

Dynamic Time Warping aneb Dynamické borcení času aneb Dynamic Programming aneb Dynamické programování

Jan Černocký

FIT VUT Brno

Obsah

1	Úvod	1
2	Omezení cesty	3
3	Definice váhových funkcí	3
4	Normalizační faktor	4
5	Lokální omezení cesty	5
6	Efektivní výpočet $D(\mathbf{O}, \mathbf{R})$	5
7	Realisace rozpoznávače založeného na DTW	6
7.1	Trénování – tvorba referencí neboli vzorů	6
8	Rozpoznávání (klasifikace)	7

1 Úvod

Potřebujeme určit *vzdálenost* (či *podobnost*) referenční sekvence vektorů o délce R :

$$\mathbf{R} = [\mathbf{r}(1), \dots, \mathbf{r}(R)] \quad (1)$$

a testovací sekvence vektorů o délce T :

$$\mathbf{O} = [\mathbf{o}(1), \dots, \mathbf{o}(T)] \quad (2)$$

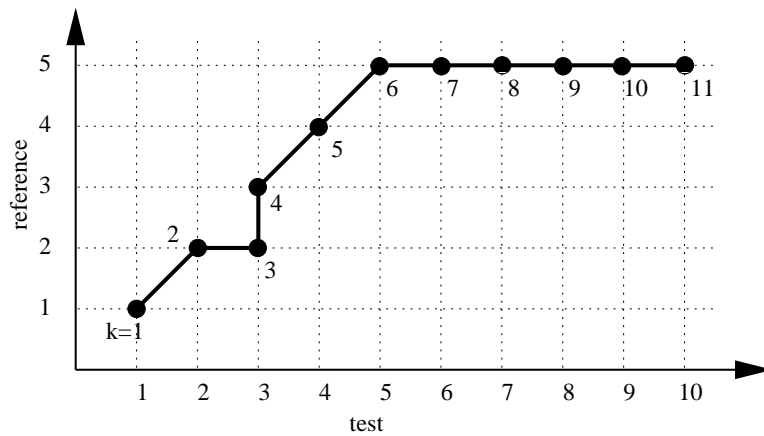
Co čert nechtěl, obecně je samozřejmě $R \neq T$.

Jedna z možností je využití jediné transformační funkce času:

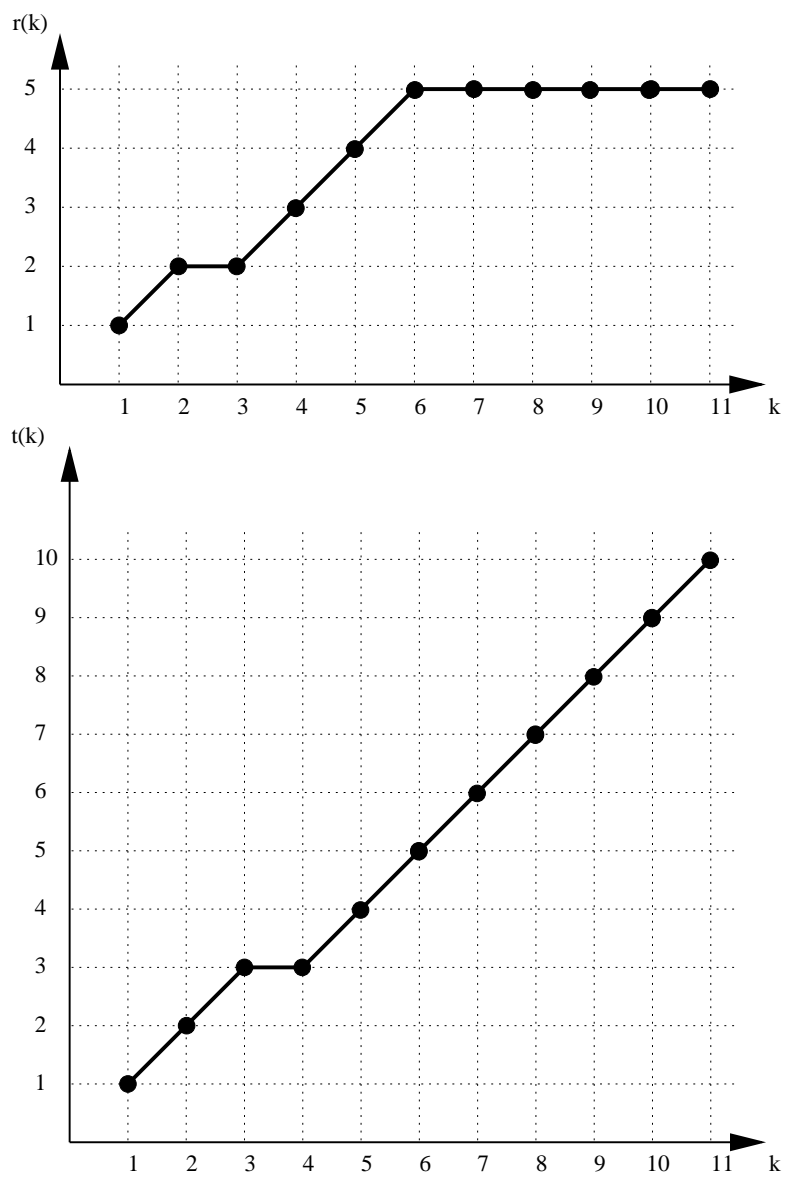
$$D(\mathbf{O}, \mathbf{R}) = \sum_{i=1}^R d[\mathbf{o}(w(i)), \mathbf{r}(i)] \quad (3)$$

