ZÁPADOČESKÁ
UNIVERZITA
V PLZNI

Fakulta aplikovaných věd

DISERTAČNÍ PRÁCE

k získání akademického titulu doktor

v oboru

Kybernetika

Ing. Jan Vaněk

DISKRIMINATIVNÍ TRÉNOVÁNÍ AKUSTICKÝCH MODELŮ

Školitel: Prof. Ing. Josef Psutka CSc.
Katedra: Katedra kybernetiky

Plzeň, 2009

University of West Bohemia

Faculty of Applied Science

DISERTATION THESIS

submitted for the degree of Doctor of Philosophy

in the field of

Cybernetics

Ing. Jan Vaněk

DISCRIMINATIVE TRAINING OF ACOUSTIC MODELS

Advisor: Prof. Ing. Josef Psutka CSc.
Department of Cybernetics

Pilsen, 2009

Prohlášení

Prohlašuji, že jsem tuto disertační práci vypracoval samostatně s použitím odborné literatury a pramenů, jejichž úplný seznam je její součástí.

V Plzni dne Jan Vaněk

Poděkování

Tato disertační práce vznikla za podpory:

- Ministerstva školství, mládeže a tělovýchovy v rámci projektu MŠMT 2C06020: *Eliminace jazykových bariér handicapovaných diváků České televize.*
- Grantové agentury Akademie věd v rámci projektu GA AV ČR 1QS101470516: *Automatické vyhledávání klíčových slov v proudu zvukových dat.*
- Ministerstva vnitra České republiky v rámci projektu VD20072010B160: *Překlenutí jazykové bariéry komplikující vyšetřování financování terorismu a závažné finanční kriminality.*
- Poděkování patří rovněž distribuovanému superpočítačovému projektu *MetaCentrum*, který vznikl v rámci výzkumného záměru MSM6383917201.

Dále bych chtěl poděkovat:

- svému školiteli Prof. Ing. Josefu Psutkovi CSc.,
- všem členům oddělení umělé inteligence Katedry kybernetiky, zvláště Ing. Mgr. Josefu V. Psutkovi, Ph.D., Ing. Aleši Pražákovi, Ph.D., Ing. Luboši Šmídlovi, Ph.D. a Ing. Janu Trmalovi.
- mé rodině a přítelkyni za jejich podporu, kterou mi v průběhu studia poskytovali.

Anotace

Tato disertační práce se zabývá diskriminativním trénováním akustických modelů pro systémy automatického rozpoznávání řeči. Jelikož je v současné době většina těchto systémů založena na skrytých Markovových modelech, je tato práce zaměřena na trénování právě těchto modelů.

V úvodní části práce je představen systém automatického rozpoznávání řeči, tak jak je v současné době nejčastěji používán. V následující kapitole je podrobněji popsáno akustické modelování založené na maximalizaci věrohodnosti, ze kterého většina diskriminativních metod trénování vychází. V další již rozsáhlejší kapitole jsou postupně podrobně přestaveny a popsány různé metody diskriminativního trénování, včetně jejich četných modifikací. Zároveň, kde to bylo možné, jsou i uváděny konkrétní výsledky, které byly dosaženy pomocí těchto metod na pracovištích ve světě. V závěru této kapitoly byly porovnány jednotlivé metody mezi sebou a výsledky diskutovány.

V praktické části práce bylo navrženo hned několik metod diskriminativního trénování a jejich modifikací. Některé z těchto metod dosáhly prokazatelně lepších výsledků než dosud publikované varianty. Jednalo se zejména o stabilizaci výpočtu a diskriminativní určení vah jednotlivých složek modelu.

Metody navržené v této práci (označované jako MMI, MMI-FD, MMI+MMI-FD a MMI-TF) byly porovnány na mnoha experimentech a několika korpusech. Pro velké úlohy, kde je akustický model již velice komplexní, se nejlépe hodí metoda MMI-FD. Ostatní metody se spíše hodí na úlohy menšího rozsahu, kde akustický model není tak komplexní. Z těchto metod dosahovala nejlepších výsledků metoda MMI-TF a to jen po jedné iteraci. Do všech těchto metod byla rovněž implementována také diskriminativní adaptace, která dosahuje velmi dobrých výsledků na experimentech se skupinovými modely. Jak diskriminativní trénování, tak i adaptace byly testovány také na úloze detekce klíčových slov. I zde bylo těmito metodami dosaženo významného zlepšení.

Velká pozornost byla také věnována návrhu optimalizačních technik pro implementace těchto metod, tak aby výsledky byly k dispozici v co nejkratším čase, bez extrémních nároků na výpočetní výkon a paměť počítače. Optimalizovaná verze diskriminativního trénování, implementovaná v rámci této práce je schopná natrénovat akustický model za 2-20% celkového času trénovacích nahrávek. Tento software je v současné době úspěšně používán pro trénování modelů v rámci projektů řešených na katedře.

Klíčová slova: akustické modelování, diskriminativní trénování, diskriminativní adaptace, skryté Markovovy modely.

Annotation

This thesis proposes a new approach to discriminative training of speech recognition systems, with emphasis on systems based on hidden Markov models.

The first part describes existing speech recognition systems and the maximum likelihood method commonly used for the acoustic models today. Existing methods of discriminative training and their variations are then discussed and, when available, practical results are compared.

The second part presents several new methods for discriminative training, some of them give significantly better results than models reported by others. The main innovation relates to the stability of the algorithm and how the mixture weights are being set.

The new methods (acronyms MMI, MMI-FD, MMI+MMI-FD, MMI-TF) were tested on many experiments with several corpora. For large problems with complex acoustic models, we recommend the MMI-FD method. For less complex models, MMI-TF generates the best results, surprisingly in a single iteration. All four methods already include discriminative adaptation, which proved to be particularly useful for clustered models. In our tests of keyword detection, discriminative training combined with discriminative adaptation significantly improved the results.

The third part describes the software which implements the new methods. The algorithms have been optimized to generate the results in the shortest possible time, without extreme demands on the computer storage and speed. The model can be trained in mere 2-20% of the real total time of the training speech. The software is being used for training of acoustic models in several projects currently under development at our department.

Keywords: acoustic modeling, discriminative training, discriminative adaptation, hidden Markov models.

Obsah

Seznam tabulek

Seznam obrázků

1 Úvod

Většina v současnosti používaných systémů pro automatické rozpoznávání řeči používá akustické modely založené na skrytých Markovových modelech. Standardním přístupem pro trénovaní parametrů těchto akustických modelů je iterativní algoritmus optimalizující kriterium maximální věrohodnosti. Tento algoritmus poskytuje poměrně kvalitní model, má zaručenu konvergenci iterativního procesu trénování a má relativně nízké výpočetní nároky.

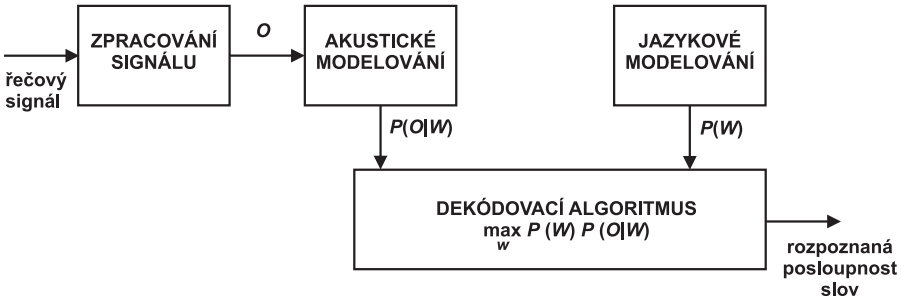
Jelikož skryté Markovovy modely jsou pouze aproximativním modelem vytváření řeči člověkem, nejsou splněny některé z předpokladů kladené na tento standardní trénovací postup. Algoritmus pak konverguje k parametrům, které jsou sice poměrně dobře odhadnuté, nicméně nejsou optimální. Je zde tedy prostor pro jiné metody, které dokáží odhadnout tyto parametry blíže k parametrům optimálním. Tím bude rovněž docházet k zlepšení výsledků celého systému rozpoznávání řeči.

Jedním z řešení jak dosáhnout lepších výsledků je využití tzv. diskriminativního trénování. Tato třída postupů a algoritmů je založena na učení se nejen z pozitivních příkladů, ale i z příkladů negativních. Tento princip vede k mnohem větší komplexnosti takovýchto metod, zároveň i k mnohem větší výpočetní náročnosti. Má však potenciál dosahovat lepších výsledků než trénování založené na maximalizaci věrohodnosti.

Cílem této disertační práce je prostudovat tuto třídu metod, vhodné metody implementovat, případně navrhnout i vlastní modifikace jednotlivých metod. Dále tyto metody porovnat jak s referenčním standardním trénovacím postupem, tak mezi sebou. Vzhledem k tomu, že drtivá většina publikovaných výsledků a experimentů byla prováděna na řečových nahrávkách v anglickém jazyce, je žádoucí zhodnotit tyto metody rovněž pro co největší množství korpusů v jazyce českém. Ten se od anglického jazyka z pohledu automatického rozpoznávání řeči v mnohém liší. Nedílnou součástí implementace takto extrémně výpočetně i paměťově náročných metod je optimalizace takovéto implementace. Je důležité, aby výsledné algoritmy byly schopny zpracovávat i největší dostupné korpusy dostatečně rychle. Je třeba rovněž počítat s tím, že do budoucna bude velikost dostupných trénovacích dat i velikost trénovaných modelů narůstat. Praktická implementace by pak měla vést k softwarovému nástroji, který by bylo možné využít v projektech řešených na katedře.

2 Postupy používané pro rozpoznávání řeči

Problematika automatického rozpoznávání řeči se vyvíjela postupně spolu s narůstajícím výkonem počítačů. Nejprve bylo řešeno rozpoznávání několika málo jedno-slovních povelů, postupně mohl být slovník rozšiřován, což vedlo k řešení úlohy diktátu izolovaných slov. Později pak bylo možné rozpoznávat i krátké sekvence slov - například série číslovek, což mohly být například telefonní čísla. V současné době je již možné rozpoznávat prakticky neomezeně dlouhé nahrávky zcela spontánní řeči. Pro tyto úlohy se v současné době používá výhradně statistický přístup založený na *skrytých Markovových modelech* (angl. *Hidden Markov Models - HMM*). Jeho zjednodušené schéma je na obrázku 1. Rozpoznávací



Obrázek 1: *Schéma statistického rozpoznávání řeči*

systém zde generuje nejpravděpodobnější posloupnost slov W^*, kde W^* je vybírána *dekódovacím algoritmem* podle vztahu $W^* = \arg\max_W P(\boldsymbol{O}|W)P(W)$. Apriorní pravděpodobnost posloupnosti slov $P(W)$ generuje *jazykový model*. $P(\boldsymbol{O}|W)$ je pravděpodobnost pozorování sekvence příznakových vektorů $\boldsymbol{O}$ při dané posloupnosti slov W, kterou generuje modul *akustického modelování*. Sekvence příznakových vektorů $\boldsymbol{O}$ vznikne analýzou vstupního akustického signálu v modulu *zpracování signálu*.

Pokud má být rozpoznávací systém robustní na změny prostředí, ve kterém je nahrávka pořizována, je potřeba se soustředit na moduly *zpracování signálu* a *akustického modelování*.

Metody zpracování řečového signálu se nejčastěji snaží napodobit proces zpracování řeči sluchovým orgánem člověka, který je k tomu dokonale vyvinut dlouhou evolucí. U člověka pak zpracovaný signál putuje do mozku, kde se miliardy neuronů, trénované po celý život člověka, postarají o porozumění promluvě. To již lze modelovat jen těžko. Proto jsou zde nasazovány nejrůznější techniky z oborů

zpracování signálu, rozpoznávání obrazů, umělé inteligence a statistiky.

2.1 Metody zpracování signálu

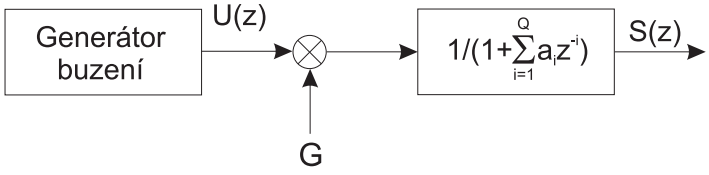
Metody zpracování signálu jsou motivovány způsobem, jakým člověk vytváří řeč, a také fyziologií lidského ucha. Zpravidla se jedná o analýzu spektrální charakteristiky akustického signálu v krátkém časovém úseku, kde lze přibližně považovat parametry hlasového ústrojí za konstantní. Tento krátký úsek se nazývá *mikrosegment*. Délka mikrosegmentů se pohybuje od 15 do 35 milisekund. Ze signálu jsou jednotlivé mikrosegmenty extrahovány použitím Hammingova okénka, které zdůrazní centrální část úseku signálu. Po sobě jdoucí mikrosegmenty se navzájem překrývají, aby byla využita i informace, která je v jednom okénku mimo centrální část. Tedy posun okénka je menší než jeho délka a bývá obvykle 10 až 15 milisekund. Na začátku zpracování je ještě vhodné zdůraznit vyšší kmitočty, které mají obecně nižší energetickou úroveň, ale pro rozpoznávání jsou stejně důležité.

2.1.1 Lineární prediktivní kódování

Metoda lineárního prediktivního kódování (angl. *Linear Predictive Coding - LPC*) [1] provádí na krátkodobém základu odhad parametrů řečové produkce. Princip metody LPC je založen na předpovědi k-tého vzorku signálu lineární kombinací Q vzorků předchozích.

$$s(k) = -\sum_{i=1}^{Q} a_i s(k-i) + Gu(k), \tag{1}$$

kde $u(k)$ je budicí signál a G zesílení modelu. Odpovídající schéma je uvedeno na obrázku 2.



Obrázek 2: *Schéma lineárního prediktivního kódování*

Určení hodnot a_i a G je prováděno pomocí metody nejmenších čtverců. Tento lineární model hlasového traktu můžeme také popsat kepstrálními koeficienty LPC.

Kepstrální koeficienty získáme pomocí Taylorova rozvoje $log[G/A(z)]$. Počet kepstrálních koeficientů pak může být nižší než původní řád modelu Q.

Z hlediska použitelnosti v moderních systémech rozpoznávání řeči tato metoda příliš nevyhovuje, protože je velice citlivá na aditivní šum.

2.1.2 Melovské frekvenční kepstrální koeficienty

Metoda Melovských frekvenčních kepstrálních koeficientů (angl. *Mel-Frequency Cepstral Coefficients - MFCC*) [1] vychází ze způsobu vnímání řeči člověkem. Konkrétně se jedná o *kritická pásma slyšení* a také o *subjektivní vnímání výšky tónů*. Po diskrétní Fourierově transformaci mikrosegmentu jsou na výkonové spektrum aplikovány trojúhelníkové pásmové filtry. Tyto filtry jsou rozmístěny na frekvenční ose nelineárně podle rovnice

$$f_{mel} = 2595 \log_{10}\left(1 + \frac{f_{Hz}}{700}\right). \tag{2}$$

Zlogaritmované energie M pásmových filtrů jsou dekorelovány zpětnou cosinovou transformací. j-tý kepstrální koeficient je vypočten z energií jednotlivých pásmových filtrů y_i, kde $i = 1 \ldots M$, následujícím způsobem

$$c(j) = \sum_{i=1}^{M} \log_{10}[y(i)] cos\Big[\frac{\pi j}{M}(i - 0,5)\Big], \qquad \text{pro } j = 0, 1, \ldots, N, \tag{3}$$

kde N je počet kepstrálních koeficientů, které jsou vypočítány. Jejich počet je často menší než je počet pásmových filtrů.

2.1.3 Perceptivní lineární prediktivní analýza

Metoda *Perceptivní Lineární Prediktivní analýzy* [24] také vychází z vnímání řeči člověkem. Je však důslednější. Kromě *kritických pásem slyšení* jsou uplatněny také *nelineární vnímání hlasitosti* a *nelineární vztah mezi intenzitou a hlasitostí*. Na mikrosegmenty je opět aplikována diskrétní Fourierova transformace. Pásmové filtry nejsou trojúhelníkové, ale mají jiný složitější tvar. Na frekvenční ose jsou rozmístěny opět nelineárně a to podle Barkovy stupnice

$$f_{Bark} = 6 \ln\left\{\frac{f_H z}{600} + \sqrt{\left(\frac{f_H z}{600}\right)^2 + 1}\right\}. \tag{4}$$

Jednotlivé filtry nejsou stejně vysoké, ale jsou přizpůsobeny *křivce stejné hlasitosti*. Dále je uplatněna nelineární závislost mezi intenzitou a hlasitostí. Ta je implementována jako $0,3$ mocnina energie jednotlivých pásmových filtrů. Posledními kroky jsou aproximace umocněných energií celopólovým modelem a výpočet kesptrálních koeficientů. Postup obdobný výpočtu koeficientů LPC.

2.1.4 Dynamické koeficienty

Dynamické koeficienty obecně vyjadřují dynamiku (změnu v čase) vektorů příznaků a zlepšují tím popis užitečné informace v akustickém signálu. Tyto příznaky jsou vypočítány z vektorů příznaků, získaných metodami popsanými výše. Dynamické koeficienty jsou k těmto vektorům přidány. Jejich výpočet se uskutečňuje numerickou aproximací derivace

$$d_t = \frac{\sum_{k=1}^{N} k(c_{t+k} - c_{t-k})}{2\sum_{k=1}^{N} k^2}, \tag{5}$$

kde c_t je kepstrální koeficient v čase t získaný některou z metod zpracování signálu. N je řád numerické derivace a určuje jak velké okolí aktuálního mikrosegmentu bude do výpočtu zahrnuto, obvykle se volí 2 až 4. Výsledkem je dynamický koeficient d_t, který je označován jako takzvaný *delta* koeficient. Vypočteny mohou být i vyšší řády derivace, v praxi se používají jen první dva, koeficienty druhého řádu se nazývají *delta-delta* nebo také *akcelerační*. Jejich výpočet je stejný, jen jako vstup neslouží statické koeficienty, ale *delta* koeficienty. Oproti základnímu vektoru příznaků dosahuje pak takto rozšířený vektor dimenze dvoj- nebo trojnásobné.

Přidání těchto koeficientů výrazně zvyšuje úspěšnost rozpoznávání, navíc jsou tyto koeficienty zcela odolné proti konstantnímu nebo pomalu se měnícímu konvolučnímu šumu a vzhledem k výpočtu přes několik mikrosegmentů, částečně také odolávají šumu aditivnímu.

2.2 Akustické modelování

Akustický model v systému rozpoznávání řeči, jak byl popsán v kapitole 2, poskytuje odhad podmíněné pravděpodobnosti $P(\boldsymbol{O}|W)$ pro pozorovanou posloupnost vektorů příznaků $\boldsymbol{O}$ a každou uvažovanou posloupnost slov W. Posloupnost vektorů $\boldsymbol{O}$ je získána zpracováním vstupního signálu a akustický model musí být schopen odhadnout podmíněnou pravděpodobnost $P(\boldsymbol{O}|W)$ pro libovolný vstupní signál. Akustický model by měl být schopen dostatečně dobře odlišit

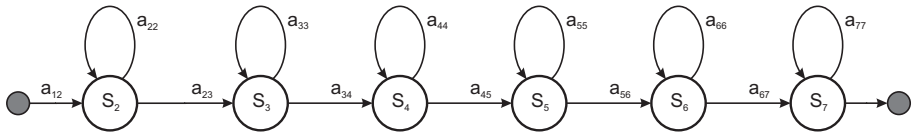
různé hlásky od sebe i ve foneticky velmi podobných posloupnostech slov. Na druhou stranu by měl být schopen zobecňovat na testovací data. Tedy poskytovat dobré odhady i v případech, kdy se od sebe liší podmínky, ve kterých je systém provozován, od podmínek trénovacích (šum na pozadí, přenosový kanál, hlas řečníka, tempo řeči, nářečí či odlišný způsob artikulace). Kromě těchto základních požadavků na akustický model je také důležité brát ohled i na výpočetní náročnost odhadu pravděpodobnosti. Zejména v systémech, které pracují v reálném čase, musí být akustický model kompromisem mezi přesností rozpoznávání a výpočetní náročností. Jako nejúspěšnější řešení těchto často protichůdných požadavků se za poslední více než dvě dvě dekády ukázal být model založený na tzv. *skrytých Markovových modelech* (angl. Hidden Markov Models - HMM).

2.2.1 Skryté Markovovy modely

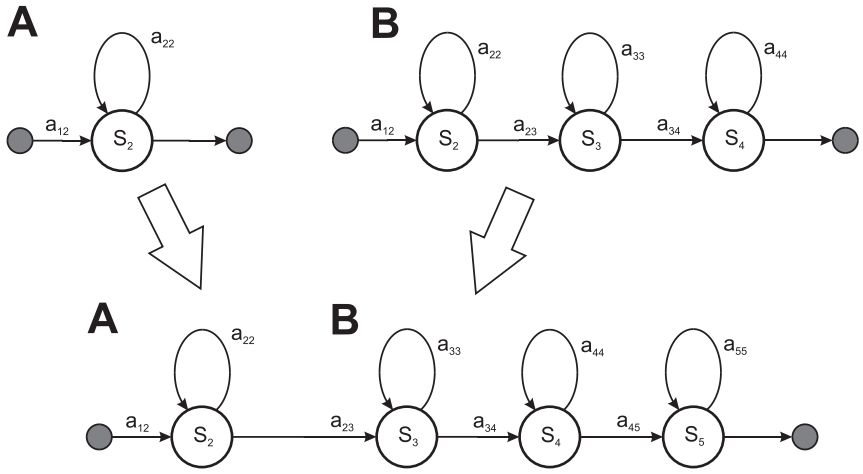
Skrytý Markovův model je model stochastického procesu, na který je možné pohlížet jako na pravděpodobnostní automat, který v diskrétních časových okamžicích generuje náhodnou posloupnost pozorování - vektorů příznaků $\boldsymbol{O} = \{\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_T\}$. V každém časovém kroku změní model svůj stav podle souboru předem daných pravděpodobností přechodu. Stav, do kterého model přejde, generuje příznakový vektor $\boldsymbol{o}_t$, a to podle rozdělení výstupní pravděpodobnosti příslušné k tomuto stavu. Při modelování řeči se využívají zejména tzv. levo-pravé Markovovy modely, které jsou vhodné pro modelování procesů, jejichž vývoj je spojen s postupujícím časem. V těchto modelech, celý proces začíná v počátečním stavu modelu příchodem prvního vektoru příznaků. Se vzrůstajícím časem postupně přechází na stavy z vyššími indexy a končí posledním příznakovým vektorem v koncovém stavu.

Konkrétní použití HMM závisí na připravované úloze. V případě rozpoznávání izolovaných slov s malým slovníkem je pro každé slovo ze slovníku vytvořen jeden model. Počet stavů pro každé slovo může být různý, ale z důvodů jednoduché implementace je obvykle volen počet stavů jednotný, který postačuje pro dobrý popis delších slov ve slovníku. Příklad HMM, který může být použitý pro modelování izolovaných slov je na obrázku 3. U úloh s větším slovníkem je modelování všech celých slov již těžkopádné a v praxi nepoužitelné. Je třeba modelovat menší jednotky než jsou slova. Takovými jednotkami mohou být například fonémy. Při modelování menších jednotek, pak máme pro trénování k dispozici mnohem větší počet trénovacích příkladů, než by tomu bylo u celých slov. Výsledný model je pak robustnější tedy velmi dobře zobecňuje a tedy není náchylný na akustické odlišnosti mezi trénovacími a testovacími nahrávkami. Fonémy bývají

modelovány jednoduššími HMM se třemi nebo pěti stavy, kde první a poslední stav je tzv. *neemitující*, tj. slouží pouze pro spojování s ostatními fonémy a negeneruje žádná pozorování ani výstupní pravděpodobnosti. Ostatní stavy jsou tzv. *emitující*. Příklad fonémových modelů a jejich řetězení je na obrázku 4. Klíčová



Obrázek 3: *Příklad lineárního levo-pravého HMM pro modelování izolovaných slov*



Obrázek 4: *Příklad tří- (A) a pěti-stavového (B) HMM a jejich řetězení*

část HMM, která má na funkci tohoto akustického modelu největší vliv, je model výstupní pravděpodobnosti jednotlivých stavů. Ten může být navržen několika způsoby:

Diskrétní rozložení Při tomto rozložení jsou vektory pozorování kvantovány metodou *vektorové kvantizace* podle *kódové knihy*. Vektorový kvantizér přiřazuje každý vektor pozorování k jednomu vzoru z kódové knihy. Takže každé pozorování nabývá pouze diskrétních hodnot - je to index do kódové knihy. Výstupní pravděpodobnost stavu je pak modelována jednoduše, jako relativní výskyt vzoru

v kódové knize v trénovacích datech. Tento přístup má velmi malé výpočetní nároky, tedy lze ho snadno použít v systémech, které musí pracovat v reálném čase i na přenosných zařízeních s malým výpočetním výkonem. Na druhou stranu oproti modelům se spojitým rozložením dosahuje nižší úspěšnosti rozpoznávání a v dnešní době již i mobilní zařízení mají k dispozici dostatečný výkon pro nasazení komplexnějších akustických modelů.

Spojité rozložení se směsí normálních hustotních funkcí (angl. Gaussian Mixture Model - GMM) Je navrženo pro modelování obecných hustotních funkcí, zejména v prostorech s vysokou dimenzí, kde hodnoty příznakových vektorů v jednotlivých dimenzích mohou nabývat libovolných reálných hodnot. Výstupní pravděpodobnost pro jeden vektor pozorování $\boldsymbol{o}$ a GMM, který je charakterizován parametry λ, je vážený součet M rozložení - složek. Tato pravděpodobnost $p(\boldsymbol{o}|\lambda)$ je dána vzorcem

$$p(\boldsymbol{o}|\lambda) = \sum_{i=1}^{M} c_i p_i(\boldsymbol{o}), \tag{6}$$

kde $\boldsymbol{o}$ je N-dimenzionální vektor pozorování, $p_i(\boldsymbol{o})$, $i = 1, \ldots, M$, jsou hustoty jednotlivých složek modelu a c_i, $i = 1, \ldots, M$ jsou váhy těchto složek. Každá složka modelu je pak N-dimenzionální normální rozložení, jehož hustota pro vektor pozorování $p_i(\boldsymbol{o})$ je dána

$$p_i(\boldsymbol{o}) = \frac{1}{(2\pi)^{N/2}|\boldsymbol{C}_i|^{1/2}} \exp\left[(\boldsymbol{o}-\boldsymbol{\mu}_i)^T \boldsymbol{C}_i^{-1}(\boldsymbol{o}-\boldsymbol{\mu}_i)\right], \tag{7}$$

kde $\boldsymbol{\mu}_i$ je vektor středních hodnot a $\boldsymbol{C}_i$ je kovarianční matice. Váhy jednotlivých složek pak splňují následující předpoklad

$$\sum_{i=1}^{M} c_i = 1. \tag{8}$$

Celá směs rozložení je tedy charakterizována vektorem středních hodnot, kovarianční maticí a váhou pro všech M složek modelu. Tyto parametry mohou být souhrnně reprezentovány pro celý HMM včetně pravděpodobností přechodu a_{ij} následovně:

$$\lambda = \{a_{ij}, c_i, \boldsymbol{\mu}_i, \boldsymbol{C}_i\}, \quad i = 1, \ldots, M. \tag{9}$$

Z důvodů větší robustnosti odhadů parametrů modelu a zvýšení rychlosti jeho vyhodnocování je často redukována kovarianční matice na pouze diagonální. Komplexnější hustotní funkce je výhodnější modelovat spíše větším počtem složek diagonálních než modelem s plnými kovariančními maticemi. Při použití diagonálních kovariančních matic je však třeba zajistit, aby vstupní příznakové

vektory byly v jednotlivých dimenzích nekorelované. To lze zajistit buď vhodnou metodou zpracování signálu nebo lze použít různých dekorelačních technik. Některé z nich umožňují i redukci dimenze příznakových vektorů, což může významně zrychlit celý systém. Mezi tyto techniky patří například *Principal Component Analysis - PCA*, *(Heteroscedastic) Linear Discriminant Analysis - (H)LDA* [2] či *Independent Component Analysis - ICA* [3].

Trifonové modely Náhrada modelu celých slov za modely jednotlivých fónů a jejich řetězení přináší větší flexibilitu modelu, ale takovéto modely nedostatečně zachycují informaci o okolí modelovaného fonému - tzv. *koartikulaci*. Jednou z možností jak tento koartikulační kontext modelovat, která je v současné době velmi využívána, je nasazení tzv. *kontextově závislých fonémů*. Pokud uvažujeme levý a pravý kontext, pak se jedná o *trifon*. Jistým nedostatkem trifonových modelů je relativně velké množství všech možných trifonů (kombinace všech uspořádaných trojic fonémů). Tento nedostatek lze významně zmírnit aplikací technik shlukování trifonů s akusticky podobnými kontexty, rovněž je také možné shlukovat modely jednotlivých stavů náležící k různým kontextům od jednoho fonému [1]. Shlukování samotné pak probíhá na základě předem připraveného fonetického rozhodovacího stromu nebo na základě trénovacích dat.

2.3 Metody jazykového modelování

Jazykový model je po modulu zpracování signálu a akustickém modelu další důležitou částí systému rozpoznávání mluvené řeči. Úkolem jazykového modelu je poskytnout dekódovacímu algoritmu odhad apriorní pravděpodobnosti $P(W)$ pro libovolnou posloupnost slov W. Součástí jazykového modelu je také slovník všech slov, který systém dokáže rozpoznat včetně jejich fonetického přepisu. Ke každému slovu může existovat ve slovníku i několik výslovnostních variant, dekódovací algoritmus pak uvažuje všechny varianty. Zároveň se ve slovníku mohou vyskytovat různá slova, která se vyslovují stejně a tedy mají stejný fonetický přepis. Mezi nimi může dekódovací algoritmus rozhodnout jen na základě slovního kontextu. Informace o kontextu se právě snaží zachytit a modelovat jazykový model. Jazykový model může odhadovat pravděpodobnost $P(W)$ různým způsobem, v současné době jsou však jednoznačně nejpoužívanější tzv. stochastické n-gramové jazykové modely.

2.3.1 Stochastický n-gramový jazykový model

Pro konstrukci jazykového modelu požadujeme znalost apriorních pravděpodobností $P(w_1^K)$ všech posloupností slov libovolné délky K. Všechny tyto pravděpodobnosti je obtížné a téměř nemožné ocenit. Provádí se proto jejich aproximace, kdy je brána v úvahu pouze omezená historie posloupnosti slov. Posloupnosti, které se shodují na $n-1$ posledních slovech jsou zařazeny do stejné třídy. Uvedené modely se nazývají obecně *n-gramové modely*. Pro nejběžnější konkrétní hodnoty n jsou pak používány tyto názvy: *zerogramový model* pro $n = 0$, *unigramový model* pro $n = 1$, *bigramový model* pro $n = 2$ a pro $n = 3$ *trigramový model*. Modely většího řádu se používají již velmi zřídka, jelikož je problém nashromáždit dostatečné množství trénovacích dat pro modely takto vysokého řádu. Na závěr poznamenejme, že pro flexibilní český jazyk, který navíc nemá pevný pořádek slov ve větě, je řádově obtížnější zkonstruovat dobrý jazykový model pro danou úlohu než třeba pro jazyk anglický.

2.3.2 Posouzení kvality stochastického jazykového modelu

Stejně jako při konstrukci libovolného jiného modelu i v jazykovém modelování je dobré umět posoudit kvalitu daného modelu. Nejvěrohodnějším kritériem kvality je test celého systému rozpoznávání na dostatečném množství testovacích dat. Pro praktický vývoj jazykových modelů je však dobré mít i nějaké kriterium kvality, které umožní posoudit kvalitu jazykového modelu jako takového. Nejpoužívanější mírou, kterou lze posoudit kvalitu jazykového modelu je tzv. *perplexita*. Perplexitu PP definujeme vztahem

$$PP = \frac{1}{\sqrt[K]{\bar{P}(w_1 w_2 \dots w_K)}}, \tag{10}$$

kde K je počet slov ve vyhodnocovaném textu, $\bar{P}(w_1 w_2 \dots w_K)$ je odhad pravděpodobnosti této posloupnosti slov, daným jazykovým modelem. Hodnotu perplexity mohou ovlivňovat dva faktory, samotná kvalita jazykového modelu, ale zároveň i složitost dané úlohy. Čím je slovník pro danou úlohu větší a jazyk s volnějšími vazbami, tím je hodnota perplexity větší a samotné rozpoznávání řeči obtížnější. Tedy hodnotou perplexity lze porovnávat nejen jazykové modely na konkrétní úloze, ale i jednotlivé úlohy nebo textové korpusy mezi sebou. Samotná hodnota PP pak odpovídá takové složitosti úlohy, jako kdybychom měli úlohu se všemi slovy stejně pravděpodobnými a počet slov ve slovníku se rovnal hodnotě PP.

Definice perplexity vyžaduje slovník, ve kterém jsou všechno slova z daného textu. Jestliže se vyskytuje v textu slovo, které není ve slovníku (angl. Out-Of-Vacabulary - OOV) perplexita bude nabývat nekonečné hodnoty. Pro vyhodnocení takovéhoto textu je nutné tyto slova do slovníku přidat. Nebo lze vyhodnotit perplexitu jen na slovech ze slovníku, pak je ovšem pro dobré porovnání kvality jazykových modelů též uvádět i (relativní) počet OOV slov.

2.3.3 Metody pro odhad pravděpodobností n-gramových modelů

Jádrem konstrukce n-gramového jazykového modelu je odhad n-gramových pravděpodobností z trénovacího korpusu - textu. Pro odhad těchto pravděpodobností je možné použít například metodu *Maximální věrohodnosti* (angl. Maximum Likelihood - ML) [1]. Ta odhaduje pravděpodobnosti výskytu slov na základě relativních četností n-gramů v trénovacím textu. Uvážíme-li ovšem, že pokud je ve slovníku V slov, je počet všech možných n-tic V^n. Množství trénovacích dat, pro kvalitnější odhad pravděpodobnosti všech těchto n-tic je extrémní již pro velmi malé řády modelu (bigram, trigram). Odhad řídce se vyskytujících jevů pak může být velmi nepřesný, některé se například vůbec v trénovacím textu nemusí vyskytnout a měly by pak nulovou pravděpodobnost. Tedy rozpoznávací systém by je nemohl nikdy rozpoznat. Pro tyto případy využíváme tzv. metody *vyhlazování* (angl. smoothing). Ty přiřadí n-gramům, které se vůbec nebo jen ojediněle vyskytovaly v trénovacím textu o něco vyšší (alespoň nenulovou) pravděpodobnost a naopak četnějším n-gramům zase pravděpodobnost mírně sníží, tak aby platila věta o úplné pravděpodobnosti. Vyhlazovacích metod existuje velké množství, zde je výčet nejdůležitějších z nich:

- Bayesova metoda odhadu [9]
- Goodův-Turingův odhad [1]
- Ústupové (backing-off) a interpolační schéma [10, 11]
- Katzův diskontní model [12, 13]

2.4 Dekódovací techniky

Modul dekodéru má za úkol vybrat nejpravděpodobnější posloupnost slov podle kriteria

$$W^* = \arg\max_W P(\boldsymbol{O}|W)P(W), \tag{11}$$

na základě informací od akustického a jazykového modelu. Implementace dekódovacího algoritmu v praxi se setkává s hned několika zásadními problémy. Vzhledem k obrovskému množství všech posloupností slov, které by musely být vyhodnoceny je možné pouze nějakým aproximativním či heurictickým způsobem vybrat ty nejpravděpodobnější z nich. Situaci komplikuje též absence apriorní znalosti o počtu slov v rozpoznávané promluvě. Vhodné metody pro řešení tohoto problému jsou například *Viterbiho algoritmus* doplněný o tzv. *prořezávání* pro snížení výpočetní náročnosti prohledávání [21, 1]. Nebo tzv. *zásobníkový dekodér* (angl. stack decoder) využívající pro uložení rozvíjených hypotéz zásobník [22].

V praktických aplikacích je třeba základní kritérium (11) trochu rozšířit. Je třeba vhodně vyvážit příspěvky akustického a jazykového modelu. K tomuto vyvážení je používána tzv. *váha či měřítko jazykového modelu* (angl. language model scale factor). Další úprava kriteria je nutná pro korekci příliš velkého množství chyb typu vložení (inzerce). To je kompenzováno tzv. *penaltou vložení*, která mění měřítko $P(O|W)P(W)$ v závislosti na počtu slov promluvy ve vyhodnocované hypotéze. Pokud zapracujeme výše uvedené modifikace do vzorce 11 dostáváme rovnici ve tvaru

$$W^* = \arg\max_W \left[\log P(\boldsymbol{O}|W) + \kappa_1 \log\left(P(W) + \kappa_2 H\right)\right], \tag{12}$$

kde κ_1 je váha jazykového modelu, κ_2 je penalta vložení slova a H je počet slov aktuální hypotézy. Při přípravě reálného systému rozpoznávání řeči jsou pak parametry κ_1 a κ_2 nastavovány experimentálně, penaltu vložení můžeme nastavovat s přihlédnutím k poměru chyb vložení (inzerce) a smazaní (deletizace).

