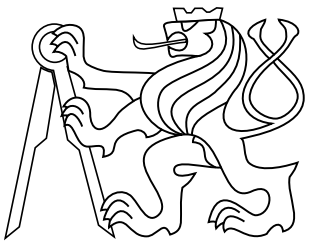
Diplomová práce

# Rozpoznávání řeči s malým slovníkem s nástroji KALDI

*Miroslav Forman*



10.ledna 2016

Vedoucí práce: doc. Ing. Petr Pollák, CSc.
Konzultant: Ing. Petr Mizera

České vysoké učení technické v Praze
Fakulta elektrotechnická, Katedra obvodů

České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra radioelektroniky

# ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: **Miroslav Forman**

Studijní program: Komunikace, multimédia a elektronika
Obor: Multimediální technika

Název tématu: **Rozpoznávání řeči s malým slovníkem s nástroji KALDI**

Pokyny pro vypracování:

1. Seznamte se s metodami rozpoznávání řeči na bázi GMM-HMM se zaměřením na rozpoznávání frází s pevnou gramatickou strukturou.

2. Realizujte rozpoznávač aplikačních frází s předpokládaným použitím pro ovládání různých uživatelských zařízení. Pro implementaci použijte nástroje ze sady KALDI.

3. Analyzujte úspěšnost navrženého rozpoznávače na datech z dostupných řečových databází SPEECON a CZKCC.

Seznam odborné literatury:

[1] X. Huang, A. Acero, H.-W. Hon. Spoken Language Processing. Prentice Hall, 2001.

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

[3] Uhlíř, J. a kol.: Technologie hlasových komunikací. Nakladatelství ČVUT, Praha, 2007.

[4] D. Povey, et al.: The Kaldi Speech Recognition Toolkit. In IEEE 2011 Workshop on Automatic Speech Recognition and Understanding, Big Island, Hawaii, US, Dec, 2011.

Vedoucí: doc.Ing. Petr Pollák, CSc.

Platnost zadání: do konce zimního semestru 2016/2017

L.S.

doc. Mgr. Petr Páta, Ph.D.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 30. 9. 2015

## Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne: _____ podpis: _____

## Poděkování

Děkuji vedoucímu práce doc. Ing Petru Pollákovi za věnovaný čas a cenné odborné rady při tvorbě této práce. Děkuji také Ing. Petru Mizerovi za vytvoření akustických modelů použitých v práci a za jeho odborné rady a připomínky při tvorbě skriptů při práci s KALDI.

# Abstrakt

Tato práce se zabývá návrhem rozpoznávače řeči s malým slovníkem pro použití ovládání různých uživatelských zařízení, zejména ovládání navigace a funkcí v automobilu. Pro tyto účely bylo vytvořeno více jednotlivých rozpoznávačů řeči zaměřených na různé odlišné úlohy rozpoznávání. Každý rozpoznávač byl sestrojen na bázi modelování kontextově nezávislých fonémů pomocí skrytých Markových modelů (HMM) s nástroji balíčku KALDI.

Diplomová práce obsahuje podrobný popis všech bloků vyskytujících se v každém z rozpoznávačů. Z databází SPEECON a Temic byly vybrány testovací promluvy, které byly použity v experimentální části práce. Parametrizací dat byly získány řečové příznaky těchto testovacích promluv. Z výše uvedených databází byla také čerpána data pro trénovací množinu pro použité akustické modely. Dále pro tuto práci byly vytvořeny kombinované jazykové modely na bázi gramatiky a unigramu pro jednotlivé položky v gramatice. Příprava dat a práce s nástroji KALDI byla provedena v prostředí Linux. U experimentální části práce byly zkoumány úspěšnosti rozpoznávání pro výše zmíněné jazykové modely pro dva připravené akustické modely. Pro vyhodnocení úspěšnosti rozpoznávání řeči bylo využíváno klasifikační kritérium WER (word error rate). Nejlepších výsledků dosahoval rozpoznávač názvů měst, který na AM1 s nulovým OOV měl pro dva druhy rozpoznávání hodnotu WER menší než 3%.

## Klíčová slova

Rozpoznávání řeči; HMM; WFST; malý slovník; jazykový model; gramatika; unigramy; výslovnostní slovník; hlasová volba; ovládání v automobilu; KALDI

# Abstract

This thesis describes the design of small vocabulary speech recognizer for application of handling various devices, mainly voice controlled navigation and other functions in automobile. For this purpose several speech recognizers aimed on different tasks of recognition were designed. Each recognizer was built on the base of modeling context-independent phones using Hidden Markov Models (HMM) and WFST approach using KALDI toolkit.

The thesis contains the detailed description of each particular block of created recognizers. The utterances from SPEECON and Temic databases were used in the experimental part. The data parametrization was used for gaining speech features of these utterances. Training data for acoustic models were also taken from above mentioned databases SPEECON and Temic. Furthermore, the language model combining unigrams of particular words and fixed grammar was designed for created recognizers. The recognizers were implemented using KALDI toolkit under Linux OS. The recognition accuracy for particular above mentioned tasks were analysed in the experimental part for two different acoustic models using standard WER criterion (word error rate). The best results were achieved for the recognizer of cities, where value of WER for two kinds of recognition with AM1 and zero OOV was less than 3%.



## Keywords

Automatic speech recognition; HMM; WFST; small vocabulary; language model; grammar; unigrams; lexicon; voice controll; automotive application; KALDI

# Obsah

# Zkratky

| | |
|---|---|
| AM | akustický model |
| ASR | Automatic Speech Recognition |
| C | kompozice |
| CMN | kepstrální průměrová normalizace |
| CMVN | kepstrální průměrová a varianční normalizace |
| CVN | kepstrální varianční normalizace |
| DFT | diskrétní kosínová transformace |
| DFT | diskrétní Fourierova transformace |
| FST | konečné stavové automaty |
| G | gramatika |
| GM | gramatický model |
| HMM | skryté Markovy modely |
| L | lexikon |
| LM | jazykový model |
| LPC | lineární prediktivní analýza |
| LVCSR | velký slovník pro rozpoznávání spojité řeči |
| MFCC | Mel-frequency Cepstral Coefficients |
| OOV | slova nevyskytující se ve slovníku |
| PLP | Perceptual Linear Predictive coding |
| SNR | odstup signál šum |
| WFST | váhované konečné stavové automaty |

# 1 Úvod

Hlasová komunikace patří mezi nejzákladnější způsoby dorozumívání se mezi lidmi. Tento způsob komunikace se lidé snaží vytvořit i pro používání a ovládání různých technologií a strojů. Tato disciplína se nazývá rozpoznávání řeči (ASR). V dnešní době je rozmach této technologie stále více a více znatelnější. Lidé ji využívají téměř na denním pořádku, ať už pomocí hlasových instrukcí zadávají do navigace cílovou destinaci nebo chtějí pomocí hlasových povelů zahájit telefonní komunikaci. Vývoj ASR začal již v polovině minulého století. Ale až poslední dobou, kdy výpočetní výkon procesorů je mnohonásobně vyšší, začala tato technologie být rozsáhle využívána. A pořád její potenciál nebyl zcela naplněn.

Rozpoznávání řeči má široké spektrum využití. Ve většině případů má za úkol věci zjednodušit a ušetřit čas. Místo manuálních povelů a úkonů využívat povely hlasové. Jeden z příkladů využití je například přepis mluveného slova do textové podoby a naopak textu do mluveného slova [3]. ASR se ale také využívá pro identifikaci mluvčího třeba v zabezpečovacích systémech. Problematika ASR zahrnuje širokou škálu příbuzných oborů od akustiky, fonetiky, teorie informací, zpracování signálu, statistiky a pravděpodobnosti až po problematiku daného jazyka [12]. Jde o velmi složitý proces, který je velmi náročný na výpočetní výkon a řeší se rozkouskováním na jednotlivé méně náročné kroky. Požadavky na rozpoznávání řeči v reálném čase jsou zcela jistě rychlost a přesnost rozpoznávání. U rozpoznávání jsou tyto vlastnosti protichůdné a musí se mezi nimi zvolit správný kompromis.

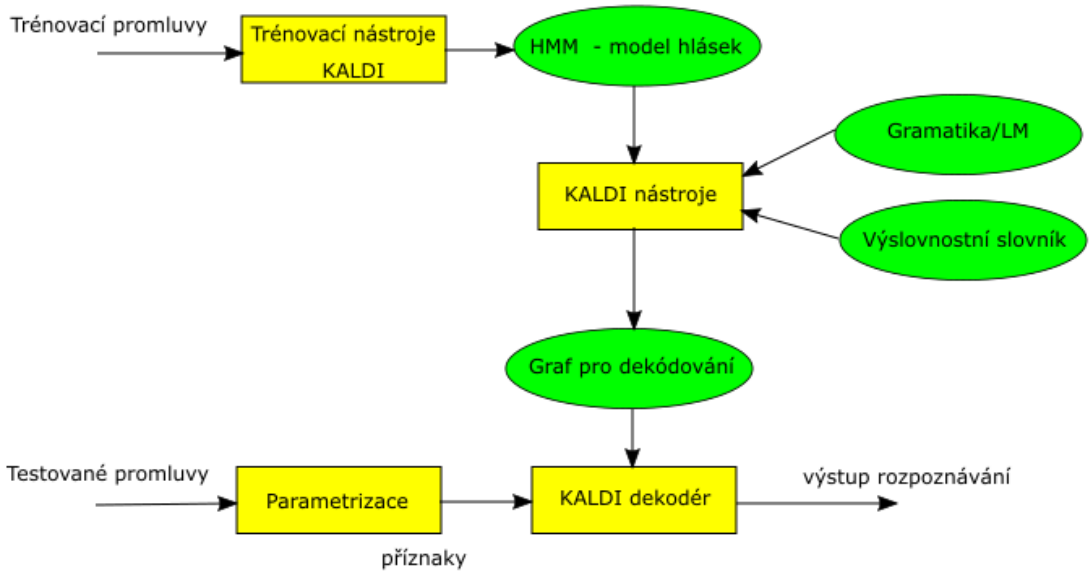
Diplomová práce se zabývá návrhem rozpoznávače řeči s malým slovníkem pro použití ovládání různých uživatelských zařízení a funkcí v automobilu, zejména hlasové ovládání navigace. Pro tyto účely bylo vytvořeno několik rozpoznávačů řeči zaměřených na odlišné úlohy rozpoznávání. Hlavní důraz byl kladen na sestavení kombinovaného jazykového modelu na bázi gramatiky a unigramu pro jednotlivé položky v gramatice.

Práce je rozdělena do tří částí. V první části jsou popsány principy vytvořených rozpoznávačů, na kterých pracují. Druhá část je věnována implementaci samotných rozpoznávačů s nástroji KALDI s různými gramatickými modely. Jsou zde popsány postupně jednotlivé bloky vytvořených rozpoznávačů. Poslední část je experimentální a jsou zde ukázány úspěšnosti rozpoznávání pro jednotlivé rozpoznávače.

# 2 Principy rozpoznávání řeči

Pod pojmem rozpoznávání řeči rozumíme převod mluveného projevu do textové podoby. Dřívější metody rozpoznávání řeči (metody celých slov) byly založeny na porovnávacích technikách. Příchozí signál byl porovnáván s referenčními vzorky uchovávanými v databázi. V této databázi musely být uchovány vzorky pro každé slovo, které rozpoznávač uměl rozpoznávat. Dále se metoda vylepšila o rozpoznávání jednoduchých slovních spojení s malým slovníkem (například číslovky). Tyto jednoduché typy rozpoznávání jsou ale nedostatečnými pro nezávislé rozpoznávače mluvené řeči s velkým slovníkem. Dnešní podoba metod rozpoznávání řeči je založena na tom, že k rozpoznávanému vzorku (akustickému signálu) hledáme optimální cestu pravděpodobnostním modelem. Numerické hodnoty a struktura v modelu jsou zkonstruovány z několika dalších dílčích modelů. Dílčí modely popisují akustiku jednotlivých hlásek (fonémů), výslovnost slov a pravidla skládání vět. Akustický model se nejprve trénuje na velkém vzorku dat řeči s různými mluvčími. Model výslovnosti je slovník obsahující výslovnost slov ve fonetické abecedě. Jazykový model může být pouze soubor pravidel pro skládání vět, ale také statistický model popisující pravděpodobnost různých kombinací slov.

Na obrázku 1 je znázorněno možné blokové schéma rozpoznávače. V této práci byly sestrojeny rozpoznávače na základě tohoto blokového schématu. Dále v této práci budou postupně rozebrány jednotlivé jejich bloky.




**Obr. 1** Blokové schéma rozpoznávače

## 2.1 Parametrizace

Pro úlohu rozpoznávání řeči je vhodné signál řeči nejprve předzpracovat a extrahovat jeho příznaky důležité pro samotné rozpoznávání. To se provádí pomocí parametrizace. Rozpoznávače pak nepracují s celým signálem řeči, ale jen s parametry, které jsou pro rozpoznávání vhodné. Pro parametrizaci signálu je důležitý poznatek, že lidská řeč je nestacionární kvaziperiodický signál, ale pro časové úseky 10-30ms se její parametry považují za stacionární. Díky tomuto poznatku lze řeč rozdělit na takovéto úseky a pracovat s jejich parametrizovanou podobou [3].

Před samotnou parametrizací signálů je nutné testovací promluvy nejprve upravit. Musí se provést preemfáze a segmentace signálu na menší úseky. Následně jsou úseky řečového signálu připraveny pro parametrizaci a ta z nich extrahuje řečové příznaky. Mezi nejčastější typy parametrizace patří parametrizace typu LPC, PLP a MFCC. Více je zde zmíněn typ MFCC. Parametrizace vychází z modelu produkce lidské řeči, proto je níže uveden její vznik.

### 2.1.1 Lidská řeč

Základem vzniku lidské řeči jsou artikulační orgány hlasového ústrojí člověka. Ty vytvářejí řečové kmity, které jsou fyzikální reprezentací řeči. Tvorbu řeči má za úkol pak hlasový trakt. Ten se skládá z dechového, hlasového a artikulačního ústrojí. Dechové ústrojí je tvořeno plícemi a svaly funkčně s nimi spjatými. Akustický řečový signál (zvuková vlna) je vybuzen právě proudem vzduchu z plic.

Hlasové ústrojí je uloženo v hrtanu a jeho nejdůležitější částí jsou hlasivky. Prostor mezi hlasivkami tvoří hlasivkovou štěrbinu. Pokud člověk nemluví, tato štěrbina je odkrytá, aby přes ni mohl volně proudit dech. Při vytváření hlasu se hlasivky stahují, stávají se pružnými a kmitají. Kmitání hlasivek je základ lidského hlasu. Frekvence kmitů hlasivek závisí na jejich vlastnostech a určuje základní tón. Pomocí tohoto procesu vznikají jednotlivé navazující segmenty tvořící řeč.

Nejmenší fonetické jednotky řeči se nazývají fonémy, jejichž hlasovým projevem jsou hlásky. Na bázi rozpoznávání fonémů už lze sestrojit rozpoznávače řeči. Při tvorbě řeči jednotlivá slova vznikají změnou parametrů hlasového traktu. Tyto změny nejsou ale skokové, to má za následek vzájemné ovlivňování předchozích a následujících hlásek. Tedy hláska může znít jinak v závislosti na sousedních hláskách. Proto lze využit i jednotky, které tyto změny popisují, například difón a trifón [13].

### 2.1.2 Preemfáze a segmentace

Při výstupu vlny z artikulačního ústrojí do volného prostoru dochází k útlumu intenzity zvuku pro vyšší frekvenční složky. Pro zvýraznění vyšších frekvenčních složek je signál filtrován filtrem FIR 1. řádu, který je definován rovnicí:

$$s'[n] = s[n] - m \cdot s[n-1], \tag{1}$$

kde m je koeficient preemfáze. Jeho typická hodnota je 0.95 - 1.

Při segmentaci signálů dochází k nežádoucímu prosakování ve spektru. Prosakování vzniká při nespojitostech signálu na okrajích segmentu. V jeho důsledku se pak ve spektru objevují frekvence, které nemají s původním signálem nic společného. Pro omezení tohoto vlivu se váhuje signál vhodným oknem:

$$s_w[n] = s(n) \cdot w(n) \tag{2}$$

Nejčastěji se využívá Hammingovo okno, které má optimální vlastnosti pro ASR. Popis Hammingova okna délky N vzorků je vyjádřeno rovnicí:

$$w[n] = 0.54 - 0.46 \cos \frac{2\pi n}{N} \tag{3}$$

KALDI balíček při parametrizaci dat používá Poveyho okno (podle Daniela Poveyho - hlavní autor balíčku KALDI ), které má podobné spektrum i vlastnosti jako okno Hanningovo. Vztah pro toto okno je uveden níže [10].

$$w[n] = (0.5 - 0.5 \cos \frac{2\pi n}{N})^{0.85} \tag{4}$$

### 2.1.3 Parametrizace MFCC

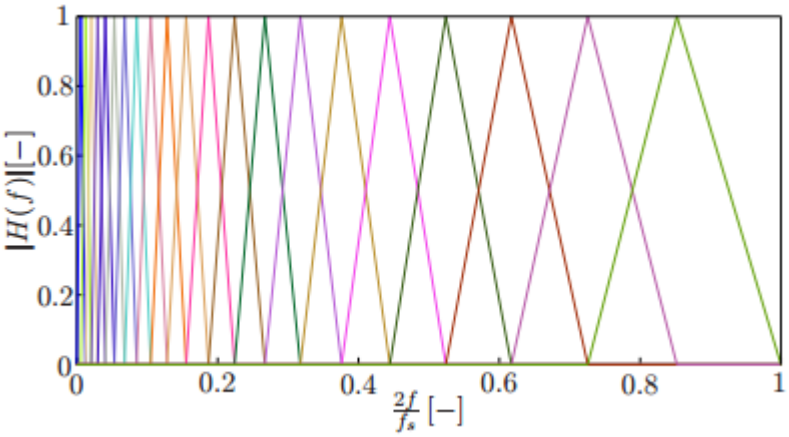
Parametrizace MFCC je založena na bázi melovských kepstrálních koeficientů a je pro ni důležitý poznatek, že lidský sluch pracuje na principu spektrální analýzy [4]. Jeho vnímání zvukových frekvencí je nelineární a s rostoucí frekvencí se snižuje jeho frekvenční rozlišení. Toto je u MFCC zahrnuto pomocí převodních vztahů mezi klasickou frekvencí s osou v Hz a melovskou frekvencí s osou v melech:

$$w_{mel} = 2595 \log_{10}(1 + \frac{f}{700}) \tag{5}$$

$$f = 700 \cdot (10^{\frac{f_{mel}}{2595}} - 1) \tag{6}$$

Na spektrum signálu se aplikuje banka filtrů s trojúhelníkovou přenosovou frekvenční charakteristikou a nelineární frekvenční osou v melech. Banka filtrů je

tvořena M pásmy, typický počet pásem je 20 - 30. Všechny filtry v bance mají stejnou šířku pásma v melovské stupnici a mají mezi sebou 50% překryv. Melovská banka filtrů je na obrázku 2, kde každé její pásmo je vyznačeno jinou barvou.




**Obr. 2** Melovská banka filtrů (převzato z [1])

Nejčastější realizace banky filtrů je pomocí DFT (diskrétní Fourierova transformace) transformace, tzn. frekvenční charakteristika jednoho filtru $H_{mel.j}[k]$ je určena v bodech odpovídajících řádu DFT. Výkonové melovské kepstrum se vypočítá v jednotlivých pásmech j následujícím vztahem:

$$g_j = \ln \cdot \sum_{k=0}^{N/2} |S[k]|^2 H_{mel.j}[k], \tag{7}$$

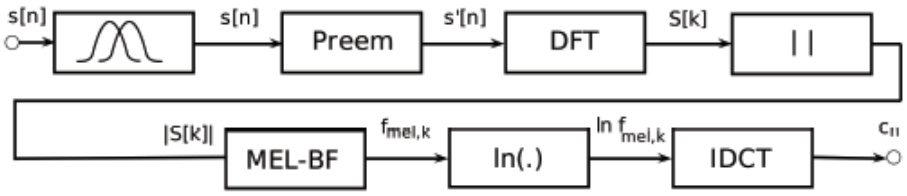
kde j=0, 1, . . . , P a P je celkový počet pásem melovské banky filtrů.

