Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Diplomová práce

Metody statistické sémantické analýzy

Plzeň 2016 David Steinberger

Místo této strany bude zadání práce.

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 23. června 2016

David Steinberger

Poděkování

Děkuji především vedoucímu diplomové práce Ing. Miloslavu Konopíkovi, Ph.D. za výborné vedení práce a všem anotátorům, kteří se podíleli na vývoji nové testovací datové sady pro český jazyk.

Abstract

The thesis deals with statistic semantic similarity focused on the word2vec tool. It introduces extensions for the Czech language based upon stemming and character n-grams. The achieved results improve the original tool by 12% on the Czech language and by 3% on English. The new model is providing good results even on small training data. In this thesis, we introduce two new training corpora and one large dataset based on similarity of word pairs. The dataset is compiled from 9 differenet sources, it contains words in their contexts, it distinguishes between the similarity and relatedness of the word pairs. The final inter-rater agreement reaches 0.81 correlation, which is fully comparable with english datasets.

Abstrakt

Tato práce se zabývá statistickou sémantickou podobností a zaměřuje se na nástroj word2vec. Byla navržena rozšíření s ohledem na český jazyk založená na stemmování a n-gramech znaků. Výsledky této práce podávají na českém jazyce o 12% lepší výsledky než původní model. Na anglickém jazyce bylo dosaženo zlepšení o 3%. Nový model poskytuje dobré výsledky i při velmi malém množství trénovacích dat. V rámci práce byly vytvořeny dva trénovací korpusy a jedna obsáhlá testovací datová sada založená na podobnosti dvojic slov. Sada byla získána z 9 různých zdrojů dvojic slov, obsahuje slova v kontextech, odlišuje podobnost a souvislost slov. Výsledná mezi anotátorská shoda dosáhla korelaci 0,81, která je plně srovnatelná s anglickými datovými sadami.

Obsah

1 Úvod

Sémantická analýza slov je jedna z úloh zpracování přirozeného jazyka (NLP[1]). NLP se zabývá převodem lidského přirozeného jazyka do takové podoby, která je zpracovatelná počítači. Tato práce je zaměřena na sémantickou podobnost, která je jednou z podúloh sémantické analýzy slov a prokázala se jako důležité vylepšení mnohých dalších NLP úloh, jakými jsou rozpoznávání pojmenovaných entit (named entity recognition)[41], zpracování závislostí (dependency parsing)[22] nebo strojový překlad (machine translation)[31].

Nejnovější vědecké výsledky sémantické podobnosti slov využívají statistický přístup a neuronové sítě[3, 29]. Tyto přístupy mapují slova do vektorového prostoru tak, že podobná slova se nacházejí u sebe a je možné je porovnávat. Mezi nejpokročilejší nástroje patří *word2vec* předvedený Mikolovem a kol.[30], který využívá jednoduchosti, díky níž je možné trénovat síť na značně větších trénovacích datech. Tato práce se zabývá vylepšením této metody a využitím na českém jazyce.

Některá známá rozšíření *word2vec* se zabývají změnou kontextu[21], využítím pozice v kontextu[22] nebo dvojjazyčným modelem[45]. Bohužel všechna tato rozšíření jsou testována na anglickém jazyce. V souvislosti s *word2vec* a českým jazykem se zabývají práce [7, 31], avšak nevěnují se samotnému využití metody na českém jazyce a pouze metodu využívají jako příznak k dalšímu zpracování.

K porovnání dosažených výsledků je v současné době pro češtinu dostupná pouze jedna referenční datová sada založená na analogii slov[43] a přeložená *Rubestein-Goodenough datová sada*[17], která je pro výpočet výsledků příliš malá. Cílem této práce je také vytvořit dostatečně velkou českou datovou sadu založenou na podobnosti slov.

V následující kapitole jsou zmíněny některé další metody sémantické podobnosti slov, podrobně popsána metoda *word2vec* a referenční anglické testovací datové sady. V kapitole *Analýza* jsou navržena možná vylepšení metody *word2vec* a zvážen způsob jejich evaluace. Kapitola *Výsledky* popisuje implementaci těchto rozšíření a tvorbu českýho trénovacích korpusů a testovací datové sady. Kapitola *Výsledky* obsahuje informace o konfiguraci parametrů nemodifikovaného algoritmu na nově vytvořeném českém korpusu a výsledky předvedených vylepšení algoritmu. V závěru jsou zhodnoceny dosažené výsledky.

[1] Z anglického Nature language processing.

2 Teorie

2.1 Sémantická podobnost

Nejprve je nutné upřesnit rozdíl mezi sémantickou *podobností* a *souvislostí*. Podobnost slov určuje, jak velký význam daná slova sdílejí. Čím větší je podobnost, tím větší je pravděpodobnost, že při záměně slov bude věta stále dávat stejný smysl. Na druhou stranu souvislost slov určuje pravděpodobnost, že se daná slova mohou vyskytnout ve stejném kontextu.

