

Stringvergleich: Minimum Edit Distance

Katja Markert

Institut für Computerlinguistik
Uni Heidelberg
`markert@cl.uni-heidelberg.de`

November 1, 2022

- Bisher: Generierung und Erkennung von Strings
- mit der Hilfe von regulären Ausdrücken und Automaten
- Jetzt: Vergleich von Strings

Warum Stringvergleich?

- 1 Könnte morphologische Relation indizieren:
walk — walks, sleep — sleeps
- 2 **Rechtschreibkorrektur:** *giraffe* bessere Verbesserung für *graffe* als *graffity* (wenn man Kontext ignoriert). Warum?
- 3 **Rechtschreibkorrektur:** 0.05% - 38% der Wörter eines Textes sind inkorrekt, je nach Textsorte (siehe J&M, 2nd edition, Ch 2)
- 4 ähnliche Argumente bei OCR oder handwriting recognition
- 5 **Koreferenzerkennung:** *United States, States, United States of America* wahrscheinlicher die gleiche Koreferenzkette als *United States, Mexico und Guatemala*

Warum Stringvergleich?

- 1 Könnte morphologische Relation indizieren:
walk — walks, sleep — sleeps
- 2 **Rechtschreibkorrektur:** *giraffe* bessere Verbesserung für *graffe* als *graffity* (wenn man Kontext ignoriert). Warum?
- 3 **Rechtschreibkorrektur:** 0.05% - 38% der Wörter eines Textes sind inkorrekt, je nach Textsorte (siehe J&M, 2nd edition, Ch 2)
- 4 ähnliche Argumente bei OCR oder handwriting recognition
- 5 **Koreferenzerkennung:** *United States, States, United States of America* wahrscheinlicher die gleiche Koreferenzkette als *United States, Mexico und Guatemala*

Warum Stringvergleich?

- 1 Könnte morphologische Relation indizieren:
walk — walks, sleep — sleeps
- 2 **Rechtschreibkorrektur:** *giraffe* bessere Verbesserung für *graffe* als *graffity* (wenn man Kontext ignoriert). Warum?
- 3 **Rechtschreibkorrektur:** 0.05% - 38% der Wörter eines Textes sind inkorrekt, je nach Textsorte (siehe J&M, 2nd edition, Ch 2)
- 4 ähnliche Argumente bei OCR oder handwriting recognition
- 5 **Koreferenzerkennung:** *United States, States, United States of America* wahrscheinlicher die gleiche Koreferenzkette als *United States, Mexico* und *Guatemala*

- Lexikonbasiert
- Markiere alle Wörter, die nicht im Lexikon verzeichnet sind
- Voraussetzung: korrekte Tokenisierung! (s. nächste Vorlesung)
- Würden Sie zu dieser Erkennung am liebsten ein besonders großes und vollständiges Lexikon benutzen?
- Ist die Erkennung von Nichtwörtern ausreichend zur Rechtschreibkorrektur?

- Lexikonbasiert
- Markiere alle Wörter, die nicht im Lexikon verzeichnet sind
- Voraussetzung: korrekte Tokenisierung! (s. nächste Vorlesung)
- Würden Sie zu dieser Erkennung am liebsten ein besonders großes und vollständiges Lexikon benutzen?
- Ist die Erkennung von Nichtwörtern ausreichend zur Rechtschreibkorrektur?

- Lexikonbasiert
- Markiere alle Wörter, die nicht im Lexikon verzeichnet sind
- Voraussetzung: korrekte Tokenisierung! (s. nächste Vorlesung)
- Würden Sie zu dieser Erkennung am liebsten ein besonders großes und vollständiges Lexikon benutzen?
- Ist die Erkennung von Nichtwörtern ausreichend zur Rechtschreibkorrektur?