Výsledných N melovských kepstrálních koeficientů se dopočítá DCT (diskrétní kosínová transformace) pro daná výkonová pásma melovského kepstra g (následující rovnice).

$$c_i = \sqrt{\frac{2}{P}} \sum_{j=1}^{P} g_j \cos(\frac{\pi i}{P}(j - 0.5)), \tag{8}$$

pro i=0, 1, . . . , N

Zjednodušené schéma je znázorněno na následujícím obrázku (obrázek 3).




**Obr. 3** Schéma výpočtu mel koeficientů (převzato z [5])

### 2.1.4 Dynamické koeficienty

Pomocí výše uvedených postupů se získají statické příznaky, jejich vektor je vyčíslen jen pomocí vzorků řečového signálu v daném okénku. Statické příznaky, ale nepopisují změny v posloupnosti jednotlivých příznakových vektorů. Proto se dále k statickým vektorům přidávají dynamické koeficienty [17]. Tyto koeficienty popisují změny v časovém okolí sledovaného segmentu a tím zlepšují úspěšnost rozpoznávání. Vztahy jednotlivých koeficientů jsou uvedeny níže:

- **Delta koeficienty $\Delta$**

$$\Delta_k[i] = \frac{\sum_{m=1}^{M} m(c_k[i+m] - c_k[i-m])}{\sum_{m=1}^{M} m^2}, \tag{9}$$

  kde $\Delta$ je vektor delta koeficientů, c je vektor statických koeficientů a M značí typickou hodnotu okolí pro aproximaci derivace.

- **Akcelerační koeficienty - delta $\Delta$ - delta $\Delta$-$\Delta$**

$$\Delta - \Delta_k[i] = \frac{\sum_{m=1}^{M} m(\Delta_k[i+m] - \Delta_k[i-m])}{\sum_{m=1}^{M} m^2}, \tag{10}$$

  kde $\Delta$ je vektor delta koeficientů, c je vektor statických koeficientů a M značí typickou hodnotu okolí pro aproximaci derivace.

### 2.1.5 Energie signálu

Další veličina, která se přidává k získaným příznakům, je energie signálu. Pro popsání energetické úrovně segmentu signálu se mohou použít tři metody, ale při parametrizaci se použije pouze jedna z nich. Mezi tyto metody patří:

- **Energie signálu**

$$E = \sum_{n=1}^{N} s^2[n] \tag{11}$$

- **Logaritmus energie**

$$E = \log \sum_{n=1}^{N} s^2[n] \tag{12}$$

- **Nultý kepstrální koeficient** (vztah 8 pro $i$=0)

## 2.2 Normalizační techniky

Rozpoznávače pro lidskou řeč jsou velmi často ovlivňovány okolním prostředím. Některé z nich mají ale schopnost rozpoznávat s podobnou úspěšností i při změně jejich pracovního prostředí. Tyto druhy rozpoznávačů jsou nazývány robustní. Pokud v trénovací množině dat budou stejné okolní podmínky jako u testovacích signálů, bude to vést k lepší úspěšnosti rozpoznávání. Proto pro dosažení podobné úspěšnosti rozpoznávání v jiném prostředí je nutné mít velkou a variabilní množinu trénovacích dat. Na úspěšnost rozpoznávání budou mít dopad tyto vlastnosti: řečník, prostředí a přenosový kanál [15].

Vliv řečníka zasahuje do většiny rozpoznávacích úloh. V testovací množině by měly být zastoupeny promluvy různých řečníků s jinými vlastnostmi. Mezi důležité vlastnosti řečníků mající vliv na rozpoznávání jsou: pohlaví řečníka, jeho původ (různý přízvuk), věk (dospělí vs. dítě) a příležitost, u které nahrávka byla pořízena (čtení hlásek, poslechový test, diktát atd ...)

Vliv prostředí je další kategorií, která má výrazný vliv na úspěšnost rozpoznávání. Jde hlavně o pozadí (šum, hluk) v promluvách a jeho stacionární či nestacionární charakter. Stacionární šum je téměř konstantní na pozadí celé stopy. Může být způsobený třeba klimatizací nebo motorem v autě a snižuje SNR - tedy odstup signálu od šumu. Nestacionární šum je zastoupený hlavně impulsními zvuky, které nemají konstantní amplitudu v čase.

Přenosový kanál je ovlivněn druhem zvoleného mikrofónu. Téměř každý mikrofón má jiné vlastnosti (například přenosové a směrové charakteristiky) a použití mikrofonu s jinými vlastnostmi při rozpoznávání bude mít také dopad na úspěšnost rozpoznávání. Tyto vlastnosti ovlivňují hlavně šířku pásma, frekvenční charakteristiku a doprovodné rušení pozadí.

### 2.2.1 CMN/CMVN

Normalizace CMN (cepstral mean normalization) a CMVN (cepstrum mean and variance normalization) kompenzují vliv různých druhů řečníků, vliv variability prostředí i vliv přenosového kanálů. CMN a CMVN jsou momentové příznakové normalizace aplikující se na kepstrální vektory [15]. Redukují konvoluční zkreslení na bázi odečtení průměrných hodnot kepstra. Tyto hodnoty mohou být napočítány více způsoby například po promluvách nebo pro určité řečníky.

Pro CMN se musí nejprve spočítat střední hodnota v každém segmentu a tu pak odečíst od původního koeficientu v čase $t$. Vztah pro výpočet střední hodnoty je na následujícím řádku:

$$\mu_t[i] = \frac{1}{N} \sum_{n=0}^{N-1} x_n[i], \tag{13}$$

kde $i$ je příznakový vektor počítán přes konečnou délku okna N.

Odečtení průměrných hodnot kepstra od původního koeficientu se provede pomocí této rovnice:

$$\hat{x}_t[i] = x_t[i] - \mu_t[i], \tag{14}$$

kde $\mathrm{x}_t$[i] je $i$-tý komponent originálního příznakového vektoru v čase $t$.

CMVN je kombinace CMN normalizace s CVN (cepstral variance normalization). CVN se vypočítá podobně jako CMN, ale místo průměrných hodnot se zde počítá s variancí (rovnice 15).

$$\hat{x}_t[i] = \frac{x_t[i] - \mu_t[i]}{\sigma_t}, \tag{15}$$

kde $\sigma_t$ je variance (rovnice 16).

$$\sigma_t^2[i] = \frac{1}{N} \sum_{n=t-\frac{N}{2}}^{t+\frac{N}{2}-1} (x_n[i] - \mu_t[i])^2 \tag{16}$$

## 2.3 Metody rozpoznávání řeči

V této kapitole bude více přiblížen statistický přístup, rozpoznávání řeči na bázi HMM (skryté Markovy modely) a problematika váhovaných konečných automatů a akceptorů.

### 2.3.1 Statistický přístup u rozpoznávání řeči

U statistického rozpoznávání izolovaných slov pro příklad máme vektor příznaků $\mathbf{O} = \{\mathbf{o}_1,\mathbf{o}_2,...,\mathbf{o}_t\}$, který obsahuje parametrické vektory řečového signálu pro

každý rámec. Jeden vektor $\mathbf{o}_t$ může například obsahovat MFCC nebo PLP koeficienty. Vektor W = $\{w_1,w_2,...,w_t\}$ je slovník všech slov. Potom pro nalezení nejpravděpodobnějšího slova ze slovníku k danému vektoru příznaků platí následující rovnice.

$$\bar{W} = \arg\max P(W|\mathbf{O}) = \arg\max \frac{P(W)P(\mathbf{O}|W)}{P(\mathbf{O})}, \tag{17}$$

kde P(W) je pravděpodobnost posloupnosti rozpoznávaných slov, P($\mathbf{O}$) značí pravděpodobnost posloupnosti vektorů pozorování a P($\mathbf{O}$|W) ukazuje pravděpodobnost generování vektoru příznaků $\mathbf{O}$ při vyslovení slova W [11]. Tedy ke každému slovu $w_i$ bude přiřazen model M, generující sekvenci příznaků $\mathbf{O}$.

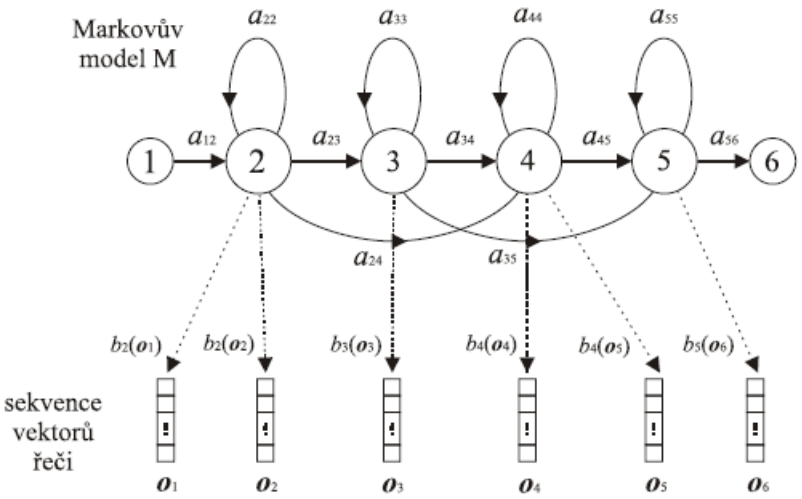
### 2.3.2 Skryté Markovy modely

HMM modely jsou zvláštním případem stochastických konečných automatů, kde je generována diskrétní sekvence přechodů z jednotlivých stavů za účasti přechodové pravděpodobnosti. Na obrázku 4 je příklad HMM modelu se čtyřmi emitujícími stavy(2, 3, 4 a 5) a dvěma stavy neemitujícími. První a poslední stav je neemitující a slouží jako vstup a výstup z modelu.

Typicky pro řeč se využívá levo-pravý model. V každém okamžiku $\mathbf{t}$ se změní stav podle pravděpodobnosti přechodu $\mathbf{a}_{i,j}$ do stavu $\mathbf{s}_j$. A při této změně se generuje vektor pozorování $\mathbf{o}_t$ s výstupní pravděpodobností $\mathbf{b}_j(\mathbf{o}_t)$(rovnice 18) [16].

$$P(\mathbf{O}|M) = \sum_X a_{x(0)x(1)} \prod_{t=0}^{T} b_{x(t)}(\mathbf{o}_t) a_{x(t)x(t+1)}, \tag{18}$$

kde $\mathbf{x}$ je stav, ve kterém model je v čase $\mathbf{t}$.




**Obr. 4** Markovův model se šesti stavy (převzato z [14])

Přechodové pravděpodobnosti $a_{i,j}$ jsou konstantní po celou dobu průběhu. A jejich součet vycházející z jednoho stavu je roven 1.

$$\sum_j a_{i,j} = 1. \tag{19}$$

Na obrázku 4 nalezneme tři druhy přechodových pravděpodobností - $\mathbf{a}_{i,i}$ pravděpodobnost setrvání v nacházejícím se stavu, $\mathbf{a}_{i,i+1}$ pravděpodobnost přechodu do následujícího stavu a $\mathbf{a}_{i,i+2}$ pravděpodobnost přeskočení stavu. Častý je jejich zápis v maticové formě.

$$\mathbf{A} = \begin{pmatrix} 0 & a_{1,2} & 0 & 0 & 0 & 0 \\ 0 & a_{2,2} & a_{2,3} & a_{2,4} & 0 & 0 \\ 0 & 0 & a_{3,3} & a_{3,4} & a_{3,5} & 0 \\ 0 & 0 & 0 & a_{4,4} & a_{4,5} & 0 \\ 0 & 0 & 0 & 0 & a_{5,5} & a_{5,6} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \tag{20}$$

Funkce distribuce výstupní pravděpodobnosti $\mathbf{b}_j(\mathbf{o}_t)$ nám udává rozdělení pravděpodobnosti vektoru $\mathbf{o}_t$ ve stavu $\mathbf{s}_j$ a v čase $\mathbf{t}$. Každý prvek vektoru $\mathbf{o}_t$ můžeme ve stavovém grafu (obrázek 5) přiřadit do shluku definovaným středem $\mu$ a variancí $\mathbf{r}$. Tyto shluky se dají popsat Gaussovským rozdělením:

$$N(\mathbf{o}_t; \mu_t; \Sigma_j) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp[-\frac{1}{2}(\mathbf{o}_t - \mu_t)^T \sum_j^{-1} (\mathbf{o}_t - \mu_t)], \tag{21}$$

kde $\mathbf{n}$ je dimenze vektoru $\mathbf{o}_t$ a $\Sigma$ je kovarianční matice.

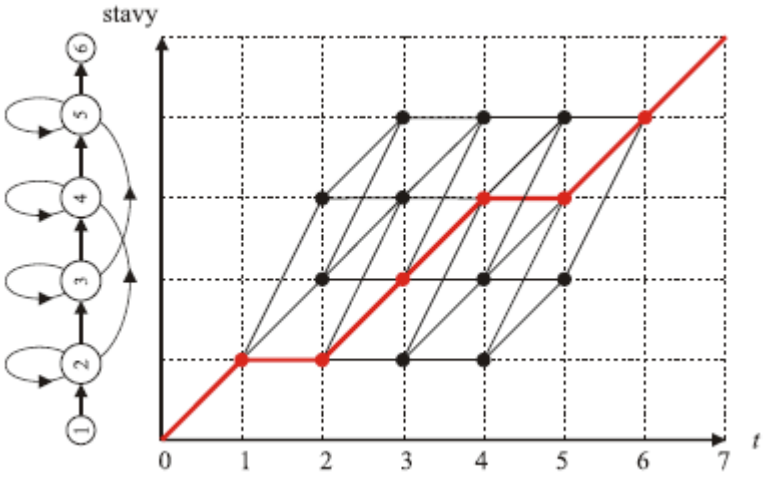
Ze stavového grafu (obrázek 5) vyčteme pořadí stavů, kterými model prochází. U našeho příkladu stavová posloupnost byla X = [1, 2, 2, 3, 4, 4, 5, 6]. Délka stavové posloupnosti závisí na počtu vektoru pozorování. Pokud $\mathbf{K}$ je počet vektorů pozorování, pak stavová posloupnost bude $\mathbf{K} + 2$ dlouhá. Pravděpodobnost generování $\mathbf{o}_t$ modelem $\mathbf{M}$ při zvolení cesty $\mathbf{X}$ je definována takto:

$$P(\mathbf{O}, X|M) = a|_{x(0)x(1)} \prod_{t=1}^{T} b_{x(t)}(\mathbf{o}_t) a_{x(t)x(t+1)}. \tag{22}$$

Je to součin všech pravděpodobností na cestě $\mathbf{X}$. Pro náš příklad má tvar: $\mathrm{a}_{12}\mathrm{b}_2(\mathbf{o}_1)\mathrm{a}_{22}\mathrm{b}_2(\mathbf{o}_2)\mathrm{a}_{23}\mathrm{b}_3(\mathbf{o}_3)\ldots\mathrm{a}_{56}\mathrm{b}_5(\mathbf{o}_6)$.

Ve většině případů je známá pouze posloupnost $\mathbf{o}_t$, stavová posloupnost $\mathbf{X}$ nikoliv. Existují dva způsoby výpočtu pravděpodobnosti generování posloupnosti $\mathbf{o}_t$ modelem $\mathbf{M}$. Prvním způsobem je výpočet sumy pravděpodobností všech možných cest modelem $\mathbf{M}$:

$$P(\mathbf{O}|M) = \sum_X a_{x(0)x(1)} \prod_{t=1}^{T} b_{x(t)}(\mathbf{o}_t) a_{x(t)x(t+1)}. \tag{23}$$




**Obr. 5** Stavový graf - přiřazení $\mathbf{o}_t$ jednotlivým stavům (převzato z [14])

Druhým způsobem je Viterbiho pravděpodobnost, která uvažuje pouze nejpravděpodobnější stavovou posloupnost:

$$P^*(\mathbf{O}|M) = \max_X \{a_{x(0)x(1)} \prod_{t=1}^{T} b_{x(t)}(\mathbf{o}_t) a_{x(t)x(t+1)}\}. \tag{24}$$

### 2.3.3 Váhované konečné automaty a akceptory

Rozpoznávače s velkými slovníky, které jsou založeny na HMM modelech, mohou být reprezentovány pomocí váhovaných konečných automatů (WFST).V této práci je využíván KALDI toolkit, který tyto automaty využívá [10]. WFST převádějí sekvenci vstupních znaků na sekvenci výstupních znaků s váhou odpovídající cestě s těmito znaky. Dále udávají váhu přenosu také na vstupní a výstupní znak. Do váhy jsou zahrnuty pravděpodobnosti přechodu, doba trvání znaku, případně nějaké postihy nebo jakákoliv vlastnost, co by mohla přispět k váze cesty, od počátečního symbolu ke konečnému. WFST jsou definovány na *semiringu* S jako:

$$T = (\Sigma, \Delta, Q, I, F, E, \lambda, \rho), \tag{25}$$

kde $\Sigma$ je konečná vstupní množina symbolů, $\Delta$ je konečná výstupní množina symbolů, Q je konečná sada stavů, I je podmnožinou Q a je sadou vstupních stavů, F je podmnožinou Q a je sadou koncových stavů, E je konečná sada přechodů mezi stavy, $\lambda$ je vstupní váhovací funkcí a $\rho$ je výstupní váhovací funkcí.

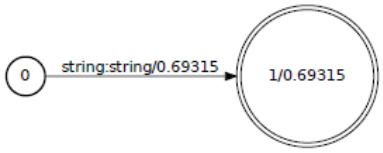
*Semiring* je systém se dvěma operátory $\oplus$ a $\otimes$. Následující tabulka (tabulka 1) ukazuje příklady použití *semiringu* ve WFST tool-kitech [6].

**Tab. 1** Příklady *semiringu*

| **Semiring** | **Množina** | $\oplus$ | $\otimes$ | $\bar{0}$ | $\bar{1}$ |
|---|---|---|---|---|---|
| Boolean | {0,1} | $\vee$ | $\wedge$ | 0 | 1 |
| Pravděpodobnost | $\Re_+$ | $+$ | $\times$ | 0 | 1 |
| Log | $\Re \cup \{-\infty,+\infty\}$ | $\otimes_{log}$ | $+$ | $+\infty$ | 0 |
| Tropical | $\Re \cup \{-\infty,+\infty\}$ | min | $\wedge$ | 0 | 1 |

Kde $\vee$ je logický OR, $\wedge$ je logický AND a x $\otimes_{log}$ y = -log($e^{-x}$ + $e^{-y}$).

Operátor $\otimes$ se používá pro výpočet váhy na dané cestě, $\oplus$ slouží pro sečtení nákladu sekvence přes všechny možné cesty. U pravděpodobnostního *semiringu* váhy reprezentují reálná čísla nebo pravděpodobnosti a u logaritmického jsou váhy dány záporným logaritmem pravděpodobnosti. Na obrázku 6 je ukázán jednoduchý model přenosu e WFST na množině E. E množina reprezentuje sadu všech konečných stringů ve slovníku. Pro grafické znázornění WSFT se používá program graphviz. Díky grafickému zobrazení je lehké porozumět, co jaký automat dělá. Pokud se ale pracuje s větším množstvím dat ve grafu, jeho vykreslení už není možné. Stavy jsou v grafu znázorněny jako kruhy, kde počáteční stav je tučně zvýrazněn a konečný stav je znázorněn dvojitým provedením kruhu [2].




**Obr. 6** WFST jednoduchý model

- p[e] značí z jakého stavu se automat přesouvá (0)
- n[e] značí do jaké stavu se automat přesouvá (1)
- i[e] je vstupní symbol (string)
- o[e] je výstupní symbol (string)
- w[e] je danná váha (0.5)

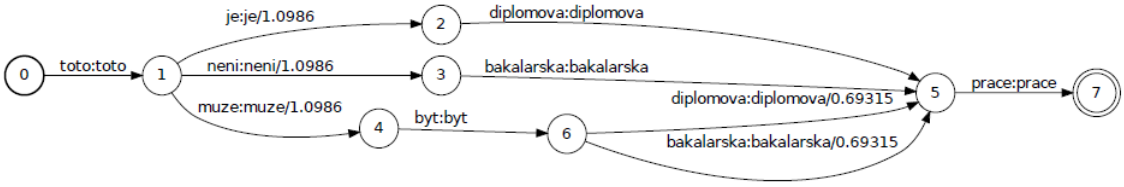
Jedna možná cesta $\pi$ je zkonstruována ze sekvence přechodů v automatu jako $\pi = e_1,\ldots,e_k$. Tato cesta je sestavena ze stringů z množiny E a je váhována logaritmickým *semiringem*.