Podobnost	Souvislost
auto - automobil	auto - silnice
brusle - boty	brusle - hokej
voda - kapalina	kapalina - laboratoř
král - šlechtic	král - hrad
čepice - klobouk	klobouk - věšák

Tabulka 2.1: Příklady podobných a souvislých slov.

Například ve větě: *Po nádvoří se procházel král.* je možné zaměnit slovo *král* podobným slovem *šlechtic* a věta dává téměř stejný smysl jako věta původní. Ale v případě záměny se souvislým slovem *hrad*, nedává věta žádný smysl. Je však zřejmé, že slova *král* a *hrad* se mohou s velkou pravděpodobností vyskytnout ve větě vedle sebe.

2.2 Metody sémantické podobnosti slov

Metody je možné na základě jejich realizace rozdělit na metody založené na apriorní znalosti nebo na metody statistické, založené na distribuční hypotéze.

2.2.1 Manuální ontologie

Tento přístup je založen na ručně vyráběných databázích. Výroba dostatečně velkého množství označkovaných slov je velmi pracná a časově náročná. Největší lexikální databází je jednoznačně *WordNet*. Tato databáze je tvořena *slovními tvary (word forms)*, které mohou mít různé *významy (senses)*. Jeden *tvar* může mít více *významů*, avšak spojení právě jednoho *tvaru* s právě

jedním *významem* je vždy unikátní. *WordNet* podporuje syntaktické kategorie v podobě podstatných jmén, sloves, přídavných jmén a příslovcí. Ostatní slovní druhy nejsou zahrnuty. Dále jsou pro jednotlivé tvary definovány jejich sémantické vazby (synonyma, hyponyma, hypernyma atd.), přičemž nejdůležitější a základní vazbou je synonymum. Množina synonym tvoří tzv. *synset*, který definuje právě jeden sémanticky stejný *význam*[34]. V současné verzi 3.1. je obsaženo 155 287 slov zorganizovaných do 117 659 synsetů.

2.2.2 Automatické ontologie

Automatické ontologie se zbavují největšího nedostatku ručně vyráběných databází v podobě manuálního rozšiřování, a to za cenu větší nepřesnosti. Ale díky automatickému rozšiřování obsahují značně větší množství dat. Nejznámějšími zástupci jsou *DBPedia*[20], která používá data vyextrahovaná z *Wikipedie*[1] nebo *YAGO*[42], která ještě tato data kombinuje s *WordNetem*.

2.2.3 Statistické metody

Tyto metody jsou plně automatizované a význam slov se učí na základě distribuční hypotézy. Již v roce 1954 Harris na základě této hypotézy dokázal, že slova v podobných kontextech mají podobný význam[14]. Základním přístupem těchto metod je mapování slov do mnoharozměrného vektorového prostoru[44]. Na základě kontextu se vytváří vektory tak, že slova v podobných kontextech mají podobné vektory.

Díky tomu lze podobnost lehce matematicky vyjádřit pomocí podobnosti samotných vektorů, jako například *euklidovskou vzdáleností* nebo *city block vzdáleností*. Nejpoužívanější mírou podobnosti v sémantických prostorech je tzv. *kosínová podobnost*[44], která měří podobnost mezi vektory jako kosínus úhlu mezi nimi:

$$cos(x, y) = \frac{x \cdot y}{||x|| \cdot ||y||} = \frac{\sum_{i=1}^{n} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{n} x_i^2 \cdot \sum_{i=1}^{n} y_i^2}} \tag{2.1}$$

HAL

Hyperspace Analogue to Language (HAL)[24] je založen na velice jednoduchém principu. Pro každé slovo trénovací množiny je prozkoumáno jeho fixně

[1] https://www.wikipedia.org

dané okolí, tzv. *okno*. Slova v okně jsou váhovány podle vzdálenosti od aktuálně zkoumaného slova. Blíže postavená slova mají větší vliv na sémantiku než dále postavená.

Pomocí dříve zmiňovaných oken se vytvoří $|W| \times |W|$ matice sousednosti M, kde W je velikost slovníku[2]. Jestliže slovo w_j je nalezeno v okolí slova w_i, přičte se hodnota váhové funkce $f(\Delta k)$ na pozici m_{ij}. Řádkové a sloupcové vektory M nesou informaci o sousednosti jednotlivých slov. Obecně se jedná o velice řídkou matici, protože v okolí slova se většinou nachází poměrně malé množství slov vzhledem k velikosti slovníku.

