Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra kybernetiky

BAKALÁŘSKÁ PRÁCE

PLZEŇ, 2016 **MARTIN JAHN**

Prohlášení

Předkládám tímto k posouzení a obhajobě bakalářskou práci zpracovanou na závěr studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni.

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

V Plzni dne 5. května 2016

……

Poděkování

Rád bych poděkoval vedoucímu této bakalářské práce Ing. Mgr. Josefu Psutkovi, Ph.D. za jeho podporu a čas, který mi věnoval při řešení této práce.

Abstrakt

Tato bakalářská práce se zabývá přípravou dat pro tvorbu akustického modelu z webu České televize. Je zde popsána statická metoda rozpoznávání řeči s využitím Markovových modelů. V práci se dále nachází zevrubný návod na přípravu dat a trénování akustického modelu pomocí HTK spolu s ukázkami jednotlivých kroků. Důležitým bodem práce je analýza formátu dat na webu ČT a následná tvorba souborů k trénování. Závěr práce se věnuje metodám synchronizace titulků.

Abstract

This work deals with the preparation of data for an acoustic modelling from the Czech Television web. It contains a description of the statistic method of speech recognition that uses Markov models. Very detailed instructions for preparation and training of the acoustic model and samples of the individual steps are described here. A very important part of this work analyzes structure of the data from the Czech Television web and shows a procedure creating files for the training of the acoustic model. The last part focuses on methods that synchronize closed captions.

Klíčková slova

akustický model, rozpoznávání řeči, skryté Markovovy modely, pravděpodobnost, skryté titulky

Keywords

acoustic model, speech recognition, Hidden Markov Models, probability, closed captions

Obsah:

Úvod

Rozpoznávání mluvené řeči je zcela automatický převod řeči do textu. Tento proces není v žádném případě triviální, a to je důvod, proč se ještě nepodařilo vyvinout takový stroj, který by bez problémů rozpoznával lidskou promluvu.

Během mého studia jsem absolvoval několik předmětů, které se touto problematikou zabývaly a velice mě zaujaly. To byl jeden z hlavních důvodů, proč jsem si vybral téma své práce právě z tohoto oboru.

Tato bakalářská práce se zabývá přípravou dat z webu České televize a tvorbou akustického modelu pomocí nástroje HTK (Hidden Markov Model Toolkit). Data budou získána na webu ČT, kde se nachází velké množství různých pořadů se skrytými titulky, např. sportovní pořady, dokumenty, zábavné pořady, zprávy apod. Tyto pořady jsou komentovány různými řečníky a odehrávají se v různých akustických prostředích. Právě kvůli této vlastnosti slouží jako výborná trénovací data pro tvorbu akustického modelu, jehož cílem je zlepšit funkčnost stávajícího systému pro titulkování živých pořadů ČT. Tento problém je jedním z hlavních důvodů dlouholeté spolupráce katedry kybernetiky v Plzni s ČT.

Takto získaná data bude potřeba upravit do požadovaných formátů, aby se z nich dal natrénovat akustický model. To znamená, že bude navržen postup automatického zpracování těchto dat, který bude následně ověřen natrénováním jednoduchého akustického modelu a následným rozpoznávacím experimentem. S největší pravděpodobností se ukáže, že jednotlivé titulky nebudou odpovídat zvukové stopě. Tento problém by mohl mít neblahý vliv na úspěšnost rozpoznávání, a proto bude muset být nalezeno jeho řešení. Všechny získané výsledky budou zaznamenány a vzájemně porovnány.

1. Teoretický úvod

1.1 Problémy automatického rozpoznávání řeči

Pojem rozpoznávání řeči znamená automatický převod mluvené řeči do textové podoby. Touto problematikou se zabývá spousta výzkumných týmu již několik desítek let. Za tuto dobu vznikla spousta systémů, které jsou schopny rozpoznat jak izolovaná slova (např. číslovky), tak také souvislou promluvu. Problém rozpoznávání izolovaných slov je dnes již historií a většina výzkumných týmů se zabývá problematikou rozpoznávání spojité promluvy. Rozpoznávání souvislé promluvy je logicky daleko těžší a přináší s sebou více problémů.

a) Odlišný hlas více řečníků
   Každý člověk má jiný hlas. To je způsobeno odlišnou stavbou hlasového ústrojí, tj. odlišný tvar nosní dutiny, odlišná frekvence kmitání hlasivek apod. Díky tomuto faktu nikdy nenajdeme dva řečníky, kteří by měli naprosto shodný hlas, a to komplikuje úspěšnost rozpoznávání.
b) Odlišnost hlasu jednoho řečníka v různých situacích
   Každý člověk má jinou barvu hlasu v různých situacích. To znamená, že stejná promluva bude jinak znít, když se bude člověk smát a jinak, když bude rozčílený. To vše se odrazí na změně řečového signálu a logicky také na kvalitě rozpoznání promluvy.
c) Měnící se akustické pozadí během promluvy
   Během promluvy může být řečový signál rušen okolním šumem (např. komentování sportovních utkání v hlučném prostředí, telefonování v metru, apod.). Vysoká přítomnost šumu výrazně stěžuje identifikaci začátku a konce slova nebo dokonce rozpoznání vyslovených slov.
d) Spontánně pronášená promluva
   Je velký rozdíl, jestli je rozpoznávaná promluva předem připravená, tj. čtená řeč anebo předem nepřipravená, tj. spontánní řeč. Během spontánní promluvy používá řečník velké množství „vycpávkových slov“(např. a, no, hm, apod.), jsou slyšet hlasité nádechy, člověk se „zakoktává“ a nějaké slovo řekne i vícekrát za sebou. Člověk také při spontánní promluvě používá velmi často nespisovné výrazy, zejména nespisovné koncovky u slov (např. kterej, významnej, apod.). Všechny tyto aspekty výrazně stěžují rozpoznávání.

1.2 Metody automatického rozpoznávání řeči

Automatické rozpoznávání souvislé řeči je mnohem komplikovanější než rozpoznávání izolovaných slov. Při rozpoznávání jednotlivých slov, příkazů (např. hlasové zadávání příkazů do navigace) se řečník snaží o správnou a srozumitelnou výslovnost každého slova. Také rozpoznávací slovník je o poznání menší než při rozpoznávání souvislé řeči. V současné době existuje několik metod rozpoznávání řeči.

1.2.1 Metody porovnávání se vzory

Mezi těmito metodami se nejvíce využívá metoda Dynamického borcení času (angl. Dynamic Time Warping, zkr. DTW). Při rozpoznávání je slovo zpracováváno jako celek. Metoda hledá ve slovníku jeho vzor, ke kterému má slovo nejblíže (nejlepší shoda). Vzdálenost mezi slovem a jeho vzorem se určuje pomocí dynamického programování. Tato metoda je vhodná pro rozpoznávání jednotlivých slov.

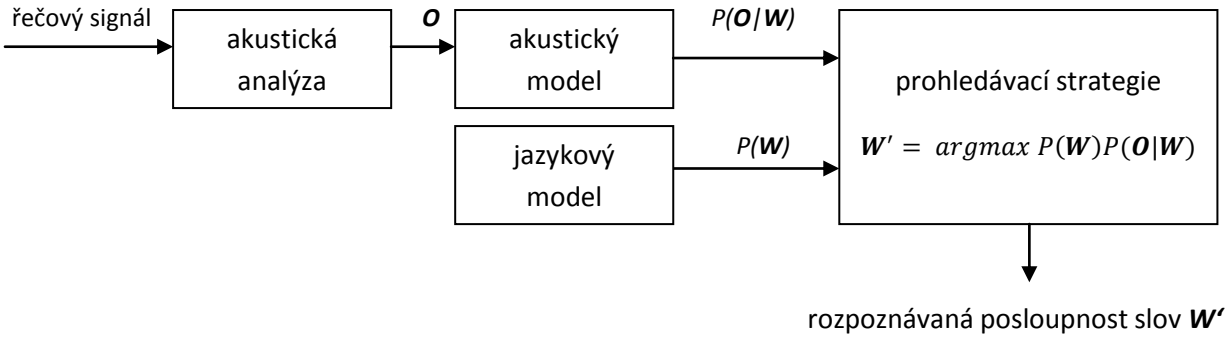
1.2.2 Metody využívající statistický přístup

Statistický přístup využívá dvě hlavní součásti, tj. akustický procesor a lingvistický dekodér. Předpokládejme, že $\boldsymbol{W} = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{w}_3, \ldots \boldsymbol{w}_M\}$ je posloupnost slov a $\boldsymbol{O} = \{\boldsymbol{o}_1, \boldsymbol{o}_2, \boldsymbol{o}_3, \ldots \boldsymbol{o}_L\}$ je posloupnost vektorů příznaku. Akustický procesor transformuje řečový signál na vektorovou posloupnost příznaků a lingvistický dekodér se snaží rozpoznat, jaká slova byla řečena. Cílem je nalézt nejpravděpodobnější posloupnost slov $\boldsymbol{W}'$ pro vektor příznaků $\boldsymbol{O}$. Zde se využívá Bayesova vztahu pro podmíněnou pravděpodobnost:

$$\boldsymbol{W}' = \underset{\boldsymbol{w}}{argmax}\, P(\boldsymbol{W}|\boldsymbol{O}) = \underset{\boldsymbol{w}}{argmax}\, \frac{P(\boldsymbol{W})\, P(\boldsymbol{O}|\boldsymbol{W})}{P(\boldsymbol{O})}$$

kde $P(\boldsymbol{W})$ je apriorní pravděpodobnost slov $\boldsymbol{W}$, $P(\boldsymbol{O}|\boldsymbol{W})$ je pravděpodobnost, že za podmínky vyslovení posloupnosti slov $\boldsymbol{W}$ bude generována posloupnost vektorů příznaků $\boldsymbol{O}$ a $P(\boldsymbol{O})$ je apriorní pravděpodobnost vektorů příznaků $\boldsymbol{O}$. Jelikož pravděpodobnost $P(\boldsymbol{O})$ není funkcí $\boldsymbol{W}$, můžeme jí vynechat. Posloupnost slov $\boldsymbol{W}'$ lze tedy určit pomocí maximalizace sdružené pravděpodobnosti $P(\boldsymbol{W},\boldsymbol{O})$

$$\boldsymbol{W}' = \underset{\boldsymbol{w}}{argmax}\, P(\boldsymbol{W}, \boldsymbol{O}) = \underset{\boldsymbol{w}}{argmax}\, P(\boldsymbol{W}) P(\boldsymbol{O}|\boldsymbol{W})$$



Obr. 1.1 Blokové schéma systému rozpoznávání řeči

1.3 Akustické modelování

Akustický model je velice důležitou částí systému rozpoznávání řeči, který má za úkol co nejpřesněji a nejrychleji odhadnout podmíněnou pravděpodobnost $P(\boldsymbol{O}|\boldsymbol{W})$ pro jakýkoliv vektor příznaků $\boldsymbol{O}$ a každou uvažovanou posloupnost slov. Skupina metod, založených na statistickém přístupu, modeluje slova a souvislé promluvy pomocí skrytých Markovových modelů (angl. Hidden Markov Model, zkr. HMM). Princip této metody vychází z představy generování řeči člověkem. Při tomto ději je hlasové ústrojí během velmi krátké chvíle (např. 10 ms) nastaveno pro vyslovení určité hlásky (je ve stacionárním stavu). V tomto stavu je následně hlasovým ústrojím vytvářen krátký signál, jehož vlastnosti závisí na nastavení hlasového a artikulačního ústrojí. Každé slovo může být modelováno jako celek jedním skrytým Markovovým modelem, ale daleko častěji jsou slova rozdělena na menší části (slabiky, fonémy, apod.) a každá tato část má svůj vlastní skrytý Markovův model. Celé slovo potom vznikne zřetězením těchto subslovních modelů.

