

# Počítačové zpracování řeči

Přednáška 8

HMM – trénování, implementace

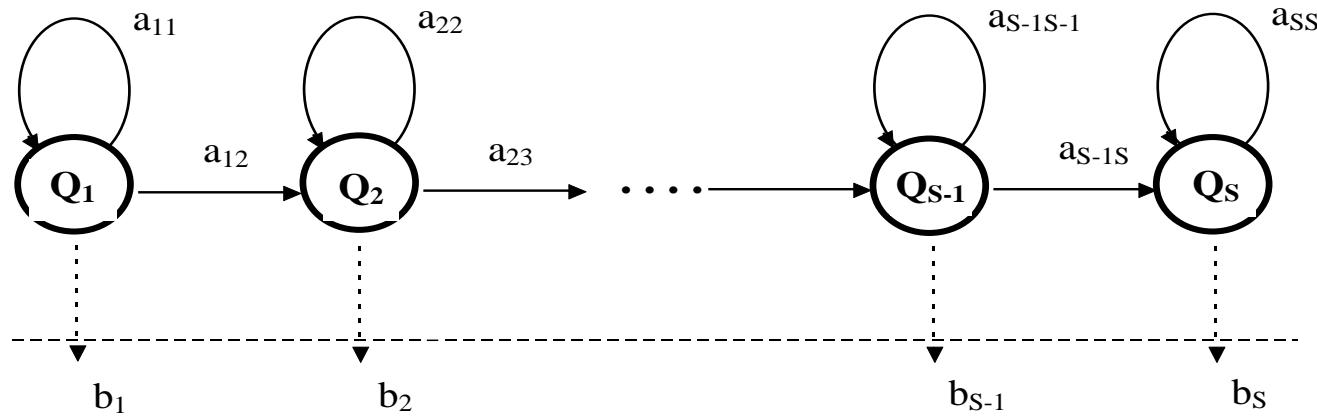
# Zkušenosti z předchozí úlohy

- Viterbiho algoritmus lze snadno naprogramovat modifikací algoritmu DTW
- Natrénované SD modely daly - i s jediným příznakem a pouze 6 stavů - lepší výsledky než DTW pro stejná data (úspěšnost 75 %, oproti cca 60 %)
- Rychlosť rozpoznávání je výrazně vyšší  
(zejména díky malému počtu stavů, který je několikanásobně menší než počet framů referencí)
- Malý počet parametrů modelu oproti referencím  
např. 10 referencí x 50 framů x P příznaků      versus      1 model x 6 stavů x (P příznaků x 2 + 2)
- Zdánlivě složité výpočty založené na Gaussově pravděpodobnostním rozložení se výrazně zjednoduší a zrychlí při aplikaci (přirozeného) logaritmu

# HMM - připomenutí

Metoda HMM (Hidden Markov Model – skryté Markovovy modely) reprezentuje slovo **stavovým modelem** s pravděpodobnostními parametry

Typická struktura slovního HMM - takzvaný *levo-pravý model*



$Q_s \dots$  stavy (šipky naznačují možné přechody mezi nimi – buď setrvání nebo přechod doprava)

$a_{ij} \dots$  přechodová pravděpodobnost – pravděpodobnost, že (v aktuálním framu) model přejde ze stavu  $i$  do stavu  $j$  ( $a_{ii}$  je tedy pravděpodobnost setrvání ve stavu  $i$ )

$b_s(x) \dots$  výstupní pravděpodobnostní rozložení – funkce určující míru pravděpodobnosti, že příznakový vektor  $x$  patří ke stavu  $s$  – v praxi se používá normální (Gaussovo) rozložení

