VYSOKÉ UČENÍ
TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

Ústav radioelektroniky

Diplomová práce

magisterský navazující studijní obor
Elektronika a sdělovací technika

Student:	Bc. Jindřich Hampl	***ID:***	83779
Ročník:	2	***Akademický rok:***	2009/2010

NÁZEV TÉMATU:

Zpracování signálů pomocí skrytých Markovových modelů

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s metodami statistického zpracování číslicových signálů. Vytvořte přehledovou studii o dostupných softwarových nástrojích k danému účelu. Kontaktujte distributora softwarových produktů HTK a vyřiďte si jejich stažení a legální volné používání pro akademické účely. Naučte se pracovat se základním modulem HTK. Vypracujte stručnou českou verzi návodu na zpracování signálů pomocí skrytých Markovových modelů (HMM) s využitím modulu HTK. Vytvořte samostatně databázi vhodných akustických signálů, které budou použitelné jako vstupní signály pro ukázkové příklady. Vytvořte jednoduché ukázkové příklady na zpracování reálných signálů pomocí HMM. Příklady dokumentujte podrobným popisem tak, aby byl uživatel HTK schopen jejich samostatné reprodukce.

DOPORUČENÁ LITERATURA:

[1] PSUTKA, J., MÜLLER, Z., MATOUŠEK, J., RADOVÁ, V. Mluvíme s počítačem česky. Praha: Academia, 2006.

[2] YUNG, S., EVERMANN, G., GALES, M., HAIN, T., KERSHAW, D., MOORE, G. The HTK book. Cambridge: Cambridge Press, 2005.

Termín zadání:	8.2.2010	***Termín odevzdání:***	21.5.2010

Vedoucí práce: prof. Ing. Milan Sigmund, CSc.

prof. Dr. Ing. Zbyněk Raida
Předseda oborové rady

UPOZORNĚNÍ:

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami:

1. Pan/paní

Jméno a příjmení: Bc. Jindřich Hampl

Bytem: Mikulovská 173, Pardubice, 53002

Narozen/a (datum a místo): 18. března 1984 v Pardubicích

(dále jen „autor“)

a

2. Vysoké učení technické v Brně

Fakulta elektrotechniky a komunikačních technologií

se sídlem Údolní 53, Brno, 602 00

jejímž jménem jedná na základě písemného pověření děkanem fakulty:

prof. Dr. Ing. Zbyněk Raida, předseda rady oboru Elektronika a sdělovací technika (dále jen „nabyvatel“)

Čl. 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):

☐ disertační práce

☒ diplomová práce

☐ bakalářská práce

☐ jiná práce, jejíž druh je specifikován jako ..... (dále jen VŠKP nebo dílo)

Název VŠKP: Zpracování signálů pomocí skrytých Markovových modelů

Vedoucí/ školitel VŠKP: prof. Ing. Milan Sigmund, CSc.

Ústav: Ústav radioelektroniky

Datum obhajoby VŠKP: _____

VŠKP odevzdal autor nabyvateli*:

☒ v tištěné formě – počet exemplářů: 2

* hodící se zaškrtněte

☒ v elektronické formě – počet exemplářů: 2

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.

3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.

4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2

Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizovaní výpisů, opisů a rozmnoženin.

2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.

3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti

   ☒ ihned po uzavření této smlouvy

   ☐ 1 rok po uzavření této smlouvy

   ☐ 3 roky po uzavření této smlouvy

   ☐ 5 let po uzavření této smlouvy

   ☐ 10 let po uzavření této smlouvy (z důvodu utajení v něm obsažených informací)

4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3

Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.

2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.

3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.

4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: 21. května 2010

…….. ……

Nabyvatel Autor

ABSTRAKT

Jedna z nejpoužívanějších metod pro rozpoznávání řeči je založena na skrytých Markových modelech. Řečový signál můžeme považovat za sled po sobě jdoucích částí signálu s specifickými statistickými parametry. Skrytý Markovův model odpovídá statistickému modelu s konečným počtem stavů, který může být užitečný pro signály, jako je například řeč. Modul HTK je soubor programů, který je nejvíce používán pro práci se skrytými Markovovými modely.

KLÍČOVÁ SLOVA

Skryté Markovy modely, zpracování signálu, rozpoznávání spojité řeči, HTK.

ABSTRACT

One of the most common methods for isolated words recognition is based on Hidden Markov models. Speech signal can be considered as a sequence of successive parts of the signal with specific statistical parameters. Hidden Markov model corresponds to the statistical model with the final number of states, which may be useful for signals such as speech. HTK module is a software tools, which is mostly used to work with hidden Markov models.

KEYWORDS

Hidden Markov Models, signal processing, speech recognition, HTK.

HAMPL, J. *Zpracování signálů pomocí skrytých Markových modelů.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2010. 53 s. Vedoucí diplomové práce prof. Ing. Milan Sigmund, CSc.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma Zpracování signálů pomocí skrytých Markových modelů jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne 21. května 2010 .....

(podpis autora)

PODĚKOVÁNÍ

Děkuji vedoucímu diplomové práce prof. ing. Milan Sigmund, CSc. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

V Brně dne 21. května 2010 .....

(podpis autora)

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

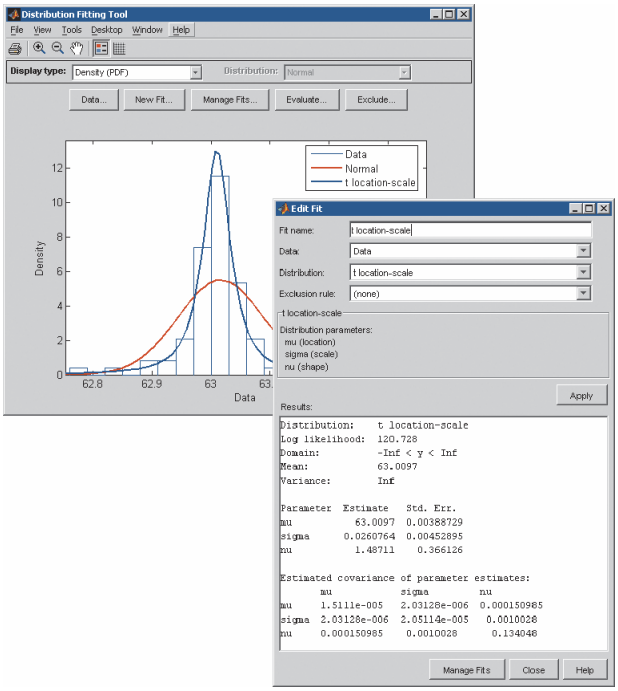
Systémy pro rozpoznávání a syntézu řeči (obecněji pro zpracování řečových signálů) jsou známé už několik desítek let. Rychlý rozvoj výpočetní techniky umožňuje vkládat novější, kvalitnější, ovšem výpočetně náročnější algoritmy k zpracování signálů. To je jen jeden z mnoha důvodů rozšíření metod zpracování řeči využívající skrytých Markových modelů (HMM). Skryté Markovovy modely odpovídají statistickým modelům s konečným počtem stavů. Tyto modely mohou být užitečné pro statistický popis po částech stacionárních signálů jako je řeč, což je případ i této diplomové práce. Rozpoznávání řečových signálů bude prováděno pomocí softwaru HTK (Hidden Markov Model Toolkit), jenž rozpoznává signály právě pomocí HMM. Ačkoli HTK nepatří mezi nejnovější software (byl vytvořen již v 90. letech), stále je vhodný pro rozpoznávání i díky mnoha aktualizacím. Skryté Markovovy modely jsou nazývány skryté, jelikož vzniká náhodný dvojitý proces a ne jen náhodná sekvence stavů. U skrytých Markovových modelů stavy generují výstupní vektor s nějakou určitou pravděpodobností. Výstupem celého prcesu rozpoznávání je sled vektorů. Jednotlivé stavy už není možné pozorovat. Cílem této diplomové práce je vytvořit jednoduché ukázkové příklady na zpracování reálných signálů pomocí HMM s využitím HTK v s obrazovým i písemným doprovodem v českém jazyce, tak aby každý (i začínající) uživatel byl schopen samostatně rozpoznávat vlastní (nebo předem připravené) zvukové signály v audio formátu wav. Rozpoznávací databáze nebudou velké (počet slov nebude v tisících), tudíž se projeví tato skutečnost pozitivně na datové velikosti databází přiložených na CD, rozpoznávání bude probíhat pomocí slov, ne pomocí hlásek, vyskytovat se zde budou zjednodušující mechanismy, které se projeví na procentuální úspěšnosti správného rozpoznávání. Cílem tedy není dosáhnout 100 % úspěšnosti rozpoznávání, jenž stejně je většinou nemožné dosáhnout, ale relativně vysoké procentuální úspěšnosti rozpoznávání slov při malém počtu příkazů, programů, s malou časovou náročností a bez nutnosti používat další externí software nebo alespoň software jenž je k dispozici v standardní instalaci operačního sytému Windows, v němž budou tato rozpoznávání (a všechny další kroky od nahrávání zvukového signálu až po vyhodnocení při rozpoznávání) provedena.

1 STATISTICKÉ ZPRACOVÁNÍ ČÍSLICOVÝCH SIGNÁLŮ – SOFTWARE

Nejvíce využívanými softwarovými nástroji pro statistické zpracování signálů jsou Matlab a Maple. Další softwarové nástroje nejsou tolik využívané jako výše jmenované. Matlab je mnohem výhodnější pro dané použití; už jen díky svým možnostem, ale i díky neustálému rozšiřování. Ovšem pro některé uživatele může být přespříliš složitým softwarem.

1.1 MATLAB

Oblíbený softwarový nástroj, použitelný pro statistické zpracování číslicových nástrojů, je Matlab, speciálně pak jeho softwarová nástavba Statistics Toolbox (nejnověji ve verzi 7.2). Tento software poskytuje komplexní sadu nástrojů pro hodnocení a umožňuje jednoduše porozumět údajům (výsledkům neboli výstupům). Obsahuje funkce a interaktivní nástroje pro modelování dat, analýzy historických trendů, simulace systémů, rozvoj statistických algoritmů a učení a výuky statistiky. Toolbox podporuje širokou škálu úkolů, od výpočtu základní popisné statistiky k rozvoji a vizualizaci multidimenzionality nelineárních modelů. Nabízí bohatou sadu statistických typů a interaktivní grafiky.



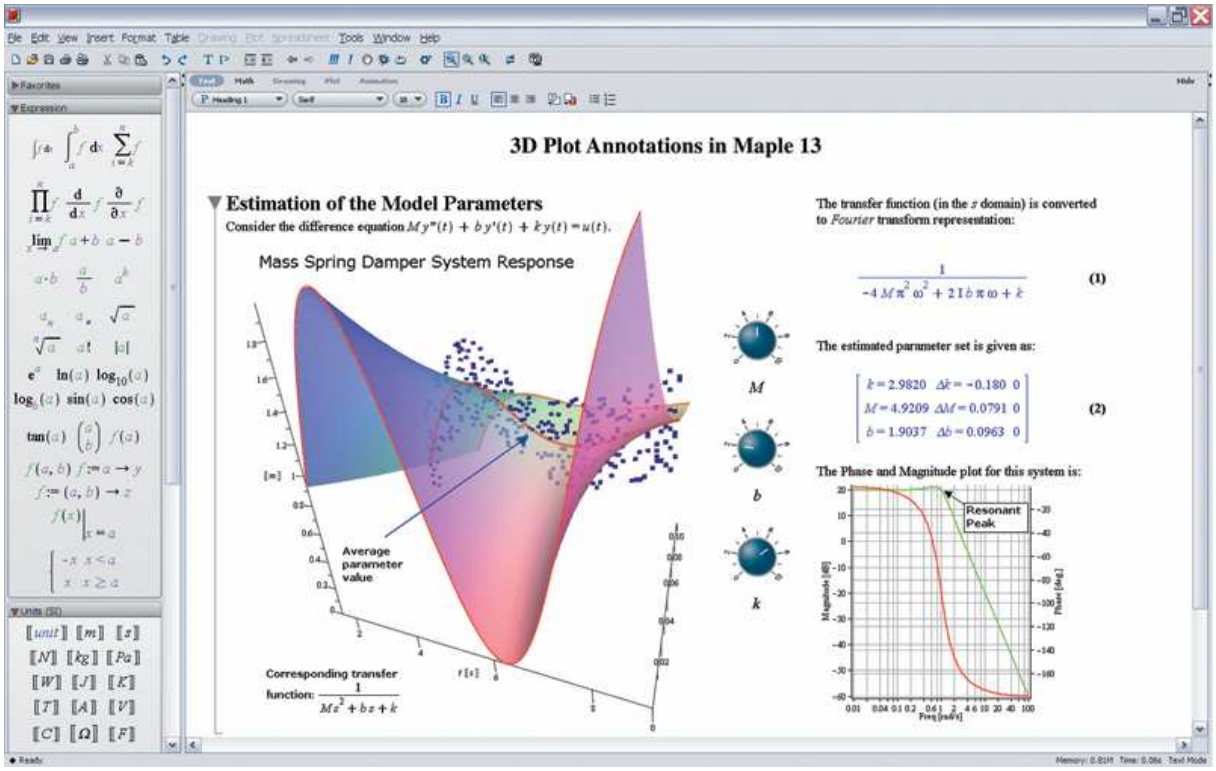
Obr. 1.1: Vizuální zobrazení Matlab Statistic Toolboxu (převzato z [32])

Všechny funkce toolboxu jsou napsány v otevřeném zdrojovém jazyce MATLAB®, takže si můžeme prohlédnout algoritmy, upravit zdrojový kód a vytvářet vlastní funkce.

Matlab je také často využíván na Vysokých školách, stejně tak i u mnoha firem. Jak vypadá vizuální okno toolboxu, je možno si prohlédnout na obrázku 1.1

1.2 MAPLE

Tento softwarový nástroj se nyní vydává ve verzi 13 (grafické okno zobrazeno na obr. 1.2) a je také oblíbený pro svojí jednoduchost. Ovšem více než pro statistické zpracování číslicových signálů se hodí pro zpracování matematických funkcí. Maple i Matlab spolu mohou spolupracovat, ovšem nepřímo, pomocí příslušného dalšího softwaru.



Obr. 1.2: Grafické menu programu Maple 13 (převzato z [31])

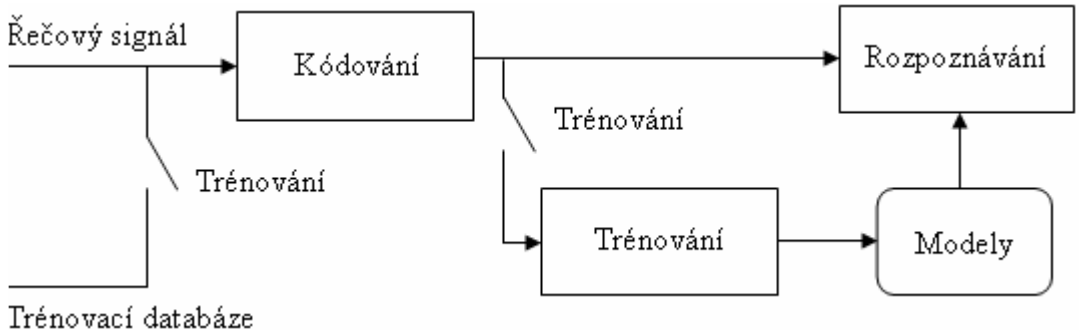
2 SKRYTÉ MARKOVY MODELY

2.1 Úvod do rozpoznávání řeči

Cílem předzpracování je zredukovat velké množství redundantních informací, obsažených v řečovém signálu a vybrat vhodné parametry pro jeho popis. Výstupem předzpracování je datový tok vektorů, popisující jednotlivé řečové rámce

$$\mathbf{O} = \mathbf{o}_1, \mathbf{o}_2, \mathbf{o}_3 ..... \mathbf{o}_T \tag{2.1}$$

O se nazývá pozorovanou promluvou, $\mathbf{o}_i$ jsou vektory pozorování a T je počet pozorování. Cílem rozpoznávání řeči je přiřadit pozorované promluvě text, informaci, kterou do ní řečník vložil, když ji vyslovil.

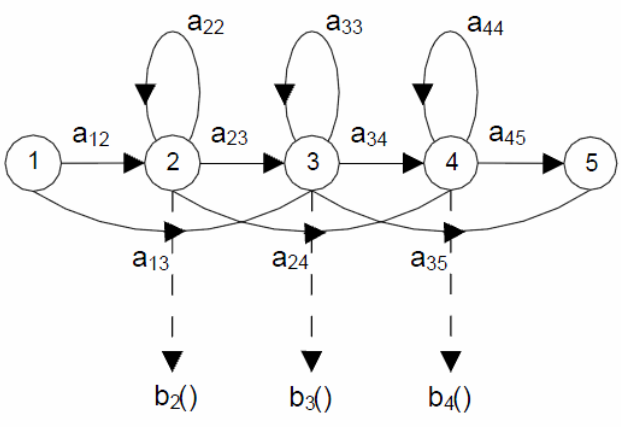


Obr. 2.1: Blokové schéma rozpoznávače

2.2 Architektura a popis skrytých Markových modelů

Skrytý Markovův model (HMM) odpovídá statistickému modelu nějakého automatu s konečným počtem stavů, který může být užitečný pro signály, jako je například řeč. Řečový signál je možno za určitého zjednodušení považovat za sled po sobě jdoucích kvazistacionárních částí signálu s specifickými statistickými parametry. Kvazistacionární segmenty signálu (nebo také jejich kombinace) mohou mít význam jednotlivých úseků řeči jako např. hláska či slovo atd. a reprezentují znaky nezbytné pro přenos určité informace. Když se přiřadí jednotlivé kvasistacionární segmenty řečového signálu příslušným modelům, které jsou zastoupeny jednotlivými stavy v HMM modelu, poté je možno zjistit (určit) informační obsah, jenž je obsažen v řečovém signálu.

Na obrázku 2.2 *je* a_{13} pravděpodobnost přechodu mezi 1. a 3 stavem. Pokud je výstupem skrytého Markovova modelu náhodný vektor, jehož pravděpodobnost závisí pouze na aktuálním stavu, pak výstupem celého procesu je sled vektorů a jednotlivé stavy nejsou pozorovatelné (jsou skryté).



Obr. 2.2: Příklad Markovového modelu s 5 stavy (převzato z [15]).

Algoritmy skrytých Markovových modelů jsou založeny na statistice. Skrytý Markovův model vychází z představy procesu, který přechází mezi stavy a po každém přechodu vygeneruje vektor pozorování. Matematicky je model popsán maticí přechodových pravděpodobností **A**, kde prvek a*ij* udává pravděpodobnost přechodu ze stavu *i* do stavu *j*, vektorem počátečních pravděpodobností **p** - pravděpodobností s kterými se bude proces vyskytovat na počátku generování ve stavu daném indexem vektoru, a pravděpodobnostními rozděleními generování vektorů pozorování stavy $b_j(\mathbf{o}_t)$. Pro řeč jsou používány tzv. levo-pravé modely, kdy matice přechodových pravděpodobností umožňuje pouze přechody do vyšších stavů nebo setrvání ve stavu stávajícím.

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{bmatrix}, \ \pi = \begin{bmatrix} \pi_1 \\ \pi_2 \\ \vdots \\ \pi_{1N} \end{bmatrix} \tag{2.2}$$

Pravděpodobnostní rozdělení $b_j(\mathbf{o}_t)$ může být diskrétní nebo spojité. Při použití spojitého rozdělení pravděpodobnosti, je zapotřebí jej aproximovat některými elementárními funkcemi. Nejčastěji je použita Gaussova funkce, popřípadě směs Gaussových funkcí.