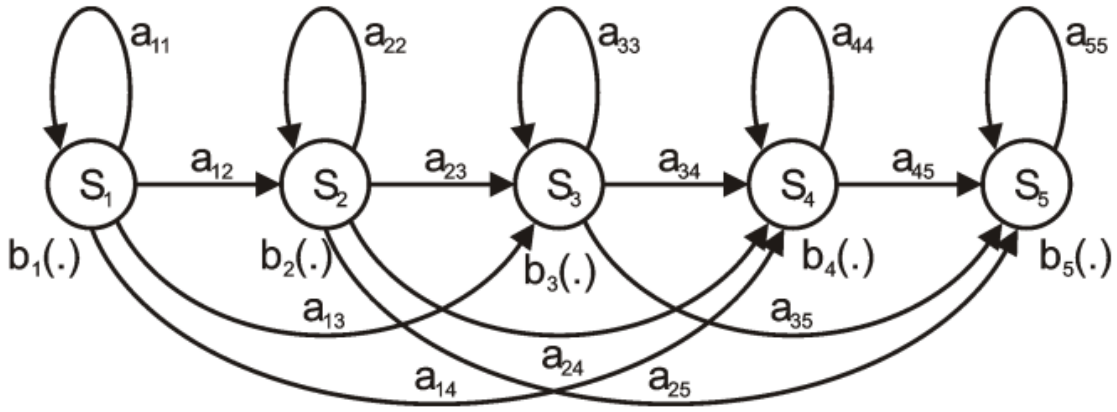
1.3.1 Struktura skrytého Markovova modelu

Skrytý Markovův model je zvláštním případem stochastických konečných automatů, který v diskrétních časových okamžicích generuje náhodnou posloupnost vektorů příznaků $\boldsymbol{O} = \{\boldsymbol{o}_1, \boldsymbol{o}_2, \boldsymbol{o}_3, \ldots \boldsymbol{o}_L\}$. V každém časovém kroku mění (emituje) model svůj stav s_i podle předem daných pravděpodobností přechodů a_{ij}. Stav s_i, do kterého model přejde, potom generuje příznakový vektor $\boldsymbol{o}_i$. Pravděpodobnosti přechodu a_{ij} jsou podmíněné a určují, s jakou pravděpodobností přechází model ze stavu s_i do stavu s_j. Součet všech pravděpodobnostních přechodů a_{ij} je roven jedné

$$\sum_{j=1}^{N} a_{ij} = 1$$

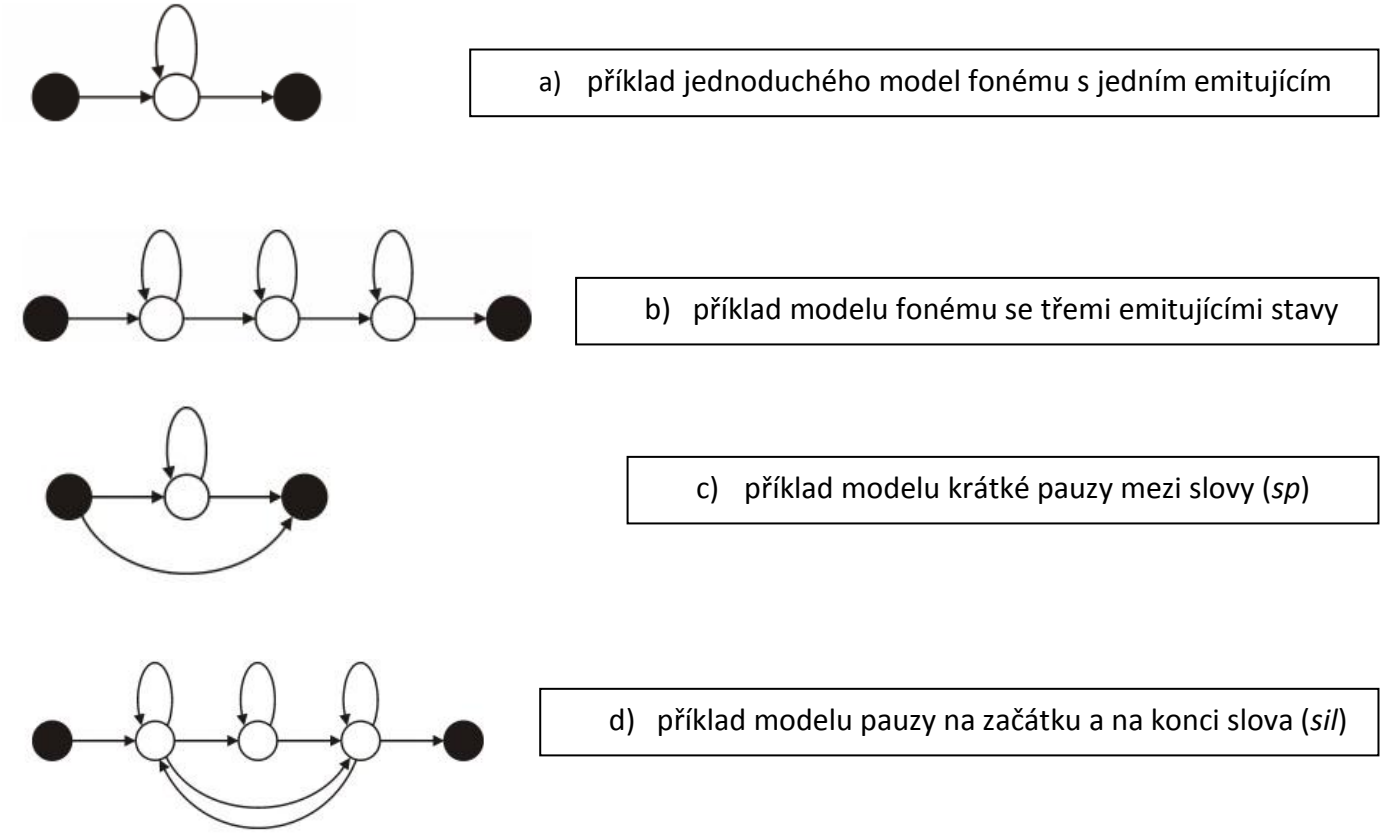
Při modelování souvislé řeči se využívají zejména levo-pravé Markovovy modely (angl. left-to-right models), které jsou velice výhodné pro modelování procesů, jejichž vývoj je spojen

s postupujícím časem. Proces začíná u počátečního stavu modelu, kam přijde první spektrální vzor a postupně přechází na stavy s vyššími indexy nebo zůstává ve stejném stavu, dokud nedosáhne koncového stavu. Průchod modelem jde tedy ve směru zleva doprava.



Obr. 1.2 Pětistavový skrytý Markovův model slova.

Skrytý Markovův model může mít více různých podob. Na obrázku 1.3 jsou znázorněny některé jeho možné struktury.



Obr. 1.3 Struktura modelů fonémů, krátké mezislovní pauzy (*sp*) a dlouhé pauzy (*sil*).

Slova a celé promluvy jsou, jak již bylo řečeno výše, vytvářeny na základě spojování (zřetězení) modelů fonémů a to takovým způsobem, že neemitující konečný stav modelu fonému a neemitující počáteční stav následujícího fonému splynou v jeden neemitující stav. V praxi se velmi často využívají pětistavové modely, které modelují fonémy nebo trifony, tj. foném zohledňující levý a pravý kontext.

Monofonový kontext pro promluvu: „Sběr odpadu“
sil *z b j e r* ***sp*** *o t p a d u* ***sil***

Trifonovy kontext pro promluvu: „Sběr odpadu“
sil *sil-z+b z-b+j b-j+e j-e+r e-r+o* ***sp*** *r-o+t o-t+p t-p+a p-a+d a-d+u d-u+sil* ***sil***

1.4.1 Trénování parametrů skrytého Markovova modelu

Zjišťování hodnot parametrů Markovova skrytého modelu se provádí pomocí statické indukce – jinak řečeno trénováním parametrů na základě anotovaných trénovacích dat, kde je úkolem stanovit hodnoty těchto parametrů modelu. Existují dva způsoby trénování skrytého Markovova modelu. Trénování izolovaných jednotek promluvy a trénování vložených jednotek promluvy. V prvním případě je každý skrytý Markovův model trénován pomocí krátkých, oddělených úseků řečového signálu, kde každý úsek má pevně definované hranice svého začátku a konce. Při rozpoznávání souvislé řeči se však využívá druhý způsob trénování – trénování vložených jednotek. Při tomto procesu se na základě známé promluvy vytvoří příslušný složený skrytý Markovův model a parametry jednotlivých vložených modelů fonémů jsou trénovány v rámci trénování složeného modelu celé promluvy.

Ke stanovení hodnoty parametrů modelu se nejčastěji využívá metoda maximální věrohodnosti (angl. Maximum Likelihood, zkr. ML). Kritérium maximální věrohodnosti předpokládá pravděpodobnostní model ve tvaru $P(x|\lambda)$ s neznámými parametry λ, které se snaží ocenit (najít) pomocí trénovací sady $x_1, x_2,\ldots,x_N$. Tato metoda využívá tzv. Fisherovu funkci věrohodnosti, která má tvar

$$F(x_1, x_2, \ldots, x_N|\lambda) = \prod_{n=1}^{N} P(x_n|\lambda)$$

kde hledáme maximum této funkce přes parametry λ

$$\hat{\lambda} = \underset{\lambda}{argmax} \prod_{n=1}^{N} P(x_n|\lambda)$$

V praxi se raději pracuje s logaritmem funkce

$$\hat{\lambda} = \underset{\lambda}{argmax}\, log \prod_{n=1}^{N} P(x_n|\lambda)$$

Cílem je nalézt optimální hodnoty pravděpodobností přechodu a_{ij} a parametrů hustotních funkcí $b_j(.)$. Toho však z důvodu složitosti úlohy nelze dosáhnout (nemá explicitní řešení). K vyřešení této úlohy slouží EM algoritmy, což jsou vlastně iterativní numerické metody. V úlohách rozpoznávání řeči se skrytými Markovovými modely se využívá Baum-Welchova algoritmu, což je speciální případ EM algoritmů. Velmi podrobný výklad tohoto algoritmu lze najít v [2].

1.4 Jazykové modelování

Jazykový model je vedle akustické analýzy a akustického modelu další velmi důležitou částí celého systému rozpoznávání řeči. Jeho úkolem je poskytnout co nejrychleji a nejpřesněji apriorní pravděpodobnosti $P(\boldsymbol{W})$ pro libovolnou posloupnost slov $\boldsymbol{W}$. Pravděpodobnost $P(\boldsymbol{W})$ lez vyjádřit vztahem

$$P(\boldsymbol{W}) = \prod_{k=1}^{K} P(\boldsymbol{w}_k | \boldsymbol{w}_{k-1} \dots \boldsymbol{w}_1)$$

Systémy rozpoznávání řeči pracují s obrovskými slovníky, a proto není možné pravděpodobnosti $P(\boldsymbol{W})$ dostatečně dobře odhadnout. Velmi často se proto provádí jejich aproximace

$$P(\boldsymbol{W}) \approx \prod_{k=1}^{K} P(\boldsymbol{w}_k | \boldsymbol{w}_{k-1} \dots \boldsymbol{w}_{k-n+1})$$

kdy všechny historie $\boldsymbol{w}_{k-1 \dots}\boldsymbol{w}_{k-n+1}$, které se shodují v posledních *n-1* slovech, jsou zařazeny do stejné třídy. Tyto modely se nazývají n-gramové modely, tj. posloupnost *n* za sebou náhodně jdoucích slov. V praxi se nejčastěji používají bigramy (n = 2) a nebo trigramy (n = 3).