# Jak natrénovat parametry HMM (1)?

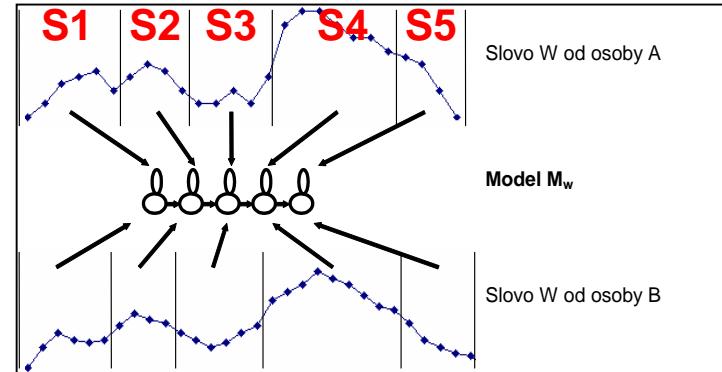
Pro zjednodušení uvažujme 1-rozměrný příznakový vektor  $\mathbf{x}$  (např. ENE)

Předpokládáme, že  $b_s(x)$  má gauss. rozdělení se dvěma parametry  $\mu_s, \sigma_s$

$$b_s(x) = \frac{1}{\sqrt{2\pi} \cdot \sigma_s} \exp\left[-\frac{(x - \mu_s)^2}{2\sigma_s^2}\right]$$

Mějme alespoň 2 nahrávky pro každé slovo  
(čím více nahrávek, čím lepší model získáme)

a pro každé slovo sekvenci jeho příznaků

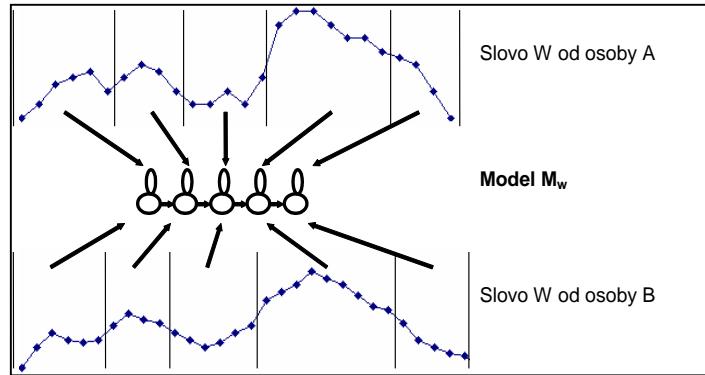


Pokud víme, které framy patří k jednotlivým stavům  
(jak je ukázáno na obrázku, kde svislé čáry vymezují 5 stavů)

pak stř. hodnotu určíme jako  $\mu_s = \frac{1}{N_s} \sum_{n=1}^{N_s} \mathbf{x}_n$  a rozptyl jako  $\sigma_s^2 = \frac{1}{N_s} \sum_{n=1}^{N_s} (\mathbf{x}_n - \mu_s)^2$   
kde  $N_s$  je počet framů přiřazených stavu s

# Jak natrénovat parametry HMM (2)?

Stejně jednoduše se dají určit i přechodové pravděpodobnosti



Pravděpodobnost přechodu ze stavu  $s$  do  $s+1$

$$a_{ss+1} = \frac{K}{N_S}$$

Proč? U každého slova dojde k přechodu ze stavu  $s$  do  $s+1$  právě jednou, u  $K$  slov tedy  $K$ -krát, a to vydělíme souhrnným počtem framů přiřazených ke stavu  $s$ .

Pravděpodobnost setrvání v tomtéž stavu  $s$

$$a_{ss} = 1 - a_{ss+1}$$

Jedná se o pravděpodobnost opačného jevu

**Model je natrénovaný, když pro každý jeho stav známe všechny parametry, tj. hodnoty  $\mu_s, \sigma_s, a_{ss}, a_{ss+1}$**

# Jak trénovat HMM v reálu (1)?

**Ve skutečnosti nevíme**, který frame patří k jakému stavu.

(Z tohoto důvodu se metodě HMM říká **skryté** Markovovy (markovské) modely)

Metoda **trénování HMM** (tj. určování jejich parametrů) je proto **iterativní**

## 1. Inicializační krok

Framy všech nahrávek daného slova přiřadíme rovnoměrně jednotlivým stavům, z nich pak určíme parametry ( $\mu_s, \sigma_s, a_{ss}, a_{ss+1}$ ) první iterace trénovaného modelu