3 Cíle disertační práce

Ze schematu statistického rozpoznávání řeči (obrázek 1) uvedeného v kapitole 2 je zřejmé, že akustické modelování je jednou z klíčových částí systému a tedy jeho kvalita zásadně ovlivňuje kvalitu celého systému. Jak již bylo stručně uvedeno v úvodu, standardní postup trénování založený na maximalizaci věrohodnosti má jisté výhody, existují však metody komplexnější, které umožňují dosáhnout lepších výsledků. Tato disertační práce je věnována třídě tzv. diskriminativních technik a jejich praktické implementaci.

3.1 Dílčí cíle práce

1. Prostudovat a popsat jak standardní trénovací postup založený na maximalizaci věrohodnosti tak metody diskriminativního trénování. Z dostupných publikovaných výsledků zjistit jakých bylo dosahováno zlepšeních oproti standardnímu postupu a na jakých úlohách.

2. Navrhnout a realizovat software, který bude umožňovat jak standardní trénování podle kriteria maximální věrohodnosti, tak trénování založené na kriteriích diskriminativních. Software by měl být navržen tak, aby ho bylo možné snadno rozšiřovat o další metody, případně stávající metody snadno modifikovat. Dále musí umožňovat paralelní spouštění na několika procesorech, počítačích nebo v superpočítačovém centru. Finální implementace by měla podporovat jak operační systém Windows, tak Linux a měla by být navržena natolik vhodně, aby výsledné metody mohly být aplikovány i na největší korpusy dat a trénování velmi velkých modelů trvalo únosnou dobu.

3. Ve výzkumné části práce se věnovat návrhu zcela nových či modifikovaných nebo kombinovaných diskriminativních metod trénování a rovněž se věnovat vhodnému nastavení parametrů těchto metod.

4. Experimenty provést na maximálním množství různých dostupných dat. Soustředit se zejména korpusy obsahující řečové nahrávky v českém jazyce a na úlohy řešené na katedře kybernetiky.

4 Základní algoritmy pro trénování akustických modelů

V této kapitole bude stručně popsán základní postup při trénování trifonových akustických modelů, který pak zpravidla bývá brán jako výchozí model pro techniky diskriminativního trénování, které budou rozebírány v následujících kapitolách. Pro trénovaní modelů může být použit například velmi populární *HTK toolbox* [23], kde je i detailně popsán následující postup.

4.1 Příprava dat

Před samotným trénováním akustických modelů je třeba připravit hned několik věcí:

- Samotné trénovací nahrávky je třeba zpracovat některou metodou zpracování signálu (viz kapitola 2.1) do posloupnosti příznakových vektorů. Dále je třeba je rozdělit na vhodně veliké úseky, například věty.
- Ke všem trénovacím promluvám musí být připraven referenční přepis. Tedy textový soubor, který obsahuje přesný přepis vyřčených slov, včetně anotace neřečových událostí a ruchů či šumů zaznamenaných v nahrávce. Přepis může obsahovat i časové značky označující například začátky a konce vět, nebo změny řečníků.
- Rovněž musí být navržena fonetická abeceda, tedy seznam fonémů, které budou akustickým modelem modelovány. Do fonetické abecedy patří i neřečové události, tedy i pauzy, nádechy či jiné druhy zvuků vyskytující se v záznamech, které ale nepatří k vyřčeným slovům.
- Slovník výslovností. Ten musí obsahovat všechna slova, která se vyskytují v přepisech. U všech těchto slov musí být uveden přepis na jednotlivé fonémy z fonetické abecedy. Některá slova mohou mít několik výslovnostních variant, ve slovníku by měly být uvedeny všechny možné. Naopak, některá různá slova mohou mít stejný přepis.
- Návrh struktury akustického modelu - HMM. Tedy vzorový HMM, kterým pak budou modelovány všechny monofony/trifony. Vzor HMM definuje počet stavů modelu, jeho dimenzi, typ modelu výstupní pravděpodobnosti a možné přechody mezi jednotlivými stavy.

- Pro tvorbu trifonového modelu je ještě třeba připravit tzv. *fonetický rozhodovací strom*. Ten obsahuje pravidla pro shlukování, podle kterých je možné určit akusticky podobné trifony. V případě, že bychom použili metodu shlukování trifonů řízenou pouze daty, není tento rozhodovací strom potřeba.

4.2 Tvorba monofonového jednosložkového modelu

Monofonový jednosložkový model je základní model, ze kterého se vychází při tvorbě trifonového modelu a může být vytvořen dvěma způsoby. Buď máme k dispozici již nějaký obdobný model (například z jiného korpusu) - startovací model. Startovací model musel být pořízen z dat se stejnou vzorkovací frekvencí a na data musela být aplikována stejná metoda zpracování signálu. Dále musí mít i totožnou fonetickou abecedu. Pak stačí použít několik trénovacích iterací k adaptaci startovacího modelu na nová data.

Pokud žádný vhodný startovací model není k dispozici je třeba celý model natrénovat od začátku (tzv. *flat start*). Nejprve se spočtou globální statistiky trénovacích dat (vektor středních hodnot, kovarianční matice). Těmito celkovými odhady parametrů se iniciují všechny stavy HMM všech fonémů. Pravděpodobnosti přechodu jsou nastaveny u všech HMM rovněž stejně jako apriorní přibližný odhad. Takovýmto způsoben je získán startovací model. V tomto případě by počet trénovacích iterací měl být větší než u předchozího případu, jelikož se startovací model od žádaného natrénovaného modelu velmi liší.

4.3 Tvorba trifonového modelu

Trifonový model bere v úvahu okolní kontext každého fonému, tedy v první fázi je třeba vygenerovat model se všemi trifony, které přicházejí v úvahu. Tedy všechny možné trojice z fonetické abecedy. Jelikož při běžných velikostech fonetických abeced, je výsledný nakombinovaný model tak veliký, že ho není možné dobře natrénovat s dostupných dat (některé kombinace fonémů se vyskytují velice zřídka nebo nemusí být v trénovacích datech vyslovena vůbec). Proto se hned v další fázi tvorby trifonového modelu přistupuje ke shlukování. Výsledkem shlukovacího algoritmu je seznam trifonů, které vzhledem k nedostatku dat a akustické podobnosti nemají vlastní model, ale jejich model je společný s jiným trifonem. Dále je vytvořena pro všechny trifony od jednoho fonému množina společných - svázaných stavů. Všechny tyto trifony pak místo vlastního modelu výstupní

pravděpodobnosti jednotlivých stavů mají odkaz na některý z těchto společných svázaných stavů patřících tomuto fonému.

Poslední fází při trénování trifonového modelu je přidávání složek do modelu výstupní pravděpodobnosti jednotlivých stavů. Jelikož jednosložkové modely zřídkakdy dokáží popsat vícerozměrné rozložení dat náležící tomu kterému stavu, je třeba zvýšit počet složek modelu na takové množství, aby trénovací data byla dobře modelována, ale zároveň aby model dobře zobecňoval na budoucí data testovací/provozní. Tedy při návrhu trifonového modelu je třeba zvážit, kolik svázaných stavů by měl finální model mít a kolika složkami budou tyto stavy modelovány. To ovlivňuje celkový počet odhadovaných parametrů a s přihlédnutím k množství trénovacích dat i robustnost odhadu těchto parametrů. Konkrétní nastavení je zpravidla určeno na základě zkušeností a jemněji doladěno experimentálně. Přidávání složek zpravidla probíhá postupně, vždy se přidá jedna složka do všech stavů modelu. Zde se většinou dělí složka s největší vahou na dvě nové. Poté následuje několik reestimací nových parametrů. Po ustálení odhadovaných hodnot parametrů se přidá další složka a proces dále pokračuje stejným způsobem dokud není dosaženo cílového počtu složek, konvergence odhadovaných parametrů i kriteria trénování.

4.4 Odhad parametrů

V předchozí části byl postup trénování popsán velice stručně. Z hlediska výkladu některých diskriminativních metod trénování by bylo vhodné popsat podrobněji proces odhadu nových parametrů modelu - jednu reestimeci [1]. Pro odhad nových parametrů máme k dispozici počáteční model, trénovací promluvy ve formě posloupnosti příznakových vektorů - vektorů pozorování, přepis, ze kterého je možné pomocí slovníku a dalších pomocných záznamů vzniklých během trénování a shlukování trifonů vygenerovat posloupnost stavů, které odpovídají trénovacím promluvám. Nejprve je třeba spočítat podmíněnou pravděpodobnost $P(\boldsymbol{O}|\lambda)$, kde λ je (skrytým Markovovým) modelem trénovací promluvy, která má formu posloupnosti vektorů pozorování $\boldsymbol{O} = \{\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_T\}$. Tuto pravděpodobnost je pak možné vyjádřit i pro každou složku jednotlivých stavů modelu λ. Ve skutečnosti ovšem známe pouze pozorovanou posloupnost $\boldsymbol{O}$ a základní posloupnost stavů $S = s(0), s(1), \ldots, s(T+1)$ je skrytá. Podmíněná pravděpodobnost $P(\boldsymbol{O}|\lambda)$ proto musí být vypočtena sumací přes všechny možné posloupnosti stavů S. Ze vstupního modelu známe nebo jsme schopni vypočítat přechodové a výstupní

pravděpodobnosti a_{ij} a $b_j(.)$, pak v obecném případě platí

$$\begin{aligned} P(\boldsymbol{O}, S|\lambda) = \textstyle\sum_S P(\boldsymbol{O}, S|\lambda) = \textstyle\sum_S P(\boldsymbol{O}|S,\lambda)P(S|\lambda) = \\ = \textstyle\sum_S a_{s(0)s(1)} b_{s(1)}(o_1) a_{s(1)s(2)} b_{s(2)}(o_2) \ldots b_{s(T)}(o_T) a_{s(T)s(T+1)} = \\ = \textstyle\sum_S a_{s(0)s(1)} \prod_{t=1}^{T} b_{s(t)}(\boldsymbol{o}_t) a_{s(t)s(t+1)}, \end{aligned} \tag{13}$$

kde $s(0)$ je chápáno jako vstupní neemitující stav modelu a $s(T+1)$ jako výstupní neemitující stav modelu. Přímý výpočet podle vztahu (13) je z hlediska výpočetní náročnosti prakticky neproveditelný, proto byl pro určení $P(\boldsymbol{O}|\lambda)$ navržen mnohem efektivnější způsob výpočtu, tzv. *forward-backward algoritmus*. Jde o iterační proceduru, kterou lze řešit výpočtem odpředu nebo odzadu vzhledem k pozorované posloupnosti.

4.4.1 Forward-backward algoritmus

Při výpočtu odpředu (angl. forward) definujeme sdruženou pravděpodobnost $\alpha_j(t)$ jako pravděpodobnost pozorování prvních t řečových vektorů $\{\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_t\}$ a jevu, že proces se nachází v čase t ve stavu s_j, a to za podmínky daného modelu λ

$$\alpha_j(t) = P(\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_t, s(t) = s_j|\lambda). \tag{14}$$

Hodnoty $\alpha_j(t)$ lze počítat rekurzivně. Budeme-li stále předpokládat, že první a poslední stav modelu jsou neemitující, je postup následující:

1. Inicializace

$$\begin{aligned} \alpha_1(1) &= 1 \\ \alpha_j(1) &= a_{1j} b_j(\boldsymbol{o}_1) \qquad \text{pro } 1 < j < N. \end{aligned} \tag{15}$$

2. Rekurze pro $t = 2, 3, \ldots, T$

$$\alpha_j(t) = \left[\sum_{i=2}^{N-1} \alpha_i(t-1) a_{ij} \right] b_j(\boldsymbol{o}_t) \quad \text{pro } 1 < j < N. \tag{16}$$

3. Výsledná pravděpodobnost

$$P(\boldsymbol{O}|\lambda) = \sum_{i=2}^{N-1} \alpha_i(T) a_{iN}. \tag{17}$$

Při výpočtu odzadu (angl. backward) definujeme pravděpodobnost $\beta_j(t)$ jako podmíněnou pravděpodobnost pozorování posloupnosti posledních $T-t$ řečových vektorů $\{\boldsymbol{o}_{t+1}, \boldsymbol{o}_{t+2}, \ldots, \boldsymbol{o}_T\}$ za podmínky, že model λ je v čase t ve stavu s_j

$$\beta_j(t) = P(\boldsymbol{o}_{t+1}, \boldsymbol{o}_{t+2}, \ldots, \boldsymbol{o}_T | s(t) = s_j, \lambda). \tag{18}$$

Hodnoty $\beta_j(t)$ lze opět počítat rekurzivně:

1. Inicializace
$$\beta_j(T) = a_{iN} \qquad \text{pro } 1 < i < N. \tag{19}$$

2. Rekurze pro $t = T-1, \ldots, 1$
$$\beta_j(t) = \sum_{j=2}^{N-1} a_{ij} b_j(\boldsymbol{o}_{t+1}) \beta_j(t+1) \quad \text{pro } 1 < i < N. \tag{20}$$

3. Výsledná pravděpodobnost
$$P(\boldsymbol{O}|\lambda) = \sum_{j=2}^{N-1} a_{1j} b_j(\boldsymbol{o}_1) \beta_j(1). \tag{21}$$

Lze snadno ukázat, že výsledná pravděpodobnost $P(\boldsymbol{O}|\lambda)$ může být též vyčíslena využitím proměnných $\alpha_i(t)$ a $\beta_i(t)$

$$P(\boldsymbol{O}|\lambda) = \sum_{j=2}^{N-1} \alpha_i(t) \beta_i(t) \tag{22}$$

pro $1 \leq t \leq T$. Pro úplnost ještě uveďme, že přímý výpočet podle algoritmu forward-backward vede obvykle k numerickým problémům, a proto jsou při rekurzivním výpočtu užívány buď normalizační koeficienty, nebo logaritmus pravděpodobnosti, které významně zvyšují numerickou stabilitu výpočtu.

4.4.2 Baumův-Welchův algoritmus

Jelikož pro odhad parametrů modelu v podstatě neexistuje explicitní řešení, byl vyvinut numerický *Baumův-Welchův algoritmus*. Tento algoritmus je iterativní a jedná se o speciální případ tzv. EM (Expectation-Maximization) algoritmu a vychází z *kritéria maximální věrohodnosti* (angl. Maximum Likelihood

- ML). Z odvození algoritmu [1] plyne, že pro každý nový odhad $\bar{\lambda}$ parametrů modelu platí $P(\boldsymbol{O}|\bar{\lambda}) > P(\boldsymbol{O}|\lambda)$, s výjimkou stavu, kdy bylo dosaženo optimálního nastavení parametrů, tj. pro $P(\boldsymbol{O}|\bar{\lambda}) = P(\boldsymbol{O}|\lambda)$. Poznamenejme, že Baumovým-Welchovým algoritmem lze nalézt parametry modelu, které zabezpečí dosažení pouze lokálního maxima funkce $P(\boldsymbol{O}|\lambda)$, přičemž toto lokální maximum závisí na volbě počátečních podmínek, tj. apriorní volbě hodnot parametrů modelu či předchozím průběhu trénování. Počáteční podmínky (počáteční model) jsou definovány jako $\lambda \equiv \{a_{ij}, c_{jm}, \boldsymbol{\mu}_{jm}, \boldsymbol{C}_{jm}\}$, kde a_{ij} je pravděpodobnost přechodu, $\boldsymbol{\mu}_{jm}$, $\boldsymbol{C}_{jm}$ a c_{jm} jsou postupně: vektor středních hodnot, kovarianční matice a váha směsi normálního rozložení pro stav j a složku m, kde $1 \leq i, j \leq N$ a $1 \leq m \leq M$, N je počet stavů HMM a M je počet směsí normálních rozložení, kterými je modelována výstupní pravděpodobnost těchto stavů.

Nyní si uvedeme vztahy pro odhad nových parametrů modelu - jednu reestimaci (iteraci). Parametry modelu obvykle odhadujeme na základě souboru E známých trénovacích promluv $\{\boldsymbol{O^e}\}_{e=1}^{E}$. Maximum věrohodnostní funkce, kterého se snažíme trénováním dosáhnout, pak bude mít pro tento případ tvar

$$\hat{\lambda} = \arg\max_{\lambda} \sum_{e=1}^{E} \log P(\boldsymbol{O}^e|\lambda). \tag{23}$$

Nejprve se pro každý příklad e spočítají algoritmem forward-backward hodnoty $P(\boldsymbol{O}^e|\lambda)$ a $\gamma_j^e(t)$ pro $t = 1, \ldots, T$

$$P(\boldsymbol{O}^e|\lambda) = \sum_{j=2}^{N-1} \alpha_j^e(t)\beta_j^e(t), \tag{24}$$

$$\gamma_j^e(t) = \frac{\alpha_j^e(t)\beta_j^e(t)}{P(\boldsymbol{O}^e|\lambda)} \tag{25}$$

a pro všechna $j = 2, \ldots, N-1$ a $m = 1, \ldots, M$ hodnoty $\gamma_{jm}^e(t)$

$$\gamma_{jm}^e(t) = \begin{cases} \frac{1}{P(\boldsymbol{O}^e|\lambda)} a_{1j} c_{jm} b_{jm}(\boldsymbol{o}_t^e)\beta_j^e(t) & \text{pro} \quad t = 1 \\ \frac{1}{P(\boldsymbol{O}^e|\lambda)} \sum_{i=2}^{N-1} \alpha_i^e(t-1) a_{ij} c_{jm} b_{jm}(\boldsymbol{o}_t^e)\beta_j^e(t) & \text{pro} \quad t \geq 2 \end{cases} \tag{26}$$

Vztahy pro určení nových hodnot parametrů mají formu vážených průměrů a jsou určeny následujícími reestimačními vztahy:

Pravděpodobnosti přechodů

$$\bar{a}_{1j} = \frac{1}{E} \sum_{e=1}^{E} \frac{1}{P(\boldsymbol{O}^e|\lambda)} \alpha_j^e(1)\beta_j^e(1) \qquad \text{pro} \quad 1 < j < N, \tag{27}$$

$$\bar{a}_{ij} = \frac{\sum_{e=1}^{E} \frac{1}{P(\boldsymbol{O}^e|\lambda)} \sum_{t=1}^{T_e-1} \alpha_j^e(t) a_{ij} b_j(\boldsymbol{o}_{t+1}^e) \beta_j^e(t+1)}{\sum_{e=1}^{E} \frac{1}{P(\boldsymbol{O}^e|\lambda)} \sum_{t=1}^{T_e} \alpha_i^e(t) \beta_i^e(t)} \qquad \text{pro} \quad 1 < i, j < N, \quad (28)$$

$$\bar{a}_{ij} = \frac{\sum_{e=1}^{E} \frac{1}{P(\boldsymbol{O}^e|\lambda)} \alpha_i^e(T_e) \beta_i^e(T_e)}{\sum_{e=1}^{E} \frac{1}{P(\boldsymbol{O}^e|\lambda)} \sum_{t=1}^{T_e} \alpha_i^e(t) \beta_i^e(t)} \qquad \text{pro} \quad 1 < i < N, \quad (29)$$

Parametry hustotních funkcí výstupních pravděpodobností se vyčíslují pro $1 < j < N$ a $1 \leq m \leq M$

$$\bar{c}_{jm} = \frac{\sum_{e=1}^{E} \sum_{t=1}^{T_e} \gamma_{jm}^e(t)}{\sum_{e=1}^{E} \sum_{t=1}^{T_e} \gamma_j^e(t)}, \quad (30)$$

$$\bar{\boldsymbol{\mu}}_{jm} = \frac{\sum_{e=1}^{E} \sum_{t=1}^{T_e} \gamma_{jm}^e(t) \boldsymbol{o}_t^e}{\sum_{e=1}^{E} \sum_{t=1}^{T_e} \gamma_{jm}^e(t)}, \quad (31)$$

$$\bar{\boldsymbol{C}}_{jm} = \frac{\sum_{e=1}^{E} \sum_{t=1}^{T_e} \gamma_{jm}^e(t) (\boldsymbol{o}_t^e - \bar{\boldsymbol{\mu}}_{jm})(\boldsymbol{o}_t^e - \bar{\boldsymbol{\mu}}_{jm})^T}{\sum_{e=1}^{E} \sum_{t=1}^{T_e} \gamma_{jm}^e(t)} \quad (32)$$

5 Diskriminativní trénování akustických modelů

I v současné době je trénování akustických modelů založené na kriteriu maximální věrohodnosti (angl. Maximum Likelihood - ML) velice populární. Využití Baumova-Welchova algoritmu, který zaručuje konvergenci k lokálnímu maximu kriteria, poskytuje poměrně přesný model a to nepříliš výpočetně náročným způsobem. ML odhad navíc poskytuje tu výhodu, že pokud jsou dodrženy nezbytné předpoklady, je tento odhad (sub)optimální (vzhledem k počátečním podmínkám) a tedy není důvod trénovat takový model jiným způsobem [25]. Mezi tyto předpoklady patří: Vektory pozorování jsou skutečně generovány HMM, kde modelovaná hustotní funkce výstupních pravděpodobnosti odpovídá reálné, vektory pozorování jsou na sobě navzájem nezávislé, množství trénovacích dat je neomezené (nekonečné). Bohužel ani jeden z těchto předpokladů není splněn při modelování reálné řeči pomocí HMM.

Vzhledem k tomu, že nejsou splněny některé z předpokladů pro ML odhad, nezaručuje pak tento odhad optimální parametry modelu. Pokud tedy ML kriterium není pro tento případ optimální, mohou existovat kritéria, která budou vést k odhadům parametrů modelu takovým, které budou k optimálnímu řešení blíže a výsledný systém rozpoznávání řeči bude dosahovat lepších výsledků. Jedním z takovýchto řešení je použití metod založených na tzv. diskriminativních kriteriích.

V nejobecnější rovině lze popsat ML odhad parametrů jako trénování z pozitivních vzorů. Tedy parametry jednoho konkrétního stavu HMM jsou trénovány pouze z dat náležící tomuto stavu. Tomuto přístupu ze základních algoritmů pro klasifikaci odpovídá Bayesovský klasifikátor. Oproti tomu diskriminativní metody využívají k trénování parametrů jak pozitivní, tak negativní příklady. Tedy jeden stav HMM je trénován nejen z dat tohoto stavu, ale i s přihlédnutím k datům ostatních stavů. Z tohoto základního rozdílu, lze snadno dovodit, že diskriminativní metody trénování budou principiálně komplikovanější a výpočetně náročnější. K diskriminativním metodám patří též většina standardních klasifikačních technik jako například: umělé neuronové sítě, Support Vector Machines (SVM) či lineární klasifikátory.

Dalším z problematických faktorů při trénování akustických modelů je obecnost či robustnost modelu. Tím je myšlena schopnost zobecňování modelů vytvořených z trénovacích dat na data testovací (dále v textu jako "schopnost zobecňovat na testovací data"). Většina trénovacích kriterií bere v úvahu pouze trénovací data a tedy jejich maximalizace nemusí dostatečně dobře odpovídat maximalizaci úspěšnosti rozpoznávání na testovacích datech. U ML odhadu je třeba stanovit počet stavů a počet složek modelu - tedy celkový počet odhadovaných parametrů

úměrně k dostupným trénovacím datům tak, aby nedocházelo k tzv. přetrénování - tedy k tomu, že výsledný model již nedokáže zobecňovat na testovací data. Ale i v případě vhodného počtu parametrů může během trénovaní docházet k problémům. Například odhad variancí podle vzorce (32) může u některých složek konvergovat k nule, to sice vede k prudkému nárůstu kriteriální funkce, kterou chceme maximalizovat (konverguje k nekonečnu), ovšem takový odhad není žádoucí, způsobuje numerické problémy při výpočtech a výsledný model je takřka nepoužitelný. Tento problém lze řešit například stanovením minimální variance, pod kterou nový odhad nesmí klesnout. Podobné problémy, kdy slepá maximalizace (minimalizace) kritéria nevede ke vhodnému řešení, se v komplexních úlohách, mezi které trénování akustických modelů patří, vyskytuje poměrně často, diskriminativní metody nejsou výjimkou.

5.1 Maximalizace vzájemné informace

Jedním z nejčastěji používaným, nebo alespoň diskutovaným, kritériem je kriterium *Maximalní vzájemné informace* (angl. Maximal Mutual Information - MMI) [26]. Toto kritérium vyjadřuje snahu maximalizovat pravděpodobnost správné hypotézy oproti všem ostatním. Přesněji maximalizuje aposteriorní pravděpodobnost správné posloupnosti slov $P(\boldsymbol{O}^e|W_R^e)$ rozpoznanou dekódovacím algoritmem oproti všem možným posloupnostem slov pro všechny trénovací promluvy

$$F_{MMI}(\lambda) = \sum_{e=1}^{E} \log \frac{P_\lambda(\boldsymbol{O}^e|W_R^e)P(W_R^e)}{\sum_{W\in\hat{\mathcal{W}}} P_\lambda(\boldsymbol{O}^e|W)P(W)}, \tag{33}$$

kde W_R^e je referenční posloupnost slov promluvy e, jmenovatel pak vyjadřuje součet pravděpodobností všech možných posloupností slov $\hat{\mathcal{W}}$ včetně referenční.

Pro maximalizaci MMI kriteria (33) tedy musí být čitatel maximalizován, přičemž jmenovatel minimalizován. Část v čitateli je identická s ML kriteriem, tedy stejně jako v ML případě, je žádoucí maximalizovat pravděpodobnost všech vektorů pozorování odpovídajících konkrétním stavům HMM vzhledem k referenční transkripci. Rozdílem je člen ve jmenovateli, který je třeba minimalizovat snížením pravděpodobností pro ostatní možné posloupnosti slov.

Stejně jako v ML případě se u MMI setkáváme s několika problémy:

- Je obtížné maximalizovat toto kritérium, explicitní vztah pro maximalizaci neexistuje a i nepřímé, iterativní postupy jsou obtížné a jen některé obecně zajišťují konvergenci.

- Maximalizace je extrémně výpočetně náročná.
- MMI přístup má obecně horší schopnost zobecňovat na data testovací než ML přístup. Je potřeba většího množství dat vzhledem k počtu odhadovaných parametrů.
- Numerická stabilita algoritmu je ještě problematičtější než v ML případě.
- V případě jednodušších úloh je možné snadno během několika iterací docílit 100% úspěšnosti na trénovacích datech. Poté již není možno nijak dále touto metodou model optimalizovat.

Vzhledem k výše uvedeným skutečnostem, se jeví jako praktické, natrénovat celý model klasickým postupem popsaným v kapitole 4 podle ML kriteria a až výsledný model podrobit několika iteracím algoritmu pracujícím s kriteriem MMI.

MMI diskriminativní kriterium nemůže být optimalizováno použitím standardního Baumova-Welchova algoritmu. Jediné známé metody, které konvergují k maximu MMI kriteria jsou tzv. *steepest gradient decsent* a rozšířený Baumův-Welchův algoritmus [27]. Vzhledem k vysoké dimenzi prostoru vektoru pozorování, gradientní přístup může vyžadovat obrovské množství iterací pro získání optimálního řešení [28]. Tedy rozšířený Baumův-Welchův algoritmus je nejčastější algoritmus používaný pro odhad parametrů modelu v MMI diskriminativním trénování. Detailně bude rozšířený Baumův-Welchův algoritmus popsán níže.

Výpočetní náročnost MMI odhadu parametrů pramení z vyčíslení členu ve jmenovateli kriteria (33). Jmenovatel vyžaduje součet přes všechny možné posloupnosti slov s jazykového modelu. Tedy je třeba vyčíslit pravděpodobnosti pro každou možnou posloupnost slov v každé iteraci trénovacího algoritmu. Toto může být provedeno pro úlohu s malým slovníkem, ale s narůstajícím množstvím slov ve slovníku se stává tento postup prakticky neproveditelný. Jak bude popsáno dále, tento postup lze aproximovat a zjednodušit takovým způsobem, že ho lze využít i v úlohách s velkým slovníkem.

5.1.1 Odhad nových parametrů modelu

Jak bylo výše uvedeno, pro nový MMI odhad parametrů HMM nelze použít klasický Baumův-Welchův algoritmus. V [27] a [30] byl původní algoritmus postupně rozšířen pro použití s MMI kriteriem a HMM se spojitě modelovanými výstupními pravděpodobnostmi pomocí směsi normálních rozložení. Vztahy pro nové odhady

středních hodnot a diagonálních kovariančních matic jsou následující:

$$\bar{\boldsymbol{\mu}}_{jm} = \frac{\left\{\boldsymbol{\Theta}^{num}_{jm}(\boldsymbol{O}) - \boldsymbol{\Theta}^{den}_{jm}(\boldsymbol{O})\right\} + D\boldsymbol{\mu}_{jm}}{\left\{\gamma^{num}_{jm} - \gamma^{den}_{jm}\right\} + D} \tag{34}$$

$$\bar{\boldsymbol{C}}_{jm} = \frac{\left\{\boldsymbol{\Theta}^{num}_{jm}(\boldsymbol{O}^2) - \boldsymbol{\Theta}^{den}_{jm}(\boldsymbol{O}^2)\right\} + D(\boldsymbol{C}_{jm} + \boldsymbol{\mu}^2_{jm})}{\left\{\gamma^{num}_{jm} - \gamma^{den}_{jm}\right\} + D} - \bar{\boldsymbol{\mu}}^2_{jm}, \tag{35}$$

kde $\boldsymbol{\Theta}_{jm}(.)$ je suma všech vektorů pozorování nebo suma jejich druhých mocnin vážená aposteriorní pravděpodobností, že v čase t je daný vektor pozorování generovaný stavem j a složkou modelu m:

$$\boldsymbol{\Theta}_{jm}(\boldsymbol{O}) = \sum_{e=1}^{E}\sum_{t=1}^{T_e} \boldsymbol{o}^e(t)\gamma^e_{jm}(t) \tag{36}$$

$$\boldsymbol{\Theta}_{jm}(\boldsymbol{O}^2) = \sum_{e=1}^{E}\sum_{t=1}^{T_e} [\boldsymbol{o}^e(t)]^2\gamma^e_{jm}(t), \tag{37}$$

kde $\boldsymbol{O}$ je posloupnost trénovacích promluv $\left\{\boldsymbol{O}^1, \boldsymbol{O}^2, \ldots, \boldsymbol{O}^E\right\}$, kde každá promluva $\boldsymbol{O}^e$ je posloupností vektorů pozorování $\{\boldsymbol{o}^e(1), \boldsymbol{o}^e_2, \ldots, \boldsymbol{o}^e(T_e)\}$ a $\gamma^e_{jm}(t)$ je aposteriorní pravděpodobnost, že $\boldsymbol{o}^e(t)$ v čase t je generován stavem j a složkou m HMM. D ve vzorcích (36) a (37) vyjadřuje stabilizační konstantu, jejíž vliv a vhodný postup výpočtu bude diskutován níže.

Tedy v (36) a (37) je γ_{jm} suma pravděpodobnosti, že se Markovův proces nachází ve stavu j a složce m. Horní indexy *num* a *den* označují, zda je suma pravděpodobnosti počítána přes model čitatele (*num*) či model jmenovatele *den* kriteria MMI (33).

Poznamenejme ještě, že rovnice (37) slouží pouze pro diagonální kovarianční matice. Nicméně je možné ji rozšířit i na plné kovarianční matice, jak je ukázáno v [32]. Pozitivní vliv nahrazení diagonálních matic plnými je však diskutabilní. Vzhledem k velikému nárůstu odhadovaných parametrů, radikálně klesá robustnost jejich odhadů, zároveň vznikají i četné problémy numerického charakteru při inverzi těchto matic. Diskriminativní techniky jsou navíc na poměr odhadovaných parametrů a dostupných trénovacích dat velmi citlivé.

Stejně jako střední hodnoty a kovarianční matice je možné odhadnout diskriminativně i váhy složek c_{jm} či přechodové pravděpodobnosti a_{ij}. Nicméně modifikace těchto odhadů nepřináší významné zlepšení úspěšnosti rozpoznávání, některé metody odhadu mohou naopak zanášet další nestabilitu do iterativního procesu a

ten nemusí konvergovat [31]. Původně odvozený vztah pro výpočet nového odhadu vah je [26]

$$\bar{c}_{jm} = \frac{c_{jm}\left\{\frac{\delta F(\lambda)}{\delta c_{jm}} + G\right\}}{\sum_{\hat{m}} c_{j\hat{m}}\left\{\frac{\delta F(\lambda)}{\delta c_{j\hat{m}}} + G\right\}}, \tag{38}$$

kde $F(\lambda)$ je hodnota kriteria a derivace $\frac{\delta F(\lambda)}{\delta c_{jm}}$ je dána

$$\frac{\delta F(\lambda)}{\delta c_{jm}} = \frac{1}{c_{jm}}(\lambda_{jm}^{num} - \lambda_{jm}^{den}). \tag{39}$$

Avšak hodnota derivace $\frac{\delta F(\lambda)}{\delta c_{jm}}$ může být dosti veliká pro velmi malé c_{jm}, tedy může snadno převážit stabilizační konstantu G, pokud není dostatečně velká. V praxi je konstanta G nastavena tak, aby všechny váhy zůstali kladné. Potom se však může stát, že některou ze složek vynucená velká hodnota stabilizační konstanty nebude vhodná pro jiné stavy. Tímto problémem se například zabývala práce [33], kde byla rovněž navržena robustnější aproximace problematické derivace:

$$\frac{\delta F(\lambda)}{\delta c_{jm}} \cong \frac{\lambda_{jm}^{num}}{\sum_{\hat{m}} \lambda_{j\hat{m}}^{num}} - \frac{\lambda_{jm}^{den}}{\sum_{\hat{m}} \lambda_{j\hat{m}}^{den}}. \tag{40}$$

Tato aproximace či zjednodušení zjevně není korektní, nicméně je ověřeno v praxi jako funkční ([28, 31, 32, 59]). V této práci je navrženo jiné řešení, které je teoreticky správnější, například nové odhady nemohou nabývat nesmyslných hodnot (například záporné váhy). Do detailu je popsáno v následují kapitole.

Vztahy pro diskriminativní odhady přechodových pravděpodobností lze nalézt například v [34], kde je ovšem také diskutován příspěvek modifikace pravděpodobností přechodu - pro HMM se shlukovanými trifony a svázanými stavy je příspěvek zanedbatelný, rovněž modifikace vah nepřináší žádné radikální zlepšení úspěšnosti rozpoznávání.

Nastavení stabilizační konstanty D je velmi diskutovaný problém. Její vhodné nastavení je kompromisem mezi stabilitou iteračního algoritmu a rychlostí trénování. Příliš malá stabilizace umožňuje rychlé trénování, ovšem v následujících iteracích může být model degradován, případně může docházet k numerickým nestabilitám při výpočtu, jako jsou záporné odhady variancí a podobně. Příliš velká stabilizace pak zbytečně prodlužuje dobu již tak velmi výpočetně náročného trénování. Původně navržený postup [30] pro určení konstanty D spočíval v určení minimální hodnoty této konstanty, která pak zaručuje kladné odhady všech variancí. Skutečně použitá konstanta D pak byla jejím násobkem:

$$D = D_{fact} D_{min}, \tag{41}$$

kde D_{fact} byl většinou roven 2.

V praxi se pak ukazuje, že globální konstanta D není vhodná, jelikož různé stavy modelu je třeba stabilizovat nezávisle. Vliv stanovení konstanty D nezávisle pro každý stav i pro každou složku normálního rozložení je popsán v [29] a [34]. V [34] je pak navrženo určovat konstantu nezávisle pro každou složku jako maximum z dvojnásobku hodnoty, která zaručí pozitivní variance a γ_{jm}^{den}. V rámci této disertační práce byl navržen i vlastní postup, ovšem po mnoha úpravách a modifikacích se velmi blíží výše uvedenému. Popsán je v následující kapitole.

5.1.2 Výpočet statistik

Pro výpočet nových odhadů středních hodnot a kovariančních matic je třeba ve vztazích (36) a (37) určit $\gamma_{jm}(t)$. A to jak pro čitatel kriteria $\gamma_{jm}^{num}(t)$, tak pro jmenovatel $\gamma_{jm}^{den}(t)$. V případě ML odhadu určujeme $\gamma_{jm}(t)$ pomocí *forward-backward* algoritmu. V MMI případě použijeme tento algoritmus rovněž, pro výpočet $\gamma_{jm}^{num}(t)$, jelikož čitatel je prakticky identický s ML odhadem. Určení jmenovatele $\gamma_{jm}^{den}(t)$ je obtížnější. Pro úlohy s velmi malým slovníkem by bylo ještě realizovatelné použít stejný *forward-backward* algoritmus jen místo referenčního přepisu trénovací promluvy postupně nasčítávat pravděpodobnosti pro všechny možné posloupnosti slov. Bohužel i s velmi malým slovníkem by to bylo výpočetně velice náročné a pro jen o něco vetší slovníky prakticky nerealizovatelné.