Matematicky lze vyjádřit M následujícím způsobem:

$$m_{i,j} = \sum_{k=1}^{|T|} \begin{cases} \lambda & t(k) = a \wedge b \in \{t(k + \Delta k); \forall \Delta k\} \\ 0 & \end{cases} , \tag{2.2}$$

kde λ je výsledek váhové funkce, T trénovací korpus, $t(k)$ je slovo korpusu s indexem k a Δk je velikost okna. Váhovou funkci lze zvolit libovolně na základě požadavků řešené úlohy. Nejjednoduší váhová funkce může být rovna inverzní vzdálenosti právě zkoumaných slov:

$$f(x) = \frac{1}{|x|} \tag{2.3}$$

LSA

Latent Semantic Analysis (LSA)[19] je metoda vytvářející vektorovou reprezentaci slova na základě výskytu slova v určitém kontextu. Za kontext je možné považovat odstavec, dokument, sadu dokumentů nebo jakýkoliv libovolný celek.

V prvním kroku se dostupný text převede na matici, kde každá řádka reprezentuje právě jedno unikátní slovo ze slovníku a každý sloupec zvolený kontext. Na každé pozici v matici je uložena informace kolikrát se slovo dané řádkou objevilo v kontextu daném sloupcem. Tato frekvence může být váhována předem danou funkcí, která vyjadřuje důležitost v daném kontextu nebo obecně v nějaké doméně. Známým váhováním je například *tf-idf*[23], které bere v úvahu frekvenci v jednom určitém kontextu a inverzní frekvenci napříč všemi kontexty.

V dalším kroku se provede rozklad na singulární čísla[23] (SVD[3]) a redukuje se dimenzionalita, čímž dochází ke generalizaci modelu a vytvoření vztahů mezi slovy. Právě volba této konečné dimenze určuje vlastnosti výsledného modelu.

[2]Za slovník se považuje množina unikátních slov korpusu.

[3]Z anglického Singular Value Decomposition.

GloVe

Globální vektory (Global Vectors - GloVe)[37] je jedna z posledních prezentovaných metod. V první řadě se vytvoří matice sousednosti stejně jako v HAL s inverzní vzdáleností jako váhovou funkcí. Jádrem metody je tzv. log-bilineární regrese, která při učení modeluje vektory slov tak, aby jejich skalární součet byl roven podmíněné pravděpodobnosti těchto slov získané z matice sousednosti.

I přesto, že autoři této metody naměřili v některých případech téměř o 11% lepší výsledky než podává *word2vec*, jsou tyto výsledky zpochybňovány. Podle Y. Goldberga je zlepšení pouze o 2-3%[12] a podle R. Řehůnka dosahuje *word2vec* stále lepších výsledků, nehledě na paměťovou a časovou náročnost[49].

2.3 Word2vec

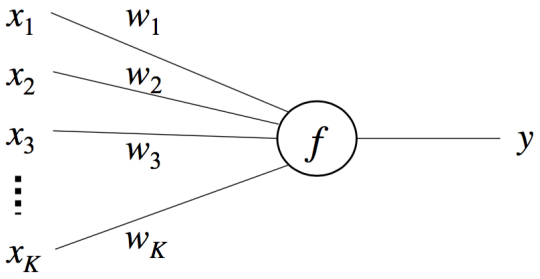
Metoda *word2vec*[13, 30, 32, 39] a její rozšíření jsou hlavním cílem této práce. Tato metoda byla vybrána na základě její popularity, značné dokumentace, volně dostupného kódu, zamlouvajícím se výsledkům a neprouzkoumaným možnostem na českém jazyce (viz kapitola *3 Analýza*).

Jak již bylo řečeno, *word2vec* využívá jednoduchosti za účelem využití velkého množství nestrukturovaných trénovacích dat. Zjednodušený a optimalizovaný kód umožňuje na jednom počítači natrénovat až 1,6 miliardy slov při dimenzi vektoru až 300 za pouhý jeden den. Žádné z předchozích známých řešení nebylo schopné natrénovat více než několik set miliónů slov s velikostí vektoru okolo 50-100[30]. Díky takto natrénovaným vektorům je možné porovnávat podobnost, ale navíc tyto vektory nesou informaci o některých lingvistických jevech a vzorech.

Jedním z těchto lingvistických jevů je analogie slov (word analogy). Například výsledný vektor $vec(''Čechy'') - vec(''Praha'') + vec(''Anglie'')$ je blíže vektoru $vec(''Londýn'')$ než jakémukoliv jinému slovnímu vektoru[33].

Tento přístup vychází z architektury jazykových modelů[4] založených na neuronových sítích (NNLM[5]) prezentovaných v [26, 28], kde jsou slovní vektory naučeny nejprve pomocí jednoduché neuronové sítě s jednou skrytou vrstvou a poté jsou vektory použity pro trénování složité sítě z [3]. Nástroj *word2vec* poskytuje dva různé modely sítě, dva různé učící přístupy a celé spektrum konfigurace. Všechny tyto možnosti jsou popsány v následujících kapitolách.