## 2. Přiřazovací krok

S využitím aktuálního modelu provedeme rozpoznávání všech trénovacích nahrávek daného slova a zpětným trasováním ve Viterbiho alg. nalezneme nové (už ne rovnoměrné, ale obvykle lepší) přiřazení mezi framy a stavы

## 3. Reestimační krok

Pro toto nové přiřazení určíme nové parametry modelu

## 4. Opakování, případně konec

Pokud se parametry liší od předchozí iterace, nebo se liší celkové skóre ( $> \epsilon$ ), nebo nedosáhneme max. počtu iterací, jdeme zpět na krok 2, jinak ukončíme trénování

# Jak trénovat HMM v reálu (2)?

## Praktické poznámky k jednotlivým krokům

### 1. Inicializační krok

ad rovnoměrné rozdělení framů: pokud počet framů slova není dělitelný počtem stavů, budou mít některé stavy o 1 frame více než ostatní (není to problém), výpočet parametrů provedeme podle známých vztahů

### 2. Přiřazovací krok

Pro každé slovo patřící k danému modelu aplikujeme Viterbiho algoritmus, určíme log. skóre a provedeme backtracking, přiřazení si uložíme pro krok 3

### 3. Reestimační krok

s nově přiřazenými framy určíme parametry modelu podle známých vztahů,

### 4. Opakování, případně konec

Pro zastavení iteračního cyklu použijeme jedno z kritérií:

- dosažení max. počtu iterací (např. 10) – nejčastější kritérium,
- porovnání součtu všech dílčích skóre (z kroku 2) pro všechna slova trénovací sady,
- porovnání nových a předchozích parametrů modelů

Takto natrénujeme modely všech slov ve slovníku

# Jak s vícepříznakovými vektory (1)

1-rozměrné Gaussovo rozdělení (pro 1 příznak)  $b_s(x) = \frac{1}{\sqrt{2\pi\sigma_s^2}} \exp\left[-\frac{(x-\mu_s)^2}{2\sigma_s^2}\right]$

Při  $P$  příznacích musíme pracovat s  $P$ -rozměrným gaussovským rozdělením

$$b_s(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^P \det \Sigma_s}} \cdot \exp\left[-\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_s)^T \boldsymbol{\Sigma}_s^{-1} (\mathbf{x} - \boldsymbol{\mu}_s)\right]$$

$$\boldsymbol{\mu}_s = \frac{1}{N_s} \sum_{n=1}^{N_s} \mathbf{x}_n$$

$$\boldsymbol{\Sigma}_s = \frac{1}{N_s} \sum_{n=1}^{N_s} (\mathbf{x}_n - \boldsymbol{\mu}_s)(\mathbf{x}_n - \boldsymbol{\mu}_s)^T$$

Kovarianční matici:

$$\boldsymbol{\Sigma}_s = \begin{pmatrix} \sigma_{11}^2 & \sigma_{12}^2 & \dots & \sigma_{1P}^2 \\ \sigma_{21}^2 & \sigma_{22}^2 & & \sigma_{2P}^2 \\ \dots & & & \\ \sigma_{P1}^2 & \sigma_{P2}^2 & & \sigma_{PP}^2 \end{pmatrix}$$

na hlavní diagonále leží **rozptyly jednotlivých příznaků**,  
na ostatních pozicích jsou kovariance („vzájemné  
rozptyly“)  
pro **nekorelované příznaky** jsou hodnoty mimo  
diagonálu nulové (v praxi velmi malé) a **lze je zanedbat**

# Jak s vícepříznakovými vektory (2)

Pro příznaky, které jsou nekorelované, nebo u nichž lze zanedbat malé hodnoty mimo diagonálu, se původní matice zredukuje na **diagonální**, tj. na vektor rozptylů