kde $w(i)$ je definována tak, aby srovnání bylo lineární. Je ovšem podstatně lepší když:

- je srovnání *řízeno* přímo vzdálenostmi jednotlivých vektorů.
- když definujeme obecnou časovou proměnnou k a a zavedeme *dvě* transformační funkce:
 - $r(k)$ pro referenční sekvenci.
 - $t(k)$ pro testovací sekvenci.



Obrázek 1: Cesta pro srovnání dvou sekvencí.



Obrázek 2: Funkce $r(k)$ a $t(k)$ pro krokování referenční a testovací sekvence.

Pak můžeme srovnání jednotlivých vektorů zakreslit pomocí **cesty**: Obr. 1. Počet kroků cesty označíme jako K . Referenci budeme zobrazovat na svislé ose, test na vodorovné. Z této cesty můžeme odvodit průběh funkcí $r(k)$ a $t(k)$, které “krokují” jednotlivé sekvence, viz Obr. 2.

Cesta C je jednoznačně dána svou délkou K_C a průběhem funkcí $r_C(k)$ a $t_C(k)$. Pro tuto cestu je vzdálenost sekvencí \mathbf{O} a \mathbf{R} dána jako:

$$D_C(\mathbf{O}, \mathbf{R}) = \frac{\sum_{k=1}^{K_C} d[\mathbf{o}(t_C(k)), \mathbf{r}(r_C(k))] W_C(k)}{N_C} \quad (4)$$

kde $d[\mathbf{o}(\cdot), \mathbf{r}(\cdot)]$ je vzdálenost dvou vektorů, $W_C(k)$ je váha odpovídající k -tému kroku cesty a N_C je normalizační faktor závislý na vahách.

Vzdálenost sekvencí \mathbf{O} a \mathbf{R} je dána jako *minimální vzdálenost* přes soubor všech možných cest (všechny možné délky, všechny možné průběhy):

$$D(\mathbf{O}, \mathbf{R}) = \min_{\{C\}} D_C(\mathbf{O}, \mathbf{R}). \quad (5)$$

Je třeba vyřešit 3 věci:

1. přípustné průběhy funkcí $r(k)$ a $t(k)$. Není možné, aby se cesta vracela zpět, “skákala” přes několik vektorů, atd.
2. definovat váhovací funkci a normalizační faktor.
3. vyrobit algoritmus, který vypočítá $D(\mathbf{O}, \mathbf{R})$ co nejrychleji.