Jednou z možností jak tento problém překlenout je využití dekódovacího algoritmu. Můžeme tak rozpoznat například několik hypotéz [40] a provést *forward-backward* algoritmus jen na těchto nejpravděpodobnějších hypotézách. Obrovské množtví posloupností slov, která jsme nevyhodnocovali bude mít mnohem menší pravděpodobnost, takže je možné takto dosáhnout velmi efektivně přibližně stejného výsledku. Pro ještě vetší efektivnost algoritmu, můžeme tyto nejlepší hypotézy určit pouze jednou a použít je i v dalších MMI iteracích. Avšak fixní počet hypotéz nemůže zachytit dostatečné množství možností a variant, zejména u úloh s velkým slovníkem. Lepší možností, místo hypotéz je vytvořit tzv. mřížky (angl. lattices), které dokáží úsporně uložit informace odpovídající velkému množství hypotéz. Většinou jsou používány mřížky slovní [28].

Slovní mřížka je uspořádaný graf, kde hrany tvoří jednotlivá slova ze slovníku a uzly jsou časově definované hranice mezi nimi. Tato mřížka je generována upraveným dekódovacím algoritmem. Velikost mřížky lze ovlivnit nastavením prořezávání (angl. pruning beam). Jelikož generovaní mřížky je velice výpočetně náročné, dělá se obvykle pouze jednou a pak je stejná mřížka používána pro

více iterací diskriminativního trénování. V dalších iteracích se jen přepočtou pravděpodobnosti, případně upraví hranice mezi jednotlivými slovy.

Tímto způsobem lze aplikovat MMI metodu i v poměrně rozsáhlých úlohách. Jednou z prvních úspěšných aplikací byl srovnávací test *NIST Hub5* v roce 2000 [39]. Trénovací data obsahovala celkem 256 hodin řeči, slovník měl 54 tisíc slov. Akustický model obsahoval 6165 stavů, každý byl modelován pomocí 16-ti složek. Po dvou iteracích MMI trénování byla snížena chyba rozpoznávání o 2.7% z $45,6\%$ na $42,9\%$. Další porovnání výsledků MMI lze nalést například v [37, 38, 42, 35, 34].

5.1.3 Frame-discriminativní modifikace

Jak bylo diskutováno výše, jeden z největších problémů trénování na základě MMI kriteria je špatná schopnost zobecňovat na testovací data. Jedno z řešení tohoto problému je zvýšení počtu konkurujících si stavů modelu. Při výpočtu jmenovatele MMI kriteria je to v každém čase pouze několik málo konkurujících si hypotéz - stavů modelu. Frame-diskriminativní (angl. Frame-Discriminative - FD) modifikace původního MMI kriteria, se zaměřuje pouze na akustickou část modelu pro výpočet jmenovatele a nechává konkurovat si všechny stavy akustického modelu navzájem [32, 35]. Tedy nepředpokládá, že v trénovacím textu se objeví dostatečné množství všech možných posloupností slov, které by vytvořily dostatečné množství konkurujících si stavů. Modifikované kriterium pro MMI-FD je

$$F_{FD}(\lambda) = \sum_{e=1}^{E} \log \frac{P_\lambda(\boldsymbol{O}^e|W_R^e)}{P_\lambda(\boldsymbol{O}^e|M^{gen})}, \tag{42}$$

kde M^{gen} je FD model jmenovatele, který je obvykle uvažován v každém čase jako součet pravděpodobností všech stavů akustického modelu [35]:

$$P_\lambda(\boldsymbol{O}^e|M^{gen}) = \prod_{t=1}^{T_e} \sum_{j \in M^{gen}} b_j(\boldsymbol{o}^e(t)) P(s_j|M^{gen}), \tag{43}$$

kde $\boldsymbol{o}^e(t)$ je vektor pozorování trénovací promluvy e v čase t, $b_j(\boldsymbol{o}^e(t))$ je výstupní pravděpodobnost j-tého stavu akustického modelu, $P(s_j|M^{gen})$ je apriorní pravděpodobnost j-tého stavu akustického modelu. Tuto apriorní pravděpodobnost, lze určit z předchozí iterace trénování, v některých případech však je lepší informaci o apriorní pravděpodobnosti do výpočtu nezahrnovat.

Tato modifikace tedy mění způsob výpočtu $\gamma_{jm}^{den}(t)$. Zbylý výpočet nových parametrů již zůstává beze změny. Tato modifikace má, kromě výše uvedených, i

další příznivé vlastnosti. Vzhledem k tomu, že pracuje pouze s akustickou částí modelu, nevyžaduje generování hypotéz ani mřížek. Jediné co je pro výpočet potřeba, je vyčíslení pravděpodobností všech stavů modelu v každém čase. To je oproti generování a zpracování mřížek či hypotéz časově mnohem méně náročné (5-15 krát) a to i v případě, že se mřížky či hypotézy generují pouze jednou. U mohutnějších akustických modelů pak i jen výpočet pravděpodobnosti všech stavů může být dosti zdlouhavý, proto byly navrženy některé aproximace, které se snaží celý proces urychlit [41, 35]. Tento tzv. *Roadmap* algoritmus využívá faktu, že z celkového velkého množství normálních distribucí, pouze velmi malé množství má pravděpodobnost větší než 0 a snaží se v první fázi rychlým výběrem určit ty, které má cenu vyhodnocovat a které je možné ignorovat. V [35] je ukázáno, že vhodným výběrem přibližně 4% normálních distribucí lze dosáhnout téměř identických výsledků. V této práci je v kapitole 7 naopak ukázáno, že využitím schopností dnešních počítačů, lze výpočet zrychlit ještě významněji bez ztráty přesnosti či hrubých aproximací.

5.1.4 Váha akustického modelu

Při výpočtu pravděpodobnosti posloupnosti slov se obvykle váží (násobí) logaritmy pravděpodobnosti z jazykového modelu kladným číslem - váhou LM. Toto zvýšení váhy jazykového modelu je nutné, jelikož akustický model generuje řádově nižší pravděpodobnosti. Důvodem jsou některé nesplněné úvodní předpoklady akustického modelu, který je jen aproximací modelu reálné produkce řeči. Jak je diskutováno v [38], je možné tuto váhu implementovat i v diskriminativním trénování. Ovšem zde je nutné použít váhu inverzní a aplikovat ji na model akustický. Zvýší se tím konkurence mezi stavy a rozšíří se množství hypotéz či stavů s generovaných mřížek, které se budou trénování účastnit a tedy se i zvýší schopnost výsledného modelu zobecňovat na testovací data. Upravený vztah pro MMI kriterium je pak následující:

$$F_{MMI}(\lambda) = \sum_{e=1}^{E} \log \frac{P_\lambda(\boldsymbol{O}^e|W_R^e)^\kappa P(W_R^e)}{\sum_{W\in\hat{\mathcal{W}}} P_\lambda(\boldsymbol{O}^e|W)^\kappa P(W)}, \tag{44}$$

kde κ je váha akustického modelu, obvykle určená jako $1/\kappa_{LM}$, kde κ_{LM} je váha jazykového modelu obvykle používaná pro danou úlohu pro rozpoznávání.

5.1.5 Využití slabšího jazykového modelu

Jazykový model použitý pro generování mřížek pro diskriminativní trénování může být odlišný od jazykového modelu používaného pro rozpoznávání. Například *unigram* může být použit pro generování trénovacích mřížek a *trigram* pro rozpoznávání. Použití slabšího modelu pro generování mřížek umožňuje vytvořit větší množství v jednom čase si konkurujících hypotéz a tím zvýšit schopnost výsledného modelu lépe zobecňovat na testovací data.

5.1.6 Hybridní kriterium

Spíše než zvyšování množství konkurujících si stavů, pro zvýšení schopnosti zobecňovat, je možné použít hybridní kriterium (v [38] jako *H-criterion*). Jedná se o interpolaci mezi kriteriem ML a MMI:

$$\alpha F_{MMI} + (1 - \alpha) F_{ML}. \tag{45}$$

Tento postup je podrobně diskutován v [38]. Tento postup zjevně vede k větší schopnosti zobecňovat, avšak určení optimální hodnoty α je obtížné a efekt tohoto postupu klesá s narůstajícím množstvím trénovacích dat. Nicméně tento postup brání možnosti přetrénování, tedy určení vhodného počtu MMI reestimací pak není tak kritické jako při optimalizaci čistého MMI kriteria.

5.1.7 I-smoothing

Výše zmíněné H-kriterium používá fixní poměr mezi ML a MMI kriteriem. Jiný přístup tzv. *I-smoothing* [36] používá rovněž interpolaci mezi ML a MMI kriteriem. Tato interpolace je pro každou složku nastavována individuálně v závislosti na množství trénovacích dat dostupných pro tuto složku. Jedná se o drobnou modifikaci standardního MMI přístupu, kde akumulovaný počet trénovacích dat čitatele γ_{jm}^{num} je zvýšen o τ, přičemž akumulátory pro výpočet středních hodnot $\boldsymbol{\Theta}_{jm}^{num}(\boldsymbol{O})$ a variancí $\boldsymbol{\Theta}_{jm}^{num}(\boldsymbol{O}^2)$, jsou navýšeny takovým způsobem, že je hodnoty těchto středních hodnot a variancí nezmění:

$$\acute{\gamma}_{jm}^{num} = \gamma_{jm}^{num} + \tau, \tag{46}$$

$$\acute{\boldsymbol{\Theta}}_{jm}^{num}(\boldsymbol{O}) = \boldsymbol{\Theta}_{jm}^{num}(\boldsymbol{O}) + \frac{\tau}{\gamma_{jm}^{num}} \boldsymbol{\Theta}_{jm}^{num}(\boldsymbol{O}), \tag{47}$$

$$\acute{\boldsymbol{\Theta}}_{jm}^{num}(\boldsymbol{O}^2) = \boldsymbol{\Theta}_{jm}^{num}(\boldsymbol{O}^2) + \frac{\tau}{\gamma_{jm}^{num}}\boldsymbol{\Theta}_{jm}^{num}(\boldsymbol{O}^2). \tag{48}$$

Tedy všechny akumulátory čitatele jsou vynásobeny faktorem $1 + \frac{\tau}{\gamma_{jm}^{num}}$. Pro další výpočet nových odhadů podle rovnic (34) a (35) jsou použity takto upravené akumulátory. Vhodná hodnota τ, odpovídá potřebnému počtu dat k robustnímu odhadu jedné složky modelu. Pro MMI trénování je vhodná hodnota τ od 50 do 150.

5.2 Minimalizace chyby klasifikace

Dalším velmi populárním kriteriem pro diskriminativní trénování je *Minimální chyba klasifikace* (angl. Minimum Classification Error - MCE) [63, 62, 61]. MCE kriterium přímo minimalizuje chybu v rozpoznané posloupnosti slov. Oproti tomu MMI kriterium maximalizuje pravděpodobnost správné posloupnosti slov oproti ostatním.

5.2.1 Diskriminační funkce

Trénovací promluvu e a k ní náležící posloupnost slov W^e reprezentujeme posloupností vektorů pozorování $\boldsymbol{O}^e = \{\boldsymbol{o}_1^e \boldsymbol{o}_2^e \dots \boldsymbol{o}_T^e\}$. Dále je možné vygenerovat nejpravděpodobnější segmentaci na posloupnost stavů HMM $S^e = \{s_1^e \dots s_t^e \dots s_T^e\}$ například pomocí Viterbiho algoritmu. Pak jsme schopni sestavit diskriminační funkci pro W^e [61]:

$$\begin{aligned} g^e(\boldsymbol{O}^e|\lambda) &= \log P(W^e) + \log P_\lambda(\boldsymbol{O}^e|S^e) \\ &= \log P(W^e) + \sum_{t=1}^{T_e} \log a_{s_{t-1}^e s_t^e} + \sum_{t=1}^{T_e} \log b_{s_t^e}(\boldsymbol{o}_t^e), \end{aligned} \tag{49}$$

kde $a_{s_{t-1}^e s_t^e}$ jsou pravděpodobnosti přechodu a $b_{s_t^e}(\boldsymbol{o}_t^e)$ je výstupní pravděpodobnost stavu s_t^e v čase t pro vektor pozorování $\boldsymbol{o}_t^e$. Diskriminační funkce je tedy velice podobná ML kriteriu či čitateli MMI kriteria.

Dále je třeba sestavit obdobnou funkci pro všechny ostatní - nesprávné posloupnosti slov. Všechny ostatní posloupnosti slov označme $\mathcal{W}^- = \mathcal{W} - W^e$, kde $\mathcal{W}$ je množina všech možných posloupností slov v dané úloze. Pak diskriminační funkce ostatních posloupností slov může být definována jako

$$g^{\mathcal{W}^-}(\boldsymbol{O}^e|\lambda) = \max_{W \in \mathcal{W}^-} g^W(\boldsymbol{O}^e|\lambda), \tag{50}$$

kde bereme v úvahu jen nejlepší konkurenční hypotézu a je to limitní případ obecnější formulace

$$g^{\mathcal{W}^-}(\boldsymbol{O}^e|\lambda) = \log\left[\frac{1}{|\mathcal{W}^-|}\sum_{W\in\mathcal{W}^-}\exp^{g^W(\boldsymbol{O}^e|\lambda)\psi}\right], \tag{51}$$

kde $|\mathcal{W}^-|$ je celkový počet nesprávných posloupností slov, parametr ψ řídí množství hypotéz, které budou brány v potaz. Malé hodnoty ψ mohou zvýšit počet těchto hypotéz a výsledný model bude mít větší schopnost zobecňovat na testovací data. Vážení parametrem ψ funguje obdobně jako vážení akustického modelu u kriteria MMI v kapitole 5.1.4. V praxi, pro vyhodnocení funkce (50) se používá buď N-nejlepších hypotéz nebo slovních mřížek [64].

5.2.2 Míra chyby klasifikace

Míra chyby klasifikace u kriteria MCE porovnává hodnoty diskriminačních funkcí pro správnou posloupnost slov W^e s hodnotou diskriminační funkce pro všechny ostatní posloupnosti slov $|\mathcal{W}^-|$:

$$d^e(\boldsymbol{O}^e|\lambda) = -g^e(\boldsymbol{O}^e|\lambda) + g^{\mathcal{W}^-}(\boldsymbol{O}^e|\lambda). \tag{52}$$

Pro výpočet lze využít jak rovnici (50) tak (51). Výsledné znaménko míry chyby klasifikace odpovídá správnosti nebo nesprávnosti nejpravděpodobnější rozpoznané hypotézy, přesně to však platí jen při výpočtu s využitím rovnice 50.

5.2.3 Ztrátová funkce

Nejtypičtější funkce používaná pro MCE je sigmoida:

$$\ell(d^e(\boldsymbol{O}^e|\lambda)) = \ell(d^e) = \frac{1}{1+e^{-\alpha d^e}}, \tag{53}$$

kde pro přehlednost $d^e = d^e(\boldsymbol{O}^e|\lambda)$. Tedy zjednodušeně, pokud míra chyby klasifikace je kladná, ztrátová funkce se bude blížit 1 a naopak, pro zápornou míru se bude hodnota ztrátové funkce blížit 0. Účinek ztrátové funkce závisí na její strmosti, kterou lze nastavit parametrem α. Kromě nejběžnější sigmoidy lze použít i jiné funkce, například sigmoidu s prahem [79], která pod určitou zápornou hodnotou - práh - má již ztrátovou funkci rovnou 0. Pro některé úlohy tato ztrátová funkce může usnadnit proces optimalizace, nicméně zavádí další parametr, který

musí být vhodně nastaven. Mezi další používané ztrátové funkce patří po částech lineární funkce [65, 66], ale lze použít i lineární variantu, kdy $\ell(d^e) = d^e$. Zde ovšem ztrátová funkce již není v obvyklém intervalu od 0 do 1.

5.2.4 Optimalizační metody

Pro minimalizaci MCE kriteria - součet ztrátových funkcí všech trénovacích promluv - je používána řada různých optimalizačních metod [67, 68, 69, 70, 71]. Zde budou popsány některé z nich:

Probabilistic Descent je velice jednoduchá, ale i velice efektivní metoda založená na metodě největšího spádu [72]. Základem je výpočet gradientu ztrátové funkce pro každý blok dat $\boldsymbol{O}_n$. Kdy tento blok dat, může tvořit jen jeden vektor pozorování, ale i větší blok dat - vhodný pro lepší paralelizaci výpočtu. Úprava parametrů modelu pak probíhá v opačném směru tohoto gradientu. Velikost modifikace parametrů je závislá na velikosti gradientu a je řízena koeficientem rychlosti učení ϵ_n:

$$\lambda^{(n+1)} = \lambda^{(n)} - \epsilon_n \nabla_\lambda \ell(d(\boldsymbol{O}_n|\lambda^{(n)})), \tag{54}$$

kde $\lambda^{(n)}$ označuje sadu parametrů modelu v iteraci n. Síla tohoto algoritmu je v tom, že využívá redundance v datech k urychlení konvergence [73] tím, že nové parametry modelu jsou modifikovány hned po zpracování každého bloku trénovacích dat - on-line. Velikost bloku je určitým kompromisem. Úpravou parametrů modelu po každém vektoru pozorování vede k nejrychlejší konvergenci - nejmenšímu počtu iterací. Větší bloky dat však lze rozdělit mezi více procesorů či počítačů, takže ačkoli algoritmus bude konvergovat ve větším počtu iterací, finální model může být k dispozici v kratším čase.

Quickprop [74] je jednoduchá dávkově orientovaná metoda druhého řádu motivovaná klasickou Newtonovou metodou optimalizace. Quickprop byl poprvé použit pro optimalizaci MCE kriteria v [75] a pro MMI kriterium v [32]. Může být použit v dávkovém režimu, tedy lze ho jednoduše paralelizovat na více počítačích. Nový odhad parametrů pomocí Newtonovy metody je

$$\lambda^{(n+1)} = \lambda^{(n)} - (\nabla^2 F(\lambda))^{-1} \nabla F(\lambda). \tag{55}$$

Obecně, pokud je Hessova matice pozitivně definitní a počáteční parametry modelu λ jsou dostatečně blízko optimálním hodnotám, Newtonova metoda konverguje velice rychle k lokálnímu minimu kriteriální funkce [76]. Bohužel ve většině případů

není zaručeno, že je Hessova matice pozitivně definitní a dobře podmíněná. Navíc, velikost Hessovy matice - čtvercová matice, kde strana čtverce je počet optimalizovaných parametrů akustického modelu (řádově desetitisíce až statisíce) - je tak obrovská, že je v praxi nerealizovatelná. V metodě Quickprop je z této matice realizována pouze diagonála a druhá parciální derivace parametru p_i je aproximována jako

$$\frac{\partial^2 F(\lambda^{(n)})}{\partial p_i^2} = \frac{\frac{\partial F(\lambda^{(n)})}{\partial p_i} - \frac{\partial F(\lambda^{(n-1)})}{\partial p_i}}{\Delta p_i^{(n-1)}}, \tag{56}$$

kde $\lambda^{(n)}$ je sada optimalizovaných parametrů modelu v iteraci n, p_i je i-ty parametr modelu λ a $\Delta p_i^{(n-1)}$ je velikost změny hodnoty parametru p_i oproti předchozí iteraci. Toto je velice hrubá aproximace, avšak cílem není co nejpřesněji aproximovat Hessovu matici, nýbrž posunout hodnoty parametrů blíže k optimálním hodnotám. Quickprop dále umožňuje urychlit konvergenci tím, že sleduje znaménko gradientu v po sobě jdoucích iteracích. Pokud je znaménko gradientu stále stejné, znamená to, že prvek na diagonále aproximované Hessovy matice má příliš malou hodnotu a je tedy rozumné ho zvýšit o konstantu učení ϵ:

$$\lambda^{(n+1)} = \lambda^{(n)} - \left[(\nabla^2 F(\lambda))^{-1} + \epsilon\right] \nabla F(\lambda). \tag{57}$$

Pro urychlení, ale i větší stabilizaci algoritmu, jsou možné další modifikace, jako například absolutní nebo relativní maximální změna parametru a další různé modifikace [76, 77]. I přesto je Quickprop stále velice snadno implementovatelný a v praxi efektivní algoritmus.

Rprop [78], což je zkratka z anglického *Resilient back-propagation*, je dávkový optimalizační algoritmus, známý z oblasti umělých neuronových sítí. Jeho základním principem je odstranit často nepřesné a aproximativní odhady velikosti parciálních derivací na úpravu parametrů. V této metodě je použito pouze znaménko z parciálních derivací a velikost posunu parametrů je řízena proměnnou $\Delta p_i^{(n)}$ pro n-tou iteraci následovně:

$$\Delta p_i^{(n)} = \begin{cases} -\Delta_i^{(n)} & : \text{ pokud } \frac{\partial F(\lambda^{(n)})}{\partial p_i} > 0 \\ +\Delta_i^{(n)} & : \text{ pokud } \frac{\partial F(\lambda^{(n)})}{\partial p_i} < 0 \\ 0 & : \text{ jinak} \end{cases} \tag{58}$$

Velikost posunu $\Delta_i^{(n)}$ je pro každý parametr jiná a je během algoritmu upravována následujícím způsobem:

$$\Delta_i^{(n)} = \begin{cases} \eta^+ \cdot \Delta_i^{(n-1)} & : \text{ pokud } \frac{\partial F(\lambda^{(n-1)})}{\partial p_i} \cdot \frac{\partial F(\lambda^{(n)})}{\partial p_i} > 0 \\ \eta^- \cdot \Delta_i^{(n-1)} & : \text{ pokud } \frac{\partial F(\lambda^{(n-1)})}{\partial p_i} \cdot \frac{\partial F(\lambda^{(n)})}{\partial p_i} < 0 \\ \Delta_i^{(n-1)} & x \text{ jinak} \end{cases}, \tag{59}$$

kde $0 < \eta^- < 1 < \eta^+$. Tedy pokud parciální derivace parametru p_i má stále stejné znaménko, zřejmě velikost posunu je nedostatečná a je třeba jí zvětšit. Naopak pokud se znaménka během posledních iterací střídají, hodnota parametru přeskakuje okolo hodnoty optimální a je třeba krok zjemnit - tedy velikost posunu zmenšit. Běžně používané hodnoty pro úpravu velikostí posunu jsou například $\eta^- = 0,5$ a $\eta^+ = 1,2$. Stejně jako u ostatních metod i zde jsou možné různé modifikace, některé z nich jsou popsány například v [78].

5.3 Minimalizace chyb ve slovech a ve fonémech

Metody založené na MMI a MCE kriteriích fungují velice dobře na úlohách s malým slovníkem, ovšem v úlohách s velkým slovníkem, kde je velmi veliký počet odhadovaných parametrů je jejich implementace problematická (zejména u MCE). Dále schopnost zobecňovat na testovací data je omezená, ačkoli byly vyvinuty různé modifikace, které se snaží tento nedostatek potlačit. Jako alternativa k MCE byla vyvinuta metoda založená na minimalizaci chyby ve slovech (angl. Minimum Word Error - MWE) [36]. MWE maximalizuje očekávanou přesnost rozpoznávání slov a může být jednoduše implementována pomocí slovních mřížek. Zároveň byla vyvinuta i metoda pro minimalizaci chyb ve fonémech (angl. Minimum Phone Error - MPE), která používá stejný přístup, ovšem na úrovni jednotlivých fonémů.

Pro lepší porovnání s MMI kriteriem si zopakujme nejdříve samotné MMI kriterium, kde na rozdíl od (33) a (44) bylo vážení akustického modelu ještě rozšířeno i na jazykovou část kritéria [36], jelikož se předpokládá, že jazykový model, již byl vážen váhou jazykového modelu $1/\kappa$ a tedy tímto je pouze navrácen k původním hodnotám, prakticky tedy odpovídá vzorci (44):

$$F_{MMI}(\lambda) = \sum_{e=1}^{E} \log \frac{P_\lambda(\boldsymbol{O}^e|W_R^e)^\kappa P(W_R^e)^\kappa}{\sum_{W \in \hat{\mathcal{W}}} P_\lambda(\boldsymbol{O}^e|W)^\kappa P(W)^\kappa}. \tag{60}$$

Tedy MMI maximalizuje aposteriorní pravděpodobnost správné posloupnosti slov. Jmenovatel kriteria pak může být aproximován slovní mřížkou, která obsahuje i alternativní hypotézy posloupností slov.

5.3.1 MWE kriterium

Samotné MWE kriterium je definováno takto:

$$F_{MWE}(\lambda) = \sum_{e=1}^{E} \log \frac{\sum_{W \in \hat{\mathcal{W}}} P_\lambda(\boldsymbol{O}^e|W)^\kappa P(W)^\kappa \text{RawAccuracy}(W)}{\sum_{W \in \hat{\mathcal{W}}} P_\lambda(\boldsymbol{O}^e|W)^\kappa P(W)^\kappa}, \tag{61}$$

kde RawAccuracy(W) je míra úspěšnosti rozpoznávání rozpoznané posloupnost slov W, κ je váha akustického modelu. Toto kriterium je tedy váženým průměrem správně rozpoznaných slov ze všech možných posloupností slov. Maximalizací MWE kriteria se zvyšuje počet správně rozpoznaných slov v nejpravděpodobnějších posloupnostech. Pro efektivní výpočet RawAccuracy(W) ze slovních mřížek byla navržena následující aproximace:

$$\text{RawAccuracy}(W) = \sum_{w \in W} \text{WordAcc}(w), \tag{62}$$

$$\text{WordAcc}(w) = \begin{cases} -1 + 2e & \text{v případě totožného slova} \\ -1 + e & \text{v případě jiného slova} \end{cases}, \tag{63}$$

kde e vyjadřuje míru překrývání slova w v čase se slovem ze správné posloupnosti slov.

5.3.2 MPE kriterium

Místo maximalizace úspěšnosti rozpoznávání na slovech, můžeme maximalizovat úspěšnost rozpoznávání na úrovni fonémů [42]. Formální zápis MPE kritéria je pak identický s (61) s výjimkou úspěšnosti RawAccuracy(W), která v MPE vyjadřuje (relativní) počet správně rozpoznaných fonémů. Statistiky pro nové odhady parametrů lze u MPE rovněž vypočíst ze slovní mřížky, ve které musí být navíc zahrnuta informace o hranicích jednotlivých fonémů. Nové odhady samotné se pak vypočítají pomocí stejných vztahů (34) a (35) jako v MMI případě, detailně je celý postup popsán v [42].

Pokud mají MWE nebo MPE dosahovat dobrých výsledků, je třeba použít metodu I-smoothing (viz kapitola 5.1.7) pro zvýšení schopnosti zobecňovat na testovací data a zároveň tím ochránit trénovací proces proti možnosti přetrénování [36]. MWE metoda obvykle vykazuje lepší úspěšnost rozpoznávání na trénovací sadě, MPE je však robustnější a má větší schopnost zobecňovat, takže dosahuje na testovacích datech lepších výsledků než MWE. V úloze *Switchboard/Call Home* s 68 i 256 hodinami trénovacích dat dosahoval MPE model o přibližně $0,5\%$ absolutně lepší úspěšnosti než MMI model s I-smoothingem [36].

5.4 Diskriminativní trénování z fonémových mřížek

Jak již bylo uvedeno výše, diskriminativní metody odhadují správné parametry iterativně a většina z nich vyžaduje celý rozpoznávací běh pro tvorbu slovních mřížek, ze kterých vypočítává statistiky pro nové odhady. Tyto slovní mřížky jsou generovány většinou unigramovým jazykovým modelem, aby obsahovaly dostatečné množství konkurenčních hypotéz, dále je třeba nastavit práh prořezávání dostatečně benevolentně. To dělá generování mřížek i výpočet statistik extrémně časově náročné, rovněž uložené mřížky na disku pak zabírají nemalé místo. Často se mřížky generují pouze jednou a ukládají se i hranice jednotlivých fonémů, které se pak při výpočtu statistik považují za "přesné". Tyto aproximace pak mírně snižují kvalitu natrénovaného modelu, nicméně značně urychlují průběh trénování [34]. V [56] je pak ukázáno jak lze efektivně, v případě unigramového jazykového modelu, generovat mřížku bez označení jednotlivých slov, a ukládat pouze fonetický přepis s integrovanými pravděpodobnostmi jednotlivých slov. Takto lze vytvořit poměrně kompaktní mřížku, která se vejde bez problému do paměti počítače i v úlohách s velkým slovníkem. Další možností je, generovat místo slovních mřížek obdobným způsobem mřížky fonémové [55].

5.4.1 Kombinace kriterií

Výše uvedeným způsobem pro generování fonémových mřížek (jak je popsáno v [55]) lze s výhodou vypočítat potřebné statistiky pro MMI i MPE diskriminativní metody najednou. Vhodnou kombinací MMI a MPE metod pak lze dosáhnout až relativní $12,5\%$ pokles chyby rozpoznávání, oproti ML modelu, kdežto samotná metoda MMI dosahuje $9,4\%$ pokles a MPE $7,5\%$. Vše bylo trénováno na 400 hodinách dat z korpusů *Broadcast News - BN* a *TDT4*, testováno pak na 3 hodinách z korpusu *BN*.

5.4.2 Modifikace MPE kriteria na MPFE

Ve stejném článku [55] byla rovněž navržena úprava MPE metody. MPE metoda nedostatečně penalizuje chyby typu smazání (delatizace). Dále míra chybovosti ve fonémech RawAccuracy je obvykle příliš omezená a naakumulované statistiky jsou výrazně nižší než u MMI, což vede k méně robustním odhadům nových parametrů [36]. Pro korekci těchto problémů, byla v [55] navržena úprava

míry fonémové chybovosti:

$$\text{PhoneFrameAcc}(q) = \sum_{t=start(q)}^{end(q)} P(s_t \in S(q)|W, \boldsymbol{O}), \tag{64}$$

kde q je vyhodnocovaný foném; $S(q)$ označuje posloupnost stavů HMM, které modelují foném q; $start(q)$ a $end(q)$ reprezentují čas začátku a konce fonému q; $P(s_t \in S(q)|W, \boldsymbol{O})$ je aposteriorní pravděpodobnost HMM stavu s_t náležícího k $S(q)$ v čase t, dané posloupnosti vektorů pozorování $\boldsymbol{O}$ a posloupnosti slov W. Tato pravděpodobnost může být vypočtena standardním forward-backward algoritmem (viz kapitola 4.4.1). Použitím míry (64) dostáváme tzv. *Minimum Phone Frame Error - MPFE* kriteriální funkci:

$$F_{MPFE}(\lambda) = \sum_{e=1}^{E} \sum_{S} P_\lambda(s|\boldsymbol{O}^e)\text{FPhoneAccuracy}(s), \tag{65}$$

kde

$$\text{FPhoneAccuracy}(s) = \sum_{q \in S} \text{PhoneFrameAcc}(q) \tag{66}$$

je míra počtu příznakových vektorů (angl. frame), které jsou klasifikovány ke správnému fonému pro hypotézu S. Maximalizace kritéria 65 je tedy MPFE trénováním. Vzhledem k formálnímu zápisu odpovídajícímu MPE metodě, je MPFE i implementačně velice podobné. Vzhledem k tomu, že všechny konkurující hypotézy jsou v čase stejně dlouhé, nemohou zde nastat problémy s benevolencí vůči chybám typu smazání jako u MPE. Navíc $FPhoneAccuracy$ má větší dynamický rozsah než obdobná míra používaná v MPE, tedy množství akumulovaných statistik je větší a to přispívá k větší robustnosti výsledných odhadů.

Lepší úspěšnost rozpoznávání byla prokázána na modelu trénovaném z 1400 hodin korpusů *Switchboard* a *Fisher*. Byly porovnávány dvě varianty: kombinace MMI-MPE a MMI-MPFE. Testovací sada pocházela z oficiálních srovnávacích testů NIST CTS z let 2001 až 2004. Kombinace MMI-MFPE překonávala MMI-MPE přibližně o $0,5\%$ absolutně a oproti ML o 2%.

5.5 Metody založené na optimalizaci bezpečnostního pásma

Jelikož i přes velké úsilí a množství navržených metod je v diskriminativním trénování stále velkým problémem robustnost výsledných modelů, které velice

dobře fungují na datech trénovacích, ale data testovací se svými odlišnostmi dosahují jen mírného zlepšení. Z metod navržených pro standardní klasifikaci nejlépe řeší tento problém metoda pomocných vektorů (angl. Support Vectors - SV) [9, 20]. Tento přístup optimalizuje kriterium složené ze dvou částí. První část je mírou o úspěšnosti klasifikace trénovacích dat. Druhá část kritéria nese informaci o "šířce bezpečnostního pásma" mezi klasifikovanými třídami. Tedy pokud se během trénování snažíme, nejen o největší úspěšnost na trénovacích datech, ale i o co největší oddělení klasifikovaných tříd od sebe, je pravděpodobné, že pokud testovací data nebudou rozložením přesně odpovídat datům trénovacím, výsledná klasifikace dopadne lépe.

Využití stejného přístupu se v nedávné době objevuje i u diskriminativního modelování akustických modelů založených na HMM [14, 15, 16, 17, 18, 19]. Zatím jsou tyto metody na počátku svého vývoje a první výsledky jsou dosahovány na poměrně velice jednoduchých úlohách. Jako základní referenční úlohu si všechny výše citovaní vybrali *TIDIGITS*, v ní jde o rozpoznávání sérií číslovek. Tyto číslovky jsou většinou modelovány každá jedním HMM, tedy nejsou tu trénovány žádné fonémy ani trifony, ale vždy jedna číslovka jako celek. Chybovost rozpoznávání se v této úloze hodnotí buď na slovech/číslovkách, kde se díky jednoduchosti úlohy blíží téměř k 0%, nebo na celých promlouvaných sekvencích číslovek, kde je pak chybovost o něco větší.

V [14] je použita úprava MCE metody nazvaná *Incrementally Regulated Discriminative Margin*. Jedná se o zakomponování postupně se zvětšujícího bezpečnostního pásu do běžného MCE kritéria. Oproti standardnímu MCE přístupu byla chyba na celých sekvencích redukována o 19% relativně na $0,55\%$.

V [15] je použit přístup nazývaný *Large Margin Estimation - LME* a popisuje zejména nově navržený algoritmus *SemiDefinite Programing - SDP*, který dokáže velice efektivně odhadnout nové parametry modelu, takže trénování modelu LME přístupem není tak extrémně časově náročné jako například porovnatelné gradientní metody. Pro modely s 32 složkami pro každý stav HMM, bylo dosaženo následující chybovosti na celých sekvencích číslovek: $1,34\%$ pro standardní ML přístup, $0,90\%$ pro MCE, $0,66\%$ pro gradientní LME a nejlepšího výsledku bylo dosaženo s nově navrženou metodou LME-SDP $0,53\%$. O rok později byl publikován navazující článek [18], kde byla navržena další nová metoda pro odhad parametrů LME založená na tzv. *konickém programování druhého řádu* (angl. Second Order Cone Programing - SOCP). Ta nedosahuje sice tak dobrých výsledků co se týče chybovosti: $0,66\%$. Ale je oproti gradientní metodě o několik řádů rychlejší a oproti LME-SDP je rychlejší 223 krát při odhadu modelu s 32 složkami.

V [16] je použit přístup založený na minimální divergenci (angl. Minimum Dive-

gence - MD) jako nové míře chybovosti v diskriminativním trénování. Pro definici míry chybovosti mezi dvěma libovolnými HMM pak byla použita tzv. *Kullback-Leibler Divergence - KLD*. Pro modely s 6-ti složkami pro každý stav HMM, bylo dosaženo následující chybovosti na slovech/číslovkách: $1,17\%$ pro standardní ML přístup, $0,63\%$ pro MPE a $0,48\%$ pro MD. Chybovost na celých sekvencích číslovek ani modely s větším počtem složek nejsou v článku bohužel uvedeny a tedy nelze tyto výsledky přímo srovnat s ostatními.