Pravděpodobnostní rozdělení aproximované směsi Gaussových křivek je možno popsat následujícím vztahem:

$$b_j(\mathbf{o}(t)) = \sum_{m=1}^{M} c_m \aleph(\mathbf{o}(t); \mu_{jm}, \Sigma_{jm}), \tag{2.3}$$

kde $\aleph(\mathbf{o}(t); \boldsymbol{\mu}_{jm}, \Sigma_{jm})$ jsou jednotlivé směsové komponenty, c_j jsou váhy mezi nimi a M je jejich počet. Každá směsová komponenta vloží do pravděpodobnostního rozdělení jednu Gaussovu křivku pro každou dimenzi vektoru pozorování. Směsová komponenta je charakterizována vektorem středních hodnot (středů Gaussových křivek) $\boldsymbol{\mu}_{jm}$ a kovariační maticí Σ_{jm}. Vektor středních hodnot i kovariační matice jsou jak pro stav (j), tak pro směšovanou komponentu (m) unikátní.

$$\aleph(\mathbf{o}(t); \mu_j, \Sigma_{jm}) = \frac{1}{\sqrt{(2\pi)^P \left|\Sigma_{jm}\right|}} e^{-\frac{1}{2}(o(t)-\mu_{jm})^T \sum_{jm}^{-1} (o(t)-\mu_{jm})} \tag{2.4}$$

P je dimenze vektoru pozorování. Za předpokladu, že jednotlivé koeficienty vektoru pozorování nejsou mezi sebou korelovány, můžeme použít diagonální kovariační matici a urychlit tak výpočet. [34]

2.3 Trénování skrytého Markovova modelu

Trénování skrytého Markovova modelu spočívá ve stanovení všech parametrů. V odhadu přechodové matice a pravděpodobnostních rozdělení, tedy vektorů středních hodnot a kovariačních matic. Dále je možno předpokládat, že proces generování vektorů pozorování bude vždy začínat v prvním stavu a končit v posledním, proto vektor počátečních pravděpodobností bude mít první prvek jedničkový a ostatní nulové.

2.3.1 Inicializace modelu

Cílem inicializace je získání základních odhadů parametrů, které jsou potom pomocí iteračního algoritmu zlepšeny. Existuje několik přístupů k inicializaci. Nejednoduší je nastavení všech parametrů pravděpodobnostních rozdělení na stejné hodnoty odhadnuté ze všech trénovacích dat. Tento postup je použit hlavně při skládání promluv z krátkých jednotek, třeba fonémů, není-li známa přesně jejich pozice.

$$\hat{\boldsymbol{\mu}}_j = \frac{1}{T}\sum_{t=1}^{T}\mathbf{o}(t) \tag{2.5}$$

$$\hat{\boldsymbol{\Sigma}}_j = \frac{1}{T}\sum_{t=1}^{T}(\mathbf{o}(t)-\mu_j)(\mathbf{o}(t)-\mu_j)^T \tag{2.6}$$

Pokud je známo přesně, kde začíná v signálu úsek popisovaný trénovaným modelem, je možno použít důkladnější postup inicializace, který lze shrnout do několika kroků. Postup umožní získat lépe natrénované modely.
1. Vybrat z trénovacích dat data náležející trénovanému modelu.
2. Rozdělit každou pozorovanou promluvu lineárně *N* dílu, kde *N* je počet stavů modelu, a zařadit vektory z odpovídajících si dílů do stejných skupin. Pomocí každé skupiny vektorů bude odhadnuto pravděpodobnostní rozdělení v jednom stavu.
3. Rozdělit každou skupinu na *M* podskupin, kde *M* je počet směsových komponent. Při dělení by měla být brána v úvahu Euklidova vzdálenost mezi vektory, která by měla být minimální. K tomuto účelu lze použít libovolný algoritmus vektorové kvantizace.
4. Dle vztahů (2.5) a (2.6) vypočítat z každé podskupiny střední hodnoty a kovariační matice a těmi inicializovat parametry pravděpodobnostních rozdělení. Váhy mezi směsovanými komponentami jsou nastaveny podle počtu vektorů v podskupině. Jejich součet musí být jedna.
Inicializace přechodové matice není tak důležitá, všechny možné přechody mohou být nastaveny na stejnou pravděpodobnost [34]

2.3.2 Přetrénování modelu

K přetrénování skrytého Markovova modelu (také přeodhadnutí parametrů) se používá Baum-Welchův algoritmus. Jde o iterační algoritmus, který na základě parametrů modelu a pozorované promluvy nebo pozorovaných promluv vypočte parametry nové. Baum-Welchův

algoritmus pracuje v následujících několika krocích.

1. výpočet dopředných pravděpodobností
2. výpočet zpětných pravděpodobností
3. výpočet pravděpodobnosti s jakou vektor pozorování náleží stavu
4. akumulace pravděpodobností přechodu mezi dvěmi stavy
5.odhad nových pravděpodobnostních rozdělení na základě vektorů pozorování a pravděpodobností, s jakými náleží stavu.
6.odhad matice přechodových pravděpodobností na základě akumulovaných pravděpodobnostní přechodu mezi dvěmi stavy.

Výpočet dopředných pravděpodobností

Pro model ***M*** s *N* stavy je dopředná pravděpodobnost výskytu procesu v čase *t* ve stavu *j* definována jako:

$$\begin{aligned} &\alpha_1(1) = b_1(\mathbf{o}(1)) \\ &\alpha_j(1) = 0 \qquad\qquad pro\ 2 \le j \le N \\ &\alpha_j(t) = b_j(\mathbf{o}(t))\sum_{i=1}^{N}\alpha_i(t-1)a_{ij} \quad pro\ 1 \le j \le N \\ &\qquad\qquad\qquad\qquad\qquad\qquad 2 \le t \le T \end{aligned} \tag{2.7}$$

Kvůli přetečení registru bývají hodnoty α pro každý čas *t* normovány tak, aby jejich součet byl jedna. Stejný koeficient je použit na normování hodnot zpětných pravděpodobností pro odpovídající si čas.

Výpočet zpětných pravděpodobností

Pro model ***M*** s *N* stavy je zpětná pravděpodobnost výskytu procesu v čase *t* ve stavu *j* definována jako:

$$\begin{aligned} &\beta_N(T) = b_N(\mathbf{o}(T)) \\ &\beta_j(T) = 0 \qquad\qquad pro\ 2 \le j \le N-1 \\ &\beta_j(t) = \sum_{i=1}^{N} a_{ij} b_j(o(t+1))\beta_i(t+1) \quad pro\ 1 \le j \le N \\ &\qquad\qquad\qquad\qquad\qquad\qquad 2 \le t \le T-1 \end{aligned} \tag{2.8}$$

Výpočet pravděpodobnosti, s jakou vektor pozorování náleží stavu

Po výpočtu dopředných a zpětných pravděpodobností je možno přistoupit k výpočtu pravděpodobností, s jakými jednotlivé vektory náleží stavům. Získané hodnoty pomohou k odhadu nových parametrů pravděpodobnostních rozdělení

$$\gamma_j(t) = \frac{\alpha_j(t)\beta_j(t)}{\sum_{i=1}^{N}\alpha_i(t)\beta_i(t)} \quad pro\ 1 \le j \le N\ a\ 1 \le t \le T \tag{2.9}$$

Akumulace pravděpodobností přechodu mezi dvěmi stavy

$$\zeta_{ij}(t) = \begin{cases} \alpha_i(t)a_{ij}b_j(\mathbf{o}(t+1))\beta_j(t+1) & t \ne T \\ \alpha_i(t)a_{ij} & t = T \end{cases} \tag{2.10}$$

$\xi_{ij}(t)$ je pravděpodobnost, že proces vygeneruje první část vektoru pozorování **o**(l) ... **o**(t), v čase *t* bude ve stavu *i*, přejde do stavu *j* a dogeneruje zbytek vektoru pozorování **o**(t + l) ... **o***(T)*.

Nová přechodovou matici se získá sečtením výše uvedených pravděpodobností pro všechny časy, setříděním do matice a normováním jejich řádků na součet jedna. Postup lze popsat vztahem:

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T} \zeta_{ij}(t)}{\sum_{j=1}^{N}\sum_{t=1}^{T} \zeta_{ij}(t)} \tag{2.11}$$

$\hat{a}_{ij}$ jsou nové přechodové pravděpodobnosti a *N* je počet stavů modelu.

Odhad nových pravděpodobnostních rozdělení

Nejdříve pravděpodobnost s jakou vektory náležejí stavům, musí být rozdělena pro jednotlivé směsové komponenty dle následujícího vztahu.

$$\gamma_{jm}(t) = \frac{\gamma_j(t) c_{jm} \aleph(\mathbf{o}(t); \mu_j, \sum_{jm})}{b_j(t)} \tag{2.12}$$

Následně se odhadnou nové vektory středních hodnot a kovariační matice:

$$\hat{\boldsymbol{\mu}}_j = \frac{\sum_{t=1}^{T} \gamma_{jm}(t)\mathbf{o}(t)}{\sum_{t=1}^{T} \gamma_{jm}(t)} \tag{2.13}$$

$$\hat{\boldsymbol{\Sigma}}_j = \frac{\sum_{t=1}^{T} \gamma_{jm}(t)\left(\mathbf{o}(t) - \mu_j\right)\left(\mathbf{o}(t) - \mu_j\right)^T}{\sum_{t=1}^{T} \gamma_{jm}(t)} \tag{2.14}$$

Přepočet vah směsových komponent:

$$\hat{c}_{jm} = \frac{\sum_{t=1}^{T} \gamma_{jm}(t)}{\sum_{t=1}^{T} \gamma_j(t)} \tag{2.15}$$

2.3.3 Trénování spojeného modelu

Jednou ze základních vlastností skrytých Markovových modelů je možnost jejich spojování. Spojení lze provést sloučením přechodových matic modelů. Při trénování modelů dochází k situaci, kdy není známé přesné rozmístění jednotek, z kterých je promluva modelována. Pokud je známa posloupnost jednotek s jakou se v promluvě nalézají, lze jednotky zřetězit a vytvořit model celé promluvy. Tento model je poté trénován, a nalezení optimální pozice každé jednotky je přenecháno trénovacímu algoritmu. [34]

2.4 Určení pravděpodobnosti generování pozorovaných vektorů modelem

Při rozpoznávání řeči nejde o to, aby model generoval vektory příznaků, ale aby na základě pozorované posloupnosti vektorů se určila pravděpodobnost s jakou by ji model generoval. Obecně může proces procházet stavy modelu libovolnou cestou, která není vyloučena přechodovou maticí (přechod nemá nulovou pravděpodobnost). Pravděpodobnost průchodu procesu například stavy 1-2-2-3-4-4 a vygenerování vektorů pozorování $\mathbf{o}=[\mathbf{o}_1, \mathbf{o}_2, \mathbf{o}_3, \mathbf{o}_4, \mathbf{o}_5, \mathbf{o}_6]$ lze vypočítat následně:

$$P = b_1(\mathbf{o}_1)\, a_{12}\, b_2(\mathbf{o}_2)\, a_{22}\, b_2(\mathbf{o}_3)\, a_{23}\, b_3(\mathbf{o}_4)\, a_{34}\, b_4(\mathbf{o}_5)\, a_{44}\, b_4(\mathbf{o}_6)\ .$$

Při stanovení pravděpodobnosti s jakou by model generoval pozorovanou posloupnost vektorů, můžeme postupovat dvěmi způsoby. Můžeme uvažovat všechny cesty nebo si vybrat cestu jedinou, tu nejoptimálnější. Chceme-li respektovat všechny cesty, použijeme algoritmus dopředný nebo zpětný. Pravděpodobnost generování pozorované promluvy je dána koeficientem $\alpha_N(T)$ v případě dopředného algoritmu, nebo koeficientem $\beta_1(1)$ v případě zpětného algoritmu.

$$\begin{aligned} P(\mathbf{O}|M) &= \alpha_N(T) \\ P(\mathbf{O}|M) &= \beta_1(1) \end{aligned} \tag{2.16}$$

Pokud stačí jedna nejvíce pravděpodobná cesta, lze použit Viterbiho algoritmus. Pravděpodobnost se počítá stejně jako u dopředného algoritmu, jen suma je nahrazena nalezením maxima. [34]

2.5 Rozpoznávání plynulé řeči

K rozpoznávání plynulé řeči se používají tzv. dekodéry. Při inicializaci rozpoznávač nejprve vytvoří modely slov zřetězením menších jednotek (fonémů) a načte slovní síť (jazykový model). Slovní síť má za úkol omezit posloupnosti za sebou jdoucích slov na posloupnosti, které se v jazyce vyskytují a případně jednotlivým posloupnostem přiřadit jejich pravděpodobnosti. Dekodér na základě akustických dat a dat ze slovní sítě buduje řetězec možností. Čím více možností je prozkoumáno, tím je rozpoznávač přesnější. Velkým problémem je potřebná výpočetní náročnost algoritmu a velké paměťové nároky na uložení takového seznamu, proto je strom při vytváření také prořezáván a ponechány pouze kvalitní možnosti. Algoritmus, který se často používá pro rozpoznávání plynulé řeči, se nazývá token-passing (viz. [33]. Tento algoritmus používá k dekódování řeči tzv. žetonů. Žeton přechází od stavu ke stavu a cestou na sebe navazuje všechny pravděpodobnosti: pravděpodobnosti přechodů mezi stavy, pravděpodobnosti generování vektorů pozorovaní a pravděpodobnosti ze slovních sítí. V uzlových bodech je žeton zmnohonásoben a poslán do všech dalších možných cest nebo jen do *n* nejvhodnějších. Žeton, který dorazí do cílového uzlu a má největší pravděpodobnost, je určující, a z cesty, kterou prošel je zrekonstruována věta. Aby nedocházelo k přetížení počítače či zaplnění paměti, bývají žetony v určitých časech oproti sobě porovnávány a žetony s malou pravděpodobností jsou rušeny.

3 HTK

HTK (the Hidden Markov Models) je soubor programů, které jsou určeny k tvorbě a práci se skrytými Markovovými modely. Tento modul (toolkit), byl původně vyvinut ve Speech Vision and Robotics Group na Cambridgské univerzitě. Během let procházel postupným vývojem a dnes, kdy poslední aktuliazace byla vydána v roce 2009 a to verze 3.4.1, se jedná o velmi rozsáhlý a účinný nástroj k rozpoznávání spojité řeči. Modul HTK se pro práci s HMM osvědčil, a tak se dnes využívá i v mnoha jiných oblastech, ne jen v zpracování řeči. Algoritmy založené na skrytých Markových modelech nacházejí uplatnění při rozpoznávání řeči, jejich další využití můžeme nalézt při automatické tvorbě přepisů řečových nahrávek a při různých experimentech v oblasti zpracování řečových signálů.

Hlavní výhodou HTK je, že poskytuje široké spektrum funkcí a nástrojů při budování rozpoznávačů založených na skrytých Markových modelech. U jednotlivých nástrojů je zároveň poskytnuta velká variabilita jednotlivých nastavení. Jednotlivé nástroje HTK jsou ovládané příkazovým řádkem, což je už u menších rozpoznávačů nevýhodné. Nastavování a opětovné přestavování jednotlivých parametrů je jednak nepohodlné a jednak zabírá značný čas. Organizace souborů jako je změnách vstupních soborů, např. při testování, je při počtu desítek či stovek souborů příliš obtížné. Z uvedených skutečností pramení, že by bylo výhodnější mít vhodné grafické prostředí, které by práci s HTK zrychlilo, zefektivnilo a zpříjemnilo.

Tento softwarový nástroj byl navržen k rozpoznávání spojité řeči, ale díky tomu, že zvládne pracovat i s parametrickým popisem dat, se začal prosazovat i v jiných oblastech, při analýze prozodie řečníka, při tvorbě inventáře řečových jednotek pro syntézu řeči, při zpracování obrazových dat a jiné. Modul HTK je možno použít ve všech fázích návrhu rozpoznávače řeči. Může spočítat parametry ze zvukových dat, vytváří strukturu modelu HMM, inicializuje a trénuje modely, rozpoznává řečové nahrávky a analyzuje přesnost rozpoznávání. Značná část programu je také určena k práci s jazykovými modely. Pokud bychom chtěli detailnější popis postupu návrhu rozpoznávače spojité řeči, rozdělili bychom jej do čtyř fází: příprava dat, trénování, testování a vyhodnocení

3.1 Příprava dat v HTK

Modul HTK při trénování využívá parametrický popis dat. Muže se jednat například o melovské frekvenční kepstrální koeficienty (MFCC – vysvětlení viz [19]), lineární predikční koeficienty (LPC – podrobněji viz [20]) nebo percepční lineární predikční koeficienty (PLP; viz [21]). Nejprve je třeba tyto parametry ze zvukových nahrávek vypočítat. K tomu slouží nástroj HCopy (viz tabulka [2.1]), který tyto parametry ze zvukových souboru typu *.wav, *.raw nebo jiných zvukových formátu vypočítá. Pomocí nástroje HList (viz tabulka [3.1]) je poté možné vypsat obsahy těchto souborů. Pokud k trénovacím datům již existují transkripce (jak na úrovni slov nebo na úrovni hlásek), je potřeba tyto transkripce konvertovat do formátu čitelných modulem HTK. K tomuto účelu slouží např. nástroj HLEd (viz tabulka [3.1]), pomocí něhož je vytvořen rovněž z několika transkripcí jeden Master Label File (viz tabulka [3.1]), který je pak výhodnější pro další zpracování. Jestliže je možnost pořídit vlastní nahrávky, využívá se také nástroj HSLab (viz tabulka [3.1]), který rovněž umožní nahrávky ihned označit. [1]

Proces rozpoznávání se dělí na dvě části. Nejdříve je potřeba navrhnout vhodnou strukturu modelu HMM (určit počet stavů, inicializovat matice přechodových pravděpodobností apod.) a dále natrénovat na patřičném množství trénovacích dat jednotlivé modely. Modul HTK k zjištění parametru modelu HMM používá Baum-Welchův algoritmus (viz [23]). V dalším kroku je provedeno samotné rozpoznávání, přičemž modul HTK pro tyto účely využívá token passing algoritmus (algoritmus putující značky; více viz [22]).

3.2 Trénování

Při trénování je definována topologie každého modelu HMM a rovněž nastavena počáteční hodnoty matice přechodových pravděpodobností. V prvním cyklu se data rovnoměrně rozdělí a vypočítají se střední hodnoty a rozptyly. V dalších cyklech je rovnoměrné rozdělení nahrazeno Viterbiho algoritmem. Během inicializace je z trénovacích dat první sada parametru vypočítána nástrojem HInit. K dalšímu přetrénování se pak používá nástroj HERest, který využívá Baumuv-Welchuv algoritmus. [1]

3.3 Rozpoznávání

Pro rozpoznávání používá modul HTK jeden ze tří programů: HVite, HLRecsore a HDecode. HVite (vysvětlení všech funkcí viz tabulka [3.1]) používá k rozpoznávání již zmíněný token passing algoritmus. Na vstup programu se předává síť možných slov, která se v řeči mohou vyskytovat a pouze tato slova se mohou v řeči objevit. Dále je nutné uložit slovník s fonetickou transkripcí jednotlivých slov a nakonec sadu modelů HMM. HVite pak dokáže na základě těchto informací určit hranice slov v řečových datech, která jsou buď uložena na disku nebo která přichází přímo ze zvukové karty. Síť možných slov lze vygenerovat automaticky např. pomocí programu HBuild nebo HParse (obojí viz tabulka [3.1]). Pokud máme vytvořenou sít, můžeme nástrojem HSGen (viz tabulka [3.1]) vygenerovat náhodné posloupnosti slov, která se mohou na vstupu vyskytovat. Pořadí výskytu těchto slov se však může řídit některými pravidly. Toto pořadí muže sloužit ke kontrole správně vytvořené sítě neboť tyto věty se poté mohou použít k nahrání nových řečových dat. [1]

3.4 Vyhodnocení

Jakmile je rozpoznávač připravený, je provedeno testování a analýza rozpoznávání na testovacích datech. Testovací nahrávky by měly být označené na úrovni hlásek, nicméně může postačit i transkripce na úrovni slov. Nejdříve je nutno nahrávky zpracovat natrénovanými modely a poté je zapotřebí zjistit úspěšnost rozpoznávání na základě statistik získaných nástrojem HResults (viz tabulka [3.1]). Ze statistik se zjistí počet správně rozpoznaných slov (nebo delších úseků) a počet chyb vzniklých substitucí, vynecháním nebo přidáním. [1]

3.5 Další možnosti zlepšení rozpoznávání

Obecně se doporučuje používat při trénování co největší objem trénovacích dat. Dobrý rozpoznávač tak muže být natrénovaný až na desítkách hodin řeči. Nicméně problém nastává při získávání tak dlouhých nahrávek. To je dosti časově náročná a také nákladná práce. Je možné například nahrávky pořídit z rozhlasu či televize při vysílání zpravodajských poradu, nicméně tyto nahrávky nejsou tak kvalitní jako nahrávky, které byly pořízené v bezodrazové komoře, což muže být nežádoucí při využití rozpoznávače pro syntézu řečového signálu. Na druhou stranu, někdy jsou příliš kvalitní nahrávky nežádoucí, protože neobsahují šumový signál pozadí a znějí nepřirozeně.