- Suche nach der wahrscheinlichsten korrekten Wortform
- Distanzmaß zwischen inkorrekt und potentiell korrekter Wortform
 - *graffe* vs. *giraffe*, *graffity*, *raffle*
- Einfachste Möglichkeit: **Minimum Edit Distance**
- berücksichtigt Kontext nicht
- nicht probabilistisch

Definition

MED zwischen zwei Zeichenketten ist die minimale Anzahl von Operationen, die benötigt wird, um die eine Zeichenkette in die andere zu überführen. Operationen sind hierbei *einfügen*, *ersetzen*, *löschen*.

Beispiel:

- 3 mal Ersetzen
- 1 mal Löschen
- 1 mal Einfügen

I	N	T	E	*	N	T	I	O	N
↓	↓	↓		↓	↓				
*	E	X	E	C	U	T	I	O	N

Alignierung/alignment zeigt die Korrespondenz zwischen zwei Zeichenketten und bestimmt Operationen.

- Levenshtein Distanz (1966): Jede der drei Operationen hat die gleichen Kosten (=1) $\rightarrow$ MED zwischen *Intention* und *Execution* = 5
- Alternative:
 - Einfügen und Löschen: Kosten 1
 - Ersetzen = Löschen + Einfügen. Kosten 2MED zwischen *Intention* und *Execution* = 8
- MATCH kostet nichts
- Kosten können also verschieden sein. MED ist kostenabhängig.
- Es kann mehrere gleich gute Alignierungen geben

Der Raum aller Editiersequenzen ist riesig

- Können nicht alle naiv durchsuchen
- Viele verschiedene Pfade führen zum gleichen Zustand.
- Sollten uns nur den kürzesten Pfad zu jedem Zustand merken

- Klasse von Algorithmen, die auf Tabellen operieren
- Probleme werden auf Teilprobleme zurückgeführt
- Berechnete Teilergebnisse werden abgespeichert
- Ein einmal berechnetes Teilproblem muss nicht nochmals berechnet werden
- Lösung des Gesamtproblems setzt sich aus berechneten Teilergebnissen zusammen

Varianten, die wir noch sehen werden

- Viterbi Algorithmus für HMM POS Tagging
- CKY (und Earley) Algorithmus (Chart Parsing)

Haupteinsicht

Ist eine Sequenz von Änderungsoperationen optimal (minimal), so ist auch jede Teilsequenz optimal (minimal).

*optimale
Sequenz
=
minimale
Distanz*

*optimale
Teil-
sequenz*

i	n	t	e	n	t	i	o	n	
n	t	e	n	t	i	o	n		Löschen
e	t	e	n	t	i	o	n		Ersetzen
e	x	e	n	t	i	o	n		Einfügen
e	x	e	c	n	t	i	o	n	
e	x	e	c	u	t	i	o	n	

Seien V, W Zeichenketten und x, y einzelne Zeichen. Dann gilt:

$$\begin{aligned} med(Vx, Wy) = \min(\\ & med(V, W) + c_{sub}(x, y), \\ & med(V, Wy) + c_{del}(x), \\ & med(Vx, W) + c_{ins}(y) \\ &) \end{aligned}$$

Beispiel:

$$\begin{aligned} med(inte, exec) = \min(\\ & med(int, exe) + c_{sub}(e, c), \\ & med(int, exec) + c_{del}(e), \\ & med(inte, exe) + c_{ins}(c) \\ &) \end{aligned}$$

Für eine Zeichenkette $X = x_1 \dots x_n$, definieren wir $X[1 \dots i] := x_1 \dots x_i$

Zwischen zwei Zeichenketten $X = x_1 \dots x_n$, $Y = y_1 \dots y_m$, sei
 $med(i, j) := med(X[1 \dots i], Y[1 \dots j])$.