Výpočet s **využitím logaritmů** se tak významně zjednoduší

$$\begin{aligned} \ln(b_s(\mathbf{x})) &= \ln\left(\frac{1}{\sqrt{(2\pi)^P \det \Sigma_s}}\right) - \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_s)^T \boldsymbol{\Sigma}_s^{-1}(\mathbf{x} - \boldsymbol{\mu}_s) = && \text{„zmizel“ exponenciální člen} \\ &= \ln\left(\frac{1}{\sqrt{(2\pi)^P \cdot (\sigma_{11}^2 \cdot \sigma_{22}^2 \cdots \sigma_{PP}^2)}}\right) - \frac{1}{2}\left(\sum_{p=1}^P (x_p - \mu_{sp})^2 / \sigma_{pp}^2\right) = && \text{zjednodušen součin vektoru a matice} \\ &= -\frac{1}{2}(P \cdot \ln(2\pi) + \sum_{p=1}^P \ln(\sigma_{pp}^2)) - \frac{1}{2}\left(\sum_{p=1}^P (x_p - \mu_{sp})^2 / \sigma_{pp}^2\right) && \text{zjednodušen první člen} \end{aligned}$$

toto je konstanta (pro daný stav), tj. lze ji předpočítat a uložit

Výpočet se pak stává velmi jednoduchý (součet členů podobných jako u 1D gaussovky)

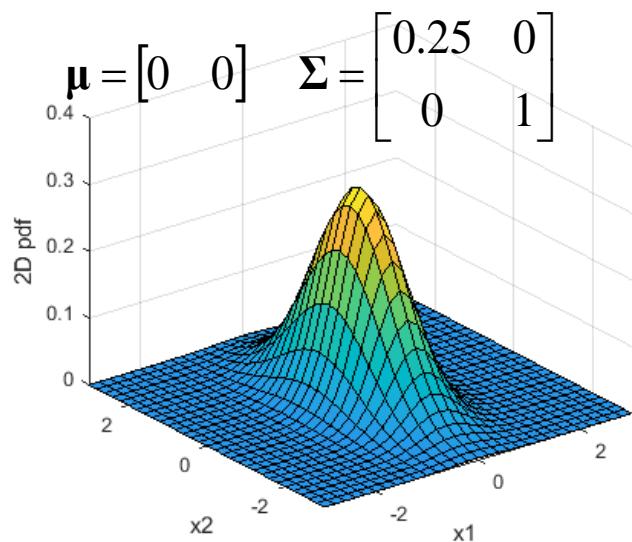
V Matlabu: `ln_b = const(s) - 0.5 * sum((x - mu(s))^2 ./ sigma2(s))`

# Jak s vícepříznakovými vektory (3)

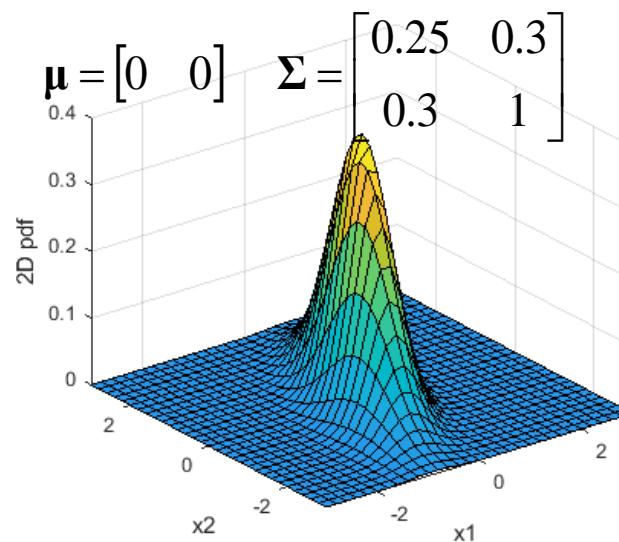
Illustrace multidimenzionálního Gaussova rozložení:

1D (pro 1 příznak)