V [17] je navržen nový přístup zvaný *Soft Margin Estimation - SME*, který odvozuje nové kritérium pro HMM modely z obdobné metody používané pro standardní klasifikaci pomocí přístupu pomocných vektorů. Toto kriterium je pak optimalizováno gradientními metodami, které jsou v této práci popsány v kapitole 5.2.4. Pro modely s 32 složkami pro každý stav HMM, bylo dosaženo následující chybovosti na celých sekvencích číslovek: $1,49\%$ pro standardní ML přístup, $1,02\%$ pro MCE, $0,72\%$ pro gradientní LME a pro nově navrženou metodu SME $0,67\%$. O rok později pak v [19] je navržen přístup nazvaný *Soft Margin Feature Extraction - SMFE*, který spojuje odhad parametrů HMM modelu s odhadem transformační matice vstupních vektorů pozorování. Touto metodou bylo dosaženo velmi nízké chyby sekvencí číslovek: $0,39\%$ a to na model s jen 16-ti složkami a i jen jednosložkový model dosahuje chyby $0,87\%$. Což jsou v současné době nejlepší dosažené výsledky.

5.6 Diskriminativní adaptace

Kromě samotného trénování akustických modelů, patří do problematiky akustického modelování i adaptace těchto natrénovaných modelů na reálné podmínky. V praxi přichází v úvahu hned několik scénářů a tomu odpovídajících metod.

Lineární transformace je používána tam, kde je velmi malé množství adaptačních dat. Pro adaptaci pomocí lineární transformace lze využít i ne úplně přesný přepis již rozpoznané části promluvy a zbytek této promluvy rozpoznávat již s modelem adaptovaným s větší přesností [44, 45]. Nejpopulárnější metoda je Maximálně věrohodná lineární regrese (angl. Maximum Likelihood Linear Regresion - MLLR) [43]. Ta přináší dobré výsledky na ML trénovaných modelech. Zajímavé však je její použití na diskriminativní model. V [38] dosahuje v úloze *Hub5* metoda MLLR stejného zvýšení úspěšnosti rozpoznávání jak na ML tak i na MMI trénovaných modelech. To je v kontrastu s [51], kde adaptace MMI modelu nevedla ke zvýšení úspěšnosti rozpoznávání. Zde se jednalo o korpus *English Spontaneous Scheduling Task - ESST*.

Problémy samozřejmě způsobuje adaptace založená na jiném kritériu než jakým byly trénovány adaptované modely. Proto byly navrženy rovněž diskriminativní adaptační algoritmy, které tento problém odstraňují. Nicméně využití diskriminativní adaptace komplikuje fakt, že právě diskriminativní metody, jsou více závislé na množství trénovacích dat, kterých je při adaptaci citelný nedostatek. Ale i přesto lze dosáhnout lepších výsledků než u metody MLLR. Diskriminativní lineární transformace (angl. Discriminative Linear Transform - DLT) [52] je jedna z možností. DLT využívá H-kritéria (viz kapitola 5.1.6), což je interpolace mezi ML a MMI kriteriem. Transformační matice je zde počítána iterativně. Konvergence algoritmu není zaručena, ale i tak vede v praxi k dobrým výsledkům. V úloze, kde byl model rodilých mluvčích adaptován na nerodilé mluvčí, dosáhl tento přístup o $0,8\%$ absolutně lepšího výsledku než standardní MLLR. Kromě DLT existují i jiné úspěšné metody diskriminativní adaptace, některé z nich jsou popsány například v [46] a [47].

V případě, že je adaptačních dat k dispozici více, například několik desítek minut z prostředí, kde má být systém nasazen, lze použít jiné metody. Nejpoužívanější metoda v tomto případě je metoda Maximální aposteriorní pravděpodobnosti (angl. Maximum A Posteriori probability - MAP) [48]. Stejně jako v předchozím případě, lze tuto metodu využít pro adaptaci jak ML trénovaných modelů, tak modelů diskriminativních, kde ovšem výsledek nebude adekvátní. I pro tento případ byly vyvinuty metody diskriminativní adaptace [49, 50]. V [50] je popsán obecný způsob adaptace, který lze využít jako pro MMI adaptaci tak pro MPE či MWE adaptaci. Apriorní informace o původním modelu je v tomto přístupu zakomponována do I-smoothing členu. Tento přístup byl testován na korpusu *Switchboard*, kde bylo z 265 trénovacích dat natrénovány dva modely ML a MMI. Oba s 6684 stavy a 16 složkami na stav. Tento model byl adaptován na data z korpusu *Voicemail*, na kterém byly i výsledné modely testovány. Velikost adaptačních dat se pohybovala v rozsahu od 1 do 28 hodin a byly testovány všechny čtyři kombinace $ML \rightarrow MAP$, $ML \rightarrow MMI - MAP$, $MMI \rightarrow MAP$ a $MMI \rightarrow MMI - MAP$. Prvotní neadaptovaný MMI model dosahoval o 4% absolutně lepších výsledků oproti původnímu ML modelu. Tento rozdíl, se v závislosti na množství adaptačních dat snížil, nicméně u obou metod adaptace byl rozdíl podobný a i pro 28 hodin byl stále 2%. Porovnání MAP a MMI-MAP vychází rovněž pro diskriminativné verzi lépe. Rozdíl je opět závislý na množství adaptačních dat a postupně vzrůstá od téměř totožného výsledku na jedné hodině, po 2% absolutně pro 15 a více hodin.

5.7 Diskriminativní na řečníka adaptivní trénování

Trénovací data, ze kterých se trénuje akustický model, obsahují nahrávky od velkého množství řečníků s velmi rozdílnými hlasy. Pokud při rozpoznávání používáme adaptační metody, není nutné mít model natrénovaný různými hlasy tak, aby zachycoval tuto variabilitu řečníků. Tato variabilita v trénovacích nahrávkách naopak neumožňuje natrénovat dostatečně přesný model. V metodě zvané anglicky *Speaker Adaptive Training - SAT* [53] je využívána adaptace na řečníka také v procesu trénování, takže výsledný model netrpí tak velikou variabilitou trénovacích promluv a je přesněji natrénován.

V diskriminativním na řečníka adaptivním trénování (angl. Discriminative Speaker Adaptive Training - DSAT) je použit totožný přístup, jen adaptační metoda a metoda trénování samotná je diskriminativní. V [54] je pro adaptaci použita metoda DLT (viz kapitola 5.6) a pro trénování MMI a MPE. DSAT s MMI trénováním dosahoval o $0,6\%$ absolutně větší úspěšnosti rozpoznávání a DSAT s MPE o $0,8\%$ absolutně oproti MMI trénovanému modelu bez DSAT.

5.8 Další diskriminativní metody a modifikace

V této kapitole budou popsány některé další metody a modifikace. Tyto metody oproti výše uvedeným nejsou tak osvědčené a populární, prověřené dlouhodobější praxí, proto budou popsány spíše stručněji, nicméně nebylo by vhodné, kdyby v této práci nebyly uvedeny, jelikož v budoucnu se některé z nich mohou ještě rozvinout a mohou být používány šířeji.

5.8.1 fMPE

Tato metoda je založena na modifikaci vektoru pozorování (angl. feature vector), která bude zlepšovat kriterium MPE, proto je tato metoda označována jako *feature vector MPE - fMPE* [57]. Základní vztah pro transformaci je:

$$\boldsymbol{y}_t = \boldsymbol{x}_t + \boldsymbol{M}\boldsymbol{h}_t, \tag{67}$$

kde $\boldsymbol{x}_t$ je originální vstupní vektor pozorování v čase t a $\boldsymbol{y}_t$ je modifikovaný vektor pozorování. $\boldsymbol{h}_t$ je vektor vygenerovaných příznaků pro čas t, který má velkou dimenzi, a je funkcí vstupního vektoru pozorování. Tento vektor je pak transformován zpět do prostoru stejné dimenze jako mají $\boldsymbol{x}_t$ a $\boldsymbol{y}_t$ pomocí transformační

matice $\boldsymbol{M}$. Pokud je použita opravdu velká dimenze, pak vektor $\boldsymbol{h}_t$ je velice *řídký*, tedy většina hodnot ve vektoru je nulová (nebo téměř nulová) a jen na několika místech jsou vyšší hodnoty. Tedy jen několik řádků transformační matice $\boldsymbol{M}$ bude mít vliv na generování modifikovaného vektoru pozorování $\boldsymbol{y}_t$ v čase t. Vztah (67) je definován jako modifikace vstupního vektoru pozorování $\boldsymbol{x}_t$, vektor $\boldsymbol{y}_t$ by mohl být generován i rovnou, bez součtu se vstupem, ovšem tato forma usnadňuje trénování matice $\boldsymbol{M}$, která takto může být inicializována jako nulová.

První fáze fMPE spočívá ve vygenerování vektoru $\boldsymbol{h}_t$ s vysokou dimenzí z originálního vstupního vektoru příznaků $\boldsymbol{x}_t$. Tento vektor může být sestaven například z jednotlivých pravděpodobností všech složek normálního rozložení z akustického modelu. Pokud je těchto složek až příliš veliké množství, mohou být poshlukovány do menšího počtu. Tento vektor může být dále rozšířen svými levými a pravými kontexty v určitém okolí. Výsledná dimenze vektoru $\boldsymbol{h}_t$ může dosahovat stovek tisíc. Ve druhé fázi probíhá samotné trénování matice $\boldsymbol{M}$, jak již bylo zmíněno výše, inicializována je jako nulová a z tohoto počátečního nastavení je trénována gradientními metodami (hlavní gradientní metody jsou popsány v kapitole 5.2.4). Z gradientních metod jsou vhodné zejména ty, které nevyžadují akumulování statistik pro odhady derivací druhých řádů, jelikož vzhledem k dimenzi matice $\boldsymbol{M}$ (a tedy počtu optimalizovaných parametrů) by to bylo již zřejmě prakticky obtížně realizovatelné. Detailněji je možné metodu prostudovat v [57]. Funkčnost tohoto přístupu byla ověřena hned na několika úlohách, například na úloze NIST RT-04 bylo dosaženo zlepšení u fMPE oproti ML modelu o 1.9% absolutně, u standardního MPE o 1.9% absolutně oproti ML. Kombinace fMPE a MPE pak dosahovala zlepšení o 3.0% absolutně.

5.8.2 Boosted-MMI

tzv. *Boosted-MMI* [58] je modifikace metody MMI, kde je navýšen vliv (angl. boosted) těch konkurenčních cest, při výpočtu jmenovale kritéria, které mají větší chybovost. Tato chybovost je vyjadřována stejným způsobem jako v metodě MPE (případně MWE), takže se jedná o míru chybovosti ve fonémech vzhledem k referenčnímu přepisu promluvy. Použitím toho přístupu tedy dostáváme metodu, která je jakousi kombinací mezi MMI a MPE. Souvislost mezi těmito metodami lze nejlépe demonstrovat na vztazích jednotlivých kriterií. MMI kriterium (viz (33) či (44)):

$$F_{MMI}(\lambda) = \sum_{e=1}^{E} \log \frac{P_\lambda(\boldsymbol{O}^e|W_R^e)^\kappa P(W_R^e)}{\sum_{W\in\hat{\mathcal{W}}} P_\lambda(\boldsymbol{O}^e|W)^\kappa P(W)}. \tag{68}$$

MPE kriterium (viz kapitola 5.3):

$$F_{MPE}(\lambda) = \sum_{e=1}^{E} \log \frac{\sum_{W \in \hat{\mathcal{W}}} P_\lambda(\boldsymbol{O}^e|W)^\kappa P(W) A(W, W_R^e)}{\sum_{W \in \hat{\mathcal{W}}} P_\lambda(\boldsymbol{O}^e|W)^\kappa P(W)}, \tag{69}$$

kde $A(W, W_R^e)$ je míra chybovosti, mezi hypotézou W a referenčním přepisem W_R^e počítaná na fonémech. Nové kriterium Boosted-MMI (zkr. BMMI) pak definujeme

$$F_{MMI}(\lambda) = \sum_{e=1}^{E} \log \frac{P_\lambda(\boldsymbol{O}^e|W_R^e)^\kappa P(W_R^e)}{\sum_{W \in \hat{\mathcal{W}}} P_\lambda(\boldsymbol{O}^e|W)^\kappa P(W) \exp(-bA(W, W_R^e))}, \tag{70}$$

kde b je parametr tzv. *boosting factor* určující velikost vlivu míry chybovosti (např. $b = 0,5$). Tímto je tedy zesilován vliv těch hypotéz, které mají větší chybovost a tedy odhady nových parametrů budou tyto chyby více potlačovat. Míra chybovosti může být definována i jiným způsobem, například na slovech (odpovídá pak kriteriu MWE) nebo na stavech HMM a pak odpovídá spíše některým metodám pracujícím s tzv. *bezpečnostním pásmem* (viz kapitola 5.5).

V [58] byla tato metoda testována na korpusu *English Broadcast News - EBN*. Pro trénování bylo použito 700 hodin trénovacích dat, pro testování byla použita sada *NIST RT-04*. Všechny natrénované modely vycházely z ML modelu, který měl 6 tisíc stavů a celkem 250 tisíc složek. ML model dosahoval chyby na slovech $20,5\%$, MMI metoda $18,9\%$, MPE metoda $18,6\%$, kombinace MMI-MPE $18,1\%$. Nově navržená metoda BMMI dosahovala chybovosti $17,9\%$ a po optimalizaci paramtru vážení akustického modelu dokonce $17,3\%$.

5.8.3 Diskriminativní dělení složek modelu

Standardně se metody diskriminativního trénování používají jen pro úpravu parametrů modelu. Struktura modelu včetně množství složek zůstává nezměněná a pochází z modelu trénovaného podle kritéria ML. Diskriminativně trénované parametry jsou pak trénovány většinou iterativně a konvergují (pokud je konvergence vůbec zaručena) jen k lokálnímu extrému, který závisí na počátečním ML modelu. Vzhledem k této závislosti a ke zjištění, že diskriminativní kritéria vedou k lépe natrénovaným modelům, je také snaha využít diskriminativní kritérium již v některých částech trénování, které jsou zatím trénovány standardně podle kritéria ML. Prakticky poslední fází trénování je přidávání složek modelu (viz kapitola 4.3). V této fázi trénování je většinou největší složka stavu (složka s největší vahou) rozdělena na dvě, které dohromady modelují data této původní složky. Parametry těchto nových složek, jsou pak v dalších iteracích trénování náležitě upraveny. Je

možné změnit tento postup a pro jednotlivé složky vyhodnotit očekávanou změnu například MMI kritéria při rozdělení této složky. V každé iteraci pak dělí jen ty složky, jejichž rozdělením dojde k největšímu nárůstu MMI kritéria [60].

Tímto postupem bylo dosaženo až 41% relativního úbytku chyby v úloze rozpoznávání serií číslovek [60]. Ve složitější úloze na korpusu *Wall Street Journal - WJS* [29] bylo dosaženo snížení chyby o 3 až 5% relativně. Ovšem jen pro malý počet složek modelu. Pro větší počty složek se účinnek snižuje.

5.9 Závěrečné shrnutí diskriminativních metod

Z výše uvedených popisů jednotlivých metod a výsledků tedy není pochyb, že vhodná diskriminativní kritéria poskytují odhady parametrů modelu, které jsou blíže optimálním hodnotám než modely trénované podle kritéria ML a tedy výsledné systémy rozpoznávání řeči dosahují lepších výsledků. Jednotlivé metody je ovšem velmi těžké obecně zhodnotit a porovnat, jen velmi malý počet publikací srovnává výsledky více než například dvou těchto metod. Je to pochopitelné, jelikož praktická implementace je velice náročná, navíc záleží nejen na zvoleném kritériu, ale na přesné metodě, její implementaci a nastavení parametrů. Z prostudované literatury se poměrně často ukazuje, že místo objektivně vhodnějších kriterií či diskriminativních metod, dosahují lepších výsledků metody lépe implementované a optimalizované, tedy zejména metody dlouhou dobu vyvíjené na daném pracovišti. Jedno ze srovnání většího množství metod je například v [80], kde byl pro trénování použit korpus *Wall Street Journal - WJS* a pro testování *North American Business news - NAB*. Trénovací data obsahovala 81 hodin, testovací jednu hodinu. Natrénovaný model měl 7000 stavů s 32-mi složkami a jednu sdílenou diagonální kovarianční matici. Testovací slovník obsahoval 65 tisíc slov. Hodnocena byla chybovost jednotlivých metod na slovech tzv. *Word Error Rate - WER*. Výsledky jsou v tabulce 1. Jednotlivé dikriminativní metody se ve výsledcích příliš neliší a jejich výsledky jsou prakticky totožné, jen MPE zaostává za ostatními a výsledky má blíže k ML výchozímu modelu. Jak bylo zmíněno výše, výsledky ovšem závisí i na konkrétní implementaci a nastavení, takže na základě těchto výsledků nelze obecně konstatovat, že MPE kriterium je z testovaných nejhorší.

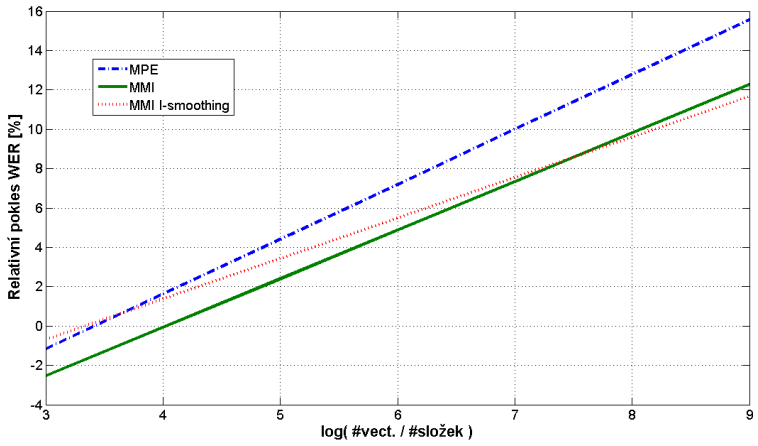
Velmi zajímavé je rovněž zkoumat robustnost odhadů jednotlivých metod, tedy schopnost zobecňovat na testovací data. I zde je ovšem problém mít k dispozici několik metod s porovnatelnou implementací. V [42] jsou z toho to pohledu porovnány metody MPE, MMI a MMI s I-smoothingem. Všechny experimenty byly vyhodnoceny a výsledky jednotlivých metod byly aproximovány

	WER [%]
ML	9,35
MMI	8,97
MCE	8,97
MWE	9,03
MPE	9,30

Tabulka 1: Porovnání chybovosti diskriminativních metod

Porovnání chybovosti (WER) jednotlivých diskriminativních metod na korpusu NAB

lineární funkcí. Podle výsledků v obrázku 5, kde vertikální osa odpovídá relativnímu poklesu chyby ve slovech (WER) a horizontální osa odpovídá logaritmu poměru množství trénovacích dat (počet vektorů pozorování) vůči celkovému počtu složek akustického modelu. Je zřejmé, že čím větší počet trénovacích dat je k dispozici, tím lepších výsledků všechny vyhodnocené diskriminativní metody dosahují. Ovšem je třeba tento graf hodnotit i z opačné strany. Diskriminativní metody dosáhnou lepších výsledků oproti ML, jestliže počet složek modelu je spíše nižší. Poměrně často lze v některých publikacích sledovat vyhodnocování přínosu diskriminativního trénování na modelech s menším než optimálním (z hlediska ML) množství složek, dosažené výsledky se pak mohou jevit jako vynikající, ačkoli na komplexnějších akustických modelech může být dosahováno pouze mírného či žádného zlepšení. Na závěr ještě poznamenejme, že uvedené lineární aproximace jsou velice hrubé a velmi záleží na konkrétní úloze, nicméně základní závislost úspěšnosti DT metod na množství trénovacích dat a komplexnosti trénovaného modelu je prokazatelná i na mých vlastních experimentech.



Obrázek 5: *Relativního poklesu WER v závislosti na množství trénovacích dat a komplexnosti akustického modelu.*

6 Vlastní vývoj, experimenty a výsledky

V této části práce bude postupně prezentován vývoj vlastního systému diskriminativního trénování. Prezentovány a diskutovány budou rovněž výsledky navržených metod. Vzhledem k cílům práce je většina experimentů prováděna na českých korpusech. Avšak navržené metody jsou na jazyku nezávislé a lze je využít jak pro anglický jazyk tak i pro většinu ostatních jazyků.

6.1 Úvodní experimenty na korpusu UWB S01

Pro úvodní experimenty byl vybrán korpus *UWB S01* [4] a to hned z několika důvodů. Za prvé, tento korpus obsahuje dostatečné množství řečníků i dat pro trénování různě velkých akustických modelů. Všechna trénovací data jsou nahrávána za kontrolovaných podmínek, tedy použitá data jsou velmi dobré kvality. Tento korpus je pro různé účely využíván na naší katedře už řadu let a tedy všechny nahrávky a jejich přepisy byly několikrát kontrolovány a promluvy se špatnou kvalitou signálu, poruchami či špatným přepisem byly vyřazeny.

6.1.1 Popis korpusu UWB S01

Motivací pro vytvoření tohoto korpusu byla potřeba standardizovaného korpusu čtené řeči pro český jazyk, který by byl obdobou například anglického korpusu *Wall Street Journal (WSJ)*. V úvodní části tvorby korpusu bylo nahráno 100 řečníků. Každý z řečníků přečetl 150 vět pocházejících ze tří hlavních českých deníků, což činilo přibližně 20 minut záznamu na řečníka. Použité věty byly speciálně vybrány tak, aby byly foneticky co nejbohatší a obsahovaly co největší množství trifonů, a tedy, aby trifonový akustický model natrénovaný z tohoto korpusu byl kvalitní. Nahrávání probíhalo v relativně tiché kanceláři a mezi řečníky byly jak muži tak ženy různého věku. Každá věta byla nahrávána paralelně dvěma mikrofony. První pro co nejkvalitnější záznam zvuku byl *Sennheisser HMD 410-6*, do kterého řečník mluvil přímo. Druhým mikrofonem byl stolní mikrofon *Sennheisser ME 65*, který zachycoval řeč spolu se zvuky okolního prostředí. Vznikly takto dva paralelní záznamy obsahující stejnou řeč ovšem za různých podmínek. Pro další experimenty byla používána pouze kvalitnější část dat z mikrofonu *Sennheisser HMD 410-6*.

Postupem času byl korpus postupně rozšiřován o další řečníky, v současné době

jsou k dispozici nahrávky od přibližně 800 řečníků, nicméně ne všechny části korpusu jsou tak precizně prověřeny. Pro následující experimenty byly vybrány první dvě části korpusu, které patří mezi nejlépe prověřené a tedy i výsledky dosažené na těchto datech budou věrohodné. Pro trénování bylo použito 100 řečníků, celková velikost vybraných trénovacích dat byla $6,3$ hodiny. Pro testování byla vybrána data jiných 100 řečníků, kde od každého z nich byla náhodně vybrána jedna věta. Celková délka testovacích vět pak byla 25 minut. Vybrané věty obsahovaly celkem 1400 slov.

6.1.2 Popis zpracování signálu

Analogový signál byl během nahrávání vzorkován $44,1$ kHz a ukládán s 16-ti bitovou přesností. Na tento signál bylo aplikováno Hammingovo okénko délky 32 milisekund s posunem 10 milisekund. Takto vytvořené segmenty byly zpracovány pomocí FFT a dále byly spočteny Melovské frekvenční koeficienty (MFCC). Bylo použito 27 trojúhelníkových frekvenčních filtrů a výstupem bylo 15 kepstrálních koeficientů včetně koeficientu energetického. Tento vektor pozorování byl doplněn o dynamické příznaky prvního a druhého řádu (tzv. *delta* a delta-delta koeficienty). Výsledná dimenze příznakového prostoru pak byla 45. Jednotlivé použité metody zpracování signálu jsou v této práci popsány v kapitole 2.1.

6.1.3 Referenční modely

Jak bylo popsáno v kapitole 5 většina diskriminativních metod netrénuje akustický model od začátku, ale využívá model trénovaný ML kritériem, které je dostatečně stabilní a osvědčené a lépe se hodí pro start trénování. Pro svoje experimenty jsem použil referenční postup popsaný v kapitole 4. Modely byly trénovány pomocí software HTK [23]. Pro porovnání různých algoritmu trénování a jejich vlastností byly vytvořeny čtyři modely. Všechny jsou založeny na fonetické abecedě obsahující 42 fonémů, jejíž součástí jsou i symboly pro krátkou a dlouhou pauzu, dále nádech, šum a kliknutí myši, všechny fonémy či trifony byly modelovány třístavovými HMM (se třemi emitujícími stavy). Konkrétní rozdíly mezi modely jsou následující:

- ***Monof10*** - je monofonový model. Počet stavů HMM je 123 a pro modelování každého z nich je použity 2 složky. Tento model byl trénován jen z prvních 10-ti řečníků (11 minut dat) a sloužil spíše jen pro rychlé ladící testy.

- ***Monof100*** - je monofonový model. Počet stavů HMM je rovněž 123 a pro modelování každého z nich je použito 8 složek a všechna trénovací data.
- ***TrifSmall*** - je trifonový model se sdílenými stavy. Počet stavů HMM po shlukování je 425 a pro modelování každého z nich je rovněž použito 8 složek.
- ***TrifLarge*** - je trifonový model se sdílenými stavy, kde práh pro shlukování byl nastaven menší a počet stavů HMM po shlukování u tohoto modelu vzrostl na 1424. Pro modelování každého z nich je rovněž použito 8 složek.

6.1.4 Vyhodnocení úspěšnosti rozpoznávání

Pro samotné rozpoznávání natrénovaných modelů byl použit software vytvořený na katedře kybernetiky. Pro co nejlepší posouzení akustické části systému, byl použit pouze zero-gramový jazykový model, ve kterém byla obsažena všechna slova z testovacích nahrávek se stejnou pravděpodobností. Perplexita úlohy byla 2424.

Výsledkem rozpoznávání řeči je odhadovaná nejpravděpodobnější posloupnost slov, která případně může obsahovat i časové značky. Pro testovací data máme k dispozici pro každou testovací větu referenční přepis, tedy posloupnost slov, která byla opravdu řečena. Pro vyhodnocení celkové kvality systému rozpoznávání řeči je možné použít několik různých měr. Například lze použít tzv. *přesnost* (angl. Accuracy - A_{cc}), která je definována vztahem:

$$A_{cc} = \frac{N - D - S - I}{N} 100[\%], \tag{71}$$

kde N je celkový počet slov v referenčním přepisu testovacích vět, S je počet chyb vzniklých záměnou slov, D je počet chyb vzniklých vynecháním (smazáním) a I je počet chyb vzniklých vložením nepatřičného slova. Poměr chyb I a D lze ovlivnit nastavením penalty vložení, která zabraňuje rozpoznávacímu algoritmu vkládat akusticky vhodnější nejkratší slova místo delších slov správných. Tato penalta byla nastavena takovým způsobem, aby *přesnost* rozpoznávání byla co největší. V literatuře se ovšem spíše než *přesnost* uvádí chyba na slovech (Word Error Rate - WER), která je jen zbytek do 100% k A_{cc} (tedy $WER = 100\% - A_{cc}$). WER je užívána zejména proto, že její relativní pokles je jako míra zlepšení výsledků nělterého algoritmu poměrně dobře přenosná na jiné korpusy a úlohy. Tedy pokud nějaký konkrétní algoritmus sníží WER na jedné úloze o 1% z 10% na 9% a na jiné obtížnější úloze o 4% z 40% na 36% je na první pohled účinek algoritmu rozdílný (1% versus 4%). Po přepočtení na relativní pokles WER však na obou úlohách dosahuje 10%.

Vyhodnocení chyby u všech referenčních modelů je v tabulce 2.

WER [%]	
Monof10	25,41
Monof100	19,63
TrifSmall	13,28
TrifLarge	10,14

Tabulka 2: Chyba rozpoznávání pro všechny referenční modely

6.1.5 Nová implementace ML reestimačního algoritmu

Jako první věc, na kterou je třeba se soustředit při implementaci nových metod, je samotný reestimační algoritmus pracující s kritériem ML (základní Baumův-Welchův algoritmus). Na tom lze ověřit, zda parametry modelu konvergují a tedy použitá implementace je správná. Rozšířený Baumův-Welchův algoritmus používaný pro diskriminativní trénování z toho základního samozřejmě vychází, tedy rovněž praktická implementace je jen rozšířením a tedy chyby, které by byly v základní verzi, by se projevily i ve verzi rozšířené. Kromě kvality odhadů je při trénování důležitá i paměťová a výpočetní efektivita použité implementace. U úloh s komplexnějšími akustickými modely a trénovacími daty ve stovkách hodin, pak může paměťová či výpočetní náročnost přesáhnout možnosti dostupných počítačů nebo trénování výsledného modelu může trvat mnohem déle, než je únosné. Tedy na výslednou implementaci je potřeba pohlížet i z tohoto pohledu.

Tato implementace Viterbi, Forward-Backward a Baumova-Welchova algoritmu byla navržena natolik obecně, aby mohla sloužit jak pro standardní ML trénování tak pro budoucí trénování diskriminativní. Pro ML trénování jsou možné dvě varianty. Buď Viterbiho algoritmem určit časové hranice posloupnosti stavů referenční promluvy. Pro každý příznakový vektor je takto vybrán pouze jeden stav modelu. Nebo pomocí Forward-Backward algoritmu určit pravděpodobnosti jednotlivých stavů modelu pro všechny časové okamžiky/vektory pozorování. Zejména u Forward-Backward algoritmu může být problém s velkou paměťovou náročností u dlouhých promluv a velkých modelů. Většina stavů modelu má však v daném čase prakticky nulovou pravděpodobnost a nenulová je jen u některých z nich. Proto je třeba tomu implementaci přizpůsobit a v paměti počítače uchovávat jen některé pravděpodobnosti přesahující určitý práh. Dále je potřeba vyřešit problémy s nu-

merickou nestabilitou výpočtu, kterých se může vyskytnout u těchto algoritmů celá řada. Část optimalizací použitých v implementaci je popsána v kapitole 7.

Pro otestování kvality, stability a rychlosti byl proveden experiment sestávající se z 10-ti ML reestimací, kde vstupem byly postupně všechny referenční modely. Rychlost jedné reestimace nepřesahovala 10 minut ani pro největší model. Výpočetní čas příliš na velikosti nezávisel, největším problémem byla spíše rychlost disku. Při využití Solid State Disků (SSD) lze dosáhnout výrazně kratších časů (do 5-ti minut, tedy více než 75-krát rychleji než reálný čas trénovacích nahrávek), ale zejména pak není veliký problém najednou trénovat více úloh či modelů z dat uložených na tom samém disku. Výsledky pro Viterbiův algoritmus jsou v tabulce 3. Pro Baumův-Welchův algoritmus v tabulce 4. Z tabulek je zřejmé, že Baumův-Welchův algoritmus dosahuje lepších výsledků, než Viterbiho aproximace, zejména při malém množství trénovacích dat. U modelu *Monof10* je rozdíl mezi těmito dvěma přístupy největší. Zajímavé ovšem je, že výsledky dosažené nově navrženou implementací Baumova-Welchova algoritmu jsou lepší, než referenční modely. Tento pokles WER by mohl být způsoben i jen větším množstvím reestimací, ovšem pro kontrolu byl proveden stejný počet dalších reestimací i softwarem HTK a k žádnému dalšímu poklesu chyby nedocházelo. Z prezentovaných výsledků je tedy patrné, že nová implementace Baumova-Welchova algoritmu je plně funkční a dokonce předčí i implementaci referenční.

Pro další ilustraci procesu trénování byl průběh WER vynesen do grafu. V grafu 6 jsou zobrazeny data z tabulky 4, tedy průběh WER v jednotlivých iteracích vlastní implementace Baumova-Welchova algoritmu. V dalším grafu 7 je pak vynesen průběh ML kriteria (v logaritmech) pro jednotlivé modely během trénování, kde je patrná konvergence algoritmu a dále je také zřejmé, že více komplexní modely dosahují větších hodnot kriteria.

6.1.6 Nově navržená metoda diskriminativního trénování založená na kriteriu MMI-FD

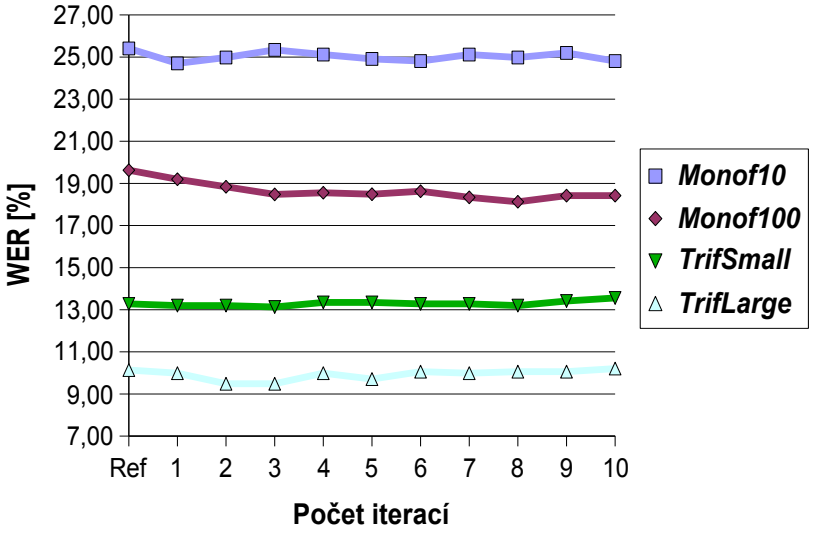
Jednou z prvních navržených diskriminativních metod je metoda založená na kriteriu MMI ve frame-diskriminativní verzi (viz kapitola 5 a 5.1.3). Tato metoda (bude nadále značena MMI-FD) nevyžaduje generování mřížek ani rozpoznávací běh trénovací promluvy. Proto jí lze snadno aplikovat na různá data a úlohy bez nutnosti připravit pro tuto úlohu jazykový model pro trénovací data, což je někdy velice komplikované a u některých úloh ani nejsou k dispozici data pro natrénování takovéhoto jazykového modelu.

Počet iterací	WER [%]			
	Monof10	*Monof100*	*TrifSmall*	*TrifLarge*
Ref.	25,41	19,63	13,28	10,14
1	25,86	19,63	12,99	10,56
2	26,12	19,49	13,20	10,35
3	26,12	18,99	13,13	9,92
4	25,91	18,70	13,06	9,92
5	26,34	18,87	13,35	10,06
6	26,70	18,92	13,28	9,99
7	26,84	19,13	13,20	10,28
8	26,48	19,13	13,13	10,42
9	26,34	18,92	12,85	10,99
10	26,41	18,99	13,13	11,21

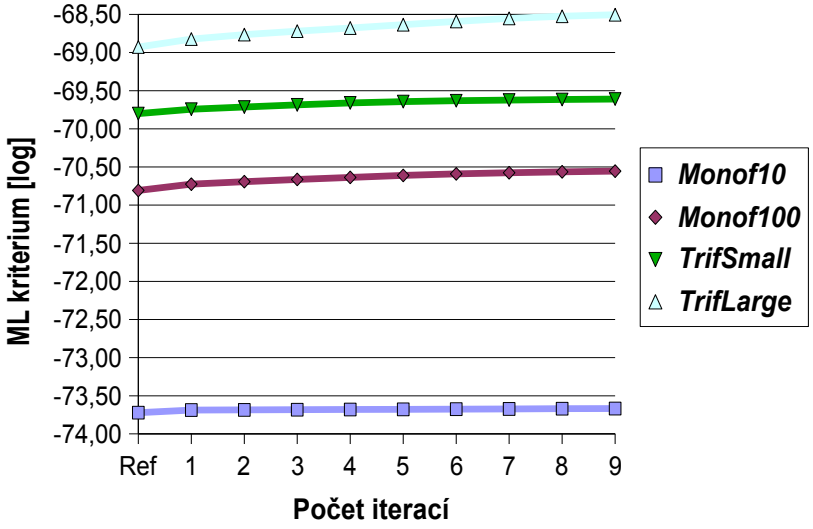
Tabulka 3: Chyba rozpoznávání pro všechny referenční modely pro Viterbi algoritmus

Počet iterací	WER [%]			
	Monof10	*Monof100*	*TrifSmall*	*TrifLarge*
Ref.	25,41	19,63	13,28	10,14
1	24,70	19,20	13,20	9,99
2	24,98	18,84	13,20	9,49
3	25,34	18,48	13,13	9,49
4	25,12	18,56	13,35	9,99
5	24,91	18,49	13,35	9,71
6	24,81	18,63	13,28	10,06
7	25,12	18,34	13,28	9,99
8	24,98	18,13	13,20	10,06
9	25,20	18,42	13,42	10,06
10	24,81	18,42	13,56	10,21

Tabulka 4: Chyba rozpoznávání pro všechny referenční modely pro Baumův-Welchův algoritmus



Obrázek 6: *Chyba rozpoznávání pro všechny referenční modely pro Baumův-Welchův algoritmus.*



Obrázek 7: *Průběh ML kriteria v jednotlivých iteracích.*

Tento přístup se snaží tedy oproti standardnímu MMI zvýšit počet konkurujících si stavů v každém čase tím, že nechává konkurovat všechny stavy modelu najednou. Tento přístup se blíží speciálnímu případu MMI, ve kterém bychom měli tak velký a variabilní slovník a uvažovali tolik hypotéz, že by se nám mohly vyskytnout v daném čase všechny stavy modelu se stejnou apriorní pravděpodobností. U této metody je nutné akumulovat dvě skupiny statistik, statistiky čitatele kriteria (*num*) a statistiky jmenovatele (*den*). Výpočet statistik čitatele je totožný se základní verzí Baumova-Welchova algoritmu, která byla diskutována a testována výše, tedy i její implementaci lze využít. Ta používá forward-backward algoritmus pro určení $\gamma_{jm}^{num}(t)$, což je aposteriorní pravděpodobnost, že m-tá složka stavu s_j generuje výstupní vektor $\boldsymbol{o}_t$ v čase t na základě referenční posloupnosti stavů S^R a posloupnosti příznakových vektorů $\boldsymbol{O}$. Na základě vypočtených $\gamma_{jm}^{num}(t)$ přepočteme akumulátory $\boldsymbol{\Theta}_{jm}^{num}(\boldsymbol{O})$ a $\boldsymbol{\Theta}_{jm}^{num}(\boldsymbol{O}^2)$, které pak slouží k výpočtu nových odhadů parametrů podle vzorců (34) a (34) v kapitole 5.1.1.