Výhody n-gramových modelů jsou takové, že se dají poměrně snadno odhadnout a jsou vhodné pro jazyky s relativně pevným pořadím slov ve větách, kde existují silné statistické závislosti mezi výskyty za sebou následujících slov.

Naproti tomu hlavním problémem n-gramových modelů je nedostatek trénovacích dat. Například slovník čítající 50 000 slov by měl $50\,000^3 = 1{,}25 * 10^{14}$ trigramů a to je tak obrovské množství trénovacích dat, kterého v podstatě nelze dosáhnout.

V této bakalářské práci se pracuje se zerogramovými modely, tj. že všem slovům je přiřazena stejná pravděpodobnost.

2. Trénování akustických modelů pomocí HTK

2.1 Příprava k trénování

2.1.1 Soubory potřebné k trénování

Před začátkem trénování je nutné vytvořit několik souborů, které jsou potřeba pro trénování akustického modelu pomocí HTK.

a) nahrané trénovací promluvy (.wav soubory)
b) seznam trénovacích promluv pro parametrizaci *param.scp* ve formátu:
   /wav/veta001.wav /htk/veta001.htk
   /wav/veta002.wav /htk/veta002.htk
   …

   kde v levém sloupci je cesta k souborům a názvy souborů, které chceme parametrizovat a v pravém sloupci je cesta k souborům a názvy výsledných parametrizovaných souborů. (Tento soubor by měl obsahovat jak trénovací, tak také testovací data).
   Pozn. Pro úplné pochopení – soubory v levém sloupci musí existovat a jsou to trénovací promluvy ve formátu .wav. Soubory v pravém sloupci se vytvoří během parametrizace do předem vytvořeného adresáře *htk*.

c) seznam testovacích promluv (parametrizovaných) *test.scp* ve formátu:

   /htk/veta001.htk
   /htk/veta002.htk
   …

d) seznam trénovacích promluv (parametrizovaných) *train.scp* ve formátu:

   /htk/veta001.htk
   /htk/veta002.htk
   …

e) soubor s přepisem (transkripcí) všech promluv na úrovni slov *words.mlf* ve formátu:

   #!MLF!#
   "/veta001.lab"*
   měli
   jste
   jednoho
   nepřítele
   .
   "/veta002.lab"*
   zákazník
   volající
   …

Každá věta začíná řetězcem v uvozovkách (např. *"*/veta001.lab"*), který je nápadně podobný jménu souboru. Pokaždé, když HTK pracuje s nahrávkou, respektive s parametrizovanou nahrávkou a potřebuje její transkripci, hledá ji v souboru se stejným jménem, jaké má soubor obsahující nahrávku, ale jinou koncovkou (.lab).
Např. při zpracování souboru */htk/veta002.htk*, se bude přepis této promluvy hledat v souboru */htk/veta002.lab*. Při použití takzvaného *Master Label File* (MLF), hledá HTK všechny transkripce v něm a to tak, že pátrá po řetězci, který odpovídá jménu příslušného souboru s transkripcí. Využívá se tzv. "divoká karta", symbol *, který odpovídá libovolnému řetězci. V našem případě se tedy vybere transkripce uvedená za řetězcem "*/veta002.lab".
Každé slovo v MLF souboru musí být na samostatné řádce a přepis každé věty musí být ukončen tečkou, která je rovněž na samostatné řádce (viz příklad výše).

f) slovník *dict_sp_* ve formátu:

a	*a _sp_*
absence	*a p s e n c e _sp_*
absolutní	*a p s o l u t n I _sp_*
…	
žvýkání	*Z v I k A N I _sp_*

V levém sloupci jsou slova, která se vyskytují v promluvách, v pravém sloupci je jejich fonetická transkripce, vyjádřená posloupností fonů z české fonetické abecedy.

g) soubory *monophones0* a *monophones1* obsahující seznam symbolů fonetické abecedy, které jsou použity při fonetické transkripci. Rozdíl mezi oběma soubory je pouze v tom, že soubor *monophones1* obsahuje navíc symbol krátké pauzy *_sp_*.

2.1.2 Vytváření souborů s přepisem na úrovni fonémů

V této práci nejsou promluvy modelovány na základě slov, nýbrž na základě fonů. Každý fon je reprezentován skrytým Markovovým modelem – HMM (viz kapitola 1.3.1). Jednotlivé fony reprezentují příslušné HMM, a proto je potřeba mít, kromě transkripce promluv na úrovni slov, také transkripci na úrovni fonů. Takovou transkripci lze vytvořit pomocí předem připravené slovní transkripce (*words.mlf*), slovníku výslovností (*dict*) a programu z HLEd z balíku HTK.

*HLEd –l * –d dict_sp.txt –i phones0.mlf mkphones0.led words.mlf*

Tento program je v podstatě editor, který dokáže manipulovat s MLF soubory pomocí sady příkazů.

Parametry programu:

-T 1	Nastavuje úroveň trasování na hodnotu 1. Úroveň trasování určuje, nakolik detailní budou zprávy vypisované programem. Tento parametr lze použít pro všechny programy HTK.
*–l **	Způsobuje, že do jmen souborů v souboru MLF je vložen symbol * ("divoká karta") místo skutečné cesty.
–d dict	Načte slovník výslovností ze souboru *dict_sp_*.
–i phones0.mlf	Výstupní soubor s transkripcí na úrovni fonů.
mkphones0.led	Soubor s příkazy, které má HLEd vykonat. Vytvoříme soubor, který bude obsahovat tyto příkazy: *EX* *IS _sil_ _sil_* *DE _sp_*

Příkaz *EX* způsobí, že každé slovo ze souboru *words.mlf* je nahrazeno příslušnou fonetickou transkripcí nalezenou ve slovníku (*dict*_sp_). Příkaz *IS _sil_ _sil_* vloží fon *_sil_* (silence, dlouhá pauza) na začátek a konec každé věty (promluvy). A příkaz *DE _sp_* odstraní všechny výskyty fonu _sp_ (short pause, krátká pauza), protože pro počáteční fázi trénování nejsou modely *_sp_* potřeba. Poté, co budou modely zhruba natrénovány, bude odvozen model *_sp_* z modelu *_sil_* (viz kapitola 2.2.2). Dále je tedy potřeba další soubor s fonovou transkripcí, který bude obsahovat fony *_sp_*. Ten se dá vyrobit analogickým postupem, tj.:

*HLEd –l * –d dict_sp.txt –i phones1.mlf mkphones1.led words.mlf*

kde jediný rozdíl oproti předchozímu volání programu HLEd je v tom, že *mkphones1.led* obsahuje pouze řádky:

EX

*IS _sil_ _sil_*

Nově vyrobené soubory vypadají následovně:

phones0.mlf
#!MLF!#
"/veta001.lab"*
*_sil_*
m
N
e
l
i
s
t
e
j
e
d
n
o
h
o
n
e
p
R
I
t
e
l
e
*_sil_*
.
…

phones1.mlf
#!MLF!#
"/veta001.lab"*
*_sil_*
m
N
e
l
i
*_sp_*
s
t
e
*_sp_*
j
e
d
n
o
h
o
*_sp_*
n
e
p
R
I
t
e
l
e
*_sp_*
*_sil_*
.
…

2.1.3 Parametrizace řečových dat

Parametrizace znamená převod nahrávek ze zvukového formátu (.wav) na posloupnost vektorů parametrů (LPC, PLP, MFCC). Nahrávky lze sice parametrizovat „za běhu“ při trénování, ale je to časově náročnější. Proto je lepší si nahrávky zparametrizovat předem. K tomuto účelu slouží program HCopy:

HCopy -T 1 -C CF_param.mfc -S param.scp

Parametry programu:

-S param.scp	Seznam souborů pro parametrizaci (viz kapitola 2.1.1).
-C CF.mfc	Obsah konfiguračního souboru *CF.mfc* určuje, v jakém formátu jsou vstupní data a v jakém formátu mají být data výstupní. Jinými slovy říká, co má vlastně HCopy se vstupními soubory udělat.

Parametrizovat se bude do MFCC 13 koeficientů (Melovské frekvenční kepstrální koeficienty, angl. Mel-frequency cepstral coefficients) + delta + delta delta koeficienty (první a druhé časové derivace každého koeficientu). Dostaneme tedy vektor popisující jeden mikrosegment (10-30ms) o velikosti 39 koeficientů.

Výstupní adresář (htk) na zparametrizovaná data je potřeba vytvořit **před** samotnou parametrizací! Stejně tak je výhodné vytvořit najednou výstupní adresáře na trénované modely – *hmm0* až *hmmB* (viz kapitola 2.2.1).

2.2 Tvorba monofonních modelů

V této fázi trénování se bude pracovat s modely monofonů. To znamená, že každý fon z české fonetické abecedy bude reprezentován jedním HMM.

2.2.1 Definice topologie HMM a jejich inicializace

Topologie skrytých Markovových modelů byla již definována v kapitole 1.3.1. Definice takového modelu v HTK formátu vypadá následovně (z konvence se ukládá do souboru s názvem *proto*). Jedná se o pětistavový model.

```
~o <VecSize> 39 <MFCC_0_D_A>
~h "proto"
<BeginHMM>
<NumStates> 5
<State> 2
<Mean> 39
0.0 0.0 0.0 ...
<Variance> 39
1.0 1.0 1.0 ...
<State> 3
<Mean> 39
0.0 0.0 ...
<Variance> 39
1.0 1.0 ...
<State> 4
<Mean> 39
0.0 0.0 ...
<Variance> 39
1.0 1.0 ...
<TransP> 5
0.0 1.0 0.0 0.0 0.0
0.0 0.6 0.4 0.0 0.0
0.0 0.0 0.6 0.4 0.0
0.0 0.0 0.0 0.7 0.3
0.0 0.0 0.0 0.0 0.0
<EndHMM>
```

Parametr *<VecSize>* určuje délku vektoru parametrů (39) a jejich typ *(<MFCC_0_D_A>*). Zkratka *MFCC_0_D_A* znamená, že jsou používány Melovské kepstrální koeficienty (*MFCC*) s nultým koeficientem (*_0*) plus delta koeficienty (*_D*) plus akcelerační (delta-delta) koeficienty (*_A*). Z toho vyplývá také uvedená délka vektoru parametrů 3*13 MFCC.

Za řetězcem *~h* následuje jméno modelu (v tomto případě *proto* neboli prototyp). Řetězec <BeginHMM> uvozuje začátek modelu, <NumStates> značí počet stavů modelu.
Pravděpodobnost generování jednotlivých vektorů parametrů v emitujících stavech se řídí Gaussovým rozdělením pravděpodobnosti, u kterého potřebujeme znát střední hodnotu a kovarianci, respektive vektor středních hodnot a kovarianční matici. Často se ale používá diagonální kovarianční matice, kde stačí uchovávat vektor diagonálních prvků.
Sekvence:

<State> 3
<Mean> 39
0.0 0.0 0.0 ...

tedy značí, že Gaussovo rozdělení pro třetí stav HMM má vektor středních hodnot o délce 39 prvků (<*Mean 39*>), jejichž seznam je o řádku níž.
Za řetězcem *<TransP>* následuje údaj o velikost matice přechodů a pod ním vlastní matice přechodu. V matici je důležité, zda je na konkrétní pozici nulová či nenulová hodnota. Je-li $a_{ij} > 0$, pak je definován přechod ze stavu *i* do stavu *j*. V případě, kdy $a_{ij} = 0$, přechod ze stavu *i* do stavu *j* definován není.
Střední hodnota a kovariance prototypu HMM je přepočítána pomocí programu HCompV:

HCompV –C CF.mfc –f 0.01 –m –S train.scp –M hmm0 proto

Tento program spočítá celkovou střední hodnotu a kovarianci ze všech trénovacích dat a nastaví všechny Gaussova rozdělení v modelu prototypu (proto) na právě spočtené hodnoty.