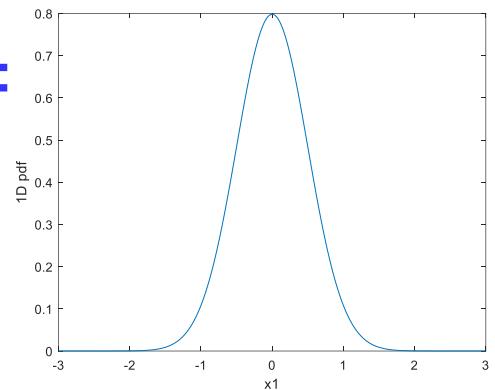
2D (pro 2 příznaky) - pdf lze vykreslit v 3D prostoru



nekorelované příznaky



korelované příznaky



Pro  $D > 2$  už nelze vykreslit, nanejvýš lze zobrazit pouze 2D řezy

# Jak s vícepříznakovými vektory (4)

## Výhody příznaků MFCC:

- jsou nekorelované, můžeme tedy pracovat s **vektorem** jejich rozptylů (místo celé matice),
- jejich **počet** může být výrazně **menší** (např. 12) než u spektrálních,
- díky tomu lze rozpoznávání (i trénování) výrazně **zrychlit**,
- jsou **méně závislé** na hlasových charakteristikách jednotlivých mluvčích,
- díky tomu jsou vhodné zejména pro **SI rozpoznávání**

# Jak volit počet stavů HMM

**Neexistuje jednoznačná odpověď, obvykle se hledá experimentálně.**

**Pozn.**

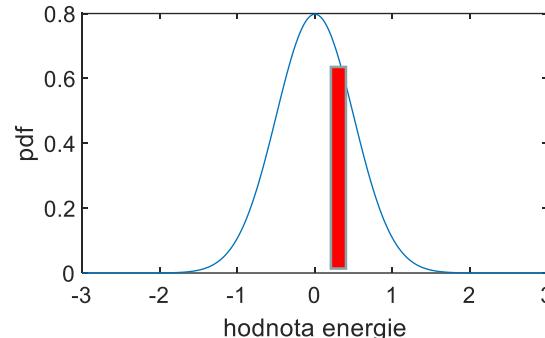
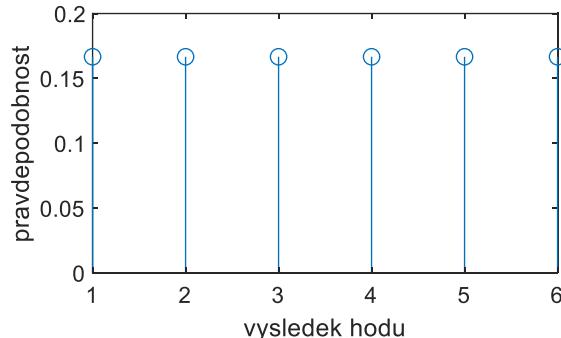
- HMM jsou natolik **flexibilní**, že se model naučí pro každé zvolené  $S$
- příliš malé hodnoty ( $S < 5$ ) povedou na horší modely, příliš velké hodnoty ( $S > 20$ ) už výsledky příliš nezlepší, rozpoznávání bude pomalejší
- vhodná volba je v rozsahu (8 – 20) a obvykle se volí stejné  $S$  pro modely všech slov
- $S$  nemůže větší než počet framů nejkratšího slova
- HMM lze používat i pro modelování krátkých zvuků (hlásek, ruchů, atd.), pak se obvykle volí  $S = 3$

# Poznámka k názvosloví a interpretaci

Pravděpodobnostní rozložení u diskrétních a spojitéch jevů:

Diskrétní jevy: např. hod kostkou, sportka

Spojité jevy: nabývají nekonečně mnoha hodnot



U spojitého jevu se pravděpodobnost určí jako

$$P(a \leq x \leq b) = \int_a^b f(x) dx$$

Věrohodnost: hodnota  $f(x)$  pro konkrétní  $x$

Vlastnosti věrohodnosti: souvisí s pravděpodobností, může být větší než 1

Jaký význam má výraz?

$$P(\mathbf{X}, \mathbf{M}) = b_1(x_1) \cdot a_{11}b_1(x_2) \cdot a_{12}b_2(x_3) \cdot a_{22}b_2(x_4) \dots$$

↑  
pravděpodobnost  
↑  
věrohodnost

Správnější pojmenování hodnoty  $P(\mathbf{X}, \mathbf{M})$  : pravděpodobnostní skóre (nebo jen skóre)

