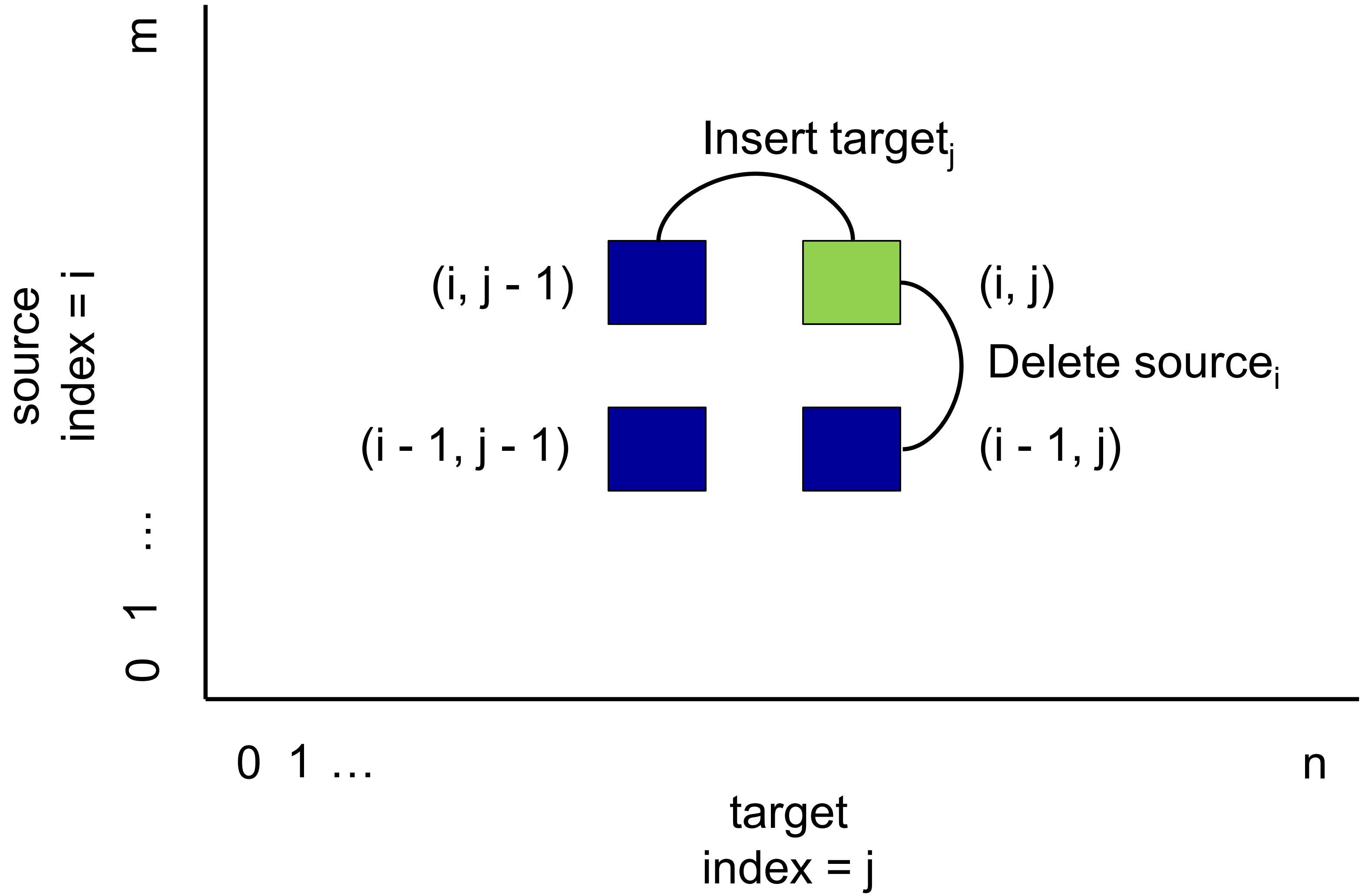
Minimum Edit Distance

- Cost of operation:
  - Insertion: 1
  - Deletion: 1
  - Substitution: 2 if the two characters are different; else 0
- Compute minimum edit distance by dynamic programming