Výpočet akumulátorů jmenovatele je poměrně jednoduchý. Je jen třeba určit postupně všechny $\gamma_{km}^{den}(t)$, tedy aposteriorní pravděpodobnosti, že m-tá složka stavu s_k generuje vektor pozorování $\boldsymbol{o}_t$ v čase t. Tato pravděpodobnost je počítána z hlediska frame-diskriminativního kriteria, tedy žádná topologie modelu není uvažována a pravděpodobnost je počítána jen z modelů výstupní pravděpodobnosti stavů, respektive ze všech složek modelu, kde jsou uvažovány pouze jejich váhy. Pro úplnost ještě dodejme, že suma všech vypočtených $\gamma_{km}^{den}(t)$ je rovna jedné. Na základě vypočtených $\gamma_{km}^{den}(t)$ přepočteme akumulátory $\boldsymbol{\Theta}_{km}^{den}(\boldsymbol{O})$ a $\boldsymbol{\Theta}_{km}^{den}(\boldsymbol{O}^2)$.

Do této metody je integrován I-smoothing (viz kapitola 5.1.7). Podle předběžných experimentů se obecně doporučovaná hodnota $\tau_I = 100$ jeví jako vhodná a bude používána ve všech následujících experimentech pokud nebude uvedeno jinak. Dále bylo navrženo a tetováno hned několik různých variant stabilizací, ze kterých byla ta nejlepší verze vybrána do finální implementace. Ta dosahovala stabilně nejlepších výsledků a nejblíže má ke stabilizaci popsané v [34]. Stabilizační konstanta D je určována nezávisle pro každou složku každého stavu modelu jako

$$D = D_{fact}\gamma_{jm}^{den}. \tag{72}$$

Nastavením D_{fact} určujeme, zda algoritmus bude během jednotlivých iterací více stabilní (pro větší hodnoty) nebo budou změny v modelu radikálnější (pro menší hodnoty). Pro hodnoty D_{fact} menší než 1 nelze zaručit stabilní odhad variancí, proto je hodnota 1 brána jako minimum. V praxi je pak zbytečné volit hodnoty výrazně větší než 2, jelikož se tím trénování výrazně zpomaluje a nakonec model dosahuje obdobné kvality jako pro nižší stabilizační konstanty, jen po mnohem větším počtu iterací. Pro ilustraci vlivu stabilizace byly provedeny experimenty pro $D_{fact} = 1$ a pro $D_{fact} = 2$.

Výsledky jsou uvedeny v tabulce 5. Z těchto výsledků je patrné, že navržená metoda MMI-FD dosahuje velice dobrých výsledků, dochází k redukci WER u všech testovaných modelů, přibližně o 1% až $1,5\%$ absolutně. U menších modelů je však algoritmus méně stabilní a v pozdějších iteracích proces trénování nekonverguje a kvalita natrénovaného modelu se zhoršuje. S větší stabilizační konstantou k tomuto jevu dochází rovněž, nicméně tento efekt není tak výrazný. Je tedy klíčové na rozdíl od ML trénování nastavit vhodný počet reestimací. Vzhledem k velké časové náročnosti výpočtu se obvykle realizuje stejně jen malé množství iterací. U největšího modelu *TrifLarge*, který má optimální počet stavů (z hlediska referenčního ML trénovacícho algoritmu), k poklesu kvality při velkém množství iterací nedochází, je to zřejmě způsobeno tím, že velké množství stavů si navzájem velice konkuruje a není zde takový prostor pro výraznou úpravu parametrů, tedy velké množství stavů má určitou schopnost trénovací proces stabilizovat. Nejvhodnější obecné nastavení podle zpracovaných experimentů je $D_{fact} = 1$ a dvě reestimace. Tímto způsobem dosáhneme na všech modelech významné zlepšení ve velice krátkém čase.

Počet	WER [%]					
iterací	*Monof100*		*TrifSmall*		*TrifLarge*	
D_{fact}	1	2	1	2	1	2
Ref.	19,63	19,63	13,28	13,28	10,14	10,14
1	19,77	18,77	12,21	12,28	9,78	9,78
2	19,13	18,92	12,21	12,71	8,92	9,21
3	18,92	18,99	12,71	12,63	8,85	9,14
4	18,23	19,34	12,63	12,56	8,71	8,92
5	21,34	18,77	13,13	12,56	8,78	8,92
6	20,49	18,84	13,06	12,71	8,49	8,85
7	22,98	19,41	13,35	12,56	8,78	8,49
8	21,98	19,77	13,63	13,06	8,49	8,57
9	24,70	20,06	13,56	13,06	8,57	8,64
10	21,84	20,41	13,70	13,20	8,28	8,57

Tabulka 5: Chyba rozpoznávání MMI-FD pro různé nastavení stabilizace výpočtu

Po zhodnocení předchozích výsledků může být vhodné měnit hodnotu D_{fact} dy-

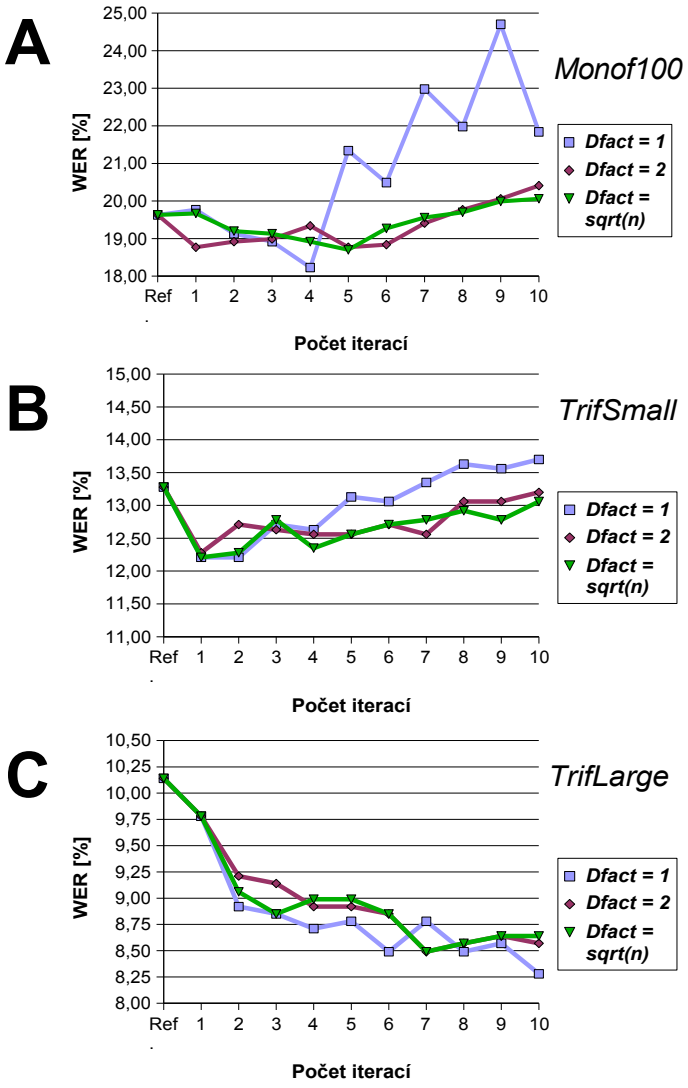
namicky v závislosti na probíhající iteraci. V počáteční jedné či několika iteracích nechat stabilizaci menší a v následujících iteracích ji zvýšit, aby nedocházelo k poklesu kvality natrénovaného modelu pokud by trénování nebylo zastaveno po vhodném počtu iterací. Po několika úvodních experimentech byl program rozšířen o dynamické nastavení, kde $D_{fact} = \sqrt{n}$, kde n je číslo prováděné iterace. Tedy pro první iteraci $D_{fact} = 1$, pro druhou $D_{fact} = \sqrt{2}$ a tak dále. Výsledky takovéhoto postupu jsou v tabulce 6 a pro lepší ilustraci pro všechny modely v grafu 8. Dynamická stabilizace může být vhodným řešením, nicméně pro $D_{fact} = 2$ je dosahováno poměrně podobných výsledků. V praxi je většinou snaha o dosažení kvalitního modelu po co nejmenším počtu iterací. Při velmi malém počtu iterací není dynamická stabilizace nutná.

Počet	WER [%]		
iterací	*Monof100*	*TrifSmall*	*TrifLarge*
Ref.	19,63	13,28	10,14
1	19,67	12,21	9,78
2	19,20	12,28	9,06
3	19,13	12,78	8,85
4	18,92	12,35	8,99
5	18,70	12,56	8,99
6	19,27	12,71	8,85
7	19,56	12,78	8,49
8	19,70	12,92	8,57
9	19,99	12,78	8,64
10	20,06	13,06	8,64

Tabulka 6: Chyba rozpoznávání MMI-FD pro dynamickou stabilizaci

Jelikož je metoda MMI-FD rozšířením nově navržené implementace ML trénování, které bylo testováno a zhodnoceno výše jako lepší než referenční algoritmus, není vhodné použít metodu MMI-FD rovnou na referenční modely, ale bude lepší provést nejprve alespoň jednu iteraci nového ML trénování, která referenční modely z HTK přetrénuje. Vstupní model do MMI-FD bude pak lépe odpovídat použitému algoritmu a výsledky diskriminativního trénování by měly být lepší.

Výsledky tohoto postupu, tedy diskriminativního trénování MMI-FD po jedné



Obrázek 8: *Závislost výsledku MMI-FD na použité stabilizaci. Pro přehlednost postupně pro všechny tři modely. A - model Monof100, B - model TrifSmall a C - model TrifLarge*

nově implementované ML iteraci jsou v tabulce 7 a grafu 9. V tomto experimentu byl nastaven $D_{fact} = 1$ a výsledky je třeba porovnávat s tabulkou 5. Dosažené chybovosti jsou obdobné jako v původní verzi, nicméně výsledky po dvou reestimacích jsou konzistentnější a dosahují dobrých hodnot u všech modelů. Tedy kombinace jedné iterace ML a dvou diskriminativních při nastavení $D_{fact} = 1$ se jeví jako dobrý a rychlý obecný postup pro metodu MMI-FD. Pro úplnost byl ještě proveden stejný experiment i na nejmenším modelu *Monof10*, kde bylo dosaženo po dvou DT iteracích poklesu chyby z $25,41\%$ na $23,34\%$. Zde byl tento počet iterací zároveň optimální.

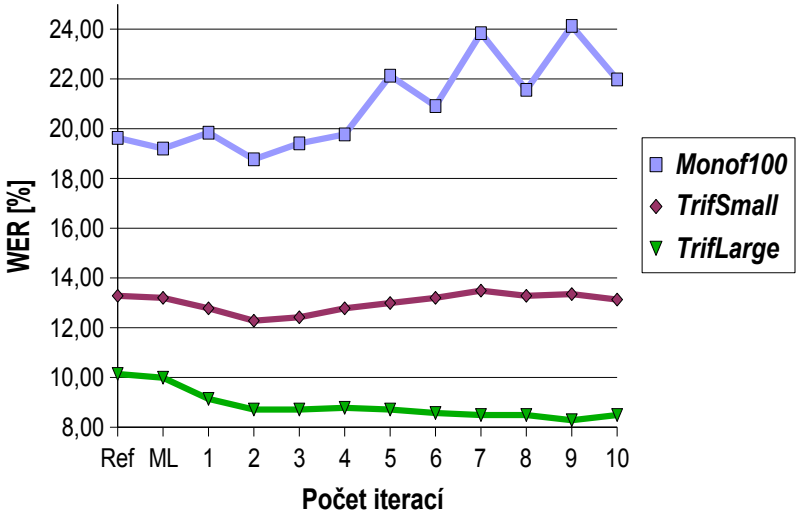
Počet	WER [%]		
iterací	*Monof100*	*TrifSmall*	*TrifLarge*
Ref.	19,63	13,28	10,14
ML	19,20	13,20	9,99
1	19,84	12,78	9,14
2	18,77	12,28	8,71
3	19,41	12,42	8,71
4	19,77	12,78	8,78
5	22,13	12,99	8,71
6	20,91	13,20	8,57
7	23,84	13,49	8,49
8	21,56	13,28	8,49
9	24,13	13,35	8,28
10	21,98	13,13	8,49

Tabulka 7: Chyba rozpoznávání pro MMI-FD po jedné vlastní iteraci ML

Kromě nové metody stabilizace, byl navrhnut i nový postup pro výpočet vah, jelikož původně navržený postup v [26] je problematický (diskutováno v kapitole 5.1.1) a nejčastěji používaná aproximace z [33] zjevně není korektní. Tento navržený postup je velice elegantní a jednoduchý:

$$\bar{c}_{jm} = \frac{\gamma_{jm}^{num}}{\sum_{\hat{m}} \gamma_{j\hat{m}}^{num}} - G \frac{\gamma_{jm}^{num} \gamma_{jm}^{den}}{\sum_{\hat{m}} \gamma_{j\hat{m}}^{num} \sum_{\hat{m}} \gamma_{j\hat{m}}^{den}}, \tag{73}$$

kde γ_{jm} jsou akumulátory dané složky pro čitatel i jmenovatel kriteria, $\sum_{\hat{m}} \gamma_{j\hat{m}}$



Obrázek 9: *Chyba rozpoznávání MMI-FD po jedné iteraci ML.*

je pak suma těchto akumulátorů přes všechny složky daného stavu. Konstantou G lze nastavit míru diskriminace. Pro $G = 0$ se jedná o ML odhad, pro $G \to 1$ o odhad diskriminativní. Jelikož v některých případech by mohlo pro $G = 1$ docházet k nulovým odhadům vah a pak následným numerickým problémům při dalším výpočtu, doporučuji volit G menší než 1, například $0,95$. Po provedení tohoto odhadu je třeba ještě všechny váhy normalizovat na součet rovnající se jedné. U tohoto přístupu nemůže docházet k nesmyslným odhadům záporných vah. Celý přístup je poměrně logický, ML odhad váhy je modifikován podle toho, zda se daná složka relativně více či méně podílí na modelování dat konkurenčních stavů. Složky, které se podílejí na modelování dat konkurenčních stavů více, budou mít nový odhad váhy menší, kdežto složkám, které modelují spíše správně jen data vlastního stavu, bude váha zvýšena.

Po prvních experimentech s novými vahami však bylo patrné, že odhady vah mají na celkovou úspěšnost rozpoznávání v podstatě zanedbatelný vliv, ačkoli diskriminativní kriterium mírně vzrostlo oproti odhadům vah navržených v [26] či v [33].

Dalšími parametry modelu, které je možné diskriminativně odhadovat jsou přechodové pravděpodobnosti. Vzhledem k tomu, že v prostudované literatuře (např. [34]) je vliv úpravy přechodových pravděpodobností považován za zanedbatelný, byl proveden nejprve test citlivosti na nastavení těchto pravděpodobností.

Teprve pokud by se prokázalo, že úprava přechodových pravděpodobností dostatečně ovlivní výsledky rozpoznávání, má cenu diskriminativní přechodové pravděpodobnosti implementovat. V testu citlivosti byly všechny přechodové pravděpodobnosti u modelu *TrifSmall* nastaveny na $p_{i,i} = 0,7$ a $p_{i,i+1} = 0,3$, což se samozřejmě velmi liší od původních hodnot. Výsledek rozpoznávání však byl oproti referenčnímu modelu jen o $0,13\%$ absolutně horší. Což je téměř zanedbatelná hodnota na tak radikální změnu těchto pravděpodobností. Z tohoto důvodu považuji implementaci diskriminativního odhadu přechodových pravděpodobností za prakticky zbytečnou.

Na závěr byly provedeny experimenty zkoumající vliv diskriminativní modifikace jednotlivých parametrů modelu. Byl použit referenční výše doporučený postup trénování, tedy jedna iterace ML a následně dvě iterace MMI-FD se stabilizačním faktorem $D_{fact} = 1$. Byly natrénovány tři verze modelů:

- Pouze ML odhady všech parametrů (značeno ML).
- Diskriminativní odhady středních hodnot (značeno DT-M).
- Diskriminativní odhady středních hodnot a variancí (značeno DT-MV).
- Diskriminativní odhady středních hodnot, variancí a vah složek (značeno DT-MVW).

Modifikované parametry	WER [%]		
	Monof100	*TrifSmall*	*TrifLarge*
Ref.	19,63	13,28	10,14
ML	18,84	13,13	9,49
DT-M	19,41	12,28	8,85
DT-MV	18,70	12,28	8,71
DT-MVW	18,77	12,28	8,71

Tabulka 8: Porovnání vlivu diskriminativní modifikace středních hodnot, variancí a vah složek.

Výsledky experimentu jsou v tabulce 8 a je patrné, že na tomto experimentu se chová monofonový model odlišně od modelů trifonových. U monofonového modelu má klíčový vliv na pokles chyby rozpoznávání modifikace variancí (spolu s

modifikací středních hodnot). U trifonových modelů má klíčový vliv modifikace středních hodnot a modifikace variancí ani vah složek již významně nepřispívá k poklesu chyby rozpoznávání.

Další experiment, který byl proveden zkoumal vliv nastavení tzv. I-smootgingu (viz kapitola 5.1.7). Výše již bylo zmíněno, že toto vylepšení bylo již integrováno do základní verze MMI-FD trénování a podle předběžných experimentů byla konstanta I-smoothingu $\tau_I = 100$. V samotné metodě však bylo provedeno velké množství úprav a změn, takže výsledky předběžných experimentů již nemusí být relevantní. Proto byl proveden experiment zkoumající vliv nastavení I-smoothingu pro doporučovaný postup trénování MMI-FD.

Hodnota	WER [%]		
τ_I	*Monof100*	*TrifSmall*	*TrifLarge*
0	18,77	12,92	9,28
50	18,63	12,42	8,64
100	18,77	12,28	8,71
150	18,99	12,28	8,78
200	18,99	12,35	8,71

Tabulka 9: Porovnání vlivu nastavení konstanty I-smoothingu na chybu rozpoznávání.

Z výsledků uvedených v tabulce 9 je patrné, že vhodné nastavení stále odpovídá předběžným experimentům, tedy za vhodné lze považovat hodnoty od 50 do 150. Navíc se ukazuje, že pro referenční postup a metodu MMI-FD je nastavení konstanty poměrně robustní a není velký rozdíl mezi jednotlivými nastaveními s výjimkou úplného vypnutí I-smoothingu pro $\tau_I = 0$.

6.1.7 Diskriminativní trénování na základě kritéria MMI

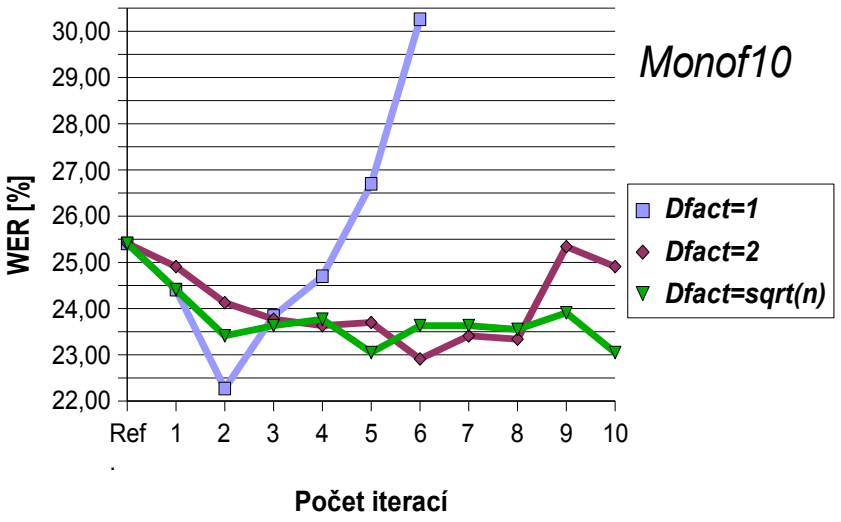
Základní metoda MMI, tak jak je popsána v kapitole 5.1, vyhodnocuje jmenovatele kriteria buď přímým výpočtem všech možných posloupností (hypotéz) nebo častěji aproximací pomocí mřížek či výběrem omezeného počtu nejpravděpodobnějších hypotéz. Takovýto výpočet statistik bere v úvahu konkrétní slovník i použitý akustický model. Z toho plyne fakt, že slovník a jazykový

model použitý při diskriminativním trénování bude určitým způsobem zanesen i v natrénovaných parametrech akustického modelu. Pokud je k dispozici dostatečné množství dat a jazykový model velice dobře odpovídá budoucímu testovacímu, stává se tento fakt výhodou. V případě menšího množství trénovacích dat, pak může docházet k horší robustnosti natrénovaného modelu. V případě výrazně odlišného trénovacího slovníku od slovníku testovacího může docházet i ke zhoršení dosahovaných výsledků po diskriminativním tréninku. Oproti tomu výše testovaná metoda MMI-FD je robustnější a v úlohách, kde není dopředu možné odhadnout testovací slovník, či pro detekci zatím neurčených klíčových slov, je metoda MMI-FD zřejmě vhodnější.

Pro praktickou implementaci MMI metody byl použit již vytvořený kód používaný na naší katedře. Ten byl upraven Alešem Pražákem (kterému tímto děkuji) tak, aby umožňoval po vytvoření slovní mřížky algoritmem Forward-Backward spočítat potřebné statistiky. Tento kód umožňuje používat pouze zerogramové jazykové modely. Výpočet statistik čitatele MMI kritéria probíhá stejným algoritmem jako u výše uvedených metod ML a MMI-FD.

Počet	WER [%]		
iterací	*Monof10*		
D_{fact}	1	2	$\sqrt{(n)}$
Ref.	25,41	25,41	25,41
1	24,41	24,91	24,41
2	22,27	24,13	23,41
3	23,84	23,77	23,63
4	24,70	23,63	23,77
5	27,70	23,70	23,05
6	30,26	22,91	23,63
7	-	23,41	23,63
8	-	23,34	23,55
9	-	25,34	23,91
10	-	24,91	23,05

Tabulka 10: Chyba rozpoznávání MMI pro různé nastavení stabilizace výpočtu na modelu *Monof10*



Obrázek 10: *Chyba rozpoznávání MMI pro různé nastavení stabilizace výpočtu na modelu* Monof10.

Výsledky této metody pro různé typy stabilizace výpočtu a nejmenší model *Model10* jsou uvedeny v tabulce 10 a grafu 10. Zde se ukazuje, ze metoda dosahuje na tomto modelu lepšího poklesu chyby než MMI-FD (WER $23,34\%$ po dvou iteracích s $D_{fact} = 1$), ačkoli při silnějších stabilizacích je metoda srovnatelná. Výsledky na větších trifonových modelech budou uvedeny níže (v tabulce 11) spolu s kombinovanou metodou, která bude rovněž uvedena níže.

Metoda MMI tak i její modifikace MMI-FD mají mnoho společného. Podle předběžných experimentů dosahuje metoda MMI lepších výsledků na malých modelech a naopak MMI-FD lépe zobecňuje a dosahuje tak lepších výsledků na modelech komplexnějších. Tyto zkušenosti vedly k návrhu nové kombinované metody, kterou označíme MMI+MMI-FD. Ta spočívá v kombinaci aposteriorní pravděpodobnosti jmenovatele MMI kriteria. Kombinace je prováděna v logaritmické doméně, ve které jsou pravděpodobnosti obvykle počítány. Kombinace pak vzniká podle vztahu

$$\log \hat{\gamma}_j(t) = \alpha \log \gamma_j^{MMI}(t) + (1-\alpha) \log \gamma_j^{MMI-FD}(t), \tag{74}$$

kde $\log \gamma_j$ je logaritmus aposteriorní pravděpodobnosti, že se Markovův proces nachází v čase t ve stavu j a α je koeficient určující váhu jednotlivých metod. Ještě poznamenejme, že po výpočtu je třeba znovu aposteriorní pravděpodobnosti

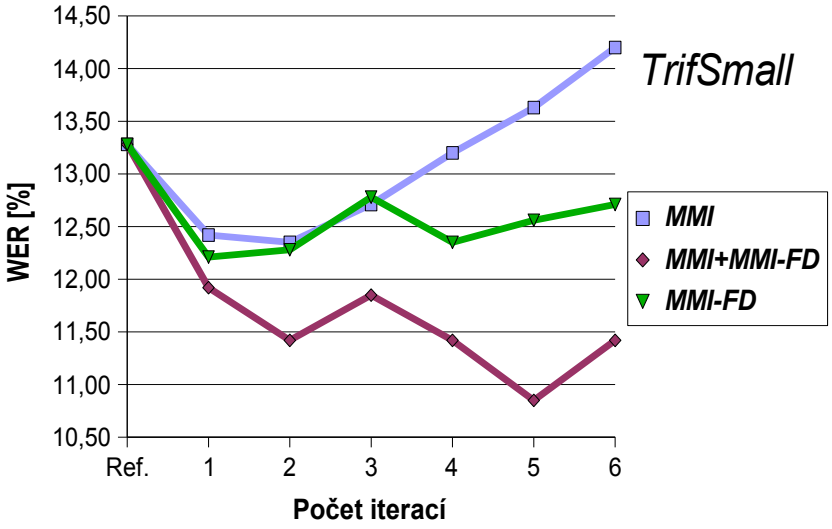
znormovat. Z provedených experimentů vyplývá, že nastavení kombinační váhy je poměrně robustní, pro hodnoty $0,25$ až $0,75$ je dosahováno porovnatelných výsledků. U nižších či vyšších hodnot už výsledky konvergují k jednotlivým kombinovaným metodám. Částečně bylo možné pozorovat závislost optimálního nastavení na prováděné iteraci, nicméně návrh obecně použitelné metody pro nastavení vhodného koeficientu závislého na prováděné iteraci nedosahoval obecně lepších výsledků než statický koeficient. Vzhledem k robustnosti nastavení je tedy obecně vhodné nastavení $\alpha = 0,5$.

Výsledky jak pro samotnou metodu MMI tak pro metodu kombinovanou MMI+MMI-FD s koeficientem $\alpha = 0,5$ jsou uvedeny v tabulce 11 a grafech 11 a 12. V tomto případě byly experimenty prováděny na trifonových modelech *TrifSmall* a *TrifLarge*. Byla použitá dynamická stabilizace s faktorem $D_{fact} = \sqrt{n}$. Uvedené výsledky potvrzují výše uvedené závěry z předběžných experimentů. U menšího modelu *TrifSmall* dosahují metody MMI a MMI-FD porovnatelných výsledků, avšak kombinovaná metoda MMI+MMI-FD dosahuje výsledků výrazně lepších. Dochází zde k poklesu chyby o přibližně jedno procento absolutně. U většího modelu *TrifLarge* je však situace odlišná. Metoda MMI ani metoda kombinovaná nedosahuje tak dobrých výsledků jako MMI-FD. Zde se tedy projevuje horší schopnost metody MMI zobecňovat na testovací data u komplexních modelů.

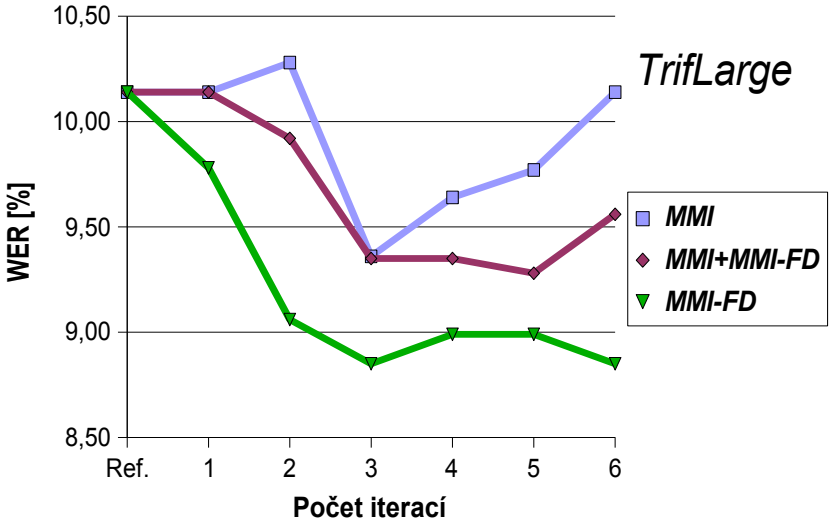
Počet iterací	WER [%]					
	TrifSmall			*TrifLarge*		
	MMI	komb.	MMI-FD	MMI	komb.	MMI-FD
Ref.	13,28	13,28	13,28	10,14	10,14	10,14
1	12,42	11,92	12,21	10,14	10,14	9,78
2	12,35	11,42	12,28	10,28	9,92	9,06
3	12,71	11,85	12,78	9,36	9,35	8,85
4	13,20	11,42	12,35	9,64	9,35	8,99
5	13,63	10,85	12,56	9,77	9,28	8,99
6	14,20	11,42	12,71	10,14	9,56	8,85

Tabulka 11: Chyba rozpoznávání MMI, MMI-FD a kombinace MMI+MMI-FD pro modely *TrifSmall* a *TrifLarge*

Na závěr je třeba podotknout, že použitý algoritmus pro výpočet slovních mřížek nepodporuje jiné než zero-gramové jazykové modely a jeho implementace je



Obrázek 11: *Chyba rozpoznávání MMI, MMI+MMI-FD a MMI-FD pro model TrifSmall.*



Obrázek 12: *Chyba rozpoznávání MMI, MMI+MMI-FD a MMI-FD pro model TrifLarge.*

s narůstajícím slovníkem extrémně náročná na paměť. S 3 GB limitem alokované paměti je možno zpracovávat úlohy, kde trénovací data jsou rozdělena po větách a slovník nepřevyšuje 10 tisíc slov. Pro úlohy s většími slovníky lze použít pouze metodu MMI-FD.

6.1.8 Metoda založená na diskriminaci trifonů

K návrhu této metody vedly zkušenosti získané výše uvedenými experimenty. Cílem bylo navrhnout takovou metodu, která bude rovněž na slovníku a jazykovém modelu nezávislá (jako MMI-FD), ale zároveň bude odpovídat topologii akustického modelu - tedy faktu, že promluva je modelována posloupností fonémů, které jsou pak modelovány většinou pomocí trifonů. Takto vznikla metoda založená na diskriminaci trifonů, která bude nadále označována MMI-TF. Tato metoda je odvozena od metody MMI, používá kritérium ve stejném tvaru. Jen jmenovatel kriteria je vypočítáván jiným způsobem. V základní metodě MMI je ve jmenovateli suma pravděpodobností všech možných posloupností slov. V metodě MMI-TF je to suma pravděpodobností všech možných posloupností fonémů. Tato suma je podobně jako u metody MMI vypočítávána pomocí generování mřížky, jednotlivé hrany v mřížce však nereprezentují slova, ale pouze konkrétní fonémy. Jedná se tedy o fonémovou mřížku. Na tuto mřížku je následně aplikován forward-backward algoritmus pro výpočet aposteriorních pravděpodobností jednotlivých stavů. Z těchto aposteriorních pravděpodobností jsou spočítány nové odhady středních hodnot a variancí už stejným způsobem jako u metody MMI či MMI-FD.

Tato metoda byla navržena tak, že generování fonémové mřížky prakticky neprobíhá, ale vše je již integrováno do samotného forward-backward algoritmu. Pro realizovatelnost výpočtu je třeba aplikovat prohledávání s prořezáváním, jinak by vzhledem k extrémním paměťovým a výpočetním nárokům forward-backward algoritmus nebyl vůbec proveditelný. Toto prořezávání je prováděno ve třech úrovních: u pravděpodobností jednotlivých stavů, pravděpodobností přechodu a při ukládání pravděpodobností mezi dopřednou a zpětnou fází forward-backward algoritmu. U všech je použitý stejný práh, který určuje minimální pravděpodobnost, kterou musí vyhodnocovaný stav v daném čase převyšovat, jinak bude označen za neaktivní. Vzhledem k tomu, že je v každém čase pravděpodobnost normalizována, lze tento práh nastavit fixně. V případě, že tímto způsobem nemá forward-backward algoritmus žádné řešení (neprojde žádná hypotéza) algoritmus se pro danou promluvu opakuje s benevolentnějším prahem a to tak dlouho dokud není nalezeno řešení. Kromě výše prahu prořezávání je třeba rovněž určit tzv. penaltu vložení. Ta je totožná s penaltou vložení používanou při rozpoznávání

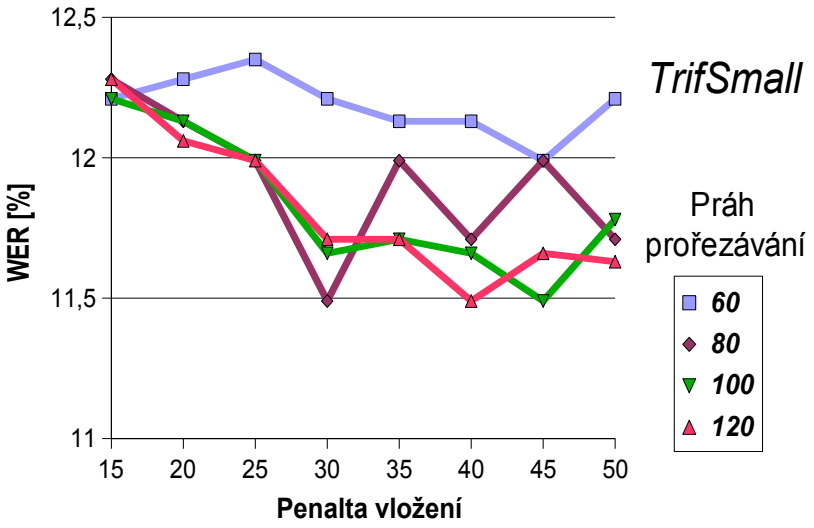
slov, jen zde je aplikována na jednotlivé fonémy a tedy její optimální hodnota bude nižší.

Tento postup byl testován na modelech *TrifSmall* a *TrifLarge*. Z předběžných experimentů bylo patrné, že nejlepších výsledků tato metoda dosahovala hned v první iteraci, v dalších pak již docházelo k růstu chyby rozpoznávání a také forward-backward algoritmus začínal mít problémy s generováním mřížek. Tedy použitá stabilizace (shodná s metodami MMI a MMI-FD s $D_{fact} = 1$) není pro tuto metodu příliš vhodná. Práh prořezávání (v záporné logaritmické doméně, ve které je výpočet implementován) bylo vhodné volit mezi 60 a 120. Pro hodnoty vyšší, výpočet dosahoval obdobných výsledků jako pro práh 120, jen časová a paměťová náročnost rostla. Pro práh 60 a nižší už rostla chyba rozpoznávání a pro velké množství promluv musel být stejně pro selhávání při forward-backward algoritmu hodnota prahu zvyšována. Výsledky pro model *TrifSmall* jsou v grafu 13 a pro model *TrifLarge* v grafu 14. Z výsledků je patrné, že pro téměř všechna nastavení je chyba nižší než u původního referenčního modelu (13, 28% pro *TrifSmall* a 10, 14% pro *TrifLarge*). Hodnota prahu pro prořezávání by měla být 100 a více. Nastavení penalty vložení má poměrně široký interval, vhodné hodnoty se pohybují mezi 30 a 50. U modelu *TrifSmall* dochází k výraznějšímu snížení chyby než u metody MMI-FD, nicméně u modelu *TrifLarge* je tomu naopak. Lze tedy opět tvrdit, stejně jako u metody MMI, že tato metoda hůře zobecňuje, než metoda MMI-FD, ale u méně komplexnějších modelů dosahuje lepších výsledků. Ačkoli je tato metoda velmi odlišná, její výsledky odpovídají přibližně metodě MMI+MMI-FD, ovšem výsledný model byl dosažen jen jednou iterací algoritmu.