3.6 Oblasti využití HTK

Modul HTK byl vyvinut především k rozpoznávání spojité řeči nebo izolovaných slov, nicméně během několika let vývoje se z tohoto souboru programu stal velmi účinný nástroj správy modelu HMM, a proto se modul HTK začal nasazovat i v jiných oblastech, kde se skryté Markovy modely využívají. Ne vždy je ovšem výhodné použít tento dosti komplikovaný modul. Příkladem muže být rozpoznávání statických obrazů, kde stačí použít jednodušší metody. [1]

Tudíž se může používat i v genovém inženýrství, ekonomice, v robotice a všude jinde, kde se modely HMM aplikují [14].

3.6.1 Zpracování řeči

V poslední době je vyvíjen software, který bude schopen automaticky vygenerovat titulky k mluvené řeči nebo videozáznamu. Jisté úspěchy byly zaznamenány při nasazení těchto programů na různé projevy či diskuze např. z parlamentu, Evropského soudu atd. Takový program muže například na základě obličeje mluvčího a jeho prozodických charakteristik zjistit z databáze jeho jméno a dále muže k tomuto jménu přikládat jeho přeloženou pasáž v řečové nahrávce. Tyto systémy jsou dnes schopny pracovat i v reálném čase [6].

Automatické vytváření titulků má i velké uplatnění v různých databázových systémech, kde můžeme na základě různých hesel vyhledávat v nahrávkách určité pasáže, které nás zajímají, a nemusí se tak přehrávat celé sekvence. Toho chce v budoucnu využívat např. společnost Google Inc., která by v nahrávkách umístěných na Internetu vyhledávala stejně jako v textových souborech [7].

Další oblastí, kde se modul HTK využívá, je analýza prozodie. Tato analýza se například používá v jazykových výukových systémech, kde systém přehraje nahrávku a člověk vyslovenou frázi znovu zopakuje. Systém pak zaznamenanou řeč analyzuje a rozhoduje, zdali byla použita správná intonace, správný přízvuk atd. Na základě těchto informací pak muže vyzvat člověka k zopakování fráze nebo upozorní, kde dělá člověk chybu. Podobně se mohou chovat i programy, které slouží jako pomůcky pro logopedy. [8]

Na základě změn hlavních kmitočtů základního tónu, intenzity a trvání řeči je možné určovat emoce řečníka, jako např. strach, úžas atd. [9].

Stejně tak je realizovatelné na základe těchto prozodických vlastností zjišťovat, o kterého řečníka se jedná. Tohoto faktu je možno využívat v různých identifikačních systémech. Podobné principy muže rozpoznávač využívat také k detekci řečové aktivity. [10]

3.6.2 Zpracování obrazu

I když byl zprvopočátku modul HTK určen především ke zpracování řeči, používá parametrickou reprezentaci dat a tudíž je v podstatě jedno, zda modely HMM jsou trénovány na sadě parametrů vypočítaných z akustických nebo obrazových dat. Takové netradiční použití modulu HTK je např. při rozpoznávání gest. V tomto případě snímají systémy pohyb předmětu a na základě křivek, které předmět při pohybu kopíruje, provádí různé operace. Toho lze například využít u tzv. bezdotykového psaní nebo kreslení. V tomto případě je uložen povel systému, který objekt má sledovat, např. konec prstu nebo zelený míč. Systém pak pohyb tohoto objektu snímá a v případě kreslení převádí přímo do křivek v některém grafickém editoru nebo v případě psaní rozpoznává, která písmena pohyb kopíruje [11], [12].

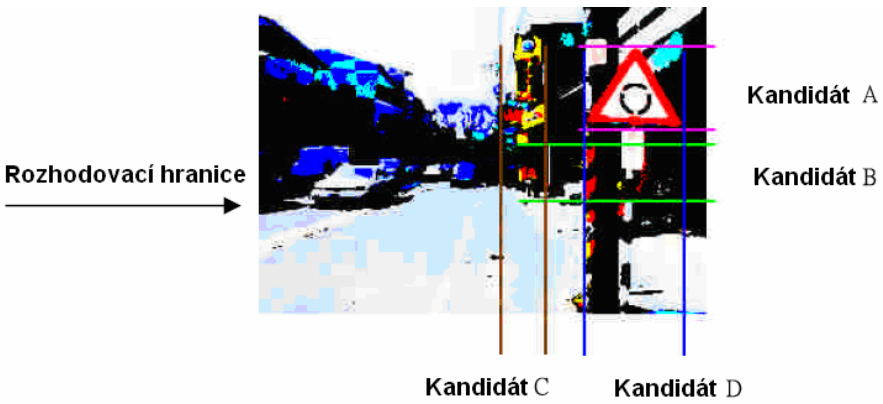
Vícestavové modely HMM se mohou nasadit k odezírání ze rtu. Jestliže si totiž rozdělíme videosekvenci, při které došlo k vyslovení některé hlásky, na určitý počet snímku, pak můžeme snímky přidělit jednotlivým stavům modelu HMM a na základě posloupnosti snímku rozhodnout, o jakou hlásku se jedná. V podstatě sledujeme, v jaké poloze se ústa řečníka na snímku nachází. Modul HTK je možné rovněž využít k rozpoznávání statických obrazů. [13]

3.6.3 Lékařství

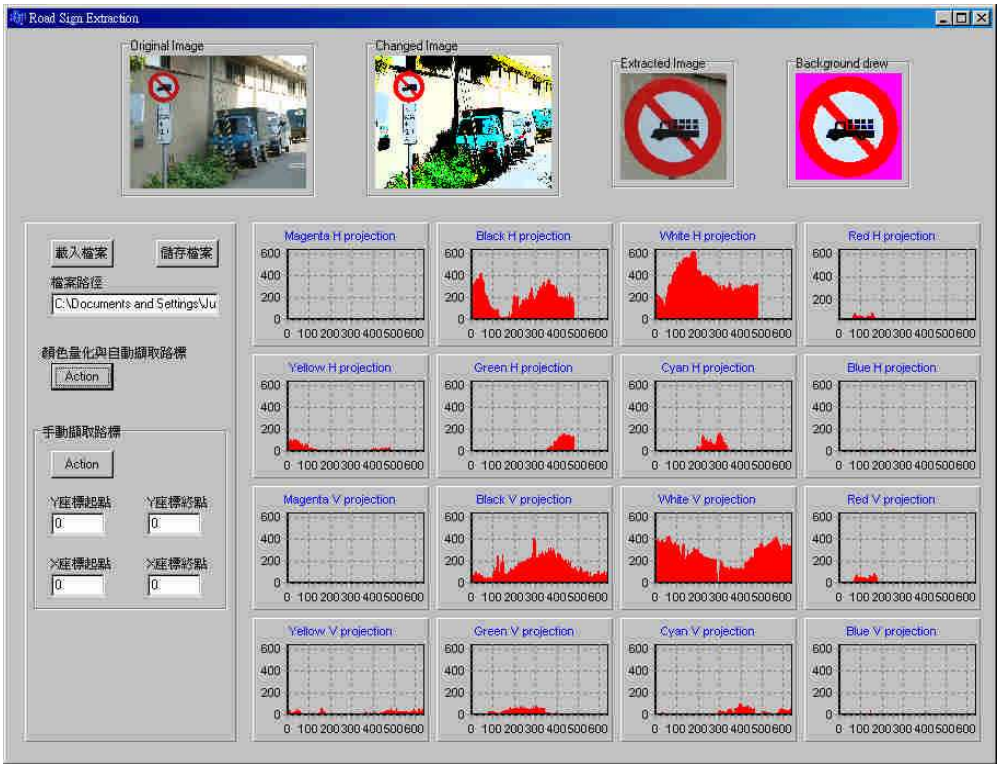
V lékařství se může využívat HTK pro automatický poslech (automatické vyšetření sluchem) nebo další lékařské postupy (související s rozpoznáváním hlasu, řeči či zvuků).

3.6.4 Automobilová doprava

Jak již bylo v úvodu řečeno, modul HTK lze nasadit kdekoliv, kde se používá parametrický popis dat. V automobilové a nákladní dopravě (na dálnicích; např. v Německu) je možno najít několik příkladů využití HMM jako např. navrhování značek na silnici v reálném čase pomocí obrazů z kamery. Tento typ návrhu je realizován pomocí kamer, jenž snímají prostředí za značkou a podle okolního prostředí určuje typ značky. Vyhodnocování je ukázáno na obr. 3.1, kde jsou vidět dobře hraniční čáry, jenž má nastaven program pro určení, jaký typ značky má použít. Na obrázku 3.2 je vidět softwarově rozložení pravděpodobností a následné zobrazování značky pomocí softwarového nástroje. Tento softwarový nástroj se používá v Kanadě a USA.



Obr. 3.1: Ukázka způsobu vyhodnocování typu dopravních značek (převzato z [29]).



Obr. 3.2: Ukázka z programu pro vyhodnocování dopravních značek (převzato z [29]).

3.7 Přehled jednotlivých dílčích programů modulu HTK

Modul HTK obsahuje mnoho programů, jenž slouží k trénování modelu HMM, rozpoznávání, adaptaci, k tvorbě jazykových modelu atd. V tabulce 3.1 je uveden přehled nejdůležitějších programů a jejich funkcí dostupných v modulu HTK v nejnovější dostupné verze 3.4.1. Nápovědu k těmto programům je možné obdržet zadáním názvu do příkazové řádky nebo vyhledáním v manuálu modulu HTK. [2]

Tab. 3.1: Přehled dostupných programů balíku HTK (převzato z [1]).

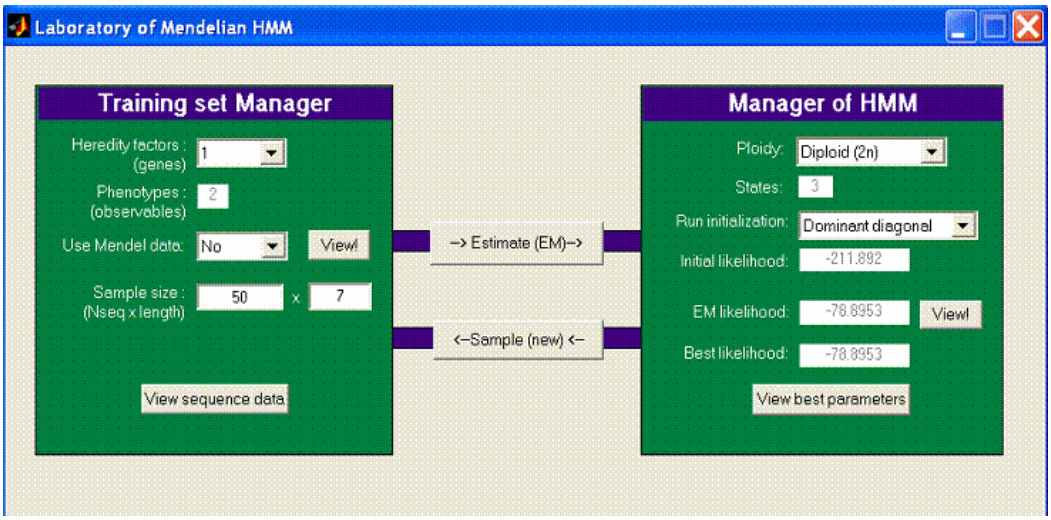
Název programu	Funkce
HBuild	Konvertuje vstupní soubory reprezentující jazykové modely
	do několika různých formátů a vrací standardní HTK mřížku.
HCompV	Vypočítává střední hodnoty a kovarianční matice z množiny
	trénovacích dat.
HCopy	Provádí konverzi formátu vstupních dat (nejčastěji z dat vypočítává
	parametry).
Hdman	Vytváří slovník fonetické transkripce, čitelný v modulu HTK,
	z několika zdrojů.
HDecode	Rozpoznávač využívající velký slovník.
HRest	Přepočítává parametry sady HMM pomocí Baumova-Welchova algoritmu.

HHEd	Skripty ovládaný editor, který modifikuje sady HMM.
HInit	Provádí inicializaci jednotlivých modelů HMM na základě posloupnos-
	tí pozorování.
HLEd	Jednoduchý editor určený k práci se soubory, které označují nahrávky.
HList	Zobrazuje soubor s různými parametry
HParse	Konvertuje lidsky čitelnou formu rozpoznávací sítě do nečitelné HTK podoby.
HLMCopy	Slouží ke kopírování jazykových modelů, během kterého může dojít
	k různým modifikacím.
HLStats	Vypočítává různé statistiky, které se využívají při analýze akustických
	trénovacích dat, nebo při tvorbě jednoduchých jazykových modelů
HMMIRest	Slouží k přetrénování sady modelů HMM, při kterém využívá
	rozdílná trénovací kritéria.
HQuant	Vytváří tabulky zpracovatelné moduly HTK, které obsahují kódové
	knihy, z nichž každá odpovídá jednomu datovému toku.
HRest	Přepočítává parametry jednotlivých modelů HMM pomocí Baum-
	Welchova algoritmu.
HResults	Pomůcka pro vyhodnocení výsledků rozpoznání – na základě správných přepisů rozpoznaných slov vyhodnocuje chybovost.
HSGen	Načítá síť slov ve standardním SLF formátu a na základě této sítě
	generuje náhodné věty.
HSLab	Interaktivní editor sloužící ke značení zvukových souborů.
HSmooth	Vyhlazuje Gaussovské směsi kontextově závislých nebo diskrétních
	modelů HMM.
HVite	Viterbiho dekodér neboli rozpoznávač. Pro natrénované modely a neznámá slova provede výpočet Viterbiho pravděpodobností a výběr maxima. Model, který “vyslal” slovo s největší pravděpodobností, je rozpoznán..
LAdapt	Adaptuje existující jazykový model z dodaných textových dat.
LGCopy	Kopíruje jeden nebo více souborů s gramatikou do jedné nebo více sad
	výstupních souborů.
LGList	Vypisuje obsah souborů s gramatikou pro jazykové modely.
LGPrep	Prohledává textové soubory určené k trénování jazykových modelů
	a generuje sady gramatik.
LLink	Vytvoří soubor s odkazy potřebnými k práci s jazykovými modely.
LMerge	Kombinuje jazykové modely a vytváří model určený pro specifický
	slovník.
LNorm	Normalizuje jazykové modely.
LSubset	Porovnává slovní mapy s mapami tříd a vytváří novou mapu slov,
	která obsahuje slova z mapy tříd.

4 DALŠÍ PROGRAMY PRO ROZPOZNÁVÁNÍ POMOCÍ HMM

4.1 Mendel HMM Toolbox

Pokud je zapotřebí používat HMM v softwaru Matlab, nejjednodušší cestou je Mendelův toolbox pro Matlab.



Obr. 4.1: Pracovní okno programu Mendelian HMM (převzato z [16]).

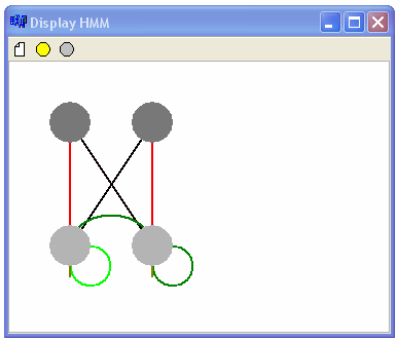
Tento software umožňuje v tréninkovém nastavovacím manažeru jednoduše nastavit vzorkovací velikost jako násobek počtu sekvencí a délky, poté je možno užít Mendelova data, zvolit počet fenotypů (viz [24]) či počet genů (dědičnosti). V manažeru HMM je možno nastavit buď Diploid (viz [25]) - 2n mód nebo tetraploid (viz [26]) - 4n mód, počet stavů, inicializaci a poté spočítat inicializační pravděpodobnost, odhadovanou pravděpodobnost a nejlepší pravděpodobnost výskytu. Mezi jednotlivými manažery je možno si zvolit odhad nebo vzorkování. Toto jsou možnosti uvedeny v grafickém editoru (zobrazeném na obrázku 4.1), v prostředí Matlab můžeme tento program dále upravovat [16].

4.2 HMMSim

Pro definici, trénování a simulaci diskrétních skrytých Markových signálů se používá také výukový softwarový nástroj HMMSim. Tento nástroj vznikl ve Španělsku a vyznačuje se velkou jednoduchostí. Tento software , jehož grafické příkazové okno je na obr. 4.2, dovoluje vytvořit HMM pro daný typický rozpoznávací problém, cvičit HMM jeho parametry podle výcvikového procesu, a simulovat chování implementovaný HMM.

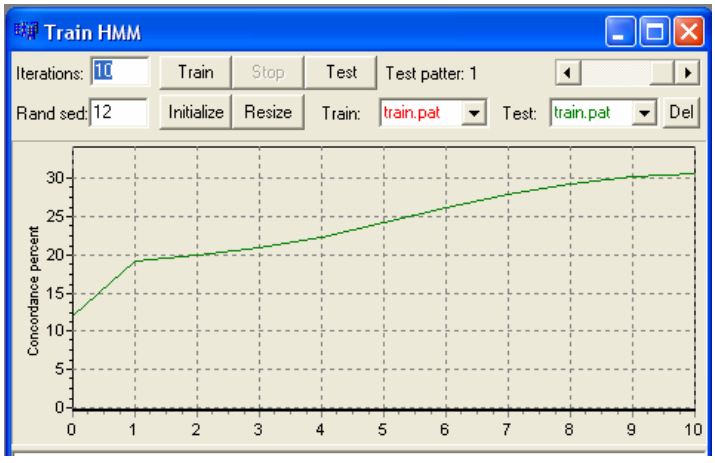
Obr. 4.2: Pracovní okno HMMSim s vysvětlením funkcí jednotlivých tlačítek

Obrázek 4.3 ukazuje displej neboli vizualizační okno pro definovaný HMM. Toto okno rozhraní je navrženo k tomu, aby reprezentovalo levo-pravé modely sekvenční HMM, které jsou většinou použity v typických rozpoznávacích aplikacích. Pravděpodobnosti přechodů mezi různými stavy obrazce jsou reprezentované shodně s barevným kódem.



Obr. 4.3: Pracovní okno HMMSim zobrazující stavy a jejich přechody

Na obrázku 4.4 je zobrazeno trénování HMM pomocí natrénovaného modelu, počtu vzorků, počtu iterací (v tomto případě 10), je zde možno i graficky zachytit průběh natrénovaného modelu.



Obr. 4.4: Trénovací rozhraní programu HMMSim

Tento program je velmi jednoduchý, bohužel ale ve free verzi neumožňuje využít některých vlastností programu, jako je např. stopnutí, trénování HMM nebo uložení HMM nebo uložení kódu v jazyce C++.

4.3 Voicebox

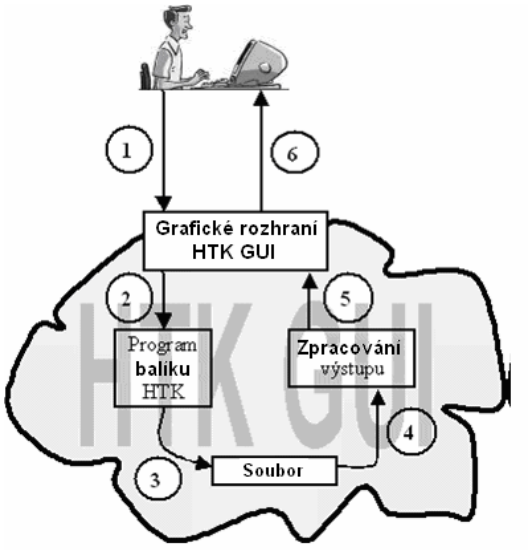
Tento nástroj byl vytvořen pro použití HTK v prostředí Matlab, jelikož unixové prostředí není tak uživatelsky přívětivé a rozšířené jako výše zmiňovaný program využívající Mendelův Toolbox. Tento nástroj je dostupný na [17] v formě matlabovských souborů s příponou .m a umožňuje ve většině případů provést to stejné, co software HTK.