Minimum Edit Distance

function min-edit-distance(source, target) **returns** min-distance
m ← length(source)
n ← length(target)
create a matrix d[m + 1, n + 1]
d[0, 0] ← 0
for each row i **from** 1 **to** m **do**
    d[i, 0] ← d[i – 1, 0] + del-cost(source$_i$)
for each column j **from** 1 **to** n **do**
    d[0, j] ← d[0, j – 1] + ins-cost(target$_j$)
for each row i **from** 1 **to** m **do**
    for each column j **from** 1 **to** n **do**
        d[i, j] ← min(d[i – 1, j] + del-cost(source$_i$),
                d[i, j – 1] + ins-cost(target$_j$),
                d[i – 1, j – 1] + subst-cost(source$_i$, target$_j$))
return d[m, n]

Minimum Edit Distance



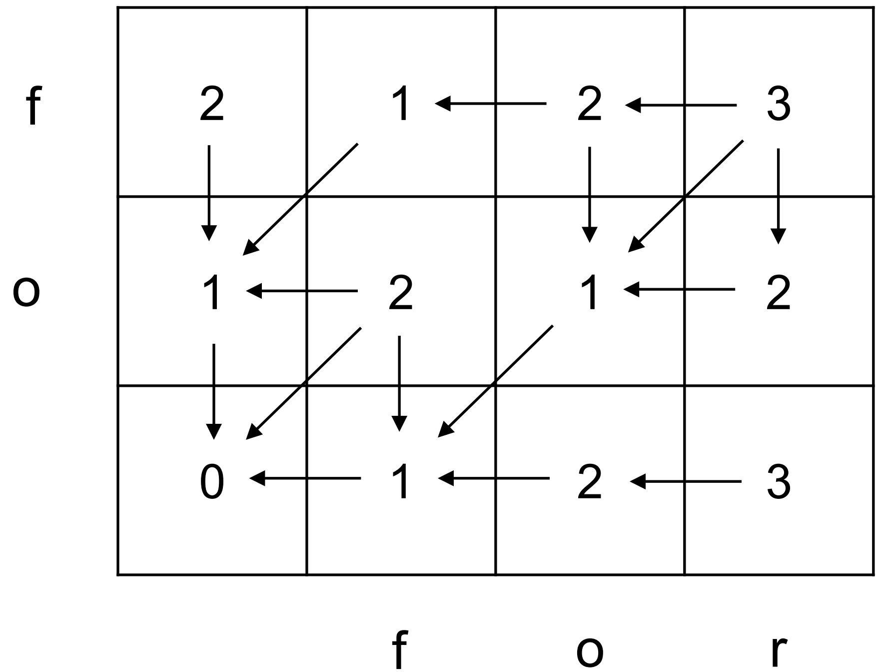
Minimum Edit Distance

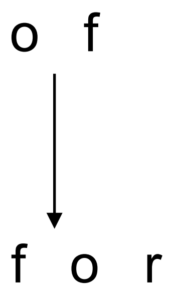
f				
o				
		f	o	r

o f

f o r

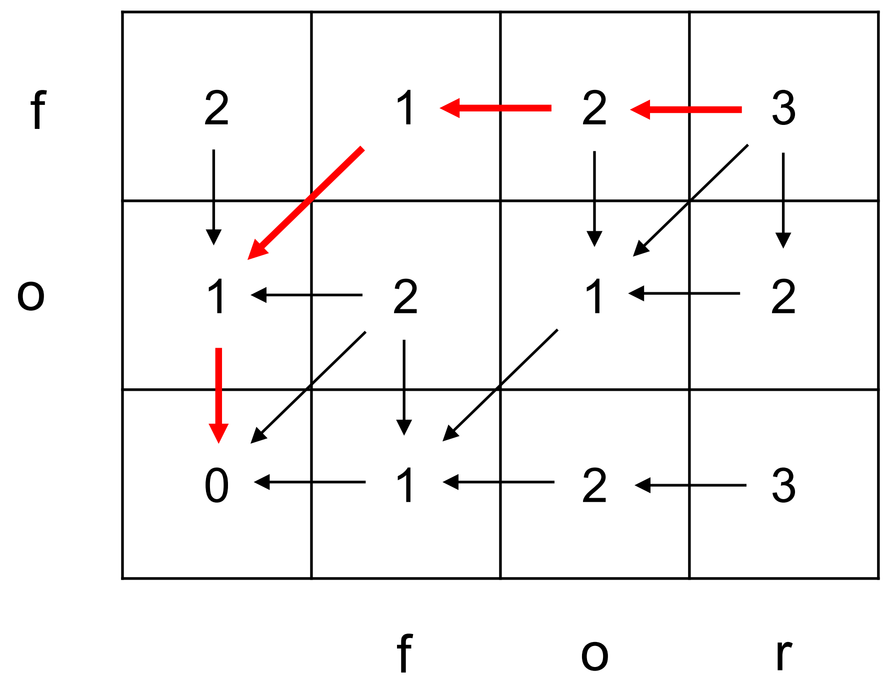
Minimum Edit Distance

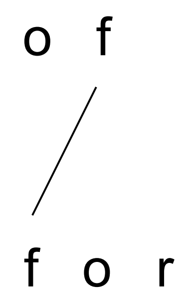




Time complexity: $O(mn)$