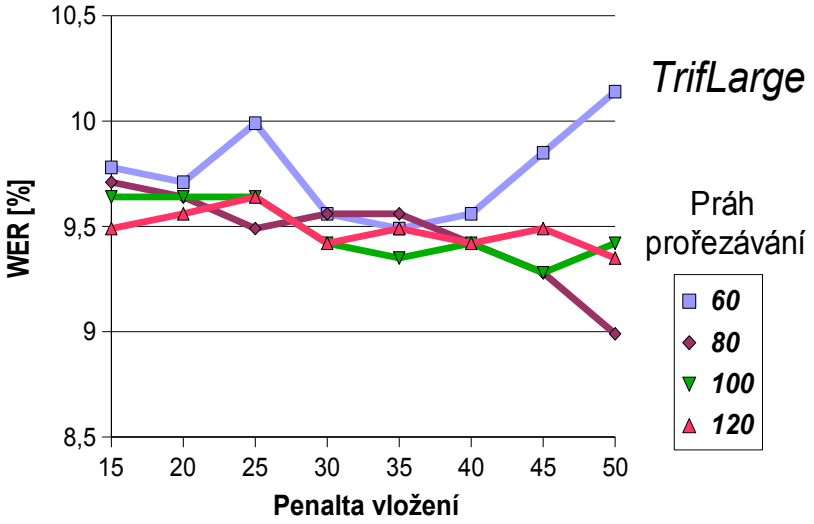
6.1.9 Shrnutí experimentů na korpusu UWB S01

Na tomto korpusu byla provedena celá řada experimentů a vyvinuto několik metod a jejich modifikací. Rovněž bylo vyladěno i nastavení mnoha parametrů na tomto korpusu. Než budou tyto metody a nastavení testovány na dalších korpusech a úlohách, aby se prokázala jejich obecnost a přenositelnost, je třeba shrnout dosažené výsledky.

V kapitole 6.1.3 bylo popsáno trénování a výsledky referenčních modelů, ze kterých pak jednotlivé diskriminativní metody vycházely. V kapitole 6.1.5 pak byla testována vlastní verze Baumova-Welchova a forward-backward algoritmu, které jsou pak použity i v metodách diskriminativního trénování. Tato verze dosahovala u některých modelů lepších výsledků než referenční implementace v HTK toolboxu. Doba jedné iterace dosahovala až 1/75 reálného času trénovacích nahrávek. Dále byla v kapitole 6.1.6 představena nově navržená metoda MMI-FD. Byl navržen



Obrázek 13: *Chyba rozpoznávání MMI-TF pro model TrifSmall.*



Obrázek 14: *Chyba rozpoznávání MMI-TF pro model TrifLarge.*

a úspěšně otestován nově navržený algoritmus stabilizace výpočtu včetně dynamické stabilizace a diskutován vliv jeho nastavení. Dále byl nově navržen postup pro diskriminativní výpočet vah, který je na rozdíl od nejpoužívanější aproximace navržené v [33] korektní a dosahuje mírného zlepšení diskriminativního kriteria. V kapitole 6.1.7 pak byla popsána vlastní modifikace metody MMI a navržena kombinace aposteriorních pravděpodobností s metodou MMI-FD, aby se tak zkombinovaly výhody obou metod. Tedy dobré výsledky na méně komplexních modelech z metody MMI se schopností zobecňovat z metody MMI-FD. Na středně komplexním modelu pak bylo dosaženo touto kombinací nejlepšího výsledku. Na největším modelu však stále dosahuje nejmenší chyby metoda MMI-FD, která se jeví pro velké modely jako nejvhodnější. V kapitole 6.1.8 byla pak navržena a implementována metoda MMI-TF založená na diskriminaci trifonů. Ta dosahuje na méně komplexních modelech velice dobrých výsledků hned po první iteraci. Na nejkomplexnějším modelu se však opět projevuje nedostatečná schopnost této metody zobecňovat na testovací data.

Nejobecnější metodou, která navíc není závislá na slovníku a jazykovém modelu, je metoda MMI-FD. Vhodné obecné nastavení bylo stanoveno na dvě iterace algoritmu se stabilizačním faktorem $D_{fact} = 1$, kterým předchází jedna iterace nově implementovaného ML algoritmu, která upraví vstupní model trénovaný jinou implementací ML metody. Při tomto nastavení je dosahováno relativního poklesu chyby o 4% až 14%. Nejlepších výsledků je dosahováno u nejkomplexnějšího modelu, kde je rovněž dosahováno nejmenší chyby i absolutně. To je velice zajímavé zjištění, jelikož většina diskriminativních metod právě na nejkomplexnějších modelech dosahuje horších výsledků než na modelech méně komplexních. Optimalizace této metody spolu s vhodným nastavením, které poskytuje dobrý model již po dvou iteracích, dosahuje velmi krátkých časů potřebných pro trénování. Pro největší testovaný model celé dikriminativní přetrénování trvá přibližně $0,03$ násobek reálného času trénovacích nahrávek včetně úvodní jedné ML iterace.

6.2 Experimenty na úloze automatického titulkování parlamentních přenosů

Na naší katedře je v současné době řešen projekt "'Eliminace jazykových bariér handicapovaných diváků České televize"' (MŠMT 2C06020), kde jako pilotní úloha bylo uvedeno do reálného provozu automatické titulkování parlamentních přenosů [5]. Jedná se o softwarový balík, který dokáže v reálném čase ze zvuku přenášeného z právě probíhajícího zasedání parlamentu ČR generovat skryté titulky, které si

mohou neslyšící zapnout na svém televizním přijímači. Jedná se o velice náročnou úlohu, jelikož jde o reálné nahrávky s velmi velkým slovníkem, navíc je třeba generovat titulky s minimálním zpožděním. Řešitelnost této úlohy je na hranici současného vývoje jednotlivých metod rozpoznávání řeči a zároveň na hranici výkonu současných počítačů. V rámci této práce, ale i řešení daného projektu, byly aplikovány výše popsané metody diskriminativního trénování na tuto úlohu.

6.2.1 Popis úlohy, trénování a testování akustických modelů

Pro řešení této úlohy bylo zaznamenána a precizně anotovány trénovací nahrávky (záznamy parlamentních přenosů) o celkové délce 100 hodin. Tyto nahrávky byly zaznamenány se vzorkovací frekvencí 44.1 kHz a ukládány s 16-ti bitovou přesností. Tento vstupní akustický signál byl zpracován do posloupnosti příznakových vektoru metodou PLP s 27-mi filtry a 12-ti koeficienty. K výsledným vektorům byly připojeny delta a akcelerační dynamické příznaky. Celková dimenze výsledných příznakových vektorů byla 36. Vektory příznaků jsou generovány s posunem 10 milisekund, tedy rychlostí 100 příznakových vektorů za sekundu.

Výchozí akustický model byl trénován pomocí HTK toolboxu. Jednalo se o třístavový HMM se shlukovanými trifony. Celkový počet stavů a počet složek na stav byl optimalizován experimentálně. Optimální počet stavů, vzhledem k použité metodě trénování, byl 5385 stavů s 8-mi složkami na stav.

Jazykový model byl trénován z normalizovaných přepisů parlamentních jednání z let 2004 až 2008. Celkem bylo v dané době k dispozici 23 milionů slov trénovacího textu (přibližně 130 MB textu). Z tohoto textu byl natrénován bigramový jazykový model, málo frekventovaná slova byla ručně zkontrolována, zda neobsahují překlepy. Slova, která se vyskytla v trénovacím textu pouze jednou byla z jazykového modelu i slovníku vyjmuta, tím se výrazně snížila velikost slovníku, zároveň však narostl počet slov, která ve slovníku nejsou (tzv. OOV) na $1,2\%$. Slovník obsahoval 131 tisíc výslovností slov, 118 tisíc slov/unigramů, bigramů pak bylo $8,8$ milionu. Pro modelování jmen poslanců, senátorů, vládu a prezidenta byly použity třídy.

Pro důkladné otestování metod pro akustické modelování byly připraveny dva testy.

- *Zerogram* - Jednalo se o test se zerogramovým akustickým modelem ze slov vyskytujících se pouze v testovacím záznamu. S tímto nejjednodušším jazykovým modelem se naplno projeví kvalita akustického mod-

elu. Byla testována 1 hodina parlamentního přenosu, který se nevyskytuje v trénovacích datech. Záznam byl rozdělen na kratší úseky po jednotlivých mluvčích. Celkem jazykový model obsahoval 4003 slov s celkem 4920-ti výslovnostmi.

- *Bigram* - Zde byl použit třídní bigramový jazykový model popsaný výše. Výsledky toho testu se co nejvíce blíží výsledkům při reálném nasazení systému. Rozpoznávané promluvy jsou identické s předchozím testem.

6.2.2 Výsledky testovaných metod diskriminativního trénování

Nejprve byla testována metoda MMI-FD se dvěma různými stabilizačními faktory. Výsledky jsou uvedeny v tabulce 12 a grafech 15 a 16. U obou testů došlo k snížení chyby rozpoznávání, u testu *Zerogram*, který je citlivější na akustickou část systému, je pokles téměř o 3% absolutně. U testu *Bigram* je pokles chyby necelé jedno procento absolutně. Je tedy zřejmé, že v úlohách s velmi velkým slovníkem, je důležijejší spíše jazyková část systému než část akustická. Z výsledků dále vyplývá, že nastavení metody MMI-FD doporučené v kapitole 6.1.9 je velmi dobře přenostilené a dosahuje velmi dobrých výsledků i na velmi odlišné úloze. Co se týče rychlosti, tak kompletní dikriminativní trénování (jedna iterace ML a dvě iterace MMI-FD) trvalo na tomto modelu přibližně $0,05$ až $0,1$ reálného času trénovacích nahrávek v závislosti na použitém počítači.

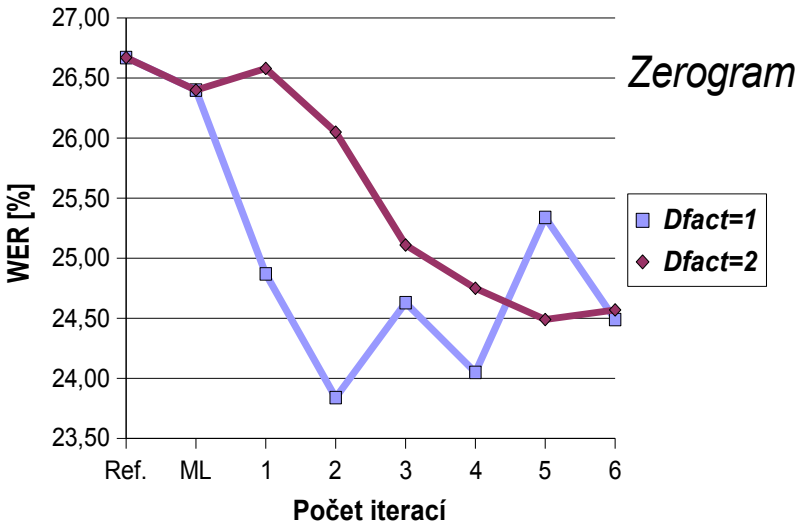
Pro otestování deklarované dobré schopnosti metody MMI-FD zobecňovat byl natrénován výrazně větší referenční model. Tento model měl 12135 stavů, tedy více než dvojnásobek referenčního modelu, který byl testován předtím. Každý stav měl rovněž 8 složek. Oproti předchozímu referenčnímu modelu chyba rozpoznávání mírně vzrostla na $27,09\%$ na testu *Zerogram* a na $20,27\%$ na testu *Bigram*. Po doporučeném postupu - tedy jedna reestimace ML a dvě MMI-FD - klesla chyba rozpoznávání na $24,94\%$ na testu *Zerogram* a na $19,68\%$ na testu *Bigram*. Pokles chyby byl o něco menší než u menšího modelu. Nicméně se tímto opět potvrdila velmi dobrá shopnost metody MMI-FD zobecňovat.

Z dalších metod byla zkoušena MMI, ovšem poskytnutá implementace (viz kapitola 6.1.7) generování mřížek a forward-backward algoritmu, nebyla pro velké slovníky vhodná a výpočet byl z důvodů extrémních paměťových i výpočetních nároků nerealizovatelný.

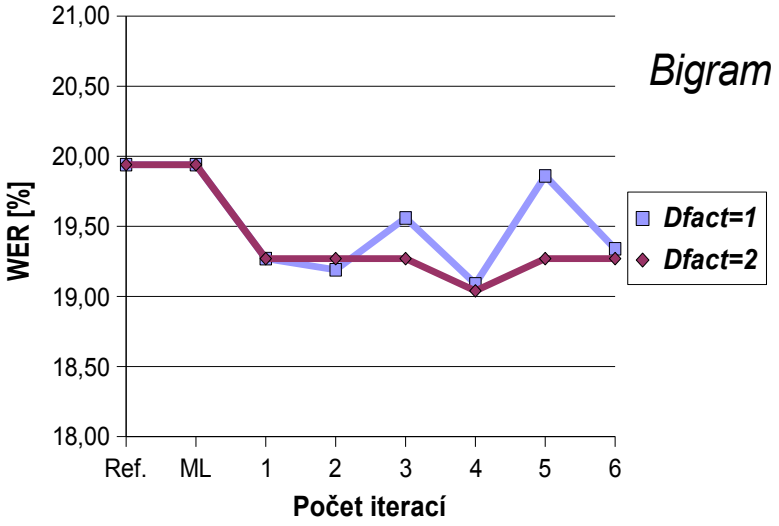
Metoda MMI-TF (viz kapitola 6.1.8) byla navržena a implementována tak, aby bylo možné nasazení na velkých úlohách. Byla provedena jedna iterace trénování (na modelu s 5385-ti stavy) a provedeny oba testy. Na testu *Zerogram* klesla chyba

Počet iterací	WER [%]			
	Zerogram		*Bigram*	
D_{fact}	1	2	1	2
Ref.	26,67	26,67	19,94	19,94
ML	26,40	26,40	19,94	19,94
1	24,87	26,58	19,27	19,27
2	23,84	26,05	19,19	19,27
3	24,63	25,11	19,56	19,27
4	24,05	24,75	19,09	19,04
5	25,34	24,49	19,86	19,27
6	24,49	24,57	19,34	19,27

Tabulka 12: Chyba rozpoznávání MMI-FD na datech z parlamentu



Obrázek 15: *Chyba rozpoznávání MMI-FD - Zerogram.*



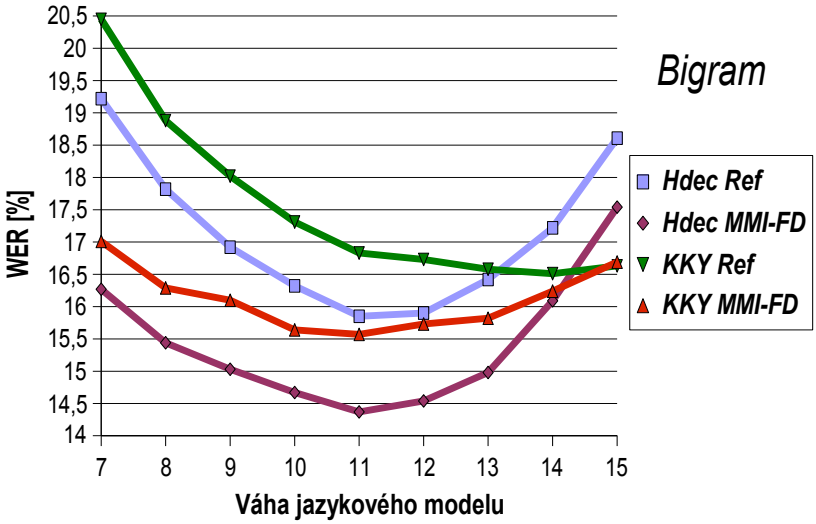
Obrázek 16: *Chyba rozpoznávání MMI-FD - Bigram.*

rozpoznávání na $25,91\%$ a na testu *Bigram* na $19,37\%$. Tedy aplikace této metody vede ke snížení chyby rozpoznávání, nicméně oproti metodě MMI-FD dosahuje MMI-TF mírně horších výsledků.

6.2.3 Porovnání LVCSR dekodérů

Kromě interního rozpoznávacího systému pro úlohy s velkým slovníkem (angl. Large Vocabulary Continuous Speech Recognition - LVCSR), byl testován i dekodér *HDecode* z HTK toolboxu verze 3.4 [23]. Jelikož byl v poslední době katedrální rozpozpoznávací systém značně vylepšen, byla použita jeho nejnovější verze. Tedy uvedené výsledky nejsou kompatibilní s výsledky uvedenými v předchozí podkapitole a dosahovaná chyba rozpoznávání je obecně nižší. Oba rozpoznávací systémy byly testovány na totožné sadě nahrávek i jazykovém modelu (shodné s testem *Bigram*) pro referenční model s 5385-ti stavy a jeho diskriminativní verzi (metoda MMI-FD). Výsledky pro různé nastavení váhy jazykového modelu jsou v grafu 17. Z výsledků je patrné, že systém *HDecode* (značený v grafu *Hdec*) dosáhl lepších výsledků, zejména u diskriminativního modelu, kde byl rozdíl přibližně 1% absolutně. Tím pádem i rozdíl mezi referenčním modelem a diskriminativně trénovaným se zvětšil a metoda MMI-FD snížila chybu rozpoznávání u dekodéru *HDecode* z $15,85\%$ na $14,37\%$ tedy o $1,48\%$ abso-

lutně což je 9, 3% relativně. Je tedy patrné, že hodnocení úspěšnosti metod akustického modelování závisí i na použitém rozpoznávacím systému. Vyšší chybovost katedrálního dekodéru je zřejmě způsobena četnými optimalizacemi na rychlost výpočtu, jelikož tento dekodér je primárně určen pro aplikace běžící v reálném čase. Zajímavá je také u tohoto dekodéru závislost výsledků na nastavené váze jazykového modelu. Pro ML model vychází optimální nastavení na 14, kdežto u MMI-FD modelu se nastavení značně liší a nejlepších výsledků je dosahováno při váze 11.



Obrázek 17: *Porovnání dekodérů KKY a HDecode.*

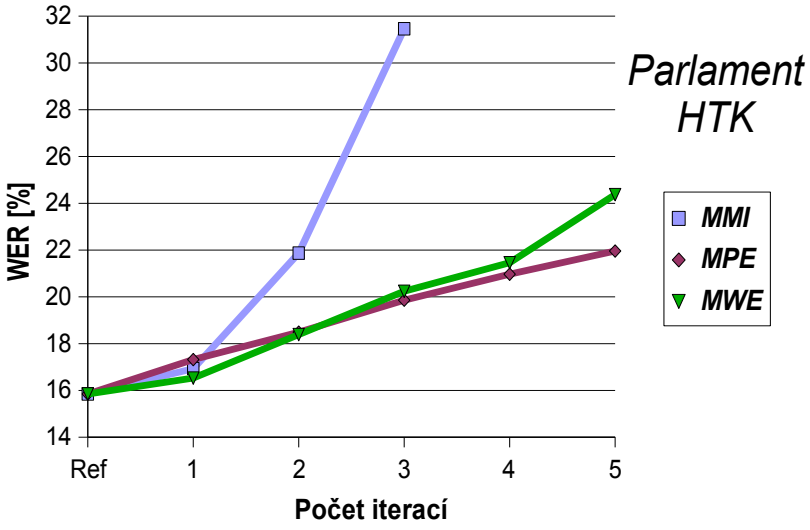
6.2.4 Testování metod MMI, MWE a MPE z HTK

Bylo by dobré, pokud by vlastní metody bylo možné přímo porovnat s některou z implementovaných metod diskriminativního trénování používanou ve světě. Toto porovnání by bylo jistě cennější než jen nepřímé porovnávání schopností jednotlivých metod na odlišných úlohách. Pro toto porovnání jsou v toolboxu HTK 3.4 k dispozici metody MMI, MWE a MPE [23]. Bohužel tyto metody byly do HTK integrovány až v poslední verzi a jejich dokumentace není příliš podrobná. Ani po velkém úsilí nebylo možné na vlastních datech diskriminativní trénování v HTK zprovoznit, běh programu často končil výjimkou, která nebyla programově nikterak ošetřena a bývala zachycena až operačním systémem. To velice znesnadňovalo

přípravu vstupních dat, jelikož jejich formát není v dokumentaci přesně definován a program, který se po spuštění zhroutí, aniž by upozornil, kde může být problém, je takřka nepoužitelný. Po komunikaci přímo s universitou v Cambridgi, kde je HTK vyvíjen, my byl doporučen pouze příklad použití diskriminativního trénování na *Resource Management* korpusu, jelikož pro další vývoj a servis toolboxu HTK nemají dostatek finančních prostředků (HTK toolbox byl primárně vyvinut jen v rámci dvou projektů od Microsoftu a Darpy). Tedy nebylo možné obdržet konkrétní technickou podporu při zprovozňování HTK na vlastních datech. Následně byl tedy zakoupen *Resource Management* korpus a uvedené příklady použití zprovozněny (podrobnější experimenty na tomto korpusu jsou popsány v kapitole 6.3.1). Podle tohoto příkladu byl postupně připravován korpus z parlamentních přenosů pro použití s HTK. V první řadě bylo třeba přeložit HTK toolbox s podporou velkých slovníků. Standardně je používán datový typ *unsigned short*, který limituje velikost slovníku jen na 65 tisíc. Tento limit na většinu úloh v anglickém jazyce dostačuje, pro rozpoznávání českého jazyka jsou potřeba slovníky řádově větší, tento limit by byl dostatečný pouze pro některé specializované úlohy. Dále byla největším problémem rozdílná topologie neřečových událostí v akustickém modelu. Z tohoto důvodu nebylo možné vygenerovat slovní mřížky, jelikož implementovaný forward-backward algoritmus na této topologii selhával. Zde byl nutný přímý zásah do zdrojových kódů HTK, aby byl problém odstraněn a mřížky bylo možné vygenerovat. Tedy po velmi náročném úsilí bylo možné diskriminativně trénovat modely v HTK na úloze parlamentu.

Výsledky natrénovaných modelů však vůbec nebyly dobré, jak ukazuje graf 18. Ani jedna z testovaných metod s doporučeným nastavením nevedla ke snížení chyby rozpoznávání, naopak, každá další iterace vedla ke zhoršení výsledku. Zřejmě v programu stále někde docházelo k problémům, ale nebylo možné zjistit, kde by k těmto problémům mohlo docházet. Tedy referenční implementace diskriminativního trénovaní použitelná na všechna vlastní data nebyla pro porovnání k dispozici. Porovnat vlastní metody s metodami v HTK lze tedy jen na korpusu *Resource Management* (v kapitole 6.3.1). Tato úloha je však již na dnešní poměry velice jednoduchá (slovník 1000 slov, malé množství dat).

Na závěr je třeba poznamenat, že kritika se netýká celého toolboxu HTK, ale jen části pro diskriminativní trénování ve verzi 3.4. HTK toolbox je jinak velice kvalitní a dobře použitelný, na naší katedře je používán pro trénování úvodních modelů podle kritéria ML.



Obrázek 18: *Výsledky diskriminativních metod implementovaných v HTK.*

6.3 Testy na dalších korpusech

6.3.1 Testy na Resource Management korpusu

Tento korpus byl vyvinut americkou agenturou DARPA (angl. Defense Advanced Research Project Agency) a publikován v roce 1988. Jedná se o korpus anglických řečových nahrávek, který slouží pro vývoj a porovnávání systémů pro automatické rozpoznávání řeči. Korpus obsahuje čtené promluvy 160-ti řečníků z různých částí USA. Data byla nahrávána 16 kHz s 16-ti bitovým rozlišením přes mikrofon Sennheiser HMD-414. Pro trénování je vyhrazeno 4000 vět, které mají celkovou délku $3,8$ hodin. Pro testování je vyhrazeno několik testovacích sad. V následujícím experimentu byla používána testovací sada označená *feb98*, která obsahuje 400 vět (přibližně 20 minut řeči). Jazykový model je bigramový založený na pravidlech.

V ukázkovém příkladu pro použití HTK diskriminativního trénování byly i skripty pro zpracování nahrávek a trénování referenčních modelů. Pro zpracování nahrávek byla použita metoda MFCC s delta a akceleračními příznaky, celková dimenze příznakových vektorů pak byla 39. Testovány byly dva trifonové modely: jednosložkový a 6-ti složkový HMM, oba měly 1532 stavů . Všechny výsledky jsou uvedeny v souhrnné tabulce 13. Na jednosložkovém modelu je chyba rozpoznávání

WER [%]		
Metoda	1 složka	6 složek
Ref.	6,48	2,81
HTK-MMI	4,02	2,81
HTK-MPE	3,98	2,89
HTK-MWE	4,14	2,93
MMI	4,27	2,69
MMI-FD	4,74	2,77
MMI+MMI-FD	4,34	2,69
MMI-TF	4,18	2,81

Tabulka 13: Chyba rozpoznávání HTK i vlastních metod na datech RM korpusu

všemi metodami redukována poměrně výrazně o 2 až 2, 5%. Jednosložkové modely se také nejsnadněji diskriminují, jelikož každý stav modelu přímo odpovídá jednomu konkrétnímu normálnímu rozpožení. Avšak ani nejlepší výsledek dosažený na jednosložkovém modelu nemá menší chybu, než referenční model se 6-ti složkami. Na tomto modelu se pak porařilo snížit chybu jen nepatrně. Na jednosložkovém modelu nejlépe dopadla HTK implementace metody MPE, nejhůře pak vlastní implementace metody MMI-FD. Na 6-ti složkovém modelu nejlepších výsledků dosáhla vlastní implementace MMI, jak samostatně, tak i v kombinaci s MMI-FD. Nejhorších výsledků dosahuje HTK implementace MWE. Výsledky na 6-ti složkovém modelu se však příliž od sebe neliší.

6.3.2 Testy na rozšířeném korpusu UWB S01

Jak bylo popsáno v kapitole 6.1.1 jedná se korpus postupně nahrávaný a anotovaný na naší katedře. Během poslední doby byl postupně rozšiřován, úvodní experimenty však byly prováděny jen na první sadě nejlépe prověřených dat. V současné době obsahuje však tento korpus nahrávky 800 lidí (384 mužů a 416 žen). Tyto nahrávky tvoří celkem 220 hodin řeči. Na těchto kompletních datech byl natrénován 16-ti složkový trifonový model s 4922 stavy. Tento model byl diskriminativně přetrénován metodou MMI-FD v této práci doporučeným postupem (jedna iterace ML následovaná dvěma iteracemi MMI-FD). Oba modely byly testovány na 100 minutách čtené řeči (5 mužů a 5 žen). Pro tento test byl použit

zerogramový jazykový model obsahující 2190 slov. Chyba rozpoznávání (WER) byla redukována z $40,19\%$ na $39,02\%$.

6.3.3 Testy na korpusu SpeechDat(E)

SpeechDat(E) je databáze telefonní řeči, zaměřená na jazyky střední a východní Evropy. Celkem obsahuje 5 jazyků: češtinu, polštinu, slovenštinu, maďarštinu a ruštinu. Pro otestování diskriminativního trénování byla použita pouze česky mluvená část korpusu, která obsahuje nahrávky 1000 řečníků. Trénovací data celkem obsahovala 37457 vět 722 řečníků, jejichž celková délka byla 63 hodin. Dále byly z dat řečníků, kteří nebyli zařazeni do trénování, vytvořeny 3 testovací sady (značené *testA*, *testB*, *testC*). *TestA* obsahoval 2700 nahrávek 52 řečníků, *testB* obsahoval 2693 nahrávek 52 řečníků a *testC* obsahoval 4976 nahrávek 96 řečníků. Jazykový model pro testy byl použit zerogramový, slovník se pro jednotlivé testy lišil a obsahoval 3757 slov pro *testA*, 3864 slov pro *testB* a 6009 slov pro *testC*.

Nahrávky byly uloženy ve formátu 8-bit, 8kHz A-law. Dále z nich byly vypočteny PLP kepstrální příznakové vektory, které byly doplněny delta a akceleračními koeficienty. Pro potlačení vlivu variability přenosového kanálu byla použita kepstrální normalizace. Celková dimenze prostoru příznakových vektorů byla 36. Z takto zpracovaných trénovacích dat byly natrénovány dva trifonové modely. První, méně komplexní, obsahoval 2033 stavů, každý modelovaný 8-mi složkami (označen *model-2k*). Druhý, komplexnější, obsahoval 6768 stavů a měl rovněž 8 složek na stav (označen *model-6k7*). Tyto modely byly vytvořeny standardním trénovacím postupem v HTK podle kriteria ML. Na tyto modely pak byla aplikována metoda diskriminativního trénování MMI-FD. Výsledky experimentu jsou uvedeny v tabulce 14. Byly testovány oba modely na všech třech testovacích sadách. Výsledný rozpoznaný text byl vyhodnocen třemi způsoby:

- WER včetně neřečových událostí - zde jak referenční, tak v rozpoznané přepisy obsahovaly i neřečové události, chyba na slovech pak byla vyhodnocována standardním způsobem, kde byly neřečové události brány stejně jako ostatní slova.
- WER - standardní chyba rozpoznávání na slovech, neřečové události se v přepisech neobjevují a tedy nejsou brány v potaz.
- PER - pro detailnější vyhodnocení byla rovněž vyčíslena i chyba ve fonémech PER (Phoneme Error Rate). Zde byl jak rozpoznaný tak referenční přepis pomocí výslovnostního slovníku přepsán na řetězec fonémů.

WER [%] včetně neřečových událostí						
	model-2k			*model-6k7*		
	TestA	*TestB*	*TestC*	*TestA*	*TestB*	*TestC*
Ref.	73,66	72,21	75,10	74,03	73,00	75,62
MMI-FD	61,15	61,40	63,91	60,33	60,43	63,40
WER [%]						
	model-2k			*model-6k7*		
	TestA	*TestB*	*TestC*	*TestA*	*TestB*	*TestC*
Ref.	48,68	44,48	51,69	49,07	45,31	52,24
MMI-FD	32,10	30,92	36,23	31,04	29,55	35,57
PER [%] - chyba rozpoznávání ve fonémech						
	model-2k			*model-6k7*		
	TestA	*TestB*	*TestC*	*TestA*	*TestB*	*TestC*
Ref.	22,24	19,55	22,74	21,26	19,23	22,13
MMI-FD	14,62	13,64	15,34	13,06	12,50	14,92

Tabulka 14: Výsledky rozpoznávání MMI-FD na korpusu SpeechDat(E)

Na korpusu SpeechDat(E) dosáhla metoda MMI-FD velmi dobrých výsledků. Pokles WER se pohyboval mezi 14 až 16% absolutně! V relativním měřítku to odpovídá poklesu chyby až o 35%.

6.4 Diskriminativní trénování v úloze detekce klíčových slov

Na naší katedře je řešena i problematika detekce klíčových slov v záznamech řeči a to zejména v rámci projektů "Automatické vyhledávání klíčových slov v proudu zvukových dat" (GA AV ČR 1QS101470516) a "Překlenutí jazykové bariéry komplikující vyšetřování financování terorismu a závažné finanční kriminality" (MV VD20072010B160). I v této úloze je akustický model jednou z klíčových součástí, navíc má jeho kvalita větší vliv na kvalitu celkového systému než třeba v úloze rozpoznávání řeči s velkým slovníkem, kde má velký význam také jazykový model, zejména při zpracování češtiny. Tedy pokud diskriminativně natrénovaný model dosahuje lepších výsledků v úloze rozpoznávání řeči, měl by takto natrénovaný model dosahovat lepších výsledků i v úloze detekce klíčových slov. Jelikož množství dat dostupných pro trénování je dostatečné množství jen pro akustické modelování, příprava případného jazykového modelu pro metody diskriminativního trénování, které tento model vyžadují, je problematické. Buď mohou být použita jiná odpovídající dostupná data, nebo může být použit již natrénovaný jiný jazykový model. Tato náhradní řešení však nejsou ideální a vzhledem k tomu, že slova, která budou systémem detekována nejsou dopředu známa, může mít nevhodný jazykový model použitý pro diskriminativní trénovaní neblahý vliv. V následujícím experimentu tedy byla s výhodou použita metoda MMI-FD, která pracuje pouze akusticky a žádný jazykový model nevyžaduje, je tedy pro úlohu detekce klíčových slov mimořádně vhodná.

Pro trénování akustického modelu byla k dispozici následující data:

- Telefonní nahrávky dodané v rámci projektu VD20072010B160 - 54 hodin.
- Telefonní nahrávky z katedrálního korpusu - 1000 řečníků, 102 hodin.
- Korpus SpeechDat(E) - 722 řečníků, 63 hodin.
- Další telefonní data z interních katedrálních zdrojů - 38 hodin.

Celková délka trénovacích data byla 257 hodin. Tato data byla zpracována metodou PLP s kepstrální normalizací, s použitím delta a akceleračních koeficientů byla

celková dimenze příznakových vektorů 36. Vzhledem k požadavku na vysokou rychlost detekce klíčových slov byla komplexita akustických modelů omezena. Níže budou uvedeny výsledky pro dva trifonové akustické modely: *model-2k*, který obsahoval 2033 stavů a *model-5k9*, který obsahoval 5915 stavů. U obou modelů byl každý stav modelován 8-mi složkami normálního rozložení. Základní modely byly natrénovány pomocí HTK toolboxu. Pro diskriminativní trénování byl použit postup doporučovaný v této práci (jedna ML iterace následována dvěma MMI-FD iteracemi). Testovací data obsahovala 3278 vět, které tvořily celkem přibližně jednu hodinu řeči. V těchto nahrávkách bylo detekováno celkem 569 klíčových slov.

Kvalita systémů pro detekci klíčových slov je vyhodnocována nejčastěji pomocí tzv. ROC křivky (angl. *Receiver Operating Characteristic*). Jedná se o graf, ve kterém jsou vyneseny výsledky pro postupně se měnící práh detekce. Na vertikální ose je míra úspěšnosti detekce hledaných klíčových slov (tzv. *detection rate*). Na horizontální ose je počet chybně detekovaných slov (tzv. *false alarms*), která jsou obvykle normalizována na určitou časovou jednotku, aby bylo možné porovnávat výsledky pro různě dlouhé nahrávky či testovací sady. Na základě ROC křivky jsou pak vyhodnocovány další kritéria kvality. První kriterium nazývané EER (z angl. *Equal Error Rate*) je velikost chyby systému pro práh, který zaručuje stejný počet chyb typu chybné přijetí (*false alarm*) a chybného odmítnutí (*false reject*). Pro konkrétní test, přesně takovýto práh často nemůže být zvolen, hodnota EER se pak počítá interpolací z nejbližších okolních prahů. Jelikož EER vyjadřuje chybu systému, je cílem toto kriterium pokud možno minimalizovat. Dalším velice často používaným kriteriem odvozeným s ROC křivky je tzv. FOM (z angl. *Figure Of Merit*), jedná se o integrovanou plochu pod ROC křivkou, tedy oproti EER se snaží vyjádřit jedním číslem charakter celé křivky. Ideální systém by měl $FOM = 1$, proto je FOM kriterium vyjadřováno v procentech. Cílem je toto kriterium maximalizovat, ak aby se FOM kriterium blížilo co nejvíce 100%.

Výsledky pro *model-2k* jsou uvedeny na obrázku 19, výsledky pro *model-5k9* jsou na brázku 20. Obrázky se skládají vždy ze dvou ROC křivek, horní je pro HTK referenční model, dolní pro MMI-FD dikriminativní trénování. Z každé ROC křivky byly vyhodnoceny kriteria EER a FOM. U modelu *model-2k* bylo diskriminativním trénováním dosaženo poklesu EER o $0,58\%$ a kriterium FOM vzroslo o $1,11\%$. U druhého většího modelu (*model-5k9*) byl rozdíl ještě výraznější: EER pokleslo o $3,16\%$ a FOM vzrostlo o $3,97\%$. Tedy výsledky potrvdily očekávání, že dikriminativní modely budou dosahovat lepších výsledků. Zajímavé opět je, že v této práci navržená metoda MMI-FD dosahuje lepších výsledků na komplexnějším modelu, než na modelu jendoduším ačkoli u věšiny ostatních diskriminativních metod je tento efekt opačný a u komplexnějších modelů (které dosahují absolutne lepších výsledků) nedosahují tak dobrých výsledků jako na modelech jednodušších.

Tedy opět se potvrzuje velmi dobrá schopnost metody MMI-FD zobecňovat.

V úloze detekce klíčových slov byla s úspěchem aplikována rovněž metoda diskriminativní adaptace založená na MMI-FD kriteriu. Použití adaptace na cílová data ještě snížili EER o několik procent a rovněž o několik procent bylo zvýšeno i kriterium FOM. Bohužel, výsledky adaptace pro výše uvedené modely nebyly v době odevzdání této práce k dispozici, úspěšnost diskriminativní adaptace však byla prokázána na předchozích experimentech, které však měly odlišnou metodu zpracování řečového signálu.