- pokud p[$\pi$]=n[$\pi$] automat je nastaven do smyčky
- w[$\pi$] je váha cesty, která byla získána pomocí $\otimes$ z jednotlivých přechodů na cestě
- P($Q_1$,$Q_2$) obsahuje všechny možné cesty ze stavu $Q_1$ do stavu $Q_2$
- P($Q_1$,x,y,$Q_2$) obsahuje všechny možné cesty ze stavu $Q_1$ do stavu $Q_2$, které přenáší string x do stringu y

[T](x,y) je součet všech cest, které přenáší string x do stringu y. Vypočítá se z následující rovnice:

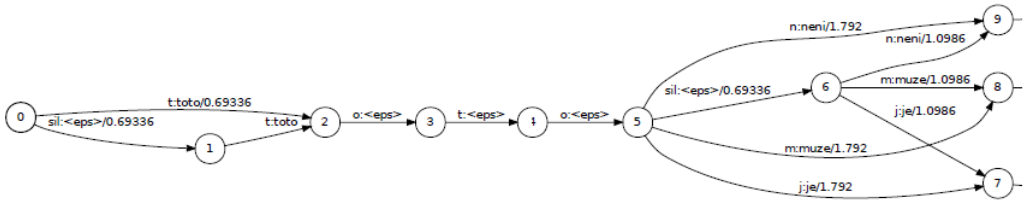
$$[T](x,y) = \oplus_{\pi \epsilon P(I,x,y,F)} \lambda(p[\pi]) \otimes w[\pi] \otimes \rho(n[\pi]) \tag{26}$$

WFST se hodně využívají v automatickém rozpoznávání řeči (ASR). Jejich příklad využití je znázorněn na obrázku 7, kde je jednoduchý automat s konečně stavovým jazykovým modelem. Povolené slova jsou specifikovány mezi jednotlivými stavy na cestě, stejně jako jejich pravděpodobnosti výskytu. Tyto automaty obsahují sadu stavů, počáteční stav, konečný stav a přechody mezi nimi. Tedy každý přechod má svůj zdrojový stav, stav kam se má dostat, popis co přenáší a váhu (pravděpodobnost) přechodu.




**Obr. 7** Jednoduchý model WFST na bázi slov

Na obrázku 5 je ukázán automat na bázi slov. Ve skutečnosti každé slovo je pak nalezeno dekodérem ve slovníku (lexiconu), kde je znázorněna výslovnost hledaného slova, a udělá se to samé s výslovností. Tedy výslovnost se rozdělí na jednotlivé stavy. Pak každý přechod mezi těmito stavy znamená jeden foném. Protože každý člověk může vyslovovat stejné slovo jinak, tak i tady jsou znázorněné jednotlivé možné cesty pomocí vah (pravděpodobností). Následně výslovnostní model nahradí slovo v gramatice (obrázek 8).




**Obr. 8** Jednoduchý model WFST na bázi fonémů, symbol *sil* značí pauzu

Fonetická stromová reprezentace může být v tento moment použita za účelem snížení nadbytečné informace na cestě a tedy zvýšení efektivity rozpoznávače.

Toto platí obzvlášť pro rozpoznávače s velkými slovníky. Strom je následně převeden na HMM strukturu pomocí softwarového rozhraní, které je většinou spjato s danou topologii modelu (monofón, trifón).

Váhované konečné akceptory (WFSA) jsou hodně podobné WFST a liší se pouze tím, že nemají *o[e]* - výstupní symbol. U akceptorů se určuje dále:

- P($Q_1$,x,$Q_2$) obsahuje všechny možné cesty ze stavu $Q_1$ do stavu $Q_2$, které akceptují strinx x
- [A](x) je součet všech cest se vstupním stringem x

$$[A](x) = \oplus_{\pi \epsilon P(I,x,F)} \lambda(p[\pi]) \otimes w[\pi] \otimes \rho(n[\pi]) \tag{27}$$

Jejich využití v ASR je při práci se slovníkem.

### 2.3.4 Kompozice a determinizace automatů

Kompozice se u WFST používá ke kombinování různých vrstev reprezentace. Například výslovnostní slovník může být složen z gramatiky na bázi slov, který bude vytvářet přechod mezi fóny a slovy vyskytujícími ve slovníku. WFST reprezentují binární vztah mezi stringy [6].

**Tab. 2** Slovník, fóny a číselný vztah mezi stringy

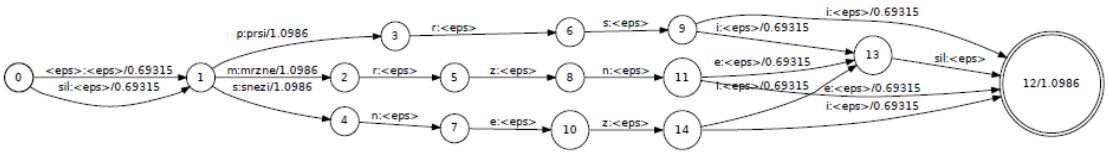
| Slovník | Fóny | Číselný vztah mezi stringy |
|---|---|---|
| auto a u t o | a 1 | auto 1 |
| | u 2 | |
| | t 3 | |
| | o 4 | |

Kompozici dvou automatů je reprezentována jejich vztahem $\mathbf{T} = \mathbf{T}_1 \circ \mathbf{T}_2$, který obsahuje přesně jednu cestu ze stringu $u$ do stringu $w$ pro každý propojený pár. Automat $\mathbf{T}_1$ mapuje přechod stringu $u$ do stringu $v$ a druhý automat $\mathbf{T}_2$ mapuje přechod stringu $v$ do stringu $w$. Celková váha cesty z $\mathbf{T}$ je pak spočtena z jednotlivých vah souvisejících cest z $\mathbf{T}_1$ a z $\mathbf{T}_2$. Výpočet je proveden stejnou operací jako je vypočtena váha cesty z jednotlivých přechodů. Pokud váhy přechodů jsou reprezentovány pravděpodobnostmi, výpočetní operací je $\prod$. Pokud je reprezentace provedena pomocí logaritmu pravděpodobnosti nebo záporným logaritmem pravděpodobnosti, výpočetní operací je $\sum$. Operace pro výpočet vah u WFST jsou specifikovány jako výše zmíněný *semiring*. Pro kompozici $\mathbf{T}$ platí:

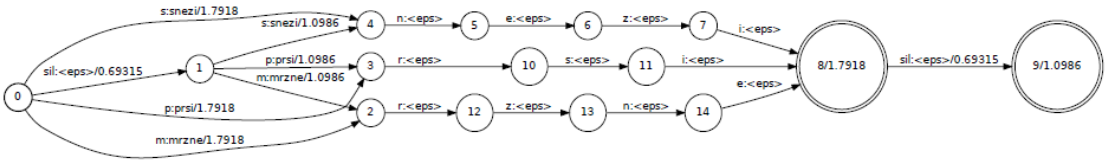
- její počáteční stav je pár počáteční symbol z $\mathbf{T}_1$ a počáteční symbol z $\mathbf{T}_2$
- její koncové stavy jsou složeny z páru koncový stav z $\mathbf{T}_1$ a koncový stav z $\mathbf{T}_2$
- obsahuje přechod $t$ z ($q_1$,$q_2$) do ($r_1$,$r_2$) pro každý pár přechodu $t_1$ z $q_1$ do $r_1$ a $t_2$ z $q_2$ do $r_2$, tak aby byl výstupní symbol $t_1$ stejný jako vstupní symbol $t_2$

- přechod $t$ je složen ze vstupního symbolu $t_1$ a výstupního symbolu $t_2$ a jeho váha je kombinací vah $t_1$ a $t_2$

U deterministických automatů má každý stav maximálně jeden přechod s jednotlivým vstupním symbolem. Rozdíl mezi nedeterministickým a deterministickým automatem je ukázán na obrázcích 9 a 10. Klíčovou výhodou deterministického automatu od nedeterministického je obsah ne-nadbytečné informace. Tím, že obsahuje pouze jednu cestu vstupního stringu, se snižuje náročnost na čas a úložný prostor pro provedení procesu. Toto je zejména důležité u ASR u výslovnostního slovníku. Z těchto důvodů se v ASR používá algoritmus převodu nedeterministických automatů na automaty deterministické, které jsou ekvivalentní. Dva WFST automaty jsou k sobě ekvivalentní, pokud mají stejný výstupní string a váhu do všech vstupních stringů. Distribuce váhy nebo výstupních symbolů na cestě ale stejná být nemusí [6]. Pokud se tento algoritmus aplikuje na svazek řetězových automatů, kde každý reprezentuje výslovnost jednoho slova, dostaneme stromovou strukturu automatu. Ve skutečnosti u některých automatů dostaneme mnohem kompaktnější strukturu než je ta stromová. Všechny nedeterministické konečné automaty (FST) mohou být převedeny na deterministické automaty. U WFST existují ve velmi malém množství případy, kdy toto nelze provést. Jedná se u některé případy WFST automatů ve smyčce.




**Obr. 9** Jednoduchý nedeterministický automat




**Obr. 10** Jednoduchý deterministický model předchozího automatu

Pro odstranění nadbytečných cest algoritmus musí vypočítat kombinaci váhování všech cest se stejným vstupním symboly. Cílem je nechat nejvíce pravděpodobnou cestu se stejnými symboly, to vede na použití *Viterbiho* aproximace.

### 2.3.5 Sestrojení rozpoznávače

Tato část vysvětluje, jak se sestrojují statisticky kompilovaný a optimalizovaný rozpoznávač na bázi WFST, který převádí kontextově závislé fóny na slova. Tento rozpoznávač je spojený z akustického, slovníkového a gramatického modelu.

**L** značí výslovnostní slovník, který například může mít tvar a formát slovníku jako v tabulce 2. Rozkládá jednotlivé stringy na samostatné fóny. **G** značí gramatický model, ve kterém jsou dány pravidla stringů z jakých a do kterých stavů se přesouvají. Dále se v gramatickém modelu musí určit konečný stav/stavy. Počáteční stav je vždy v nule. Ukázka dané gramatiky pro obrázek 7 je zobrazena v následující tabulce (tabulka 3). Více podrobněji je gramatický model řešen v kapitole 3.4.2.

**Tab. 3** Gramatický model pro obrázek 7, kde p značí pravděpodobnost přechodu

| **Vstup. stav** | **Výstup. stav** | **Vstup. symbol** | **Výstup. symbol** | **-log(p)** |
|---|---|---|---|---|
| 0 | 1 | toto | toto | 0 |
| 1 | 2 | je | je | 1.0986 |
| 1 | 3 | neni | neni | 1.0986 |
| 1 | 4 | muze | muze | 1.0986 |
| 2 | 5 | diplomova | diplomova | 0 |
| 3 | 5 | bakalarska | bakalarska | 0 |
| 4 | 6 | byt | byt | 0 |
| 6 | 5 | diplomova | diplomova | 0.69315 |
| 6 | 5 | bakalarska | bakalarska | 0.69315 |
| 5 | 7 | prace | prace | 0 |

| **Konečný stav** | **-log(p)** |
|---|---|
| 7 | 0 |

Všechny stringy, které se vyskytují v gramatickém modelu rozpoznávače musí být obsažené ve slovníku. Pokud je toto splněno rozpoznávač propojí všechny stringy v gramatice se stringy ve slovníku a tím se získá jejich rozklad na fonémy (rovnice 28).

$$\mathbf{L} \circ \mathbf{G} \tag{28}$$

Dále rozpoznávač použije kompozici pro implementování kontextově nezávislé substituce. Jednou z hlavních výhod používání automatů v ASR je jejich schopnost zobecnění kontextově nezávislé substituce symbolu do kontextově závislého modelu. Například pokud máme trifónový model v kontextově nezávislém automatu, převod do kontextově závislého automatu se provede jednoduše pomocí kompozice [6]. Nejprve si musíme sestavit kontextově závislý automat, který mapuje kontextově nezávislé fóny do kontextově závislých trifónů. Tento automat bude mít stav pro každou dvojici fónů a přechody pro každý kontextově závislý model. V následujícím kroku prohodíme vztah automatu, aby mapoval kontextově závislé trifóny do kontextově nezávislých fónů, pomocí výměny vstupních symbolů na výstupní a naopak. Pokud **C** reprezentuje kontextovou závislost kontextově závislých trifónů do kontextově nezávislých fónů, potom:

$$\mathbf{C} \circ (\mathbf{L} \circ \mathbf{G}) \tag{29}$$

vytvoří automat mapující kontextově nezávislé fóny do jednotlivých slov - *stringů*, které jsou obsaženy v gramatice **G**. Dalším krokem aplikujeme na tento automat minimalizaci a determinizace za předpokladu, že automat může být determinizován.

$$\min(\det(\mathbf{C} \circ (\mathbf{L} \circ \mathbf{G}))) \tag{30}$$

Automat může být determinizován za předpokladu, že všechny jeho komponenty jsou determinizovatelné. Pokud **G** je n-gramový jazykový model, tak **C** i **G** jsou determinizovatelné. Problém může být u slovníku **L**. Ve slovníku se mohou vyskytovat dva různé stringy se stejnou výslovností, pak je $L$ nedeterminizovatelný. Pro vyřešení determinizovatelnosti se vytvoří $\bar{L}$, který se bude lišit od **L**, pouze v přidaných *disambiguate* symbolech na konci každého slova se stejnou výslovností. Tyto symboly mají za úkol odlišit od sebe jednotlivá slova se stejnou výslovností. Po vytvoření $\bar{L}$ se musí pro tuto variantu slovníku vytvořit nová $\bar{C}$ a konečný rozpoznávač má následující tvar:

$$\min(\det(\bar{C} \circ (\bar{L} \circ \mathbf{G}))) \tag{31}$$

# 3 Realizace rozpoznávače s malým slovníkem

Cílem této práce bylo vytvoření rozpoznávače řeči s malým slovníkem s nástroji KALDI. Pro tyto účely bylo vytvořeno více variant rozpoznávačů řeči s odlišnými úlohami rozpoznávání. Konkrétně byly vytvořeny varianty pro rozpoznávání jmen, příjmení, jmen a příjmení, číslic, měst a ulic. Vytvoření samotných rozpoznávačů se skládá z více dílčích částí. Nejdříve je zde popsán balíček KALDI, pomocí kterého byly rozpoznávače sestaveny. Další částí je práce s testovacími promluvami: databáze promluv, se kterými se pracovalo, jejich příprava a následná parametrizace. Jednou z hlavních částí diplomové práce je vytvoření slovníku a gramatiky pro jednotlivé varianty rozpoznávačů. Posledním krokem je vlastní dekódování dat a určení úspěšnosti rozpoznávání.

## 3.1 Kaldi

Balíček Kaldi se využívá pro práci zaměřenou na rozpoznávání řeči. Je napsaný v C++ a jeho licence je pod hlavičkou Apache License v2.0. Používá knihovny OpenFst - konečných stavových automatů. Dále součástí tohoto balíčku jsou maticové knihovny ve standardu ATLAS a CLAPACK, díky které podporuje lineární algebru. Na jeho vytvoření se nejvíce podílel Daniel Povey [10].

Základem tohoto balíčku jsou operace na základě WSFT. Jak je vidět na obrázku (obrázek 1, kapitola 2), využívá se hlavně jako nástroj pro trénování dat a dekodér. Balíček obsahuje mnoho skriptů s příklady využití a je na uživateli jestli se už nechá inspirovat některými z nich. Uživatel by ale měl být přinejmenším seznámen se skriptováním v shellu/bashi. Často je nutné zasáhnout do skriptu a udělat úpravu pro správné použití. Dále je balíček optimalizován pro práci s LVCSR a využívá trubky (pipes), které výrazně snižují náročnost na hard disku pro I/O. Balíček se využívá hlavně na linux/unix typu systémech.

Pomocí skriptů se využívají například tyto funkce Kaldi balíčku: *featbin/*, *sgmbin/*, *gmmbin/*, *fstbin/*, *transform/*, *sgmm/*, *decoder/*, *hmm/*, *gmm/*, *feat/*, *matrix/*, *tree/*, *util/*, *fstext/* a *lm/*, které dále mohou využívat ATLAS/CLAPACK a OpenFST knihovny. Na následujících řádcích je ukázán příklad užití balíčku ve skriptu.

```
gmm-decode-faster  --verbose=2 \
                   --config=conf/file \          standartní argumenty
                   --print-args=true \
                   --acoustic-scale=0.09 \       specifické argumenty aplikace
                   model.mdl \
                   ark:decoding_graph.input \    I/O soubory
                   scp:feature.input \
                   ark:text_output \
```

## 3.2 Příprava dat z databází

V práci se pracovalo se dvěma řečovými databázemi SPEECON a Temic. Z těchto databází byly vybrány trénovací data pro použité akustické modely. Dále z uvedených databází byly použity promluvy jako testovací množiny dat.

Pro rozpoznávače se z databazí SPEECON a Temic musely vybrat testovací promluvy. To se udělalo pomocí dvou skriptů local/speecon_data_prep.sh a local/temic_data_prep.sh. Tyto skripty byly částečně předpřipravené, ale bylo potřeba do nich specifikovat typy testovacích promluv pro každou databázi. Cílem tohoto kroku je vytvoření pro každou testovací množinu dat mapování mezi typem mluvčího a názvů promluv a naopak (skp2utt, utt2spk). Dále se v kroku vytvoří přepis testovacích promluv a data pro parametrizaci (test.text a wav.scp). Ve výše uvedených skriptech se vytvářejí požadované složky na chtěné cestě a volají se další skripty pro tvorbu požadovaných souborů. Na následujícím listu je ukázána forma a účel vytvořených souborů.

- **test.text** soubor obsahující identifikátory promluv a jejich obsah. Používá se u dekódování při zjišťování úspěšnosti rozpoznávání.

  ```
  SA097CO1      pelhřimov
  SA097CP1      miloslava navrátilová
  fe79bo102032  kamenný újezd
  ```

- **wav.scp** soubor se kterým pracuje nástroj parametrizace dat. Na jeho formát má vliv, jestli je u přípravy dat zvolen parametr ctucopy nebo ne. Pokud není je nutné přidat převod z raw formátu promluvy do wav formátu pomocí nástroje SOX. V projektu se nepracovalo s parametrem ctucopy.

  ```
  SA097CO1 sox -t raw -r 16000 -e signed-integer -b 16 -c 1 data/*
  */SPEECON/ADULT1CS/BLOCK09/SES097//SA097CO1.CS0 -t wav -|
  ```

  * značí pokračování řádku v souboru

- **utt2spk** typ souboru obsahující id promluv a jejich řečníků, používá se u vytváření MFCC parametrizace

| | |
|---|---|
| SA097CO1 | SA097 |
| SA097CP1 | SA097 |
| fe79bo102032 | fe79b |

- **spk2utt** typ souboru obsahující id řečníků a jejich promluv, využívá se u výpočtu cmvn statistik

| | |
|---|---|
| SA097 | SA097CO1 |
| SA097 | SA097CP1 |
| fe79b | fe79bo102032 |

Dále jsou v této sekci uvedeny popisy databází a testované množiny dat z nich sestavené.

### 3.2.1 SPEECON

SPEECON je databáze řečových promluv obsahující foneticky bohatý materiál. V databázi se vyskytují promluvy od 590 rozdílných dospělých řečníků a 50 dětských. Každý řečník namluvil zhruba 322 promluv (cca 30 minut záznamu). Obsahem záznamů jsou například: číslice, jména a příjmení, názvy měst, ulic, foneticky bohaté věty a emailové adresy. Nahrávky jsou pořízené ze čtyř různých míst a to z: domova, kanceláře, automobilu a z veřejného místa. Každá promluva byla nahrána čtyřmi různými mikrofony lišícími se typem a vzdáleností od řečníka. Typ mikrofonu je v databázi popsán v příponě souboru (CS0-CS3). Data s označením CS0 a CS1 byla pořízená headsetovým mikrofonem a měla by mít nejlepší kvalitu. Použité mikrofony při nahrávání promluv u databáze SPEECON jsou uvedeny v následujícím listu [8].