[4]Metoda *word2vec* je také jazykový model, jehož výstupem jsou zároveň slovní vektory.
[5]Z anglického Neural network language model.



Obrázek 2.1: Ukázka umělého neuronu.

2.3.1 Úvod do neuronových sítí

Jelikož základem *word2vec* jsou neuronové sítě (neural networks)[39], je nutné tento pojem definovat. Neuronová síť je matematický model inspirovaný biologickými neuronovými sítěmi. Tato síť se skládá z množiny tzv. *umělých neuronů*, které jsou uspořádány do jednotlivých vrstev a mají mezi sebou vztahy, které se nazývají *synapse*. Jednotlivé synapse jsou určeny váhami, které se během učení sítě mění.

Na obrázku 2.1 je zobrazen samotný neuron. $\{x_1, \ldots, x_K\}$ jsou vstupní hodnoty, $\{w_1, \ldots, w_K\}$ jednotlivé váhy, y skalární výstup a f je aktivační funkce (někdy také nazývána jako spojovací, rozhodovací nebo přenosová funkce). Neuron poté funguje na základě aktivační funkce:

$$y = f(u), \tag{2.4}$$

kde u je skalární číslo, které je vstupem neuronu definováno jako:

$$u = \sum_{i=0}^{K} w_i x_i \tag{2.5}$$

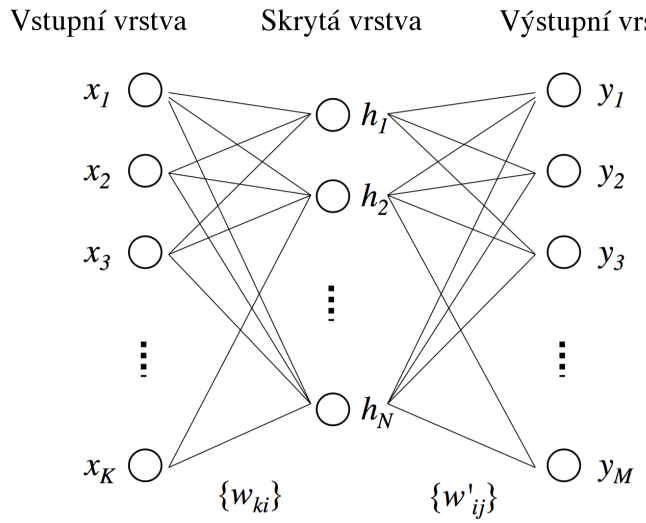
Pomocí vektorů vyjádřeno jako:

$$u = \mathbf{w}^T \mathbf{x} \tag{2.6}$$

Jako aktivační funkce může být zvolena například logistická funkce, definována jako:

$$f(u) = \frac{1}{1 + e^{-u}} \tag{2.7}$$

Tato funkce se často volí kvůli jejím dvěma důležitým předpokladům: (1) výstup y je vždy v intervalu mezi 0 a 1, a (2) funkce je spojitá a snadno diferencovatelná.



Obrázek 2.2: Ukázka neuronové sítě s jednou skrytou vrstvou.

Každý neuron s aktivační funkcí se nazývá perceptron. Učící algoritmus pro *perceptron* je perceptronový algoritmus. V každém kroku aktualizuje hodnotu vah následovně:

$$\mathbf{w}^{new} = \mathbf{w}^{old} - \eta \cdot E, \tag{2.8}$$

kde E je chybová funkce, kterou je nutné definovat a při trénování minimalizovat. Vyhovující funkce může vypadat následovně:

$$E = \frac{1}{2}(t - y)^2, \tag{2.9}$$

poté tuto funkci E zderivujeme podle w_i:

$$\frac{\partial E}{\partial w_i} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial u} \cdot \frac{\partial u}{\partial w_i} \tag{2.10}$$

$$= (y - t) \cdot y(1 - y) \cdot x_i \tag{2.11}$$

Jakmile máme derivaci můžeme aplikovat stochastický gradientní sestup:

$$\mathbf{w}^{new} = \mathbf{w}^{old} - \eta \cdot (y - t) \cdot y(1 - y) \cdot \mathbf{x} \tag{2.12}$$

2.3.2 Continuous bag-of-words

Continuous bag-of-words (CBOW)[30] je jeden z předvedených modelů metody *word2vec*. Snaží se na základě krajních slov v kontextu určit slovo prostřední. Tento model se nazývá bag-of-words, protože nebere ohled na pořadí slov v okně a toto pořadí neovlivňuje samotnou projekci. Pravděpodobnost výstupního slova w_j podmíněná kontextem o velikosti C je definována jako:

$$p(w_j|w_{I,1},\ldots,w_{I,C}) = y_j = \frac{exp(u_j)}{\sum_{j'=1}^{V} exp(u_{j'})}, \tag{2.13}$$

kde y_j je výstupní neuron reprezentující slovo w_j, V slovník a u_j skóre slova w_j určené jako:

$$u_j = \mathbf{v'}_{w_j}^T \cdot \mathbf{h}, \tag{2.14}$$

kde $\mathbf{v'}_{w_j}$ je výstupní vektor slova w_j a $\mathbf{h}$ projekční vrstva.