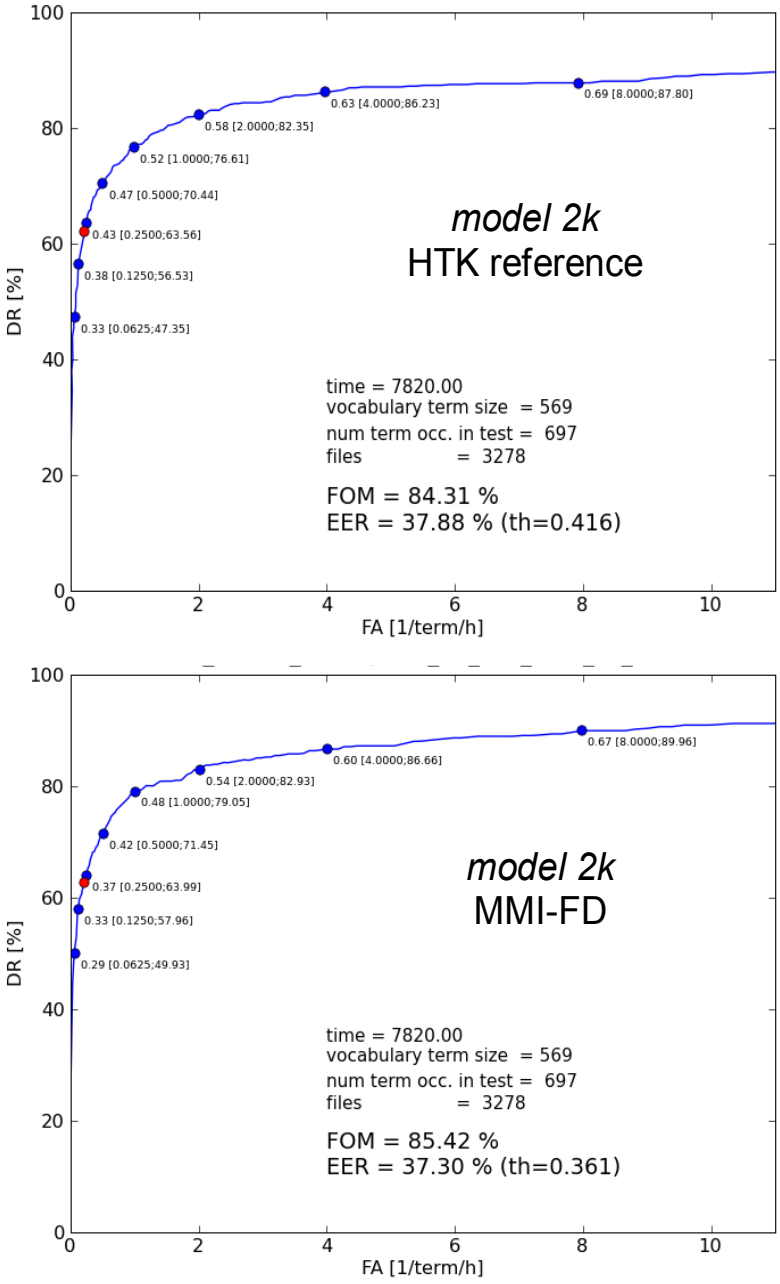
6.5 Trénování skupinových modelů

Je velmi dobře známo, že akustický model natrénovaný jen na konkrétního řečníka, vždy dosahuje na řeči tohoto řečníka lepších výsledků než model obecný. Jen v několik málo úlohách je možné tento postup aplikovat. Dá se však zobecnit jiným způsobem, a to tak, že jsou vymezeny určité skupiny řečníků, pro které je trénován speciální akustický model. Nejběžnější a často používané je rozdělení trénovacích dat na mužské a ženské nahrávky (angl. tzv. *gender-dependent acoustic models*). Pokud tedy bude mluvit muž, bude nasazen model specializovaný na muže, v případě ženy zase obráceně. V tomto případě je ovšem potřeba detekovat, zda mluvčí je muž či žena.

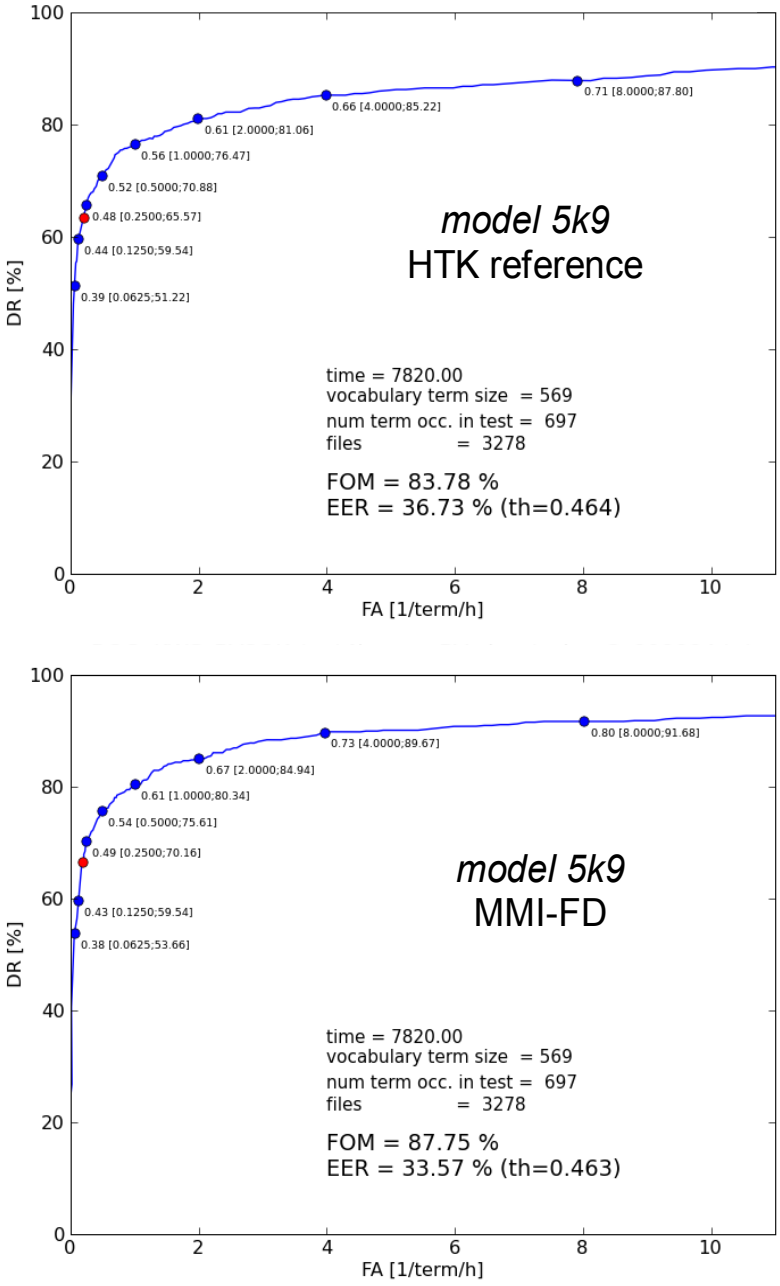
Pro trénování *gender-dependent* modelů bývá využita informace o pohlaví řečníka, která je většinou součástí anotace. Ovšem mohou se vyskytovat muži, kteří mají hlas spíše akusticky blíže k hlasu ženskému a na opak, dále je také občas v anotacích pohlaví označeno špatně, případně tato informace může u některých dat chybět úplně. Pro tyhle případy je vhodný automatický shlukovací algoritmus, který byl navržen v [6] a pro tento účel použit v [7].

Tento algoritmus je založen na kritériu, které je velice podobné kriteriu ML. Cílem tohoto algoritmu je rozdělit trénovací sadu (jednotlivé trénovací promluvy) do skupin tak, aby byly tyto skupiny co nejvíce homogenní (z pohledu akustického modelu). Počet skupin-shluků n musí být dopředu určen, pro muže a ženy je přirozeně $n = 2$. Samotný algoritmus pak probíhá následovně:

1. Na počátku jsou trénovací nahrávky rozděleny do n shluků. Tyto shluky by měly mít podobnou velikost. V případě trénování shluků mužů a žen, je úvodní rozdělení dáno anotací trénovacích nahrávek. V obecném případě může být použito náhodného roztřídění nahrávek do shluků.



Obrázek 19: *ROC křivky pro detekci klíčových slov pro model* model-2k. *Nahoře HTK referenční model, Dole MMI-FD diskriminativní trénování.*



Obrázek 20: *ROC křivky pro detekci klíčových slov pro model* model-5k9. *Nahoře HTK referenční model, Dole MMI-FD diskriminativní trénování.*

2. Proběhne (pře)trénování všech shluků z dat do nich přiřazených.

3. Vypočte se aposteriorní pravděpodobnost $P(u|M)$ pro každou nahrávku u a všechny modely M. Je použit výpočet s referenčním přepisem (angl. tzv. *forced-alignment*).

4. Každá nahrávka je přiřazena tomu shluku, jehož model má největší pravděpodobnost $P(u|M)$, která byla vypočtena v předchozím kroku:

$$M_{t+1}(u) = \arg\max_{M} P(u|M). \tag{75}$$

5. Pokud se shluky stále mění, pokračuje se krokem 2. Jinak je algoritmus ukončen.

Úplná konvergence algoritmu není zaručena, proto je vhodné, jako ukončovací podmínku stanovit určitý práh, kolik trénovacích nahrávek může být přeřazeno do jiného shluku, aby algoritmus pokračoval. Případně lze jednoduše algoritmus omezit počtem iterací.

Pokud tedy chceme ověřit, že dělení do skupin podle pohlaví je správné, použijeme jako úvodní rozdělení právě tuto informaci a spustíme algoritmus. Ten jednotlivé skupiny-shluky poopraví tak, že budou více akusticky homogenní, tedy přeřadí muže s ženským hlasem k ženám a naopak a zároveň opraví chyby v anotacích. Akustické modely natrénované z takto roztříděných nahrávek by pak měly dosahovat menší chyby v rozpoznávání řeči.

6.5.1 Experiment s gender-dependent modely

Pro ověření výše popsaného postupu byl sestaven následující experiment. Data pro trénování pocházela z korpusu UWB S01, který je popsán v kapitole 6.3.2. Rovněž základní postup pro natrénování referenčního i diskriminativního akustického modelu byl totožný (tyto modely budou označovány *GI* a *GI-DT*). Dále byly stejným postupem jako referenční model natrénovány modely pro muže a ženy. Kde pro rozdělení trénovacích dat byla použita informace z anotace (modely označené *GD*). Dále byl použit výše popsaný shlukovací algoritmus, kterým bylo dosaženo nové rozdělení na muže a ženy (modely označené *ClusterGD*, diskriminativní verze označena jako *ClusterGD-DT*). Průběh shlukovacího algoritmu je ilustrován v tabulce 15 (Muži značeni M a ženy F). Během jednotlivých iterací je přesouvána stále menší část nahrávek - algoritmus konvergoval po sedmi iteracích. Celkově bylo přesunuto několik procent nahrávek. Modely natrénované z těchto

nových shluků dosahovaly menší chyby rozpoznávání o $0,6\%$ absolutně oproti rozdělení na muže a ženy podle anotace (viz tabulka 16), pokud předpokládáme, že by bylo pohlaví řečníka vždy určeno správně. Pokud by však bylo pohlaví řečníka detekováno špatně, byl by nárůst chyby rozpoznávání veliký.

Tabulka 15: Průběh přesunů mezi shlukem mužů a žen

Počet iterací (i)	[%]			
	$M_{i-1} \rightarrow M_i$	$M_{i-1} \rightarrow F_i$	$F_{i-1} \rightarrow F_i$	$F_{i-1} \rightarrow M_i$
1	96,81	4,81	95,76	2,63
2	99,37	0,90	99,21	0,52
3	99,34	0,37	99,93	0,36
4	99,75	0,25	99,69	0,31
5	99,65	0,25	99,78	0,32
6	99,90	0,06	99,94	0,10
7	99,97	0,01	99,99	0,03

Tabulka 16: Porovnání různých variant gender-dependent modelů

	WER [%]	
GI	40,19	
GI − DT	39,02	
Identifikace shluku	správná	nesprávná
GD	37,50	64,08
ClusterGD	36,89	63,57
ClusterGDDT	35,81	61,92
GIMLAdapt	38,08	52,18
GIDTAdapt	36,99	46,60

Například u systémů pracujících v reálném čase nemůžeme očekávat, že by detektor muž/žena fungoval naprosto přesně, hned na počátku promluvy. Je tedy třeba hledat takové metody trénování těchto modelů, které nevykazují takový rozdíl, mezi správnou a špatnou detekcí řečníka a navíc dosahují dostatečně nízké

chybovosti, pokud je řečník detekován správně. V tomto experimentu byly zkoumány čtyři různé metody trénování takovýchto modelů:

- *ClusterGD* Jedná se o sadu modelů (model pro muže a model pro ženy), která je natrénována stejným postupem jako model referenční (*GI*). Model pro muže i pro ženy je trénován od začátku zcela odděleně.
- *ClusterGD-DT* Sada modelů vycházející ze sady předchozí. Modely byly diskriminativně přetrénovány metodou MMI-FD.
- *GI-MLAdapt* Sada modelů vycházející z referenčního modelu *GI*. Ten je adaptován na muže a na ženy pomocí metody MAP.
- *GI-DTAdapt* Sada modelů vycházející z modelu *GI-DT*. Ten je adaptován na muže a na ženy pomocí diskriminativní adaptace MAP založené na kriteriu MMI-FD.

Výsledky pro všechny metody jsou v tabulce 16. Nejlepších výsledků pro ideálně fungující detektor řečníka dosahuje sada modelů *ClusterGD-DT*. Nejmenší rozdíl mezi správnou a nesprávnou detekcí je u sady modelů *GI-DTAdapt*, tedy pokud očekáváme větší chybovost v detekci řečníka, je tento trénovací postup nejlepší volbou. Pokud detektor řečníka funguje velice dobře, je vhodné použít spíše postup *ClusterGD-DT*. Detailněji je tento experiment popsán v [7].

6.5.2 Experiment s větším počtem skupin

Pokud poměrně dobře funguje rozdělení trénovacích dat do dvou skupin, mohlo by být možné rozdělit trénovací data i do více než dvou skupin. Hlavní otázkou pak ovšem je, podle čeho toto dělení provést. Rozdělení na muže a ženy se samo nabízí, pro větší počet shluků už ovšem žádné jednoduché dělení není. Ovšem, vzhledem k úspěšnému použití automatického shlukovacího algoritmu v minulé kapitole, je možné toto dělení přenechat tomuto algoritmu.

Experiment, kde bylo takto provedeno dělení do dvou a čtyř shluků je podrobně popsán v [8]. U dělení do dvou shluků je postup totožný s dělením na muže a ženy, jen počáteční rozdělení je vygenerováno náhodně (označeno $2Cl$). Algoritmus po několika iteracích dokonverguje ke dvěma akusticky homogenním skupinám (v tomto případě se jedná přibližně o muže a ženy). Při dělení do čtyř shluků můžeme využít dvou postupů, buď dělit shluky hierarchicky, tedy nejprve nahrávky rozdělit na dva shluky a pak každý z nich na další dva (označeno $4Cl_{Hi}$),

nebo algoritmus spustit rovnou na začátku pro čtyři shluky (označeno $4Cl_{Di}$). Všechny tyto přístupy byly v [8] testovány. Kromě různých možností a počtu shluků bylo také testováno několik trénovacích metod. Stejně jako v předchozím experimentu se jednalo o referenční postup (značen ML), diskriminativní trénování MMI-FD (značeno DT) a standardní i diskriminativní adaptaci (značeno ML_{Adapt} a DT_{Adapt}). Pro úplnost ještě uveďme, že referenční model, který byl natrénován klasickým způsobem, je označen jako SC_ML a jeho diskriminativní varianta jako SC_DT.

Výsledky pro všechny kombinace možností dělení i trénovacích metod jsou v tabulce 17. V tabulce jsou uváděny chyby rozpoznávání a to jak pro případ, kdy detektor shluku funguje správně (pro každou nahrávku detekuje ten shluk, který dává nejmenší chyby rozpoznávání), tak i pro případ, že detektor poskytuje pouze tu nejhorší variantu - tedy, že pro každou nahrávku určí ten shluk, který dává největší chybu rozpoznávání. Z uvedených dvou sloupců je možné vyhodnotit citlivost použité trénovací metody na chybovost detektoru. Nejlepších výsledků bylo dosaženo pro přímé dělení do čtyř shluků a diskriminativní adaptaci (označeno $4Cl_{Di}\_DT_{Adapt}$). Zde je chyba rozpoznávání nejmenší pro správně fungující detektor a ani pro nejhorší možnou variantu detekce nárůst chyby rozpoznávání není veliký ve srovnání s ostatními metodami.

6.6 Závěrečné shrnutí výsledků

Na závěr byly shrnuty výsledky všech vlastních diskriminativních metod. Tyto výsledky jsou uvedeny v tabulkách 18 a 19, kde jsou postupně pro jednotlivé korpusy vyhodnoceny testované metody. Je zde uvedena velikost trénovaného modelu (počet stavů a počet složek na stav), dosažená chyba WER a její absolutní a relativní pokles oproti referenčnímu modelu.

Kromě relativně standardní úlohy automatického rozpoznávání pomocí jednoho univerzálního modelu, byly vlastní metody pro diskriminativní trénování testovány i na úloze detekce klíčových slov a na rozpoznávání řeči pomocí skupinových modelů. Souhrn výsledků je v tabulkách 20 a 21.

Z uvedeného výčtu výsledků experimentů je patrné, že diskriminativní metody opravdu poskytují kvalitnější odhady parametrů akustických modelů. Na některých úlohách byla snížena chyba rozpoznávání velmi významným způsobem. Pro úlohy, kde není příliš velké množství trénovacích dat a akustický model není příliš komplexní, je nejvhodnější metodou zřejmě MMI-TF. Pro rozsáhlejší úlohy s většími modely se hodí spíše metoda MMI-FD. U této metody, na rozdíl od

Tabulka 17: Výsledky pro experiment s dělením do čtyř skupin

	WER [%]	
SC_ML	28,63	
SC_DT	26,40	
Identifikace shluku	správná	nejhorší
$2Cl\_ML$	28,24	33,35
$2Cl\_DT$	25,99	30,71
$2Cl\_ML_{Adapt}$	28,36	32,47
$2Cl\_DT_{Adapt}$	25,97	28,64
$4Cl_{Hi}\_ML$	27,38	47,86
$4Cl_{Hi}\_DT$	24,83	44,17
$4Cl_{Hi}\_ML_{Adapt}$	27,17	37,22
$4Cl_{Hi}\_DT_{Adapt}$	25,61	30,52
$4Cl_{Di}\_ML$	25,31	45,82
$4Cl_{Di}\_DT$	25,99	43,31
$4Cl_{Di}\_ML_{Adapt}$	25,35	40,97
$4Cl_{Di}\_DT_{Adapt}$	23,34	32,72

ostatních (zde testovaných i jinde publikovaných), nedochází k snižování efektivnosti s nárůstem velikosti akustického modelu, naopak u několika experimentů bylo prokázáno, že na větších modelech funguje ještě lépe než na modelech menších. U experimentů na úloze titulkování parlamentních přenosů se snížil efekt diskriminativního trénování jen nepatrně i u modelu s dvojnásobným počtem stavů. Dále na této úloze bylo prokázáno, že tato metoda dosahuje podobného relativního snížení chyby jak u zerogramového testu, tak u testu s bigramovým modelem, který obsahoval 120 tisíc slov.

Tabulka 18: Souhrnné výsledky vlastních metod na všech datech, která byla k dispozici. První část.

Použitá metoda	Počet stavů	Počet složek	WER [%]	Pokles WER absolutní [%]	Pokles WER relativní [%]
Korpus UWB S01 - 6,3 hodiny - zerogram test					
MMI-FD	425	8	12,28	1,00	7,5
MMI	425	8	12,35	0,93	7,0
MMI+MMI-FD	425	8	11,42	1,86	14,0
MMI-TF	425	8	11,49	1,79	13,5
MMI-FD	1424	8	8,71	1,43	14,1
MMI	1424	8	10,28	-0,14	-1,3
MMI+MMI-FD	1424	8	9,92	0,22	2,2
MMI-TF	1424	8	9,42	0,72	7,1
Parlament - 100 hodin - zerogram test					
MMI-FD	5285	8	23,84	2,83	10,6
MMI-TF	5285	8	25,91	0,76	2,8
MMI-FD	12135	8	24,94	2,15	7,9
Parlament - 100 hodin - bigram test (HDecode)					
MMI-FD	5285	8	14,37	1,48	9,3

Tabulka 19: Souhrnné výsledky vlastních metod na všech datech, která byla k dispozici. Druhá část.

Použitá metoda	Počet stavů	Počet složek	WER [%]	Pokles WER absolutní [%]	Pokles WER relativní [%]
Resource Management korpus - 3,8 hodin - test *feb98*					
MMI-FD	1532	1	4,74	1,74	26,9
MMI	1532	1	4,27	2,21	34,1
MMI+MMI-FD	1532	1	4,34	2,14	33,0
MMI-TF	1532	1	4,18	2,30	35,5
MMI-FD	1532	6	2,77	0,04	1,4
MMI	1532	6	2,69	0,12	4,2
MMI+MMI-FD	1532	6	2,69	0,12	4,2
MMI-TF	1532	6	2,81	0,00	0,0
Korpus UWB S01 - 220 hodin - zerogram test)					
MMI-FD	4922	16	39,02	1,17	2,9
Korpus SpeechDat(E) - 63 hodin - zerogram test)					
MMI-FD (test A)	2033	8	32,10	16,58	34,1
MMI-FD (test B)	2033	8	30,92	13,57	30,5
MMI-FD (test C)	2033	8	36,23	15,46	29,9
MMI-FD (test A)	6768	8	31,04	18,03	36,7
MMI-FD (test B)	6768	8	29,55	15,76	34,8
MMI-FD (test C)	6768	8	35,57	16,67	31,9

Tabulka 20: Souhrnné výsledky v úloze detekce klíčových slov

Data z několika korpusů - 257 hodin						
Použitá metoda	Počet stavů	Počet složek	EER [%]	Pokles EER absolutní [%]	FOM [%]	Narůst FOM absolutní [%]
MMI-FD	2033	8	37,30	0,58	85,42	1,09
MMI-FD	5915	8	33,57	3,16	87,75	3,97

Tabulka 21: Souhrnné výsledky v experimentech se skupinovými modely

Použitá metoda	Počet stavů	Počet složek	WER [%]	Pokles WER absolutní [%]	Pokles WER relativní [%]
Korpus UWB S01 - 220 hodin - 2 shluky - zerogram test					
MMI-FD adaptace	2x 4922	16	36,99	3,20	8,0
Parlament - 100 hodin - 4 shluky - zerogram test					
MMI-FD adaptace	4x 5385	8	23,34	5,29	18,5

7 Implementace algoritmů s využitím vlastností moderních počítačů

Při vývoji algoritmů, které zpracovávají rozsáhlá řečová data, která mají desítky někdy i stovky hodin, je velmi důležitá také časová a paměťová náročnost výpočtu. Celý problém je navíc umocněn tím, že se zpravidla jedná o iterativní algoritmy, které vyžadují několika-násobný průchod trénovacími daty.

Obecně se považuje trénovací čas systémů pro rozpoznávání řeči za nedůležitý faktor, jelikož tyto systémy jsou natrénovány dopředu a při nasazení těchto systémů již trénování neprobíhá. Při vývoji trénovacích metod je však situace odlišná. Jelikož je nutné při vývoji těchto metod provést stovky či tisíce různých experimentů s různými daty pro různé varianty, modifikace a nastavení, časová náročnost algoritmů také ovlivňuje rychlost vývoje těchto metod. Tedy příliš pomalé implementace jednotlivých metod spolu s nedostatečným harwareovým vybavením mohou zpomalit i samotný vývoj nových metod, naopak dobře optimalizované algoritmy spolu s velkou výpočetní kapacitou mohou vývoj významně urychlit. Pro snížení časové náročnosti výpočtu existují v zásadě tři cesty, které mohou být navzájem kombinovány:

- Algoritmické optimalizace - využití různých aproximací, prořezávání či jiných výpočetních postupů, které nejsou tak výpočetně náročné a jejich odchylky od původního algoritmu jsou z pohledu dané aplikace zanedbatelné.
- Využití "hrubé síly" - tedy paralelizace výpočtu na větší počet jader procesoru, počítačů, či využití celých gridů. Na českých univerzitách lze například využít celorepublikovou síť výpočetních serverů *MetaCentrum* (viz http://meta.cesnet.cz).
- Využití všech výpočetních prostředků moderních počítačů - tedy optimalizace na úrovní programového kódu, kde algoritmus výpočtu zůstává shodný, jen je upraven tak, aby mohl lépe využít vlastnosti moderních počítačů. Jedná se tedy zejména o využití rozšířených vektorových výpočetních instrukcí (např. SSE a SSE2), optimalizovanou alokaci paměti, odpovídající dané architektuře, vícevláknová implementace pro využití všech dostupných jader procesoru a v poslední době velice populární využití grafických karet pro zpracování datově paralelizovatelných částí algoritmu.

7.1 Vyhodnocení výstupních pravděpodobností akustického modelu

Při trénování akustických modelů, tvoří velkou část nutných výpočtů vyhodnocení výstupní pravděpodobnosti akustického modelu. Trifonový akustický model obsahuje běžně několik tisíc stavů, kde je každý z těchto stavů modelován několika složkami (nejčastěji 8 či 16) normálního rozdělení s vysokou dimenzí (obvykle mezi 32 až 45). Pro všechny tyto složky, kterých je celkem několik desítek až stovek tisíc, je třeba vyhodnotit pravděpodobnost

$$p_i(\boldsymbol{o}) = \frac{1}{(2\pi)^{N/2}|\boldsymbol{C}_i|^{1/2}} \exp\left[(\boldsymbol{o}-\boldsymbol{\mu}_i)^T \boldsymbol{C}_i^{-1}(\boldsymbol{o}-\boldsymbol{\mu}_i)\right], \tag{76}$$

kde $\boldsymbol{\mu}_i$ je vektor středních hodnot, $\boldsymbol{C}_i$ je kovarianční matice a $\boldsymbol{o}$ je aktuální vektor pozorování. Typický počet vektorů pozorování za jednu sekundu řeči je 100. Tedy pro vyhodnocení všech výstupních pravděpodobností u diagonálního modelu s 5-ti tisíci stavy, $16 - ti$ složkami na stav a dimenzí 45 je za jednu sekundu potřeba provést 800 milionů operací sčítání/odčítání, 720 milionů operací násobení/dělení a 8-milion-krát vypočítat exponenciální funkci. Pro snížení výpočetní náročnosti jsou v praxi používány následující optimalizace:

- Výpočty prováděny v logaritmické škále čímž odpadá exponenciála a násobení je převedeno na mnohem rychlejší sčítání, pro součet dvou zlogaritmovaných pravděpodobností lze využít různé aproximace, které se obejdou bez výpočetně velice náročných logaritmů a exponenciál (v této práci navržená aproximace je popsána níže).
- Celý úvodní zlomek je na datech nezávislý a jeho logaritmus je vyhodnocen dopředu.
- Variance jsou dopředu invertovány a tím je dělení převedeno na násobení.

U metod diskriminativního trénování, kde zabírá výpočet výstupních pravděpodobností relativně velkou část výpočtů, je jedním z možných řešení, které se velice často používá, rychlá detekce významných složek akustického modelu. Jelikož jen malá část složek pro příznakový vektor v daném čase má nenulovou pravděpodobnost, je takřka zbytečné vypočítávat všechny složky a stačí vyhodnotit jen ty s relevantní pravděpodobností. Problémem samozřejmě je, jakým výpočetně nenáročným způsobem určit, které složky jsou významné a které ne. Jedním z těchto algoritmů je například tzv. *Roadmap* algoritmus [35]. Další možností je přenechat část tohoto výběru na samotném dekódovacím algoritmu s

jeho prořezáváním a vyhodnocovat jen ty stavy akustického modelu, které se v daném čase v rozpoznávací síti objevují. Avšak každý předvýběr či prořezávání sebou vždy nese riziko poklesu úspěšnosti rozpoznávání, tyto aproximace jsou používány zejména proto, že není jiné řešení, které by bylo výpočetně stejně nebo ještě méně náročné.

Při trénování akustického modelu není sice požadavek malé výpočetní náročnosti tak přísný, ale o to větší důraz je zde kladen na přesnost. Vzhledem k tomu, že se většinou jedná o iterativní algoritmy, které mají zaručenu konvergenci, mohou přílišné aproximace degradovat proces trénování, který pak nemusí konvergovat. Bylo by tedy vhodné se v procesu trénování hrubých aproximací pokud možno vyvarovat.

7.2 Optimalizace aplikované a využité v této práci

V průběhu řešení problematiky diskriminativního trénování musela být rovněž řešena problematika extrémní výpočetní náročnosti, která pramení ze základního principu diskriminativních metod, kde jsou v každém čase uvažovány všechny stavy modelu (pokud není použito prořezávání). Při implementaci v jazyce C++ byly využity následující optimalizace:

- V první řadě je to využití více-jádrových procesorů, nebo počítačů s více procesory. Rozdělit výpočet výstupních pravděpodobností mezi několik vláken není problém, jelikož tato úloha je lehce paralelizovatelná, jelikož se skládá z nezávislých částí. Buď lze zpracování jednoho souboru (v off-line variantě - trénování) či vstupního bufferu (v on-line variantě - rozpoznávání v reálném čase) rovnoměrně rozdělit na několik částí, které pak zpracuje jedno z vláken. Druhou možností je, rozdělit akustický model, každé vlákno pak vyhodnocuje pravděpodobnosti pro část stavů modelu.

- Dalším důležitým prvkem je vhodná reprezentace dat v paměti. Parametry středních hodnot a dopředu invertovaných variancí by neměly být uloženy objektově, nýbrž za sebou v blocích a to tak, aby jednotlivé vektory začínaly na adrese dělitelné 128bit. V případě nutnosti je vhodné rozšířit dimenzi příznakových vektorů uměle na velikost dělitelnou čtyřmi (v případě jednoduché přesnosti - 32bit float) nebo dvěma (v případě dvojité přesnosti - 64bit double). Bloky za sebou uložených vektorů středních hodnot a variancí by měly mít velikost odpovídající velikosti L2 cache daného procesoru, proto je dobré umožnit velikost bloku nastavit externě. Dále je vhodné vyhodno-

covat několik příznakových vektorů najednou, opět to umožní vyšší využití rychlých registrů, L1 a L2 cache procesoru.

- Před samou realizací je třeba zhodnotit zda by nebylo dostačující výpočet provádět v jednoduché přesnosti, tím by se výpočet významně urychlil, z mých zkušeností vyplývá, že jednoduchá přesnost, je dostačující pro vhodně implementovaný výpočet výstupních pravděpodobností v logaritmické doméně. Dále je možné využít rychlých implementací logaritmů a exponenciál s omezenější přesností.

- Po výpočtu logaritmu pravděpodobnosti jednotlivých složek, je třeba vypočítat součet těchto pravděpodobností, jehož výsledek je třeba opět konvertovat do logaritmické domény. Vzhledem k omezené přesnosti není vhodné jednoduše všechny výsledky převést z logaritmické domény zpět do pravděpodobností, zde je sečíst a opět zlogaritmovat. Je lepší použít funkci přímo určenou pro součet dvou zlogaritmovaných pravděpodobností, jejichž výsledek má být opět logaritmus pravděpodobnosti. Podrobněji je toto diskutováno níže v kapitole 7.2.1.

- Dalším důležitým způsobem jak zvýšit rychlost výpočtu je využití SSE instrukcí. Je to sada tzv. SIMD (Single Instruction Multiple Data) instrukcí, které umožňují některé operace provádět "najednou" s několika různými operandy. Základní sada SSE instrukcí umožňuje provádět základní operace typu sčítání, odčítání, násobení, dělení, druhou mocninu, druhou odmocninu a některé další se čtyřmi čísly s jednoduchou přesností najednou. Tedy s využitím SSE instrukcí se může daný výpočet až 4x zrychlit. Pro operace, s dvojitou přesností je určena sada SSE2, kde je teoretické zrychlení 2x. Současné překladače umí těchto instrukcí využívat rovněž, ovšem ne na všech místech je to možné zařídit automaticky. Proto je vždy lepší použití těchto instrukcí naprogramovat přímo.

- Dále je také dobré celý program upravit tak, aby bylo možné rozsáhlejší úlohy rozdělit i mezi několik počítačů nebo uzlů gridu. V metodách diskriminativního trénování lze akumulovat statistiky nezávisle pro jednotlivé části trénovací množiny a vždy před koncem dané iterace tyto statistiky shromáždit a vypočítat nové odhady parametrů modelu. Tyto nové modely opět rozdistribuovat a pokračovat paralelně v další iteraci.

7.2.1 Součet pravděpodobností v logaritmické doméně

Pokud počítáme výstupní pravděpodobnosti v logaritmické doméně, dostáváme se k problému jak vyhodnotit součet těchto pravděpodobností, jejichž výsledek má být opět logaritmus pravděpodobnosti. Tato situace nastává v případech, kdy vypočítáváme celkovou pravděpodobnost stavu z pravděpodobností jednotlivých složek a také při normalizaci vektoru logaritmovaných pravděpodobností na jedničku. Jak již bylo popsáno výše, jednoduchý vztah

$$p^{log} = \log(\exp(p_1^{log}) + \exp(p_2^{log})) \tag{77}$$

není příliš numericky vhodný a stabilní, zejména při implementaci v jednoduché přesnosti, rovněž je poměrně výpočetně náročný - vyžaduje tři výpočetně náročné operace logaritmus/exponenciála.

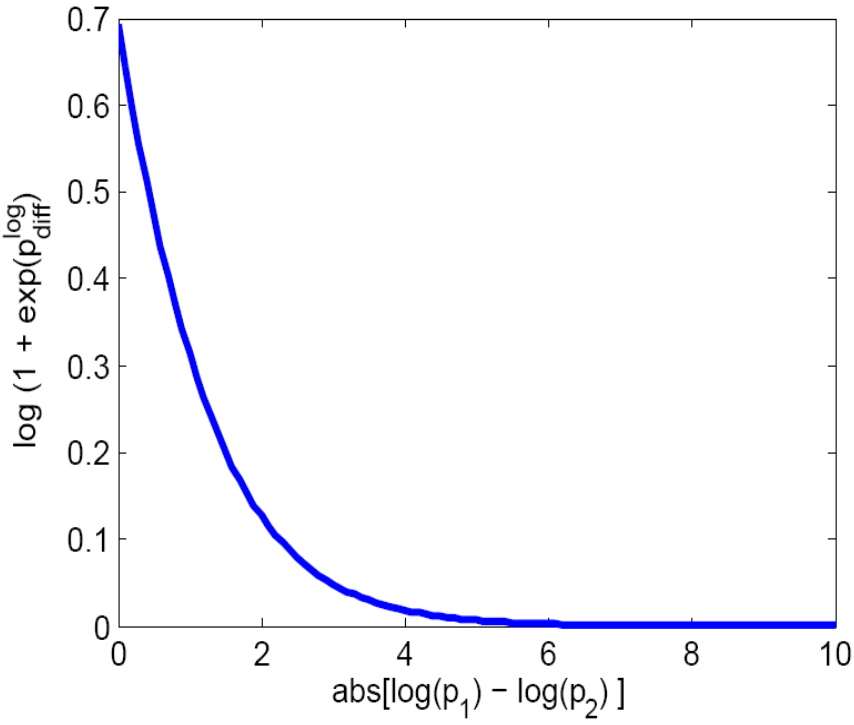
Numericky vhodnější je řešení, kde je zachováno maximum ze vstupních logaritmovaných pravděpodobností a vypočte se pouze aditivní příspěvek druhé menší pravděpodobnosti:

$$\begin{aligned} p_{max}^{log} &= \max(p_1^{log}, p_2^{log}) \\ p_{diff}^{log} &= |(p_1^{log} - p_2^{log})| \\ p^{log} &= p_{max}^{log} + \log(1 + \exp(p_{diff}^{log})). \end{aligned} \tag{78}$$

Tato verze je i méně výpočetně náročná, jelikož byla jedna z exponenciál nahrazena jednou podmínkou a dvěma operacemi sčítání/násobení. Navíc je možnost dále výpočet urychlit aproximací aditivní části rovnice (78). Tato část rovnice je funkce, která je závislá na rozdílu logaritmovaných pravděpodobností, který může nabývat pouze kladných hodnot. Pro ilustraci je vyobrazena na grafu 21. Z grafu je patrné, že funkce je velice hladká a s narůstajícím rozdílem pravděpodobností konverguje k nule. Proto je možné jí velice dobře aproximovat.

