Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra kybernetiky

Bakalářská práce

Plzeň, 2015 Ondřej Duspiva

PROHLÁŠENÍ

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

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

V Plzni dne ……. …….

Poděkování

Rád bych poděkoval panu Doc. Ing. Luďkovi Müllerovi, Ph.D. a panu Ing. Aleši Pražákovi, Ph.D. za odbornou pomoc a rady při zpracování této bakalářské práce.

Dále bych rád poděkoval MUDr. Anně Pondělíkové, MUDr. Kristýně Götzové a Gabriele Menclové, za odborné konzultace a pomoc při přípravě korpusů pro jazykový model.

Abstrakt

Tato práce se věnuje tvorbě jazykového modelu pro rozpoznávání řeči. V první části jsou popsány obecné postupy při řešení této problematiky pomocí n-gramových modelů a neuronových sítí. Popsány jsou metody tvorby a vyhlazování jazykového modelu a taktéž způsob posuzovaní kvality takového modelu.

Druhá část je věnována stručnému úvodu do problematiky gramatiky a jazyka.

Ve třetí části je pak popsán postup, při tvorbě jazykového modelu pro malé množství testovacích dat.

Závěrečná část je věnována návrhu vlastní metody pro velmi malé množství trénovacích dat, popřípadě jejich úplnou absenci. V této části jsou také popsány experimenty provedené na vytvořených jazykových modelech a zhodnoceny výsledky

Klíčová slova: jazykový model, n-gram, neuronová síť, perplexita, gramatika, jazykové rozpoznávání, vyhlazování

Abstract

This thesis is aimed to creating language model for speech recognition. The first part describes general procedurs of solution of this issue using n-gram models and neural networks. There is description of creating and smoothing language model as well as way of recognition of quality of that model.

Second part is dedicated to a brief introduction on grammar and language.

In third part is described the procedure for creating low resources language model for speech recognition

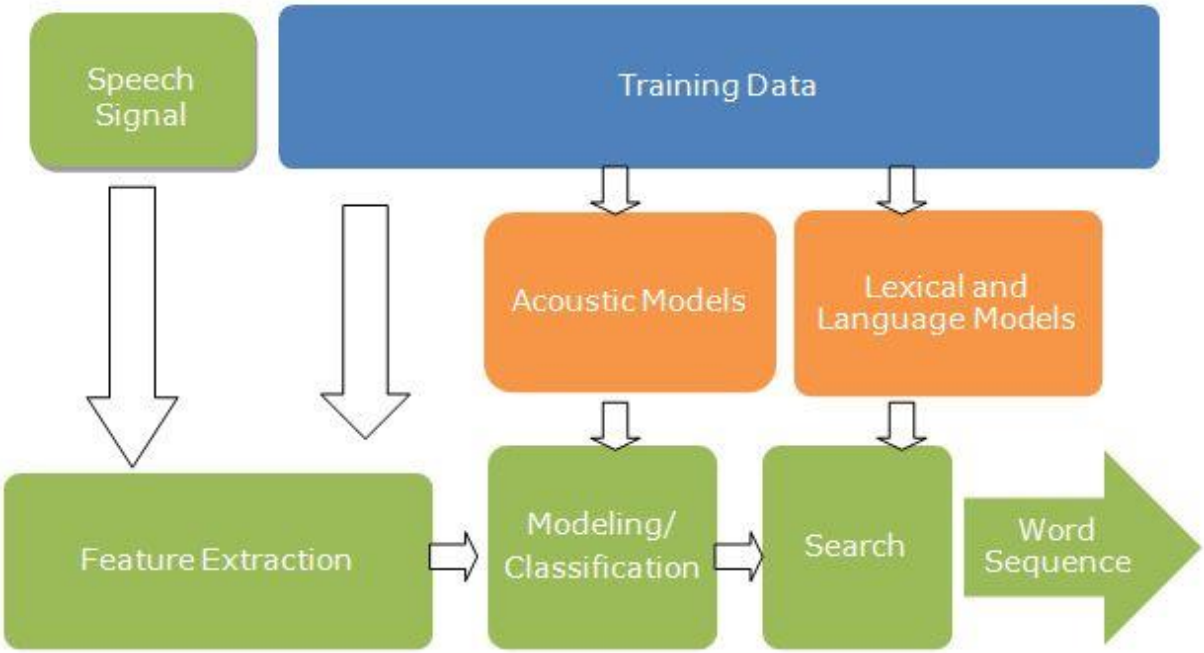
The final part is devoted to design of own method for very small size of training data or their total absence. In this section are also described experiments conducted on linguistic model developed with own method and also evaluated results.

Keywords: language model, n-gram, neural network, perplexity, grammar, speech recognition, smoothing

Obsah

1 Úvod

Tématem této bakalářské práce je tvorba jazykového modelu pro hlasové rozpoznávání. Mluvená řeč je totiž nejpřirozenějším a nejspontánnějším způsobem komunikace mezi lidmi. Stále se zlepšující a vyvíjející se schopnosti výpočetní techniky navíc nabízejí možnost rozvoje tohoto druhu komunikace mezi člověkem a strojem. Dialog mezi člověkem a počítačem může být velmi prospěšný a v mnoha případech může také ulehčit život. Jedná se ovšem o složitou problematiku, v jejímž rámci je nutné technicky a algoritmicky vyřešit mnoho dílčích úloh, jako je například zpracování řečového signálu, syntéza řeči nebo automatické rozpoznávání řeči, včetně porozumění významu.



Obr. 1 - Schéma pro automatické rozpoznávání řeči

1.1 Cíl práce a motivace

Cílem této práce je nalezení metody pro tvorbu jazykového modelu pro hlasové rozpoznávání, při nedostatku nebo absenci trénovacích dat. Takové konverzace se v reálných situacích objevují relativně často a jedná se například o konverzace na úřadech, u lékaře či na poště. V takových případech jsou promluvy řečníků diskrétní a obsahují citlivá data. Proto není možné pořídit dostatečné množství záznamu.

Rozpoznávání řeči je ale velmi silným nástrojem, který může pomoci například sluchově postiženým při komunikaci s okolním světem. Právě konverzace na zmíněných místech (úřad, pošta, nemocnice) jsou součástí jejich každodenního života, a tedy správné

rozpoznávání slov v promluvách může značně usnadnit komunikaci, a také zkrátit čas strávený snahou o vzájemnou výměnu informací.

2 Rozpoznávání řeči a jazykový model

2.1 Rozpoznávání řečového signálu

Úloha automatického rozpoznávání řečového signálu, může být řešena dekompozicí do několika subúloh. Než si určíme jednotlivé dílčí úlohy, je třeba nejprve definovat několik pojmů. Řekněme, že $W = \{w_1, w_2, \ldots, w_n\}$ je posloupnost N slov a $\boldsymbol{O} = \{\boldsymbol{o_1}, \boldsymbol{o_2}, \ldots, \boldsymbol{o_T}\}$ je akustická informace, tj. posloupnost vektorů příznaků, odvozená z řečového signálu. Z této posloupnosti se pak lingvistický dekodér pokouší rozpoznat jednotlivá vyslovená slova. Snažíme se nalézt posloupnost slov, u níž bude maximální podmíněná pravděpodobnost naší posloupnosti slov za podmínky dané posloupnosti příznaků $\boldsymbol{O}$. Hledanou posloupnost slov označme W^*, podmíněnou pravděpodobnost potom můžeme označit $P(W|\boldsymbol{O})$. Použijeme Bayesovo pravidlo:

$$W^* = \underset{W}{argmax}\ P(W|\boldsymbol{O}) = \underset{W}{argmax}\ \frac{P(W)P(\boldsymbol{O}|W)}{P(\boldsymbol{O})} \tag{2.1}$$

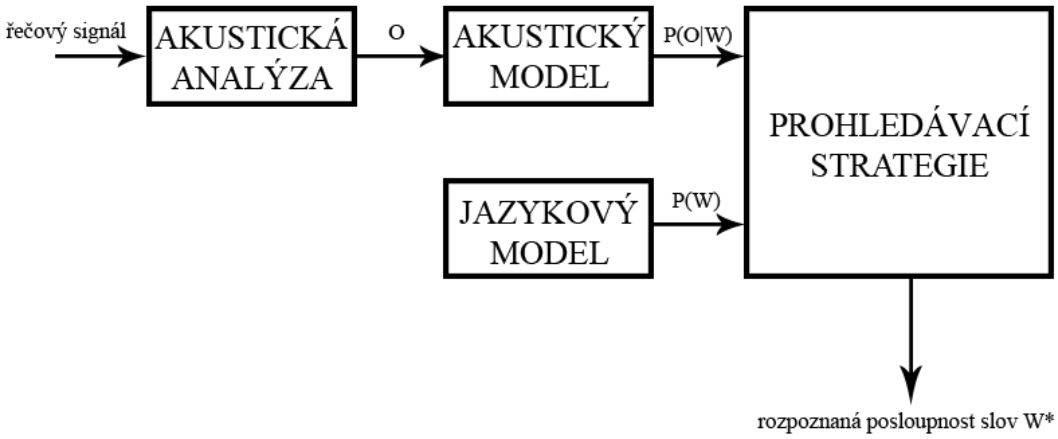
$P(\boldsymbol{O}|W)$ je pravděpodobnost, že nalezneme vektory příznaků $\boldsymbol{O}$ za podmínky vyslovení posloupnosti slov W. $P(W)$ je apriorní pravděpodobnost posloupnosti slov W a $P(\boldsymbol{O})$ apriorní pravděpodobnost výstupních vektorů příznaků. Vzhledem k tomu, že $P(\boldsymbol{O})$ není funkcí W, můžeme ho z rovnice vypustit.

$$W^* = \underset{W}{argmax}\ P(W)P(\boldsymbol{O}|W) \tag{2.2}$$

Nalezení hledané posloupnosti slov W^* je tedy problémem nalezení dvou pravděpodobností $P(W)$ a $P(\boldsymbol{O}|W)$. Jak již bylo zmíněno výše, celou úlohu rozdělíme na několik subúloh:

1) Akustická analýza řečového signálu, která slouží k nalezení posloupnosti vektorů příznaků $\boldsymbol{O}$.
2) Tvorba akustického modelu pro ocenění podmíněné pravděpodobnosti $P(\boldsymbol{O}|W)$.
3) Tvorba jazykového modelu sloužícího k ocenění apriorní pravděpodobnosti $P(W)$.

4) Použití prohledávací strategie. Snažíme se nalézt vektory příznaků ***O***, tak aby součin pravděpodobností *P(**O**|W)* P(W)* byl maximální. Vzhledem k výpočetní náročnosti tohoto hledání je třeba použít suboptimální prohledávácí strategie.



Obr. 2 - Systém rozpoznávání řeči dekomponovaný do jednotlivých bloků podle řešených subúloh

2.2 Jazykový model

Jazykový model je důležitou součástí systému rozpoznávání řeči. Jeho úkolem je poskytnout co nejrychleji a nepřesněji apriorní pravděpodobnost *P(W)* posloupnosti slov *W*. Ta pak poslouží bloku prohledávacích strategií pro nalezení rozpoznané posloupnosti slov *W**. Jazykový model pracuje vždy s konkrétním jazykem. Ten je definovaný konkrétním slovníkem a souborem pravidel, podle nichž se řetězí slova do větných celků, obě tyto zmíněné části určují jednotlivé jazyky a oddělují je od jiných. Jazykový model se pak snaží modelovat zmíněné zákonitosti, a tím zúžit a omezit možné posloupnosti slov *W*.

Omezení jazykového modelu mohou být deterministická nebo stochastická. U prvního jmenovaného typu omezení je předpoklad, že jsou volena pouze slova, která se nacházejí ve slovníku systému. Slovo navíc nemůže být vysloveno jinak, než definuje slovník. V tomto případě tedy některé posloupnosti slov nejsou vůbec přípustné.

Oproti tomu modely se stochastickým omezením, jinak také pravděpodobnostním, uvažují všechny možné posloupnosti slov *W*, ale přidělují jim různou pravděpodobnost. Tato pravděpodobnost bývá obvykle ve formě podmíněné pravděpodobnosti $P(W|\mathscr{X})$. Je třeba také poznamenat, že posloupnost vyslovených slov *W* se liší podle situace a prostředí, ve kterém řečník danou posloupnost pronáší. Proto je ideální pro každou jednotlivou situaci

připravit zvláště jazykový model, který situaci odpovídá. Konkrétní jazykový model tak bere v úvahu celý kontext, tj. jazyk, téma, prostředí, úmysl, význam sdělení, které chce mluvčí pronést, vliv stresu či řečníkovu náladu atd.

Problém tvorby jazykového modelu přináší opět dvě dílčí úlohy, první z nich je vlastní konstrukce jazykového modelu. To v podstatě znamená způsob určování apriorních pravděpodobností *P(W)* pro všechny možné posloupnosti slov *W*. Druhý úkol je pak již spojen s aplikací v reálné situaci. Aby rozpoznávání pomocí dekódovacího bloku bylo co nejúčinnější, neměl by jazykový model čekat na skončení promluvy, ale na základě její části generovat pravděpodobné posloupnosti slov *W* již v jejím průběhu.

V jednoduchých úlohách rozpoznávání hlasových povelů můžeme použít deterministické omezení ano/ne. Pravděpodobnosti *P(W)* pak nabývají pouze hodnoty 1 nebo 0. V této práci se ovšem budeme věnovat rozpoznávání souvislé mluvené řeči.
V takovém případě již deterministické omezení nebude dostačující, a proto použijeme omezení stochastické. Předpokládejme tedy, že řečník může pronést libovolnou posloupnost slov *W*. Každé z nich pak budeme přiřazovat různou pravděpodobnost *P(W)*. Poznamenejme také, že žádná posloupnost slov by neměla mít nulovou pravděpodobnost, neboť v takovém případě bychom ji zcela vyloučili z možného výstupu rozpoznávání.