Parametry programu:

-*C CF.mfc* Konfigurační soubor, který obsahuje důležitá nastavení.

-*f 0.01* Způsobí vytvoření makra *vFloor1*, které obsahuje dolní mez kovariance.

-*m* Tento parametr zajistí, že se spočítá nejen kovariance, ale i střední hodnota.

-*S train.scp* Seznam parametrizovaných trénovacích promluv (viz kapitola 2.1.1).

-*M hmm0* Jméno adresáře, do kterého bude uložen výstup (přepočítaný model *proto*).

-*proto* Definovaný prototyp (viz výše).

Jak bylo již zmíněno v kapitole (2.1.3), výstupní adresář *hmm0* je potřeba vytvořit **před** spuštěním programu**!** Doporučuji si rovnou vytvořit i zbývající adresáře *hmm1* až *hmmB* (11 adresářů), do kterých se budou později ukládat reestimované (natrénované) modely. Předejde se tak zbytečným chybám.

Jako další krok je potřeba v adresáři *hmm0* vytvořit tzv. *Master Macro File* (MMF) obsahující definice HMM pro jednotlivé monofony. Na začátku budou mít všechny modely stejné parametry, a to parametry modelu *proto* vypočtené v předchozím kroku. Požadovaný MMF tedy vytvoříme tak, že pro každý monofon vytvoříme kopii modelu *proto* a ten vždy pouze jenom přejmenujeme (řetězec v uvozovkách za parametrem *~h*). Kopírovat modely ručně je zdlouhavé, a proto existuje skript, který to dokáže automaticky. Jeho vstupem je soubor *monophones0* (obsahující seznam všech monofonů kromě krátké pauzy *_sp_*) a soubor s modelem *proto*. Program se spouští v adresáři *hmm0*.

MakeMMF proto monophones0 vFloors models

Parametry programu:

- *proto* Vstupní soubor *proto* modelu (výstupní soubor předešlého příkazu HcompV).

- *monophones0* Seznam monofonu, pro které chci proto rozkopírovat.

- *vFloors* Soubor, který vznikne jako meziprodukt v předchozím kroku.

- *models* Výstupní soubor s modely.

Nyní jsou modely inicializované a připravené k trénování. Trénování (reestimace) se provádí pomocí programu HERest, který je postaven na Baum-Welchově algoritmu (známý též jako forward-backward algoritmus). Program se spouští v adresáři obsahujícím podadresář *hmm0*, v němž jsou inicializované HMM:

HERest -T 1 -C CF.mfc -I phones0.mlf -t 250.0 150.0 1000.0 -S train.scp -H hmm0/MODELS -M hmm1 monophones0

Parametry programu:

-*C CF.mfc* Konfigurační soubor (stejný jako u HCompV).

-*I phones0.mlf* Soubor s monofonní transkripcí trénovacích promluv.

-*t 250.0 150.0 1000.0* Nastavení práhu prořezávání pro forward-backward algoritmus.

-*S train.scp* Seznam trénovacích dat.

-*H hmm0/MODELS* Soubor (Master Macro File) s modely, které vstupují do procesu reestimace.

-*M hmm1* Adresář pro uložení výstupního souboru (reestimovaných modelů) . Musí být vytvořen před spuštěním programu HERest.

-*monophones0* Seznam monofonů a zároveň jmen monofonních modelů.

Program funguje tak, že modely, jejichž názvy jsou uvedeny v souboru *monophones0*, se hledají v *hmm0/MODELS* a pokud se najdou, tak se přesunou do programu a reestimují se s použitím trénovacích dat uvedených v *train.scp*. Nově vzniklé modely se uloží opět do souboru *MODELS*, ale tentokrát do adresáře *hmm1*. Pokud se nějaký model z *monophones0* v souboru *MODELS* nevyskytuje, program zahlásí chybu a skončí. Pro lepší dosažení výsledků, je nutné reestimaci několikrát zopakovat. Další spuštění programu HERest budou vypadat následovně.

HERest -T 1 -C CF.mfc -I phones0.mlf -t 250.0 150.0 1000.0 -S train.scp -H hmm1/MODELS -M hmm2 monophones0
HERest -T 1 -C CF.mfc -I phones0.mlf -t 250.0 150.0 1000.0 -S train.scp -H hmm2/MODELS -M hmm3 monophones0
HERest -T 1 -C CF.mfc -I phones0.mlf -t 250.0 150.0 1000.0 -S train.scp -H hmm3/MODELS -M hmm4 monophones0

2.2.2 Úprava modelů pauz

Do této doby byl používán pouze model dlouhé pauzy *_sil_*. Tento model měl stejnou strukturu jako ostatní HMM. Aby byl tento model schopen modelovat různě dlouhé pauzy a dokázal absorbovat všelijaké šumy v pauzách, musí se změnit jeho topologie. Je potřeba přidat přechody ze stavu 2 do stavu 4 a také ze stavu 4 do stavu 2 (viz kap. 1.3.1, obr. 1.3 c)). Dále bude vytvořen model krátké pauzy *_sp_*, který bude mít pouze jeden emitující stav, svázaný pouze s prostředním stavem modelu *_sil_* (bude s ním sdílet parametry). Průběh uvedeného postupu je následující:

I. Libovolným textovým editorem se otevře soubor *MODELS* z adresáře *hmm4* a vytvoří se model *_sp_* tak, že se okopíruje prostřední stav modelu *_sil_* a příslušná část matice přechodů.

Vypadá-li tedy model *_sil_* takto:

```
~h "_sil_"
<BeginHMM>
<NumStates> 5
<State> 2
…
<State> 3
<Mean> 39
-4.74 2.88 –1.03 …
<Variance> 39
1.11 2.85 1.92 …
<GConst> 1.01
…
<TransP> 5
0.00 1.00 0.00 0.00 0.00
0.00 0.67 0.33 0.00 0.00
0.00 0.00 0.84 0.16 0.00
0.00 0.00 0.00 0.95 0.05
0.00 0.00 0.00 0.00 0.00
<EndHMM>
```

bude model *_sp_* vypadat následovně:

```
~h "_sp_"
<BeginHMM>
<NumStates> 3
<State> 2
<Mean> 39
-4.74 2.88 –1.03 …
<Variance> 39
1.11 2.85 1.92 …
<GConst> 1.01
<TransP> 3
0.00 1.00 0.00
0.00 0.84 0.16
0.00 0.00 0.00
<EndHMM>
```

II. Použije se program HHEd, který dokáže editovat soubory s HMM:

HHEd –T 1 –H hmm4/MODELS sil.hed monophones1

Parametry programu:

-H hmm4/MODELS	Vstupní soubor.
-monophones1	Seznam monofonů, tentokrát i s modelem *_sp_*.
-sil.hed	Soubor s příkazy, které má HHEd vykonat.

Do souboru *sil.hed* se napíše následující sekvence:

AT 2 4 0.2 {_sil_.transP}
Přidá do modelu *_sil_* přechod ze stavu 2 do stavu 4, který má pravděpodobnost 0.2.

AT 4 2 0.2 {_sil_.transP}
Přidá do modelu *_sil_* přechod ze stavu 4 do stavu 2, který má pravděpodobnost 0.2.

AT 1 3 0.3 {_sp_.transP}
Přidá do modelu *_sp_* přechod ze stavu 1 do stavu 3.

TI silst {_sil_.state[3],_sp_.state[2]}
Sváže třetí (prostřední) stav modelu *_sil_* se druhým (prostředním) stavem modelu *_sp_*. Výsledný sdílený stav bude v souboru *MODELS* uložen jako makro jménem *silst*.
V adresáři *hmm4* je teď uložen soubor *MODELS*, který obsahuje obě varianty pauzy (*_sp_, _sil_*). Nyní je potřeba model s novou pauzou *(_sp_*) reestimovat. Reestimace se pro dobré natrénování provede opět vícekrát, ale teď již se soubory *phones1* a *monophones1*, které obsahují model krátké pauzy.

HERest -T 1 -C CF.mfc -I phones1.mlf -t 250.0 150.0 1000.0 -S train.scp -H hmm4/MODELS -M hmm5 monophones1

HERest -T 1 -C CF.mfc -I phones1.mlf -t 250.0 150.0 1000.0 -S train.scp -H hmm5/MODELS -M hmm6 monophones1

HERest -T 1 -C CF.mfc -I phones1.mlf -t 250.0 150.0 1000.0 -S train.scp -H hmm6/MODELS -M hmm7 monophones1

2.2.3 Přerovnání trénovacích dat

Program HVite využívá Viterbiho algoritmus. Pomocí tohoto programu lze rozpoznávat a také provádět tzv. "přerovnání" (realignment) trénovacích dat. Přerovnání spočívá v tom, že program z transkripce na úrovni slov vytváří novou transkripci na úrovni monofonů (MLF soubor) a pokud ve slovníku existuje více variant fonetické transkripce daného slova, HVite vybere z nich tu, která nejlépe odpovídá trénovacím datům a dosud natrénovaným modelům.

*HVite –T 1 -l * –y lab -o SWT -b _SIL_ -C CF.mfc -m -a -H hmm7/MODELS –i aligned.mlf -t 250.0 -I words.mlf -S train.scp dict.txt monophones1*

Parametry programu:

*–l **	Způsobuje, že do jmen souborů v souboru MLF je vložen symbol * ("divoká karta") místo skutečné cesty.
-y lab	Jména souborů (řetězce v uvozovkách na začátku transkripce každé promluvy) v MLF souboru budou mít koncovku .lab .
-o SWT	Formát výstupního MLF souboru. Při tomto nastavení je potlačen výpis skóre (S) celých slov (W) a časových údajů (T), což má v praxi ten důsledek, že jsou vypisovány pouze monofony a výstupní soubor má stejný formát jako phones1.mlf.
*-b _SIL_*	Vkládá slovo *_SIL_* (reprezentující dlouhou pauzu *_sil_*) na začátek a konec každé promluvy. Je třeba si uvědomit, že transkripce na úrovni slov (*words.mlf*) tuto pauzu na začátku ani na konci nemá, na rozdíl od monofonní transkripce.
-C CF.mfc	Konfigurační soubor, stejný jako pro HERest.
-a	Říká programu, že má provést přerovnání (a ne třeba rozpoznávání, které také umí).
-m	Tento parametr nastavuje, že chceme výstup na úrovni fonémů.
-H hmm7/MODELS	Vstupní HMM
-i aligned.mlf	Výstupní MLF soubor s novou monofonní transkripcí.
-t 250.0	Prořezávací práh pro "beam search".
-I words.mlf	Vstupní MLF soubor s transkripcí na úrovni slov.
-S train.scp	Seznam trénovacích promluv.
-monophones1	Seznam monofonů .
-dict	Slovník výslovností.

Před přerovnáním je dobré upravit slovník tak, že každá řádka ve slovníku bude zdvojena. Na prvním řádku bude fonetická transkripce krátkou pauzou *_sp_* a druhá řádka bude končit dlouhou pauzou *_sil_*. HVite si pak bude moci vybrat, která pauza za slovem bude lépe vyhovovat akustickým trénovacím datům.

Například:

*a a _sp_*
*a a _sil_*
*absence a p s e n c e _sp_*
*absence a p s e n c e _sil_*
*absolutní a p s o l u t N I _sp_*
*absolutní a p s o l u t N I _sil_*
...

Výsledkem přerovnávání bude tedy MLF soubor *aligned.mlf*, který obsahuje nové monofonní transkripce trénovacích promluv. Transkripce těch trénovacích promluv, které se nepodařilo "zarovnat", se v *aligned.mlf* neobjeví. Takové trénovací promluvy jsou s největší pravděpodobností špatně přepsané a je třeba je vyhledat a odstranit z *train.scp*. Nový seznam trénovacích promluv bude *aligned.scp*. K tomuto účelu slouží program.