- **CS0** Senheiser ME104 ve vzdálenosti menší než 10cm
- **CS1** Nokia Lavalier HDC-6D ve vzdálenosti menší než 10cm
- **CS2** pro tento kanál byly promluvy nahrány ze čtyř mikrofonů s podobnými vlastnostmi ve vzdálenosti 1m: Senheiser ME64, AKG Q400 Mk3 T, Mikrofonbaun Haun MBNM-550 E-L a Peiker ME15/V520-1
- **CS3** Mikrofonbaun Haun MBNM-550 E-L ve vzdálenosti 2-3m

Promluvy v databázi byly nahrány se vzorkovacím kmitočtem 16kHz s lineárním šestnácti-bitovým kvantováním.

### 3.2.2 Temic

Temic je česká databáze řečových promluv pořízených v prostředí automobilu s různými podmínkami, kde mluvčí vždy seděl na sedadle spolujezdce. Obsah promluv této databázi zahrnuje například foneticky bohaté věty, jména, příjmení, jména a příjmení, číslovky, telefonní čísla, názvy měst, ulic atd. Řečníci v promluvách jsou zastoupeny oběma pohlaví ve zhruba stejném množství a pocházejí z různých míst v České Republice [5]. Promluvy se dělí v databázi do tří skupin podle věku mluvčího. Do databáze byly nahrány dva sety promluv s různými mikrofony:

- **ils_s** Senheiser pro blízkou vdálenost a **ils_a** AKG pro větší vzdálenost
- **ils_p** Peiker pro větší vzdálenost

Vzorkovací frekvence pro Temic byla 44,1 kHz s lineárním šestnácti-bitovým kvantováním a následně byly promluvy převzorkovány do konvence SPEECON na 16 kHz $f_s$ (ils_s,ils_a a ils_p do CS0, CS2 a CS3).

### 3.2.3 Testovací množiny v databázi SPEECON

Názvy v této databázi mají tuto formu SA097CO1.CS0, kde prvních pět znaků znázorňuje typ mluvčího, poslední tři znaky před tečkou popisují typ promluvy a znaky za tečkou určují typ kanálu.

- **Množina jmen - příjmení** - v promluvách je vždy řečeno vlastní jméno a příjmení, název promluvy má poslední tři znaky "*CP1.*"

- **Množina číslic** - skládá se ze tří druhů promluv: "*CI[1-4].*" jedna izolovaná číslice, "*CB[1-2].*" sekvence izolovaných číslic, "*CC1.*" string složený z číslic

- **Množina měst** - název promluvy má poslední tři znaky "*CO1.*"

- **Množina ulic** - název promluvy má poslední tři znaky "*CO2.*"

### 3.2.4 Testovací množiny v databázi Temic

Promluvy - jejich názvy v této databázi mají tuto formu fe74bo301021.CS0, kde prvních pět znaků znázorňuje typ mluvčího, další dva znaky popisují typ promluvy a znaky za tečkou určují znovu typ kanálu.

- **Množina jmen a příjmení** - v promluvách je vždy řečeno vlastní jméno a příjmení, název promluvy obsahuje znaky "*v1*"

- **Množina vlastních jmen** - v promluvách je vždy řečeno jen vlastní jméno, název promluvy obsahuje znaky "*o4*"

- **Množina příjmení** - v promluvách je vždy řečeno jen příjmení, název promluvy obsahuje znaky "*o3*"

- **Množina jmen** - skládá se z výše uvedených množin pro Temic.

- **Množina číslic** - skládá se ze dvou druhů promluv: "*u1*" - promluva obsahuje 10 číslic, "*u3*" - promluva obsahuje jednocifernou číslici

- **Množina měst** - název promluvy obsahuje znaky "*o1*"

- **Množina ulic** - název promluvy obsahuje znaky "*o2*"

## 3.3 Parametrizace dat testovacích množin

V této části je potřeba zparametrizovat data ve všech testovacích množinách. U našeho rozpoznávače pro parametrizaci jsou využity předpřipravené skripty z KALDI steps/make_mfcc.sh a steps/compute_cmvn_stats.sh. K parametrizaci je nutné mít připravené soubory wav.scp, utt2spk a spk2utt. Výstupními soubory jsou pak feats.scp a cmvn.scp, které odkazují na napočítané příznaky a CMVN statistiky. Jejich formát je uveden na následujících řádcích. Využívají se u dekódování rozpoznávače.

- **feats.scp** soubor obsahující identifikátory promluv a cestu k souboru napočítaných řečových příznaků.

  SA097CO1 /workspace/forman/recipes_ctu/speecon_forman/s6/mfcc/*
  */conf/mfcc.conf/raw_mfcc_speecon_city.1.ark:9

- **cmvn.scp** soubor obsahující id řečníků a cestu k souboru napočítaných cmvn koeficientů.

  SA097 /workspace/forman/recipes_ctu/speecon_forman/s6/mfcc/conf/*
  */mfcc.conf/cmvn_speecon_city.ark:6

Skript steps/make_mfcc.sh využívá pro vytvoření MFCC nástroj:

*compute-mfcc-feats −−verbose=2 –config=$mfcc_config *
*scp,p:$logdir/wav_$name.JOB.scp ark:- \ | *
*copy-feats −−compress=$compress ark:- *
*ark,scp:$mfccdir/**
*/raw_mfcc_$name.JOB.ark,$mfccdir/raw_mfcc_$name.JOB.scp *

Skript steps/compute_cmvn_stats.sh využívá pro výpočet CMVN nástroj:

*compute-cmvn-stats −−spk2utt=ark:$data/spk2utt scp:$data/feats.scp ark*

Více je samotná parametrizace popsána v kapitole 2.1.

## 3.4 Jazykové modely na bázi gramatiky

Tato část práce zobrazuje základní popis k vytvoření gramatiky a slovníku pomocí KALDI a následně jejich vytvoření pro každý druh rozpoznávače zvlášť.

### 3.4.1 Lexikon

Tato část je věnována tvorbě slovníku (**L**) a tvorbě podpůrných souborů, které jsou potřeba pro jeho vytvoření. Slovník je důležitou součástí celkového rozpoznávače a zahrnuje všechny možné stringy, které se mohou vyskytovat v gramatickém modelu. Jeho úlohou je přepis jednotlivých kontextově závislých stringů do kontextově nezávislých fonémů. Může obsahovat mnohem více stringů než je v gramatickém modelu. Naopak pokud by gramatický model obsahoval string neobsažený ve slovníku, rozpoznávač by vyhodil chybu. Je důležité dbát, aby string z gramatiky byl naprosto totožný se stringem ve slovníku (velká malá písmena a stejné kódování textu). Rozpoznávač ve skutečnosti neumí přímo pracovat s proměnou typu string a je nutné přiřadit všem stringům unikátní id v podobě integeru. Proto nestačí mít slovník v textové podobě, ale musí se připravit do formy L.fst (pro KALDI).

Pro tvorbu **L** je nutné si připravit tyto soubory: lexicon.txt, phones.txt, silence_phones.txt, nonsilence_phones.txt, optional_silence.txt a words.txt.

Soubor phones.txt může obsahovat znaky pro ticho nebo pauzu mezi slovy u promluv, ale také znaky pro nějaké rušivé elementy v promluvách. Z tohoto souboru se dále vytvoří soubory silence_phones.txt (silence), nonsilence_phones.txt (nonsilence) a optional_silence.txt (optional silence) viz tabulka 4. Znak sil v našem případě značí ticho, <eps> značí prázdný stav. Soubory plynoucí z phones.txt jsou pro všechny námi sestrojené rozpoznávače stejné a váží se na použitý akustický model (AM).

**Tab. 4** Soubory zahrnující fonémový model

| phones.txt | silence | nonsilence | optional silence |
|---|---|---|---|
| <eps> | sil | <eps> | sil |
| sil | | a | |
| a | | aa | |
| aa | | au | |
| au | | b | |

Pro experiment v této práci byl vytvořen jeden lexicon.txt pro číslice, jeden pro jména a příjmení a 6 variant pro města a ulice. Tento soubor obsahuje přepis slov do fonémů. Více je toto probráno v podkapitolách tvorby lexikonu a gramatiky pro jednotlivé rozpoznávače.

Pokud v tomto bodě máme připraveny soubory pro fonémy a lexicon.txt musíme přiřadit jednotlivým fonémům a stringům id (integer). Soubor pro převod stringů na id je words.txt (tabulka 5). Soubor phones.txt musí být identický s použitými fonémy v AM, tedy id a počet všech použitých fonémů v AM musí být totožný jako u slovníku. Například v AM se nevyskytovaly stringy s přehlasovanou hláskou ö. Do slovníku byla zahrnuta ulice flöglova. Skript pro vytváření přepisu do fonémů by hlásku ö převedl jako ooo, ale ta se nevyskytovala v AM a rozpoznávač by dále se slovníkem neuměl pracovat. Proto se tato hláska musí nahradit jiným podobným fonémem vyskytujícím se v AM, kterým je například "e".

**Tab. 5** Soubory words.txt a phones.txt

| words.txt | phones.txt |
|---|---|
| <eps> 0 | <eps> 0 |
| !sil 1 | sil 1 |
| abraham 2 | a 2 |
| abrahamová 3 | aa 3 |
| abrahám 4 | au 4 |
| abrahámová 5 | b 5 |

Přímé vytvoření slovníku se vytvořilo pomocí před-připraveného skriptu v KALDI /utils/prepare_lang.sh. Tento skript převede tyto připravené soubory na L a L_disambig (viz kapitola 2.3.5). Převedení souborů do podoby, se kterou KALDI umí pracovat (.fst), se provede následujícím kódem:

```
fstcompile      --isymbols=phones_disambig.txt \
                --osymbols=words.txt \
                --keep_isymbols=false \
                --keep_osymbols=false lexicon_disambig_fst.txt \
| fstaddselfloops
| fstarcsort     --sort_type=olabel > L_disambig.fst \
```

### 3.4.2 Gramatika

Pod pojmem gramatika v ASR se rozumí soubor pravidel, jak rozpoznávač má pracovat. Například pokud rozpoznávač rozpozná mužské vlastní jméno, už mu dále nemůže přiřadit ženské příjmení. Gramatika je vytvořena téměř pro každou variantu rozpoznávače ve dvou variantách: zerogramu a unigramu LM (jazykový model). Pod pojmem zerogram se skrývá gramatika, kde všechny slova mají stejnou pravděpodobnost výskytu. Unigram LM už pracuje s jiným pravděpodobnostním rozdělením. Naše rozpoznávače pracují s pravděpodobnostmi vypočítanými na základě četností výskytu daných slov v určitých níže popsaných zdrojů.

Pro vytvoření gramatiky si nejprve musíme vytvořit soubor grammar_fst.txt a ten následně převést do souboru G.fst, se kterým KLADI už umí pracovat:

```
fstcompile  --isymbols=words.txt \
            --osymbols=words.txt \
            --keep_isymbols=false \
            --keep_osymbols=false \
            grammar_fst.txt > G.fst \
```

Soubor grammar_fst.txt musí obsahovat počáteční stav, který je vždy v nule, a stav konečný. Jeho forma je znázorněna v tabulce 3 (kapitola 2.3.5). Pro KALDI jsou váhy značeny jako -log(p), kde p je pravděpodobnost přechodu a log je logaritmus přirozený. Pro sestavení všech druhů zmíněných gramatik byl vytvořen skript *local/create_rule_based_G.sh*. Gramatiky a vytváření lexikonu pro jednotlivé varianty sestrojených rozpoznávačů jsou uvedeny v následujících podkapitolách.

### 3.4.3 Lexikon a gramatika pro rozpoznávač číslic

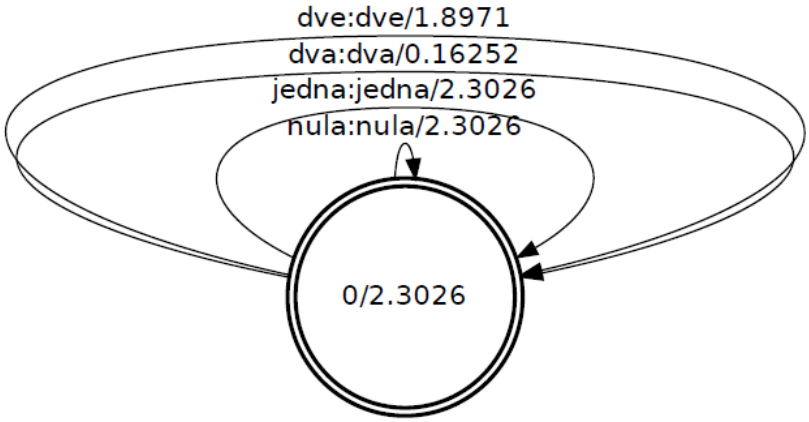
U rozpoznávání jednotlivých číslic bude lexikon zahrnovat jednotlivé čislice (0-9). Tedy velmi jednoduchý slovník by byl složen z 10 stringů. Dále se musí uvažovat i jiné vyslovení jednotlivých číslic, například číslice 7 může být vyslovena jako sedm ale i sedum. Pro všechny možné výslovnostní varianty jednotlivých číslic se čerpalo z četnostního výskytu číslic (tabulka 6). Druhý sloupec této tabulky zahrnuje všechny stringy obsažené v našem slovníku pro číslice (tabulka 7).

Gramatika pro rozpoznávání jednotlivých číslic je nejjednodušší ze zmíněných gramatik. Její forma je uvedena v tabulce 8. Rozpoznávač u této varianty pracuje v cyklu. Počáteční stav - stav 0, je zároveň i stavem konečným (obrázek 11). Rozpoznávač je nastaven do cyklu, protože není předem jasné kolik číslic bude v promluvě.

Takto rozpoznávač může rozpoznat jedno i více číslic za sebou. Rozdělení pravděpodobnosti jednotlivých číslic (0-9) je rovnoměrné, ale berou se v potaz jednot-

**Tab. 6** Základní tvary a četnosti číslic

| Číslice | Základní tvary | Četnost tvaru v procentech |
|---|---|---|
| 0 | nula | 100 |
| 1 | jedna | 100 |
| 2 | dva | 85 |
| | dvě | 15 |
| 3 | tři | 100 |
| 4 | čtyři | 79 |
| | štyři | 11 |
| | štyry | 6 |
| | čtyry | 4 |
| 5 | pět | 100 |
| 6 | šest | 100 |
| 7 | sedum | 77 |
| | sedm | 23 |
| 8 | osum | 78 |
| | osm | 22 |
| 9 | devět | 100 |




**Obr. 11** Zjednodušený gramatický model číslic

livé výslovnostní varianty číslic. Například číslice *2* má pravděpodobnost výskytu 1/10, která je rozdělena do dvou výslovnostních variant: dva a dvě. Každá výslovnostní varianta jednotlivé číslice má pak i svoji pravděpodobnost podle četnosti výskytu dané varianty. Číslice *2* je v 85 procentech zastoupena výslovností dva, tedy celková pravděpodobnost výslovnosti dva v naší gramatice je 0.1 * 0.85.

**Tab. 7** Lexicon.txt pro číslice

| String | Výslovnost |
|---|---|
| nula | n u l a |
| jedna | j e d n a |
| dva | d v a |
| dvě | d v j e |
| tři | t rr i |

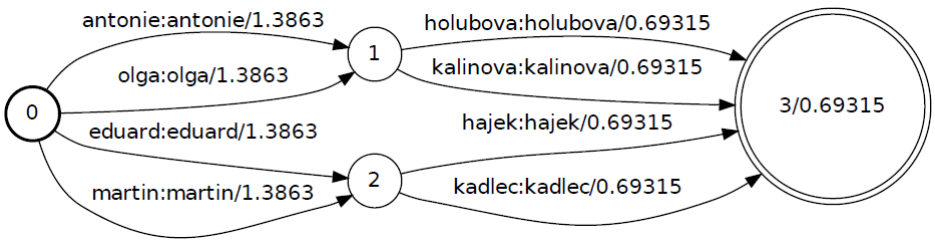
**Tab. 8** Gramatika pro rozpoznávání číslic

| Vstup. stav | Výstup. stav | Vstup. symbol | Výstup. symbol | -log(p) |
|---|---|---|---|---|
| 0 | 0 | nula | nula | 2.3 |
| 0 | 0 | jedna | jedna | 2.3 |
| 0 | 0 | dva | dva | 2.465 |
| 0 | 0 | dvě | dvě | 4.2 |
| 0 | 0 | tři | tři | 2.3 |
| 0 | | | | 0 |

### 3.4.4 Lexikon a gramatika pro rozpoznávač jmen a příjmení

U rozpoznávání jmen a příjmení jsou vytvořeny už gramatiky jako zerogram i unigram LM. U zerogramu je vytvořena gramatika se stejným pravděpodobnostním rozdělením - všechny jména z jednotlivých stavů mají stejnou pravděpodobnost výskytu. Druhý model je udělaný pro pravděpodobnosti vypočítané z četností jmen a příjmení v ČR - každé jméno i příjmení má různou pravděpodobnost výskytu. U obou variant je součet pravděpodobností ze vstupních stavů vždy 1. Vytvořený lexicon.txt je pro obě dvě varianty totožný.

Gramatika jmen a příjmení je rozdělena do 4 stavů. Z počátečního stavu (ze stavu 0) jsou dva možné přechody. Přechod ze stavu 0 do stavu 1 je pro ženské vlastní jméno, pro mužské vlastní jméno to je přechod ze stavu 0 do stavu 2. Ženskému příjmení je přiřazen přechod ze stavu 1 do stavu 3, tomu mužskému ze stavu 2 do stavu 3. Stav 3 je stav konečný. Na obrázku 12 je znázorněn zjednodušený gramatický model pro rozpoznávač jmen.




**Obr. 12** Zjednodušený gramatický model celých jmen

Pro vytvoření této gramatiky se musely vytvořit čtyři podpůrné soubory, pro každý přechod jeden soubor obsahující možné rozpoznávané stringy. Tedy podpůrný soubor pro přechod 0-1 obsahoval všechna možná ženská vlastní jména. Seznamy jmen a příjmení jsou použity z databáze četnosti jmen a příjmení ministerstva vnitra. Tato databáze byla aktualizovaná k srpnu 2015. V naší gramatice jsou obsáhlá všechna vlastní jména z této databáze s četností 150 a vyšší a všechna příjmení s četností 100 a více.

**Tab. 9** Gramatika pro rozpoznávání jmen a příjmení

| **Vstup. stav** | **Výstup. stav** | **Vstup. symbol** | **Výstup. symbol** | **-log(p)** |
|---|---|---|---|---|
| 0 | 1 | marie | marie | 6.7522 |
| 0 | 1 | jana | jana | 6.7522 |
| 0 | 2 | jiří | jiří | 6.7522 |
| 0 | 2 | jan | jan | 6.7522 |
| 1 | 3 | nováková | nováková | 9.037 |
| 1 | 3 | svobodová | svobodová | 9.037 |
| 2 | 3 | novotný | novotný | 9.037 |
| 2 | 3 | dvořák | dvořák | 9.037 |
| 3 | | | | 0.6931 |

Rozdíl mezi vytvořenými zerogramem a unigramem LM je v souboru akorát v přiřazené váze (v tabulce 9 poslední sloupec). První model pracuje se stejným pravděpodobnostním rozdělením. V naší gramatice je obsaženo 514 ženských vlastních jmen a 340 mužských vlastních jmen. Pravděpodobnost výskytu jednoho unikátního vlastního jména pro tento model ze stavu 0 do stavu 1 (nebo 2) je 1/854. Podobné je to i pro příjmení, ale jak ženské tak mužské jdou z jiného stavu a na pravděpodobnost má vliv jen počet daného příjmení. Ženských příjmení je v modelu obsaženo 8413 stejně jako těch mužských.

U unigramu LM jmen a příjmení se pracuje s četnostní pravděpodobností. Důvodem k tomuto kroku, je aby rozpoznávač přiřazoval častější jména nejasným rozpoznáváním. Četnosti jsou vzaty z databáze četnosti jmen a příjmení ministerstva vnitra. Pro přiřazení pravděpodobnosti jednotlivým jménům se vydělí četnost jména součtem četností všech jmen obsažených v daném stavu.