2.3 Stochastický jazykový model

Jak už bylo řečeno úkolem stochastického jazykového modelu je určit apriorní pravděpodobnost *P(W)* pro každou posloupnost slov *W*. Pro každých *K* slov můžeme určit pravděpodobnost této posloupnosti jako pravděpodobnost každého jednotlivého slova za podmínky, že před ním je vyřčeno slovo jiné. Pro trojici slov w_1, w_2, w_3, určíme *P(W)* jako $P(w_1)P(w_2|w_1)P(w_3|w_1w_2)$. Obecně tedy pro jakoukoliv K-ticy slov *w* můžeme určit podle následujícího vztahu:

$$\begin{aligned} P(W) &= P(w_1^K) = P(w_1, w_2, \dots, w_K) \\ &= P(w_1)P(w_2|w_1)P(w_3|w_1w_2) \dots P(w_K|w_1w_2 \dots w_{K-1}) \\ &= P(w_1)P(w_2|w_1^1)P(w_3|w_1^2) \dots P(w_K|w_1^{K-1}) = \prod_{i=1}^{K} P(w_i|w_1^1) \end{aligned} \tag{2.3}$$

Pro pravděpodobnost libovolné počáteční části $w_1w_2 \dots w_k$ $(k \leq K)$ této posloupnosti pak platí:

$$P(w_1^k) = P(w_1^{k-1})P(w_k|w_1^{k-1}) = P(w_1)P(w_2|w_1^1)P(w_3|w_1^2) \dots P(w_k|w_1^{k-1}), \quad k = 2, \dots, K \tag{2.4}$$

Tento rozklad je vhodný především z důvodu praktické implementace jazykového modelu. Ten, jak již bylo zmíněno, potřebuje rozpoznávat posloupnosti slov již v průběhu promluvy. Tj. musí určovat pro úspěšné dekódování pravděpodobnost $P(w_1^k)$ postupně. V této reprezentaci pro její výpočet používá již spočtenou pravděpodobnost $P(w_1^{k-1})$. Bylo by možné použít i jiné rozklady, ale vzhledem k výše zmíněným skutečnostem je tento nejvhodnější. V praktické implementaci je pak také vhodné zavést slovo pro začátek a konec věty. V takovém případě pak posloupnost slov *W* nemusí být pouze jednou větou, ale jejich soustavou a navíc nese informaci o tom, kde jednotlivé věty končí a kde začínají. Je také praktické umožnit dekódovacímu mechanismu vypustit hypotézy posloupností slov s nízkou apriorní pravděpodobností *P(W)*. Toto zjednodušení vede k výraznému snížení výpočetní náročnosti.

2.4 N-gramové modely

N-gram je obecně definován, jako sled *n* po sobě jdoucích prvků. Mohou to být písmena, slova, ale obecně jakékoliv zřetězené prvky. V našem případě je n-gram pochopitelně posloupnost n-slov, kterou budeme používat pro hlasové rozpoznávání. Jazykový model pro jednotlivé n-gramy musí počítat apriorní pravděpodobnost n-tice slov *W*. Ideální by bylo, kdybychom dokázali spočítat všechny n-tice všech délek. To je ovšem vzhledem k enormní výpočetní náročnosti v podstatě nerealizovatelné. Provádí se proto aproximace, při níž se posloupnosti slov omezí na k-tice o stejné délce. Takovým modelům pak říkáme n-gramové modely. V závislosti na tom, jaké volíme *n,* pak n-gramy nazýváme zerogramy (n=0), unigramy (n=1), bigramy (n=2) a trigramy (n=3). Obecně mohou mít n-gramy libovolnou délku, ale s rostoucí velikostí *n* stoupá i výpočetní náročnost.

N-gramy jsou ideální pro jazyky s pevným řádem slov ve větě. Pro konkrétní téma pak existují relativně pevná pravidla pro to, jak vypadají jednotlivé věty a zjednodušuje se tak i odhad pravděpodobnosti jednotlivých n-gramů. Estimace jednotlivých pravděpodobností je postavena na relativní četnosti n-gramů, pokud máme dostatečně velký trénovací korpus (hlouběji se tomuto tématu budeme věnovat v následujících kapitolách).

Aproximací jazykového modelu do n-gramového modelu můžeme také upravit vztah pro výpočet apriorní pravděpodobnosti $P(w_1^k)$ z předchozí kapitoly. V n-gramovém modelu je

podmíněná pravděpodobnost $P(w_k|w_1^{k-1})$ závislá pouze na n-1 předchozích slovech a její výpočet tak bude vypadat:

$$P(w_1^k) \approx \prod_{i=1}^{k} P(w_i|w_{i-n-1}^{i-1}) \tag{2.5}$$

2.5 Posouzení kvality jazykového modelu

U jazykových modelů je také žádoucí, abychom měli nějakou možnost ohodnotit jejich kvalitu a mohli je porovnat mezi sebou a také určit, zda pro hlasové rozpoznávání mají významnou roli. Jako měřítko kvality můžeme například použít procentuální úspěšnost rozpoznávání s jazykovým modelem a bez něj. Tento údaj nám dá potom jasnou představu o tom, jak významný je jazykový model při rozpoznávání a existuje-li více jazykových modelů, pak nám procentuální vyjádření jasně ukáže, který z nich je pro řešený problém nejvhodnější. Kvalitu jazykového modelu můžeme ovšem posuzovat i odděleně, tj. nezávisle na ostatních částech systému pro rozpoznávání mluvené řeči.

Možností jak posuzovat kvalitu jazykových modelů je mnoho, ale v praxi je nejpoužívanější tzv. perplexita. Tento anglický výraz by mohl být do češtiny přeložen jako zmatek, komplikovanost či složitost.

Pro vysvětlení pojmu perplexita musíme nejprve spočítat apriorní pravděpodobnost slov $P(W) = P(w_1w_2\ldots w_K)$ výskytu slov W. Označme taky odhad této pravděpodobnosti $\bar{P}(W)$. Čím větší bude tato hodnota pro dostatečně obsáhlý testovací korpus, tím bude jazykový model kvalitnější a tím větší pak bude mít jazykový model význam pro vlastní rozpoznávání. Řekněme tedy, že máme testovací korpus jako posloupnost slov W čítající K prvků. Tato posloupnost bude obsahovat i znaky určující začátky a konce vět. Odhad pravděpodobnosti $\bar{P}(W)$ je také vhodné normalizovat. Pro tuto normalizaci použijme úvahu, že každá K-tice slov je v průměru K-krát menší než pravděpodobnost jednoho slova. Pak můžeme říct, že normalizace bude K-tá odmocnina $\bar{P}(W)$ a můžeme tedy psát:

$$\sqrt[K]{\bar{P}(w_1w_2\ldots w_K)} \tag{2.6}$$

Tato normalizace nám umožňuje porovnávat obecně odhad pravděpodobnosti $\bar{P}(W)$ různě dlouhých řetězců, a tedy i jazykové modely postavené na korpusech různých délek. Slouží také k relevantnímu porovnání jednoho jazykového modelu na dvou rozdílných korpusech. Pokud bude hodnota výrazu uvedeného výše nižší, ukazuje to na menší účinnost jazykového modelu. To může být způsobeno v zásadě dvěma důvody. První příčinou může být nízká kvalita jazykového modelu a druhým důvodem pak vysoká entropie (neuspořádanost) samotného jazyka. Pokud má jazyk, na němž je jazykový model postaven hodně volná pravidla (vysokou entropii) je hodnota pravděpodobnosti *P(W)* i jejího odhadu $\bar{P}(W)$ nižší než u jazyků s pevnou strukturou. Abychom tuto informaci dokázali definovat, zaveďme pojem perplexita korpusu (někdy také označovaná pouze jako perplexita). Značí se *PP* a její hodnota je dána převrácenou hodnotou normalizovaného tvaru odhadu pravděpodobnosti posloupnosti slov $\bar{P}(W)$.

$$PP = \frac{1}{\sqrt[K]{\bar{P}(w_1 w_2 \dots w_K)}} \tag{2.7}$$

Tento vztah můžeme použít jak na testovací tak i na trénovací korpus. Někdy je vhodné použít perplexitu korpusu v logaritmické podobě. *LP* tedy označme logaritmus perplexity korpusu a můžeme tedy psát:

$$LP = \log_2 PP = -\frac{1}{K}\bar{P}(W) \tag{2.8}$$

Pro n-gramové modely je vhodné tento vztah upravit následujícím způsobem:

$$LP = \log_2 PP = -\frac{1}{K}\sum_{i=1}^{K} \log_2 \bar{P}(w_1 | w_1 w_2 \dots w_{i-2} w_{i-1}) \tag{2.9}$$

Pokud tento vztah aplikujeme na konkrétní n-gramový model například trigram, můžeme vztah aproximovat do následujícího tvaru:

$$LP = -\frac{1}{K}\sum_{i=1}^{K} \log_2 \bar{P}(w_1 | w_{i-2} w_{i-1}) \tag{2.10}$$

Pro výpočet perplexity korpusu můžeme také vyjít z teorie informace. Zde je třeba uvážit generátor slov *W* coby informační zdroj. Pokud tento zdroj generuje dostatečně dlouhé řetězce slov *W*, potom LP (logaritmus perplexity korpusu) konverguje podle pravděpodobnosti ke křížové entropii $H(P,\bar{P})$ tohoto zdroje.

$$H(P,\bar{P}) = -\lim_{K\to\infty}\frac{1}{K}\sum_{i=1}^{K} P(W)\log_2 \bar{P}(W) \tag{2.11}$$

K v tomto vztahu je počet slov v textu generovaného zmíněným zdrojem. Upravme tento vztah ještě do následující podoby.

$$H(P) = -\lim_{K\to\infty}\frac{1}{K}\sum_{i=1}^{K} P(W)\log_2 P(W) \tag{2.12}$$

Nyní upravíme tento vztah do následující podoby. Zde je patrné, že křížová entropie je dána horním odhadem entropie zdroje produkujícího korpus. To znamená, že platí:

$$H(P,\bar{P}) \geq H(P) \tag{2.13}$$

Perplexita PP je pak pro dostatečně velký korpus:

$$PP = 2^{LP} = 2^{H(P,\bar{P})} \tag{2.14}$$

Z výše uvedených vztahů je zřejmé, že nejlepší jazykový model je takový, který dává přesný odhad pravděpodobnosti *P(W)*. Jazykové modely ovšem nedokáží přesně odhadnout tuto pravděpodobnost. Z čehož plyne, že i *PP* (perplexita korpusu) je závislá jak na korpusu, tak na použitém jazykovém modelu.

Čím vyšší tedy perplexita je, tím horší je schopnost jazykového modelu předpovědět následující slovo a můžeme ji tedy interpretovat jako střední stupeň rozvětvení jazyka. Perplexita skutečného jazyka například pro angličtinu je domněle okolo 30-50. Na textech z novin se slovní zásobou 5 000 slov je možné dosáhnout perplexity pro bigramy okolo 128 a pro trigramy 176. U hlasových dialogových systémů je pak hodnota perplexity pod 20. Z těchto příkladů je také patrné, že perplexita bigramu je nižší než u trigramu.

2.6 Metoda maximální věrohodnosti

Metoda maximální věrohodnosti (Maximum likelihood estimation – MLE) je přímou metodou odhadu pravděpodobnosti jednotlivých n-gramů z dostatečně velkého trénovacího korpusu. V jazykovém modelování je tedy třeba odhadnout pravděpodobnosti všech n-gramů $P(w_k|w_{k-n+1}\ldots w_{k-1})$. Mějme tedy korpus *S* (posloupnost slov $w_1w_2\ldots w_K$), která odpovídá posloupnosti *N=K-(n-1)* n-gramů h_nw_n, $h_{n-1}w_{n-1}$,… h_kw_k,… h_Kw_K. Přičemž historie $h_k = w_{k-n+1}\ldots w_{k-1}$.Nyní budeme odvozovat vztahy pro podmíněnou pravděpodobnost $P(w|h)$ a

sdruženou pravděpodobnost $P(w,h)$. Předpokládejme, že tyto pravděpodobnosti jsou dány parametrickými funkcemi $P(w|h;\theta)$ respektive $P(w,h;\theta)$. θ je pak množinou neznámých parametrů $\{\theta_i\}$ pro *i = 0, 1, ...* .Pro Parametry θ podmíněné pravděpodobnosti $P(w|h;\theta)$, jsou tyto parametry různé pro každou různou historii *h*. Můžeme tedy zapsat $\theta_i(h)$. Předpokládejme nyní rozdělení, v němž slova mající stejnou četnost, mají i stejnou pravděpodobnost. Parametr $\theta_r(h)$ je pak celková pravděpodobnost, tedy součet pravděpodobností všech slov *w*, která se v trénovacím korpusu objeví *r*-krát a to s konkrétní historií *h.* Toto rozdělení pak můžeme zaspat pomocí podmíněné pravděpodobností funkce takto:

$$P(w|h;\theta) = \frac{\theta_r(h)}{n_r(h)}, \qquad \theta = \{\theta_r(h)\}_{r=0,1,\dots} \tag{2.15}$$

Pro $n_r(h)$ (množinu slov, která se v korpusu vyskytnou *r*-krát s historií *h*) platí:

$$n_r(h) = \sum_{w:N(h,w)=r} 1 \quad a \quad \sum_{r=1}^{\infty} r n_r(h) = N(h) \tag{2.16}$$

Využijeme nyní větu o úplné pravděpodobnosti a dostáváme:

$$\sum_{r=0}^{\infty} \theta_r(h) = 1 \qquad a \quad \theta_r(h) \geq 0 \; pro \; \forall r, h \tag{2.17}$$

Úkolem nyní je odhadnout množinu parametrů θ, k tomu nám poslouží metoda maximální věrohodnosti, což je metoda bodového odhadu, který hledá takový bodový odhad θ^{MLE}, pro nějž je věrohodnostní funkce $F^{MLE}(\theta)$ dána následujícím vztahem

$$F^{MLE}(\theta) = \prod_{i=1}^{N} P(w_{i+n-1}|h_{i+n-1};\theta) = \prod_{i=n}^{N+n-1} P(w_i|h_i;\theta) \tag{2.18}$$

Ve svém maximu, tedy:

$$\theta^{MLE} = \underset{\theta}{\operatorname{argmax}}\, F^{MLE}(\theta) = \underset{\theta}{\operatorname{argmax}} \prod_{i=n}^{N+n-1} P(w_i|w_{i-n+1} \dots w_{i-1};\theta) \tag{2.19}$$

V tomto vztahu je w_i slovo na *i*-té pozici a *n* délka historie v použitých n-gramech. Více se ovšem využívá logaritmus této funkce $\sum_{i=n}^{N+n-1} \log P(w_i|h_i;\theta)$. Bodový odhad θ^{MLE} pak můžeme vyjádřit takto:

$$\theta^{MLE} = \underset{\theta}{\operatorname{argmax}} \sum_{i=n}^{N+n-1} \log \frac{\theta_{r_i}(h_i)}{n_{r_i}(h_i)} = \underset{\theta}{\operatorname{argmax}} \sum_{h'} \sum_{r=1}^{\infty} r n_r(h') \log \frac{\theta_{r_i}(h_i)}{n_{r_i}(h_i)} \qquad (2.20)$$

Nyní zapíšeme Lagrangeovu funkci, která hledá vázaný extrém logaritmu věrohodnostní funkce.

***Definice* 1** *Nechť* $f: \boldsymbol{R^n} \to \boldsymbol{R}, m < n, g_1, \dots g_m : \boldsymbol{R^n} \to \boldsymbol{R}$ *funkce. Položme* $V = \{x \epsilon \boldsymbol{R^n}; g_1(x) = 0 \wedge \dots \wedge g_m(x) = 0\}$.