CreateAligned.exe aligned.mlf train.scp aligned.scp ne.scp

Parametry programu:

-*aligned.mlf* Vstupní soubor s přerovnanou transkripcí trénovacích promluv
-*train.scp* Vstupní seznam trénovacích promluv
-*aligned.scp* Výstupní seznam s dobře přerovnanými promluvami
-*ne.scp* Výstupný seznam nepřerovnaných promluv

Modely se znovu reestimují s novým monofonním přepisem *aligned.mlf* a novým trénovacím seznamem *aligned.scp*. Příkazy HERest budou tedy pro reestimaci vypadat takto:

HERest -T 1 -C CF.mfc -I aligned.mlf -t 250.0 150.0 1000.0 -S aligned.scp -H hmm7/MODELS -M hmm8 monophones1

HERest -T 1 -C CF.mfc -I aligned.mlf -t 250.0 150.0 1000.0 -S aligned.scp -H hmm8/MODELS -M hmm9 monophones1

HERest -T 1 -C CF.mfc -I aligned.mlf -t 250.0 150.0 1000.0 -S aligned.scp -H hmm9/MODELS -M hmmA monophones1

HERest -T 1 -C CF.mfc -I aligned.mlf -t 250.0 150.0 1000.0 -S aligned.scp –H hmmA/MODELS -M hmmB monophones1

2.2.4 Přidávání složek

Pro dosažení lepší úspěšnosti rozpoznávání je zapotřebí přidat do modelu několik složek. Složka se přidává pomocí následujícího programu:

HHEd -T 1 -A -C CF.mfc -H hmmB/models -M hmm_2_0 add_next.hed monophones1 > hmm_2_0\log

Parametry programu:

- *CF.mfc* Konfigurační soubor.
- *hmmB/models* Vstupní model ve složce *hmmB*.
- *hmm_2_0* Výstupní model ve složce *hmm_2_0*.
- *add_next.hed* Soubor pro přidání složky.

Po přidání složky je zapotřebí nově získaný model několikrát reestimovat.

Herest -T 1 -C CF.mfc -I aligned.mlf -t 250.0 150.0 1000.0 -S aligned.scp -s hmm_2_1\stats -H hmm_2_0\models -M hmm_2_1 monophones1 > hmm_2_1\log

Herest -T 1 -C CF.mfc -I aligned.mlf -t 250.0 150.0 1000.0 -S aligned.scp -s hmm_2_2\stats -H hmm_2_1\models -M hmm_2_2 monophones1 > hmm_2_2\log

Herest -T 1 -C CF.mfc -I aligned.mlf -t 250.0 150.0 1000.0 -S aligned.scp -s hmm_2_3\stats -H hmm_2_2\models -M hmm_2_3 monophones1 > hmm_2_3\log

Herest -T 1 -C CF.mfc -I aligned.mlf -t 250.0 150.0 1000.0 -S aligned.scp -s hmm_2_4\stats -H hmm_2_3\models -M hmm_2_4 monophones1 > hmm_2_4\log

Nově získaný model, který je připraven k rozpoznávání, se tedy nachází ve složce *hmm_2_4*. Do modelu je možno přidat nekonečně mnoho složek. V určitém okamžiku (při určitém počtu přidaných složek) se však dosáhne maxima. To znamená, že přidáváním dalších složek se úspěšnost rozpoznávání nezlepšuje (může se i zhoršovat), respektive zlepšuje se zanedbatelným způsobem.

2.3 Rozpoznávání

Z předchozích kroků byl získán monofonový akustický model. V této chvíli je již vše připraveno, aby se mohlo provést rozpoznávání „neznámých“ wave. Tyto wavy jsou již zařazeny v seznamu *param.scp* a jsou tedy zparametrizované. V kroku 2.2.3 byl podrobně popsán příkaz Hvite, který se také používá k rozpoznávání neznámých promluv.

*HVite -C CF.mfc -H hmmB/models -S test.scp -i vysledek_all.txt -l * -p -60.0 –w wdnet dict.txt monophones1*

Parametry programu:

-C CF.mfc	Konfigurační soubor, stejný jako pro HERest
-H hmm/models	HMM modely
-S test.scp	Seznam testovacích promluv
-i vysledek_all.txt	Výstupní soubor
-l ''*	Způsobuje, že do jmen souborů v souboru MLF je vložen symbol * ("divoká karta") místo skutečné cesty.
-p -60.0	Volbou penalizace (číslo za -p) můžeme měnit váhu rozpoznávání krátkých slov.
-w wdnet	Rozpoznávací síť
-dict	Rozpoznávací slovník
-monophones1	Seznam monofonů (viz výše).

Posledním příkazem bude příkaz Hresult. Tento program spočte, s jakou úspěšností byly rozpoznány „neznámé“ promluvy. Je zřejmé, že musí být jasno, co v tzv. „neznámých promluvách“ bylo řečeno, jinak by nebylo možné určit, kolik promluv bylo rozpoznáno dobře. Proto je také slovo „neznámé“ v uvozovkách, protože my víme, co bylo řečeno (od lidských přepisovačů) a zjišťujeme kolik toho Hvite rozpoznal správně.

Hresults -f -I words.mlf monophones1 vysledek_all.txt > vysledek.txt

Parametry programu:

-I words.mlf	Referenční MLF soubor s přepisy testovacích nahrávek.
-monophones1	Seznam monofonů.
-vysledek_all.txt	Výsledky rozpoznávání neznámých promluv.
- vysledek.txt	Výsledný soubor.

```
---------------------- Sentence Scores ----------------------
====================== HTK Results Analysis =======================
Date: Fri Oct 16 16:29:22 2015
Ref : words1.mlf
Rec : vysledek_all.txt
------------------------ File Results --------------------------
bojove-sporty_-_magazin-bojovych-sportu_0_500.rec:  100.00( 83.33)  [H=  6, D= 0, S= 0, I= 1, N= 6]
bojove-sporty_-_magazin-bojovych-sportu_0_501.rec:   72.73( 54.55)  [H=  8, D= 2, S= 1, I= 2, N= 11]
bojove-sporty_-_magazin-bojovych-sportu_0_502.rec:   63.64( 45.45)  [H=  7, D= 3, S= 1, I= 2, N= 11]
bojove-sporty_-_magazin-bojovych-sportu_0_503.rec:   33.33( 33.33)  [H=  2, D= 3, S= 1, I= 0, N= 6]
bojove-sporty_-_magazin-bojovych-sportu_0_504.rec:   88.89( 66.67)  [H=  8, D= 1, S= 0, I= 2, N= 9]
bojove-sporty_-_magazin-bojovych-sportu_0_505.rec:   88.89( 66.67)  [H=  8, D= 0, S= 1, I= 2, N= 9]
bojove-sporty_-_magazin-bojovych-sportu_0_506.rec:   45.45( 36.36)  [H=  5, D= 1, S= 5, I= 1, N= 11]
bojove-sporty_-_magazin-bojovych-sportu_0_507.rec:   62.50( 62.50)  [H=  5, D= 2, S= 1, I= 0, N= 8]
bojove-sporty_-_magazin-bojovych-sportu_0_508.rec:   85.71( 57.14)  [H=  6, D= 1, S= 0, I= 2, N= 7]
bojove-sporty_-_magazin-bojovych-sportu_0_509.rec:   25.00( 25.00)  [H=  2, D= 6, S= 0, I= 0, N= 8]
bojove-sporty_-_magazin-bojovych-sportu_0_510.rec:  100.00( 87.50)  [H=  8, D= 0, S= 0, I= 1, N= 8]
------------------------ Overall Results ----------------------
SENT: %Correct=0.00 [H=0, S=11, N=11]
WORD: %Corr=69.15, Acc=55.32 [H=65, D=19, S=10, I=13, N=94]
===================================================================
```

Z těchto výsledků je vidět, že se rozpoznávalo 94 slov s 55.32 procentní úspěšností.

3. Praktická část

Česká televize má rozsáhlý videoarchiv iVysílání. V tomto archivu se nachází všechny možné pořady, které byly odvysílány na programech ČT. A protože Fakulta kybernetiky Západočeské univerzity v Plzni spolupracuje s ČT, může tedy tato data ve videoarchivu využívat. Některé tyto pořady obsahují přepisy ve formě skrytých titulků a dají se spolu s pořady využívat například k vytváření akustických modelů.

Praktická část (experiment) této práce by se dal rozdělit na čtyři hlavní části. Nejprve bylo zapotřebí stáhnout z webu ČT volně dostupná data (pořady a skryté titulky). Dále bylo nutné analyzovat formát těchto dat a připravit z nich soubory potřebné pro natrénování akustického modelu. Třetí část bylo samotné natrénování akustického modelu pomocí HTK. Poslední část celého experimentu vyplynula z nesouladu skrytých titulků a pořadů. Titulky nebyly správně synchronizované s pořadem (titulky byly opožděné nebo naopak v předstihu vůči zvukové stopě pořadu), a proto poslední část se tedy zabývá několika přístupy, jak tento problém vyřešit.

3.1 Získání dat z webu ČT

Data byla získána z webu ČT na adrese: http://www.ceskatelevize.cz/ivysilani. Na této adrese byly vybrány různé vzorky sportovních pořadů (magazíny bojových sportů, kulturistiky, parašutismu, karate, apod. – celkově 22 souborů) v celkové délce 10 hodin a 46 minut. Tato data byla stažena programem *get_video.py*, což je program, sloužící ke stahování pořadů z webu ČT. Tento program stáhne pořad a extrahuje z něj zvlášť video ve formátu *.mpeg* a titulky ve formátu *.txt*.

Dále bylo zapotřebí stažené video ve formátu *.mpeg* převést na zvukovou stopu ve formátu *.wav*. K tomuto účelu slouží program *ffmpeg.exe.*

Takto připravená data, tj. sportovní pořady ve formátu *.wav* a skryté titulky ve formátu *.txt* byla připravena k dalšímu zpracování.

3.1.1. Struktura skrytých titulků

Skryté titulky měly následující strukturu:

1; 2200 6000
Hezký den s Českou televizí,
hezký den s programem ČT sport

2; 6000 9080
a také s 30 minutami
u bojových sportů.

3; 9080 12520
Nejprve se ohlédneme
za Mistrovstvím ČR v karate
…

kde na první řádce jsou tři čísla. První číslo (se středníkem) říká, o kolikátou promluvu se jedná. Další dvě čísla značí začátek a konec dané promluvy (v milisekundách). Na dalších řádkách jsou potom samotné titulky, které byly v tomto časovém úseku řečeny.

3.2 Příprava souborů k trénování

Jak již bylo řečeno v kapitole 2.1.1, bylo nejprve zapotřebí vytvořit soubory potřebné k trénování. Tyto soubory byly vytvořeny ze stažených dat. K přípravě těchto souborů byl využit skriptovací programovací jazyk Python.

3.2.1. Vytvoření skriptu na rozřezání titulek

Jako první byl vytvořen skript *program_rozrezani_titulek.py*, který „rozřezal“ titulky podle času. Protože stažené pořady byly v průměru kolem 30 minut dlouhé, bylo potřeba je rozřezat na daleko menší části, kterými by se dal akustický model natrénovat. Tento skript upravil stažené titulky do následujícího formátu:

Vstup=bojove-sporty_-_mcr-v-karate-2015-pribram_0.wav
2.2 6.0 bojove-sporty_-_mcr-v-karate-2015-pribram_0_001.wav
6.0 9.08 bojove-sporty_-_mcr-v-karate-2015-pribram_0_002.wav
9.08 12.52 bojove-sporty_-_mcr-v-karate-2015-pribram_0_003.wav

kde na první řádce byl za slovem *Vstup=* název právě zpracovávaného (rozřezávaného) souboru. Na dalších řádkách potom následovaly časové značky (v sekundách), odkud kam se má daná věta vyříznout a název, pod jakým se má tato věta uložit.