Tato architektura je velice podobná NNLM, kde je odstraněna nelineární skrytá vrstva a projekční vrstva je sdílena pro všechna slova. To znamená, že všechna slova kontextu jsou promítnuta do jednoho vektoru, který tvoří projekční vrstvu $\mathbf{h}$:

$$\mathbf{h} = \frac{1}{C} \cdot (\mathbf{v}_{w_1} + \cdots + \mathbf{v}_{w_C}), \tag{2.15}$$

kde $\mathbf{v}$ jsou vstupní vektory slov.

Původní NNLM mají složitost trénování:

$$Q = N \times D + N \times D \times H + H \times V, \tag{2.16}$$

kde N je velikost okna, D počet dimenzí slovního vektoru, H velikost skryté vrstvy a V velikost slovníku. Nejnáročnější část tvoří $H \times V$, jelikož velikost slovníku je jednoznačně větší než ostatní veličiny. Tuto část řeší *word2vec* pomocí *hierarchického softmaxu* nebo *negativního vzorkování* a redukují ji na $H \times log_2(V)$ (viz následující kapitoly). Díky tomu se stává nejnáročnější částí $N \times D \times H$ a jelikož *CBOW* model nedisponuje skrytou vrstvou, celková trénovací náročnost se redukuje až na:

$$Q = N \times D + D \times log_2(V) \tag{2.17}$$

2.3.3 Skip-gram

Druhý předvedený model *Skip-gram*[30] je založený na stejném principu, ale namísto předpovídání prostředního slova v okně se snaží podle daného slova

předpovědět kontext:

$$p(w_{c,j} = w_{O,c}|w_I) = y_{c,j} = \frac{exp(u_{c,j})}{\sum_{j'=1}^{V} exp(u_{j'})}, \tag{2.18}$$

kde $w_{c,j}$ je j-té slovo kandidující na pozici c v kontextu, $w_{O,c}$ správné předpovídané slovo kontextu na pozici c, w_I je jediné vstupní slovo, $y_{c,j}$ výstupní neuron příslušící slovu w_j a pozici c v kontextu a $u_{c,j} = u_j$, jelikož výstupní vrstva sdílí výstupní váhy:

$$u_{c,j} = u_j = \mathbf{v}'^{T}_{w_j} \cdot \mathbf{h}, \text{pro } c = 1, 2, \ldots, C \tag{2.19}$$

Jelikož se jako vstup používá pouze jedno slovo, tak projekční vrstva $\mathbf{h}$ je rovna vstupnímu slovu $\mathbf{v}_{w_I}$ a je možné u_j dále zjednodušit na:

$$u_j = \mathbf{v}'^{T}_{w_j} \cdot \mathbf{v}_I \tag{2.20}$$

S rostoucí maximální vzdáleností roste i kvalita výsledných vektorů a samozřejmě roste výpočetní složitost, kde rovnice trénování má následující tvar:

$$Q = C \times (D + D \times log_2(V)), \tag{2.21}$$

kde C je maximální možná vzdálenost slov. Pro každé trénovací slovo se volí náhodně číslo R v intervalu $< 1; C >$. Na obrázku 2.3 jsou znázorněny oba modely sítí.