Lexicon.txt pro obě varianty obsahuje výslovnosti všech vyskytujících se jmen a příjmení použitých v gramatice. Je to 17680 stringů. Příklad lexiconu.txt pro jména a příjmení je uveden v tabulce 10.

Tento druh gramatiky je vytvořen zejména pro promluvy obsahující jméno a příjmení, ale protože byly k dispozici i množiny testovacích promluv s jen vlastním jménem nebo s jen příjmením, byly vytvořeny i gramatiky pro rozpoznávání pouze vlastních jmen a pouze příjmení. Předchozí gramatika by se sice měla umět vypořádat s těmito typy testovacích promluv, tím že obsahuje znak pro prázdný

**Tab. 10** lexicon.txt pro jména a příjmení

| String | Výslovnost |
|---|---|
| hana | h a n a |
| hanička | h a nn i cc k a |
| hanka | h a ng k a |
| hanna | h a n a |
| hedvika | h e d v i k a |
| helena | h e l e n a |

stav (<eps>), ale zcela jistě nebude dosahovat ideálních výsledků pro testovací promluvy složené pouze z vlastních jmen respektivě jenom z příjmení.

Tyto další varianty rozpoznávání jmen a příjmení můžou použít stejný lexikon jako používá gramatika rozpoznávající jména i příjmení. Tento lexikon obsahuje všechna jména i příjmení. Gramatika pouze pro vlastní jména je velice jednoduchá, z počátečního stavu 0 do stavu 1 jsou všechna vlastní jména a stav 1 je stav konečný. Podobné to je i pro gramatiku pro pouze příjmení, kde přechod 0-1 obsahuje všechna příjmení. Zde se ale musí upravit i pravděpodobnosti, protože jak mužské tak ženské příjmení jdou ze stejného stavu, budou mít tedy společný základ pro výpočet pravděpodobnosti.

### 3.4.5 Lexikon a gramatika pro rozpoznávač měst

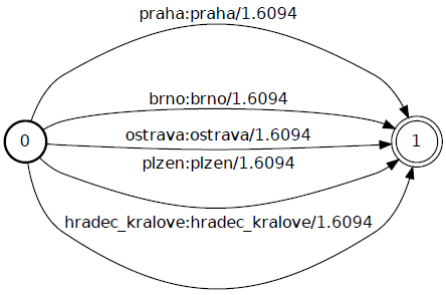
U gramatiky měst byly možné dva přístupy řešení. Problematika této gramatiky je v tom, že město může být jednoslovné (Praha), ale také může obsahovat více slov (Nové město na Moravě). Gramatiky pro oba přístupy byly vytvořeny jako zerogram i unigram LM. Unigram LM je podle četností výskytu měst ze starší verze zlatých stránek. Jelikož v této databázi se nevyskytovala všechna města co byla v testovacích promluvách, byly vytvořeny pro každý přístup 3 druhy slovníků a 6 druhů gramatik (3x zerogram a 3x unigram LM). Nejdříve byl vytvořen slovník pouze z měst vyskytujících se v testovací množině pro města (SPEECON i Temic), tedy s nulovým OOV (out of vocabulary). U četnostní gramatiky pro tento slovník byla přiřazena četnost z databáze četností měst pro města co se v ní vyskytovaly, a pro města co v ní nebyly, byla doplněna průměrná hodnota z těchto četností. Druhý slovník je vytvořen pro města jen z databáze četností měst s četností 5 a více. Třetí slovník byl vytvořen z měst s četností 50 a více z této databáze. OOV je u těchto variant slovníků rozdílné pro každý slovník a variantu testovací množiny. Druhý slovník má procento OOV menší než třetí, je v něm obsaženo více měst.

Prvním možným přístupem je braní názvu každého města jako jedno-slovnou proměnou. U implementace této myšlenky je nutné nahrazení mezery mezi slovy názvu měst jiným znakem. V této práci je zvoleno "_". Tímto se z každého města stane jedno-slovný string a gramatika bude vypadat velice jednoduše. Přechod z počátečního stavu 0 do koncového stavu 1 bude obsahovat město Praha,

Nové_město_na_Moravě atd. (tabulka 11 a obrázek 13). Následně před samotným porovnáváním správnosti dekódování se zase musí nahradit znak "_" mezerou. V tabulce 11 jsou uvedeny zástupci měst s nejvyšší četností z naší databáze pro rozpoznávání.

**Tab. 11** Gramatika pro rozpoznávání měst - nahrazení mezery

| Vstup. stav | Výstup. stav | Vstup. symbol | Výstup. symbol | -log(p) |
|---|---|---|---|---|
| 0 | 1 | praha | praha | 1.295 |
| 0 | 1 | brno | brno | 2.641 |
| 0 | 1 | ostrava | ostrava | 3.148 |
| 0 | 1 | plzeň | plzeň | 3.501 |
| 0 | 1 | pardubice | pardubice | 4.006 |
| 0 | 1 | hradec_králové | hradec_králové | 4.087 |
| 1 | | | | 0 |




**Obr. 13** Zjednodušený gramatický model měst - nahrazení mezery

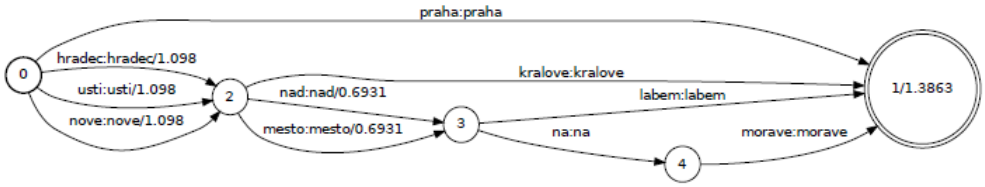
Lexikony pro oba přístupy musí být různé. U prvního přístupu se lexicon.txt skládá ze všech jednotlivých měst obsažených v gramatice (tabulka 12), kde stejně jako u ní jsou nahrazeny mezery znakem "_". Výslovnost u lexikonu tento znak přeskakuje - tedy rozdíl mezi výslovností slova "Hradec Králové" a "Hradec_Králové" není žádný.

**Tab. 12** Lexicon.txt města - nahrazení mezer

| String | Výslovnost |
|---|---|
| benátky_nad_jizerou | b e n aa t k i n a d j i z e r ou |
| brandýs_nad_labem | b r a n d ii s n a d l a b e m |
| chlumec_nad_cidlinou | ch l u m e c n a d c i d l i n ou |
| frenštát_pod_radhoštěm | f r e n ss t aa t p o d r a d h o ss tt e m |
| jablonec_nad_nisou | j a b l o n e c n a d nn i s ou |
| kostelec_nad_černými_lesy | k o s t e l e c n a d cc e r n ii m i l e s i |

Druhý přístup je mnohem složitější, kde každé slovo v názvu města je bráno jako samostatná proměná (obrázek 14). Musí se zde vytvořit pravidla pro skládání

gramatiky, proto pro tento přístup byl vytvořen skript, který z databáze měst rozřadil města s jedním slovem do přechodu z počátečního stavu 0 do konečného stavu 5. První slovo města obsahujícího více slov bylo přiřazeno do přechodu 0-1. Přechodu ze stavu 1 do stavu 5 bylo přiřazeno druhé slovo města s přesně dvěma slovy a druhá slova měst obsahující více než dvě slova byla dána do přechodu 1-2. Přechod 2-5 obsahoval třetí slova měst s přesně třemi slovy, následně třetí slova měst s více jak třemi slovy byla přidělena do přechodu 2-3. Přechod 3-5 sloužil pro čtvrtá slova měst s přesně čtyřmi slovy. Přechody 3-4 a 4-5 sloužily pro čtvrtá resp. pátá slova měst složených z pěti slov. V databázi měst, ze které bylo čerpáno, se nevyskytovalo město s více jak pěti slovy (tabulka 13).




**Obr. 14** Zjednodušený gramatický model měst - druhý přístup

**Tab. 13** Gramatika pro rozpoznávání měst - druhý přístup

| **Vstup. stav** | **Výstup. stav** | **Vstup. symbol** | **Výstup. symbol** | **-log(p)** |
|---|---|---|---|---|
| 0 | 5 | praha | praha | 0 |
| 0 | 1 | hradec | hradec | 1.098 |
| 0 | 1 | ústí | ústí | 1.098 |
| 0 | 1 | nové | nové | 1.098 |
| 1 | 5 | králové | králové | 0 |
| 1 | 2 | nad | nad | 0.6931 |
| 1 | 2 | město | město | 0.6931 |
| 2 | 5 | labem | labem | 0 |
| 2 | 3 | na | na | 0 |
| 3 | 5 | moravě | moravě | 0 |
| 4 | | | | 1.386 |

Část skriptu je uvedena níže. Nejdříve se ze souboru s četnostmi s formátem sloupců četnost - město, vybraly jen města. Dále se vytvořily soubory pro města s přesně 1,...,5 slovy a soubory s více jak 1,...,4 slovy. Následovalo rozřazení slov do jednotlivých přechodů. Posledním krokem bylo sloučení všech slov z jednotlivých přechodů do jediného souboru jako základ slov pro slovník.

```
cut -f 2-6 cities_sorted_100.iso | tr '[:upper:]' '[:lower:]' > text.txt
cat text.txt | awk '{if (NF == 1){print $0}}' | sort -u -o one_word_city.txt
cat text.txt | awk '{if (NF != 1){print $0}}' | sort -u -o 2to5_word_city.txt
cat text.txt | awk '{if (NF == 1){print $1}}' | sort -u -o part_zero_five.txt
cat text.txt | awk '{if (NF != 1){print $1}}' | sort -u -o part_zero_one.txt
```

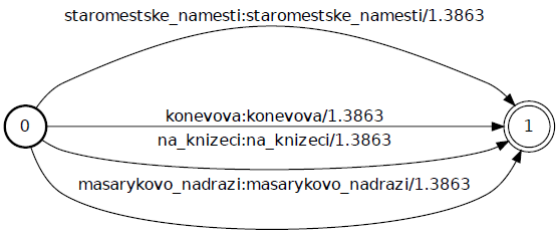
Lexicon.txt u této varianty rozpoznávání jmen a příjmení tedy obsahuje všechny slova co jsou obsažena v názvu měst: "Hradec", "Králové", "Česká", "Lípa" (tabulka 14).

**Tab. 14** Lexicon.txt města - druhý přístup

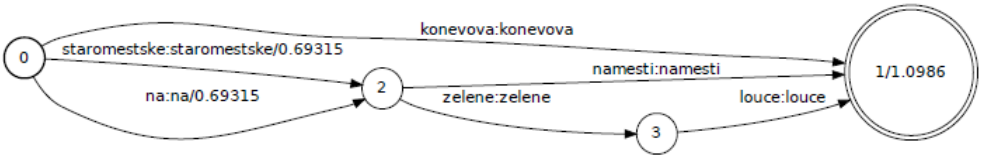
| String | Výslovnost |
|---|---|
| benátky | b e n aa t k i |
| nad | n a d |
| jizerou | j i z e r o u |
| frenštát | f r e n ss t aa t |
| pod | p o d |
| radhoštěm | r a d h o ss tt e m |

### 3.4.6 Lexikon a gramatika pro rozpoznávač ulic

Gramatika ulic je velice podobná gramatice měst. Je tu stejný problém výskytu více-slovných ulic. Znovu se braly v úvahu dva přístupy. První přístup je stejný jako u gramatiky měst, kde je název ulice brán jako jedno-slovný string (obrázek 15). A druhý přístup, každé slovo v názvu bráno jako samostatná proměná, se lišil od gramatiky měst pouze v nepoužíti přechodů 3-4 a 4-5. V databázi ulic se vyskytovaly ulice s maximálně čtyrmi slovy (obrázek 16). Pro oba přístupy jsou vytvořeny gramatiky jako zerogram i jako unigram LM s četnostmi získaných stejně jako u měst. Procento ulic vyskytujících se v testovacích promluvách a nevyskytujících se v četnostní databázi ulic je dost velké, proto i zde byly vytvořeny 3 druhy slovníků a 6 druhů gramatik. Ulice pro zařazení do slovníků měli stejné kritéria jako u měst.




**Obr. 15** Zjednodušený gramatický model ulic - nahrazení mezery




**Obr. 16** Zjednodušený gramatický model ulic - druhý přístup

Tvorba lexikonů pro oba přístupy byla složitější než u rozpoznávače měst. V názvu ulic se vyskytují i číslice nebo zkratky jmen. Pokud se v názvu ulice vyskytuje číslice (1. 2. ...atd), skript pro tvorbu výslovnosti ji přeskočí. Pokud je v názvu ulice obsažená tečka (zkratka) výslovnostní skript ji přeskočí nebo přiřadí fón "sp". Pro každou obsáhlou číslici ve slovníku je nutné přepsat její správnou výslovnost. "Sp" fón značí ticho a byl obsažen ve phones.txt pouze v jednom ze dvou použitých AM modelů, proto je nutné tento fón odstranit ze slovníků. U zkratek je vhodné jejich výslovnost upravit do celého názvu.

**Tab. 15** Lexicon.txt ulice - nahrazení mezer

| String | Výslovnost |
|---|---|
| 10.května | d e s aa t ee h o k v j e t n a |
| anglické_nábř. | a ng g l i c k ee n aa b rrr e zz ii |
| anežky_české | a n e ss k i cc e s k ee |
| antala_staška | a n t a l a s t a ss k a |
| bohuslava_martinů | b o h u s l a v a m a r tt i n uu |

**Tab. 16** Lexicon.txt města - druhý přístup

| String | Výslovnost |
|---|---|
| 1. | p r v nn ii h o |
| 1.května | p r v nn ii h o k v j e t n a |
| anglické | a ng g l i c k ee |
| nábř. | n aa b rrr e zz ii |
| ohrada | o h r a d a |

## 3.5 Tvorba HCLG grafu

Základem dekódování je vytvoření dekódovacího grafu HCLG. Vztah pro získání dekódovacího grafu je na následujícím řádku:

$$HCLG = \min(\det(H \circ \det(C \circ \det(L \circ G)))), \tag{32}$$

Graf se sestrojí pomocí kombinace OpenFst a KALDI nástrojů. Nástroj KALDI fsttablecompose je podobný fstcompose, který využívá OpenFst, ale výpočet je rychlejší. Další nástroj KALDI fstminimizeencoded zase používá několik příkazů OpenFst. A fstdeterminizestar je podobný fstdeterminize, ale odstraňuje *disambiguate* symboly jako součást determinizace [9].

V tomto kroku bychom již měli mít připravený výslovnostní slovník L a gramatiku rozpoznávačů G. Graf se sestrojí po jednotlivých krocích a pro každou

gramatiku rozpoznávače zvlášť. V prvním kroku se musí propojit G s L (rovnice 28, kapitola 2.3.5), aby se získal rozklad jednotlivých stringů v gramatice na jednotlivé fonémy, a použije se kompozice pro implementování kontextově nezávislé substituce (rovnice 29, kapitola 2.3.5). Dále se aplikuje nástroj determinizace a minimalizace z výše uvedeného postupu a sestrojí se CLG (rovnice 30, kapitola 2.3.5). Zde je uveden příklad vytvoření CLG pomocí KALDI:

```
fsttablecompose L_disambig.fst G.fst > G_1_compose.fst
fstdeterminizestar --use-log=true G_1_compose.fst > G_2_det.fst
fstminimizeencoded G_2_det.fst > LG.fst

fstcomposecontext   --context-size=3 \
                    --central-position=1 \
                    --read-disambig-syms=disambig_phones.list \
                    --write-disambig-syms=disambig_ilabels_3_1.list \
                    ilabels_3_1 < LG.fst > CLG_3_1.fst
```

Výstup CLG_3_1.fst mapuje integery vstupních symbolů CLG do trifónů.

V tento okamžik je sestrojen CLG graf a je potřeba vytvořit H graf. H značí automat s HMM strukturou a jeho vlastnosti znázorňují dobu trvání jednotlivých úseků promluv, HMM topologii a strom rozhodování. Nejdříve se vytvoří Ha.fst, což je to v podstatě H.fst, ale nemá žádné zacyklení (self-loop). Ha.fst obsahuje pouze trifóny z ilabels_3_1.list, které se vytvořily v předchozím kroku.

```
make-h-transducer   --disambig-syms-out=disambig.int \
                    --transition-scale=$tscale \
                    ilabels_3_1 $tree $model > Ha.fst
```

Pokud máme vytvořený CLG graf a H graf, tak se pomocí kompozice H a CLG sloučí a následně determinizuje. Dalším krokem se odstraní *disambiguate* symboly a provede se minimalizace. Tímto postupem se zajistí, že žádná jednotlivá část grafu nezničí vlastnost stochastického systému.

```
fsttablecompose Ha.fst CLG_3_1.fst > HCLGa_comp.fst
fstdeterminizestar --use-log=true HCLGa_comp.fst > HCLGa_det.fst
fstrmsymbols disambig.int HCLGa_det.fst > HCLGa_rmsymb.fst
fstrmepslocal HCLGa_rmsymb.fst > HCLGa_rmepslocal.fst
fstminimizeencoded HCLGa_rmepslocal.fst > HCLGa.fst
```

Máme vytvořený HCLGa.fst. Posledním krokem k vytvoření HCLG.fst je přidání zacyklení (self-loop) do finálního grafu. Toto zacyklení koresponduje se zacyklením stavů u HMM. Tím, že tento krok je samostatný, umožňuje vytvoření větších grafů.

```
add-self-loops  --self-loop-scale=$loopscale \
                --reorder=true \
                $model < HCLGa.fst > HCLG.fst
```

Parametry $loopscale a $tscale jsou voleny pro naše rozpoznávání 1. Při ladění byly dosaženy nejlepší úspěšnosti rozpoznávání pro tscale=1 a loopscale v rozmezí 0.9-1.

## 3.6 Dekódování

Dekódování je poslední fáze procesu rozpoznávání. Jeho cílem je pro daný model a promluvu vygenerovat co nejbližší sekvenci slov (stringů). Dekódování se provede pomocí nástroje KALDI *gmm-decode-faster*. Je zde snaha projít vytvořený graf Viterbi algoritmem, ale tím že vytvořené grafy jsou hodně velké, by tato operace byla moc pomalá.

```
gmm-decode-faster  --word-symbol-table=words.txt \
                   --beam=$beam \
                   --acoustic-scale=0.083333
                   $model HCLG.fst "$feats" ark,t:test.tra
```

Nástroj KALDI ve skutečnosti Viterbiho algoritmus nepoužívá, generuje "*lattices*", což je mřížka sestrojená na základě největších podobností v promluvách. Mřížka je následně převáhována různými akustickými váhami a následně se z mřížky vybere nejlepší cesta [9]. Zavádí se parametr beam, který snižuje počet možných cest. Beam ořeže všechny cesty s horší váhou než je jeho hodnota. Toto ale má vliv na úspěšnost rozpoznávání. U reálného rozpoznávání to znamená, čím větší je hodnota parametru beam, tím proces trvá déle, ale má větší úspěšnost. Cílem rozpoznávání by měla být malá časová náročnost s co největší úspěšností. Při volení hodnoty tohoto parametru tedy musíme udělat kompromis. Většinou je jeho hodnota v rozmezí 10-20. Dalším parametrem u nástroje pro dekódování je volení akustické osy (acoustic scale). Tento parametr ovlivňuje chování ořezávání cest. Pro naše rozpoznávání byla zvolena hodnota 0.083333.

Výstupem nástroje *gmm-decode-faster* je soubor test.tra, který obsahuje id promluvy a id rozpoznaných slov podle souboru words.txt (tabulka 17).

Pro tento soubor se následně vytvoří jeho téměř identická kopie s rozdílem, kde místo id rozpoznaných slov jsou přímo stringy rozpoznaných slov. V tomto procesu se také nahrazují znova znaky "_" mezerou u variant rozpoznávačů s nahrazením mezer. Název tohoto souboru je test.trans (tabulka 18).