Also ist $med(X, Y) = med(n, m)$

Formalisiert für zwei gegebene Zeichenketten

$X = x_1 \dots x_n$ source, $Y = y_1 \dots y_m$ target

Initialisierung:

- $med(0,0) = 0$
- $med(i,0) = med(i-1,0) + c_{del}([X(i)])$ für $1 < i \leq n$;
- $med(0,j) = med(0,j-1) + c_{ins}([Y(j)])$ für $1 < j \leq m$;

Rekursion:

Für alle $i = 1 \dots n$

für alle $j = 1 \dots m$

$$med(i,j) = \min(\begin{aligned} &med(i-1,j) + c_{del}(X[i]), \\ &med(i-1,j-1) + c_{sub}(X[i], Y[j]), \\ &med(i,j-1) + c_{ins}(Y[j]) \end{aligned})$$

Formalisiert für zwei gegebene Zeichenketten

$X = x_1 \dots x_n$ source, $Y = y_1 \dots y_m$ target

Initialisierung:

- $med(0,0) = 0$
- $med(i,0) = med(i-1,0) + c_{del}([X(i)])$ für $1 < i \leq n$;
- $med(0,j) = med(0,j-1) + c_{ins}([Y(j)])$ für $1 < j \leq m$;

Rekursion:

Für alle $i = 1 \dots n$

für alle $j = 1 \dots m$

$$\begin{aligned} med(i,j) = \min(\\ & med(i-1,j) + c_{del}(X[i]), \\ & med(i-1,j-1) + c_{sub}(X[i], Y[j]), \\ & med(i,j-1) + c_{ins}(Y[j]) \\ &) \end{aligned}$$

Kosten insertion, deletion 1, substitution 2, gleich lassen Null:

$$\begin{aligned} med(i,j) = \min(\\ & med(i-1,j) + 1, \\ & med(i-1,j-1) + (2 \text{ oder } 0), \\ & med(i,j-1) + 1) \\ &) \end{aligned}$$

Als Tabelle (Alignment Matrix)

Tabelle von Größe $(n+1)$ Zeilen und $(m+1)$ Spalten.

X/Y	#	1	.	j	.	.	m
#							
1							
.							
.			i-1,j-1	i-1,j			
i			i, j-1	i,j			
.							
.							
n							

- $i-1,j-1 \rightarrow i,j$: Ersetzung/substitution
- $i-1,j \rightarrow i,j$: Löschen/deletion von source ($X[i]$) zu target
- $i,j-1 \rightarrow i,j$: Einfügen/insertion von source zu target. Füge $Y[j]$ nach $X[i]$ ein.

Die obige Formalisierung erlaubt flexible Kosten. Im folgenden Beispiel benutzen wir

- $c_{del}(X[i]) = 1$, unabh. von $X[i]$;
- $c_{ins}(Y[j]) = 1$ unabh. von $Y[j]$;
- $c_{sub}(X[i], Y[j]) = 2$ für $X[i] \neq Y[j]$,
- $c_{sub}(X[i], Y[j]) = 0$ für $X[i] = Y[j]$

Beispiel: Initialisierung

X/Y	#	e	x	e	c	u	t	i	o	n
#	0	1	2	3	4	5	6	7	8	9
i	1									
n	2									
t	3									
e	4									
n	5									
t	6									
i	7									
o	8									
n	9									

X/Y	#	e	x	e	c	u	t	i	o	n
#	0	1	2	3	4	5	6	7	8	9
i	1	2	?							
n	2									
t	3									
e	4									
n	5									
t	6									
i	7									
o	8									
n	9									

Beispiel: Weiter

X/Y	#	e	x	e	c	u	t	i	o	n
#	0	1	2	3	4	5	6	7	8	9
i	1	2	3	4	5	6	7	?		
n	2									
t	3									
e	4									
n	5									
t	6									
i	7									
o	8									
n	9									

X/Y	#	e	x	e	c	u	t	i	o	n
#	0	1	2	3	4	5	6	7	8	9
i	1	2	3	4	5	6	7	6	7	8
n	2									
t	3									
e	4									
n	5									
t	6									
i	7									
o	8									
n	9									

Beispiel: Weiter

X/Y	#	e	x	e	c	u	t	i	o	n
#	0	1	2	3	4	5	6	7	8	9
i	1	2	3	4	5	6	7	6	7	8
n	2	3	4	5	6	7	8	7	8	7
t	3									
e	4									
n	5									
t	6									
i	7									
o	8									
n	9									

Bitte den Rest ausfüllen!