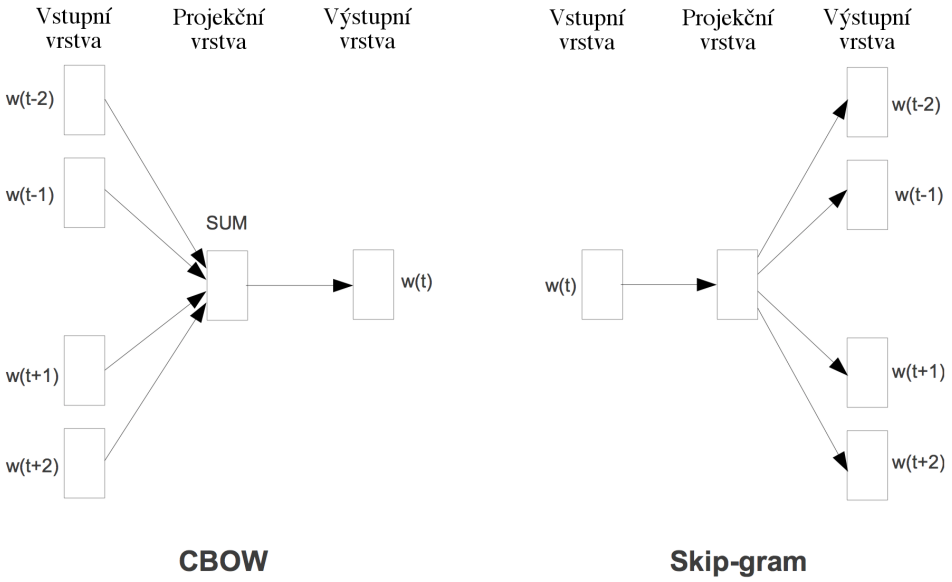
2.3.4 Hierarchický softmax

Hierarchický softmax (Hierarchical softmax)[39] je jednou z učících alternativ sítě metody *word2vec*. Jedná se o početně efektivnější alternativu plného softmaxu. Ve spojení s neuronovými sítěmi byl poprvé předveden v [36]. Místo vyhodnocování W výstupních neuronů pro výpočet pravděpodobnostního rozdělení stačí vyhodnotit okolo $log_2(W)$ neuronů.

Hierarchický softmax používá binární strom k reprezentaci výstupních neuronů sítě, kde listy stromu reprezentují slova a ostatní uzly udávají relativní pravděpodobnost svých dětí. Každé slovo w lze dosáhnout pomocí unikátní cesty od kořene stromu a tato cesta udává pravděpodobnost tohoto slova.

V tomto modelu nejsou žádné výstupní vektory reprezentující slova. Pravděpodnost, že dané slovo je výstupem se počítá následovně:

$$p(w = w_O) = \prod_{j=1}^{L(w)-1} \sigma([[n(w, j+1) = ch(n(w, j))]] \cdot \mathbf{v}'^{T}_{n(w,j)}\mathbf{h}), \tag{2.22}$$



Obrázek 2.3: Ukázka modelů sítí *word2vec*[30].

kde $ch(n)$ je levé dítě uzlu n, $\mathbf{v'}_{n(w,j)}$ je vektorová reprezentace (výstup sítě) vnitřního uzlu $n(w,j)$, $\mathbf{h}$ je projekční vrstva a $[[x]]$ je speciální funkce definována jako:

$$[[x]] = \begin{cases} 1 & \text{když } x \text{ je pravda;} \\ -1 & \text{jinak.} \end{cases} \tag{2.23}$$

Chybovou funkci je poté možné vyjádřit jako:

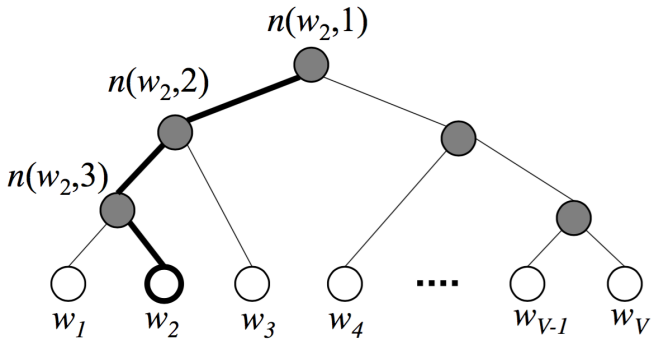
$$E = -log\ p(w = w_O|w_i) \tag{2.24}$$

$$E = -\sum_{j=1}^{L(w)-1} log\ \sigma([[n(w,j+1) = ch(n(w,j))]] \cdot \mathbf{v'}^T_{n(w,j)}\mathbf{h}) \tag{2.25}$$

word2vec navíc používá k tvorbě binárního stromu Huffmanovo kódování, které četnějším slovům přiřazuje kratší kódy, tzn. že jsou blíže ke kořenu stromu. Tento přístup se projevil jako další jednoduché urychlení trénovacího procesu.

2.3.5 Negativní vzorkování

Druhá možnost při trénování je *negativní vzorkování (Negative sampling)*[39]. Řeší problém velkého slovníku a plného softmaxu tak, že místo aktualizace



Obrázek 2.4: Ukázka binárního stromu používaného hierarchickým softmaxem.

všech výstupních vektorů při každé iteraci, aktualizuje pouze část (vzorek) z nich.

Mimo aktuálního výstupního slova (pozitivní vzorek - positive sample) potřebujeme aktualizovat i množinu jiných slov (negativní vzorky - negative samples). Pro výběr těchto slov je třeba pravděpodobnostní rozdělení, které může být libovolně zvoleno. Toto rozdělení se nazývá rozdělení šumu a značí se $P_n(w)$. *word2vec* využívá unigramové rozdělení umocněné na $\frac{3}{4}$.