**Tab. 17** Soubor test.tra

| id promluvy | id rozpoznaného stringu |
|---|---|
| fe79bv101021 | 9463 8765 |
| fe79bv103023 | 2932 4693 |
| fe79bo101031 | 60 |
| fe79bo102032 | 41 |
| fe81bo201051 | 9 |
| fe81bo202052 | 91 |

**Tab. 18** Soubor test.trans

| id promluvy | rozpoznaný string |
|---|---|
| fe79bv101021 | monika marková |
| fe79bv103023 | františek hájek |
| fe79bo101031 | litobratřice |
| fe79bo102032 | kamenný újezd |
| fe81bo201051 | boženy němcové |
| fe81bo202052 | staroměstské náměstí |

Posledním krokem dekódování je použití KALDI nástroje *compute-wer* pro vyhodnocení úspěšnosti rozpoznávání. Porovnávají se soubory test.text (id promluvy a přepis testovaných promluv) a právě test.trans.

# 4 Experimentální část

V této kapitole jsou zpracovány experimenty pro každý druh rozpoznávače pro dva AM modely a tři různé hodnoty parametru beam. Nejdříve jsou zde popsány kritéria pro vyhodnocování úspěšnosti rozpoznávání. Dále jsou zde zmíněny databáze promluv, ze kterých byly čerpané testovací promluvy, obecné nastavení rozpoznávačů s popisem AM a nakonec dosažené výsledky pro jednotlivé rozpoznávání.

## 4.1 Klasifikační kritéria

Vyhodnocení úspěšnosti rozpoznávání u našich experimentů se provádí pomocí veličiny WER (word error rate), která značí procentuální chybovost rozpoznávaných slov [1]. Vztah pro výpočet WER je uveden zde:

$$\mathrm{WER} = \frac{D + S\ + I}{N}, \tag{33}$$

kde D značí počet vynechaných slov, S jsou chybně rozpoznaná slova, I reprezentuje navíc rozpoznávaná slova a N celkový počet rozpoznávaných slov.

Jako pomocný ukazatel úspěšnosti rozpoznávání slov slouží veličina SER (sentence error rate). SER vyjadřuje chybovost rozpoznaných promluv v procentech.

$$\mathrm{SER} = \frac{S}{N}, \tag{34}$$

kde C značí počet chybně rozpoznaných promluv a N celkový počet promluv.

Pokud promluva bude obsahovat například jedno vlastní jméno pak WER=SER. Pokud ale bude obsahovat jméno i příjmení, pak N pro WER je dva a pro SER je rovné jedné.

## 4.2 Použité databáze

V tomto experimentu se používaly testovací promluvy z databází SPEECON a Temic. Pro obě databáze následuje jejich stručný popis. Detailněji jsou popsány v kapitole 3.2.

Databáze SPEECON obsahuje promluvy od 640 rozdílných řečníků z toho 50 jich je dětských. Na jednoho rozdílného řečníka připadá téměř 322 promluv, asi

půl hodiny čistého záznamu. Obsahem záznamů jsou číslice, jména a příjmení, názvy měst, ulic, foneticky bohaté věty a další. Promluvy jsou nahrané ze čtyř různých prostředí: domov, kancelář, automobil a veřejné místo [8]. Každá promluva byla nahrána čtyřmi různými mikrofony lišícími se typem a vzdáleností od řečníka. Vzorkovací kmitočet nahrávek je 16kHz s lineárním šestnácti-bitovým kvantováním.

Databáze Temic obsahuje promluvy pořízené v automobilu za různých podmínek, kde řečník seděl vždy na místě spolujezdce. Promluvy obsahují foneticky bohaté věty, jména, příjmení, jména_příjmení, číslovky, telefonní čísla, názvy měst, názvy ulic atd. Řečníci byli obou pohlaví různých věkových kategorií z různých částí ČR. Promluvy byly nahrány dvěma soustavy mikrofonů. Vzorkovací frekvence promluv byla 44,1kHz s lineárním šestnácti-bitovým kvantováním a následně byly převzorkovány na $f_s$=16kHz.

Testovací množiny z databází jsou v tabulkách u dosažených výsledků značeny zkratkami. Zkratky a obsah jednotlivých testovacích podmnožin je uveden níže. Podmnožiny jsou značeny číslem.

| | |
|---|---|
| **SP** | databáze SPEECON |
| **TE** | databáze Temic |
| **dig** | množina složená z podmnožin promluv z číslic |
| **nam** | množina složená z promluv s formou jméno a příjmení |
| **c** | množina složená z názvů měst |
| **s** | množina složená z názvů ulic |
| | |
| **SPdig1** | promluvy s izolovanými číslici |
| **SPdig2** | promluvy se sekvencí izolovaných číslic |
| **SPdig3** | promluvy složeny ze stringů z číslic |
| | |
| **TEdig1** | promluvy s deseti jednocifernými číslicemi |
| **TEdig2** | promluvy s jednou číslicí |
| **TEnam1** | promluvy s formou vlastní jméno |
| **TEnam2** | promluvy s pouze příjmením |
| **TEnam3** | obsahuje vše co je v TEnam, TEnam1 i TEnam2 |

V tabulkách u dosažených výsledků je pro každou testovací množinu uveden počet promluv v množině ($N_p$) a počet slov obsažených v promluvách v testovací množině ($N_s$).

## 4.3 Obecné nastavení rozpoznávače

Pro experimenty byly použity dva akustické modely. Jejich parametry a nastavení parametrizace jsou uvedeny níže:

- **Akustické modely**

  *AM1*

  model fonémů vychází ze 44 hlásek doplněných o pauzu "sil"
  trénovací množina pro model byla vybrána z databází SPEECON a Temic
  počet promluv v trénovací množině byl 65488 (zhruba 102 hodin záznamu)
  trifónový model obsahoval 1518 trifónů a 10035 gaussovek

  *AM2*

  model fonémů vychází ze 44 hlásek doplněných o pauzy "sil" a "sp"
  trénovací množina pro model byla vybrána z databáze SPEECON
  počet promluv v trénovací množině byl 53848 (zhruba 84 hodin záznamu)
  trifónový model obsahoval 1010 trifónů a 19253 gaussovek

- **Nastavení parametrů parametrizace:**

| | |
|---|---|
| parametrizace | MFCC_E, MFCC_0 |
| koeficient preemfáze | 0.97 |
| typ okna | Poveyho okno |
| délka okna | 25 |
| délka segmentu | 10 |
| liftrovací koeficient kepstra | 22 |
| počet pásem BF | 30 |
| počet kep. koeficientů | 12 |
| normalizace | CMVN |

## 4.4 Dosažené výsledky

V této části jsou zaznamenány výsledky experimentů pro každý rozpoznávač zvlášť. Experimenty byly provedeny pro dva použité AM modely a tři hodnoty parametru beam (5, 12, 20). Všechny grafy uvedeny níže jsou vytvořeny pro hodnotu beamu 12.

### 4.4.1 Rozpoznávač číslic

Rozpoznávání číslic proběhlo pro testovací množinu čísel z databáze SPEECON a tři její podmnožiny. A také pro testovací množinu z databáze Temic a její dvě podmnožiny.

Z tabulek 19 a 20 lze vidět 100% úspěšnost pro rozpoznávání množiny TEdig2 (jedna číslice), ale na malém vzorku. Pro větší vzorek stejného druhu promluv bude procento nejspíše větší. Vliv parametru beam na tento druh je nejvíce znatelný u podmnožiny SPdig3, kde dojde ke zlepšení až o 9.6%.

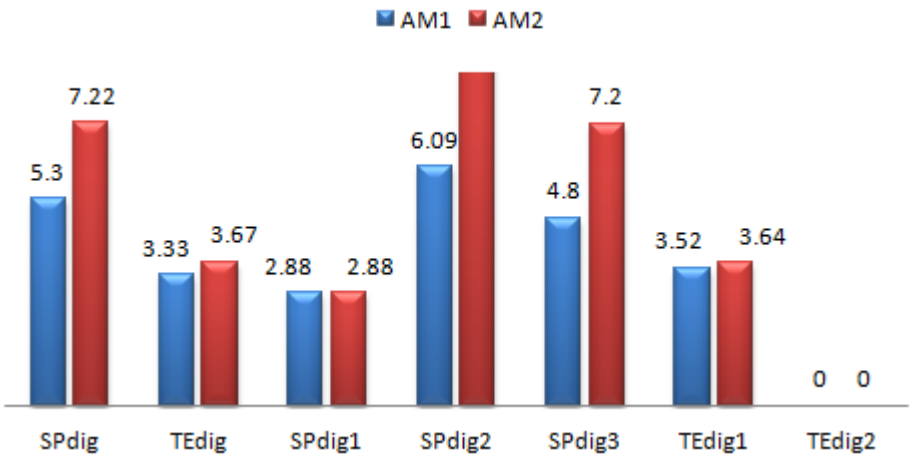
**Tab. 19** Hodnoty WER [%] pro AM1 - rozpoznávání číslic

| beam | SPdig | TEdig | SPdig1 | SPdig2 | SPdig3 | TEdig1 | TEdig2 |
|---|---|---|---|---|---|---|---|
| **5** | 7.87 | 4.56 | 3.85 | 9.39 | 11.2 | 4.77 | 0 |
| **12** | 5.3 | 3.33 | 2.88 | 6.09 | 4.8 | 3.52 | 0 |
| **20** | 4.98 | 3.22 | 2.88 | 6.09 | 4 | 3.41 | 0 |
| $\mathbf{N}_p$ | 178 | 108 | 104 | 49 | 25 | 88 | 20 |
| $\mathbf{N}_s$ | 623 | 900 | 104 | 394 | 125 | 880 | 20 |

**Tab. 20** Hodnoty WER [%] pro AM2 - rozpoznávání číslic

| beam | SPdig | TEdig | SPdig1 | SPdig2 | SPdig3 | TEdig1 | TEdig2 |
|---|---|---|---|---|---|---|---|
| **5** | 13.8 | 7 | 2.88 | 13.96 | 16.8 | 6.48 | 0 |
| **12** | 7.22 | 3.67 | 2.88 | 9.14 | 7.2 | 3.64 | 0 |
| **20** | 6.58 | 3.44 | 2.88 | 7.87 | 8 | 3.41 | 0 |
| $\mathbf{N}_p$ | 178 | 108 | 104 | 49 | 25 | 88 | 20 |
| $\mathbf{N}_s$ | 623 | 900 | 104 | 394 | 125 | 880 | 20 |

První AM od druhého má o trochu lepší výsledky, kdy na množině SPEECON má WER těsně pod 5% a na množině Temic 3.22% (obrázek 17). Podobných výsledků WER pro rozpoznávání číslic bylo dosaženo i v [1].




**Obr. 17** Porovnání hodnot WER [%] pro AM1 a AM2 u rozpoznávání číslic

### 4.4.2 Rozpoznávač jmen a příjmení

U rozpoznávání jmen a příjmení testovací promluvy z databáze SPEECON měly vždy formu jméno - příjmení. U promluv z databáze Temic bylo více variant (viz podkapitola 4.2) . Gramatický model (GM) byl vytvořen jako zerogram i unigram LM. Počet slov u TEnam není dvojnásobný počtu promluv, protože v jedné promluvě se vyskytlo jméno - příjmení - jméno - příjmení (tabulka 21).

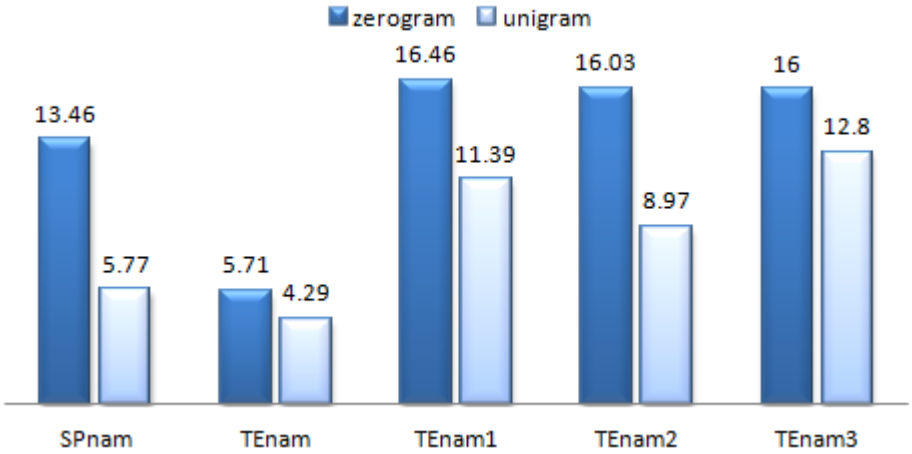
**Tab. 21** Hodnoty WER [%] pro AM1 - rozpoznávání jmen zerogram LM

| beam | SPnam | TEnam | TEnam1 | TEnam2 | TEnam3 |
|---|---|---|---|---|---|
| **5** | 21.15 | 22.14 | 16.46 | 25.64 | 57.33 |
| **12** | 13.46 | 5.71 | 16.46 | 16.03 | 16 |
| **20** | 13.46 | 6.43 | 16.46 | 16.03 | 16.89 |
| $\mathbf{N}_p$ | 26 | 69 | 79 | 156 | 304 |
| $\mathbf{N}_s$ | 52 | 140 | 79 | 156 | 375 |

**Tab. 22** Hodnoty WER [%] pro AM1 - rozpoznávání jmen unigram LM

| beam | SPnam | TEnam | TEnam1 | TEnam2 | TEnam3 |
|---|---|---|---|---|---|
| **5** | 11.54 | 10 | 17.72 | 16.67 | 50.13 |
| **12** | 5.77 | 4.29 | 11.39 | 8.97 | 12.8 |
| **20** | 5.77 | 5.71 | 10.13 | 16.03 | 13.07 |
| $\mathbf{N}_p$ | 26 | 69 | 79 | 156 | 304 |
| $\mathbf{N}_s$ | 52 | 140 | 79 | 156 | 375 |

Z tabulek 21 a 22 a grafu (obrázek 18) lze vidět menší chybovost rozpoznávání pro unigram LM . Pro tuto variantu jsou výsledky pro SPEECON množinu jen o 0.79% horší než u rozpoznávání číslic pro stejný AM. Při zvyšování beamu nad 10 docházelo už k výraznému poklesu WER (až o 37%). Je zde vidět, že pokud v testovacích promluvách je přesně jméno - příjmení, rozpoznávání vede k lepším výsledkům, než u množiny kde se mohou vyskytnout jména, příjmení nebo nejdříve jméno a následně příjmení.




**Obr. 18** Porovnání hodnot WER [%] zerogramu a unigramu LM pro AM1 u rozpoznávání jmen a příjmení

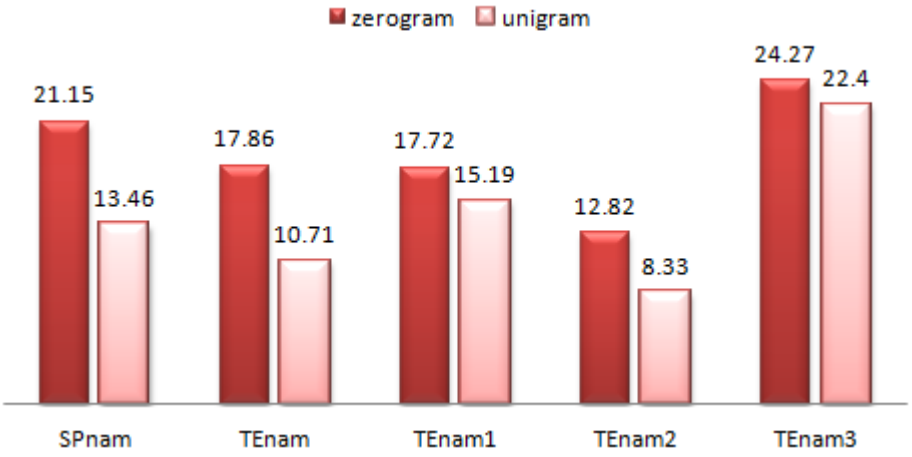
Tabulky 23 a 24 zobrazují výsledky pro druhý AM model. Porovnání mezi zerogramem a unigramem LM je podobné jako pro první AM (obrázek 19). Vliv zvyšování hodnoty beam při rozpoznávání je u tohoto AM modelu ještě znatelnější (zlepšení i skoro o 51% u TEnam3).

**Tab. 23** Hodnoty WER [%] pro AM2 - rozpoznávání jmen zerogram LM

| beam | SPnam | TEnam | TEnam1 | TEnam2 | TEnam3 |
|---|---|---|---|---|---|
| **5** | 28.85 | 54.29 | 27.85 | 35.26 | 73.87 |
| **12** | 21.15 | 17.86 | 17.72 | 12.82 | 24.27 |
| **20** | 19.23 | 16.43 | 12.66 | 12.18 | 24 |
| $\mathbf{N}_p$ | 26 | 69 | 79 | 156 | 304 |
| $\mathbf{N}_s$ | 52 | 140 | 79 | 156 | 375 |

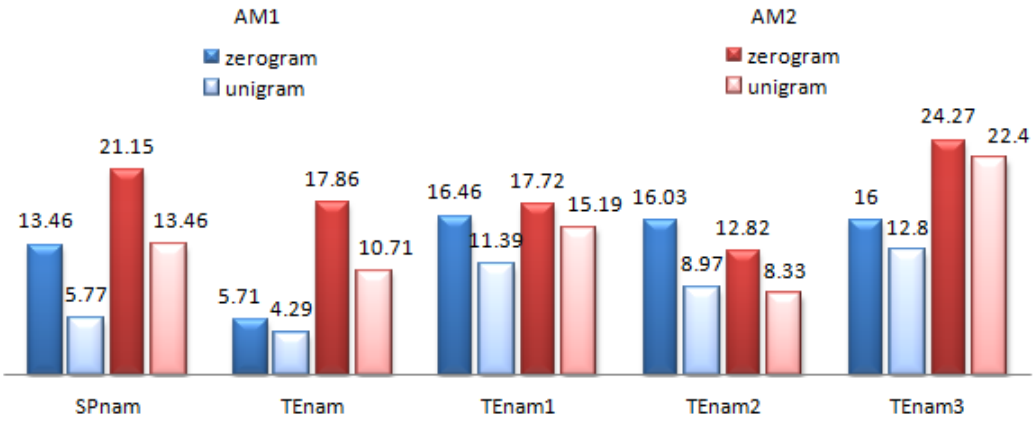
**Tab. 24** Hodnoty WER [%] pro AM2 - rozpoznávání jmen unigram LM

| beam | SPnam | TEnam | TEnam1 | TEnam2 | TEnam3 |
|---|---|---|---|---|---|
| **5** | 25 | 40.71 | 30.38 | 30.77 | 69.6 |
| **12** | 13.46 | 10.71 | 15.19 | 8.33 | 22.4 |
| **20** | 13.46 | 8.57 | 10.13 | 7.05 | 18.67 |
| $\mathbf{N}_p$ | 26 | 69 | 79 | 156 | 304 |
| $\mathbf{N}_s$ | 52 | 140 | 79 | 156 | 375 |




**Obr. 19** Porovnání hodnot WER [%] zerogramu a unigramu LM pro AM2 u rozpoznávání jmen a příjmení

Rozpoznávání pro AM2 dosahovalo horších výsledků než pro AM1. Pouze pro promluvy s jen vlastními jmény AM2 rozpoznával o trochu lépe. Nejlepšího výsledku u AM1 dosáhlo rozpoznávání jmen pro unigram LM, beam 12 a testovací množinu s formou jméno - příjmení: 5.71%. Na obrázku 20 jsou porovnány použité akustické modely pro rozpoznávání jmen a příjmení pro unigram a beam 12.




**Obr. 20** Porovnání hodnot WER [%] pro AM1 a AM2 u rozpoznávání jmen a příjmení

### 4.4.3 Rozpoznávač měst

Všechny testovací promluvy měli stejnou formu, obsahovaly pouze název města. Experimenty pro tento rozpoznávač proběhly pro dva přístupy GM. Pro první přístup byl název města brán jako jedno-slovný string (GMc1). U druhého přístupu bylo každé slovo rozpoznáváno samostatně (GMc2). Pro oba přístupy proběhly experimenty pro zerogram i unigram LM na dvou použitých AM. V tabulkách experimentů se objevuje parametr OOV (out of vocalbulary), které značí procento neobsažených měst (slov měst) ve slovníku, ale obsažených v promluvách určité množiny. OOV se liší pro každý typ vytvořeného slovníku u obou testovacích množin.