7.3 Využití grafických karet pro negrafické výpočty

U moderních počítačů dnešní doby není hlavní procesor (CPU) jediným čipem s velkým výpočetním výkonem. Další velmi výkonný čip je na grafické kartě počítače (GPU). I grafické karty prodělaly za posledních několik let rapidní vývoj, který se přizpůsoboval novým požadavkům jak v profesionální sféře 3D modelování, tak v počítačových hrách. Postupem času bylo nutné dříve velice specializovanou architekturu s fixními jednotkami pro výpočet 3D scény a jinými fixními jednotkami

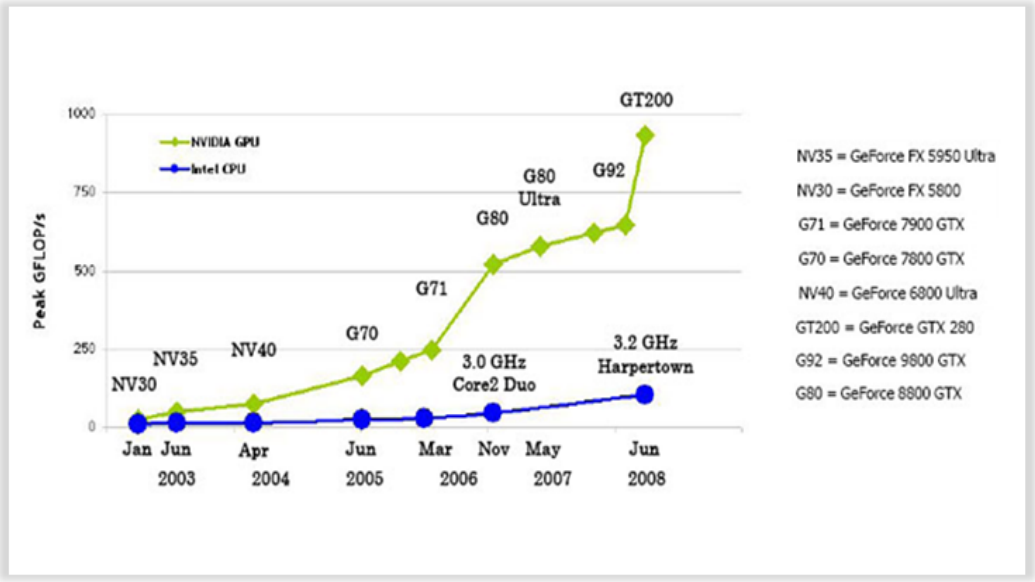


Obrázek 21: *Průběh funkce pro robustní součet logaritmů pravděpodobnosti.*

pro renderování textur zobecnit tak, aby byly k dispozici obecnější výpočetní jednotky, které budou programovatelné a zvládají širší spektrum funkcí a tím i aplikací a efektů. Od určité úrovně bylo zřejmé, že výpočetní jednotky na grafických kartách dosahují takové obecnosti, že je možné jejich výkon využít i v jiných negrafických úlohách. Navíc stále díky svojí větší specializaci dosahují mnohem vyššího výpočetního výkonu než současné CPU, jak ilustruje graf 22.

Tento vysoký výpočetní výkon je dán tedy jednak stále vysokou specializací, ale rovněž vysokým počtem nezávislých výpočetních jader a tedy i celkovým počtem tranzistorů a samotné plochy čipu. Rozdíl mezi CPU čipem (Intel Core 2 Duo - jádro Penryn - 400 milionů tranzistorů) a GPU (NVIDIA GTX 280 - 1400 milionů tranzistorů) je ilustrován na obrázku 23.

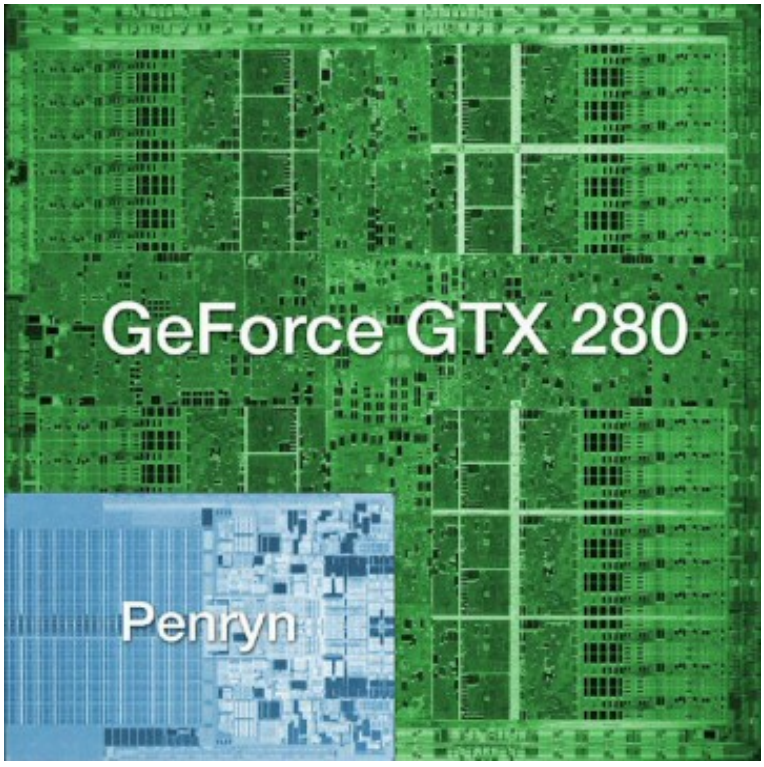
Základní rozdíl je ovšem v samotné architektuře. CPU musí být připraveno na obecné programy, ve kterých většinou nejde jen o složité výpočetní operace na rozsáhlých datech. Oproti tomu GPU je přímo navrženo pro operace s plovoucí desetinou čárkou a zpracování velkého množství dat, které lze rozdělit do paralelně zpracovatelných bloků. Zpracovávané úlohy však musí být spíše jednodušší. Tomu



Obrázek 22: *Nárůst výpočetního výkonu u CPU a u GPU v poslední době. (zdroj: www.think-techie.com)*

odpovídá i architektura jednotlivých čipů, jak je ilustrováno na obrázku 24.

Prvním velkým průkopníkem ve využívání GPU pro negrafické výpočty byl distribuovaný projekt pro simulaci proteinů *Folding@Home*. Tento projekt zahrnuje stovky menších podprojektů různých výzkumných skupin z oblasti proteomiky a molekulární biologie a distribuuje jejich výpočty mezi uživatele přihlášené k tomuto projektu. K tomuto projektu se může přihlásit kdokoli, kdo má počítač s připojením na internet. Na jeho počítači je pak s malou prioritou puštěna služba, která automaticky přijímá, zpracovává a odesílá jednotlivé datové balíčky projektu *Folding@Home*. Tento projekt vznikl v roce 2000 a jeho význam stále sílí. V roce 2006 uvedl ve spolupráci s firmami IBM a AMD/ATI novou verzi, která podporuje výpočty na grafických kartách (v té době byly podporované pouze herní konzole Playstation3 a grafické karty ATI Radeon 1300 a 1900 na běžných počítačích). V roce 2008 pak byla vydána nová verze podporující nové grafické karty od obou hlavních výrobců, jak AMD/ATI tak NVIDIA. V současné době je do projektu zapojeno přes dva miliony uživatelů s přibližně 250 tisíci aktivních CPU jader s celkovým reálným výkonem 270 TFlops/s a s 65 tisíci GPU jader s reálným výkonem 4250 TFlops/s. I z těchto čísel je patrný výkonnostní nepoměr mezi aktivními CPU a GPU. Na druhou stranu GPU implementace není tak obecná a v současné době dokáže zpracovávat jen omezenou třídu simulací.



Obrázek 23: *Rozdíl ve velikosti čipu dnešních CPU a GPU.*

Obrázek 24: *Rozdíl v architektuře CPU a GPU čipů.*

Kromě projektu *Folding@Home* se objevuje stále více dalších aplikací, které dokáží využít výkonu grafických karet i pro negrafické výpočty, zejména se jedná o simulace molekulární dynamiky, proudění plynů a tekutin, elektromagnetických polí a mechanických soustav, dále o aplikace ze zpracování signálu, komprese/dekomprese videa, zvuku i dat, šifrování a dešifrování. Velice populární je použití ve vědeckém světě, kde je možné za velice malou cenu využít velký výpočetní výkon na jednom počítači a není třeba obsazovat desítky jader výpočetních serverů.

Samozřejmě, vzhledem k velmi odlišné architektuře GPU, je odlišné i samotné programování výpočtů na grafických kartách. V současné době jsou k dispozici dva stabilní a nejvíce využívané programové nástroje:

- ATI Stream SDK - je soubor nástrojů a příkladů pro tvorbu aplikací využívajících grafické karty firmy AMD/ATI. Jedná se o upravenou verzi jazyka C, kde je GPU část přeložena speciálním překladačem a zbytek programu běžným C/C++ překladačem. Vlastní programovací nástroje je možné integrovat například do MS Visual Studia.
- NVIDIA CUDA - je rovněž soubor nástrojů a příkladů od konkurenční firmy vyrábějící grafické čipy. Rovněž zde se jedná o upravenou verzi jazyka C, oproti ATI Stream SDK má k standardnímu jazyku C ještě blíže. Rovněž zde je speciální překladač pro GPU část a celé prostředí lze rovněž zaintegrovat do MS Visual Studia.

Programování negrafických výpočtů na grafických kartách je třeba přizpůsobit dané architektuře, tak aby daný program využíval v maximální míře přednosti této architektury a její výkon. V prvé řadě si je třeba uvědomit, že velké množství poměrně jednoduchých nezávislých výpočetních jader lze optimálně využít jen pro jednodušší dobře datově paralelizovatelné problémy. Tedy takové problémy, nebo spíše jejich části, které lze rozdělit do mnoha (minimálně stovky) částí, které je možno zpracovávat nezávisle. Samotná příprava programu nejdříve spočívá v rozdělení původního algoritmu, na část, která je dobře paralelizovatelná a bude se zpracovávat na GPU, zbytek programu, který poběží stále na CPU. Jelikož má GPU svojí samostatnou paměť, je třeba před výpočtem přenést potřebná data z běžné paměti počítače do paměti GPU, potom spustit výpočet a po jeho skončení opět přes sběrnici stáhnout data zpět do běžné paměti počítače a pokračovat v programu na CPU. Výhodou toho to řešení je to, že paměť na GPU je několikrát rychlejší než běžná paměť a tedy výpočet probíhá rychleji, nevýhodou je nutné transportování dat po sběrnici. Proto program musí být navržen tak, aby tyto vlastnosti bral v potaz, tedy omezil transporty dat po sběrnici na minimum (spíše

méně větších bloků). Na GPU by pak měly být implementovány jen skutečně aritmeticky náročné úlohy, tedy aby výpočetní doba výrazně přesahovala dobu pro transport dat po sběrnici, ale i paměťové operace na samotném GPU. Zvýše uvedeného je patrné, že implementace na GPU se ani zdaleka nehodí pro všechny algoritmy, spíše naopak jen několik málo algoritmů či jejich částí se pro implementaci na GPU skutečně hodí a tato implementace bude efektivní.

Konkrétně tedy každý GPU program obsahuje dvě části:

- Přípravu dat a spuštění výpočetního jádra úlohy (tzv. *kernelu*). V této části jsou připravena data pro GPU výpočet, přenesena po sběrnici na GPU a spuštěn samotný *kernel*. Během výpočtu na GPU, mlže procesor provádět jiné operace. Po dokončení výpočtu jsou výsledná data přenesena zpět do paměti počítače.
- Výpočetní jádro - *kernel* - je funkce, která popisuje univerzální algoritmus zpracování jednoho bloku dat. Tento algoritmus je pak spouštěn na všechny paralelní bloky. Konkrétní zpracování jednotlivých bloků řídí ovladače GPU a programátor do něj již nijak nezasahuje.

Při trénování akustických modelů lze využít grafickou kartu zejména pro výpočet výstupních pravděpodobností, kde může být celá zpracovávaná promluva vyhodnocena dopředu a uspoří se tedy přenosy po PCI sběrnici z paměti CPU do paměti GPU. U metod diskriminativního trénování, kde výpočet těchto pravděpodobností tvoří velkou část, došlo k výraznému zkrácení doby trénování. Vysoký výpočetní výkon by se dal využít i v dalších částech diskriminativního trénování, ovšem zde je již problém s obtížnou paralelizovatelností výpočtu, nebo s nutností časté synchronizace výpočtu jednotlivých bloků. Zároveň režie spojená s přenosem dat přes sběrnici u jiných algoritmů mohla hrát významnou roli.

7.4 Vývoj hardware

Jak již bylo ilustrováno na obrázku 22, výkon grafických karet v poslední době roste mnohem rychlejším tempem než výkon samotných procesorů. Do konce roku 2009 má být uvedena na trh nová generace karet ATI s více než dvojnásobným výkonem generace předchozí, která byla na trh uvedena relativně nedávno: začátkem léta 2008 (a pro úplnost, ještě předchozí generace koncem roku 2007).

Situace u karet NVIDIA je méně přehledná, jelikož v poslední době byly velice často jednotliví karty a čipy přejmenovávány, kdy některé totožné čipy byly prodávány pod několika různými názvy. Oficiálně tyto karty tvoří jednotlivé řady - 8000, 9000 a nejnovější řada GTX200, reálně je mezi těmito řadami různě rozptýlena řada úprav a jedna zcela nová generace karet. Důležitý je také postupný vývoj výrobní technologie z 80nm přes 60 a 55nm na v současné době začínajících 40nm. Začátkem roku 2010 pak má být uvedena noví karta ze zcela nové generace pod označením *Fermi*, která by měla být primárně určena pro výpočty a měla by disponovat zvýšeným výkonem pro operace s dvojitou přesností.

V roce 2010 by měl být také představen projekt Intelu na nový více-jádrový procesor/grafickou kartu s názvem Larrabee. Mělo by se jednat o netypickou architekturu, který bude někde mezi současnými grafickými kartami a současnými více-jádrovými procesory. Určena by měla být přímo pro náročné paralelní výpočty. V době vzniku této práce bohužel nebylo známo ještě příliš ověřených detailů o této architektuře.

Dále do budoucna je rovněž uvažováno o spojení standardních CPU jader s mnoha GPU jádry na jednom čipu. Takovýto čip vytvořila firma IBM a je jádrem herních konzolí Palystation3. V tomto čipu je jedno CPU jádro a několik GPU jader zároveň. V současné době se vývojem takovýchto čipů zabývá AMD a zřejmě i Intel, nicméně uvedení žádného čipu postaveného na této technologii není v dohledné době ohlášen.

7.5 Vývoj software

Vzhledem k narůstající popularitě negrafických výpočtů na grafických kartách, ale i paralelního programování na klasických procesorech, vznikají rovněž nové softwarové nástroje pro vývoj aplikací na tomto hardware. V této době se aktuálně připravují dvě nová programovací prostředí:

- OpenCL (Open Computing Language) [85] - je standard programovacího jazyka pro heterogenní paralelní programování. Jelikož se jedná jen o definici jazyka, jeho využití není nijak omezeno pro použití libovolného hardware a to jak CPU tak GPU. Klíčová je ovšem podpora (a implementace OpenCL) od výrobců tohoto hardware. Jelikož konzorcium, které definovalo základ tohoto jazyka, bylo tvořeno všemi důležitými hráči na trhu, bude OpenCL zřejmě široce podporováno. V současné době na podpoře OpenCL pro svoje produkty intenzivně pracuje AMD/ATI (jak pro CPU tak pro GPU) i NVIDIA.

Ještě během roku 2009 by se měly objevit první betaverze a kompatibilní ovladače. Hlavní výhodou OpenCL je, že zdrojový kód v tomto jazyce by měl být použitelný na všech platformách (CPU, GPU různých výrobců) i operačních systémech (Windows, Linux, Mac) bez úprav.

- DirectCompute - jedná se o část nové verze DirectX 11 od společnosti Microsoft a jedná se o rozhraní pro vysoce paralelní výpočty na (nejen) grafických kartách. Podporovány by měly být Windows 7 a nové grafické karty kompatibilní s DirectX 11. Možná, že bude možné i rozšířena na Windows Vista a grafické karty s DirectX 10.1. V současné době není více informací k dispozici.

8 Závěr

Tato disertační práce se zabývá diskriminativním trénováním akustických modelů pro systémy automatického rozpoznávání řeči. Jelikož je v současné době většina těchto systémů založena na skrytých Markovových modelech, je tato práce zaměřena na trénování právě těchto modelů.

V úvodní části práce byl představen systém automatického rozpoznávání řeči, tak jak je v současné době nejčastěji používán. Byly podrobně popsány jeho jednotlivé části. V následující kapitole bylo podrobněji popsáno akustické modelování založené na maximalizaci věrohodnosti, ze kterého většina diskriminativních metod trénování vychází.

V další již rozsáhlejší kapitole byly postupně podrobně přestaveny a popsány různé metody diskriminativního trénování, včetně jejich četných modifikací. Zároveň, kde to bylo možné, byly i uváděny konkrétní výsledky, které byly dosaženy pomocí těchto metod na pracovištích ve světě. V závěru této kapitoly byly porovnány jednotlivé metody mezi sebou a výsledky diskutovány.

V praktické části práce bylo navrženo hned několik metod diskriminativního trénování a jejich modifikací. Některé z těchto metod dosáhly prokazatelně lepších výsledků než dosud publikované varianty. Jednalo se zejména o stabilizaci výpočtu a diskriminativní určení vah jednotlivých složek modelu.

Navržené metody v této práci označované jako MMI, MMI-FD, MMI+MMI-FD, MMI-TF byly porovnány na mnoha experimentech a několika korpusech. Pro velké úlohy, kde je akustický model již velice komplexní, se nejlépe hodí metoda MMI-FD, u které bylo navrženo natolik vhodné obecné nastavení, které dobře funguje na všech testovaných korpusech a úlohách. U této metody, na rozdíl od ostatních (zde testovaných i jinde publikovaných), nedochází k snižování efektivnosti s nárůstem velikosti akustického modelu, naopak u několika experimentů bylo prokázáno, že na větších modelech funguje ještě lépe než na modelech menších. U experimentů na úloze titulkování parlamentních přenosů se snížil efekt diskriminativního trénování jen nepatrně i u modelu s dvojnásobným počtem stavů. Dále na této úloze bylo prokázáno, že tato metoda dosahuje podobného relativního snížení chyby jak u zerogramového testu, tak u testu s bigramovým modelem, který obsahoval 120 tisíc slov. Ostatní metody se spíše hodí na úlohy menšího rozsahu, kde akustický model není tak komplexní. Z těchto metod dosahovala nejlepších výsledků metoda MMI-TF a to jen po jedné iteraci. Do všech těchto metod byla rovněž implementována také diskriminativní adaptace, která dosahuje velmi dobrých výsledků na experimentech se skupinovými modely. Jak diskriminativní

trénování, tak i adaptace byly testovány také na úloze detekce klíčových slov. I zde bylo těmito metodami dosaženo významných zlepšení.

Velká pozornost byla také věnována návrhu optimalizačních technik pro implementace těchto metod, tak aby výsledky byly k dispozici v co nejkratším čase, bez extrémních nároků na výpočetní výkon a paměť počítače. Vzhledem k tomu, že trénování probíhá iterativně z několika desítek či dokonce stovek hodin trénovacích nahrávek, mohl by trénovací čas snadno narůstat na řádově týdny až měsíce, což je v praxi neakceptovatelné. Optimalizovaná verze diskriminativního trénování, implementovaná v rámci této práce je však schopná natrénovat akustický model za čas 5- až 50-krát kratší než je celková doba trénovacích nahrávek. Konkrétní doba samozřejmě závisí na velikosti modelu a použitém počítači.

Výsledkem této práce je i softwarový nástroj pro diskriminativní trénování, kde jsou implementovány všechny výše uvedené metody. Ten je možné spouštět jak pod Windows tak pod Linuxem. Navíc podporuje distribuovaný výpočet - tedy zpracovaní velkého množství dat lze rozdělit mezi několik počítačů. Tento software lze snadno rozšířit o další metody, v současné době také umožňuje i výpočet různých statistik, které pak mohou být využity pro různé metody adaptace či transformace příznakových vektorů. Tento software je úspěšně používán v současné době pro trénování modelů v rámci projektů řešených na katedře. Jedná se zejména o projekty "Eliminace jazykových bariér handicapovaných diváků České televize" (MŠMT 2C06020), "Rozpoznávání mluvené řeči v reálných podmínkách" (GAČR 102/08/0707), "Překlenutí jazykové bariéry komplikující vyšetřování financování terorismu a závažné finanční kriminality" (MV VD20072010B160) a "Automatické vyhledávání klíčových slov v proudu zvukových dat" (GA AV ČR 1QS101470516).

Seznam literatury

[1] Psutka, J., Müller, L., Matoušek, J., Radová, V.: *Mluvíme s počítačem česky*, Academia, Praha, 2006, ISBN 80-200-1309-1.

[2] Psutka, J.V.: *Techniky parametrizace, dekorelace a redukce dimenze příznaků v systémech rozpoznávání řeči*, Ph.D. thesis, Západočeská univerzita v Plzni, 2007.

[3] Trmal, J., Vaněk, J., Müller, L., Zelinka, J.: *Independent Components for Acoustic Modeling* Proc. INTERSPEECH, Vol. 1, pp. 2486-2489, 2006.

[4] Radová, V., Psutka, J.: *UWB S01 Corpus – A Czech Read-Speech Corpus* Proc. ICSLP, Beijing, China, pp. 732–735, 2000.

[5] Pražák, A., Psutka, J.V., Hoidekr, J., Kanis, J., Müller, L., Psutka, J.: *Automatic online subtitling of the Czech parliament meetings* Lecture Notes in Artificial Intelligence, pp. 501-508, Springer, Berlin, 2006.

[6] Zelinka J.: *Audio-visuální rozpoznávání řeči* Ph.D. thesis, Západočeská univerzita v Plzni, 2009.

[7] Vaněk J., Psutka J.V., Zelinka J., Pražák A., Psutka J.: *Discriminative Training of Gender-Dependent Acoustic Models* Lecture Notes in Artificial Intelligence, pp. 331-338, Springer, Berlin, 2009.

[8] Vaněk J., Psutka J.V., Zelinka J., Pražák A., Psutka J.: *Training of Speaker-Clustered Acoustic Models for Use in Real-Time Recognizers* Proc. SIGMAP, Milan, Italy, 2009.

[9] Vapnik V.: *Estimation of Dependences Based on Empirical Data*, Springer-Verlag, Berlín, Německo, 1982.

[10] Jelinek, F., Mercer, R.L.: *Interpolated Estimation of Markov Source Parametres from Sparse Data*, Proc. Workshop on Pattern Recognition in Practice, Amsterdam, 1980.

[11] Jelinek, F.: *Statistical Methods for Speech Recognition*, MIT Press, Cambridge, Londýn, 1997.

[12] Katz, S.: *Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer*, IEEE Transactions on ASSP, Vol. 35, pp. 400-401, 1987.

[13] Ney, H., Martin, S., Wessel, F.: *Statistical Language Modeling Using Leaving-One-Out*, Corpus-Based Statistical Methods in Speech and Language Processing, Kluwer Academic Publishers, London, pp. 174-207, 1997.

[14] Yu, D., Deng, L., He, X., Acero, A.: *Use of Incrementally Regulated Descriminative Margins in MCE Training for Speech Recognition*, Proc. ICSLP, Pittsburg, USA, 2006.

[15] Li, X., Jiang, H.: *Solving Large Margin Estimation of HMMs via Semidefinite Programing*, Proc. ICSLP, Pittsburg, USA, 2006.

[16] Du, J., Liu, P., Soong, F.K., Zhou, J.L., Wang, R.H.: *Minimum Divergence Based Diskriminative Training*, Proc. ICSLP, Pittsburg, USA, 2006.

[17] Li, J., Yuan, M., Lee, C.H.: *Soft Margin Estimation of Hidden Markov Model Parameters*, Proc. ICSLP, Pittsburg, USA, 2006.

[18] Yin, Y., Jiang, H.: *A Fast Optimization Method for Large Margin Estimation of HMMs Based on Second Order Cone Programming*, Proc. ICSLP, Antwerp, Belgium, 2007.

[19] Li, J., Lee, C.H.: *Soft Margin Feature Extraction for Automatic Speech Recognition*, Proc. ICSLP, Antwerp, Belgium, 2007.

[20] Vapnik V.: *The Nature of Statistical Learning Theory*, Springer-Verlag, Berlín, Německo, 1995.

[21] Müller, L.: *Dekódovací techniky rozpoznávání souvislé řeči*, Habilitační práce, Západočeská univerzita v Plzni, 2002.

[22] Jelinek, F., Bahl, L.R., Mercer, R.L.: *Design of a Linguistic Statistical Decoder for the Recognition of Continuous Speech*, IEEE Transactions on IT, no 3., vol. 21, pp. 250-256, 1975.

[23] Young, S. et al.: *The HTK Book*, User's Manual, Entropic Inc., 1999.

[24] Hermansky, H.: *Perceptual Linear Predictive (PLP) Analysis of Speech*, J. Acoust. Soc. Am., vol. 87, pp. 1738-1752, USA, 1990.

[25] Brown, P.: *The Acoustic-Modelling Problem in Automatic Speech Recognition*, Ph.D. thesis, Carnegie-Mellon University, 1987.

[26] Bahl, L.R., Brown, P.F., de Souza, P.V., Mercer, R.L.: *Maximum Mutual Information Estimation of Hidden Markov Model Parameters for Speech Recognition*, Computer Speech & Language, Vol. 10, pp. 249-264, 1986.

[27] Gopalakrishnan, P.S., Kanevsky, D., Ndas, A., Nahamoo, D.: *An Inequality for Rational Functions with Applications to Some Statistical Estimation Problems*, IEEE Transactions on Information Theory, Vol. 37, pp. 107-113, 1991.

[28] Valchev, V.: *Diskriminative Methods in HMM-based Speech Recognition*, Ph.D. thesis, University of Cambridge, 1995.

[29] Valchev, V., Odell, J.J., Woodland, P.C., Young, S.J.: *MMIE Training of Large Vocabulary Recognition Systems*, Speech Communications, Vol. 22, pp. 303-314, 1997.

[30] Normandin, Y., Morgera, D.: *An Improved MMIE Training Algorithm for Speaker-Independent Small Vocabulary, Continuous Speech Recognition*, Proc. ICASSP, pp. 537-540, 1991.

[31] Normandin, Y.: *Hidden Markov Models, Maximum Mutual Information Estimation and the Speech Recognition Problem*, Ph.D. thesis, McGill University, 1991.

[32] Kapadia, S.: *The Acoustics-Modelling Problem in Automatic Speech Recognition*, Ph.D. thesis, University of Cambridge, 1998.

[33] Merialdo, B.: *Phonetic Recognition Using Hidden Markov Models and Maximum Mutual Information Training*, Proc. ICASSP, vol. 1, pp. 111-114, 1988.

[34] Woodland, P.C., Povey, D.: *Large Scale Discriminative Training of Hidden Markov Models for Speech Recognition*, Computer Speech and Language, Vol. 16, pp. 25-47, 2002.

[35] Povey, D., Woodland, P.C.: *Frame Discrimination Training of HMMs for Large Vocabulary Speech Recognition*, Proc. ICASSP, vol. 1, pp.333-336, 1999.

[36] Povey, D., Woodland, P.C.: *Minimum Phone Error and I-Smoothing for Improved Discriminative Training*, Proc. ICASSP, Orlando, USA, 2002.

[37] Woodland, P.C., Povey, D.: *Large Scale MMIE Training for Conversational Telephone Speech Recognition*, Proc. NIST Speech Transcription Workshop, College Park, USA, 2000.

[38] Woodland, P.C., Povey, D.: *Large Scale Discriminative Training for Speech Recognition*, Proc. ISCA ITRW ASR2000, 2000.

[39] Hain, T., Woodland, P.C., Evermann, G., Povey, D.: *The CU-HTK March 2000 Hub5E Transcription System*, Proc. NIST Speech Transcription Workshop, College Park, USA, 2000.

[40] Chow, Y.L.: *Maximum Mutual Information Estimation of HMM Parametres for Continuous Speech Recognition Using the N-Best Algorithm*, Proc. ICASSP, Albuquerque, New Mexico, USA, 1990.

[41] Povey, D.: *Implementation of Frame Discrimination on a Large Task*, Master thesis, Cambridge University, 1999.

[42] Povey, D.: *Discriminative Training for Large Vocabulary Speech Recognition*, Ph.D. thesis, Cambridge University, 2003.

[43] Leggetter, C.J., Woodland, P.C.: *Maximum Likelihood Linear Regression for Speaker Adaptation of Continuous Density HMMs*, Cpmputer Speeech and Language, vol. 9, pp. 171-186, 1995.

[44] Padmanabhan M., Saon, G., Zweig, G.: *Lattice-based unsupervised MLLR for speaker adaptation* Proc. ISCA ITRW ASR2000, vol. 1, pp. 128-131, 2000.

[45] Machlica L., Zajíc, Z., Pražák, A.: *Methods of Unsupervised Adaptation in Online Speech Recognition* Proc. SPECOM, St. Petersburg, Rusia, 2009.

[46] Gunawardana A., Byrne, W.: *Discriminative Speaker Adaptation with Conditional Maximum Likelihood Linear Regression* Proc. EUROSPEECH, vol 1., pp. 1203-1206, 2001.

[47] Tsakalidis S., Doumpiotis, V., Byrne, W.: *Discriminative Linear Transforms for Feature Normalization and Speaker Adaptation in HMM Estimation* IEEE Transactions on Speech and Audio Processing, vol 13., pp. 367-376, 2005.

[48] Gauvain, J.L., Lee, C.: *Maximum A Posteriori Estimation for Multivariate Gaussian Mixture Observations of Markov Chains* IEEE Transactions on Speech and Audio Processing, vol 2., pp. 291-299, 1994.

[49] Povey, D., Woodland, P.C., Gales, M.J.F.: *Discriminative MAP for Acoustic Model Adaptation* Proc. ICASSP, Hong Kong, 2003.

[50] Povey, D., Gales, M.J.F., Kim, D.Y., Woodland, P.C.: *MMI-MAP and MPE-MAP for Acoustic Model Adaptation* Proc. EUROSPEECH, Geneva, 2003.

[51] McDonough, J., Schaaf, T., Waibel, A.: *On Maximum Mutual Information Speaker-Adapted Training* Proc. ICASSP, Orlando, Florida, USA, 2002.

[52] Uebel, L.F., Woodland, P.C.: *Discriminative Linear Transform for Speaker Adaptation* Proc. ITRW ASR2001, 2001.

[53] Anastasakos, J., McDonough, J., Schwarz, R., Makhoul, J.: *A Compact Model for Speaker-Adaptive Training* Proc. ICSLP, vol. 1, pp. 1137-1140, 1996.

[54] Wang, L., Woodland, P.C.: *Discriminative Adaptation and Adaptive Training* Proc. EARS STT Workshop, 2003.

[55] Zheng, J., Stolcke, A.: *Improved Discriminative Training. Using Phone Lattices* Proc. Eurospeech, Lisbon, 2005.

[56] Huang, Kingsbury, B., Mangu, L., Saon, G., Sarykaja, R., Zeig, G.: *Improvements to the IBM Hub-5E System* Proc. NIST RT-02 Workshop, 2002.

[57] Povey, D., Kingsbury, B., Mangu, L., Saon, G., Zweig G.: *fMPE: Discriminatively Trained Features for Speech Recognition* Proc. ICASSP, Philadelphia, USA, 2005.

[58] Povey, D., Kanevsky, D., Kingsbury, B., Ramabhadran B., Saon, G., Visweswariah, K.: *Boosted MMI for Model and Feature-Space Discriminative Training* Proc. ICASSP, Las Vegas, USA, 2008.

[59] Vertanen, K.: *An Owerview of Discriminative Training for Speech Recognition* Technical Report, Cambridge University, 2004.

[60] Normandin, Y.: *Optimal Splitting of HMM Gaussian Mixture Components with MMIE Training* Proc. ICASSP, vol. 1, pp. 449-452, 1995.

[61] Chou, W., Juang, B.-H., Lee, C.-H.: *Segmental GDP Training of HMM Based Speech Recognizer*, Proc. ICASSP, vol. 1, pp. 473-476, 1992.

[62] Juang, B.-H., Katagiri, S.: *Discriminative Learning for Minimum Error Classification*, IEEE Transactions on Acoustic, Speech and Signal Processing, vol. 40, no. 12, pp. 3043-3054, 1992.

[63] Katagiri, S., Lee, C.-H., Juang, B.-H.: *New Discriminative Training Algorithms Based on the Generalized Descent Method*, Proc. IEEE Neural Networks for Signal Processing, pp. 299-308, 1991.

[64] Schlueter, R., Macherey, W., Muller, B., Ney, H.: *Comparison of Discriminative Training Criteria and Optimization Methods for Speech Recognition*, Speech Communication, vol. 34, pp. 287-310, 2001.

[65] McDermott, E., Katagiri, S.: *A Derivation of Minimum Classification Error from the Theoretical Classification Risk Using Parzen Estimation*, Computer Speech and Language, 2004.

[66] McDermott, E., Katagiri, S.: *Prototype Based Discriminative Training for Various Speech Units*, Computer Speech and Language, 1994.

[67] Katagiri, S., Lee, C.-H., Juang, B.-H.: *A Generalized Probabilistic Descent Method*, Proc. Acoustic Society of Japan, pp. 141-142, 1990.

[68] McDermott, E.: *Discriminative Training for Speech Recognition*, Ph.D. thesis, Waseda University, 1997.

[69] Schlueter, R., Macherey, W., Kanthak, S., Ney, H.: *Comparison of Optimization Methods for Discriminative Training Criteria*, Proc. EUROSPEECH, vol. 1, pp. 15-18, 1997.

[70] Li, Q., Juang, B.-H.: *A New Algorithm for Fast Discriminative Training*, Proc. ICASSP, vol. 1, pp. 97-100, 2002.

[71] He, X., Chou, W.: *Minimum Classification Error Linear Regression for Acoustic Model Adaptation of Continuous Density HMMs*, Proc. ICASSP, vol. 1, pp. 556-559, 2003.

[72] Amari, S.-I.: *A Theory of Adaptive Pattern Classifiers*, IEEE Transactions on Electronic Computers, vol. EC-16, pp. 299-307, 1967.

[73] Bottou, L.: *Une Approche theorique de l'apprentissage connectionniste: application a la Reconnaissane de la Parole*, Ph.D. thesis, University of Paris Sud, 1991.

[74] Fahlman, S.E.: *An Empirical Study of Learning Speed in Back-Propagation Networks*, Carnegie-Mellon University, Tech. Report, 1988.

[75] McDermott, E., Katagiri, S.: *String-Level MCE for Continuous Phoneme Recognition*, Proc. EUROSPEECH, vol. 1, pp. 123-126, 1997.

[76] Battiti, R.: *First- and Second- Order Methods for Learning: Between Steepest Descent and Newton's Method*, Neural Computation, vol. 4, pp. 141-166, 1992.

[77] Leroux, J, McDermott, E.: *Optimization Methods for Discriminative Training*, Proc. EUROSPEECH, 2005.

[78] Riedmiller, Braun, H.: *A Direct Adaptive Method for Fater Backpropagation Learning: The RPROP Algorithm*, Proc. IEEE ICNN, vol. 1, pp. 586-591, 1993.

[79] McDermott, E., Hazen, T.J., Leroux, J., Nakamura, A., Katagiri, S.: *Discriminative Training for Large Vocabulary Speech Recognition Using Minimum Classification Error*, IEEE Transaction of Audio Speech Language Process., vol. 15, pp. 203-223. 2007.

[80] Macherey, W., Haferkamp, L., Schlueter, R., Ney, H.: *Investigation on Error Minimizing Training Criteria for Discriminative Training in Automatic Speech Recognition*, Proc. INTERSPEECH, Lisbon, 2005.

[81] Price, P., Fisher, W., Bernstein, J., Pallett, D.: *The DARPA 1000-word Resource Management database for Continuous Speech recognition*, Proc. IEEE ICASSP, pp. 651-654, 1998.

[82] AMD/ATI Stream computing SDK, *http://ati.amd.com/technology/streamcomputing*

[83] NVIDIA CUDA development tools, *http://www.nvidia.com/cuda*

[84] Buck, I., Foley, T., Horn, D., Houston, M.: *Brook for GPUs: Stream Computing on Graphics Hardware*, Proc. SIGGRAPH, Los Angeles, USA, 2004.

[85] Khronos Group: OpenCL - Open Computing Language, *http://en.wikipedia.org/wiki/OpenCL*