4.4 HTK GUI

Tento program byl napsán v Slovensku v Bratislavě a cílem tohoto nástroje bylo vytvoření grafického user-friendly prostředí, které by poskytovalo dostatečný komfort při práci s balíkem HTK, který samotné HTK nenabízí. Výsledkem je popsaná aplikace HTK GUI naprogramována v prostředí C++ Builder a pracující pod operačním systémem MS Windows. Do aplikace není zahrnuto rozpoznávání naživo, například. z mikrofonu, který HTK podporuje. [18]

4.4.1 Základní struktura HTK GUI

HTK GUI je grafické rozhraní pro sadu HTK. Eliminuje nedostatky ovládání HTK a dělá používání HTK jednodušším. Komunikace grafického prostředí s balíkem HTK je znázorněna na následujícím obrázku (obr. 4.5).



Obr. 4.5: Princip spolupráce Uživatel-HTK GUI-HTK

Jak je vidět z obrázku, uživatel požaduje vykonání příkazu od HTK GUI (1). HTK GUI volá program sady HTK (2), aby provedl příslušnou činnost. Výstup z programu je přesměrován do souboru (3) a následně je zpracován (4). Zpracovaný výstup je pomocí grafického rozhraní

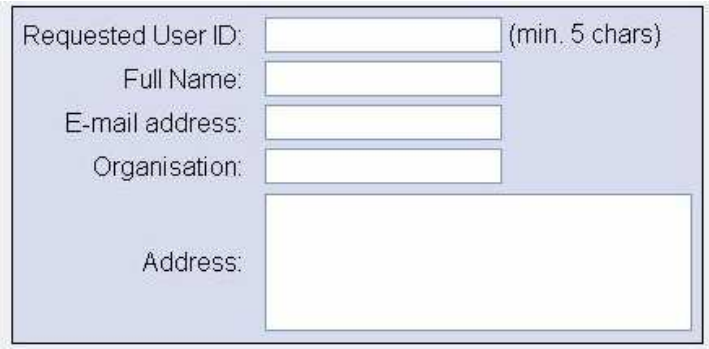
(5) nabídnut uživateli (6). Všechny údaje jsou organizovány (v případě, že si to uživatel přeje) tak, aby jednotlivé fáze vývoje rozpoznávání na sebe navazovaly, tj. aby program jako výstupní data předešlé fáze nabízel jako vstupní data v další fázi. Uživatel tudíž nemusí vést organizaci dat a nastavovat znovu a znovu např. vstupní soubory pro určitou činnost.

4.4.2 Základní popis práce s programem

Po spuštění aplikace se v horní části zobrazí nabídka pro práci s programem. Jádrem aplikace je tzv. projekt. Po vytvoření projektu je uživateli nabídnut průvodce, pomocí kterého je možné rychle a jednoduše připravit data, modely, podpůrné položky a následně rozpoznávání. Uživatel je průvodcem veden přes jednotlivé fáze vývoje rozpoznávání, které jsou realizovány v formulářích (např. formulář pro parametrizaci vstupních dat, vektorové kvantizace dat atd.), což zajišťuje pohodlné ovládání. Zkušenější uživatel může používat jednotlivé části projektu samostatně a nemusí používat průvodce. To je vhodné např. při testování vlivu různých nastavení na celkové výsledky, kterých lze dosáhnout při rozpoznávání. Pokud uživatel potřebuje například pouze zparametrizovat údaje pro pozdější použití, je to umožněno v podobě utilit, které jsou dostupné z hlavního menu, bez nutnosti otevírat projekt. Tento program umožňuje parametrizovat údaje, vektorově kvantovat, tvořit spojitý prototyp, trénovat modely, tvořit gramatiku, slovník, popisné soubory a rozpoznávat. Výhodou oproti používání HTK pomocí příkazové řádky je, že se nemusí ručně definovat seznamy souborů a modelů, které jsou při HTK nutností a celý proces trénování se tak několikanásobně urychluje.

5 NÁVOD NA STAŽENÍ MODULU HTK

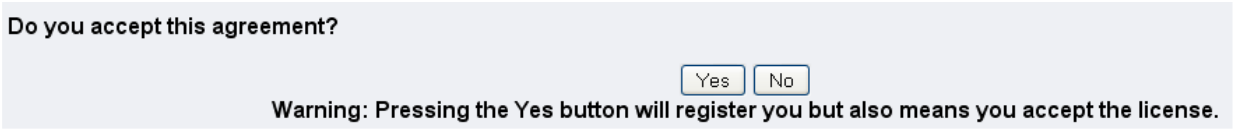
Nejdříve je zapotřebí do internetového prohlížeče (IE, Firefox, Opera, atd.) napsat: http://htk.eng.cam.ac.uk/. Zde v levém sloupci je položka Register (pod Getting HTK nebo můžeme rovnou napsat do prohlížeče http://htk.eng.cam.ac.uk/register.shtml), na tuto položku je potřeba kliknout a bude zobrazen registrační formulář, kde se vyplňují údaje zobrazené na obrázku 5.1:



Obr. 5.1: Formulář pro registraci uživatele

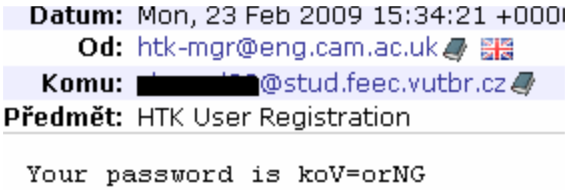
Do pole v první řádce vedle nápisu *Requested USER ID:* je nutno vepsat identifikační jméno o minimálně 5 znacích. Napravo vedle nápisu *FULL NAME:* se musí napsat do bílé kolonky jméno a příjmení. Do dalšího (3. řádku) v formuláři je nutno napsat vedle nápisu *E-mail address:* e-mailovou adresu, čili na jakou adresu se pošle heslo a poté vedle položek *Organization:* a *Address*: nejdříve název organizace a vedle nápisu *Address:* napravo do bílé kolonky adresu.

Ještě je nutno po vyplnění těchto údajů souhlasit s licenčními podmínkami a tedy kliknout úplně dole na uvedené stránce pod otázkou *Do you accept this agreement?* na položku *Yes* (viz obr. 5.2).



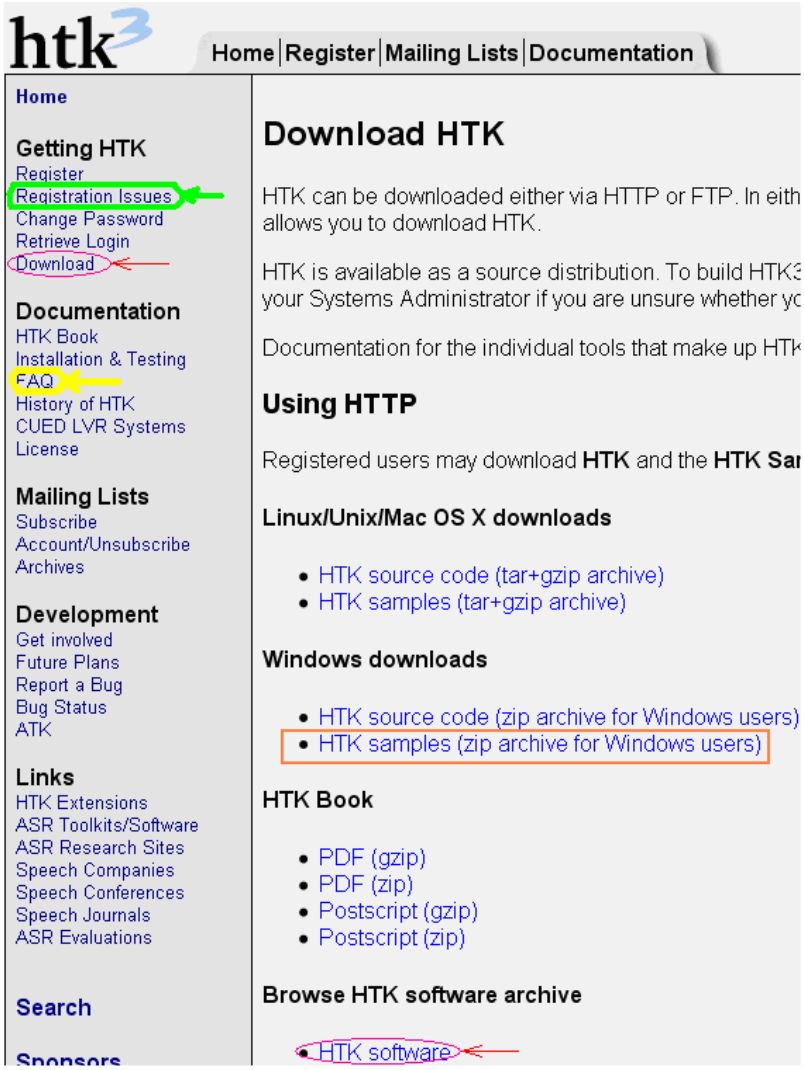
Obr. 5.2: Zobrazení potvrzovacího dialogu

Poté je zapotřebí chvíli vyčkat, než přijde na uvedený e-mail heslo (heslo většinou obsahuje 8 znaků) a přijatá zpráva vypadá ve tvaru uvedeném na obr. 5.3.



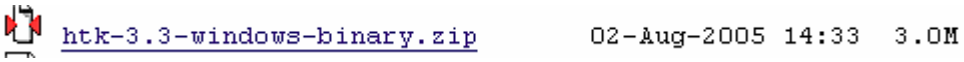
Obr. 5.3: Zobrazení příchozího e-mailu

Poté se do prohlížeče napíše http://htk.eng.cam.ac.uk/download.shtml a nebo v levé liště v sekci *Getting HTK* 5. položka odshora (poslední v této sekci) je zapotřebí najít *Download* (na obrázku 5.4 označeno červenou barvou) a kliknout na tuto položku. Uživatel se dostane na stránku, kde nahoře je napsáno *Download HTK* a je možno stáhnout program pomocí volby *Using http*, pod tímto nápisem je volba, jaký operační systém zvolit. Je zapotřebí stáhnout software pro operační systém Windows a tudíž se využije volby, jenž je napsána pod stáhnutím pro Linux,… a to *Windows downloads* a zde by se mělo kliknout na modře označený odkaz HTK source code (zip archive for Windows users).



Obr. 5.4: Zobrazení stránky pro stahování produktu

Po kliknutí je nutné zadat jméno a heslo (jenž přišlo na e-mail). Poté bude uložen zabalený program na plochu a je rozbalí se pomocí např. programů WinRar, WinZip atd. Pokud by uživatel chtěl stáhnout i vzorky HTK, klikne na modře označenou položku umístěnou hned pod odkazem na HTK zdrojový kód (je v tomto tvaru: HTK samples (zip archive for Windows users)). Tato položka je na obrázku 5.4 znázorněna oranžovým obdélníčkem. Po stažení na plochu je nutno znovu rozbalit soubor. Je možno stáhnout již zkompilovanou verzi, nad nápisem *Using FTP* v dolní části stránky v části *Browse HTK software archive* je modře označený odkaz na HTK software (v obrázku 5.4 označeno červenou barvou úplně dole), po jeho poklepnutí je možno stáhnout všechny verze HTK. Pro stažení zkompilované verze je zapotřebí najít htk-3.3-windows-binary.zip (skoro dole, velikost 3.0 MB, datum: 02-Aug-2005 14:33).

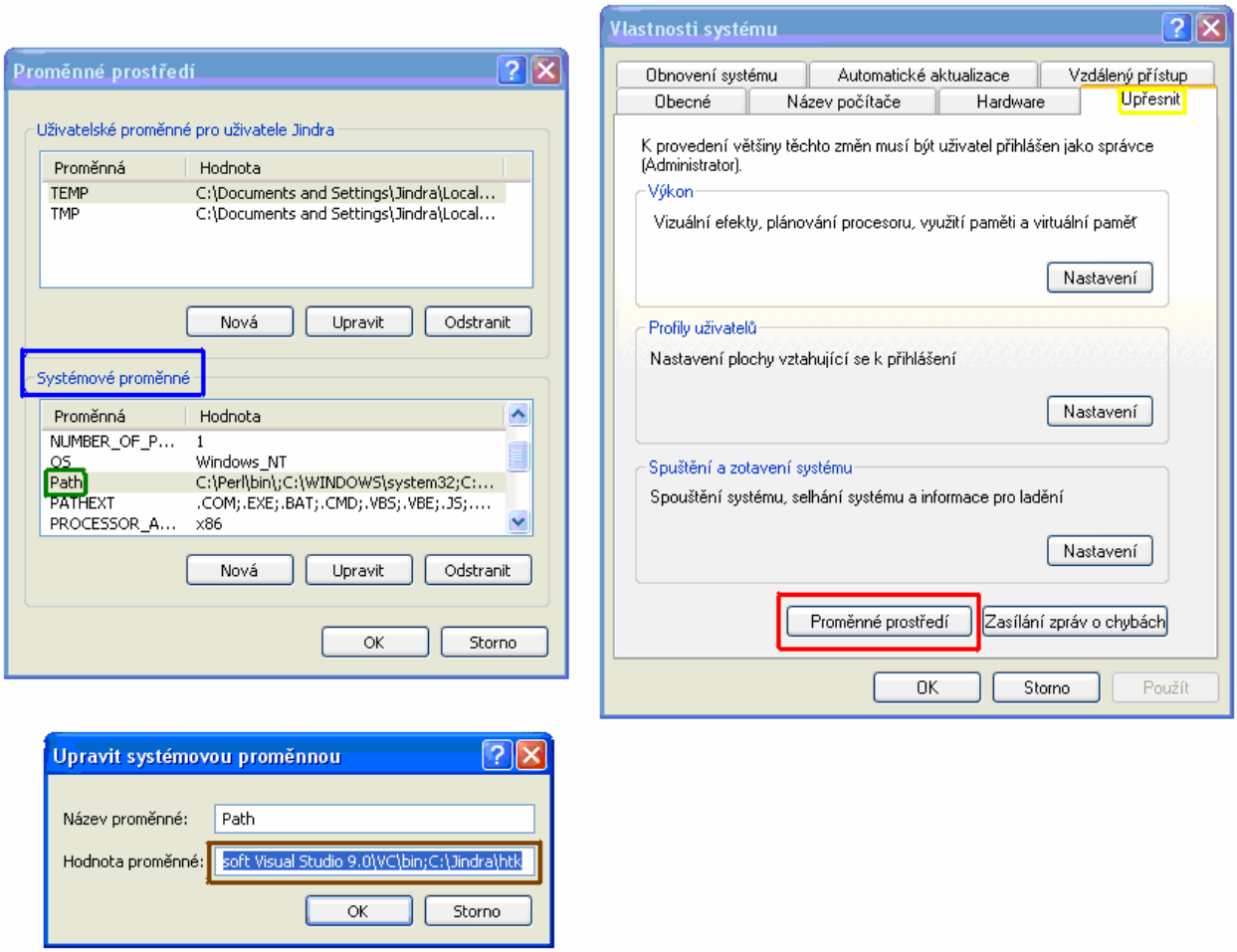


Pokud se nepodaří správně zaregistrovat, je možno na stránkách HTK kliknout na položku Registration issues (2. položka odshora v nabídce *Getting HTK* na obr. 5.4, vyznačeno na obrázku zeleně), kde jsou užitečné rady pro řešení problémů při registraci. Pokud se objeví při stahovaní jiné problémy, je možno kliknout na levé liště na hlavní stránce v paletě *Documentation* na *FAQ* (na obrázku 5.4 vyznačeno žlutě), kde je možno si přečíst o řešení dalších problémů a navíc je zde uveden e-mail pro vyřízení problémů (htk-users@eng.cam.ac.uk).

Pokud by uživatel potřeboval stáhnout rozšíření, jež může být důležité pro některé složitější příklady, musí stáhnout i utilitu *HDecode*. Je k nalezení na stránkách http://htk.eng.cam.ac.uk/extensions/index.shtml. Po kliknutí na *download HDecode from here*. Po napsání uživatelského jména a hesla (stejné jako v minulých případech) se dostane na další stranu, kde zvolí stažení verze pro Windows. Tato verze souboru je opět s příponou zip, je tedy opět nutné ji rozbalit pomocí WinRar, WinZip nebo jiným archivačním nebo rozbalovacím nástrojem.

Po stažení programu na plochu a následném rozbalení do složky např. *htk-3.3-windows-binary* (kterou nám rozbalovací program nabízí) je možno do této složky kliknout. V ní musí být složka htk, klikne se do ní a zde jsou všechny zkompilované programy (všechny s příponou exe + dávkovací soubor install.bat).

Nyní je zapotřebí spustit program *Příkazový řádek*. Je možno jej spustit klávesovou zkratkou Win(napravo od Ctrl a nalevo od Alt)+r. A poté napsat *cmd.exe*. Je možné jej najít například v menu *Start*, poté *Programy*, poté *Příslušenství* a nakonec *Příkazový řádek*. Je možno si nastavit příslušnou složku, což se provede příkazem *cd* mezera a název složky. Je vhodné si také nastavit cestu pomocí příkazu *PATH* v příkazové řádce, aby nebylo potřeba provádět při každém spuštění příkazové řádky postupné zadávání složek, ve kterých jsou umístěny všechny nástroje programu HTK. Nastavení složky je realizovatelné i bez příkazové řádky. Stačí kliknout pravým tlačítkem na ikonu *Tento počítač* umístěnou většinou vlevo nahoře (anglicky *My computer*). Poté v položce okna *Vlastnosti systému* a v záložce *Upřesnit* (označeno na obr. 5.5 žlutou barvou) dole se klikne levým tlačítkem na tlačítko *Proměnné prostředí* (označeno na obr. 5.5. červenou barvou). Otevře se nové okno a v něm v *Systémových proměnných* (na obr. 5.5. vlevo uprostřed zvýrazněno modrou barvou) je možné najít řádek začínající proměnnou *Path* (označeno na obr. 5.5 vlevo zelenou barvou), nutno kliknout na *upravit*, otevře se okno *Upravit systémovou proměnnou* a do pole *Hodnota proměnné* (označeno na obr. 5.5 hnědou barvou) za všechny ostatní se napíše středník a poté se vepíše cesta k spouštěcím souborům (např. *c:/dokumenty/htk*).



Obr. 5.5: Zobrazení oken pro nastavení cesty bez použití příkazů v příkazové řádce

V příkazové řádce nefunguje vložení pomocí stisku kláves CTRL+v, proto je nutné kliknout do příkazové řádky pravým tlačítkem myši a stisknout vložit. Stejným systémem (pomocí kliknutím na pravé tlačítko myši) je možno kopírovat obsah příkazové řádky nebo najít určitá slova). Stiskem posuvných kláves(nahoru či dolů) je možno přepínat mezi příkazy použitými v předchozích případech (čili listovat v historii příkazů). Je zapotřebí otevřít okno s příkazovou řádkou (zde se budou spouštět HTK programy) a Windows Commander: pomocí F3 je možnost prohlížet obsah textových souborů, pomocí F4 modifikovat. Po spuštění libovolného programu bez parametrů se objeví nápověda. V příkazové řádce je možno psát jak klasické lomítko (/), tak obrácené lomítko (\). Na funkci programu tato syntaxe nemá vliv, ovšem měla by se u jednotlivých příkazů dodržovat velké a malé písmena tak jak jsou uvedena níže.