3.2.2. Rozřezání titulek

Titulky byly rozřezány pomocí programu *WaveCutter.exe*.

WaveCutter.exe vystup.txt in_folder out_folder

Parametry programu:

-*I WaveCutter*.exe	Program sloužící k rozřezání titulek.
-*vystup.txt*	Výstupní soubor minulého kroku (kapitola 3.2.1).
-*in_folder.txt*	Značí cestu k souboru, který se má rozřezávat.
- *out_folder.txt*	Značí, kam se uloží nařezané promluvy.

3.2.3. Příprava vět pro normalizaci textu

Stažené titulky obsahovaly mnoho číslovek a zkratek. Pro dobré natrénování bylo zapotřebí mít tyto číslovky a zkratky ve slovní podobě a ještě k tomu dobře vyskloňované.

Před normalizací textu bylo zapotřebí upravit stažené titulky do určité podoby, tj. odstranit časové značky začátku a konce promluvy a srovnat na jeden řádek takové titulky, které k sobě patří, tzn. ty, které byly řečeny v jednom časovém intervalu (věty). Proto byl vytvořen skript *vety_pro_normalizaci.py*, který titulky upravil do požadovaného formátu:

Hezký den s Českou televizí, hezký den s programem ČT sport.
a také s 30 minutami u bojových sportů.
Nejprve se ohlédneme za Mistrovstvím ČR v karate,
které se konalo v Příbrami a konalo se 31. ledna roku 2015.

3.2.4. Normalizace textu

Jak bylo řečeno v předchozím kroku, normalizace textu je proces, který expanduje zkratky, rozvíjí číslovky a data, apod. K tomuto účelu slouží program *remotejmzwFULL.py*. Tento program se spojí s internetovou databází, ve které jsou uložena pravidla, podle kterých jsou číslovky převedeny do slovní podoby nebo slova převedena do zkratek. V této databázi jsou také uložena známá, respektive velmi často používaná slova nebo znaky a jejich fonetická transkripce.

Program *remotejmzwFULL.py* vrátí tři soubory.

a) Normalizovaný text

*hezký den s ČT , hezký den s programem ČT_sport |_*
*a také s třiceti minutami u bojových sportů . |_*
*nejprve se ohlédneme za MČR v karate , |_*
*které se konalo v Příbrami a konalo se třicátého prvního ledna roku dva tisíce patnáct. |_*

kde na konci každé řádky je znak |_, který značí konec jedné promluvy (věty).

b) Slovník známých slov

ČT	*C e s k y t e l e v i z i*
ČT	*C e s k A t e l e v i z e*
ČT	*C e s k E t e l e v i z i*
třicátého	*t Q i c A t E h o*
třiceti	*t Q i c e T i*

kde jsou všechna slova z titulků, která byla nalezena v internetové databázi a je k nim známá jejich výslovnost, respektive jejich všechny možné výslovnosti (druhý sloupeček).

c) Slovník neznámých slov

Shotokan	*s x o t o k a n*
Shotokan	*z h o t o k a n*
bodovatelné	*b o d o v a t e l n E*
přiděli	*p Q i D e l i*

kde jsou všechna slova z titulků, která nebyla nalezena v databázi. Taková slova jsou v drtivé většině jména hráčů, ale také např. cizí slova nebo špatně přepsaná slova. Všechna tato slova musí být ručně zkontrolována (jak slovo, tak jeho výslovnost) a následně zkopírována do slovníku známých slov.

3.2.5. Vytvoření souboru words.mlf

Soubor *words.mlf* byl vytvořen následujícím způsobem. Nejprve byl z titulků pomocí skriptu *divoky_karty.py* vytvořen soubor, který obsahoval názvy jednotlivých vět ve speciálním formátu (tzv. formát divoké karty – viz 2.1.1 e)). Na konci každé věty bylo číslo značící pořadí titulku.

"/bojove-sporty_-_mcr-v-karate-2015-pribram_0_001.lab"*
"/bojove-sporty_-_mcr-v-karate-2015-pribram_0_002.lab"*
"/bojove-sporty_-_mcr-v-karate-2015-pribram_0_003.lab"*

Dále, pomocí skriptu *program_words_mlf.py*, byl vytvořen finální soubor *words.mlf* a to následujícím způsobem. Nejprve se do souboru *words.mlf* zapsal název dané věty (viz předchozí krok) a ihned pod něj se zapsala první věta normalizovaného textu (viz kapitola 3.2.4 a)). Tato věta byla však rozdělena na slova, kde každé slovo bylo zapsáno na jednu řádku. Na konci každé věty byl znak /_ nahrazen tečkou. Soubor *words.mlf* obsahoval transkripci všech stažených pořadů (po poslední větě prvního pořadu následovala ihned první věta následujícího pořadu). Výsledkem byl požadovaný formát:

#!MLF!#
"/bojove-sporty_-_mcr-v-karate-2015-pribram_0_001.lab"*
hezký
den
s
ČT
hezký
den
s
programem
ČT_sport
.
"/bojove-sporty_-_mcr-v-karate-2015-pribram_0_002.lab"*
a
také
s
třiceti
minutami
u
bojových
sportů
.

3.2.6. Vytvoření souborů dict_sp a dict

V kapitole 3.2.4 byly získány dva slovníky, tj. slovník známých slov a slovník neznámých slov. Z těchto slovníků vznikly poté požadované *dict_sp a dict*. Po zkontrolování slovníku neznámých slov byl tento slovník překopírován do slovníku známých slov (tento slovník obsahoval úplně všechna slova vyskytující se ve všech pořadech). Dále byl použit skript *dict_sp.py* na úpravu tohoto slovníku. Tento skript přidal na konec každé řádky symbol *_sp_*, cože je symbol modelu krátké pauzy. Tento slovník byl použit v první části trénování akustického modelu. Výsledný slovník měl strukturu

ČT	*C e s k y t e l e v i z i _sp_*
ČT	*C e s k A t e l e v i z e _sp_*
ČT	*C e s k E t e l e v i z i _sp_*
třicátého	*t Q i c A t E h o_ sp_*
třiceti	*t Q i c e T i _sp_*

Ihned poté byl vytvořen slovník *dict*, který navíc u každého slova obsahoval model dlouhé pauzy *_sil_*. Tento slovník byl poté použit v druhé části trénování, kde se již pracovalo s modelem dlouhé pauzy. K vytvoření tohoto slovníku byl opět použit skript *dict.py*, který zdvojil každý řádek a na každém druhém řádku nahradil model krátké pauzy za model pauzy dlouhé.

ČT	*C e s k y t e l e v i z i _sp_*
ČT	*C e s k y t e l e v i z i _sil_*
ČT	*C e s k A t e l e v i z e _sp_*
ČT	*C e s k A t e l e v i z e _sil_*
ČT	*C e s k E t e l e v i z i _sp_*
ČT	*C e s k E t e l e v i z i _sil_*
třicátého	*t Q i c A t E h o_ sp_*
třicátého	*t Q i c A t E h o _sil_*

3.2.7. Vytvoření souboru train.scp

Soubor *train.scp* byl vytvořen pomocí skriptu *train.py* tak, že za znak *htk/* byl okopírován název právě zpracovávaného souboru a na jeho konec se okopírovalo číslo, značící pořadí titulku. Výsledný formát byl následující:

htk/bojove-sporty_-_mcr-v-karate-2015-pribram_0_001.htk
htk/bojove-sporty_-_mcr-v-karate-2015-pribram_0_002.htk
htk/bojove-sporty_-_mcr-v-karate-2015-pribram_0_003.htk
htk/bojove-sporty_-_mcr-v-karate-2015-pribram_0_004.htk
htk/bojove-sporty_-_mcr-v-karate-2015-pribram_0_005.htk

Tento soubor opět obsahoval všechny věty ze všech pořadů (22 pořadů).

3.2.8. Vytvoření souboru test.scp

Soubor *test.scp* byl vytvořen tak, že se náhodně vybralo několik vět ze souboru *train.scp (*viz 3.2.7), a ty se vyjmuly a vložily do souboru *test.scp*. To znamená, že věty, které se nacházejí v souboru *test.scp* se nesmí nacházet v souboru *train.scp*.

V závěrečném rozpoznávání se musí rozpoznávat neznámé věty, to znamená, že akustický model nesmí být těmito větami trénován.

3.2.9. Vytvoření souboru param.scp

Nejprve byl vytvořen „pomocný“ soubor *param.txt*, který obsahoval odkaz na nařezané věty ve formátu *.wav*. Postup tvorby tohoto souboru byl identický jako v kapitole 3.2.7.

wav/bojove-sporty_-_mcr-v-karate-2015-pribram_0_001.wav
wav/bojove-sporty_-_mcr-v-karate-2015-pribram_0_002.wav
wav/bojove-sporty_-_mcr-v-karate-2015-pribram_0_003.wav
wav/bojove-sporty_-_mcr-v-karate-2015-pribram_0_004.wav

Dále byl vytvořen další skript *param.py* , který „zkombinoval“ obsah souborů *train.scp* (viz kapitola 3.2.7) a param.txt (viz tato kapitola) tak, že první sloupeček výsledného souboru tvořily věty z *train.scp* a druhý sloupeček věty z *param.txt.*

wav/bojove-sporty_-_mcr-v-karate-2015-pribram_0_001.wav	*htk/bojove-sporty_-_mcr-v-karate-2015-pribram_0_001.htk*
wav/bojove-sporty_-_mcr-v-karate-2015-pribram_0_002.wav	*htk/bojove-sporty_-_mcr-v-karate-2015-pribram_0_002.htk*
wav/bojove-sporty_-_mcr-v-karate-2015-pribram_0_003.wav	*htk/bojove-sporty_-_mcr-v-karate-2015-pribram_0_003.htk*
wav/bojove-sporty_-_mcr-v-karate-2015-pribram_0_004.wav	*htk/bojove-sporty_-_mcr-v-karate-2015-pribram_0_004.htk*

Tímto jsou vytvořeny všechny potřebné soubory k natrénování akustického modelu pomocí HTK.

3.3 Trénování a rozpoznávání

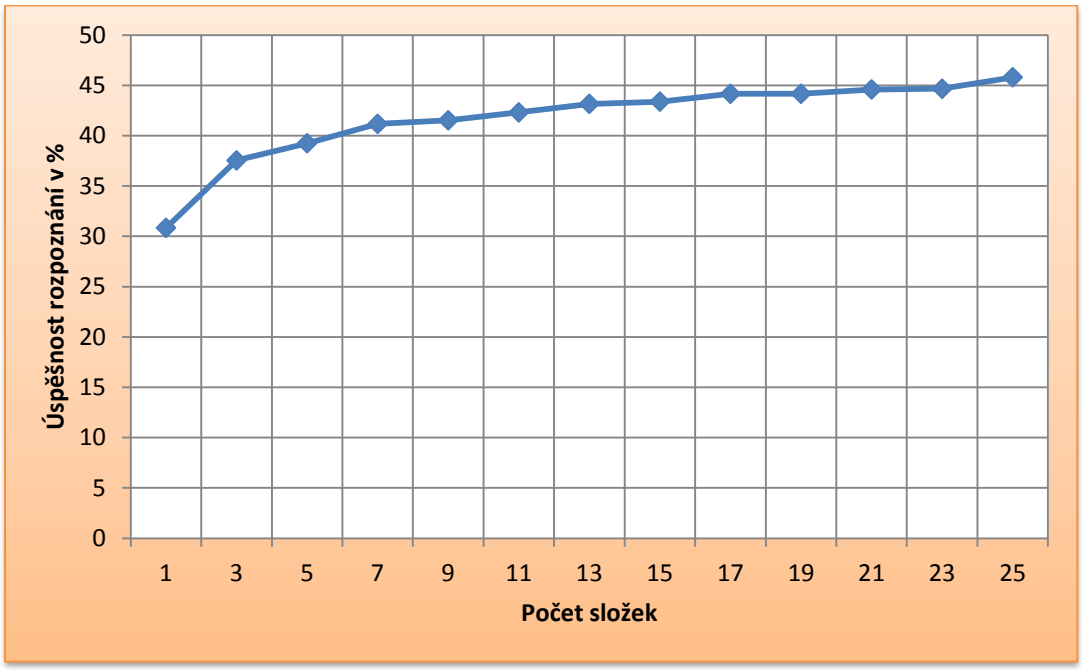
Akustický model byl natrénován pomocí návodu, kterému se věnuje celá 2. kapitola.