Řekneme, že f *má v bodě* $a \in Df \cap V$ ***vázané lokální maximum*** *podmínkou* $a \in V$, *když* $\exists K(a,\delta)$ *tak, že* $\forall x \in K(a,\delta) \cap Df \cap V$ *platí* $f(x) \leq f(a)$

Řekneme, že f *má v bodě* $a \in Df \cap V$ ***vázané lokální maximum*** *podmínkou* $a \in V$, *když* $\exists K(a,\delta)$ *tak, že* $\forall x \in K(a,\delta) \cap Df \cap V$ *platí* $f(a) \leq f(x)$

Vázaná lokální minima a maxima funkce f *se nazývají lokální vázané extrémy*

Lagrangeova funkce má následující tvar:

$$L(\theta, \xi) = \sum_{h'} \sum_{r=1}^{\infty} r n_r(h') \log \frac{\theta_{r_i}(h_i)}{n_{r_i}(h_i)} + \sum_{h'} \xi_{h'} \left(1 - \sum_{r=0}^{\infty} \theta_{r_i}(h')\right) \qquad (2.21)$$

V této rovnici vystupuje $\xi_{h'}$, což jsou Langrangeovy multiplikátory (též agrangeovy neurčité součinitele).

Věta 1 ***(Lagrange)*** *Nechť* $f: \boldsymbol{R^n} \to \boldsymbol{R}, m <$ n *funkce spojitě diferencovatelné na otevřené množině* Ω *obsahující* Va *nechť* $\forall x \in \Omega$ *platií, že hodnotamatice* $\left[\frac{\partial g_i}{\partial x_j}(x)\right]_{i,j}$ *je rovna* $m.$ *Buď* $L: \boldsymbol{R^n} \to \boldsymbol{R}$ *funkce definovaná vztahem*

$L(x_1, \ldots, x_n) = f(x_1, \ldots, x_n) + \lambda_1 g_1(x_1, \ldots, x_n) + \ldots + \lambda_m g_m(x_1, \ldots, x_n)$

Funkce L *se nazývá Lagrangeova funkce a konstanty* $\lambda_1, \ldots, \lambda_m$ *se nazývají Lagrangeovy multiplikátory*

Proložíme tedy Lagrangeovu funkci parciální derivací a dostáváme tedy:

$$\frac{\partial L(\theta, \xi)}{\partial \theta_r(h)} = \frac{r n_r(h)}{\theta_r(h)} - \xi_h = 0, r = 0,1,2, \ldots \tag{2.22}$$

Celou rovnici nyní přenásobíme parametrem $\theta_r(h)$ a sečteme přes všechna *r*, dostaneme tak:

$$\xi_h \sum_{r=0}^{\infty} \theta_r(h) = \sum_{r=1}^{\infty} r n_r(h) \tag{2.23}$$

Rovnici dále upravíme na $\xi_h = N(h)$ a dosadíme do předchozí rovnice. Tím dostáváme nejlepší odhad parametru θ_r^{MLE}. Pro ten platí tento vztah:

$$\theta_r^{MLE}(h) = \frac{r n_r(h)}{N(h)} \tag{2.24}$$

Podmíněná pravděpodobnost výskytu slova *w* za podmínky, že mu předcházela historie *h,* počítaná metodou maximální věrohodnosti pak vypadá následovně:

$$P^{MLE}(w|h) = \frac{r}{N(h)} = \frac{N(h,w)}{N(h)} = \frac{N(h,w)}{\sum_{w'} N(h,w')} \tag{2.25}$$

Velmi podobně bychom mohli určit odhad sdružené pravděpodobnosti $P^{MLE}(h,w)$. Uveďme tedy pouze výsledný vztah:

$$P^{MLE}(w,h) = \frac{r}{N} = \frac{N(h,w)}{N} = \frac{N(h,w)}{\sum_{h'w'} N(h',w')} \tag{2.26}$$

V těchto rovnicích vystupuje *r* – počet výskytů n-gramu *hw* v trénovacím korpusu, *N(h)* – počet výskytů historie *h*. Tato metoda má několik úskalí, hlavním je potřeba extrémně velkého trénovacího korpusu. Při malých četnostech určitých jevů pak dochází k velmi nepřesnému odhadu. Navíc je možné, že některé jevy (například určité n-gramy) nebudou obsaženy v testovacím korpusu. V takovém případě je možné použít vyhlazování.

2.7 Vyhlazování

N-gramový model je tedy aproximací stochastického modelu, z čehož plyne několik problémů. Zásadní nevýhodou takovéto aproximace jsou vysoké nároky na množství testovacích dat, která jsou použita na určení n-gramového modelu. Mějme tedy například bigramový model a trénovací posloupnost znaků *a b a b b a*. Všechny možné bigramy jsou potom (v závorkách jsou uvedeny četnosti): *ab (2), ba (2), bb(1)*. Nyní mějme řetězec *a b a a b a*. Na první pohled jsou si tyto řetězce velmi podobné, nicméně ve druhém z nich se vyskytuje bigram *aa*, který ale není obsažen v jazykovém modelu. Pravděpodobnost tohoto bigramu je tedy rovna nule:

$$P_{(a|a)} = 0 \tag{2.27}$$

Úpravu modelu, která zajistí, že bude každému řetězci přiřazena pravděpodobnost nenulová, nazýváme vyhlazování. Tato úprava odečte část pravděpodobnosti pozorovaným jevům (tedy takovým, které jsou obsaženy v n-gramovém modelu) a rozdělí ji mezi jevy nepozorované (ve výše uvedeném přikladu jev *aa*, tedy jevy, které nejsou obsaženy v n-gramovém modelu). Pro vyhlazení jazykového modelu máme k dispozici několik metod. Patří mezi ně například: Bayesova metoda s aditivním vyhlazováním, Goodův-Turingův odhad, metoda odhadu s postupným vynecháváním jednoho jevu, Katzův diskontní model, model s absolutním diskontem nebo Wittenův-Bellův model.

2.7.1 Bayesova metoda odhadu

Mějme diskrétní rozdělení pravděpodobnosti, tj. kdy každé slovo *w* má přiřazenou obecně jedinečnou hodnotu pravděpodobnosti $P_{(w|h;\,\theta)} = \theta_w$, a dál předpokládejme rovnoměrné apriorní rozdělení parametru θ, pak vede Bayesův přístup na odhad:

$$P^B(w|h) = \theta_w^B = \frac{N(h,w)+1}{N(h)+V} = \frac{N(h,w)+1}{\sum_W N(h,w)+V} \tag{2.28}$$

Tento způsob vyhlazování se také nazývá aditivní, neboť se k čitateli i jmenovateli přičítá určitá vhodná hodnota. Toto pochopitelně není jediná možná varianta aditivního vyhlazování. V jeho dalších modifikacích se přičítají například počty obecnějších rozdělení. Což mohou být například rozdělení, která udávají pravděpodobnost n-1 gramů. Toto vyhlazování sice plní princip, tedy přidání nenulové hodnoty nepozorovaným jevům na úkor jevů pozorovaných. Vzhledem k často vysokému počtu nepozorovaných jevů v trénovacím korpusu, ale mění výrazně pravděpodobnosti jevů pozorovaných, kterým tak hodnota pravděpodobnosti výrazně ubývá. Vzhledem k tomuto problému není Bayesova metoda odhadu, respektive aditivní vyhlazování, příliš často využívanou metodou.

2.7.2 Goodův-Turingův odhad

Tato metoda byla vytvořena britským matematikem, logikem a kryptoanalytikem Alanem Turingem a jeho kolegou Irvingem Johnem Gooodem po druhé světové válce. A to jako způsob odhadu četnosti neznámých (zatím nepozorovaných) živočišných druhů. Tato metoda je ale využitelná i v jazykovém modelování. Good-Turingův odhad tedy ubírá pravděpodobnost jednotlivým pozorovaným jevům a přidává ji jevům nepozorovaným (myšleno v trénovacím korpusu). Při použití na n-gramovém modelu je tedy pozorovaný jev zaznamenaný n-gram, který se vyskytl alespoň jednou v trénovacím korpusu o velikosti *N* (*N* je celkový počet n-gramů v trénovacím korpusu). Řekněme že n-gram se v trénovacím korpusu vyskytuje *r*-krát. Good-Turingův odhad však zavádí upravenou absolutní četnost, kterou můžeme označit r^*. Předpis pro tuto „lepší“ (ve smyslu vyřešení problému vyhlazování) četnost je dán následující rovnicí:

$$r^* = \frac{(r+1)n_{r+1}}{n_r} \tag{2.29}$$

Kde n_r je počet všech navzájem různých jevů, respektive n-gramů, které se vyskytují v trénovacím korpusu právě *r*-krát. Je možné dokázat, že tato metoda odhadu pravděpodobnosti s upravenou četností nepozorovaných jevů, je dána četností jevů (n-gramů), které se v trénovacím korpusu vyskytují jedenkrát, označují se také jako singletony. Nyní můžeme vyjádřit podmíněnou pravděpodobnost odhadnutou pomocí Good-Turnigova odhadu $P^{GT}(w|h)$ n-gramu *w* s historií *h*. Nejprve upravíme rovnici pro r^*:

$$r^* = \frac{(r+1)n_{r+1}(h)}{n_r(h)} \tag{2.30}$$

Potom $P^{GT}(w|h)$ bude:

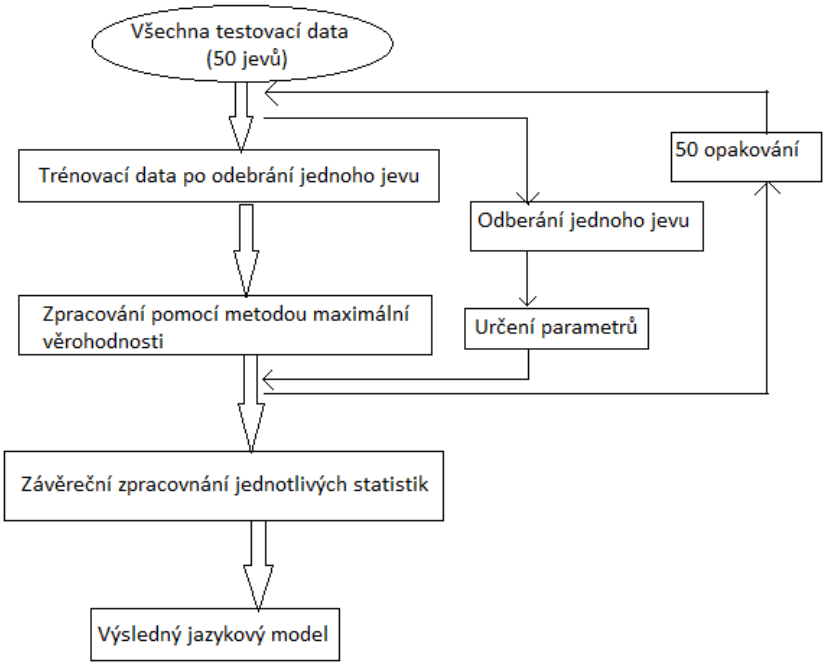
$$P^{GT}(w|h) = \frac{r^*}{N(h)} \tag{2.31}$$

2.7.3 Odhad s postupným vynecháním jednoho jevu

Tato metoda je založena na metodě odložených dat (z anglického held-out). Jejím principem je odložení části trénovacího korpusu. Ze zbytku jsou pak vypočteny statistiky (četnosti) jednotlivých jevů (n-gramů). Druhou (odloženou část) potom použijeme pro odhad parametrů jazykového modelu. Metoda tedy využívá principu, kdy na základě statistik první části trénovacího korpusu hledáme nejlepší možné parametry, které vystihují četnost a rozložení jevů druhé (odložené části). Pro odhad parametrů je většinou využita metoda nejvyšší věrohodnosti, možné je ovšem použití i jiných metod. Při použití zmíněné metody se tedy snažíme nalézt takové parametry, při kterých věrohodnostní funkce nabývá svého maxima. Tento princip snižuje „vychýlenost" odhadu parametrů, která ve velké míře vzniká, když je použit stejný testovací korpus pro určení statistik i pro odhad parametrů. Hlavním úskalím této metody je potřeba odložené části trénovacích dat, čímž snižujeme robustnost korpusu. Nabízí se použití takzvané **křížové ověřovací techniky** (anglicky cross-validation). Při této technice jsou opět trénovací data rozdělena na dvě části. Jednu použijeme pro určení statistik a druhou potom pro odhad parametrů. Tyto úlohy jednotlivých částí se posléze prohodí a první část je tedy použitá pro odhad parametrů a druhá použita pro určení statistik. Tím je využit celý testovací korpus a snižují se nároky na robustnost testovacích dat. Speciální variantou křížové ověřovací techniky je potom **metoda odhadu s postupným vynecháváním jednoho jevu** (anglicky Leaving-one-out methode, často se také používá zkratka Loo). Tato metoda vynechá pouze jeden jev (n-gram *wh*). Zbytek trénovacích dat pak použije pro určení statistik. Celý postup je pak opakován pro všechny n-gramy *wh*. Pokud tedy máme například *N* jevů, odložíme první z nich, ze zbytku (tedy *N*-1 jevů) vypočteme statistiky. Pak odložíme druhý jev a celý postup opakujeme, dokud neprojdeme všem *N* jevů. Z čehož plyne, že má-li jev (n-gram) četnost *r* po jeho odebrání je pak jeho četnost *r*-1. Pakliže se jedná o singleton, jeho vynecháním simuluje nepozorované jevy, jež nejsou obsaženy v trénovacích datech. K odhadu parametrů je pak opět použita metoda maximální věrohodnosti. Pro odhad θ pak použijeme následující vztah:

$$\theta^{Loo} = \underset{\theta}{\operatorname{argmax}} \prod_{i=n}^{N+n-1} P(w_i|h_i, S^{-i}; \theta) = \underset{\theta}{\operatorname{argmax}} \prod_{i=n}^{N+n-1} P^{Loo}(w_i|h_i; \theta) \tag{2.32}$$

V této rovnici vystupuje S^{-i}, což je trénovací korpus s vynecháním *i*-tého prvku (n-gramu) z původního korpusu a $P^{Loo}(w_i|h_i;\theta)$ je modifikovaná pravděpodobností funkce vynecháním jednoho jevu. Hlavní výhodou této metody je využití všech trénovacích dat a také simulace nepozorovaných jevů vynecháním singletonů. Na následujícím obrázku je v diagramu naznačené, jak funguje metoda odhadu s postupným vynecháním jednoho jevu na příkladu, kdy trénovací data obsahují 50 bigramů



Obr. 3 - Diagram metody odhadu s postupným vynecháním jednoho jevu

2.7.4 Ústupové schéma vyhlazování

Při řešením problému s příliš řídkými testovacími daty je možné použít také metodou ústupového vyhlazování. Princip ústupového schématu spočívá v tom, že pokud není dostatečně robustní korpus trénovacích dat, počítáme vyhlazený odhad nejen původních relativních četností, ale také z takzvaného zobecněného rozdělení n-gramů, což většinou bývají n-1 gramy. U n-gramu *wh* je v takovém případě vynecháno nejčastěji poslední slovo historie *h*. Pokud tedy máme například trigramy, budeme v ústupovém schématu vyhlazování používat i bigramy. Historie zkrácená o poslední slovo se nazývá zobecněná historie. Pro odhad pravděpodobnosti n-gramu *wh* můžeme tedy zapsat:

$$P_{BO}(w|h) = \begin{cases} d_{N(h,w)} \dfrac{N(h,w)}{N(h)} & pro\ N(h,w) > 0 \\ B(h)\beta\left(w\middle|\bar{h}\right) & pro\ N(h,w) = 0 \end{cases} \tag{2.33}$$