6 NÁVOD NA ZPRACOVÁNÍ SIGNÁLŮ S VYUŽITÍM HTK

6.1 Obecné základní vlastnosti příkazů v HTK

Nástroj HTK je navržen k běhu pomocí příkazové řádky. Každý nástroj má určitý počet požadovaných argumentů plus volitelných argumentů. Například 1 řádek může vypadat takto (HFoo je neexistující HTK nástroj, zde jen pro příklad):

```
HFoo -T 1 -f 34.3 -a -s muj_soubor soubor1 soubor2
```

Jsou zde 2 hlavní argumenty zvané *soubor1* a *soubor2* a volitelné argumenty. Volitelné argumenty jsou uvozeny jednoduchým volitelným jménem následovaným příslušnou volitelnou hodnotou. Volba hodnoty je vždy oddělena od volby jména mezerou. Hodnota *–f* volby je reálné číslo, hodnota *-T* volby je celé číslo a hodnota *–s* volby je string hodnota. Volba *-a* nemá žádnou následující hodnotu a ta je užívaná k povolení nebo zakázání nějaké vlastnosti nástroje. Volby, jejichž jména jsou velká písmena, mají stejný význam napříč všemi nástroji. Například *-T* je volba, která je vždy užívána pro ovládání výstupu trasování HTK nástroje. Navíc k argumentům příkazové řádky, operace nástroje mohou být řízený parametry uloženými v konfiguračním souboru. Například, jestliže příkaz

```
HFoo -C config -f 34.3 -a -s muj_soubor soubor1 soubor2
```

je vykonán, nástroj HFoo (fiktivní) načte parametry uložené v konfiguračním souboru do zahajovací procesní procedury.

6.2 Návod na zpracování signálů v HTK s schematickou ilustrací

Součástí tohoto návodu je příklad, který bude rozdělen na tři části: přípravu dat, vytváření monofonních signálů a rozpoznávání signálů spolu s zhodnocením výsledků. Celý proces je velmi podrobně popsán tak, aby poskytl jasný pohled na řadu funkcí HTK. Obrázky z příkazové řádky budou uvedeny až od kapitoly 6.3 dále u konkrétních příkladů.

6.2.1 Příprava dat

První fáze každého projektu rozpoznávání je příprava dat. Řečové údaje jsou potřebné jak pro trénování, tak pro testování. Je vždy třeba trénovaných dat s dobrou fonetickou rovnováhou a pokrytím. Než mohou být údaje zaznamenány, musí být postaven slovník tak, aby zahrnoval trénování a testování a musí být definována gramatika.

Gramatika
Příkladem typických vstupů mohou být například tato slova:
Honza Petr Filip Marek…….

```
( <Honza | Filip | Petr | Marek> )
```

Svislé čáry označují alternativy, hranaté závorky označují volitelné položky a klasické závorky označují jedno nebo více opakování. Kompletní gramatika může být popsána jako síť. HTK rozpoznávač požaduje slovo, které bude definováno užitím nízké úrovně formátu zvaného HTK SLF, v němž každé slovo a každá instance přechodu slovo-slovo je výslovně uvedena. Toto slovo může být vytvořeno automaticky z gramatiky použitím nástroje HPARSE, tedy za předpokladu, že soubor např. *gramatika* obsahuje příslušnou gramatiku, poté se provede:

```
HParse gramatika sit_slov
```

kde *gramatika* je soubor s gramatikou a *sit_slov* vytvořený soubor se sítí.

Slovník

Prvním krokem při vytváření slovníku je vytvoření seřazeného seznamu požadovaných slov. Pokud bude slovník obsahovat málo slov, je docela snadné vytvořit seznam požadovaných slov. Nicméně, pokud úkol bude složitější, bylo by nutné vytvořit seznam slov z větných vzorků přítomných v testovacích datech. Dále pro tvorbu robustních akustických modelů je třeba trénovat je na velký soubor vět obsahující mnoho slov, a pokud možno foneticky vyvážené. Jak bude uvedeno později, bude poskytnut krátký příklad vytvoření seznamu slov z věty. Příklad prvních pár položek může vypadat např. takto:

```
zvuky/S1 FILIP POTKAL JI V PARKU
zvuky/S2 PETR JEDL 2 HODINY ….…
```

Požadovaný seznam trénovaných slov (*seznam_slov*) by pak mohl být automaticky získán z těchto slov. Před použitím HTK by bylo třeba upravit text do vhodného formátu. Například by bylo nutné změnit všechny mezery na konci řádků a pak pomocí příkazů *sort* a *uniq* řadit slova do abecedně uspořádaného souboru, s jedním slovem na řádek. Skript *Prompts2wlist* z adresáře HTKTutorial může být použit přesně pro tento účel.

Slovník sám může být postaven ze standardního zdroje pomocí nástroje HDMAN. Obecný formát slovníku každého vstupu je:

```
Petr [VYSTUPNI_SYMBOL] Petr .
```

Tzn., že slovo PETR je vyslovováno jako např. Petr ...atd. Řetězec v hranatých závorkách určuje řetězec na výstup v případě, že je rozpoznáno slovo. Je-li vynecháno, pak slovo samo o sobě je výstup. Pokud je zahrnuta hranatá závorka, ale prázdná, pak na výstupu není nic.

Vytváření záznamů

U trénování souborů HMM musí mít každý soubor trénovaných dat příslušnou úroveň záznamu. Použité sady v tomto obecném příkladu (ani v dalších následujících konkrétních příkladech v této diplomové práci nebude mít žádnou pauzu mezi jednotlivými slovy.

Startovním bodem pro všechny sady záznamu je záznam v HTK formátu. Ten může být vytvořeno využitím textového editoru nebo skriptovacího jazyku. Příkladem je skript prompts2mlf umístěný v HTK tutoriál adresáři. Výsledkem je, že přemění okamžité prohlášení do následujícího formuláře:

```
#!MLF!#
"*/zvuky/S1.lab"
FILIP
POTKAL
JI
V
PARKU
.
"*/zvuky/S2.lab"
PETR
JEDL
atd.
```

Jak je vidět, nadpisy musí být přeměněny na název cesty, každé slovo by mělo být napsáno na jednom řádku a každé prohlášení by mělo být ukončeno do jedné vlastní periody. První řádek souboru označuje soubor jako Master Label File (MLF- vysvětlení viz. [2]). Tento soubor je vždy jediný, který obsahuje kompletní sadu záznamů. HTK umožňuje každý jednotlivý záznam uložit do svého vlastního souboru, ale je mnohem efektivnější využit MLF (Master Label File – popis řečových dat).

Forma jmen cest používaných v MLF je vzor a nikoli jméno. HTK procesy řečových souborů, které lze najít označením souboru se stejným názvem, se liší příponou. Pokud tedy soubor *S1.wav* z složky zvuky byl zpracován, HTK by hledal návěští souboru nazvané *S1.lab*. Pokud jsou používány MLF soubory, HTK prohledá soubor pro model, který odpovídá požadovanému návěští názvu souboru. Je proto možné povolit stejné projevy, k užití s různými verzemi řečových dat, které mají být uloženy v různých místech.

Kódování dat

Závěrečná fáze přípravy dat je parametrizovaní syrových řečových průběhů do sekvencí charakteristických vektorů. HTK podporuje na FFT i LPC založené analýzy. Budou použity Melovské frekvenčně kepstrální koeficienty (MFCCs), které jsou odvozeny z FFT logaritmického spektra.

Kódování (jehož architektura je blokově zobrazena na obr. 6.1) lze provést pomocí nástroje HCopy nakonfigurovaného tak, aby automaticky převedl vstup do vektorů MFCC. Aby to bylo možno realizovat, je potřeba otevřít konfigurační soubor, který specifikuje všechny potřebné parametry pro konverzi. Tento soubor otevřeme např. pomocí programu Total Commander (označit soubor a stisknutí klávesy F3 zobrazí daný soubor). Nastavení kódovacích parametrů může vypadat např tato:

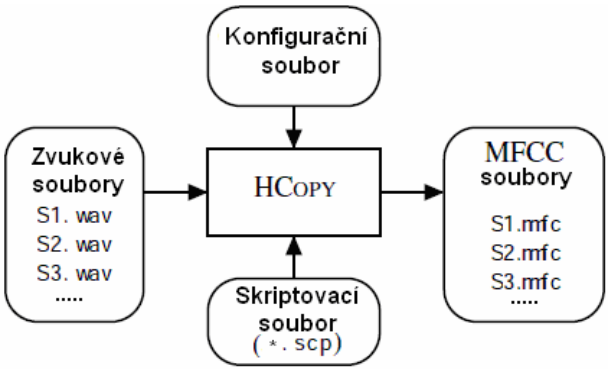
```
# Coding parameters
TARGETKIND = MFCC_0
TARGETRATE = 100000.0
WINDOWSIZE = 250000.0
USEHAMMING = T
PREEMCOEF = 0.97
NUMCHANS = 26
ENORMALISE = T
```

Některá z těchto nastavení jsou ve skutečnosti výchozím nastavenín. Specifikují, že cílové parametry budou MFCC používany jako komponenty energie, rámcová perioda je 10

milisekund (HTK používá jednotky 100ns) FFT by měl používat Hammingovo okno a signál by měl mít první řád preemfáze užitý s koeficientem 0,97. Banka filtrů by měla mít 26 kanálů. Proměnná ENORMALISE je standardně nastavená na true (pravda) a je nutná při nahrávání zvukových souborů.

Je-li třeba spustit HCopy, je potřebný seznam každého zdrojového souboru a odpovídající výstupní soubor. Například může prvních pár řádek vypadat v tomto tvaru:

```
/zvuky/S1.wav /zvuky/S1.mfc
/zvuky/S2.wav /zvuky/S2.mfc….atd.
```



Obr. 6.1: Architektura pro kódování dat

Soubory, které obsahují seznamy souborů, jsou označovány jako soubory skriptů a podle konvence jsou označovány příponou scp (i když HTK toto nevyžaduje). Soubory skriptů jsou specifikovány pomocí standardní *-S* volby a jejich obsahy jsou čteny jen jako rozšíření do příkazového řádku a to tak, aby se předešlo nutnosti použít příkazové řádky s několika tisíci argumenty.

Za předpokladu, že výše uvedený skript je uložen v souboru *trenovani.scp*, trénování dat bude kódovano spuštěním příkazu:

```
HCopy -T 1 -C konfig -S trenovani.scp
```

Podobný postup se používá pro kódování testovacích dat (je konfigurován užitím *TARGETKIND* = MFCC_0_D_A).

6.2.2 Vytváření monofonních skrytých Markových signálů

V této části bude popsáno vytvoření natrénovaných skupin jednotlivých Gaussovských monofonních skrytých Markových modelů (blokově zpracováno na obr. 6.2). Výchozím bodem bude skupina totožných monofonních skrytých Markových modelů, ve kterých je totožný každý průměr a rozptyl. Tyto modely budou poté ještě přetrénovány. Některé z položek slovníku mají více výslovností. Nicméně, když HLed slouží k rozšíření slovní úrovně MLF, aby vytvořil úrovně MLF, zvolí se první výslovnost, jež byla nalezena. Jakmile byly vytvořeny rozumné jednotlivé tóny skrytých Markových modelů, může být užit rozpoznávací nástroj HVITE k zarovnání tréninkových dat. Tímto způsobem je vytvořena nová úroveň MLF, ve které výběr výslovnosti závisí na akustické evidenci. Tento nový MLF muže být použit k vykonání konečného přehodnocení z monofonních skrytých Markových modelů.

Vytváření jednotlivých startovních tónů

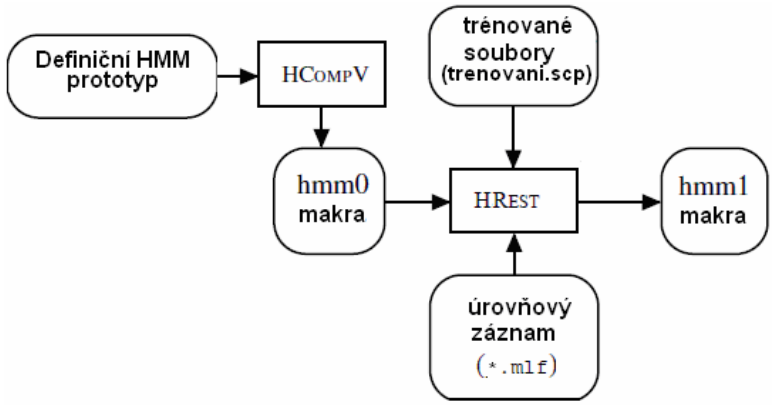
Prvním krokem v trénování skytých Markových modelů je definování prototypu modelu. Parametry tohoto modelu nejsou důležité, jeho cílem je definovat topologii modelu. Pro mnoho systémů je vhodná 3-stavová levo-pravá topologie bez jakýkoliv skoků, kde každý vektor elipsy má délku 39. Toto číslo (39) je vypočítáno z součtu délky parametrizovaného statického vektoru (MFCC_0 = 13), delta koeficientů (+13) a koeficientů zrychlení (+13).

Nástroj HTK HCompV bude skenovat sadu datových souborů, počítat globální průměr a rozptyl a nastaví všechny Gaussovy vektory v daném HMM, aby měly stejný průměr a rozptyl. Z tohoto důvodu, za předpokladu, že seznam všech trénovaných souborů je uložen v *trenovani.scp*, příkaz:

```
HCompV –C config -f 0.01 -m -S trenovani.scp -M hmmA prototyp
```

vytvoří novou verzi *prototyp* v adresáři *hmmA*, v kterém je nulový průměr a jednotkové odchylky výše byla nahrazeny globálními řečovými průměry a odchylkami. Prototyp HMM definuje parametr typu MFCC_0_D_A. To znamená, že delta koeficienty a koeficienty zrychlení jsou vypočteny a přidány do statických MFCC koeficientů vypočtených a uložených během kódování výše popsaného procesu. Aby se zajistilo, že tyto jsou počítány při načítání, konfigurační soubor může být upraven tak, aby změnit cílový druh, tedy by měl být změněn konfigurační soubor pro vstup TARGETKIND, aby *TARGETKIND= MFCC_0_D_A.*

Volba *-f* způsobí rozptyl makra, které má být generováno. Makro se rovná 0,01 násobku celkové variance. Je to vektor hodnot, které jsou často použity k nastavení odhadů rozptylu v následujících krocích. Volba *-m* žádá o průměr.



Obr. 6.2: Vytváření jednotlivých tónů

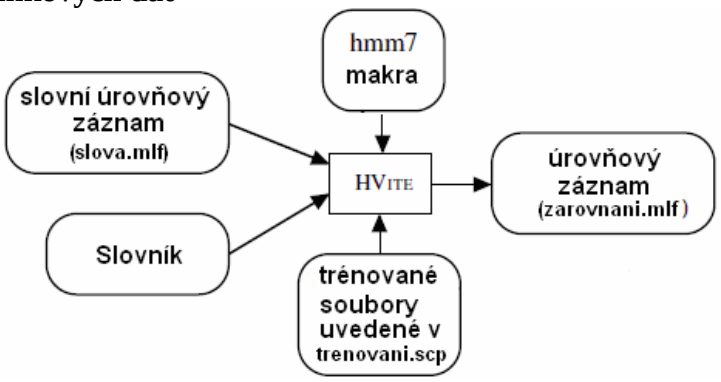
Startovací monofony uložené v adresáři *hmmA* jsou odhadovány s využitím vestavěných nástrojů pro odhad HREST takto:

```
HRest –C konfig -I *.mlf -t 250.0 150.0 1000.0 trenovani.scp –S -H
hmmA/makra -H hmmA/hmmdefinice -M hmmB monofony0
```

Důsledkem této situace jsou načteny všechny modely v *hmmA*, které jsou uvedeny v seznamu modelů např. *monofony0*. Ty jsou pak přehodnoceny pomocí údajů uvedených v *trenovani.scp* a nový model je pak uložen v adresáři *hmmB*. Většina souborů použitých k vyvolání HRest byly již popsány. Výjimkou je soubor maker. To obsahuje takzvané globální nastavení makra a rozptyl makra vytvořeného dříve. Globální nastavení makra definuje velikosti vektoru, tj. *~ o <MFCC_0_D_A> <VecSize> 39*

Pokaždé, když je spuštěn HREST, provede se jedno znovu zhodnocení. Každá nová HMM sada je uložena do nového adresáře. Vykonání HREST se opakuje ještě dvakrát, změna jména vstupního a výstupního adresáře (nastaveno s volbami *-H* a *-M*) pokaždé, dokud adresář *hmm1* obsahuje konečnou sadu inicializovaných jednotlivých tónů skrytých Markových modelů.

Zarovnávání tréninkových dat



Obr. 6.3: Zobrazení znázorňující zarovnání tréninkových dat

Slovník obsahuje více výslovností některých slov. Vytvořené modely mohou být použity k zarovnání trénovaných dat a vytváření nových záznamů. To může být provedeno jednou realizací HTK pomocí rozpoznávacího nástroje HVite, viz

```
HVite -l '*'-o SWT konfig –C -H hmm7/makra -H -i zarovnani.mlf –m -t-
250.0 I slova.mlf -S trenovani.scp slovnik monofony1
```

Tento příkaz užívá HMM uložený v *hmm7* k transformaci vstupní úrovně slov záznamu *slova.mlf* na novou úroveň záznamu *zarovnani.mlf* pomocí výslovnosti uložené v slovníku *slovnik* (viz obr. 6.3). Rozpoznávač zvažuje všechny výslovnosti pro každé slovo a na výstupy jdou výslovnosti, které nejlépe odpovídají akustickým datům.

Slovník by měl třídit nejdříve velká písmena a pak abecední sled písmen.Volba *-t* nastavuje úroveň ořízntutí na 250 a volba *-o* je použita k potlačení tisku výsledku, slovních názvů a časové hranice v MLF výstupu. Oříznutí omezuje rozsah zarovnání stavů, které zpětně dopředný algoritmus obsahuje ve svém souhrnu a může snížit množství výpočtů vyžadujících řád zesílení. Pro většinu trénovacích souborů může být stanoven velmi těsný limit ořízntutí, ale některé trénované soubory budou poskytovat chudším akustické přizpůsobení a tudíž je nutný širší limit pro ořízntutí. HREST umožňuje automatické zvyšování prahů pro ořízntutí. Např., hodnota ořízntutí je obvykle několik set. Pokud opětovný odhad selže na jakémkoli příslušném souboru, prahová hodnota se zvýší o nějakou hodnotu a soubor je přepracován. To se opakuje, dokud je jeden soubor úspěšně zpracován nebo je překročen limit ořízntutí (1000).

Jakmile je nové zarovnání vytvořeno, lze použít další 2 průchody HREST, aby jsme opětovně odhadli HMM sadu parametrů znovu. Za předpokladu, že toto je uděláno, konečné jednotlivé tóny sady HMM budou uloženy v adresáři (zde např.hmm9).

6.2.3 Rozpoznávání a hodnocení

Rozpoznávání je kompletní a vše, co je třeba, je spustit rozpoznávání a následně vyhodnotit výsledky pomocí HRESULTS nástroje pro analýzu.

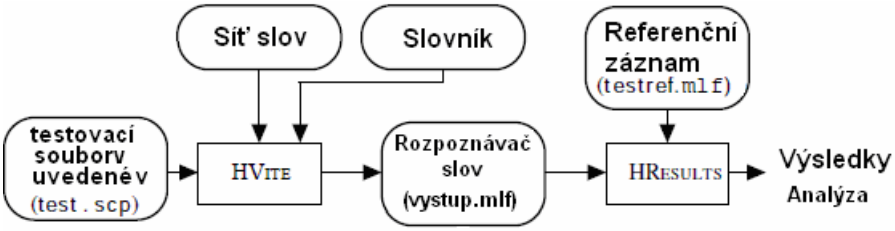
Za předpokladu, že v v souboru např. *testovani.scp* je seznam kódovaných testovacích souborů, pak každý testovací soubor bude rozpoznán a jeho transkripční výstup do MLF se zde bude jmenovat třeba *vystup.mlf* provedením následujících příkazů:

```
HVite -H hmm15/makra -H hmm15/hmmdefinice -S testovani.scp -l '*'-i
vystup.mlf -w sit -p 0.0 -s 5.0 slovnik modely
```

Volby *-p* a *-s* nastaví měřítko gramatiky. Měřítko gramatiky je velikost, o kterou je jazykový model pravděpodobnosti zmenšen před přidáním do jednotlivých znaků, jak je přenášen od konce jednoho slova k začátku příštího slova. Tento parametr může mít významný vliv na výsledky rozpoznávání.