2 Omezení cesty

1. Počáteční a koncové body

$$\left. \begin{array}{l} r(1) = 1 \\ t(1) = 1 \end{array} \right\} \text{začátek} \quad \left. \begin{array}{l} r(K) = R \\ t(K) = T \end{array} \right\} \text{konec} \quad (6)$$

2. Lokální souvislost a lokální strmost

$$\begin{array}{l} 0 \leq r(k) - r(k-1) \leq R^* \\ 0 \leq t(k) - t(k-1) \leq T^* \end{array} \quad (7)$$

v praxi se volí $R^*, T^* = 1, 2, 3$.

- $R^*, T^* = 1$: Každý vektor se musí vzít alespoň jedenkrát. $r(k) = r(k-1)$ znamená, že se opakuje.
- $R^*, T^* > 1$: Vektor(y) se mohou přeskočit.

3. Globální vymezení cesty DTW: vymezení přípustné oblasti pomocí přímek:

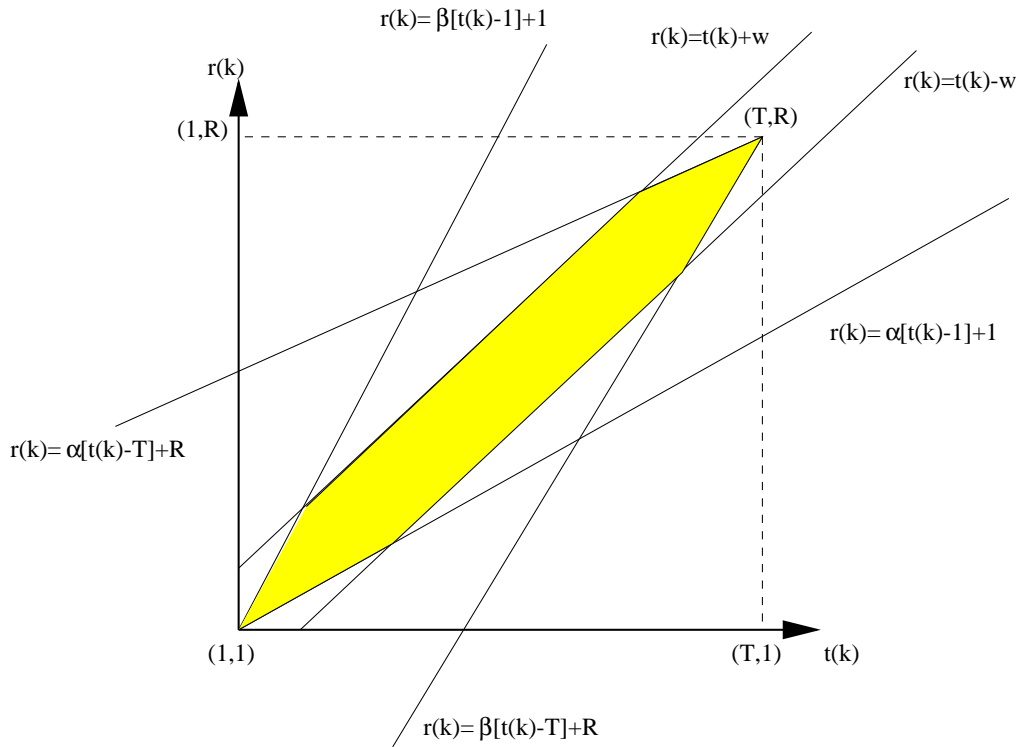
$$\begin{array}{l} 1 + \alpha[t(k) - 1] \leq r(k) \leq 1 + \beta[t(k) - 1] \\ R + \beta[t(k) - T] \leq r(k) \leq R + \alpha[t(k) - T] \end{array} \quad (8)$$

Tyto podmínky vymezují maximální “strmost” nebo “placatost” cesty DTW – viz Obr. 3

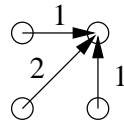
3 Definice váhových funkcí

Váhová funkce $W(k)$ závisí na lokálním “posunu” cesty. 4 typy:

- **typ a)** symetrická: $W_a(k) = [t(k) - t(k-1)] + [r(k) - r(k-1)]$.