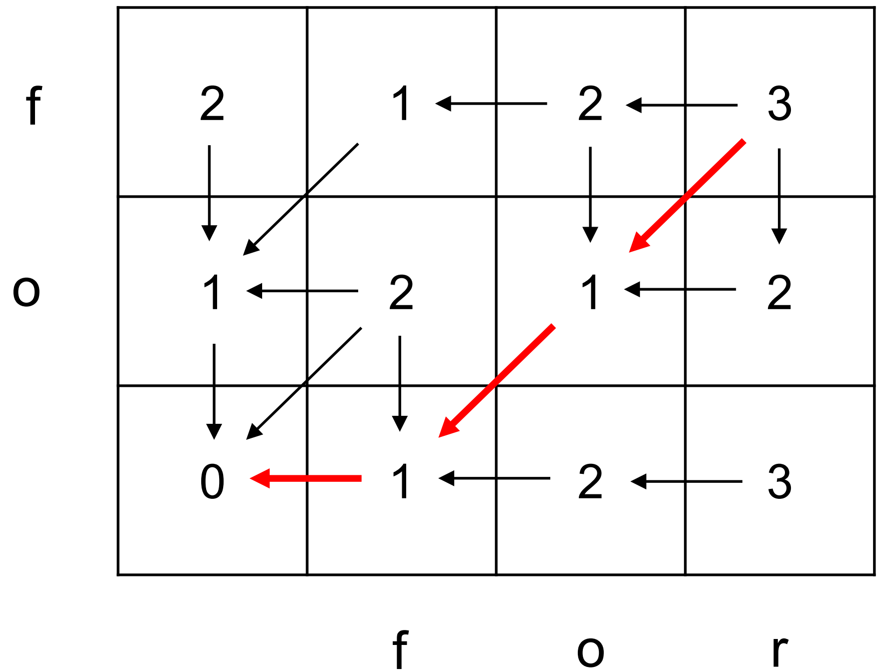
Minimum Edit Distance

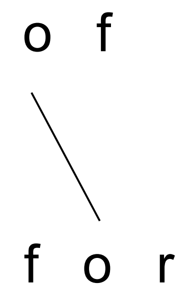




delete o
keep f
insert o
insert r

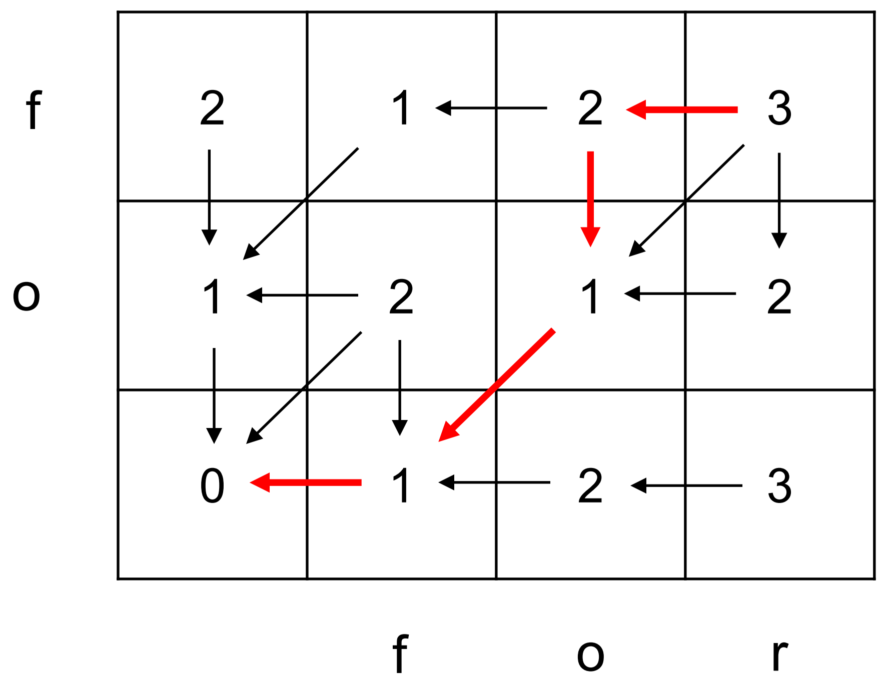
Minimum Edit Distance

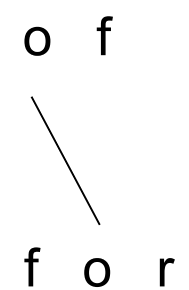




insert f
keep o
substitute f by r

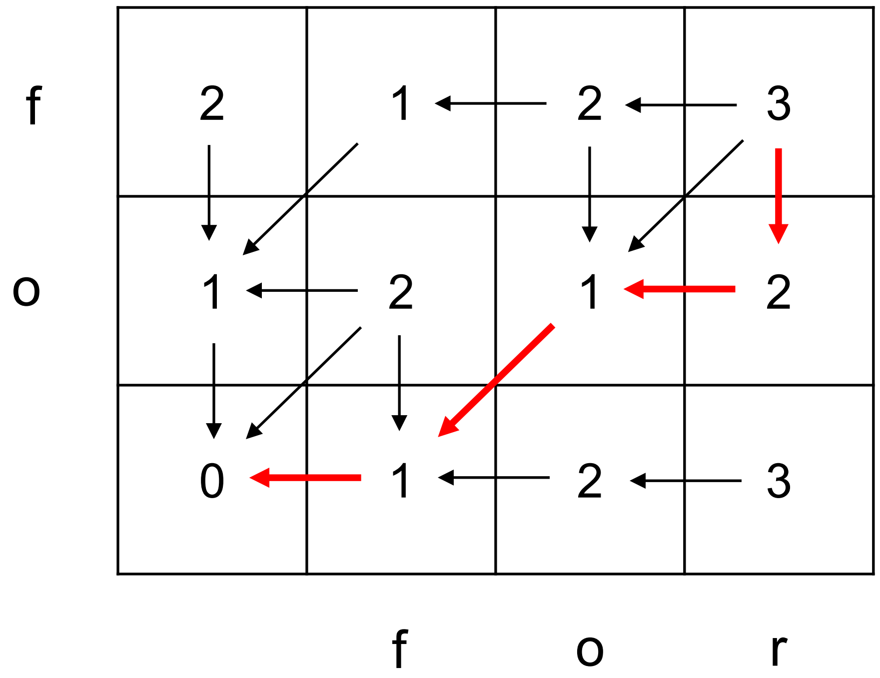
Minimum Edit Distance

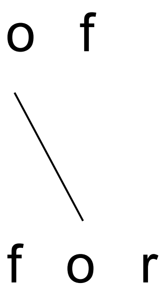




insert f
keep o
delete f
insert r

Minimum Edit Distance





insert f
keep o
insert r
delete f