Slovník obsahuje záznamy jednoduchých fonémů, zatímco seznam HMM obsahuje slovo složitějších fonémů. HVite bude vytvářet nutné konverze při načítání sítě slov *sit_slov* . Za předpokladu, že *testref.mlf* obsahuje slovní úroveň záznamů pro každou zkoušku souboru, může být skutečný výkon určující běh HResults dán tímto příkazem:

```
HResults -I testref.mlf modely vystup.mlf
```



Obr. 6.4: Zobrazení výsledné architektury programu

Výsledkem by byl formulář:

```
====================== HTK Results Analysis ==============
  Date: Sun Dec 19 16:14:45 2009
  Ref : testrefs.mlf      Rec : recout.mlf
 ---------------------- Overall Results ----------------
 SENT: %Correct=xx.xx [H=y, S=x, N=x]
 WORD: %Corr=xx.xx, Acc=xx.xx [H=x, D=x, S=x, I=x, N=x]
 ========================================================
```

Řádek začínající s *SENT:* ukazuje, že z N testu promluvy (číslo uvedené v závorce jako poslední), bylo správně rozpoznáno H, S je počet špatných rozpoznaných slov a číslo před závorkou je procentuálně počet správně rozpoznaných vzorků. Následující řádek začínající slovem *WORD:* ukazuje statistiky slovní úrovně a ukazuje, že z celkem x počtu slov (N=x), y bylo rozpoznáno správně (uvedeno H=y). Došlo k x smazaných chybám (D), x chybám substituce (S) a nějaké chybě vložení (I). Přesnost (ACC) je nižší než procento správné (Corr), protože bere v úvahu vložení chyb, které později ignoruje.

Tato kapitola popisovala rozpoznávání souvislé řeči s tím, že se dotkla oblastí, kterými se zabývá HTK: příprava dat, nástroje trénování a hodnocení.

6.3 Příklad na rozpoznávání slov ANO a NE

6.3.1 Zadání

Cílem je vytvořit rozpoznávač pro slova ANO a NE nezávislé na mluvčím. K natrénování použity data od 83 mluvčích (každý řekl ano i ne). Všechny zvukové soubory s příponou *wav*. K testování použito 9 vzorků dat, kdy se v jednotlivých souborech o délce trvání několika slov střídají slova *ano* a *ne* nebo vzorky složeny jen z slov *ano* nebo jen z slov *ne*.

6.3.2 Nastavení konfiguračního souboru

Můžeme otevřít konfigurační soubor `konfig/hcopy_wav.txt` a v něm uvidíme tyto parametry:

SOURCEKIND	= WAVEFORM	
SOURCEFORMAT	= WAV	řečové soubory jsou s příponou *wav*
ZMEANSOURCE	= FALSE	nebudeme ustředňovat signály
TARGETKIND	= MFCC_E_D_A	typ výstupních parametrů: MFCC a low-energie, delty a double-delty.
TARGETFORMAT	= HTK	
TARGETRATE	= 100000	vzorkovací perioda výstupních vektorů 10 ms
WINDOWSIZE	= 250000.0	délka okna 25 ms
NUMCHANS	= 24	počet trojúhelníkových filtrů pro výpočet MFCC
ENORMALISE	= TRUE	normování energie.

6.3.3 Obsahy jednotlivých adresářů:

data	řečová data v obvyklém formátu
hmma	modely inicializované pomocí HCompV.
hmmb	modely přetrénované pomocí HRest.
konfig	konfigurační soubory pro HTK programy.
popisrd	popisy řečových dat v Master-Label souborech.
prototyp	prototypy modelů.
seznam	seznamy modelů.
skripty	seznamy souborů pro HTK.
sit	slovní sítě pro rozpoznávání.
slovnik	slovníky.

6.3.4 Gramatika

Síť udává, jaké kombinace slov se mohou objevit na výstupu rozpoznávače. V tomto případě to může být pouze ANO nebo NE. Ručně vytvořená a lidsky čitelná síť je v *sit/stara_sit*.

Ta má tuto podobu:
(<ANO | NE>)

Tu je třeba překonvertovat do podoby, kterou umí přečíst už jen HTK:

```
HParse sit/stara_sit sit/nova_sit
```

Správné potvrzení příkazu je vyvoláno bezchybným posunutím o jeden řádek.

6.3.5 Parametrizace

Šířka okna pro výpočet jednoho rámce je 25 ms, posuv okna (frame shift) je 10 ms, dostane se tedy 100 vektorů za 1 sekundu.

V složce *skripty* je uložen soubor *trenovani.scp*, určený pro parametrizaci, v němž jsou obsaženy soubory, jenž vzniknou pro trénování z původních zvukových souborů v formátu *.wav:

```
data\Ano0.wav data\Ano0.mfc
data\Ano1.wav data\Ano1.mfc atd.
```

Parametrisace pro trénovací set se spouští příkazem (a její výsledek na obr. 6.5):

```
HCopy -T 1 -C konfig/copy_wav.txt -S skripty/trenovani.scp
```

```
C:\Jindra\VUT\MMSE\HTK FIT\pokus czech>HCopy -T 1 -C konfig/hcopy_wav.txt -S skr
ipty/trenovani.scp
data\Ano0.wav -> data\Ano0.mfc
data\Ano1.wav -> data\Ano1.mfc
data\Ano2.wav -> data\Ano2.mfc
data\Ne0.wav -> data\Ne0.mfc
data\Ne1.wav -> data\Ne1.mfc
data\Ne2.wav -> data\Ne2.mfc
```

Obr. 6.5: Znázornění parametrizace trénovacího setu slov ano a ne

V složce skripty je uložen i soubor *test.scp*, který když se otevře, uvidí se v něm vzorky vytvořené (na pravé straně v formátu *.mfc) z původních zvukových souborů a určené k testování:

```
data\Aaaa.wav data\Aaaa.mfc
data\Aaaaa.wav data\Aaaaa.mfc atd.
```

Parametrisace pro testovací set spouštěna příkazem (a provedení příkazu spolu s zobrazením příkazu je na obr. 6.6):

```
HCopy -T 1 -C konfig/hcopy_wav.txt -S skripty/test.scp
```

```
C:\Jindra\VUT\MMSE\HTK FIT\pokus czech>HCopy -T 1 -C konfig/hcopy_wav.txt -S skr
ipty/test.scp
data\Aaaa.wav -> data\Aaaa.mfc
data\Aaaaa.wav -> data\Aaaaa.mfc
data\Aan.wav -> data\Aan.mfc
data\Anana.wav -> data\Anana.mfc
data\Annan.wav -> data\Annan.mfc
data\Nanna.wav -> data\Nanna.mfc
data\Nna.wav -> data\Nna.mfc
data\Nnnnn.wav -> data\Nnnnn.mfc
data\Nnnnnn.wav -> data\Nnnnnn.mfc
```

Obr. 6.6: Znázornění parametrizace testovacích souborů

6.3.6 Trénování

Je vhodné si zobrazit Master-Label soubor *popisrd/trenovani.mlf*. Čísla před značkou udávají začátek a konec souboru ve stovkách ns. Délka zaznamenaná v MLF se vypočte jako: počet byte souboru * 0,0016 neboli počet byte souboru / 625.

Oba modely (neboli jejich prototypy umístěné ve složce *prototyp/ANO* a v *prototyp/NE*) mají složení uvedené v příloze A (7-stavové modely).

Pro model ANO se provede inicializace (dochází k výpočtu pravděpodobnostního rozdělení pomocí vzorců 2.5 a 2.6) těmito příkazy (správné provedení zobrazeno na obr. 6.7):

```
HCompV -T 7 -I popisrd/trénovani.mlf -l ANO -m -S
skripty/trenovani_htk.scp -M hmma prototyp/ANO
```

```
C:\Jindra\VUT\MMSE\HTK příklady\83 vzorku ano ne>HCompV -T 7 -I popisrd/trenovan
i.mlf -l ANO -m -S skripty/trenovani_htk.scp -M hmma prototyp/ANO
Calculating Fixed Variance
  HMM Prototype: prototyp/ANO
  Segment Label: ANO
  Num Streams  : 1
  UpdatingMeans: Yes
  Target Direct: hmma
 28 observations loaded from data/Ano0.mfc
                                        .....
 34 observations loaded from data/Ano82.mfc
4334 speech frames accumulated
Stream 1
 Mean Vector
      11.04    -3.04   -17.87     1.68     2.19   -12.08   -11.66    11.50    -3
.09   -14.49     2.11    -1.21 ...
 Variance Vector
      24.41    47.89    56.30    75.54    50.44    38.79    47.21    44.55    28
.19    32.75    27.41    22.71 ...
Updating HMM Means and Covariances
Output written to directory hmma
```

Obr. 6.7: Znázornění inicializace pro slovo ANO

Trénování modelu slova NE (výpočet pravděpodobnostního rozdělení pomocí vzorců 2.5 a 2.6) probíhá s malým rozdílem oproti trénování modelu slova ANO; dělá se tímto způsobem:

```
HCompV -T 7 -I popisrd/ trenovani.mlf -l NE -m -S
skripty/trenovani_htk.scp -M hmma prototyp/NE
```

6.3.7 Přetrénování modelů

Potřebujeme soubor *trenovani_htk.scp*, jenž obsahuje řádky:

```
data\Ano0.mfc
data\Ano1.mfc atd.
```

```
C:\Jindra\VUT\MMSE\HTK příklady\83 vzorku ano ne>HRest -T 7 -I popisrd/trenovani
.mlf -l ANO -S skripty/trenovani_htk.scp -M hmmb hmma/ANO
Reestimating HMM hmma/ANO . . .
 States   :   2  3  4  5  6 (width)
 Mixes  s1:   1  1  1  1  1 ( 39  )
 Num Using:   0  0  0  0  0
 Parm Kind:  MFCC_E_D_A
 Number of owners = 1
 SegLab   :  ANO
 MaxIter  :  20
 Epsilon  :  0.000100
 Updating :  Transitions Means Variances

 - system is PLAIN
  loading seg ANO 0.000000[0]->3028750.000000[27]
 28 observations loaded from data/Ano0.mfc
                                        .....
Ave LogProb at iter 20 = -3039.48739 using 83 examples  change =    0.00138
Estimation aborted at iteration 20
```

Obr. 6.8: Znázornění přetrénování modelu (slovo ANO)

Spušťení přetrénování pro oba modely se provádí tímto příkazem (příkaz zobrazen i na obr. 6.8 spolu s zobrazením realizace HRest):

```
HRest -T 7 -I popisrd/trenovani.mlf -l ANO -S skripty/trenovani_htk.scp
-M hmmb hmma /ANO
```

Stejným způsobem probíhá přetrénování slova NE, jen je zde v příkazu místo ANO napsáno NE, jak je možno si povšimnout níže:

```
HRest -T 7 -I popisrd/trénovani.mlf -l NE -S skripty/trenovani_htk.scp
-M hmmb hmma /NE
```

Díky přetrénování jsou v adresáři *hmmb* k disposici dva natrénované modely. Přetrénování je realizováno Baum-Welchovým algoritmem a provádí se výpočty dle vzorců 2.7 – 2.15.

6.3.8 Rozpoznávání

Vyprodukuje pro neznámé soubory přepis a uloží jej opět do *popisrd: popisrd/test_vystup.mlf*. Na rozpoznání je potřeba mít připravený slovník (zde v *slovnik/slovnik*), který obsahuje tento obsah:

```
ANO ANO
NE NE
```

A ještě je potřeba mít v složce skripty mít umístěný soubor *test_htk.scp*, do něhož když nahlédneme, uvidíme jen toto:

```
data\Aaaa.mfc
data\Aaaaa.mfc atd.
```

A v složce *seznam* by měl být soubor *modely*. V tomto souboru je soupis všech možných slov, ktré se mohou objevit na výstupu (zde jen ANO či NE každé na jiném řádku)

```
C:\Jindra\VUT\MMSE\HTK příklady\83 vzorku ano ne>Hvite -T 1 -d hmmb -S skripty/t
est_htk.scp -i popisrd/test_vystup.mlf -w sit/nova_sit slovnik/slovnik seznam/mo
dely
Read 2 physical / 2 logical HMMs
Read lattice with 5 nodes / 7 arcs
Created network with 9 nodes / 11 links
File: data\Aaaa.mfc
ANO ANO ANO  ==  [145 frames] -66.6040 [Ac=-9657.6 LM=0.0] (Act=6.9)
File: data\Aaaaa.mfc
ANO ANO ANO ANO ANO  ==  [238 frames] -61.6459 [Ac=-14671.7 LM=0.0] (Act=6.9)
File: data\Aan.mfc
ANO ANO ANO NE  ==  [131 frames] -68.9454 [Ac=-9031.8 LM=0.0] (Act=6.8)
File: data\Anana.mfc
ANO NE ANO ANO ANO  ==  [138 frames] -68.6907 [Ac=-9479.3 LM=0.0] (Act=6.9)
File: data\Annan.mfc
ANO NE NE ANO NE  ==  [178 frames] -66.4826 [Ac=-11833.9 LM=0.0] (Act=6.9)
File: data\Nanna.mfc
NE ANO NE NE ANO  ==  [205 frames] -60.4276 [Ac=-12387.7 LM=0.0] (Act=6.9)
File: data\Nna.mfc
NE NE ANO  ==  [125 frames] -62.5883 [Ac=-7823.5 LM=0.0] (Act=6.8)
File: data\Nnnnn.mfc
NE NE NE NE  ==  [165 frames] -62.9861 [Ac=-10392.7 LM=0.0] (Act=6.9)
File: data\Nnnnnn.mfc
NE NE NE NE ANO NE  ==  [231 frames] -64.6553 [Ac=-14935.4 LM=0.0] (Act=6.9)
File: data\naanna.mfc
NE ANO ANO NE NE ANO  ==  [251 frames] -64.7748 [Ac=-16258.5 LM=0.0] (Act=6.9)
```

Obr. 6.9: Znázornění rozpoznávání testovacích souborů s výskytem slov ano, ne

Rozpoznávání (jehož výsledek je na obr. 6.9 a jehož teoretický výpočet uvádí vzorec 2.16) se spustí pomocí:

```
HVite -T 1 -d hmmb -S skripty/test_htk.scp -i popisrd/test_vystup.mlf -
w sit/nova_sit slovnik/slovnik seznam/modely
```

Výsledek rozpoznávání si můžeme prohlédnout v *test_vystup.mlf* (ovšem není v uživatelsky přívětivé formě čtení).

6.3.9 Vyhodnocení

Je k dispozici popis řečových dat se správným přepisem testovacích souborů: *popisrd/test.mlf.*

Výpis testovacího souboru *test.mlf*:

```
#!MLF!#
"data/Aaaa.lab"
ANO
ANO
ANO
ANO
.
"data/Aaaaa.lab"
ANO
. atd.
```

Tento správný přepis je možno srovnat s výstupem HVite pomocí:

```
HResults -I popisrd/test.mlf seznam/modely popisrd/test_vystup.mlf
```

```
C:\Jindra\VUT\MMSE\HTK příklady\83 vzorku ano ne>HResults -I popisrd/test.mlf se
znam/modely popisrd/test_vystup.mlf
====================== HTK Results Analysis =======================
  Date: Thu May 13 11:56:01 2010
  Ref : popisrd/test.mlf
  Rec : popisrd/test_vystup.mlf
------------------------ Overall Results --------------------------
SENT: %Correct=50.00 [H=5, S=5, N=10]
WORD: %Corr=91.49, Acc=89.36 [H=43, D=2, S=2, I=1, N=47]
===================================================================
```

Obr. 6.10: Znázornění vyhodnocení testovaných souborů

Procentuální úspěšnost: $$Corr = \frac{N - D - S}{N} \cdot 100\,[\%] \tag{6.1}$$

Přesnost: $$Acc = \frac{N - D - S - I}{N} \cdot 100\,[\%] \tag{6.2}$$

kde N je celkový počet slov, počet správně rozpoznáných slov = H, S je počet špatně rozpoznaných. Smazané chyby = D, chyby substituce označeny jako S a chyby vložení se značí I. Přesnost (ACC) je nižší než procento správné (Corr), protože bere v úvahu vložení chyb, které později ignoruje.

Nejdůležitějším číslem na výstupu jsou %Corr, což udává procentuální úspěšnost správného rozpoznávání slov (zde 91,49 %). Přesnost Acc je 89,36 %, jak ukazuje obr. 6.10. U tohoto příkladu byly spojeny všechny testovací soubory a rozpoznávání proběhlo pro všechny soubory naráz.

6.4 Příklad na rozpoznávání 50 českých slov

6.4.1 Zadání

Cílem vytvořit rozpoznávač pro 50 slov. K natrénování použito 10 vzorků dat od 6 různých mluvčích. Zvukové soubory nahrány částečně v nemocnici na oddělení ORL. Všechny zvukové soubory budou s příponou wav.

6.4.2 Gramatika

Síť v tomto případě obsahuje 50 slov. Ručně vytvořená a lidsky čitelná síť je v *sit/stara_sit*.

Ta má tuto podobu (znázorněn začátek a konec sítě kvůli používaným znakům):

```
(<BAS | BEH |……. | VZTEK>)
```

Tu je třeba překonvertovat do podoby, kterou umí přečíst už jen HTK:

```
HParse sit/stara_sit sit/nova_sit
```

Správné potvrzení příkazu je vyvoláno bezchybným posunutím o jeden řádek.

6.4.3 Parametrizace

V složce *skripty* uložen soubor *trenovani.scp*, v němž jsou napsány soubory, které jsou ve formátu wav a z nichž budou vytvořeny soubory ve formátu mfc, který je vhodný pro program HTK (zde uveden jen začátek, jelikož použito mnoho testovacích souborů):

```
data/1/kup.wav data/1/kup.mfc
data/1/skvost.wav data/1/skvost.mfc atd.
```

Parametrisace pro trénovací i testovací set se spouští příkazy:

```
HCopy -T 1 -C konfig/hcopy_wav.txt -S skripty/trenovani.scp
```

Konec parametrizace označuje posuv o další řádek po vytvoření posledního souboru v formátu mfc z souboru ve formátu wav (viz obr. 6.11).

```
data/0/pral.wav -> data/0/pral.mfc
data/0/vem.wav -> data/0/vem.mfc

C:\Jindra\VUT\MMSE\HTK příklady\Novy>
```

Obr. 6.11: Znázornění konce parametrizace slov určených pro trénování

V složce skripty je uložen i soubor *test.scp*, v němž po otevření budou viděny vzorky, které se vytvoří (na pravé straně soubory s příponou mfc) z původních souborů (na levé straně s příponou wav) a které budou určené k testování (opět uveden jen příklad prvních dvou řádků):

```
data/4/4.wav data/4/4.mfc
data/0/0.wav data/0/0.mfc atd.
```

Soubory (*4.mfc* , *0. mfc* a další) se vytvoří pomocí příkazu:

```
HCopy -T 1 -C konfig/copy_wav.txt -S skripty/test.scp
```

```
C:\Jindra\VUT\MMSE\HTK příklady\Novy>HCopy -T 1 -C konfig/hcopy_wav.txt -S skrip
ty/test.scp
data/4/4.wav -> data/4/4.mfc
                                                    ....
data/9/9.wav -> data/9/9.mfc

C:\Jindra\VUT\MMSE\HTK příklady\Novy>
```

Obr. 6.12: Znázornění parametrizace testovacího setu zvukových souborů

Čím větší soubor, tím delší proces parametrizace. Konec parametrizace bude opět znázorněn posuvem o jedne řádek, čili stejným způsobem jako na obr. 6.12.

6.4.4 Trénování

Master-Label soubor *trenovani.mlf* v složce *popisrd* zobrazuje všechny trénovaná slova v formátu lab (zde znázorněny první 2 slova):

```
#!MLF!#
"data/1/sraz.lab"
0       53750000        SRAZ
.
"data/2/sraz.lab"
0       49316250        SRAZ
.
```

Čísla před značkou udávají začátek a konec souboru ve stovkách ns. Délka zaznamenaná v MLF se vypočte jako: počet byte / 625.