Chybová funkce negativního vzorkování vypadá následovně:

$$E = -log\ \sigma(\mathbf{v'}^T_{w_O}\mathbf{h}) - \sum_{w_i \in \mathcal{W}_{neg}} log\ \sigma(-\mathbf{v'}^T_{w_i}\mathbf{h}), \tag{2.26}$$

kde w_O je aktuální výstupní slovo (positive sample), $\mathbf{v'}_{w_O}$ je výstupní vektor pro dané slovo, $\mathbf{h}$ je projekční vrstva a $\mathcal{W}_{neg} = \{w_i | i = 1, \ldots, K\}$ je množina slov (negativní vzorky) navzorkovaná pomocí $P_n(w)$.

2.3.6 Podvzorkování

Další důležitý faktor, který má vliv jak na rychlost učení, tak na kvalitu výsledných vektorů, je *podvzorkování*[32]. Ve velkých korpusech mají nejvíce frekventovaná slova až miliony výskytů (např. tvary slova "být" nebo spojka "a"). Tyto slova obvykle poskytují méně informace než slova neobvyklá. Například modelu pomáhá výskyt slov "Čechy" a "Praha" více než dvojice "Čechy" a "jsou" nebo "Čechy" a "mají". Zároveň vektory těchto frekventovaných slov se mění po milionech výskytů jen velmi nepatrně.

K vyvážení nerovnoměrnosti mezi neobvyklými a četnými slovy se používá právě podvzorkování, které každé slovo w_i slovníku vynechá s pravdě-

podobností vypočtené podle vzorce:

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}}, \tag{2.27}$$

kde $f(w_i)$ je frekvence slova w_i a t zvolený práh (obvykle 10^5). Tato rovnice byla vybrána, protože agresivně podvzorkovává slova s frekvencí větší než t, ale mohou být podvzorkována i slova s nižší frekvencí.

Naopak slova, která se vyskytují v příliš malém množství, jsou také vynechána, jelikož se může jednat o překlepy nebo cizí názvy pro model nedůležité. V případě malého výskytu slova by ani vektor daného slova neodpovídal jeho významu.

2.3.7 Word2phrase

V textu se mohou objevovat fráze, kde význam fráze není pouhým spojením jejích dvou nebo více slov. Proto *word2vec* disponuje dalším nástrojem *word2phrase*[32], který jednotlivé fráze spojuje do jednoho slova.

Nejdříve se najdou slova, která se častěji objevují spolu a neobvykle v jiném kontextu. Například "AC Sparta Praha" nebo "Západočeská univerzita" jsou nahrazena jedním slovem a "nová univerzita" zůstane nepozměněno. Tímto způsobem se vytvoří důležité významové fráze a velikost slovníku vzroste jen nepatrně.

Spojování frází je založeno pouze na unigramech a bigramech:

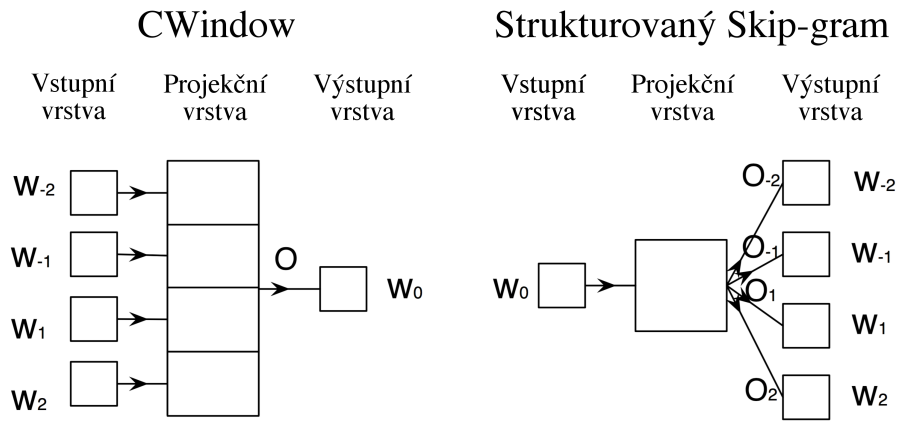
$$score(w_i, w_j) = \frac{count(w_i, w_j) - \delta}{count(w_i) \times count(w_j)}, \tag{2.28}$$

kde δ je penalizační koeficient, který zabraňuje vytváření velkého množství frází z hodně neobvyklých slov. Fráze je vytvořena právě, když výsledné skóre pro daný bigram je větší než zvolený práh. Obvykle je vhodné spustit tento proces 3-4 se zmenšujícím se prahem k vytvoření víceslovných frází.

2.3.8 Známá rozšíření

Ignorováním pozice slova v okně se ztrácí syntaktická informace a nemohou být dostatečně zachyceny vztahy mezi jednotlivými slovy. Například po přídavném jméně bude následovat další přídavné jméno nebo podstatné jméno atd. Proto se v [22] rozhodli brát v úvahu pořadí slov v okně a předvedli dvě nové architektury: *Strukturovaný Skip-gram* a *Spojité okno (Continuos Window - CWindow)*.