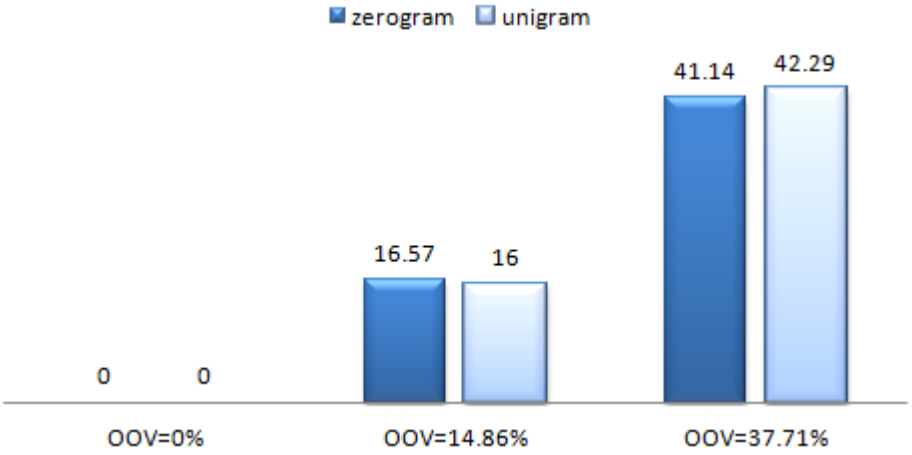
**Tab. 25** Hodnoty WER [%] pro AM1 - rozpoznávání měst GMc1 zerogram LM

| | *slovník 1* | | *slovník 2* | | *slovník 3* | |
|---|---|---|---|---|---|---|
| **beam** | **SPc** | **TEc** | **SPc** | **TEc** | **SPc** | **TEc** |
| **5** | 13.89 | 4.57 | 25 | 26.86 | 22.22 | 46.86 |
| **12** | 0 | 0 | 0 | 16.57 | 0 | 41.14 |
| **20** | 0 | 0 | 0 | 16 | 0 | 38.29 |
| $\mathbf{N}_p$ | 26 | 134 | 26 | 134 | 26 | 134 |
| $\mathbf{N}_s$ | 36 | 175 | 36 | 175 | 36 | 175 |
| **OOV** [%] | 0 | 0 | 0 | 14.86 | 0 | 37.71 |

Výsledné hodnoty experimentu pro GMc1 nastaveným na AM1 jsou zobrazeny v tabulkách 25 a 26. Z grafu 21 lze vidě stoupající chybovost rozpoznávání měst pro větší procento OOV na množině TEc. Zerogram i unigram dosahují při této variantě rozpoznávání podobné výsledky.

**Tab. 26** Hodnoty WER [%] pro AM1 - rozpoznávání měst GMc1 unigram LM

| | *slovník 1* | | *slovník 2* | | *slovník 3* | |
|---|---|---|---|---|---|---|
| **beam** | **SPc** | **TEc** | **SPc** | **TEc** | **SPc** | **TEc** |
| **5** | 13.89 | 21.14 | 19.44 | 25.71 | 27.78 | 51.43 |
| **12** | 2.78 | 0 | 0 | 16 | 0 | 42.29 |
| **20** | 0 | 0 | 0 | 15.43 | 0 | 38.29 |
| $\mathbf{N}_p$ | 26 | 134 | 26 | 134 | 26 | 134 |
| $\mathbf{N}_s$ | 36 | 175 | 36 | 175 | 36 | 175 |
| **OOV** [%] | 0 | 0 | 0 | 14.86 | 0 | 37.71 |




**Obr. 21** Závislost WER [%] na OOV pro TEc množinu pro GMc1 a AM1

V tabulkách 27 a 28 jsou zobrazeny výsledné chybovosti rozpoznávání pro druhý přístup. Porovnávat hodnoty mezi oběma GM, díky rozdílné hodnotě OOV by nebylo směrodatné. Pro stejné OOV dosahují velmi podobné hodnoty, akorát u množiny TEc zerogramu má druhý přístup chybovost větší o 1.14% pro beam 12 a 20.

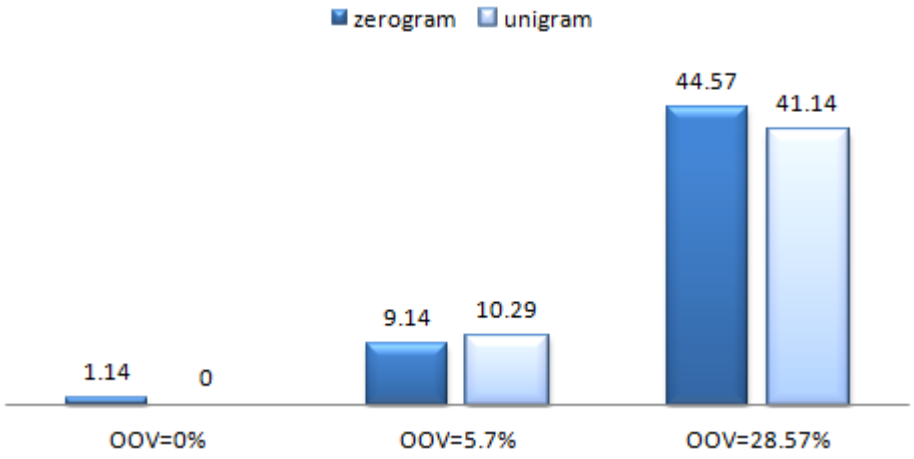
**Tab. 27** Hodnoty WER [%] pro AM1 - rozpoznávání měst GMc2 zerogram LM

| | *slovník 1* | | *slovník 2* | | *slovník 3* | |
|---|---|---|---|---|---|---|
| **beam** | **SPc** | **TEc** | **SPc** | **TEc** | **SPc** | **TEc** |
| **5** | 19.44 | 5.14 | 27.78 | 22.86 | 30.56 | 48.57 |
| **12** | 0 | 1.14 | 2.78 | 9.14 | 0 | 44.57 |
| **20** | 0 | 1.14 | 2.78 | 9.14 | 0 | 39.43 |
| $\mathbf{N}_p$ | 26 | 134 | 26 | 134 | 26 | 134 |
| $\mathbf{N}_s$ | 36 | 175 | 36 | 175 | 36 | 175 |
| **OOV** [%] | 0 | 0 | 0 | 5.7 | 0 | 28.57 |

**Tab. 28** Hodnoty WER [%] pro AM1 - rozpoznávání měst GMc2 unigram LM

| | *slovník 1* | | *slovník 2* | | *slovník 3* | |
|---|---|---|---|---|---|---|
| **beam** | **SPc** | **TEc** | **SPc** | **TEc** | **SPc** | **TEc** |
| **5** | 13.89 | 20 | 27.78 | 29.85 | 30.56 | 52.99 |
| **12** | 2.78 | 0 | 0 | 10.29 | 0 | 41.14 |
| **20** | 0 | 0 | 0 | 9.71 | 0 | 40 |
| $\mathbf{N}_p$ | 26 | 134 | 26 | 134 | 26 | 134 |
| $\mathbf{N}_s$ | 36 | 175 | 36 | 175 | 36 | 175 |
| **OOV** [%] | 0 | 0 | 0 | 5.7 | 0 | 28.57 |

Slovníky u obou přístupů obsahují stejný počet měst, ale u druhého GM jsou některé části měst součástí jiných chybějících měst ve slovníku. Proto OOV u GMc2 je menší. Na obrázku 22 je zobrazena závislost chybovosti WER na procentu OOV pro GMc2.




**Obr. 22** Závislost WER [%] na OOV pro TEc množinu pro GMc2 a AM1

**Tab. 29** Hodnoty WER [%] pro AM2 - rozpoznávání měst GMc1 zerogram LM

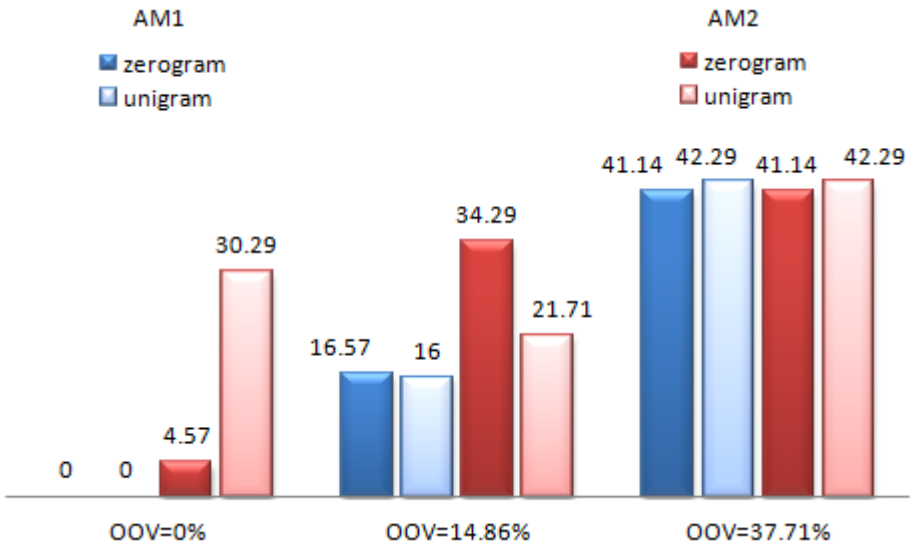
| | *slovník 1* | | *slovník 2* | | *slovník 3* | |
|---|---|---|---|---|---|---|
| **beam** | **SPc** | **TEc** | **SPc** | **TEc** | **SPc** | **TEc** |
| **5** | 22.22 | 22.29 | 69.44 | 66.29 | 36.11 | 60.57 |
| **12** | 5.56 | 4.57 | 41.67 | 34.29 | 5.56 | 41.14 |
| **20** | 0 | 0 | 13.89 | 18.86 | 0 | 39.43 |
| $\mathbf{N}_p$ | 26 | 134 | 26 | 134 | 26 | 134 |
| $\mathbf{N}_s$ | 36 | 175 | 36 | 175 | 36 | 175 |
| **OOV** [%] | 0 | 0 | 0 | 14.86 | 0 | 37.71 |

U AM2 pro gramatiku GMc1 je vidět značný vliv na nastavené hodnotě beam (tabulky 29 a 30). Rozdíl mezi nastaveným beamem 12 a 20 pro nulové OOV

**Tab. 30** Hodnoty WER [%] pro AM2 - rozpoznávání měst GMc1 unigram LM

| | *slovník 1* | | *slovník 2* | | *slovník 3* | |
|---|---|---|---|---|---|---|
| **beam** | **SPc** | **TEc** | **SPc** | **TEc** | **SPc** | **TEc** |
| **5** | 52.78 | 55.43 | 30.56 | 48.57 | 30.56 | 65.71 |
| **12** | 19.44 | 30.29 | 5.56 | 21.71 | 5.56 | 42.29 |
| **20** | 7.69 | 1.71 | 0 | 15.43 | 0 | 40 |
| $\mathbf{N}_p$ | 26 | 134 | 26 | 134 | 26 | 134 |
| $\mathbf{N}_s$ | 36 | 175 | 36 | 175 | 36 | 175 |
| **OOV** [%] | 0 | 0 | 0 | 14.86 | 0 | 37.71 |

je obrovský (28.58%). Hodnoty WER pro tento model jsou o něco slabší než pro AM1, hlavně u nulového OOV. Rozdíly WER hodnot modelů jsou vidět na obrázku 23, kde jsou zobrazeny zerogramy a unigramy GMc1 pro oba modely současně. Hodnota WER unigramu u AM2 pro slovník s nulovým OOV je tak vysoká nejspíše pro horší rozdělení pravděpodobnosti u názvů měst nevyskytujících se v čerpané databázi, kde jim byla přiřazena průměrná hodnota z ostatních názvů měst vyskytujících se ve slovníku s OOV=0%.




**Obr. 23** Závislost WER [%] na OOV pro TEc množinu pro GMc1 a oba AM

V tabulkách 31 a 32 jsou ukázány hodnoty WER pro GMc2 pro AM2. Rozdílnosti hodnot pro typy AM modelů u GMc2 byly podobné jako u GMc1. Dále graf 24 zobrazuje zerogramy i unigramy pro GMc2 pro oba AM. Je zajímavé, že unigramy mají na slovnících s menším OOV horší WER než zerogramy. Může to být dáno testovací množinou pro Temic, kde se vyskytovalo hodně měst s mnší pravděpodobností výskytu.

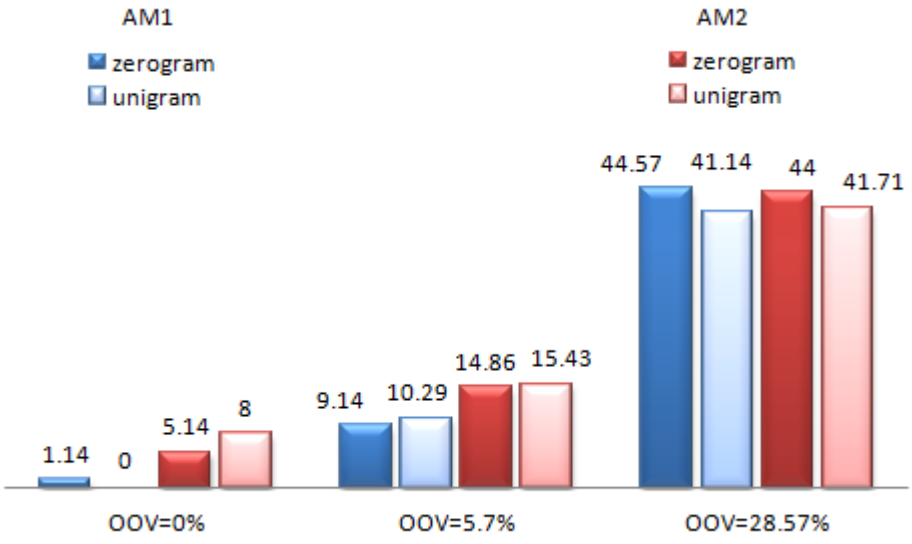
Pro celkové shrnutí úlohy rozpoznávání měst bylo dosaženo 100% úspěšnosti téměř u všech experimentů se slovníky s nulovým OOV. Nutno dodat, že experimenty pro tuto úlohu byly provedeny na velmi malém vzorku dat (160 promluv,

**Tab. 31** Hodnoty WER [%] pro AM2 - rozpoznávání měst GMc2 zerogram LM

| | *slovník 1* | | *slovník 2* | | *slovník 3* | |
|---|---|---|---|---|---|---|
| **beam** | **SPc** | **TEc** | **SPc** | **TEc** | **SPc** | **TEc** |
| **5** | 25 | 22.29 | 50 | 44 | 44.44 | 59.43 |
| **12** | 0 | 5.14 | 8.33 | 14.86 | 5.56 | 44 |
| **20** | 0 | 0 | 2.78 | 12.57 | 0 | 40.57 |
| $\mathbf{N}_p$ | 26 | 134 | 26 | 134 | 26 | 134 |
| $\mathbf{N}_s$ | 36 | 175 | 36 | 175 | 36 | 175 |
| **OOV** [%] | 0 | 0 | 0 | 5.7 | 0 | 28.57 |

**Tab. 32** Hodnoty WER [%] pro AM2 - rozpoznávání měst GMc2 unigram LM

| | *slovník 1* | | *slovník 2* | | *slovník 3* | |
|---|---|---|---|---|---|---|
| **beam** | **SPc** | **TEc** | **SPc** | **TEc** | **SPc** | **TEc** |
| **5** | 25 | 31.43 | 33.33 | 44 | 30.56 | 64.57 |
| **12** | 3.85 | 8 | 3.85 | 15.43 | 7.69 | 41.71 |
| **20** | 0 | 0.37 | 3.85 | 11.43 | 3.85 | 41.71 |
| $\mathbf{N}_p$ | 26 | 134 | 26 | 134 | 26 | 134 |
| $\mathbf{N}_s$ | 36 | 175 | 36 | 175 | 36 | 175 |
| **OOV** [%] | 0 | 0 | 0 | 5.7 | 0 | 28.57 |




**Obr. 24** Závislost WER [%] na OOV pro TEc množinu pro GMc2 a oba AM

211 slov). Na mnohem větším vzorku by se výsledek experimentu mohl pohybovat pod 5% hodnoty WER. GMc2 se prokázal jako možná varianta přístupu u tohoto rozpoznávání a nejspíše je zde větší prostor pro zlepšení chybovosti než u GMc1.

### 4.4.4 Rozpoznávač ulic

Rozpoznávání ulic je velmi podobné tomu předchozímu. Vyskytují se zde také dva gramatické modely. Jeden pracuje s názvem ulice jako jedním celkem (GMs1), druhý znovu bere každé jednotlivé slovo názvů ulic zvlášť (GMs2). I u tohoto typu rozpoznávání se vyskytuje více typů slovníků s jinými procenty OOV. Hodnota OOV je zobrazena v tabulkách experimentu pro každou testovací množinu a je ještě větší než u předchozí varianty. U grafů je zde vždy znázorněna testovací podmnožina a hodnota OOV.

**Tab. 33** Hodnoty WER [%] pro AM1 - rozpoznávání ulic GMc1 zerogram LM

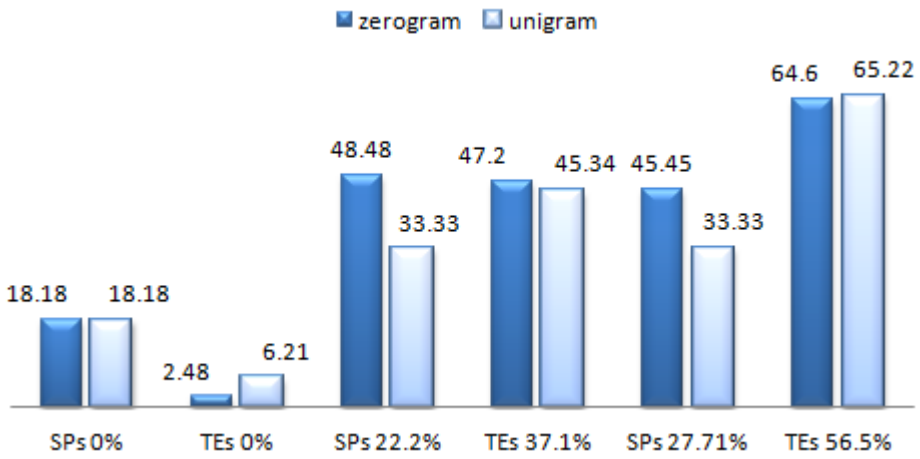
| | *slovník 1* | | *slovník 2* | | *slovník 3* | |
|---|---|---|---|---|---|---|
| **beam** | **SPs** | **TEs** | **SPs** | **TEs** | **SPs** | **TEs** |
| **5** | 39.39 | 22.36 | 81.82 | 59.63 | 78.79 | 72.67 |
| **12** | 18.18 | 2.48 | 48.48 | 47.2 | 45.45 | 64.6 |
| **20** | 18.18 | 1.24 | 42.42 | 38.51 | 36.36 | 63.35 |
| $\mathbf{N}_p$ | 26 | 107 | 26 | 107 | 26 | 107 |
| $\mathbf{N}_s$ | 36 | 175 | 36 | 175 | 36 | 175 |
| **OOV** [%] | 0 | 0 | 22.2 | 37.1 | 27.78 | 56.5 |

**Tab. 34** Hodnoty WER [%] pro AM1 - rozpoznávání ulic GMc1 unigram LM

| | *slovník 1* | | *slovník 2* | | *slovník 3* | |
|---|---|---|---|---|---|---|
| **beam** | **SPs** | **TEs** | **SPs** | **TEs** | **SPs** | **TEs** |
| **5** | 36.36 | 28.57 | 78.79 | 62.73 | 78.89 | 72.66 |
| **12** | 18.18 | 6.21 | 33.33 | 45.34 | 33.33 | 65.22 |
| **20** | 18.18 | 1.24 | 30.33 | 37.27 | 30.33 | 63.35 |
| $\mathbf{N}_p$ | 26 | 107 | 26 | 107 | 26 | 107 |
| $\mathbf{N}_s$ | 36 | 175 | 36 | 175 | 36 | 175 |
| **OOV** [%] | 0 | 0 | 22.2 | 37.1 | 27.78 | 56.5 |

V tabulkách 33 a 34 jsou uvedeny hodnoty WER pro AM1 s gramatikou GMs1. Chybovost u této varianty s porovnáním chybovosti u rozpoznávání měst je jasně vyšší. Na množině SPEECON, která má stejný vzorek promluv jako u předchozí varianty rozpoznávání, je o 18.18% horší. U promluv z Temicu je pro nulové OOV chybovost pořád hodně slušná. Opět pro dělání jasných závěrů je toto hodně malý vzorek promluv a výsledky jsou spíše orientační.