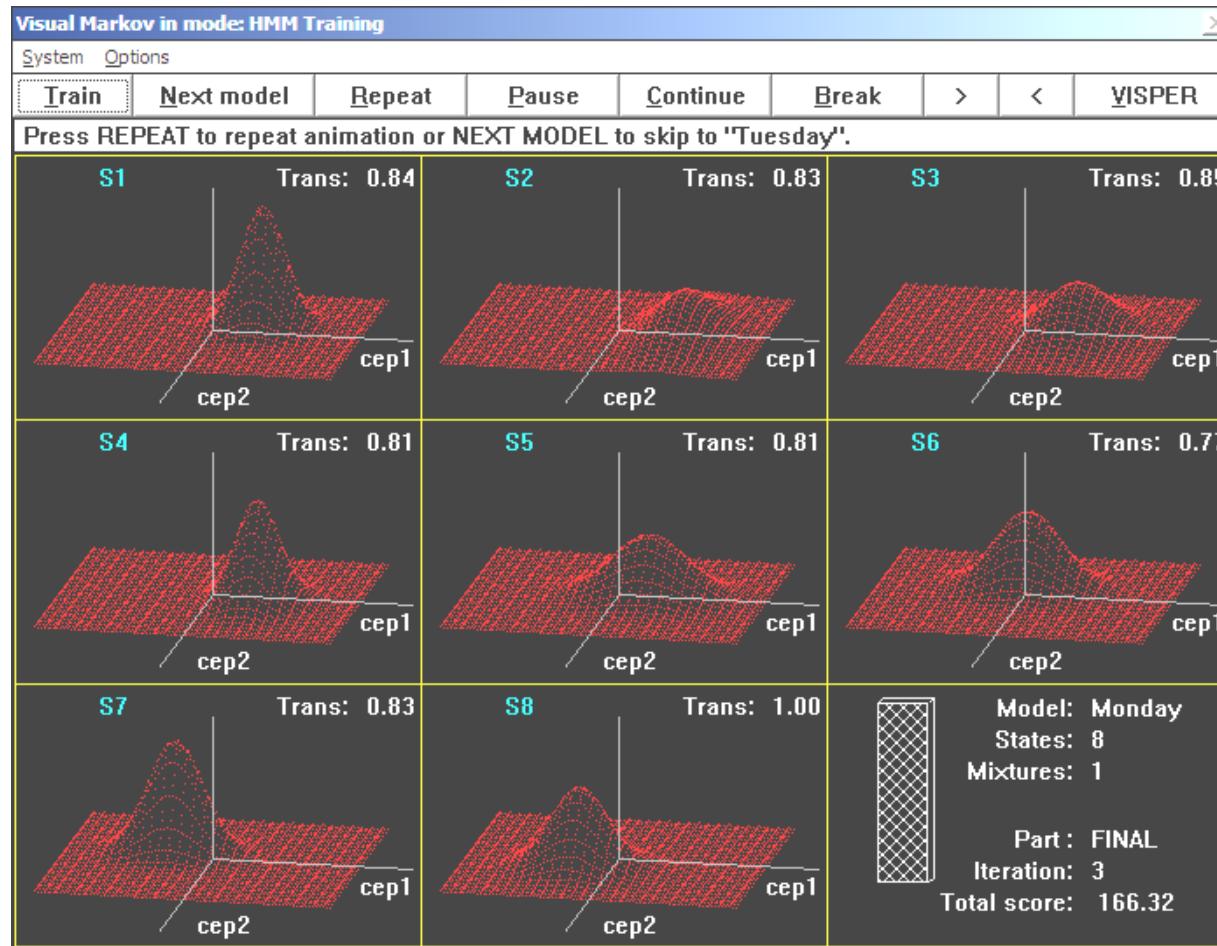
Pro log  $P(\mathbf{X}, \mathbf{M})$  je pak vhodné pojmenování: logaritmické (pravděpodobnostní) skóre

# HMM v prostředí VISPER

**VISPER podporuje HMM trénování i rozpoznávání.**

Navíc umožňuje zobrazovat tvar gaussovských rozložení i přechodové pravděpodobnosti v jednotlivých stavech během trénovacích iterací.