B(h) je ústupová váha, která obsahuje jednak přepočítané četnosti, tak také normalizační člen. Touto váhou přenásobíme pravděpodobnosti zobecněného rozdělení. Dále zde vystupuje $d_{N(h,w)}$, což je diskontní činitel. Ten snižuje relativní četnosti pozorovaných jevů a jeho hodnota se pohybuje mezi 0 a 1. Ušetřenou váhu pak předává ústupové schéma nepozorovaným jevům (n-gramům s nulovou četností). Pro ústupovou váhu platí následující vztah:

$$B(h) = \frac{1 - \sum_{w:N(h,w)>0} d_{N(h,w)} \frac{N(h,w)}{N(h)}}{\sum_{w:N(h,w)=0} \beta(w|\bar{h})} \tag{2.34}$$

V následujících odstavcích jsou vysvětleny dvě metody, které používají ústupové schéma vyhlazování (anglicky backing-off). Normalizační člen $\beta(w|\bar{h})$ je obecně rozdělení se „zkrácenou" historií $\bar{h}$. Součet všech těchto pravděpodobností musí být roven 1. Můžeme tedy psát $\beta(w|\bar{h}) = P_{BO}(w|\bar{h})$ a dále platí:

$$\sum_{w} P_{BO}(w|h) = 1 \tag{2.35}$$

Diskontní činitel *d* může být obecně pro každý n-gram různý, ale většinou se používá shodný činitel pro shodné četnosti. Vzhledem k tomu, že $N(h,w) = r$, můžeme psát součinitel jako d_r. Někdy se ovšem místo diskontního činitele zavádí diskontní faktor $\lambda_{N(h,w)} = 1 - d_{N(h,w)} = 1 - d_r$. Pokud z tohoto vztahu vyjádříme $d_{N(h,w)}$ a dosadíme do předešlé rovnice pro vyjádření ústupové váhy. Pro tuto váhu vyjádřenou diskontním faktorem dostáváme rovnici:

$$B(h) = \frac{1 - \sum_{w:N(h,w)>0} (1 - \lambda_{N(h,w)}) \frac{N(h,w)}{N(h)}}{\sum_{w:N(h,w)=0} \beta(w|\bar{h})} \tag{2.36}$$

A po úpravě:

$$B(h) = \frac{\sum_{w:N(h,w)>0} \lambda_{N(h,w)} \frac{N(h,w)}{N(h)}}{\sum_{w:N(h,w)=0} \beta(w|\bar{h})} \tag{2.37}$$

2.7.5 Witten-Bellův model

Witten-Bellův model využívá princip ústupového schématu. Zapomíná se tedy poslední (jinak také „nejstarší“) slovo historie *h* a používá tak v podstatě model n-1 gramů pro vyhlazení n-gramového modelu. Witten-Bellův model používá diskontní činitel v následujícím tvaru:

$$d_{N(h,w)} = d_h = \frac{N(h)}{N(h) + n(h)} \; pro \; N(h,w) > 0 \tag{2.38}$$

Kde *n(h)* je četnost slov, kterým předchází historie *h*. Tento činitel můžeme dosadit do výše uvedené rovnice pro odhad pravděpodobnosti a dostáváme tedy:

$$\begin{aligned} P_{WB}(w|h) &= \frac{N(h,w)}{N(h) + n(h)} \; pro \; N(h,w) > 0 \\ P_{WB}(w|h) &= \frac{n(h,w)}{N(h) + n(h)} \frac{\beta(w|\bar{h})}{\sum_{w:N(h,w)=0} \beta(w|\bar{h})} \; pro \; N(h,w) \\ &= 0 \end{aligned} \tag{2.39}$$

2.7.6 Katzův model vyhlazování

Katzovo vyhlazování bylo definováno americkým matematikem Nickem Katzem v roce 1987, jako rozšíření Good-Turingova odhadu přidáním kombinace n-gramů vyššího a nižšího řádu. Princip Katzova vyhlazování můžeme ukázat na bigramovém modelu. Pro bigram w_{i-1}^{i}s četností $r = n(w_{i-1}^{i})$, se spočítá opravená četnost s použitím rovnice:

$$n_{katz}\left(w_{i-1}^{i}\right) = \begin{cases} d_r r \; pro \; r > 0 \\ \alpha(w_{i-1}) P^{MLE}(w_i) \;\; pro \; r = 0 \end{cases} \tag{2.40}$$

Což znamená, že ohodnocení bigramů s nenulovou četností *r* je sníženo pomocí koeficientu d_r. Hodnota tohoto koeficientu je přibližně $\frac{r^*}{r}$, kde r^*je odhad Good-Turingovy metody. Ohodnocení o něž je snížena hodnota pozorovaných bigramů, je pak rozděleno mezi bigramy s nulovou četností (tedy nepozorované bigramy) pomocí jazykového modelu nižšího řádu (v tomto případě unigramy). Hodnota $\alpha(w_{i-1})$ je volena tak, že celková četnost $\sum_{w_i} n_{katz}(w_{i-1}^{i})$ se nemění, což znamená, že $\sum_{w_i} n_{katz}(w_{i-1}^{i}) = \sum_{w_i} n(w_{i-1}^{i})$. Odpovídající hodnota $\alpha(w_{i-1})$ je potom:

$$\alpha(w_{i-1}) = \frac{1 - \sum_{w_i:n(w_{i-1}^i)>0} P^{katz}(w_i|w_{i-1})}{\sum_{w_i:n(w_{i-1}^i)=0} P^{MLE}(w_i)} = \frac{1 - \sum_{w_i:n(w_{i-1}^i)>0} P^{katz}(w_i|w_{i-1})}{1 - \sum_{w_i:n(w_{i-1}^i)>0} P^{MLE}(w_i)} \quad (2.41)$$

Pro výpočet P^{katz} upravené četnosti platí:

$$P^{katz}(w_i|w_{i-1}) = \frac{n_{katz}(w_{i-1}^i)}{\sum_{w_i} n_{katz}(w_{i-1}^i)} \quad (2.42)$$

Pro bigramy s velkou četností, kde je odhad považován za spolehlivý, Katzova metoda nesnižuje ohodnocení, což znamená, že $d_r = 1$. To platí pro všechny jevy, kde $r>k$ pro dané k (pro Katzovu metodu je k obvykle voleno $k = 5$). Pro $r<k$ pak jsou pak upravené četnosti počítány z Good-Turingova odhadu. Můžeme definovat následující omezení:

$$1 - d_r = \mu(1 - \frac{r^*}{r}) \quad (2.43)$$

Pro $r \in \{1, 2, \dots, k\}$ pro určitou konstantu μ. Good-Turingův odhad předpovídá, že celková četnost, která je přiřazena bigramům s nulovou četností je $n_0 0^* = n_0 \frac{n_1}{n_0}$, můžeme tedy zapsat další omezení:

$$\sum_{r=1}^{k} n_r(1 - d_r)r = n_1 \quad (2.44)$$

Z těchto dvou rovnic dostáváme řešení pro d_r:

$$d_r = \frac{\frac{r^*}{r} - \frac{(k+1)n_{k+1}}{n_1}}{1 - \frac{(k+1)n_{k+1}}{n_1}} \quad (2.45)$$

Jak již bylo zmíněno pro bigramový model tedy Katzova metoda používá unigramový model, na nějž je aplikována metoda maximální věrohodnosti popsaná výše.

2.7.7 Interpolační schéma

Další metodou jak vyhlazovat nedostatečně robustní korpus je takzvané **lineární interpolační schéma**. Toto schéma používá jak relativní četnosti jednotlivých n-gramů, tak i

zobecněné rozdělení. Z obou těchto rozdělení je pak spočítán průměr (konkrétně se jedná o vážený průměr). Zapsat ho můžeme následujícím způsobem:

$$P_{LI}(w|h) = d_{N(h,w)} \frac{N(h,w)}{N(h)} + \left(1 - d_{N(h,w)}\right)\beta\left(w|\bar{h}\right) \tag{2.46}$$

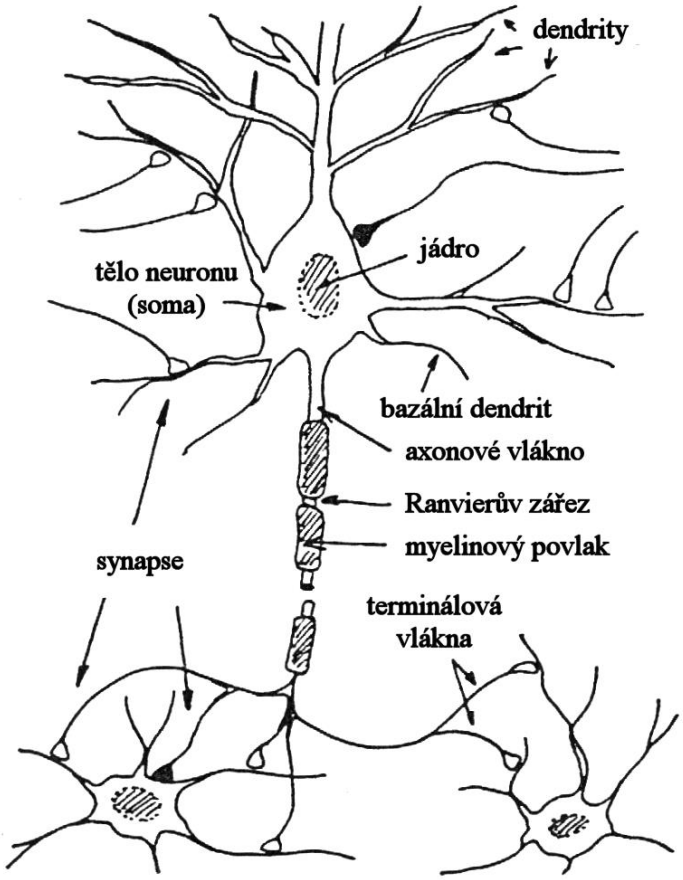
Dosadíme diskontní faktor za diskontní činitel a dostáváme:

$$P_{LI}(w|h) = (1 - \lambda_{N(h,w)}) \frac{N(h,w)}{N(h)} + \lambda_{N(h,w)}\beta\left(w|\bar{h}\right) \tag{2.47}$$

Lineární interpolaci je možné použít pro jakoukoliv kombinaci dvou jazykových modelů, nejen pro relativní četnosti a zobecněné rozdělení daného n-gramového modelu. Vzhledem k tomu, že metoda počítá vážený průměr obou rozdělení, výsledné rozdělení nebude nikdy horší, než kterékoliv z původních rozdělení.

2.8 Neuronové sítě

N-gramy pochopitelně nejsou jedinou možností jak realizovat jazykové modely, jednou z alternativ jsou například neuronové sítě (respektive umělé neuronové sítě). Umělé neuronové sítě jsou inspirovány poznatky o nervových soustavách živých organismů a neuronech, které tyto sítě tvoří. Pro neuronové sítě v živých organismech sice zatím není znám přesný algoritmus, přesto ovšem probíhá řada experimentů, které dávají slibné výsledky. Zajímavá pro naši problematiku je především schopnost získávat a předávat data, která nejsou zřejmá či schopnost řešit nelineární úlohy. Nejdůležitější vlastností je ovšem schopnost se učit a zpracovávat velké množství dat. Sítě jsou poskládané z jednotlivých neuronů, které jsou navzájem provázány a předávají si mezi sebou signál, ve formě elektrického impulsu. Tyto signály navíc transformují pomocí daných přenosových funkcí.



Obr. 4 – Neuron

Neuron živých organismů je tvořen z těla neuronu (soma) v němž se nachází jádro této nervové buňky (velikost mezi 6µm - 100µm) a jednotlivých výběžků, ty mohou být buď:

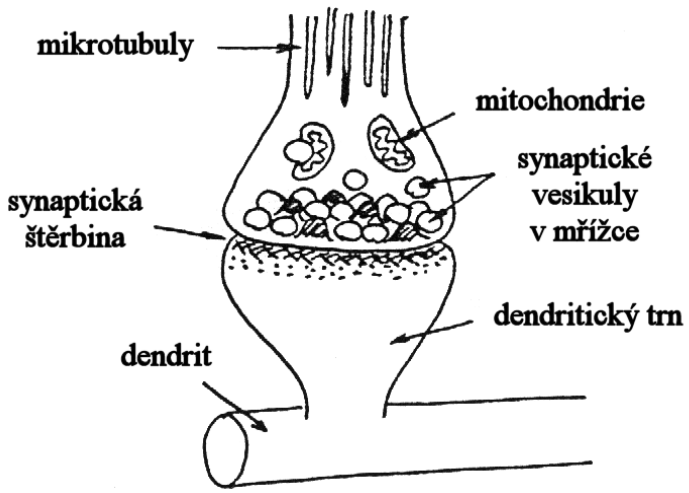
- Krátké – takzvané dendrity (dostředivé)
- Dlouhé – takzvané neurity respektive axony (odstředivé)

Dendritů může mít neuron poměrně mnoho. Tyto výběžky jsou u těla široké a posléze se větví. Na jejich povrchu můžeme najít dendritické trny. Na těchto trnech pak mají tyto dostředivé krátké výběžky synapse. Podle počtu dendritů se neurony dělí na:

- Unipolární – žádný dendrit (pouze axon)
- Bipolární – jeden dendrit
- Multipolární – několik dendritů

Struktura dendritů je velmi podobná struktuře těla neuronu. Naopak axon má každá neuron pouze jeden. Může se ale větvit a potom vytváří takzvané kolaterály. Konečné rozdělení je pak pojmenováno telodendron. Na koncích rozvětveného axiomu je speciální struktura, která při dráždění neuronu uvolňuje neurotransmitery. Pokud je axon obalený myelinovou pochvou, nazývá se nervové vlákno, tyto výběžky jsou velmi dlouhé. U velkých

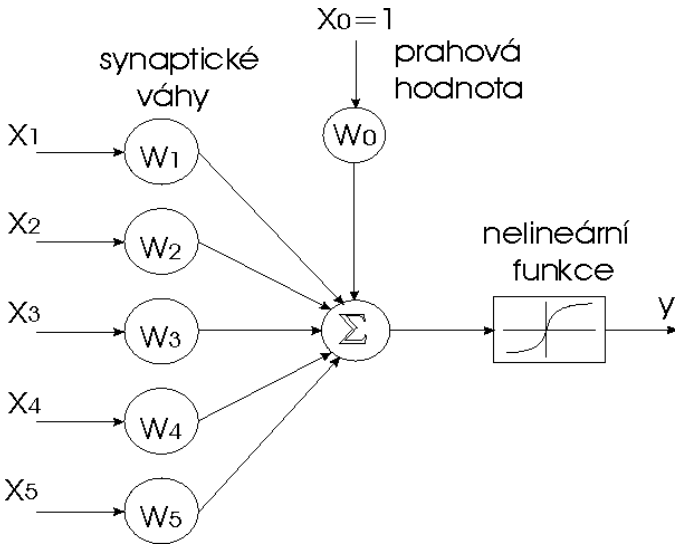
zvířat mohou dosáhnout délky až několik metrů u člověka to bývá zhruba 1m. Na následujícím obrázku je naznačená komunikace mezi dvěma neurony.