Porovnání zerogramu s unigramem je na AM1 je znázorněno pomocí grafu níže (obrázek 25). Z grafu lze vidět velmi podobné chybovosti pro obě varianty. Více se zerogram od unigramu liší akorát pro speecon množiny s větším OOV ve prospěch unigramu.




**Obr. 25** Závislost WER [%] na OOV pro GMs1 na AM1

**Tab. 35** Hodnoty WER [%] pro AM1 - rozpoznávání ulic GMs2 zerogram LM

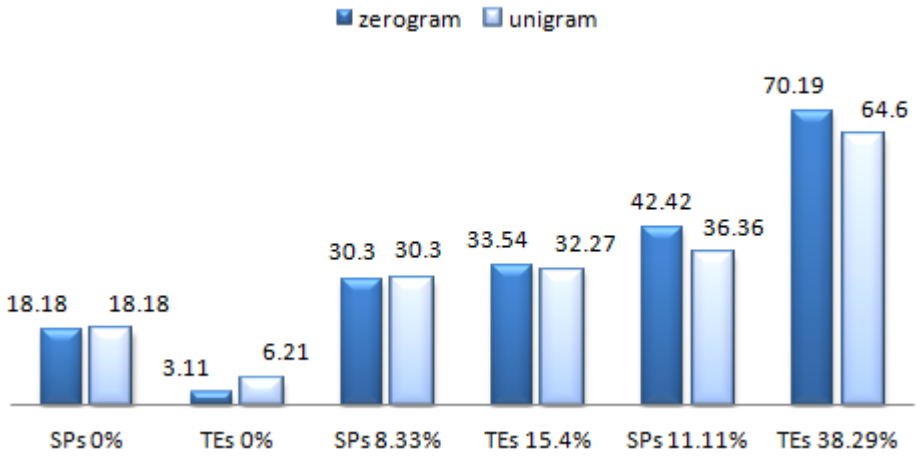
| | *slovník 1* | | *slovník 2* | | *slovník 3* | |
|---|---|---|---|---|---|---|
| **beam** | **SPs** | **TEs** | **SPs** | **TEs** | **SPs** | **TEs** |
| **5** | 57.58 | 21.12 | 78.79 | 62.11 | 93.94 | 76.4 |
| **12** | 18.18 | 3.11 | 30.3 | 33.54 | 42.42 | 70.19 |
| **20** | 12.12 | 1.86 | 27.27 | 31.06 | 33.33 | 60.87 |
| $\mathbf{N}_p$ | 26 | 107 | 26 | 107 | 26 | 107 |
| $\mathbf{N}_s$ | 36 | 175 | 36 | 175 | 36 | 175 |
| **OOV** [%] | 0 | 0 | 8.33 | 15.4 | 11.11 | 38.29 |

**Tab. 36** Hodnoty WER [%] pro AM1 - rozpoznávání ulic GMs2 unigram LM

| | *slovník 1* | | *slovník 2* | | *slovník 3* | |
|---|---|---|---|---|---|---|
| **beam** | **SPs** | **TEs** | **SPs** | **TEs** | **SPs** | **TEs** |
| **5** | 39.39 | 29.19 | 87.88 | 62.73 | 87.88 | 75.78 |
| **12** | 18.18 | 6.21 | 30.3 | 32.27 | 36.36 | 64.6 |
| **20** | 12.12 | 1.24 | 18.18 | 27.95 | 33.33 | 58.39 |
| $\mathbf{N}_p$ | 26 | 107 | 26 | 107 | 26 | 107 |
| $\mathbf{N}_s$ | 36 | 175 | 36 | 175 | 36 | 175 |
| **OOV** [%] | 0 | 0 | 8.33 | 15.4 | 11.11 | 38.29 |

Tabulky 35 a 36 uvádějí hodnoty chybovosti WER pro druhý přístup GMs2. Tato gramatika má oproti GMs1 trochu nižší hodnoty WER pro nulové OOV. Hodnotit rozdíly pro jiné hodnoty OOV znovu není směrodatné.

Na grafu 26 je zobrazena závislost WER na větším OOV pro tuto gramatiku. Rozpoznávání pro zerogram od unigramu vypadá o maličko lépe.




**Obr. 26** Závislost WER [%] na OOV pro GMs2 na AM1

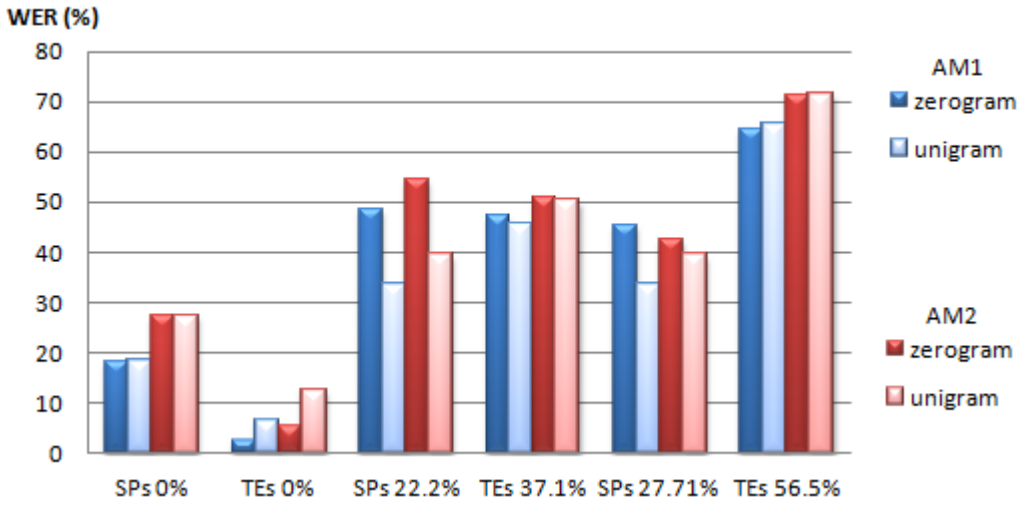
**Tab. 37** Hodnoty WER [%] pro AM2 - rozpoznávání ulic GMs1 zerogram LM

| | *slovník 1* | | *slovník 2* | | *slovník 3* | |
|---|---|---|---|---|---|---|
| **beam** | **SPs** | **TEs** | **SPs** | **TEs** | **SPs** | **TEs** |
| **5** | 33.33 | 34.78 | 72.73 | 73.91 | 81.82 | 80.12 |
| **12** | 27.27 | 5.59 | 54.55 | 50.93 | 42.42 | 71.43 |
| **20** | 18.18 | 2.48 | 36.36 | 43.48 | 36.36 | 66.46 |
| $\mathbf{N}_p$ | 26 | 107 | 26 | 107 | 26 | 107 |
| $\mathbf{N}_s$ | 36 | 175 | 36 | 175 | 36 | 175 |
| **OOV** [%] | 0 | 0 | 22.2 | 37.1 | 27.78 | 56.5 |

**Tab. 38** Hodnoty WER [%] pro AM2 - rozpoznávání ulic GMs1 unigram LM

| | *slovník 1* | | *slovník 2* | | *slovník 3* | |
|---|---|---|---|---|---|---|
| **beam** | **SPs** | **TEs** | **SPs** | **TEs** | **SPs** | **TEs** |
| **5** | 36.36 | 42.24 | 60.61 | 70.81 | 60.61 | 78.26 |
| **12** | 27.27 | 12.42 | 39.39 | 50.31 | 39.39 | 71.43 |
| **20** | 18.18 | 3.11 | 33.33 | 40.99 | 33.33 | 67.08 |
| $\mathbf{N}_p$ | 26 | 107 | 26 | 107 | 26 | 107 |
| $\mathbf{N}_s$ | 36 | 175 | 36 | 175 | 36 | 175 |
| **OOV** [%] | 0 | 0 | 22.2 | 37.1 | 27.78 | 56.5 |

Hodnoty WER pro AM2 a gramatiku GMs1 jsou uvedeny v tabulkách 37 a 38. Z grafu 27 lze určit lepší průběh pro unigramy na SPEECON testovacích množinách pro oba AM. Pro Temic množiny jsou unigramy mírně horší. AM2 má i u této varianty rozpoznávání horší výsledky.




**Obr. 27** Závislost WER [%] na OOV pro oba AM s gramatikou GMs1

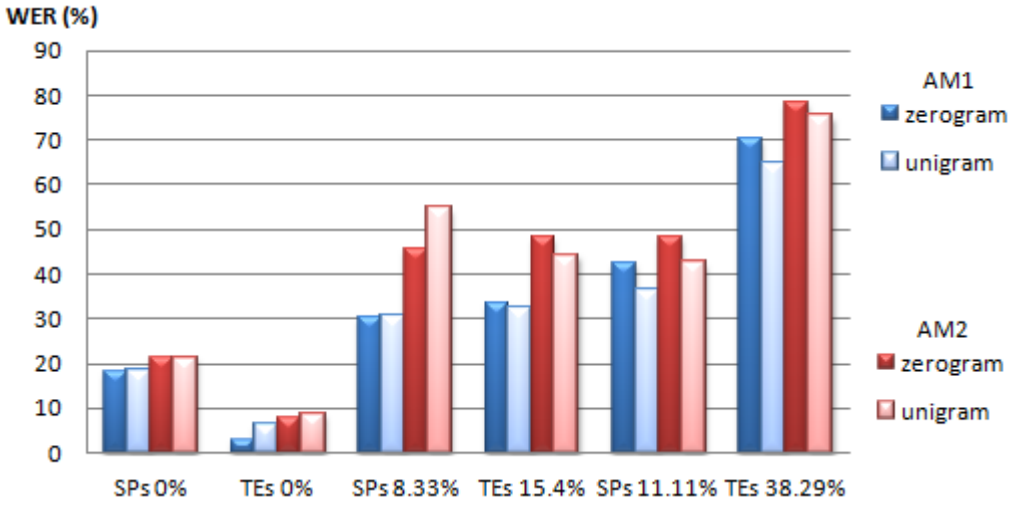
**Tab. 39** Hodnoty WER [%] pro AM2 - rozpoznávání ulic GMs2 zerogram LM

| | *slovník 1* | | *slovník 2* | | *slovník 3* | |
|---|---|---|---|---|---|---|
| **beam** | **SPs** | **TEs** | **SPs** | **TEs** | **SPs** | **TEs** |
| **5** | 57.58 | 45.96 | 69.7 | 79.5 | 72.73 | 88.82 |
| **12** | 21.21 | 8.07 | 45.45 | 48.45 | 48.48 | 78.26 |
| **20** | 18.18 | 1.86 | 21.21 | 32.92 | 30.3 | 65.22 |
| $\mathbf{N}_p$ | 26 | 107 | 26 | 107 | 26 | 107 |
| $\mathbf{N}_s$ | 36 | 175 | 36 | 175 | 36 | 175 |
| **OOV** [%] | 0 | 0 | 8.33 | 15.4 | 11.11 | 38.29 |

**Tab. 40** Hodnoty WER [%] pro AM2 - rozpoznávání ulic GMs2 unigram LM

| | *slovník 1* | | *slovník 2* | | *slovník 3* | |
|---|---|---|---|---|---|---|
| **beam** | **SPs** | **TEs** | **SPs** | **TEs** | **SPs** | **TEs** |
| **5** | 45.45 | 47.2 | 75.76 | 80.12 | 81.82 | 85.71 |
| **12** | 21.21 | 8.7 | 54.55 | 44.1 | 42.42 | 75.16 |
| **20** | 18.18 | 1.24 | 21.21 | 29.81 | 33.33 | 62.11 |
| $\mathbf{N}_p$ | 26 | 107 | 26 | 107 | 26 | 107 |
| $\mathbf{N}_s$ | 36 | 175 | 36 | 175 | 36 | 175 |
| **OOV** [%] | 0 | 0 | 8.33 | 15.4 | 11.11 | 38.29 |

Pro druhou variantu gramatiky GMs2 na AM2 jsou výsledky rozpoznávání uvedeny v tabulkách 39 a 40. Rozdíly hodnot jsou lépe patrné z grafu 28, ze kterého lze vidět velké chybovosti u slovníků s větším procentem OOV než 0%. Rozdílné gramatiky pro AM2 dosahovali znovu podobných výsledků pro nulové OOV.




**Obr. 28** Závislost WER [%] na OOV pro oba AM s gramatikou GMs2

Tato úloha rozpoznávání se ukázala jako nejvíce problematická v porovnání s ostatními úlohami rozpoznávání v této práci v experimentální části.

V experimentální části se prokázala důležitost velkého obsahu dat pro trénovací množiny, kde u AM1 byla téměř ve všech rozpoznávacích variantách prokázána menší chybovost rozpoznávání. Jako nejvíce úspěšná úloha rozpoznávání se jeví úloha pro rozpoznávání měst, kde pro AM1 pro obě gramatiky s jak zerogramy tak unigramy LM chybovost nepřekročila 3%. Tato hodnota byla, ale prokázána na velmi malém vzorku dat. U rozpoznávání jmen a příjmení dosahoval unigramový model značně lepších výsledků než ten zerogramový a jasně potvrzuje důležitost rozdílného pravděpodobnostního rozdělení.

# 5 Závěr

Cílem práce bylo systematicky zpracovat metodiku tvorby rozpoznávače řeči s malým slovníkem s využitím gramatiky na bázi unigramu jazykových modelů. Byla navržena realizace rozpoznávače pomocí nástrojového balíčku KALDI pro ovládání funkcí a zařízení v automobilu, zejména hlasové ovládání navigace. Konkrétně byly vytvořeny systémy realizující tyto úlohy: rozpoznávání číslic, rozpoznávání jmen a příjmení, rozpoznávání měst a rozpoznávání ulic.

V experimentální části se pro práci využívaly dvě řečové databáze SPEECON a Temic, ze kterých byly vybrány testovací množiny promluv pro testování úspěšnosti jednotlivých úloh. Z těchto databází byly také vybrány trénovací množiny pro použité akustické modely. Po určení testovacích množin z daných databází se množiny z parametrizovaly. Pro parametrizaci dat byla zvolena parametrizace MFCC s CMVN normalizací a pro její realizaci byly využity pomocné nástroje KALDI, díky kterým byly získány potřebné řečové příznaky.

Následně byly vytvořeny slovníky a gramatické modely na bázi zerogramu a unigramu LM pro každou jednotlivou úlohu zvlášť. U úlohy rozpoznávání jmen a příjmení byly vytvořeny gramatiky pro rozpoznávání vlastního jména, příjmení a jména - příjmení. Byla zde snaha o lepší úspěšnost rozpoznávání, pokud by promluvy měly jinou formu než jméno - příjmení. Pro rozpoznávání měst a ulic byly vyzkoušeny dva přístupy řešení gramatik. Problematikou této úlohy byl možný výskyt více slov v názvu města/ulice. První přístup bral každý název města/ulice jako jedno-slovnou proměnou. Druhým možným řešením bylo rozpoznávání každého slova v názvu města/ulice samostatně s jednoznačnými pravidly.

Dalším krokem bylo vytvoření dekódovacího grafu (HCLG) pro každý vytvořený gramatický model pomocí balíčku KALDI. Graf v sobě zahrnoval sestavení spojitosti mezi kontextově závislými slovy do kontextově nezávislých fonémů. Posledním krokem bylo samotné dekódování u něhož byly využity řečové příznaky získané parametrizací dat a právě dekódovací graf. Samotné určení chybovosti rozpoznávání bylo uděláno pomocí WER (word error rate), který porovnával přepsané testovací promluvy s výslednými produkty rozpoznávače.

Co se týče konkrétních výsledků nejvíce úspěšná úloha rozpoznávání byla úloha pro rozpoznávání měst s nulovým OOV (out of vocabulary), kde pro AM1 pro obě gramatiky s jak zerogramy tak unigramy LM chybovost nepřekročila hodnotu 3%. Nicméně toto rozpoznávání proběhlo na velmi malém vzorku dat (170 promluv, 211 slov) a reálná hodnota chybovosti pro tuto úlohu může být vyšší. Nejvíce problematické rozpoznávání bylo naopak u úlohy rozpoznávání ulic na množině

dat z databáze SPEECON, které mělo chybovost 18.18% pro nulové OOV také na velmi malém vzorku promluv. Rozpoznávání jmen a příjmení prokázalo důležitost rozdílného pravděpodobnostního rozdělení u GM. Unigramy LM pro tuto úlohu měly jednoznačně nižší chybovost než zerogramy.

Pro práci byly vytvořeny skripty s použitím nástrojů KALDI, které po jednoduché úpravě mohou sloužit dále pro podobné experimenty rozpoznávání řeči. Tyto skripty jsou součástí CD pro tuto práci.

# Bibliografie


[1] BĚHUNEK, M.: *Rozpoznávání řeči při různé kvalitě vstupního signálu.* Praha, Diplomová práce, ČVUT, 2010.

[2] DIXON, P. - FURUI, S.: *Introduction to the use of WFSTs in Speech and Language processing.* APSIPA Conference, 2009.

[3] HUANG, X. - ACERO, A. - HON, W.: *Spoken language processing.* Prentice Hall, 2001.

[4] LJOLJE, A.: *The importance of cepstral parameter correlations in speech recognition.* Computer Speech and Language, vol.8, 1994.

[5] MÁDR, V.: *Normalizační techniky pro robustní rozpoznávání řeči.* Praha, Diplomová práce, ČVUT, 2014.

[6] MOHRI, M. - PEREIRA, F. - RILEY, M.: *Speech recognition with weighted finite-state transducers.* Computer Speech & Language, vol. 16, 2002.

[7] MOHRI, M. - PEREIRA, F. - RILEY, M.: *The design principles of weighted finite-state transducer library.* Theoretical Computer Science, 2000.

[8] POLLÁK, P. - ČERNOCKÝ, J.: *Czech Speecon Adult Database.* ČVUT, Praha, 2004.

[9] POVEY, D.: *Speech recognition with Kaldi lectures.* [cit. 2015-12-29], http://danielpovey.com/kaldi-lectures.html.

[10] POVEY, D. et al.: *The Kaldi Speech Recognition Toolkit.* In IEEE 2011 Workshop on Automatic Speech Recognition and Understanding, Big Island, Hawai, US, 2011.

[11] PSUTKA, J. a kol.: *Mluvíme s počítačem česky.* Praha: Academia, 2006.

[12] PSUTKA, J.: *Techniky parametrizace, dekorelace a redukce dimenze příznaků v systémech rozpoznávání řeči.* Plzeň, Disertační práce, Západočeská univerzita v Plzni, 2007.

[13] RAJNOHA, J.: *Rozpoznávání řeči v reálných podmínkách na platformě standardního PC.* Praha, Diplomová práce, ČVUT, 2006.

[14] SMÉKAL, Z. a kol.: *Soubor programů pro práci se skrýtými Markovými modely (HTK).* Elektro revue, VUT, Brno, 2009.

[15] STRAND, M. - EGEBERG, A.: *Cepstral mean and variance normalization in the model domain.* COST278 and ISCA Tutorial and Research Workshop on Robustness Issues in Conversational Interaction, University of East Anglia, Norwich, UK, 2004.

[16] ŠTEMBERK, P.: *Implementace Rozpoznávačů řeči do multimediálních struktur.* ČVUT, Praha, 2005.

[17] TATARINOV, J.: *Detektory řečové aktivity na bázi HMM.* Praha, Disertační práce, ČVUT, 2010.

[18] UHLÍŘ, J. a kol.: *Technologie hlasových komunikací.* Praha, Nakladatelství ČVUT, 2007.

# Příloha A

# Obsah přiloženého CD

Součástí této práce je i přiložené CD. V následujících bodech je uveden obsah adresářů na tomto CD.

- Diplomova_prace_pdf - obsahuje soubor s textovou podobou diplomové práce Forman.pdf

- Diplomova_prace_skripty - obsahuje hlavní skript run.sh a podadresáře /conf a /local, které obsahují další skripty a podpůrné soubory k chodu hlavního skriptu.