Beispiel: Weiter

X/Y	#	e	x	e	c	u	t	i	o	n
#	0	1	2	3	4	5	6	7	8	9
i	1	2	3	4	5	6	7	6	7	8
n	2	3	4	5	6	7	8	7	8	7
t	3	4	5	6	7	8	7	8	9	8
e	4	3	4	5	6	7	8	9	10	9
n	5	4	5	6	7	8	9	10	11	10
t	6	5	6	7	8	9	8	9	10	11
i	7	6	7	8	9	10	9	8	9	10
o	8	7	8	9	10	11	10	9	8	9
n	9	8	9	10	11	12	11	10	9	8

$$\text{med}(X, Y) = 8$$

Beispiel: Aligierung

- Wir können uns merken, wo unsere Werte hergekommen sind
- Dies ergibt ein (oder mehrere) *alignments* und damit eine Serie von substitutions, deletions, insertions.
- Jeder vollständige Pfad absteigender Kosten von (n, m) (untere rechte Ecke) nach $(0,0)$ repräsentiert eine minimale Editiersequenz.

	#	e	x	e	c	u	t	i	o	n
#	0	1	2	3	4	5	6	7	8	9
i	1	↖↔2	↖↔3	↖↔4	↖↔5	↖↔6	↖↔7	↖6	←7	←8
n	2	↖↔3	↖↔4	↖↔5	↖↔6	↖↔7	↖↔8	↑7	↖↔8	↖7
t	3	↖↔4	↖↔5	↖↔6	↖↔7	↖↔8	↖7	↖↔8	↖↔9	↑8
e	4	↖3	←4	↖↔5	←6	←7	↖↔8	↖↔9	↖↔10	↑9
n	5	↑4	↖↔5	↖↔6	↖↔7	↖↔8	↖↔9	↖↔10	↖↔11	↖↔10
t	6	↑5	↖↔6	↖↔7	↖↔8	↖↔9	↖↔8	←9	←10	↖↔11
i	7	↑6	↖↔7	↖↔8	↖↔9	↖↔10	↑9	↖↔8	←9	←10
o	8	↑7	↖↔8	↖↔9	↖↔10	↖↔11	↑10	↑9	↖↔8	←9
n	9	↑8	↖↔9	↖↔10	↖↔11	↖↔12	↑11	↑10	↑9	↖↔8

Landau Symbole

Gibt ein Mass für die maximale Anzahl der Elementarschritte oder der Speichereinheiten in Abhängigkeit von der Grösse des gegebenen Problems an. $f = O(g)$ heisst “wächst nicht wesentlich schneller als g ” bzw. $f(x) \leq C \cdot |g(x)|$ für eine Konstante C . Es gibt auch klein-o, dann $f = o(g)$ heisst “wächst langsamer als g ”

Laufzeit: $O(nm)$

Laufzeit Backtrace: $O(n + m)$

Speicher = Tabellengrösse: $O(nm)$

Kann es Sinn machen, unterschiedliche Gewichte für unterschiedliche Löschungen, Substitutionen etc zu verwenden?

Wenn Sie *Nanen* getippt hatten (fälschlicherweise), ist es dann wahrscheinlicher, dass Sie *Nasen* oder *Namen* tippen wollten (kontextunabhängig)?