Obr. 4 – Detail dendritického trnu a ukázka synapse

2.8.1 Umělá neuronová síť

Umělé neuronové sítě jsou tedy jeden z výpočetních modelů, používaných například v umělé inteligenci. Vzorem takové sítě je, jak již bylo zmíněno, nervová soustava živých organismů a jedná se o distribuovaný výpočetní systém, který je složen z dílčích subsystémů (neuronů), které jsou mezi sebou provázány. Základním principem je fakt, že každý neuron má libovolně mnoho vstupů, ale pouze jediný výstup. Model umělého neuronu poprvé navrhli v roce 1943 pánové McCulloch a Pitts a tento model se používá dodnes. Skládá se ze tří základních částí. Tou první je vstupní část, kde jsou jednotlivé vstupy. Každý tento vstup má svoji takzvanou synaptickou váhu, pomocí níž mohou být jednotlivé vstupy buď potlačovány, nebo zvýhodňovány. Druhou částí umělého neuronu je takzvaná výkonná jednotka. Zde je zpracován signál ze vstupů a následně vygenerována výstupní odezva. Třetí část potom přenáší odezvu vypočítanou ve druhé části na další neurony. Paměť neuronu není centralizovaná autonomní jednotka, ale je realizována ve vstupní části neuronu formou váhových koeficientů.



Obr. 5 – Jednoduchý model umělého neuronu (McCulloch a Pitts)

Z tohoto modelu je patrné, že výkonná jednotka, v níž se zpracovávají vstupy neuronu je mnohem jednodušší než výkonná jednotka výpočetních systémů a je v zásadě tvořena jednoduchou funkcí. Výstupní hodnotu y, můžeme zapsat pomocí funkce $y = f(\xi)$. Dále můžeme psát:

$$\xi = \sum_{i=1}^{n} w_i x_i - \theta = \sum_{i=0}^{n} w_i x_i \tag{2.48}$$

Pro výše zmíněnou funkci *f* platí:

$$f(\xi) = \frac{1}{1 + e^{-\lambda \xi}} \tag{2.49}$$

Binární prahový neuron, je takový neuron, který má pevný počet vstupů, danou prahovou hodnotu a na výkonnou jednotku přichází vzruchy jako binární číslo tedy 1 nebo 0 (převážené synaptickou váhou). Synapse, která přichází je buď inhibiční (hodnota -1), nebo excitační (hodnota +1). Působení vzruchů na neuron můžeme naznačit v následujícím tvaru:

$$x_i(t+1) = sgn\left(\sum_{j=1}^{N} w_{ij} x_j - \mu_i\right) \tag{2.50}$$

V tomto vztahu se vyskytuje $x_i(t+1)$, což je stav i-tého neuronu, w_{ij} je synaptická váha na cestě z neuronu *j* do *i* a μ je prahová hodnota neuronu *i*. Binární prahové neurony dokáží spočítat jen jednoduché binární (logické) operace, pokud se ovšem správně zřetězí, je

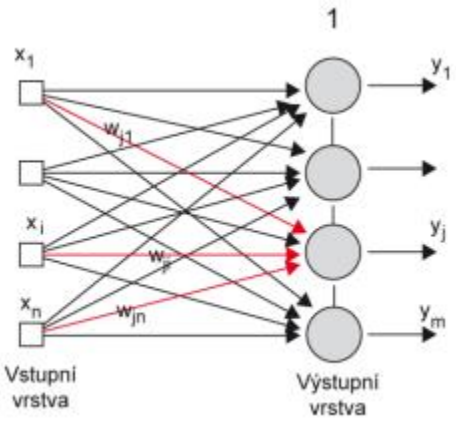
pomocí těchto jednoduchých částí neuronové sítě možné spočítat v podstatě jakoukoliv úlohu. Obecně je ovšem možné využít neurony s jakoukoliv funkcí ve výkonné části.

2.8.2 Typy neuronových sítí

Způsobů, kterými je možno zřetězit neurony do neuronových sítí je celá řada, každý takový způsob je pak vhodný pro jinou třídu úloh. Je také možné různé druhy neuronových sítí kombinovat či vzájemně doplňovat. Podobně jako v n-gramovém modelu je třeba mít určité množství dat (reprezentačních příkladů), poté co zvolíme vhodnou strukturu neuronové sítě (několik druhů bude popsáno v následujících odstavcích). Kromě trénovacích dat budeme opět i v tomto případě potřebovat i testovací a učící data. Poté co je neuronová síť vytvořena, standardně se nastaví synaptické váhy na náhodná čísla. Tyto váhy pak neuronová síť upravuje tak, aby se minimalizovala chyba, tj. odchylka mezi skutečným a požadovaným výstupem. Je třeba také vhodně zvolit trénovací algoritmus, ten je pro každou neuronovou síť unikátní, jedná se ale vždy o iterační proces.

2.8.2.1 Perceptron

Perceptron je nejjednodušším modelem dopředné neurnové sítě. Je složena pouze z jednoho neuronu. Perceptron vynalezl Frank Rosenblatt v roce 1957. Omezení jednoduchého perceptronu je ovšem velmi významné, tato struktura je totiž použitelná pouze pro množiny, které jsou lineárně separovatelné. Proto vzniklo rozšíření v podobě vícevrstvé perceptronové sítě.

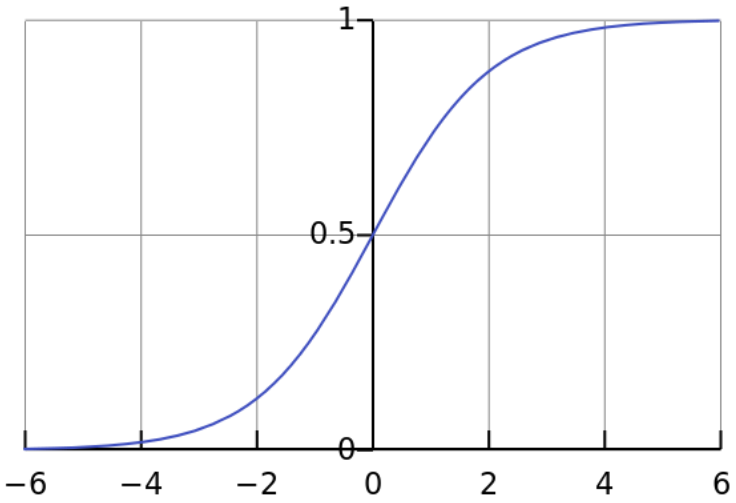


Obr. 6 - Perceptron

2.8.2.2 Vícevrstvá perceptronová síť

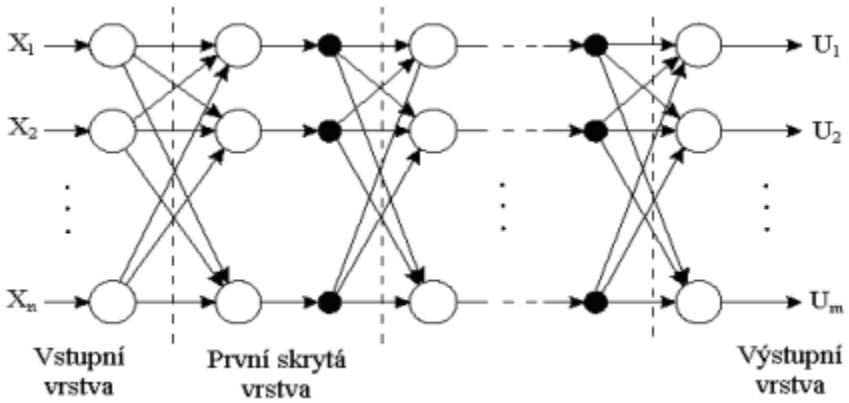
Tato síť je vůbec nejpoužívanějším typem neuronových sítí. Co se vícevrstvých perceptronových sítí týče, prvně byly použity v roce 1969, dvojící Minsky a Papert. Právě oni ukázali schopnost dvouvrstvé perceptronové sítě a její výhodu oproti jednovrstvým sítím (perceptronům). Nedokázali ovšem vyřešit problém s úpravou synaptických vah skrytých

neuronů. Tento úkol vyřešili až vědci Rumelhartem, Hintonem a Wiliams a to v roce 1986. Ti odstranili nespojitou prahovací funkci a nahradili ji spojitou (diferencovatelnou) funkcí. Tato úprava umožnila použít gradientní metodu optimalizace pro učení. Aktivační funkcí této sítě je nejčastěji sigmoida ($\frac{1}{1+e^{-\lambda\xi}}$). Což je logická matematická funkce, jejíž graf má následující tvar:



Obr. 7 – Sigmoida

Pro vícevrstvé perceptronové sítě se používá mnoho algoritmů, tím základním je takzvaný backpropagation (princip tohoto algoritmu je v testování, zda již reálný výstup odpovídá požadovanému výstupu a pokud ne, upravuje synaptické váhy, tak dlouho dokud není dosaženo požadovaného výstupu). K dispozici jsou i sofistikovanější metody učení, jako je například metoda sdružených gradientů, Levenbergova-Marquardtova metoda a další.



Obr. 8 – Struktura vícevrstvé perceptronové sítě

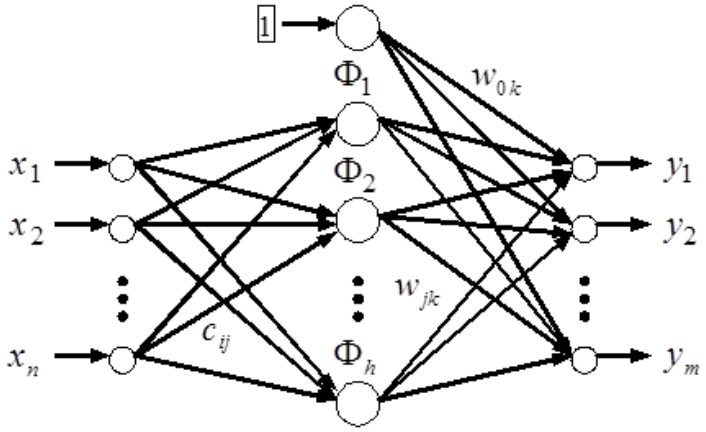
K nevýhodám této metody patří zejména obtížné řešení problematiky lokálních minim. Tyto sítě se také poměrně dlouho učí. Metody, které toto chování zlepšují, pracují například se šumem, s přidáním neuronu nebo úpravou parametru učení.

2.8.2.3 Síť RBF

RBF je zkratkou anglického výrazu Radial Basis Function (v překladu síť radiálních jednotek). Tato síť má pevný počet vrstev a ve své struktuře obsahuje dva typy neuronů:

- Radiální
- Perceptorové (nejčastěji lineární)

Váhy v první vrstvě jsou v RBF sítích nastavovány pevně a další vrstvy se potom učí podobně jako v případě vícevrstvých perceptorových sítí. Hlavní výhodou této struktury je fakt, že se učí velmi rychle. Na následujícím obrázku je vidět její struktura:



Obr. 9 – Struktura RBF sítě

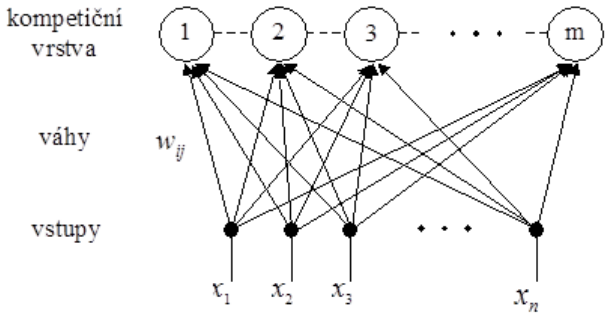
Aktivační funkce RBF sítě je potom:

$$\Phi(\xi) = e^{-\xi^2} \tag{2.51}$$

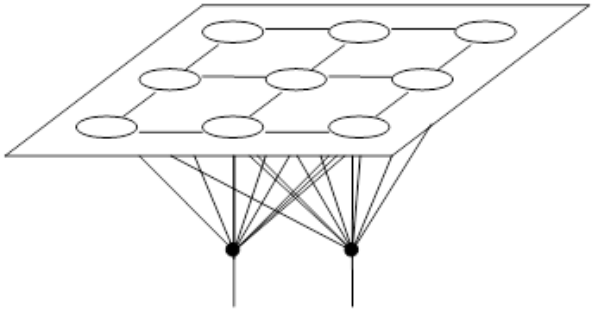
2.8.2.4 Kohonenova síť a Hopfieldova síť

Kohonenova síť je bez učitele a provádí pouze shlukovou analýzu vstupních dat. Obsahuje jednu vrstvu radiálních neuronů. Ty jsou rozmístěny do takzvané mřížky. Síť má několik modifikací a rozšíření například tak, aby byla schopna klasifikace.

a)

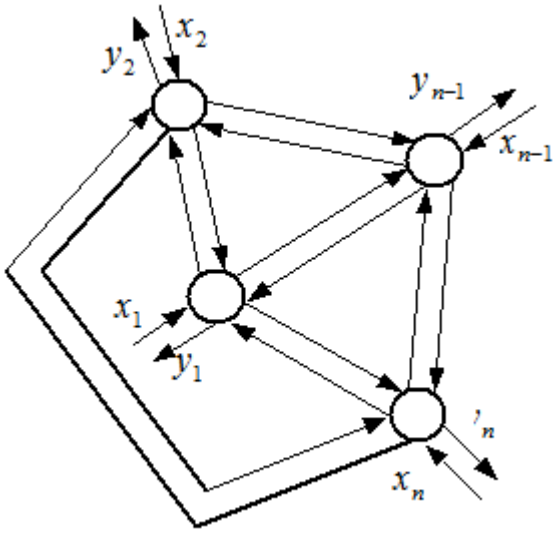


b)



Obr. 10 – Struktura Kohonenovy síť

Hopfieldova síť byla navržena v roce 1982. Jejím hlavním rysem je autoasociativní paměť. Struktura pracuje většinou pouze s binárními hodnotami vstupů a výstupů. Existuje ovšem i spojitá varianta, která se používá pro řešení optimalizačních problémů.



Obr. 11 – Struktura Hopfieldovy sítě

2.8.3 Neuronové sítě v jazykovém modelovaní

Jeden z vůbec prvních pokusů popsat jazyk pomocí umělých neuronových sítí byl představen Jeffem Elmanem. Ten použil rekurentní neuronovou síť pro modelování vět generovaných umělou gramatikou. První opravdu vážný a použitelný pokus ovšem uskutečnil Yoshua Bengio. Ten se pokoušel vytvořit jazykový model na skutečném jazyce. Využil zde poznatky z n-gramové metody a modelu založeném na třídách. Jeho práci následoval Holger Schwenk, který dokázal, že umělé neuronové sítě mají své pevné místo v problematice rozpoznávání řeči a jsou srovnatelné s n-gramovou metodou.