Je zapotřebí mít připravený soubor *trenovani_htk.scp*, jenž obsahuje všechny slova, jenž se budou trénovat už ve formátu, který vyhovuje HTK:

```
data/5/sraz.mfc
data/1/sraz.mfc  ……..
```

Všechny modely (neboli jejich prototypy umístěné ve složce *prototyp*), jenž jsou sedmistavové, mají složení (můžeme zobrazit příkazem F3 z Total Commander model slova např. HRAD), které je uvedeno v příloze A.

Pro jednotlivé modely se provede trénování těmito příkazy (dochází k výpočtu pravděpodobnostního rozdělení pomocí vzorců 2.5 a 2.6):

```
HCompV -T 7 -I popisrd/trénovani.mlf -l HRAD -m -S
skripty/trenovani_htk.scp -M hmma prototyp/HRAD
```

Pokud se bude trénovat jiné slovo, nahradíme toto slovo místa slova HRAD (vyskytuje se 2x v příkazu). Správné trénování slova HRAD znázorněno na obr. 6.13.

```
Calculating Fixed Variance
  HMM Prototype: prototyp/HRAD
  Segment Label: HRAD
  Num Streams  : 1
  UpdatingMeans: Yes
  Target Direct: hmma
 70 observations loaded from data/4/hrad.mfc
                                                    ....
```

```
 78 observations loaded from data/0/hrad.mfc
219 speech frames accumulated
Stream 1
 Mean Vector
      -8.00    -0.18    -0.57    -4.52     2.22    -1.62     5.72    -1.90     2
.12     3.69     2.13     2.99 ...
 Variance Vector
      46.32    38.47    28.07    48.83    48.10    35.25    37.49    76.38    57
.04    30.06    24.55    20.45 ...
Updating HMM Means and Covariances
Output written to directory hmma

C:\Jindra\VUT\MMSE\HTK příklady\Novy>_
```

Obr. 6.13: Znázornění správného trénování slova HRAD

6.4.5 Přetrénování modelů

Spušťení přetrénování pro model HRAD (neboli se vytvoří modely v složce *hmmb*) se děje tímto příkazem:

```
HRest -T 7 -I popisrd/trenovani.mlf -l HRAD -S
skripty/trenovani_htk.scp -M hmmb hmma/HRAD
```

```
Reestimating HMM hmma/HRAD . . .
 States   :   2  3  4  5  6 (width)
 Mixes  s1:   1  1  1  1  1 ( 39  )
 Num Using:   0  0  0  0  0
 Parm Kind:  MFCC_E_D_A
 Number of owners = 1
 SegLab   :  HRAD
 MaxIter  :  20
 Epsilon  :  0.000100
 Updating :  Transitions Means Variances

 - system is PLAIN
 0 observations loaded from data/5/sraz.mfc
                                        ....
 0 observations loaded from data/0/kvet.mfc
  loading seg HRAD 0.000000[0]->43896250.000000[77]
 78 observations loaded from data/0/hrad.mfc
                                        ....
3 Examples loaded, Max length = 78, Min length = 70
Ave LogProb at iter 1 = -4612.46908 using 3 examples
Ave LogProb at iter 2 = -4339.95605 using 3 examples  change =  272.51318
                                       .......
Estimation converged at iteration 20
```

Obr. 6.14: Znázornění přetrénování modelu HRAD

Přetrénování je realizováno Baum-Welchovým algoritmem a provádí se výpočty dle vzorců 2.7 – 2.15. Nejdůležitější řádky tohoto procesu jsou zpracovány na obr. 6.14, kde u všech jiných slov než je přetrénované by měl být počet výskytů *(observations)* roven nule. Kolikrát se slovo objeví v trénovacíh souborech, tolikrát by mělo být uvedeno v počtu načtených příkladů (x *Examples loaded*). Díky přetrénování jsou v adresáři *hmmb* všechny přetrénované modely.

6.4.6 Rozpoznávání

Rozpoznávání vyprodukuje pro neznámé soubory přepis a uloží jej do *popisrd/testvystup.mlf*. Je třeba mít připravený slovník (zde v složce *slovnik* je jen jediný soubor *slovnik*), který

obsahuje 50 řádků (stejný jako počet slov) a po stisknutí F3 (nebo F4) v Total Commnaderu (nebo jiném podobném souborovém manažeru) bude zobrazeno:

```
BAS BAS
BEH BEH atd.
```

A ještě v složce skripty musí být umístěný soubor *test_htk.scp*,do něhožpo nahlédnnutí, bude viděn seznam testovaných souborů:

```
data/4/4.mfc
```

Musí být připravený také seznam modelů v složce seznam a v něm jediný soubor modely, v němž je soupis 50 modelů:

```
BAS
BEH atd..
```

Rozpoznávání se spouští pomocí příkazu (pravděpodobnost generované promluvy – viz vzorec 2.16):

```
Hvite -T 1 -d hmmb -S skripty/test_htk.scp -i popisrd/test_vystup.mlf -
w sit/nova_sit -p -510.0 -s 0.0 slovnik/slovnik seznam/modely
```

```
Read 47 physical / 47 logical HMMs
Read lattice with 50 nodes / 142 arcs
Created network with 99 nodes / 191 links
File: data/4/4.mfc
VEM PRAL CAS BEZ KNIR KUP STESK TRI CHOV MOST KUP MEL DBAM SPAT PES TY2
EK VZTEK MOR KUP SKLO CHLUP RVAL BAS CTE LET SMIS VZTEK LOV SKVOST CHLU
IE RAM HRAD KVET MIC CTE STESK RYK SBOR CHLUP PEL VAL STESK JEZ CTE MIF
 CTE VOJ CTE SKVOST KUP  ==  [7347 frames] -57.1528 [Ac=-390831.7 LM=-2
Act=97.0)
```

Obr. 6.15: Znázornění rozpoznávání 1 souboru obsahující 50 slov

Hodnoty čísel -510.0 a 0.0 na obr. 6.15 a v příkazu znázorňují velikost ořezaní mezi jednotlivými slovy, jelikož promluva je pauzami mezi slovy, a proto by docházelo k detekci více modelů, než jich je ve skutečnosti. Proto je nutno tuto hodnotu pečlivě nastavit a podle rychlosti pauzy mezi slovy případně upravovat. Celý obsah rozpoznávání si můžeme také prohlédnout v souboru *test_vystup.mlf*.

6.4.7 Vyhodnocení

Je k dispozici popis řečových dat se správným přepisem testovacích souborů: *popisrd/test.mlf*.

Ten by měl obsahovat přesný soupis, jak by měl vypadat sled slov v testovacím souboru:

```
#!MLF!#
"data/4/4.lab"
VEM
…….
KUP
.
```

Tento správný přepis je možno srovnat s výstupem HVite pomocí:

```
HResults -I popisrd/test.mlf seznam/modely popisrd/test_vystup.mlf
```

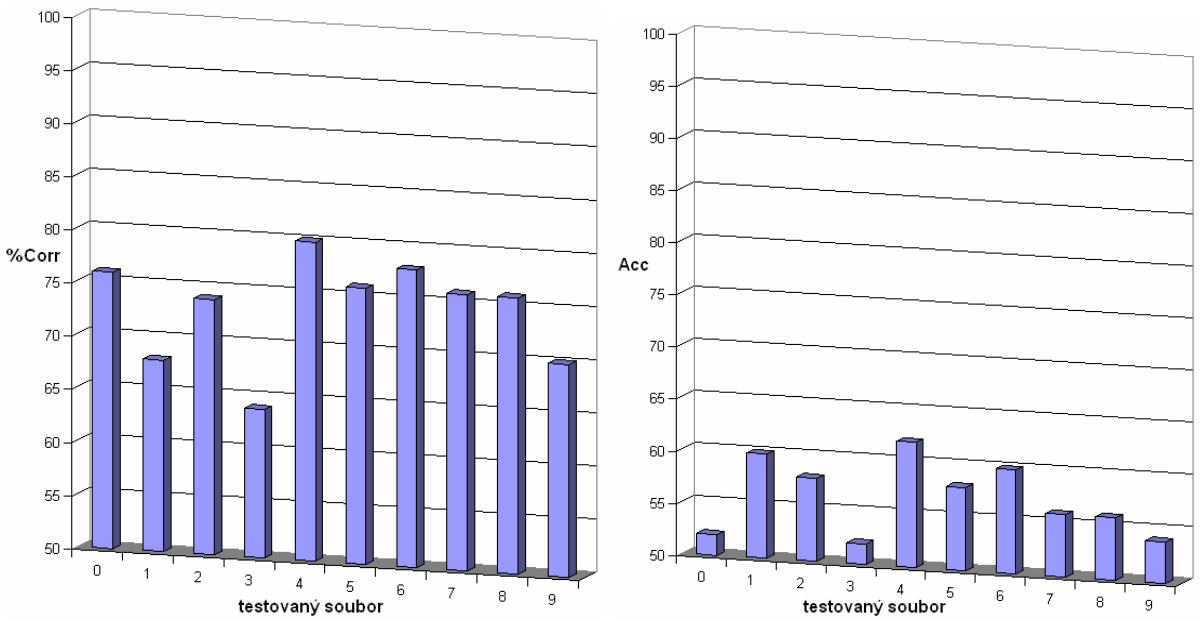
Na výstupu je vidět %Corr, která udává úspěšnost rozpoznání slov. Zde bylo dosaženo 80 % správného rozpoznání slov s přesností 62 % (Acc), což je vcelku malá procentuální úspěšnost rozpoznávání, jak ukazuje obr. 6.16.

```
C:\Jindra\VUT\MMSE\HTK příklady\Novy>HResults -I popisrd/test_new.mlf seznam/mod
ely popisrd/test_vystup.mlf
====================== HTK Results Analysis =======================
  Date: Thu May 13 10:56:47 2010
  Ref : popisrd/test_new.mlf
  Rec : popisrd/test_vystup.mlf
------------------------ Overall Results --------------------------
SENT: %Correct=0.00 [H=0, S=1, N=1]
WORD: %Corr=80.00, Acc=62.00 [H=40, D=3, S=7, I=9, N=50]
===================================================================
```

Obr. 6.16: Znázornění vyhodnocení rozpoznávání 50 slov

Pokud by nedocházelo k rozpoznávání po slovech, ale po jednotlivých hláskách, procentuální úspěšnost rozpoznávání by se měla zvýšit, ale složitost by se také zvýšila. Také je velmi problematické zavedení modelu pauzy, které stejně jako rozpoznávání po hláskách zde není obsaženo. V tomto případě (jak je vidět na obr. 6.17 hlavně vpravo v grafické závislosti přesnosti rozpoznávání ACC na typu testovaního souboru) je procentuální úspěšnost rozpoznávání velmi nízká. Je to dáno i faktem, že testované soubory měly velmi rozdílnou délku, přitom počet slov byl stále stejný.

Všechny příkazy použitelné pro všechny testovací soubory jsou uloženy v *přehled příkazů.txt*. v každém adresáři daného příkladu, kde jsou i příkazy pro různé varianty testovacích souborů.



Obr. 6.17: Znázornění procentuální úspěšnosti rozpoznávání 50 slov

6.5 Příklad na rozpoznávání číslovek 0-9

6.5.1 Zadání

Cílem je vytvořit rozpoznávač pro číslovky nula až devět. K dispozici je přibližně 30 trénovacích souborů a 17 testovacích. Soubory byly nahrány pomocí programu Záznam zvuku, jenž je volně dostupný v operačním systému Windows. Všechny použité zvukové soubory a jejich kvalita a parametry uveřejněny v tabulce 7.1. Rozpoznávání se provádí jako rozpoznávání slov (ne hlásek).

6.5.2 Gramatika

Ručně vytvořená a lidsky čitelná síť je v *sit/stara_sit*.

Obsahem tohoto souboru je toto:

```
(<NULA | JEDNA | DVA | TRI | CTYRI | PET | SEST | SEDM | OSM | DEVET>)
```

Tuto podobu je třeba překonvertovat do podoby, kterou umí přečíst už jen HTK pomocí příkazu, jež vytvoří soubor *nova_sit*:

```
HParse sit/stara_sit sit/nova_sit
```

6.5.3 Parametrizace

Obsahem hcopy_wav.txt jsou stejné parametry jako u předchozích příkladů.

Složka *skripty* obsahuje také soubor *trenovani.scp*, určený pro parametrizaci, v němž jsou obsaženy soubory, jenž vzniknou pro trénování z původních zvukových souborů v formátu wav. V něm jsou všechny trénované soubory a jména souborů, na která se přejmenují a do jaké složky se uloží. Poté je možno vidět, že HTK si zvukové soubory převede soubory do formátu, s nímž dále pracuje a to příkazem

```
HCopy -T 1 -C konfig/hcopy_wav.txt -S skripty/trenovani.scp
```

V složce skripty je uložen i soubor *test.scp*, v něm jsou vzorky vytvořené (na pravé straně v formátu mfc s parametry uvedenými v `konfig/hcopy_wav.txt`) z původních zvukových souborů a určené k testování pomocí této řádky napsané v příkazovém řádku:

```
HCopy -T 1 -C konfig/hcopy_wav.txt -S skripty/test.scp
```

6.5.4 Trénování

V souboru *popisrd\trenovani.mlf* je možno si povšimnout čísel před značkou, jenž udávají začátek a konec souboru ve stovkách ns. Délka, která se zapíše do *trenovani.mlf* se vypočte jako: počet byte souboru se stejným jménem s příponou wav * 0,0016.

Nezbytný je soubor *trenovani_htk.scp* (uložen uvnitř složky *skripty*), jenž obsahuje soupis všech souborů s příponou mfc, které budou trénovány (uloženy ve složce *data).*

Pro všechny modely se provede inicializace (dochází k výpočtu pravděpodobnostního rozdělení pomocí vzorců 2.5 a 2.6) těmito příkazy (naznačeny jen dva):

```
HCompV -T 7 -I popisrd/trenovani.mlf -l NULA -m -S
skripty/trenovani_htk.scp -M hmma prototyp/NULA
```

U jiného slova je syntaxe stejná, jen slovo nula nahradíme jiným, např. zde uvedeno *DEVĚT:*

```
HCompV -T 7 -I popisrd/ trenovani.mlf -l DEVET -m -S
skripty/trenovani_htk.scp -M hmma prototyp/DEVET
```

Po tomto postupu vznikne v složce *hmma* 10 modelů (čili počet slov vyskytujících se v trénovacích souborech).

6.5.5 Přetrénování modelů

Přetrénování pro všechny modely pomocí HRest vypadá takto (opět ukázány jen dva modely):

```
HRest -T 7 -I popisrd/trenovani.mlf -l NULA -S
skripty/trenovani_htk.scp -M hmmb hmma /NULA
```

```
HRest -T 7 -I popisrd/trénovani.mlf -l JEDNA -S
skripty/trenovani_htk.scp -M hmmb hmma /JEDNA
```

Přetrénování je realizováno Baum-Welchovým algoritmem a provádí se výpočty dle vzorců 2.7 – 2.15. Díky přetrénování je v adresáři *hmmb* k dispozici 10 natrénovaných modelů (měl by jich být stejný počet jako ve složce *hmma*).

6.5.6 Rozpoznávání

Při rozpoznání se vyprodukuje pro neznámé soubory přepis a uloží si jej HTK do *popisrd/test_vystup.mlf*. Potřebujeme mít připravený slovník (zde uvnitř *slovnik/slovnik*), který obsahuje všechny slova, jež se mohou objevit na výstupu a jejich značení v HTK vypdá takto:

```
NULA NULA
……
DEVET DEVĚT
```

A ještě je třeba mít v složce skripty umístěný soubor *test_htk2.scp*, jenž obsahuje informaci o umístění souboru, jenž bude testován a jenž HTK změnilo z původního zvukového souboru (stejný název, jen přípona *mfc*)

```
data/02.mfc
```

Ještě je také nutno vytvořit soubor *modely* v složce *seznam*. Ten obsahuje soupis všech slov použitých při rozpoznávání oddělených koncem řádky:

```
NULA
……
DEVĚT
```

```
Read 10 physical / 10 logical HMMs
Read lattice with 13 nodes / 31 arcs
Created network with 25 nodes / 43 links
File: data/02.mfc
NULA JEDNA DVA TRI CTYRI PET SEST SEDM OSM DEVET  ==  [708 frames] -60.1658 [Ac=
-36987.4 LM=-5610.0] (Act=22.9)
```

Obr. 6.18: Znázornění rozpoznávání číslovek

Rozpoznávání (neboli pravděpodobnost generované promluvy – viz vzorec 2.16) se spouští pomocí příkazu (jak je možné vidět i na obr. 6.18):

```
Hvite -T 1 -d hmmb -S skripty/test_htk2.scp -i popisrd/test_vystup2.mlf
-w sit/nova_sit -p -299.0 -s 100.0 slovnik/slovnik seznam/modely
```

6.5.7 Vyhodnocení

Je k dispozici popis řečových dat se správným přepisem testovacích souborů: *popisrd/test2.mlf*, jehož obsah je vidět zde:

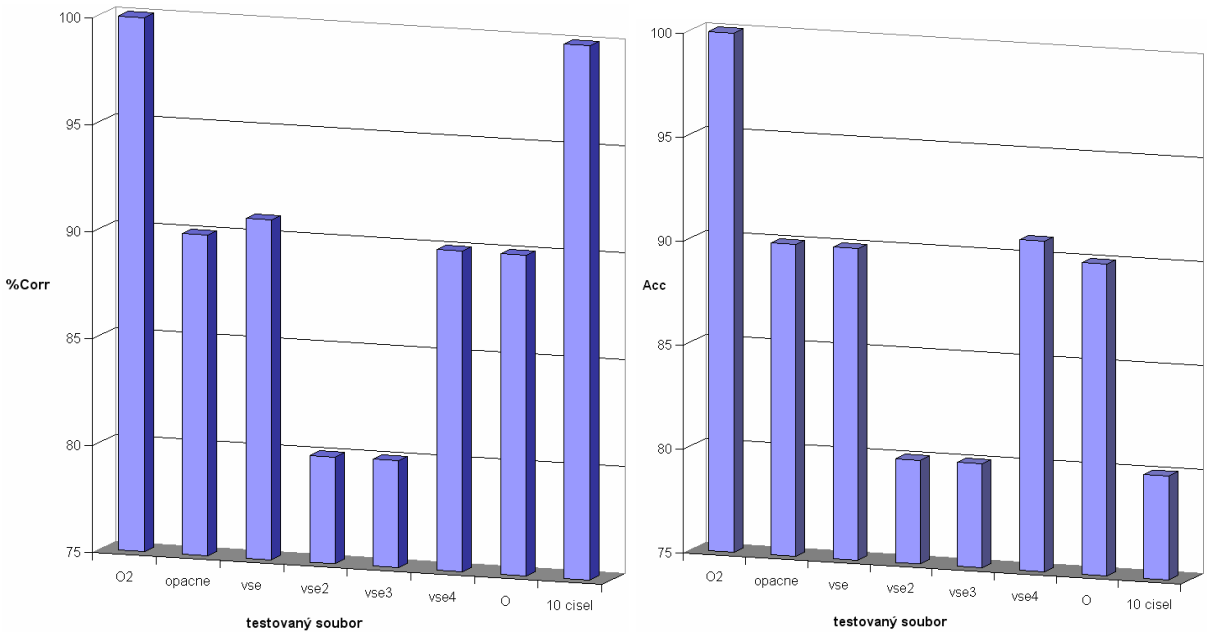
```
#!MLF!#
"data/02.lab"
NULA
….
DEVET
.
```

Tento správný přepis je možno srovnat s výstupem HVite pomocí:

```
HResults -I popisrd/test2.mlf seznam/modely popisrd/test_vystup2.mlf

====================== HTK Results Analysis =======================
  Date: Thu May 13 12:30:28 2010
  Ref : popisrd/test_new2.mlf
  Rec : popisrd/test_vystup2.mlf
------------------------ Overall Results --------------------------
SENT: %Correct=100.00 [H=1, S=0, N=1]
WORD: %Corr=100.00, Acc=100.00 [H=10, D=0, S=0, I=0, N=10]
===================================================================
```

Obr. 6.19: Znázornění vyhodnocení rozpoznávání číslovek



Obr. 6.20: Znázornění procentuální úspěšnosti rozpoznávání číslovek

Na levé části Obr. 6.20 je vidět grafická (sloupcová) závislost *%Corr* na typu testovaného souboru v procentech. Soubory vykazují stejnou nebo vyšší procentuální úspěšnost než je na pravé části obr. 6.20, kde je znázorněna procentuální závislost přesnosti *Acc* na testovaném souboru. Procentuální úspěšnost rozpoznávání je u všech testovaných zvukových souborů velmi vysoká.