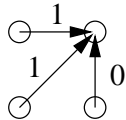


Obrázek 3: Globální omezení cesty DTW. Povolená oblast, kde se může cesta nacházet, je vyznačena žlutě.

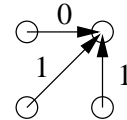


- **typ b)** asymetrická:

b1) $W_{b1}(k) = t(k) - t(k-1)$



b2) $W_{b2} = r(k) - r(k-1)$



- **typ c)** $W_c(k) = \min \{t(k) - t(k-1), r(k) - r(k-1)\}$
- **typ d)** $W_d(k) = \max \{t(k) - t(k-1), r(k) - r(k-1)\}$

4 Normalisační faktor

$$N = \sum_{k=1}^K W(k) \tag{9}$$

Pro váhovací funkci typu a) je normalisační faktor:

$$N_a = \sum_{k=1}^K [t(k) - t(k-1) + r(k) - r(k-1)] = t(K) - t(0) + r(K) - r(0) = T + R \tag{10}$$

Pro váhovací funkci typu b1) je normalisační faktor $N = T$.

Pro váhovací funkci typu b2) je normalisační faktor $N = R$.

Pro typy c), d) faktor silně závislý na průběhu cesty, je lepší použít konstantu: $N = T$.

Typ DTW		α	β	Typ $w(k)$	$g(n, m)$
I.		0	∞	a	$\min \left\{ \begin{array}{l} g(n, m-1) + d(n, m) \\ g(n-1, m-1) + 2d(n, m) \\ g(n-1, m) + d(n, m) \end{array} \right\}$
				d	$\min \left\{ \begin{array}{l} g(n, m-1) + d(n, m) \\ g(n-1, m-1) + d(n, m) \\ g(n-1, m) + d(n, m) \end{array} \right\}$
II.		$\frac{1}{2}$	2	a	$\min \left\{ \begin{array}{l} g(n-1, m-2) + 3d(n, m) \\ g(n-1, m-1) + 2d(n, m) \\ g(n-2, m-1) + 3d(n, m) \end{array} \right\}$
				d	$\min \left\{ \begin{array}{l} g(n-1, m-2) + d(n, m) \\ g(n-1, m-1) + d(n, m) \\ g(n-2, m-1) + d(n, m) \end{array} \right\}$
III.		$\frac{1}{2}$	2	a	$\min \left\{ \begin{array}{l} g(n-1, m-2) + 2d(n, m-1) + d(n, m) \\ g(n-1, m-1) + 2d(n, m) \\ g(n-2, m-1) + 2d(n-1, m) + d(n, m) \end{array} \right\}$
IV.		$\frac{1}{2}$	2	b1	$\min \left\{ \begin{array}{l} g(n-1, m) + kd(n, m) \\ g(n-1, m-1) + d(n, m) \\ g(n-1, m-2) + d(n, m) \end{array} \right\}$ kde $k = 1$ pro $r(k-1) \neq r(k-2)$ $k = \infty$ pro $r(k-1) = r(k-2)$

Tabulka 1: Typy lokálních omezení cesty a z nich vyplývající faktory α a β . $g(n, m)$ jsou vztahy pro výpočet částečné kumulované vzdálenosti. U typu IV není povolen pohyb dvakrát za sebou jen ve směru t .

5 Lokální omezení cesty

Tabulka 1 udává typy lokálních omezení cesty a z nich vyplývající faktory α a β . Význam veličiny $g(n, m)$ bude vysvětlen později.

6 Efektivní výpočet $D(\mathbf{O}, \mathbf{R})$

Výpočet minimální vzdálenosti

$$D(\mathbf{O}, \mathbf{R}) = \min_{\{C\}} D_C(\mathbf{O}, \mathbf{R}). \quad (11)$$

je jednoduchý, pokud normalizační faktor N_C není funkcí cesty a můžeme psát:

$$N_C = N \quad \text{pro } \forall C$$

Toto naštěstí většinou platí a tedy:

$$D(\mathbf{O}, \mathbf{R}) = \frac{1}{N} \min_{\{C\}} \sum_{k=1}^{K_C} d[\mathbf{o}(t_C(k)), \mathbf{r}(r_C(k))] \quad (12)$$