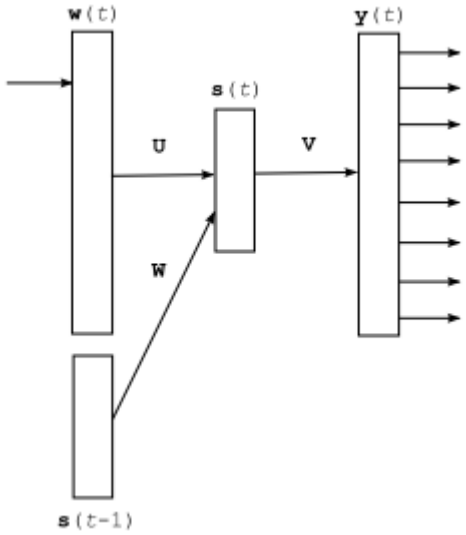
2.8.3.1 Jazykový model založený na dopředné neuronové síti

Tento model odpovídá původní Bengiově navržené metodě. Princip spočívá v tom, že vstup n-gramu neuronové sítě je vytvořen za použití fixní délky historie obsahujících *n-1* slov, kde každé slovo je kódované *1-of-V* kódováním, přičemž *V* je velikost slovníku. Každé z těchto slov je tedy spojeno s vektorem o stejné délce, jako má *V*, kde pouze na stejné pozici, jako je dané slovo ve slovníku je hodnota 1. Všechny ostatní pozice v daném vektoru jsou nulové. Toto kódování tedy přiřazuje každému slovu ortogonální reprezentaci a tu je možné lineárně promítnout na prostor nižší dimenze s použitím sdílené matice *P*. Ta je také nazývána projekční maticí. Matice *P* je sdílena různými slovy historie, což znamená, že tato matice je stejná pro slova w_{t-1}, w_{t-2} atd. V běžném případě by slovník mohl mít zhruba 50 tisíc slov, pro 5-gramový model pak vstupní vrstva sestává z 200 tisíc binárních proměnných, z nichž 4 jsou nastaveny na hodnotu 1, zbytek má hodnotu 0. Projekce se často provádí do 30 dimenzí, čímž pro náš případ dostáváme na vstupu vrstvu s dimenzí $30 * 4 = 120$. Po projekční vrstvě je použita skrytá vrstva s nelineární aktivační funkcí (většinou hyperbolický tangens nebo sigmoid) s dimenzí zhruba 100-300. Velikost výstupní vrstvy je shodná s dimenzí celého slovníku. Poté, co je síť natrénována, výstupní vrstvou je 5-gramová reprezentace pravděpodobnostního rozdělení $P(w_t|w_{t-4}, w_{t-3}, w_{t-2}, w_{t-1})$.

2.8.3.2 Jazykový model založený na rekurentní neuronové síti

Hlavní rozdíl mezi dopřednou a rekurentní neuronovou sítí, je v reprezentaci historie. Zatímco historie dopředné metody je reprezentována několika málo slovy, rekurentní model určuje historii z předchozích kroků během učení neuronové sítě. Skryté vrstvy tak reprezentují nejen n-1 předcházejících slov, ale celou dosavadní historii. Další důležitou výhodou je možnost reprezentovat složitější jevy. Což v podstatě znamená, že slova, která se objevují na různých místech historie, dokáže rekurentní neuronová síť rozpoznat efektivněji než dopředná neuronová síť. Rekurentní neuronová síť jednoduše dokáže uchovat ve skryté vrstvě určitá specifická slova, zatímco dopředná struktura potřebuje pro každou konkrétní pozici slova v historii použít konkrétní parametry. Což snižuje celkové množství parametrů

potřebné pro rozpoznávání slov, ale také robustnost trénovacího korpusu. Na následujícím obrázku je ukázána struktura rekurentní neuronové sítě:



Obr. 12 – Struktura rekurentní neuronové sítě

Vektor *w(t)* reprezentuje slovo w_t a jedná se o vektor zakódovaný metodou 1-of-V. Má tedy dimenzi shodnou s dimenzí slovníku *V* a na všech pozicích má nulu vyjma pozice, na níž se nachází slovo w_t ve slovníku. Vektor *s(t-1)* reprezentuje výstup neuronové sítě v minulém kroku uložený ve skryté vrstvě. Po natrénování této sítě je pak vektor na výstupu *y(t)* (stejné dimenze jako slovník *V*) rozdělením pravděpodobnosti $P(w_{t+1}|w_t, S(t-1))$.

3 Jazyky a gramatika

Gramatiky můžeme dělit několika způsoby. Obecně ale můžeme definovat gramatiku jako soubor logických a strukturních pravidel, pomocí nichž sestavujeme věty z větných členů a slov v určitém jazyce. Jazyk pak můžeme definovat jako znakový systém, pomocí nějž se popisují věci, akce, myšlenky a stavy. Gramatiky se dají dělit například na deterministické či stochastické. Dalším možným dělením je potom například Chomského hierarchie. Nejprve se podíváme na první zmíněné dělení.

3.1 Deterministické gramatiky

Aby bylo možné pracovat s pojmem gramatika, je nutné zavést určité značení a pojmy. Prvním z nich je abeceda, což je neprázdná konečná množina symbolů, z níž budeme skládat slova. Značit jí budeme *V*. Z této abecedy pak budeme tvořit množinu V^+, což je množina neprázdných slov, tedy množina řetězců utvořených z prvků množiny *V* (tyto

řetězce tedy budeme nazývat slova). Zaveďme ještě množinu V^*, což je množina *V* rozšířená o prázdné slovo λ. Jinak je možné tedy říct, že toto slovo neobsahuje žádný symbol z množiny *V*. Můžeme tedy formálně zapsat, že $V^* = V^+ \cup \{\lambda\}$.

Mějme nyní dvě disjunktní abecedy V_N a V_T. V_N je abecedou takzvaných neterminálů a V_T abecedou složenou z terminálních řetězců (slov). Dále pak pro každou gramatiku musí být definována konečná množina substitučních pravidel ρ. Tato pravidla převádějí řetězec slov, který obsahuje neterminál nebo slovo, které je neterminálem, na jiný řetězec slov či jednotlivé terminály a neterminály. Posledním prvkem, který definuje gramatiku je startovací (počáteční) slovo *S* z abecedy neterminálů V_N. Každá jednotlivá gramatika je pak definována čtveřicí:

$$G = (V_N, V_T, \rho, S) \tag{3.1}$$

Soubor substitučních pravidel (někdy také produkčních) je zapsán ve tvaru $\eta \rightarrow \omega$. Tyto řetězce η a ω jsou z abecedy $(V_N \cup V_T)^* = (V_N \cup V_T) \cup \{\lambda\}$. Řetěz η tedy musí obsahovat alespoň jeden neterminál. Tyto řetězce (nebo symboly) jsou v podstatě pomocnými řetězci. Slouží sice pro generování slov, ale ve výsledných slovech a větách se nevyskytují. Ty jsou tedy složeny pouze z terminálních řetězců, které bychom také mohli označit za konečné nebo finální. Prvky abecedy V_N budeme označovat malými latinskými písmeny a V_T velkými písmeny, jejich řetězce potom budeme značit písmeny řecké abecedy. Při generování jazyka začínáme aplikovat pravidlo na startovací řetězec *S*. Neterminální znak v tomto řetězci je nahrazen (substituován) slovem nebo sousledností slov při použití příslušného pravidla. Pokud vzniklý řetězec obsahuje alespoň jeden neterminál, je aplikováno opět vhodné pravidlo na tento neterminál a celý postup se opakuje do chvíle, kdy je výsledný řetězec složen pouze ze samých terminálů.

Mějme tedy dva řetězce μ a ξ, pro něž platí, že μ přímo generuje řetězec ξ, můžeme tento vztah zapsat takto:

$$\mu \Rightarrow \xi \tag{3.2}$$

Kde:

$$\mu = a\eta b, \xi = a\omega b, (\eta \rightarrow \omega) \in \rho \tag{3.3}$$

Přičemž a, b, η, ω ∈ $(V_N \cup V_T)^*$. Můžeme říct, že řetězec ξ je přímo generován řetězcem μ, co formálně můžeme zaspat:

$$\mu \Rightarrow^* \xi \tag{3.4}$$

Existuje-li taková posloupnost řetězců $w^1, w^2, w^3, \dots, w^n, n \geq 1$ taková, že $\mu = w^1, w^1 \Rightarrow w^2 \Rightarrow \cdots \Rightarrow w^n, w^n = \xi$, pak tuto posloupnost nazveme derivačním řetězcem w z řetězce μ. Můžeme také říct, že řetězec ξ je odvozen z řetězce μ.

3.2 Stochastické gramatiky

Stochastické gramatiky jsou velmi podobné, jen každému pravidlu φ: $\alpha \Rightarrow \beta$, kde $\alpha, \beta \in (V_N \cup V_T)^*$ přiřadíme pravděpodobnost $P(\varphi)$. Můžeme pak tedy psát, že:

$$\alpha \overset{P}{\Rightarrow} \beta \tag{3.5}$$

Čtveřici, která definuje každou jednu libovolnou gramatiku, můžeme tedy pro stochastickou variantu upravit následujícím způsobem:

$$G_S = (V_N, V_T, \rho_S, S) \tag{3.6}$$

Pro V_N, V_T, S platí to samé co u gramatiky deterministické, jen množina pravidel ρ_S obsahuje pravidla ohodnocená pravděpodobnostním rozdělením. Pravidlo, které je použito, je voleno náhodně. Odtud vzniklo také alternativní pojmenování stochastické gramatiky, které se jinak říká pravděpodobností gramatika. Platí pro ni, že součet všech pravděpodobností pravidel, které můžeme aplikovat na jeden určitý řetězec obsahující neterminál je roven jedné:

$$\sum_{i=1}^{n} P_i(\gamma) = 1 \tag{3.7}$$

Kde γ je řetězec obsahující neterminál, $P_i(\gamma)$ je jedno z pravidel, která můžeme aplikovat na daný řetězec. Z toho je patrné, že může existovat různé množství pravidel, které můžeme na řetězec použít a také to, že vždycky je některé z pravidel použito.

3.3 Chomského hierarchie

Tato hierarchie třídí gramatiky do 4 typů. Vytvořil ji v roce 1956 americký filosof, lingvista, společenský kritik a logik Noam Chomsky. Rozdělení odpovídá řadě, kdy každá vyšší gramatika (respektive formální jazyk), má vyšší vyjadřovací sílu. Dle Chomského teorie generování některých aspektů jazyka potřebuje komplexnější gramatiku, než generovaní

jiných aspektů. Například pro morfologický model plně postačí syntax a není tedy nutné používat gramatiku vyššího řádu.

3.3.1 Frázová gramatika (typ 0)

Tato gramatika v sobě zahrnuje všechny následující typy. Gramatika 0. typu nemá žádná omezení na tvar substitučních (produkčních) pravidel a generuje právě jazyky, které jsou rozpoznatelné Turingovým strojem, což je teoretický model počítače, vytvořený Allanem Turingem, který můžeme chápat jako počítač, s procesorovou jednotkou (konečný automat) a programu (tvořen pravidly přechodové funkce) a jakési nekonečné „pásky“. Ta má sloužit pro zápis průběžných výsledků.

3.3.2 Kontextová gramatik (typ 1)

Typ 1 je speciálním případem frázové gramatiky. V tomto typu jsou již definována omezení na pravidla ρ. Každé pravidlo z této množiny generující kontextovou gramatiku musí být ve tvaru:

$$\alpha A \beta \rightarrow \alpha \omega \beta \tag{3.8}$$

Kde $\alpha, \beta, \omega, A \in (V_N \cup V_T)^*$, přičemž jsou *A* je nenulový neterminální znak a α, β jsou řetězce libovolných terminálů a neterminálů (mohou být i nulové). A ω je libovolný nenulový řetězec. Tyto jazyky dokáže rozpoznat lineárně ohraničený Turingův stroj.

3.3.3 Bezkontextová gramatika (typ 2)

Tyto gramatiky mají pravidla ve tvaru:

$$A \rightarrow \alpha \tag{3.9}$$

Kde *A* je neterminální znak a α potom libovolný řetězec obsahující terminální a neterminální znaky, jedinou podmínkou je jeho nenulovost. Tyto jazyky pracují s faktem, že neterminál *A* se přepisuje podle daných pravidel nezávisle na kontextu.

3.3.4 Regulární gramatiky (typ 3)

Pravidla regulárních gramatik jsou omezena tím, že na levé straně produkčního pravidla mají vždy jen jeden neterminál. Ten pak generuje buď řetězec terminálů a jeden neterminál, nebo řetězec složený pouze z terminálů. Neterminál může být nahrazen i prázdným slovem. Pravidla této gramatiky jsou buď ve tvaru:

$$A \rightarrow \alpha B \tag{3.10}$$

Nebo:

$$A \rightarrow \alpha \tag{3.11}$$

Následující obrázek ukazuje rozčlenění typů gramatik Chomského hierarchie:

Obr. 13 – Chomského hierarchie

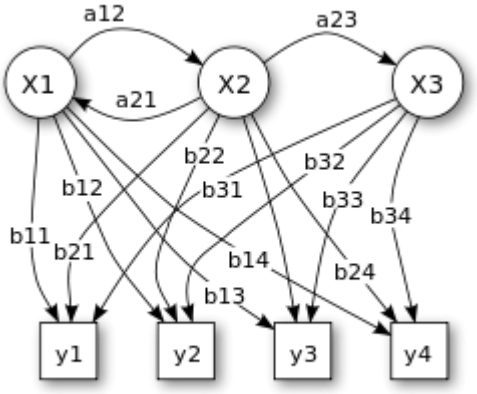
4 Přístup k tvorbě jazykového modelu

Jazykový model můžeme určit několika způsoby. Volbu vhodného postupu pro jeho tvorbu provedeme v závislosti na tom, jak velký trénovací korpus máme. Pokud je korpus dostatečně velký, bude postup při konstrukci jazykového modelu vcelku jednoduchý. Problém ovšem nastává, pokud nemáme dostatečně obsáhlý korpus. Tato problematika bude řešena v následujících kapitolách.

4.1 Skrytý Markovův model s latentní Dirichletovou alokací

4.1.1 Skrytý Markovův model (HMM)

Skrytý Markovův model (anglicky Hidden Markov Model, HMM) je statistický model pro Markovovy procesy se skrytými (nepozorovanými) stavy. Jednou z důležitých vlastností je také fakt, že počet stavů je konečný. Markovův skrytý model můžeme charakterizovat pěti parametry (je tedy možné jej zapsat $G = (Q, V, N, M, \pi)$, kde *G* je Markovův model, *Q* je vektor stavů dimenze *N* ($Q = \{q_1, \dots, q_N\}$), *V* je abeceda výstupních symbolů vektorového kvantizéru, *N* matice přechodu (určuje pravděpodobnost, s níž se přechází z jednoho stavu do jiného), *M* je matice pravděpodobnosti generovaných vzorů, která udává pravděpodobnost, se kterou je generován konečný prvek souboru spektrálních vzorů je-li systém v určitém stavu z množiny *Q*. Vektor π je vektorem pravděpodobnosti počátečních stavů.