7 DATABÁZE VHODNÝCH AKUSTICKÝCH SIGNÁLŮ

Všechny zvukové soubory v tabulce 7.1 používají formát zvuku PCM jsou s příponou wav, vzorkovací frekvence zvuku je 44 kHz a je použito 16 bitové vzorkování (staženo z [27], [28], [29] a [30]).

Tab. 7.1: Databáze vstupních signálů pro HTK

Název souboru	**Délka trvání [s]**	**Přenosová rychlost [kbps]**	**trénovaný či testovací soubor**	**Počet nahrávek**	**Vzorky použity v příkladu**	**Počet kanálů**
BAS	1	705	trénovaný	10	50 slov	1
BEH	1	705	trénovaný	10	50 slov	1
BEZ	1	705	trénovaný	10	50 slov	1
CAS	1	705	trénovaný	10	50 slov	1
CTE	1	705	trénovaný	10	50 slov	1
DBAM	1	705	trénovaný	10	50 slov	1
DUB	1	705	trénovaný	10	50 slov	1
HRAD	1	705	trénovaný	10	50 slov	1
CHCE	1	705	trénovaný	10	50 slov	1
CHLUP	1	705	trénovaný	10	50 slov	1
CHOV	1	705	trénovaný	10	50 slov	1
JEZ	1	705	trénovaný	10	50 slov	1
KNIR	1	705	trénovaný	10	50 slov	1
KOST	1	705	trénovaný	10	50 slov	1
KUP	1	705	trénovaný	10	50 slov	1
KVET	1	705	trénovaný	10	50 slov	1
LEM	1	705	trénovaný	10	50 slov	1
LET	1	705	trénovaný	10	50 slov	1
LOV	1	705	trénovaný	10	50 slov	1
MEL	1	705	trénovaný	10	50 slov	1
MIC	1	705	trénovaný	10	50 slov	1
MIR	1	705	trénovaný	10	50 slov	1
MOR	1	705	trénovaný	10	50 slov	1
MOST	1	705	trénovaný	10	50 slov	1
PEL	1	705	trénovaný	10	50 slov	1
PES	1	705	trénovaný	10	50 slov	1
PRAL	1	705	trénovaný	10	50 slov	1
RAM	1	705	trénovaný	10	50 slov	1
RVAL	1	705	trénovaný	10	50 slov	1
RYK	1	705	trénovaný	10	50 slov	1

Název souboru	Délka trvání [s]	Přenosová rychlost [kbps]	vzorkování [bity]	Počet nahrávek	Vzorky použity v příkladu	Počet kanálů
SBOR	1	705	trénovaný	10	50 slov	1
SKLO	1	705	trénovaný	10	50 slov	1
SKVOST	1	705	trénovaný	10	50 slov	1
SPAT	1	705	trénovaný	10	50 slov	1
SRAZ	1	705	trénovaný	10	50 slov	1
SRUB	1	705	trénovaný	10	50 slov	1
STESK	1	705	trénovaný	10	50 slov	1
STREL	1	705	trénovaný	10	50 slov	1
TRI	1	705	trénovaný	10	50 slov	1
TY	1	705	trénovaný	10	50 slov	1
TYZ	1	705	trénovaný	10	50 slov	1
VAL	1	705	trénovaný	10	50 slov	1
VEM	1	705	trénovaný	10	50 slov	1
VOJ	1	705	trénovaný	10	50 slov	1
VZDY	1	705	trénovaný	10	50 slov	1
VZTEK	1	705	trénovaný	10	50 slov	1
ZLO	1	705	trénovaný	10	50 slov	1
ZUB	1	705	trénovaný	10	50 slov	1
ZVYK	1	705	trénovaný	10	50 slov	1
0	67	705	testovací	1	50 slov	1
1	125	705	testovací	1	50 slov	1
2	93	705	testovací	1	50 slov	1
3	157	705	testovací	1	50 slov	1
4	73	705	testovací	1	50 slov	1
5	67	705	testovací	1	50 slov	1
6	70	705	testovací	1	50 slov	1
7	83	705	testovací	1	50 slov	1
8	78	705	testovací	1	50 slov	1
9	103	705	testovací	1	50 slov	1
Ano	1	705	trénovaný	83	ano /ne	1
Ne	1	705	trénovaný	83	ano /ne	1
Aaaa	1,47	1411	testovací	1	ano /ne	2
Aaaaa	2,4	1411	testovací	1	ano /ne	2
Aan	1,33	1411	testovací	1	ano /ne	2
Anana	1,4	1411	testovací	1	ano /ne	2
Annan	1,8	1411	testovací	1	ano /ne	2
Naanna	2,53	1411	testovací	1	ano /ne	2
Nanna	2,07	1411	testovací	1	ano /ne	2
Nnnnn	1,67	1411	testovací	1	ano /ne	2
Nna	1,27	1411	testovací	1	ano /ne	2
Nnnnnn	2,33	1411	testovací	1	ano /ne	2

Název souboru	Délka trvání [s]	Přenosová rychlost [kbps]	vzorkování [bity]	Počet nahrávek	Vzorky použity v příkladu	Počet kanálů
nula	1	1411	trénovaný	6	cisla	2
jedna	1	1411	trénovaný	6	cisla	2
dve	1	1411	trénovaný	6	cisla	2
tri	1	1411	trénovaný	6	cisla	2
ctyri	1	1411	trénovaný	6	cisla	2
pet	1	1411	trénovaný	6	cisla	2
sest	1	1411	trénovaný	6	cisla	2
sedm	1	1411	trénovaný	6	cisla	2
osm	1	1411	trénovaný	6	cisla	2
devet	1	1411	trénovaný	6	cisla	2
O2	17,1	1411	testovací	1	cisla	2
opacne	8,3	1411	testovací	1	cisla	2
vse	11,71	1411	testovací	1	cisla	2
vse2	10,18	1411	testovací	1	cisla	2
vse3	10,37	1411	testovací	1	cisla	2
vse4	8,98	1411	testovací	1	cisla	2
O	12,47	1411	testovací	1	cisla	2
10 cisel	14,58	1411	testovací	1	cisla	2

8 ZÁVĚR

Hlavními úkoly v této diplomové práci bylo seznámit se s metodami statistického zpracování číslicových signálů a napsat přehled o softwarech, jež se k danému účelu používají. Dále byl vytvořen podrobnější popis o softwarových nástrojích, jež se používají ke zpracování signálů pomocí skrytých Markových signálů. Byl vytvořen návod na stažení a obecný příklad zpracování signálů v českém jazyce pomocí HTK doplněný schématickým vyjádřením pro lepší porozumění problematiky. Poté byla vytvořena databáze vhodných akustických signálů, jenž byla využita jako vstupní signály do zadaného programu HTK a následně byly vytvořeny jednoduché ukázkové příklady na zpracování reálných signálů pomocí HMM pomocí programu HTK v prostředí příkazové řádky v operační systému Windows. Příklady byly taktéž zdokumentovány podrobným popisem tak, aby jakýkoli uživatel zadaného softwaru byl schopen jejich samostatné reprodukce. Jak je vidět z grafů uvedených v 6. kapitole této diplomové práce, procentuální úspěšnost rozpoznávání řečových signálů kolísá a nezáleží ani tolik na počtu trénovacích signálů, ale spíše na rozličnosti trénovaných osob a testovaných souborů a také na pauze mezi jednotlivými slovy. Je patrné, protože není zaveden model pauzy kvůli zjednodušení a snížení počtu příkazů v jednotlivých příkladech, že tato skutečnost má velký vliv na rozpoznávání a je klíčové nastavit správně prahy (jenž jsou ale u všech příkladů jiné), jenž označují přesah od předcházejícího vyslovovaného slova a dalšího následujícího slova. Tudíž dochází také k případům při rozpoznávání obsaženém v této práci, že místo dvou slov jsou rozpoznány tři, což je nejčastější výskyt chyby v příkladech uvedených v této práci. Ale ve všech případech (mimo jednoho) je úspěšnost rozpoznávání vysoce nad 60 %, což značí fakt, že pravděpodobnost správného rozpoznávání byla vždy mnohem vyšší než špatná detekce slov. Platil fakt, že čím více slov v testovaných souborech (a v databázi testovacích souborů), tím pravděpodobnost úspěšného rozpoznání slov klesala. Pro zvýšení pravděpodobnosti by bylo tedy nutné zavést model pauzy, zavést rozpoznávání po hláskách, provést více příkazů než je zde uvedeno před vyhodnocením a samotným rozpoznáváním řeči. Také by pomohlo přesné rozpoznávání časových period, pokud by byla testovací databází dlouhá souvětí a ne jednotlivá slova. Ale tento fakt by také vedl k časovému prodloužení všch procesů před konečným vyhodnocením.

LITERATURA

[1] SMÉKAL, Z., ATASSI, H., STEJSKAL, V.,MEKYSKA, J., *Soubor programů pro práci se skrytými Markovovými modely (HTK)*, Brno, 2009, p. 42.

[2] HTK Speech Recognition Toolkit YOUNG, S., *The HTK Book*. Cambridge University Engineering Department,2009, p. 359 . Dostupné na www: <http://htk.eng.cam.ac.uk/>.

[3] *Cygwin Information and Installation*., 2008. Dostupné na www:<http://www.cygwin.com/>.

[4] *ActivePerl – The complete and ready-to-install Perl distribution*. ActiveState Software Inc., 2008. Dostupné na www:<http://www.activestate.com/Products/activeperl/index.mhtml>.

[5] *Windows: Download Cygwin, HTK, Julius and Audacity. VoxForge*, 2008. Dostupné na www:<http://www.voxforge.org/home/dev/acousticmodels/windows/create/htkjulius/tutorial/download>.

[6] SUBHRANSU, M. *Fast Automatic Alignment of Video and Text for Search/-Names and Faces*. University of California at Berkeley, 2007, 8 p., Dostupné na www:<http://www.cs.berkeley.edu/_smaji/reports/cs281a-final-report.pdf>.

[7] *Google Audio Indexing*, 2008, Dostupné na www:<http://labs.google.com/gaudi>.

[8] TOSHIFUMI, O., ATWELL, E. *Using the HTK speech recogniser to analyse prosody in a corpus of German spoken learners' English*. School of Computing, University of Leeds, 2003, 8 p., Dostupné na www:<http://ucrel.lancs.ac.uk/publications/CL2003/papers/oba.pdf>.

[9] VLASENKO, B., WENDEMUTH, A. *Tuning Hidden Markov Model for Sérech Emotion Recognition. Cognitive Systems Group*, IESK, Otto von Guericke University, 2007. 2 p. Dostupné na www:<http://iesk.et.uni-magdeburg.de/KO/papers/daga07 bv.pdf>.

[10] McKENNA, J. *Building a Simple Speaker Identification System. School of Computing*, Dublin City University, 2004, 6 p, Dostupné na www:<http://www.computing.dcu.ie/_john/SpeakerRecSimple.pdf>.

[11] *Touchless SDK. CodePlex – Open Source Project*, 2008. Dostupné na www:<http://www.codeplex.com/touchless/>.

[12] PERRIN, S. *Gesture Recognition Using Laser-Based Tracking System*. University of Tokyo, Ishikawa Hashimoto Laboratory, 2004, 6 p., Dostupné na www:<http://www.k2.t.u-tokyo.ac.jp/members/alvaro/Publications/tracking FG 2004 final.pdf>.

[13] CHEE YAU, W. *Cluster Analysis K–means*. School of Electrical and Computer Engineering, RMIT University, 2006, 17 p., Dostupné na www: <http://www.ee.unimelb.edu.au/ISSNIP/multimedia/events/cluster%20and%20hmm.pdf>.

[14] KIM, J. A *Study on Dicodon-oriented Gene Finding using Self–Identification Learning*. School of Knowledge Science, Japan Advanced Institute of Science and Technology, 2000, 57 p., Dostupné na www:<http://www.jaist.ac.jp/library/thesis/ks-master-2000/paper/ckim/paper.pdf>.

[15] RAJNOHA J.*Rozpoznávání řeči v reálných podmínkách na platformě standardního PC*, Praha, 2006, p.10.

[16] *Mendel HMM Toolbox*, 2006, Dostupné na www:< http://www.math.uit.no/bi/hmm/>.

[17] *Voicebox*:*Speech Processing Toolbox for MATLAB*, Dostupné na www:<http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html>.

[18] KAČUR, J., SCHRETER, R., PROCHÁSKA, J., Systém HTK-GUI a rozpoznávanie reči, 2004, 12 p., Dostupné na www:<http://www.ktl.elf.stuba.sk/~kacur/clanky/ELOSYS%202004%20HTK-GUI-out.pdf>.

[19] *Mel-frequency cepstrum*, Dostupné na www:<http://en.wikipedia.org/wiki/Mel-frequency_cepstral_coefficient>.

[20] *Linear predictive coding*, Dostupné na www:<http://en.wikipedia.org/wiki/Linear_predictive_coding>.

[21] *Perceptual Linear Prediction*, Dostupné na www:<http://mi.eng.cam.ac.uk/~ajr/SpeechAnalysis/node52.html>.

[22] SUKUMAR GHOSH, *Token-passing Algorithms*, 2008, 19 p., Dostupné na www:<www.cs.uiowa.edu/~ghosh/9-18-08.ppt>.

[23] *Baum-Welch Algorithm*, Dostupné na www:<http://jedlik.phy.bme.hu/~gerjanos/HMM/node11.html>.

[24] *Fenotyp*, Dostupné na www:<http://cs.wikipedia.org/wiki/Fenotyp>.

[25] *Diploid*, Dostupné na www:<http://cs.wikipedia.org/wiki/Diploid>.

[26] *Tetraploid*, Dostupné na www:<http://cs.wikipedia.org/wiki/Tetraploid>.

[27] *Databáze signálů*, Dostupné na www:<http://amber.feld.cvut.cz/user/cmejla/databaze/databaze.html>.

[28] *Databáze číslovek*, Dostupné na www:<http://www.locallingo.com/czech/grammar/numbers.html>.

[29] *Databáze slov ano/ne*, Dostupné na www:<http://www.fit.vutbr.cz/study/courses/ZRE/public/labs/06_htk/>.

[30] *CD slovní audiometrie*, NOVÝ, P. A *Česká slovní audiometrie* Fakulta aplikovaných věd, Západočeská univerzita, Plzeň, 1994.

[31] *Scientific computing*, Dostupné na www:< *Databáze signálů*, Dostupné na www:<http://www.scientificcomputing.com/uploadedImages/Images/0912/Maple13-Rev-fig1_lrg.jpg>.

[32] *Matlab Statistics Toolbox*, Dostupné na www:< *Databáze signálů*, Dostupné na www:<http://img.brothersoft.com/screenshots/softimage/s/statistics_toolbox-226585-1238401934.jpeg>.

[33] YOUNG, S. J., RUSSELL N. H., THORNTON J. H. S, *Token passing: a simple conceptual model for connected speech for recognition systems*, Cambridge University Engineering Department, July 1989.

[34] CHALUPNÍČEK K., *Rozpoznání diktované řeči pro radiologické aplikace*, Brno, 2004, p.17-25

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

q_j	stav skrytého Markovova modelu
n_{ij}	pravděpodobnost přechodu ze stavu n_i do stavu n_j
m_j	funkce rozdělení výstupní pravděpodobnosti
HMM	(the Hidden Markov Models) skryté Markovy modely
HTK	(Hidden Markov Model Toolkit) modul pro práci s HMM
LPC	lineární predikční koeficienty
MFCC	(Mel-Frequency Cepstral Coefficients) melovské kepstrální koeficienty
PLP	(Preceptual Linear Prediction) percepční lineární predikční koeficienty
transkripce	přepis mluveného slova do textové podoby
foném	nejmenší součást zvukové stránky řeči

SEZNAM PŘÍLOH

A MODEL 7 - STAVOVÉHO MODELU

Tento model má 7 stavů (zde označeno *NumStates*), první a poslední jsou speciální nevysílací, vysílacích stavů je tedy 5. Funkce hustoty rozdělení pravděpodobnosti v jednotlivých stavech budou modelovány pomocí jedné Gaussovy funkce s diagonální kovarianční maticí. Jedna funkce hustoty rozdělení pravděpodobnosti bude tedy popsána vektorem 39 středních hodnot (*VecSize*) a vektorem 39 rozptylů (*Variance*). Parametrizace se provede pomocí 12-ti MFCC (kepstrálních) koeficientů a logaritmu energie s tím, že vektor bude doplněn odhady prvních a druhých derivací (*MFCC_E_D_A*). Za příkazem *TransP* následuje matice přechodových pravděpodobností, kde pozice hodnoty na řádku n a sloupci m udává přechodovou pravdepodobnost $a_{n,m}$. *BeginHMM* označuje začátek skrytého Markovova modelu, *EndHMM* označuje konec skrytého Markovova modelu. *State* označuje aktuální stav, *Mean* význam (velikost) a *Streams* označuje nezávislé vektory.

```
  ~o <VecSize> 39 <MFCC_E_D_A> <StreamInfo> 1 39
<BeginHMM>
  <NumStates> 7
  <State> 2 <NumMixes> 1
  <Stream> 1
  <Mixture> 1 1.0000
    <Mean> 39
      0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0
    <Variance> 39
      1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0
  <State> 3 <NumMixes> 1
  <Stream> 1
  <Mixture> 1 1.0000
    <Mean> 39
      0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0
    <Variance> 39
      1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0
  <State> 4 <NumMixes> 1
  <Stream> 1
  <Mixture> 1 1.0000
    <Mean> 39
```

```
      0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0
    <Variance> 39
      1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0
  <State> 5 <NumMixes> 1
  <Stream> 1
  <Mixture> 1 1.0000
    <Mean> 39
      0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0
    <Variance> 39
      1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0
  <State> 6 <NumMixes> 1
  <Stream> 1
  <Mixture> 1 1.0000
    <Mean> 39
      0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0
    <Variance> 39
      1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0
  <TransP> 7
   0.000e+0    1.000e+0    0.000e+0    0.000e+0    0.000e+0    0.000e+0
0.000e+0
   0.000e+0    6.000e-1    4.000e-1    0.000e+0    0.000e+0    0.000e+0
0.000e+0
   0.000e+0    0.000e+0    6.000e-1    4.000e-1    0.000e+0    0.000e+0
0.000e+0
   0.000e+0    0.000e+0    0.000e+0    6.000e-1    4.000e-1    0.000e+0
0.000e+0
   0.000e+0    0.000e+0    0.000e+0    0.000e+0    6.000e-1    4.000e-1
0.000e+0
   0.000e+0    0.000e+0    0.000e+0    0.000e+0    0.000e+0    6.000e-1
4.000e-1
   0.000e+0    0.000e+0    0.000e+0    0.000e+0    0.000e+0    0.000e+0
0.000e+0
<EndHMM>
```

B MODEL 5 - STAVOVÉHO MODELU

Tento model má 5 stavů (zde označeno *NumStates*), první a poslední jsou speciální nevysílací, vysílacích stavů je tedy 3. Ostatní parametry jsou vysvětlené u 7- stavového modelu.

```
 ~o <VecSize> 39 <MFCC_E_D_A> <StreamInfo> 1 39
<BeginHMM>
<NumStates> 5
<State> 2
<Mean> 39
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0
<Variance> 39
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0
<State> 3
<Mean> 39
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0
<Variance> 39
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0
<State> 4
<Mean> 39
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0
<Variance> 39
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0
<TransP> 5
0.0 1.0 0.0 0.0 0.0
0.0 0.6 0.4 0.0 0.0
0.0 0.0 0.6 0.4 0.0
0.0 0.0 0.0 0.7 0.3
0.0 0.0 0.0 0.0 0.0
<EndHMM>
```