Postup je následující:

1. do mřížky \mathbf{d} o velikosti $T \times R$ si zapíšeme vzdálenosti referenčních a testovacích vektorů, každý s každým, viz Příklad.

		d				
		4	3	2		
		2	3	1		
ref.		4	2	3		
		0	1	1		
						test

		g				
		inf	10	9	7	
		inf	6	6	5	
		inf	4	3	5	
ref.		inf	0	1	2	
		0	inf	inf	inf	
						test

Obrázek 4: Příklad. **d** je mřížka lokálních vzdáleností. **g** je mřížka částečných kumulovaných vzdáleností.

2. definujeme mřížku **g** s *částečnou kumulovanou vzdáleností*. Oproti mřížce **d** má **g** navíc ještě nultý řádek a nultý sloupec, které inicialisujeme na:

$$g(0, 0) = 0, \quad \text{a} \quad g(0, m \neq 0) = g(n \neq 0, 0) = \infty.$$

3. Částečnou kumulovanou vzdálenost spočítáme pro každý bod takto:

$$g(m, n) = \min_{\forall \text{předchůdci}} [g(\text{předchůdce}) + d(m, n)w(k)] \quad (13)$$

- možní předchůdci jsou dáni pomocí tabulky lokálních omezení cesty.
- váha $w(k)$ odpovídá pohybu z předchůdce do bodu $[m, n]$.
- vztahy pro výpočet Částečné kumulované vzdálenosti jsou tabelovány v Tabulce 1.

4. Konečná minimální normovaná vzdálenost je pak dána:

$$D(\mathbf{O}, \mathbf{R}) = \frac{1}{N}g(T, R) \quad (14)$$

Příklad

Viz Obrázek 4 Výsledek:

- máme vzdálenost $D = \frac{1}{3+4}7 = 1$.
- můžeme zpětně “odkrokovat optimální cestu” (cesta má 5 kroků): $t(k) = [1 \ 2 \ 2 \ 3 \ 3]$, $r(k) = [1 \ 1 \ 2 \ 3 \ 4]$.

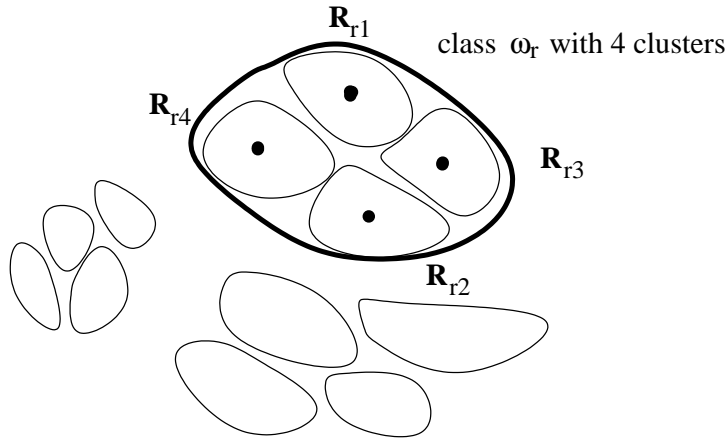
7 Realisace rozpoznávače založeného na DTW

Opakování: co vlastně chci? Přijde slovo **O**; chci ho zatřídit do třídy ω_r . Mám \tilde{N} tříd, které odpovídají slovům (např. “jedna”, “dvě”, “křížek”, atd.).

7.1 Trénování – tvorba referencí neboli vzorů

při trénování mám sekvence od jednoho nebo více řečníků, a vím, kam která patří.

1. **nejjednodušší**: ve třídě ω_r mám jen jednu referenci **R_r**.



Obrázek 5: Shluky.

2. **složitější:** pro každou třídu ω_r mám více referencí: $\mathbf{R}_{r,1} \dots \mathbf{R}_{r,\tilde{N}_r}$. Tyto mohou mít ve slovníku uložené tak, jak byly vytvořeny, nebo je lineárně normalisovat na jednotnou délku:

$$\bar{R} = \frac{1}{N} \sum_{r=1}^N \left[\frac{1}{\tilde{N}_r} \sum_{i=1}^{\tilde{N}_r} R_{r,i} \right], \quad (15)$$

kde $R_{r,i}$ je délka i -tého vzoru třídy ω_r .