Výsledné rozpoznávání se provádělo na 70ti větách, ve kterých bylo 774 různých slov (celkem 1404 slov). Pro dosažení objektivních výsledků byla při provádění různých experimentů tato testovací množina stále stejná.

3.4 Výsledky

Po prvním natrénování akustického modelu pomocí HTK bylo dosaženo poměrně nízké úspěšnosti rozpoznávání. Tato úspěšnost byla 30,84 %. Využilo se proto metody přidávání složek (viz kapitola 2.2.4). Po každém přidání složky se provedlo rozpoznávání (aby bylo vidět, jak se mění úspěšnost rozpoznávání) a opět se přidala další složka. Tento cyklus se opakoval, dokud se zvyšovala úspěšnost rozpoznávání (v tomto případě se úspěšnost rozpoznávání zvyšovala do 25. složky). Tato část práce byla velice časově náročná, proto byl vytvořen skript *slozkovani.py*, který tuto činnost dělal automaticky.

Touto operací se podařilo zvýšit úspěšnost rozpoznávání na 45,80 %, což stále nebylo uspokojující.



Obr. 3.1 Graf úspěšnosti rozpoznávání v závislosti na počtu složek.

Tento, stále poněkud špatný výsledek, byl zapříčiněn nejen různými hlasy mluvčích (každý sportovní pořad byl komentován jiným, respektive jinými komentátory), ale hlavně nesprávnou synchronizací titulků se zvukovou stopou.

3.4.1. Prodlužování vět

Nápadem na zlepšení úspěšnosti bylo prodlužování trénovacích vět. Myšlenka byla taková, že prodloužením trénovacích vět se sníží počet nesprávných hranic vět a tím pádem se zvýší počet správných slov ve větách. To znamená, že pokud nějaké slovo, patřící např. do věty číslo 19, bylo vysloveno až ve větě číslo 20, spojením těchto vět bude náhle slovo patřit do „své“ správné věty. Prodlužování vět bylo provedeno tak, že tři po sobě následující titulky, se spojily v jeden titulek a vytvořily tak novou větu.

Titulky před prodloužením (původní):

1; 2200 6000
Hezký den s Českou televizí,
hezký den s programem ČT sport
2; 6000 9080
a také s 30 minutami
u bojových sportů.
3; 9080 12520
Nejprve se ohlédneme
za Mistrovstvím ČR v karate,
4; 12520 18520
které se konalo v Příbrami
a konalo se 31. ledna roku 2015.
5; 18520 21240
My už sledujeme
první finálový zápas,
6; 21240 25320
kterým bylo kumite mužů
v kategorii do 84 kg.

Titulky po prodloužení:

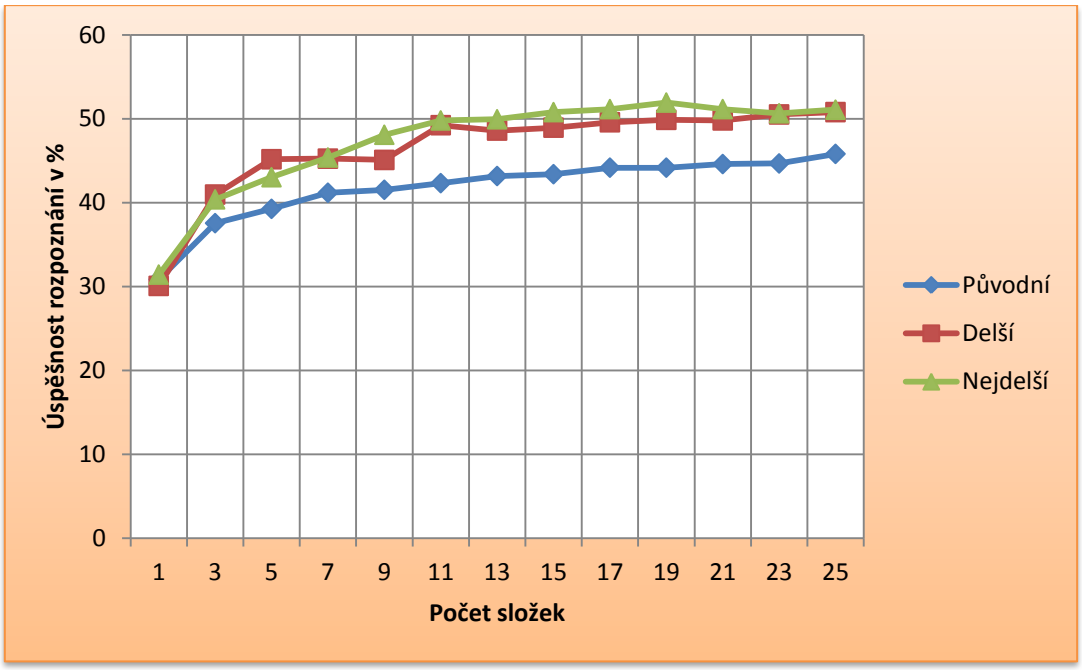
1; 2200 12520
Hezký den s Českou televizí, hezký den s programem ČT sport
a také s 30 minutami u bojových sportů.
Nejprve se ohlédneme za Mistrovstvím ČR v karate,
2; 12520 25320
které se konalo v Příbrami a konalo se 31. ledna roku 2015.
My už sledujeme první finálový zápas,
kterým bylo kumite mužů v kategorii do 84 kg.

K vytvoření těchto titulků byl vytvořen skript, který vzal číslo značící začátek prvního původního titulku a číslo značící konec třetího původního titulku. Z těchto čísel udělal začátek a konec nové promluvy a pod ně tuto promluvu přidal. Tímto způsobem tento skript zpracoval celé původní titulky a vytvořil nové (prodloužené).

Spojení titulků způsobilo změnu hranic promluv, a proto se musely celé *wavy* znovu nařezat, znovu vyrobit soubory k trénování, apod. Zkrátka, musela se zopakovat celá kapitola 3.2. Dále se opět provedl celý cyklus rozpoznávání a přidávání složek (viz výše). Úspěšnost rozpoznávání stoupla na 50,78 %, takže se ukázalo, že metoda prodlužování vět funguje. Proto byly věty ještě více prodlouženy. Nyní tak, že šest po sobě následujících titulků se spojilo v jeden titulek a vytvořilo tak novou větu. Úspěšnost rozpoznávání stoupla na 51.92 % (19tá složka).

Složka	*Původní*	*Delší*	*Nejdelší*
1	30,84	30,06	31,41
3	37,54	40,95	40,38
5	39,25	45,16	43,02
7	41,17	45,23	45,37
9	41,52	45,08	48,08
11	42,31	49,22	49,79
13	43,16	48,58	49,93
15	43,38	48,93	50,78
17	44,16	49,57	51,14
19	44,16	49,86	51,92
21	44,59	49,79	51,14
23	44,66	50,5	50,64
25	45,8	50,78	51,07

Obr. 3.2 Tabulka vývoje úspěšnosti rozpoznávání jednotlivých modelů (Původní = originální, neupravované titulky; Delší = spojení tří titulků do jednoho; Nejdelší = spojení šesti titulků do jednoho) v závislosti na počtu složek v (%).



Obr. 3.3 Graf úspěšnosti rozpoznávání jednotlivých modelů v závislosti na počtu složek.

Větší prodlužování vět už nemělo smysl, protože úspěšnost rozpoznávání se zlepšovala pouze minimálně. Proto se od této metody upustilo a hledala se jiná zlepšující metoda.

3.4.2. Synchronizace titulků

Jako další možnost, jak zlepšit úspěšnost rozpoznávání, bylo sesynchronizovat titulky se zvukovou stopou. To znamená přeorganizovat hranice titulků tak, aby to, co bylo řečeno v promluvě, skutečně přesně odpovídalo zvukové stopě. Tato operace nemohla být provedena v HTK, protože HTK je schopen přerovnat pouze pořady o maximální délce 5 minut. Proto byl použit nástroj, který byl vyvinut na katedře kybernetiky v Plzni.

Synchronizace se skládala z pěti kroků:

a) Parametrizace dat
   Nejprve bylo zapotřebí celé pořady parametrizovat. Použil se na to úplně stejný postup jako v kapitole 2.1.3.

b) Příprava slovníku
   Připravil se slovník, který byl použit v kroku c). A to tak, že se před výslovnost každého slova přidaly znaky *pt=.* K tomu sloužil program:

   slovnik_HTK2AP.pl slovnik.txt vocab.txt

   kde *slovnik.txt* je vstupní slovník a *vocab.txt* je upravený slovník ve formátu:

   dospělácké pt=dospjelAckE
   Dexteru pt=deksteru

c) Příprava modelu k synchronizaci
   Tento krok sestával ze dvou podkroků:

   UdelejImage.exe pokus.ini ..\ALG.lab

   ImageNet.exe -id XXX -net ALG.NET -vocab_in vocab.txt -acoustic ALG.lab

   Tyto příkazy upravily a připravily k synchronizaci vstupní model (nejlépe natrénovaný model z předchozích rozpoznávání, tj. model s úspěšností rozpoznávání 51,92 %).

d) Synchronizace

Synchronizace byla provedena pomocí následujícího programu:

Recognition.exe -ini ALG.INI -htk -alignment alignment.mlf -alignment_reference words.mlf -list locallist.txt

kde *alignment.mlf* byl výstupní soubor obsahující přesné hranice trifonů jednotlivých slov.

e) Vyjmutí slov

Z výstupního souboru předchozího kroku bylo zapotřebí „vydloubnout“ celá slova a jejich časové značky značící začátek a konec promluvy daného slova. Toto zařídil následující program:

dloubni_jen_slova.pl alignment.mlf _alignment.mlf

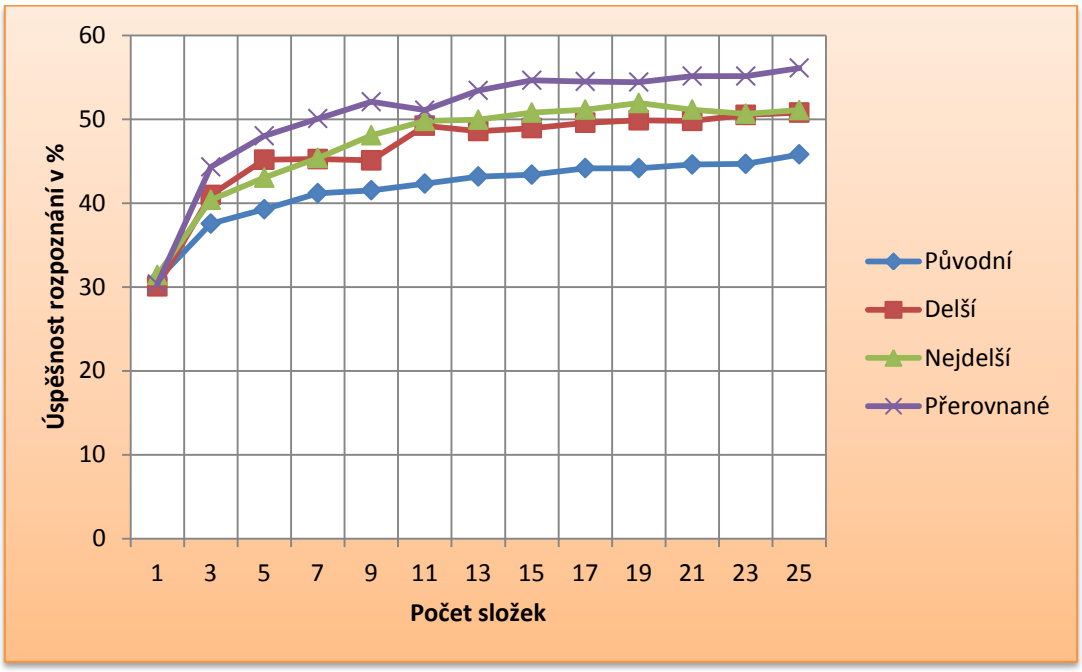
Formát výstupního souboru *_alignment.mlf*:

#!MLF!#
"/bojove-sporty_-_mcr-v-karate-2015-pribram_0.rec"*
0002,784 0003,103 hezký
0003,103 0003,293 den
0003,293 0003,422 s
0003,422 0004,280 ČT

Dále bylo zapotřebí upravit tento soubor do klasického formátu titulků (viz 3.1.1). K tomuto účelu byl vytvořen skript *prerovnani.py*. Tento skript vzal u prvního slova číslo značící začátek promluvy tohoto slova a u dvacátého slova číslo značící konec tohoto slova a vytvořil z nich začátek a konec jednoho celého titulku. Na další řádku zapsal všechna slova, která se v tomto intervalu nacházela. Takto zpracoval celý soubor *_alignment.mlf.* Získané titulky prošly opět celým cyklem přípravy dat a následným trénováním a rozpoznáváním (viz 2). Výsledky úspěšnosti rozpoznávání přerovnaných titulků a porovnání vůči nepřerovnaným titulkům jsou uvedeny v následující tabulce.

Složka	*Původní*	*Delší*	*Nejdelší*	*Přerovnané*
1	30,84	30,06	31,41	30,34
3	37,54	40,95	40,38	44,3
5	39,25	45,16	43,02	48,01
7	41,17	45,23	45,37	50,07
9	41,52	45,08	48,08	52,07
11	42,31	49,22	49,79	51,07
13	43,16	48,58	49,93	53,42
15	43,38	48,93	50,78	54,63
17	44,16	49,57	51,14	54,49
19	44,16	49,86	51,92	54,42
21	44,59	49,79	51,14	55,13
23	44,66	50,5	50,64	55,13
25	45,8	50,78	51,07	56,1

Obr. 3.4 Tabulka porovnání vývoje úspěšnosti rozpoznávání nepřerovnaných (Původní, Delší, Nejdelší – viz obr 3.2) a přerovnaných titulků v (%).



Obr. 3.5 Graf úspěšnosti rozpoznávání jednotlivých modelů v závislosti na počtu složek.

Závěr

Úkolem v této bakalářské práci bylo stáhnout data z webu České televize, analyzovat formát těchto dat, sesynchronizovat titulky se zvukovou stopou a následně natrénovat akustický model pomocí nástroje HTK.

Nejprve byla tedy stažena potřebná data z webu České televize. Byly to sportovní pořady a jejich skryté titulky v celkové délce 10 hodin a 46 minut. Dále byl analyzován formát těchto dat a pomocí skriptovacího jazyka Python byly z těchto dat vytvořeny potřebné vstupní soubory pro HTK. Během vytváření těchto souborů se nevyskytly žádné výraznější problémy. Dále byl dle návodu natrénován akustický model, jehož úspěšnost rozpoznávání byla pouhých 30,84 %. Poté byly postupně do modelu přidávány jednotlivé složky. Úspěšnost rozpoznávání se při přidání několika prvních složek výrazně zvyšovala, ale s přibývajícím počtem složek nárůst úspěšnosti pomalu klesal. Proto po dosažení 25 složek nebyly další složky přidávány. Úspěšnost rozpoznávání stoupla tímto krokem na 45,8 %, což pořád nebylo uspokojivé číslo.

Analýzou vstupních dat bylo zjištěno, že tato nízká úspěšnost rozpoznávání je způsobena špatnou synchronizací skrytých titulků se zvukovými stopami pořadů. Jako první možné řešení se zvolilo postupné prodlužování hranic titulků. Nejprve byly spojeny 3 originální titulky v jeden titulek. Takto byly postupně zpracovány všechny pořady a následně z nich byl natrénován akustický model. Úspěšnost rozpoznávání tohoto modelu vzrostla na 50,78 %. Bylo vidět, že tato metoda funguje, a proto byly titulky ještě více prodlouženy, a to tak, že bylo spojeno 6 originálních titulků v jeden titulek. Po následném natrénování vzrostla úspěšnost rozpoznávání na 51,92 %. Toto prodloužení tedy již nepřineslo výraznější zlepšení, a proto se již dále nezkoušelo spojování ještě více originálních titulků v jeden. Místo prodlužování vět byla jako další metoda vyzkoušena úprava hranic jednotlivých titulků. Stažené titulky byly automaticky přerovnány a následně byl opět z těchto dat natrénován akustický model. Úspěšnost rozpoznávání vzrostla na 56,1 %.

Díky výše popsaným metodám a postupům se podařilo zvýšit úspěšnost rozpoznávání z 30,84 % na 56,1 %. Tato poněkud nedobrá výsledná úspěšnost rozpoznávání byla zcela jistě zapříčiněna různými hlasy řečníků (každý sportovní pořad byl totiž komentován nikým jiným), či vlivem hluku na pozadí sportovních pořadů. Dalším důvodem bezpochyby bylo, že se v této bakalářské práci pracovalo se zerogramovými jazykovými modely. Při použití bigramů nebo trigramů by byla úspěšnost rozpoznávání jistě větší.

Literatura

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

[2] Psutka, J. and Müller, L. and Matoušek, J. and Radová, V.: Mluvíme s počítačem česky, Academia, Praha, 2006.

[3] Švec, J. a spol.: Normalizace textu [online]. 2011. Dostupné z <http://www.kky.zcu.cz/cs/sw/jmzw>

[4] Web ČT. [online]. Květen 2016. Dostupné z <http://www.ceskatelevize.cz/ivysilani>

[5] Young, S. et al., The HTK book (for HTK version 3.4). Cambridge University Press, 2006

Přílohy

I. Struktura skrytých titulků

1; 2200 6000
Hezký den s Českou televizí,
hezký den s programem ČT sport

2; 6000 9080
a také s 30 minutami
u bojových sportů.

3; 9080 12520
Nejprve se ohlédneme
za Mistrovstvím ČR v karate,

4; 12520 18520
které se konalo v Příbrami
a konalo se 31. ledna roku 2015.

5; 18520 21240
My už sledujeme
první finálový zápas,

6; 21240 25320
kterým bylo kumite mužů
v kategorii do 84 kg.

7; 25320 29840
S modrou šerpou, modrými chrániči
vpravo teď vidíme Petra Kubičku

8; 29840 31760
z Fight Clubu České Budějovice.

9; 31760 34040
A proti němu
se do finále probojoval,

10; 34040 37160
samozřejmě s červenou šerpou
a červenými chrániči

II. Přepis na úrovni slov (*words.mlf*)

#!MLF!#
"/bojove-sporty_-_mcr-v-karate-2015-pribram_0_001.lab"*
hezký
den
s
ČT
hezký
den
s
programem
ČT_sport
.
"/bojove-sporty_-_mcr-v-karate-2015-pribram_0_002.lab"*
a
také
s
třiceti
minutami
u
bojových
sportů
.
"/bojove-sporty_-_mcr-v-karate-2015-pribram_0_003.lab"*
nejprve
se
ohlédneme
za
MČR
v
karate
.
"/bojove-sporty_-_mcr-v-karate-2015-pribram_0_004.lab"*
které
se
konalo
v
Příbrami
a
konalo
se
třicátého
prvního
ledna

roku
dva
tisíce
patnáct
.
"/bojove-sporty_-_mcr-v-karate-2015-pribram_0_005.lab"*
my
už
sledujeme
první
finálový
zápas
.
"/bojove-sporty_-_mcr-v-karate-2015-pribram_0_006.lab"*
kterým
bylo
kumite
mužů
v
kategorii
do
osmdesáti
čtyř
kg
.
"/bojove-sporty_-_mcr-v-karate-2015-pribram_0_007.lab"*
s
modrou
šerpou
modrými
chrániči
vpravo
teď
vidíme
Petra
Kubičku
.
"/bojove-sporty_-_mcr-v-karate-2015-pribram_0_008.lab"*
z
Fight
Clubu
České_Budějovice
.

III. Slovník *dict*

desetiny	*d e s e T i n i _sp_*
desetiny	*d e s e T i n i _sil_*
Michálkovice	*m i x A l k o v i c e _sp_*
Michálkovice	*m i x A l k o v i c e _sil_*
taekwondo	*t e j k v o n d o _sp_*
taekwondo	*t e j k v o n d o _sil_*
Litoměřic	*l i t o m J e R i c _sp_*
Litoměřic	*l i t o m J e R i c _sil_*
Litoměřic	*l i t o m J e R i w _sp_*
Litoměřic	*l i t o m J e R i w _sil_*
Karolína	*k a r o l I n a _sp_*
Karolína	*k a r o l I n a _sil_*
výstup	*v I s t u p _sp_*
výstup	*v I s t u p _sil_*
výstup	*v I s t u b _sp_*
výstup	*v I s t u b _sil_*
oddílu	*o d D I l u _sp_*
oddílu	*o d D I l u _sil_*
výsledkem	*v I s l e t k e m _sp_*
výsledkem	*v I s l e t k e m _sil_*
bronzovou	*b r o n z o v y _sp_*
bronzovou	*b r o n z o v y _sil_*
šli	*S l i _sp_*
šli	*S l i _sil_*
běží	*b j e Z I _sp_*
běží	*b j e Z I _sil_*
ostatní	*o s t a t J I _sp_*
ostatní	*o s t a t J I _sil_*
ostatní	*o s t a T J I _sp_*
ostatní	*o s t a T J I _sil_*
šance	*S a n c e _sp_*
šance	*S a n c e _sil_*
Frydrychové	*f r i d r i x o v E _sp_*
Frydrychové	*f r i d r i x o v E _sil_*
dnešní	*d n e S J I _sp_*
dnešní	*d n e S J I _sil_*
finálovou	*f i n A l o v y _sp_*
finálovou	*f i n A l o v y _sil_*
stane	*s t a n e _sp_*
stane	*s t a n e _sil_*
předchozích	*p Q e t x o z I x _sp_*
předchozích	*p Q e t x o z I x _sil_*

IV. Referenční přepis jedné věty a úspěšnost jejího rozpoznání

Referenční přepis	**Rozpoznaná slova**
#!MLF!#	
"/gymnastika_-_mezinarodni-zavod-pratelstvi_0_014.lab"*	
jejím	*jejím*
soupeřkám	*soupeřkám*
bylo	*Gaetano*
tedy	*tedy*
celkem	*celkem*
jasné	*jasné*
že	*že*
právě	*právě*
ona	*ona*
bude	*bude*
favoritkou	*favoritkou*
číslo	*číslo*
jedna	*jen*
dost	*deseti*
chyb	*časových*
přinesla	*kvalifikace*
i	*mužů*
kvalifikace	*mohl*
mužů	
toto	
.	

Úspěšnost rozpoznávání

gymnastika_-_mezinarodni-zavod-pratelstvi_0_014.rec: 65.00(65.00) [H= 13, D= 2, S= 5, I= 0, N= 20]