Andere Gewichte?

sub[X, Y] = Substitution of X (incorrect) for Y (correct)

X	Y (correct)																									
	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	0	0	7	1	342	0	0	2	118	0	1	0	0	3	76	0	0	1	35	9	9	0	1	0	5	0
b	0	0	9	9	2	2	3	1	0	0	0	5	11	5	0	10	0	0	2	1	0	0	8	0	0	0
c	6	5	0	16	0	9	5	0	0	0	1	0	7	9	1	10	2	5	39	40	1	3	7	1	1	0
d	1	10	13	0	12	0	5	5	0	0	2	3	7	3	0	1	0	43	30	22	0	0	4	0	2	0
e	388	0	3	11	0	2	2	0	89	0	0	3	0	5	93	0	0	14	12	6	15	0	1	0	18	0
f	0	15	0	3	1	0	5	2	0	0	0	0	3	4	1	0	0	6	4	12	0	0	2	0	0	0
g	4	1	11	11	9	2	0	0	0	1	1	3	0	0	2	1	3	5	13	21	0	0	1	0	3	0
h	1	8	0	3	0	0	0	0	0	0	2	0	12	14	2	3	0	3	1	11	0	0	2	0	0	0
i	103	0	0	0	146	0	1	0	0	0	0	6	0	0	49	0	0	0	2	1	47	0	2	1	15	0
j	0	1	1	9	0	0	1	0	0	0	0	2	1	0	0	0	0	5	0	0	0	0	0	0	0	0
k	1	2	8	4	1	1	2	5	0	0	0	0	5	0	2	0	0	0	6	0	0	0	4	0	0	3
l	2	10	1	4	0	4	5	6	13	0	1	0	0	14	2	5	0	11	10	2	0	0	0	0	0	0
m	1	3	7	8	0	2	0	6	0	0	4	4	0	180	0	6	0	0	9	15	13	3	2	2	3	0
n	2	7	6	5	3	0	1	19	1	0	4	35	78	0	0	7	0	28	5	7	0	0	1	2	0	2
o	91	1	1	3	116	0	0	0	25	0	2	0	0	0	0	14	0	2	14	39	0	0	0	18	0	0
p	0	11	1	2	0	6	5	0	2	9	0	2	7	6	15	0	0	1	3	6	0	4	1	0	0	0
q	0	0	1	0	0	0	27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	0	14	0	30	12	2	2	8	2	0	5	8	4	20	1	14	0	0	12	22	4	0	0	1	0	0
s	11	8	27	33	35	4	0	1	0	1	0	27	0	6	1	7	0	14	0	15	0	0	5	3	20	1
t	3	4	9	42	7	5	19	5	0	1	0	14	9	5	5	6	0	11	37	0	0	2	19	0	7	6
u	20	0	0	0	44	0	0	0	64	0	0	0	0	2	43	0	0	4	0	0	0	0	2	0	8	0
v	0	0	7	0	0	3	0	0	0	0	0	1	0	0	1	0	0	0	8	3	0	0	0	0	0	0
w	2	2	1	0	1	0	0	2	0	0	1	0	0	0	0	7	0	6	3	3	1	0	0	0	0	0
x	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0	0
y	0	0	2	0	15	0	1	7	15	0	0	0	2	0	6	1	0	7	36	8	5	0	0	1	0	0
z	0	0	0	7	0	0	0	0	0	0	0	7	5	0	0	0	0	2	21	3	0	0	0	0	3	0

Von Dan Jurafsky

Evaluation von MT und Spracherkennung

Spokesman	confirms		senior	government	adviser	was	shot	
Spokesman	said	the	senior		adviser	was	shot	dead
	S	I		D				I

Sie können:

- Die Minimum Edit Distanz zwischen zwei Zeichenketten berechnen
- Die Alignierung ableiten
- Das Konzept der dynamischen Programmierung beschreiben

- * Jurafsky und Martin, Chapter 2.5 (3rd edition, online)
- * Aufgabenblatt 2/3
- Interaktive Demo für Levenshtein Distanz (mit Ausgabe aller minimalen Editiersequenzen):

<http://www.let.rug.nl/kleiweg/lev/>

- * Jurafsky und Martin, Chapter 2, 3rd edition (online)
- Gusfield, D. (1997). *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge University Press.
- Bellman, R. (1957). *Dynamic Programming*. Princeton University Press.