Obr. 14 – Skrytý Markovův model

Princip skrytého Markovova modelu můžeme vysvětlit na problému uren. V místnosti, do níž pozorovatel nevidí je *n* uren ($X_1,\ldots,X_n$). V každé z nich je známý počet míčků ($y_1,\ldots,y_k$). V každém kroku je vybrán jeden míček a je předán pozorovateli. Ten tak může sledovat posloupnost tažených míčků. Neví ovšem posloupnost vybraných uren. Volbu urny určuje náhodný výběr startovací urny a pro každý další krok poté *n*-1 vybraná urna (je-li v současnosti taženo v *n*-tém kroce). Samotný skrytý Markovův proces je tedy nepozorovatelný, ale je známa posloupnost výstupů.

4.1.2 Latentní Dirichletova alokace (LDA)

Tato metoda byla popsána pány David Blei, Andrew Ng, a Michael Jordan v roce 2003. Je to jedna z nejpokročilejších a v současnosti také nejvíce používaná metoda pro identifikaci skrytých témat v textu. Její princip využívá dvou multinomického (rozšíření binomického rozdělení do více rozměrů) a Dirichletova (rozšíření beta rozdělení do více rozměrů) rozdělení. Před tím než se provede Dirichletova latentní alokace, je nutné určit *k*. Tato konstanta říká, kolik témat v textu chceme identifikovat. Mějme tedy soubor dokumentů, pro každý z nich (označme jej *i*) jsou pak určeny parametry multinomického rozdělení $\Theta(i)$ z Dirichletova rozdělení s parametry α. Tato rozdělení mají stejné dimenze rovné právě *k*. α je vektor parametrů s reálnými čísly o hodnotě menší než jedna, tento vektor se též nazývá hyperparametr a je identický pro všechny dokumenty *i* souboru všech dokumentů. Tímto postupem tedy nalezneme pravděpodobnostní rozdělení pro všechna témata, ovšem pouze několik málo z nich bude mít nezanedbatelnou pravděpodobnost, to ostatně reflektuje fakt, že každý dokument má pouze několik opakujících se (hlavních) témat.

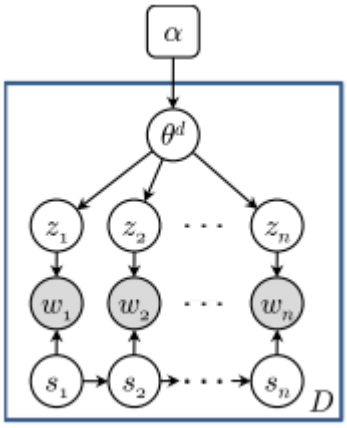
Ve druhém kroku pak pro každé slovo *w* na dané pozici *j* v jednom ze sady dokumentů (dokument *i*) vybereme z Dirichletova rozdělení (toto rozdělení má parametry

Θ(i)) téma (můžeme jej označit *z*). Výběr tohoto tématu je závislý na pozici daného slova a dokumentu, v němž se nachází. Můžeme tedy formálně zaspat, že $z = z(i, j)$.

Pro každou pozici (*i, j*) je použit výběr z multimonického rozdělení $\Phi(z(i, j))$. Toto rozdělení přiřazuje na danou pozici slovo podle tématu. Rozdělení Φ je shodné pro všechny dokumenty a jeho parametry jsou generovány Dirichletovým rozdělením.

4.1.3 HMM-LDA

Tento model, kombinuje výše popsaný skrytý Markovův model a latentní Dirichletovu alokaci pro oddělení syntaktických slov s místními závislostmi podle témat. Důležitou vlastností je i to, že vstupní data, nemusejí být takzvaně „otagovaná“ (označená popisky). HMM-LDA ukládá každé slovo do skryté vrstvy HMM. Každý stav pak generuje slova podle unigramového rozložení. Na následujícím obrázku je patrná činnost tohoto modelu, kde pro každý dokument (zde označený *d*) jsou spočítány váhy θ^d (Dirichletova alokace). Pro každé téma je pak dáno ohodnocení multinomický rozdělením z_i s parametry θ^d a pro každý stav *s* s rozdělením (opět multinomickým s parametry (π^{si-1})). Slovo w_i je pak dáno opět multinomickým rozdělením.



Obr. 15 – HMM-LDA model

4.2 Kombinace neuronové sítě a n-gramového modelu

Tato metoda je popsána v [8]. Tato metoda kombinuje neuronovou síť s trigramem, na nějž bylo použito modifikované Kneser-Neyovo vyhlazování za použití SRILM toolkit nástroje pro jazykové modelování. Dále je použit Janus recognition toolkit, který dokáže aplikovat n-gramový model. Pro rekurentní neuronovou síť je použit rnnlm toolkit.
V následující tabulce je vidět porovnání při použití trigramy s modifikovaným Kneser-Neyovým vyhlazováním (mKN), dopředné neuronové sítě (ff-NN) a kombinace těchto

modelů. V prvním řádku, je počet vět v tisících, symboly a v dalším velikost slovníku. Hodnoty uvedené v této tabulce jsou perplexity pro dané modely.

Sentences	5k	10k	25k	50k	93k
Tokens	33k	65k	156k	320k	584k
Vocabulary	3312	5236	9140	14600	21075
mKN	101.5	106.9	111.0	114.9	116.8
ff-NN	100.6	106.5	109.5	109.5	110.8
ff-NN+mKN	**90.0** (11%)	**94.5** (11%)	**97.9** (11%)	**100.6** (12%)	**102.3** (12%)

Tabulka 1 – porovnání perplexit n-gramu, neuronové sítě a jejich kombinace

5 Návrh vlastní metody pro řídká testovací data

V předchozích kapitolách jsou popsány metody pro práci s trénovacími a testovacími korpusy. Pro reálné aplikace se ovšem vyskytují i takové situace a problémy, pro něž je velmi složité získat dostatečné množství dat. Příkladem takových situací je například rozpoznávání konverzace u lékaře, kde není možné z důvodu ochrany osobních údajů a diskrétnosti vůči pacientům pořizovat nahrávky a přepisy rozhovorů, popřípadě pak rozhovory na poště, či na úřadě. Nicméně konverzace na těchto místech mají často podobnou strukturu a věty v těchto konverzacích používají relativně malý (odborný) slovník. Právě tato myšlenka je použita pro metodu návrhu, pro nedostatečný nebo žádný trénovací korpus. Při spolupráci s odborníky je možné vytvořit generátor gramatiky a slovník, pomocí nichž je pak možné vygenerovat libovolně velký objem dat a na nich pak natrénovat jazykový model. Nadále se tedy budeme věnovat návrhu této metody pro konverzaci s doktorem (konkrétně se jedná o anamnézu pacienta) a rozhovoru na poště při odesílání poštovních zásilek a vyzvednutí doručené zásilky. Úkolem je tedy vytvořit taková produkční pravidla (viz kapitola 3), pomocí nichž je možné charakterizovat tyto konverzace a slovníky, které jsou v těchto případech používány.

5.1 Generátor trénovacích dat

Pro tuto metodu je tedy klíčové správně navrhnout soubor produkčních (substitučních) pravidel, která budou generovat gramatiku. Tato pravidla pracují se třídami slov (tyto třídy také můžeme označit jako slovníková hesla), z nichž pak každou nahradíme slovy z vytvořeného slovníku. Aplikace, kterou tato metoda využívá, tedy bude používat

právě generátor gramatiky a poté metody, které vyberou slovo ze slovníku a nahradí jím třídu generovaného korpusu. Aplikace je vytvořena v jazyce Java ve vývojovém prostředí Eclipse. Jednotlivé části této aplikace jsou popsány v následujících gramatikách.

5.1.1 Generátor gramatiky

Nejprve je nutné zvolit vhodný typ gramatiky. Pro každý druh konverzace je vytvořen soubor pravidel s relativní četnostní jejich použití, přičemž některé věty jsou používány častěji než jiné, ideální volbou bude gramatika stochastická. Z Chomského hierarchie je zvolena bezkontextová gramatika, jejíž pravidla jsou ve tvaru:

$$A \rightarrow \alpha \tag{5.1}$$

Tedy kdy neterminální znak generuje posloupnost slov (terminálů a neterminálů). Tato pravidla jsou zadána pomocí textového souboru, kde na prvním řádku je startovací řetězec a na každém dalším pak jedno substituční pravidlo. Tento textový soubor se musí jmenovat *produkcni_pravidla.txt* a musí být umístěn v kořenovém adresáři aplikace. Každé pravidlo má pak následující tvar:

$$A > \{co\}\{vas\}\{trapit\_3os\}\{Z\}; (15) \tag{5.2}$$

Kde první znak (velké písmeno) je neterminální znak, který generuje řetězec tříd a popřípadě dalších neterminálních znaků. Třídy jsou terminálními znaky, které později budou nahrazeny konkrétními slovy. Každý znak nebo třída jsou uzavřeny do složených závorek. Pravidlo končí středníkem a za ním v kulatých závorkách následuje relativní četnost tohoto pravidla. Aplikace tedy přečte textový soubor s produkčními pravidly a tato pravidla uloží do dvojrozměrného pole, se kterým pak bude dále pracovat. Kdy na první pozici bude vždy neterminální znak a za ním pak následují jednotlivá pravidla, která je možné na tento znak aplikovat.

$$\begin{array}{ll} A(\{co\}\{vas\}\{trapit_{3os}\}\{Z\}; 0.21) \ldots \\ \vdots \qquad\qquad\qquad \vdots \\ Z \qquad\qquad (\{B\}\{C\}, 0.12) \ldots \end{array} \tag{5.3}$$

Relativní četnosti jsou přepočteny na pravděpodobnost použití tohoto pravidla na základě klasické definice pravděpodobnosti:

$$P(A_i) = \frac{r_{A_i}}{\sum_{j=1}^{r} r_{A_j}} \tag{5.4}$$

Kde A_i je konkrétní pravidlo pro neterminální znak *A*. r_{A_i} je relativní četnost pravidla A_i (číslo uvedené v závorce za daným pravidlem) a $\sum_{j=1}^{r} r_{A_j}$ součet relativních četností pro všechna pravidla pro neterminál *A*. Aplikace vezme startovací řetězec ve tvaru:

$$< \{A\}\{rozlouceni\} > \tag{5.5}$$

Algoritmus projde tento řetězec, a pokud je nalezen neterminál (v tomto případě *{A}*), vybere na základě pravděpodobnosti vhodné pravidlo a nahradí jím neterminální znak. Tento krok pak opakuje do chvíle, kdy ve výsledném řetězci jsou pouze terminální znaky. Výsledný řetězec potom uloží do souboru. Na tento řetězec je aplikována metoda pro naplnění slovy.

5.1.2 Naplnění slovy

Slovník pro danou gramatiku je opět textový soubor, umístěný v kořenovém adresáři aplikace a jednotlivá slovníková hesla jsou v následujícím formátu:

$$\{poslat\_inf\}: [\text{"}poslat\text{"}, \text{"}odeslat\text{"}]; \tag{5.6}$$

Ve složených závorkách je název třídy, poté následuje dvojtečka a v hranatých závorkách je pak pole výrazů (slov) pro danou třídu. Může to být i souslednost slov (například pro třídu $\{pozdrav\}$ je jeden z výrazů "$dobrý\ den$"). Všechna tato slova mají stejnou pravděpodobnost výskytu. Aplikace si opět vytvoří dvojrozměrné pole, podobně jako u generátoru gramatik za použití tříd, kde na prvním místě je název třídy následovaný výčtem možných výrazů. Algoritmus projde řetězec terminálních znaků, pro každý najde odpovídající název třídy ve slovníku a náhodně vybere výraz, kterým tento název třídy nahradí. Všechna data jsou pak uložena do textového souboru.

Poznámka: Data jsou generovány bez interpunkce, neboť jejich další zpracování požaduje odstranění interpunkčních znamének.

5.2 Zpracování trénovacích dat

V předchozích kapitolách byl popsán postup pro generování trénovacího korpusu. Ten byl vytvořen na základě pravidel, která byla konzultována s odborníky (lékaři, zaměstnanci pošty). Nyní z těchto vygenerovaných dat vytvoříme n-gramové modely. Nejprve všechna trénovací i testovací data znormalizujeme, aby byla ve shodném tvaru. Pro

jejich vytvoření použijeme sadu nástrojů **SRILM**. Tento nástroj dokáže z trénovacího korpusu vytvořit n-gramový model pomocí příkazu:

$$ngram - count \ - text\ train.txt \ - kndiscount \\ - lm\ lmw\_train \tag{5.7}$$

Kde *train.txt* je soubor s trénovacími daty. Parametr – $kndiscount$ udává metodu pro vyhlazování. V tomto případě se jedná o Kneser-Neyovu metodu. Model vytvořený na základě generovaných dat je skombinován s již existujícími modely pro co nejlepší výsledný model. Jeho kvalita je hodnocena na základě perplexity. Modely jsou kombinovány do jednoho s různými vahami, které jsou vypočteny na základě testovacích dat. Pro výpočet použijeme opět nástroj **SRILM**. Pro výpočet nejlepší kombinace slouží následující příkaz:

$$compute - best - mix\ PPL\_XX1\ PPL\_XX2\ldots \tag{5.8}$$

Výsledkem je vektor vah následujícího tvaru $lambda\ (0.248\ 0.528752\ 4.19e - 05\ 0.0354\ 0.018\ 0.166\ 0.00247\ 6.54e - 05)$. Podle něj jsou pak nastaveny váhy jednotlivých modelů. Pro tvorbu výsledného modelu pak použijeme následující příkaz:

$$ngram\ - lm\ MODELY/lm\_Lekar\ - mix - lm\ MODELY/BH\_lmw\ - \\ mix - lm2\ MODELY/INTE\_vse\_lmw\ldots - lambda\ 0.248\ - mix - \\ lambda2\ 4.19e - 05\ \ldots - write - lm\ HeldOutAnamnezy/Lekar9/ \\ lmw\_Lekar\_FINAL - unk \tag{5.9}$$

Tímto příkazem jsou tedy smíchány modely *lm_Lekar, BH_lmw, INTE_vse_lmw*, a tak dále s vahami, které zadáváme ve stejném pořadí jako modely. *Lambda* bez indexu odpovídá prvnímu modelu, *lambda2* pak přísluší v pořadí třetímu modelu. Váha druhého modelu (v tomto případě *BH_lmw*) je dopočítána podle vzorce $lambda1 = 1 - \sum_{i=0, i\neq 1}^{n} lambda_i$pro n zadaných vah *lambda*.