3. **vytvoření průměrného vzoru pro každou třídu ω_r :**

- lineární průměrování – bereme průměr vektorů lineárně srovnaných sekvencí. Nebezpečí: mohou dostat zcela nesmyslný vzor ...
- dynamické průměrování:
 - (a) najdu vzor s o délkou, která se mi líbí.
 - (b) průměrování se děje z vektorů přiřazených k tomuto prvnímu vzoru pomocí cesty DTW.

4. vytváření vzorů *shlukováním*, viz Obrázek 5. Shluky jsou tvořeny tak, aby si byly reference uvnitř shluků co nejvíce podobné, mezi shluky co nejméně podobné. Existují automatické algoritmy pro shlukování, např. Mac Queenův [1]: nejprve se všechny reference přiřadí do jednoho shluku. Pak se odštěpí nejvzdálenější od středu, “přeshlukuje se”, atd. (analogie s VQ). Shluky jsou reprezentovány *centroidy* $\mathbf{R}_{r,i}$. Výhoda shlukování oproti průměrování spočívá v tom, že třídy mohou mít složitější tvar, viz Obr. 6.

8 Rozpoznávání (klasifikace)

Je-li každá třída reprezentována jen jednou referencí, je to jednoduché:

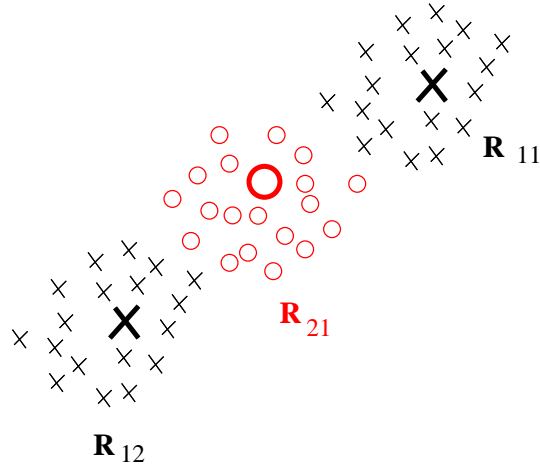
$$\omega_r^* = \arg \min_r D(\mathbf{O}, \mathbf{R}_r) \quad \text{pro } r = 1, \dots, N \quad (16)$$

Mám-li pro každou třídu více referencí, je možné použít dvě metody:

1. **1-NN** (nearest neighbor) neboli nejbližší soused:

$$\omega_r^* = \arg \min_{r,i} D(\mathbf{O}, \mathbf{R}_{r,i}) \quad \text{pro } \begin{matrix} r = 1, \dots, N \\ i = 1, \dots, \tilde{N}_r \end{matrix} \quad (17)$$

2. **k -NN** (k nearest neighbors) neboli k nejbližších sousedů:



Obrázek 6: Příklad úspěšné klasifikace při použití shlukování: v případě průměrování by se vzor třídy ω_1 kryl se vzorem třídy ω_2 .

- pro každou třídu nejprve spočítám všechny vzdálenosti $D(\mathbf{O}, \mathbf{R}_{r_i})$ a seřadím je podle velikosti od nejlepší po nejhorší:

$$D(\mathbf{O}, \mathbf{R}_{r_{(1)}}) \leq D(\mathbf{O}, \mathbf{R}_{r_{(2)}}) \leq \dots \leq D(\mathbf{O}, \mathbf{R}_{r_{(N_r)}}) \quad (18)$$

- o klasifikaci \mathbf{O} do třídy ω_r se pak rozhodnu podle průměrné vzdálenosti k nejbližších sousedů:

$$\omega_r^* = \arg \min_r \frac{1}{k} \sum_{i=1}^k D(\mathbf{O}, \mathbf{R}_{r_{(i)}}) \quad (19)$$

Reference

- [1] J. Psutka. *Komunikace s počítačem mluvenou řečí*. Academia, Praha, 1995.