Na obr. je 8-stavový model slova Monday – jeho gaussovky ve 3D a hodnoty Trans (pravděp. setrvání v daném stavu)



# Co vás ještě čeká?

## Příště (9. týden):

Seznámení s prostředím HTK, které umožňuje práci s HMM pomocí skriptů volajících již naprogramované rutiny realizující parametrizace dat, trénování, testování, vyhodnocování, atd. (Toto prostředí budeme používat i v navazujícím předmětu.)

## 10. týden:

Shrnutí a diskuse poznatků.

Zadání závěrečných úloh. Vhodná téma si sami promyslete a navrhněte. Úlohy by měly zahrnovat jednoduché aplikace rozpoznávání řeči (izolovaných slov, v počtu 12 - 25), případně i syntézu, dialog. Můžete použít DTW nebo HMM.

## Udělení zápočtu a známek.

Na základě předvedení závěrečné úlohy, nejdéle do 15.6.

# Úloha pro cvičení

## Trénování a rozpoznávání s HMM

1. Na základě přednášky, dodaných kódů a předchozích úloh sestavte program v Matlabu, který zrealizuje trénování a rozpoznávání číslic pomocí HMM
2. Natrénujte a otestujte SD a SI modely na stejných datech jako v úloze k přednášce č. 6. Jako příznaky použijte MFCC12 a MFCC12+  $\Delta$ MFCC12.
3. Porovnejte výsledky a čas rozpoznávání HMM a DTW

# Tipy pro implementaci (1)

- Stáhněte si z elearningu kódy a data, které jsem pro vás připravil. Jsou celkem 3. Jeden realizuje trénování modelu, druhý Viterbiho algoritmus a třetí výpočet logaritmické hodnoty gaussovky.
- Žádný není úplný. V každém je část nahrazena symbolem XXX, za nějž musíte doplnit chybějící kód. Jsou to však relativně malé části programu, které byste měli s pomocí přednášky a komentářů u kódu bez problémů zvládnout. Doporučuji udělat to v následujícím pořadí:
  - Doplňte řádek v modulu **computeLogGauss.m** podle dnešní přednášky.
  - Dále doplňte chybějící část v modulu **trainHMM.m**. Projděte si podrobně vysvětlení v dnešní přednášce a porovnejte si kód s popisem a komentáři a doplňte. Správnou funkci si můžete ověřit na dodaných datech z minulé přednášky (natrénovaný model s 1 příznakem) a z dneška (natrénovaný model s příznaky MFCC). Oba byly trénovány s 10 iteracemi.
  - Dále si podrobně projděte modul **computeHMMViterbi\_fast.m**, porovnejte s minulým výkladem (a případně i s vaší minulou implementací) a doplňte chybějící část. Ověřte opět na dodaných datech.
  - Kompletní ověření můžete provést podle stejného schématu jako v mému programu **Prepare\_data\_for\_Viterbi\_test\_MFCC.m**

# Tipy pro implementaci (2)

- Až budete mít moduly hotové, musíte je vhodně začlenit do vašeho experimentálního programu z úlohy č. 6
- Ve vašem programu ponechte vše až po část testování. Nově musíte před testováním vložit část, která natrénuje všech 10 modelů (10 číslic) na trén. datech, a to buď v režimu SD nebo SI.
- Pro HMM doporučuji např. 10 stavů (později můžete zkusit zvýšit až na 20). Doporučuji 10 iterací trénování
- Pak můžete vložit část testování. Ta je jednoduchá, neboť každou nahrávku z dané testovací sady „projedete“ Viterbiho algoritmem postupně se všemi 10 natrénovanými modely a zjistíte, který dosáhl nejvyššího skóre.
- Očekávejte o trochu lepší výsledky než s DTW, ale mnohem rychleji.
- Pozn. Moduly jsou ode mne napsané tak, aby se daly začlenit do mého programu, kde některá data sdílí pomocí globálních proměnných. Toto si samozřejmě upravte tak, jak to vyhovuje vám.
- Programy, výsledky a stručný komentář očekávám do pondělní půlnoci.