Abychom dosáhli co nejlepšího výsledku, použijeme metodu křížové ověřovací techniky (cross-validation). Testovací data tedy rozdělíme na deset částí. Pro každou z nich pak vyhodnotíme váhy pro kombinaci jazykových modelů a výsledný model pak otestujeme na zbytku dat. Tento postup opakujeme desetkrát. Dostáváme tedy deset jazykových modelů a pro každý z nich také spočtenou perplexitu. Abychom dostali nejlepší model, použijeme tyto perplexity pro jednotlivé odložené části a pomocí nástroje **SRILM** spočteme nejlepší váhy pro tyto dílčí kombinované jazykové modely. Pomocí těchto vah pak tyto modely

zkombinujeme a dostáváme výsledný jazykový model. Ten pak otestujeme na celém korpusu testovacích dat a vyhodnotíme jeho perplexitu. Tento postup opakujeme jednou pro kombinaci jazykovým modelů včetně modelu vytvořeného z generované gramatiky a jednou bez použití tohoto modelu. Takto získáme dvě sady výsledků, které poslouží pro zhodnocení přínosu použití metody s generovanou gramatikou.

6 Závěr a vyhodnocení výsledků experimentů

Jazykový model byl vytvořen pro dvě různé situace. První z nich je rozhovor pacienta s lékařem a druhou je podání či vyzvednutí poštovní zásilky na pobočce. Následující tabulky ukazují spočtené váhy pro jednotlivé kroky metody křížové validace (cross validation) a jejich perplexity testované na zbytku testovacích dat bez odložené části.

	lmw_Lekar	BH_lmw	INTE_vse_lmw2_2_3	IVY_vse_lmw2_2_3	MF_vse_lmw2_2_3	SUB_lmw2_2_3	TISK_vse_lmw2_2_3	TVR_vse_lmw2_2_3	Perplexita
0	0,100158	0,202276	1,49E-05	0,332749	0,030417	0,319336	0,011438	3,61E-03	659,999
1	0,06685	0,236796	6,78E-06	0,347498	0,198236	0,136558	0,013212	8,42E-04	688,311
2	0,05257	0,219964	2,64E-03	0,204239	0,149706	0,316889	0,031785	2,22E-02	665,985
3	0,014838	0,165734	1,83E-04	0,15275	0,128585	0,421914	0,108533	7,46E-03	687,503
4	0,065184	0,273372	1,06E-02	0,132787	0,189059	0,253206	0,044256	3,16E-02	696,407
5	0,000363	0,014946	2,93E-02	0,136389	0,001661	0,69149	0,125628	1,94E-04	750,868
6	0,061616	0,079747	3,38E-03	0,038223	0,130661	0,613939	0,03328	3,92E-02	579,868
7	0,206417	0,149954	1,79E-05	0,270164	0,076048	0,266961	0,003411	2,70E-02	657,46
8	0,116471	0,189381	5,95E-06	0,227808	0,165448	0,263425	0,019726	1,77E-02	676,965
9	0,072489	0,188522	1,75E-06	0,259075	0,131979	0,266274	0,069774	1,19E-02	637,242

Tabulka 2 – Váhy a perplexity pro model konverzace u lékaře

	BH_lmw	INTE_vse_lmw2_2_3	IVY_vse_lmw2_2_3	MF_vse_lmw2_2_3	SUB_lmw2_2_3	TISK_vse_lmw2_2_3	TVR_vse_lmw2_2_3	Perplexita
0	0,20875	8,73E-06	0,356325	0,029326	0,393282	0,0093	3,01E-03	733,577
1	0,245728	6,90E-06	0,346004	0,196199	0,198195	0,012882	9,84E-04	755,883
2	0,219946	3,15E-03	0,206235	0,137975	0,374211	0,036292	2,22E-02	736,231
3	0,16996	8,40E-05	0,152521	0,130152	0,435923	0,10586	5,50E-03	735,178
4	0,27554	1,71E-02	0,130157	0,194036	0,309116	0,042437	3,16E-02	778,847
5	0,015571	2,97E-02	0,135931	0,001662	0,691272	0,12567	1,90E-04	760,105
6	0,094752	3,95E-03	0,03329	0,136285	0,658723	0,035734	3,73E-02	646,508
7	0,151091	9,51E-06	0,342957	0,086538	0,382906	0,002868	3,36E-02	717,614
8	0,180723	1,52E-05	0,263044	0,174212	0,345054	0,021253	1,57E-02	727,167
9	0,187841	1,96E-06	0,25623	0,132673	0,335998	0,074996	1,23E-02	697,581

Tabulka 3 – Váhy a perplexity pro model konverzace u lékaře bez použití generovaného modelu lmw-Lekar

	lmw_Posta	BH_lmw	INTE_vse_lmw2_2_3	IVY_vse_lmw2_2_3	MF_vse_lmw2_2_3	SUB_lmw2_2_3	TISK_vse_lmw2_2_3	TVR_vse_lmw2_2_3	Perplexita
0	0,542381	0,061053	4,68E-07	0,014503	2,16E-03	0,286618	5,90E-05	9,32E-02	328,603
1	0,342017	0,186126	6,16E-05	0,078429	0,002543	0,317134	9,00E-03	0,064693	338,519
2	0,231505	0,151426	1,07E-04	0,033647	2,23E-03	0,175931	2,58E-01	0,146836	310,279
3	0,257551	0,06937	6,96E-03	0,281615	0,013088	1,76E-05	3,24E-01	0,046911	342,649
4	0,261857	0,153924	1,13E-03	0,107544	5,59E-02	0,077814	1,64E-01	0,177459	294,226
5	0,566893	0,134766	1,52E-05	0,075564	0,000616	0,064187	4,03E-03	0,153927	331,702
6	0,467932	0,089833	1,07E-03	0,119649	9,51E-03	0,049753	6,73E-02	0,19491	380,088
7	0,26997	0,083986	5,36E-07	0,076992	0,000492	0,479019	8,79E-05	0,089454	372,433
8	0,284203	0,154468	1,54E-05	0,096822	3,86E-02	0,329788	1,56E-02	0,080507	308,211
9	0,137484	0,332123	1,63E-03	0,187583	0,032886	0,004126	1,40E-02	0,290212	331,941

Tabulka 4 – Váhy a perplexity pro model konverzace na poště

	BH_lmw	INTE_vse_lmw2_2_3	IVY_vse_lmw2_2_3	MF_vse_lmw2_2_3	SUB_lmw2_2_3	TISK_vse_lmw2_2_3	TVR_vse_lmw2_2_3	Perplexita
0	0,205781	6,64E-05	0,070074	1,46E-02	0,61611	9,40E-04	9,24E-02	798,291
1	0,222013	6,13E-04	0,16038	0,002194	0,495215	2,87E-02	0,090847	740,056
2	0,20004	1,39E-04	0,025639	1,12E-03	0,314283	3,12E-01	0,147156	691,994
3	0,190467	7,04E-03	0,341377	0,003691	0,000487	4,03E-01	0,054317	870,122
4	0,168559	1,52E-02	0,089261	7,42E-02	0,295693	1,99E-01	0,157768	674,553
5	0,149976	2,25E-02	0,141194	0,000896	0,331698	1,21E-01	0,232279	749,196
6	0,135565	9,02E-03	0,125657	7,35E-03	0,187038	3,03E-01	0,2319	781,394
7	0,098639	9,10E-07	0,098761	0,002071	0,70391	6,83E-05	0,09655	806,786
8	0,203839	1,85E-05	0,109626	4,48E-02	0,530586	2,52E-02	0,085938	753,683
9	0,383347	2,33E-03	0,183336	0,023382	0,083974	1,63E-02	0,307311	744,704

Tabulka 5 – Váhy a perplexity pro model konverzace na poště bez použití generovaného modelu lmw-Posta

Pro každý ze čtyř případů (model s použitím modelu generovaného vytvořenou aplikací pro konverzaci u lékaře, model bez použití generovaného modelu a obdobně pak pro poštu) tak vznikne 10 jazykových modelů. Pro každý z nich je spočtena perplexita. Na základě této perplexity, obdobně jako u míchání dílčích modelů, jsou nalezeny váhy pro nejlepší možnou kombinaci všech deseti jazykových modelů. Tyto váhy, z nichž je následně počítán výsledný jazykový model, vycházejí takto:

	0	1	2	3	4	5	6	7	8	9
S použitím lmw-Lekar	0,2602	0,3214	0,0394	0,2536	0,0009	0,0074	0,0951	0,0050	0,0022	0,0143
Bez použití lmw-Lekar	0,2819	0,3242	0,0301	0,3100	0,0006	0,0103	0,001	0,0066	0,0088	0,0260
S použitím lmw-Posta	0,2509	0,2853	0,2278	0,1081	0,0450	0,0524	0,0067	0,0033	0,0094	0,0106
Bez použití lmw-Posta	0,2603	0,2872	0,2298	0,1195	0,0458	0,02	0,0119	0,0038	0,0108	0,0105

Tabulka 6 – Váhy pro jednotlivé modely trénované na odložených datech křížové validaci pro smíchání do výsledného jazykového modelu

Výsledné jazykové modely pro jednotlivé konverzace (u lékaře, na poště) jsou pak otestovány na celém testovacím korpusu. Následující tabulka ukazuje vliv generovaného jazykového modelu na snížení perplexity výsledného kombinovaného modelu:

	Jazykový model	Perplexita
Konverzace u lékaře	s generovaným modelem	590,796
	bez generovného modelu	593,066
Konverzace na poště	s generovaným modelem	286,048
	bez generovného modelu	654,57

Tabulka 7 – Výsledné perplexity kombinovaného modelu s použitím modelu generovaného aplikací a bez něj pro obě témata

6.1 Závěr

Je patrné, že perplexita se v obou případech (při použití generovaného modelu) snížila. Větší přínos mělo přidání jazykového modelu založeného na generované gramatice v poštovních konverzacích, kde je jednak méně robustní slovník a také jsou si promluvy svou strukturou navzájem více podobné. Pro konverzaci u lékaře je vliv použití generovaného modelu výrazně nižší, neboť promluvy mezi pacientem a lékařem jsou mnohem různorodější a při hledání příčin onemocnění a tvorbě anamnézy je řešena výrazně širší škála témat. I slovníky pro tuto problematiku jsou obsáhlejší. Lépe by model dopadl, pokud by byla oblast konverzace zúžena například pouze na rozhovor s jedním ze specialistů (například kardiologem, ortopedem apod.). Nicméně tato metoda pro velmi úzké tematické okruhy a druhy konverzací zlepšuje kvalitu jazykového modelu, má ovšem několik zásadních úskalí.

Soubor produkčních pravidel pro generovanou gramatiku i slovník jsou poměrně obsáhlé i pro velmi konkrétní konverzace. Je totiž třeba pokrýt všechny možné větné stavby, včetně těch, které neodpovídají spisovné češtině, ale při reálných situacích je možné se s nimi setkat.

Při tvorbě produkčních pravidel i slovníků je navíc třeba velká znalost situací a konverzací, pro něž je jazykový model vytvářen. To znamená rozsáhlou spolupráci s odborníkem (v ideálním případě odborníky). Například pro tvorbu lékařské anamnézy při příjmu pacienta, lékař používá dotazy na nynější onemocnění, dřívější onemocnění, zranění, operace, dále jsou dotazy směřovány na rodinu, zaměstnání i například stravovací návyky pacienta. Tato široká škála témat při rozhovoru lékaře s pacientem výrazně rozšiřuje potřebu robustnosti slovníku i pravidel pro generovanou gramatiku.

Situaci komplikuje i složitost českého jazyka. Překážkou této metody je ohebnost slov. Pokud by byla slova generována například v anglickém jazyce, slovník i soubor pravidel by byly významně méně obsáhlé. Odpadly by totiž všechny pády podstatných jmen, všechny osoby sloves a také přechýlené varianty přídavných a podstatných jmen. Velikost slovníku i

produkčních pravidel pro simulaci reálných promluv se tak se složitostí (ohebností) jazyka mohutně narůstá.

Metoda tedy bude mít tím lepší výsledky, čím specifičtější bude konverzace, pro níž je generován trénovací korpus a zlepšovat se bude i s množstvím produkčních pravidel a velikostí a odborností použitého slovníku.

7 Použitá literatura

[1] Ing. Jindra Drábková, „Tvorba jazykového modelu založeného na třídách“, Autoreferát dizertační práce, *Technická univerzita v Liberci,* 2005

[2] Michal Richter, „N-gramový jazykový model pro český spellchecker“, Bakalářská práce, *Univerzita Karlova v Praze,* 2008

[3] Petr Salajka, „Prediktivní psaní textů“, Diplomová práce, *Západočeská univerzita v Plzni,* 2013

[4] Ing. Dana Nejedlová, „Tvorba slovníků a jazykových modelů pro automatický přepis zpravodajských pořadů“, Autoreferát disertační práce, *Technická univerzita v Liberci*, 2004

[5] Filip Orság, „Rozpoznávání samohlásek“, *UIVT FEI VUT Brno*

[6] Ing. Tomáš Mikolov, „STATISTICAL LANGUAGE MODELS BASED ON NEURAL NETWORKS“, *Vysoké učení technické v Brně*, 2012

[7] Bo-June (Paul) Hsu, „Language Modeling for Limited-Data Domains“, *Department of Electrical Engineering and Computer Science - MASSACHUSETTS INSTITUTE OF TECHNOLOGY*, 2009

[8] Ankur Gandhe, Florian Metze, Ian Lane, „Neural Network Language Models for Low Resource Languages“, *LTI, Carnegie Mellon University*

[9] Ping Xu, „Cross-lingual language modeling for low-resource speech recognition“, *The Hong Kong University of Science and Technology,* 2012

[10] Mark J. F. Gales, Kate M. Knill, Anton Ragni and Shakti P. Rath, „SPEECH RECOGNITION AND KEYWORD SPOTTING FOR LOW RESOURCE LANGUAGES: BABEL PROJECT RESEARCH AT CUED“, *Cambridge University Engineering Department Trumpington Street, Cambridge, CB2 1PZ, UK*

[11] Roman Barták, „Automaty a gramatiky“, *Univerzita Karlova v Praze*

[12] Mirko Novák, „Neuronové sítě a informační systémy živých organismů“; *Grada Diplomové práce PU a MU*, 2011

[13] Stanley F. Chen and Joshua Goodman, „An Empirical Study of Smoothing Techniques for Language Modeling“, *TR-10-98 August,* 1998

[14] Jiří Valíček, „Jazykové modely pro rozpoznávání řeči v různých tematických oblastech“, Bakalářská práce, *České vysoké učení technické v Praze*, 2014

[15] Ing. Jan Hoidekr, „Metody redukce OOV ve statistických jazykových modelech založených na třídách“, *Západočeská univerzita v Plzni,* Disertační práce, 2012

[16] Ing. Karol Molnár, „Ú vod do problematiky umělých neuronových sítí“, ÚTKO FEI VUT, 2000

[17] RNDr. Jiří Klaška, Dr., „Vázané a globální extrémy“, *ÚM FSI v Brně*, 2006

[18] Josef Psutka, Jindřich Matoušek, Luděk Müller, Vlasta Radová, „Mluvíme s počítačem česky“, ISBN- 80-200-1309-1, EAN- 9788020013095, *Academia*, 2006

[19] Ing. Michal Řepka, „Základní popis umělé vrstvené neuronové sítě“, *Hornicko-geologická fakulta, Institut ekonomiky a systémů řízení,* 2007