Strukturovaný skip-gram má pro určení každé pozice v kontextu svojí výstupní vrstvu na rozdíl od obyčejného skip-gramu, který sdílí jednu vrstvu



Obrázek 2.5: Ukázka modelů CWindow a Strukturovaný skip-gram[22].

pro všechny pozice. Výpočetní náročnost zůstává stále stejná, jelikož se pro každé slovo v okně mění výstupní vrstva. V případě CWindow má analogicky každé slovo v kontextu svojí projekční vrstvu na rozdíl od CBOW, kde je jedna vrstva sdílená. Tato architektura má při trénování vyšší výpočetní náročnost, protože při dopředné propagaci se používají všechny projekční vstvy.

Na úlohách určování slovních druhů (part-of-speech tagging) a zpracování závislostí se povedlo navýšit úspěšnost původní metody *word2vec* v rámci jednoho procenta.

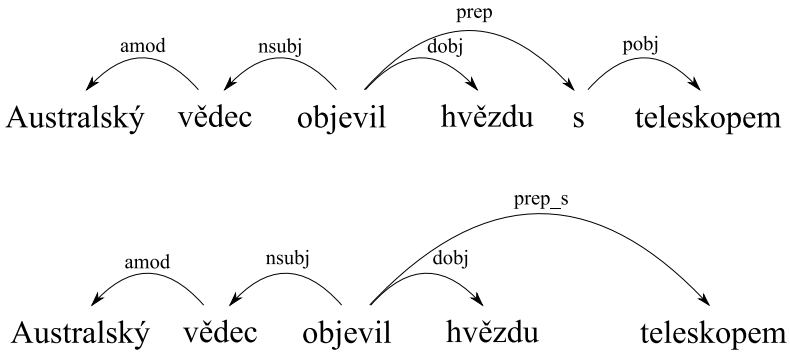
Další prozkoumané rozšíření se zabývá použitím jiného kontextu slova. Místo bag-of-words kontextu používá kontext založený na závislostech (Dependency-Based context)[21]. Tento kontext je definovaný tak, že pro každé cílové slovo w s modifikátory (slovo v nějakém syntaktickém vztahu se zkoumaným slovem) $m_1, \ldots, m_k$ a začátkem (slovo, od kterého se syntaktický kontext odvíjí) h, kontext tvoří:

$$(m_1, lbl_1), \ldots, (m_k, lbl_k), (h, lbl_h^{-1}), \tag{2.29}$$

kde *lbl* je typ závislosti mezi začátkem a modifikátorem (například nsubj, dobj, prep_with, amod) a lbl^{-1} slouží k označení inverzního vztahu.

Výsledné vektory lépe modelují syntaktické vztahy, to znamená, že například přídavnému jménu budou více podobná jiná přídavná jména než podstatná jména, ke kterým se dané přídavné jméno váže a podobně.

Dalším nástrojem vycházejícím z *word2vec* je *feat2vec* a model *FEMA (**F**eature **EM**beddings for domain **A**daptation)*[47], který se zabývá změnou domén a na místo vektorů slov vytváří vektory featur. To znamená, že nebere



SLOVO	KONTEXT
australský	vědec/amod^{-1}
vědec	australský/amod, objevil/nsubj^{-1}
objevil	vědec/nsubj, hvězdu/dobj, telescope/prep_s
hvězdu	objevil/dobj^{-1}
teleskopem	objevil/prep_s^{-1}

Obrázek 2.6: Ukázka kontextu se závislostmi[21].

v úvahu pouze výskyt slova, ale celou řadu příznaků daného slova a navíc je každé slovo spojeno s binárním vektorem domén. Výsledné vektory jsou následně použity *support vector machine* nebo *conditional random field* klasifikátorem k dalšímu zpracování. Tento přístup byl otestován na určování slovních druhů a lépe se vypořádá se změnou domény textu než klasický *word2vec*.

V případě, že je k dispozici k trénování zarovnaný dvojjazyčný text, lze využít *dvojjazyčný joint model*[45]. Rozšíření je založeno na předpokladu, že slova mohou mít více významů a trénování vektorů ve více jazycích může vést k upřesnění významu slova a obecně ke zlepšení výsledných vektorů. Také se dá použít jako rozšíření trénovací množiny v jazycích, kde není dostatečné množství trénovacích dat (např. Arabština).

Oba jazyky (zdrojový a cílový) jsou obsaženy ve stejném modelu tak, že vstupní vrstva obsahuje slovníky obou jazyků, stejně jako vrstva výstupní. Síť je poté trénována ve čtyřech fázích. V první fázi se odhaduje cílové slovo podle cílového kontextu, v druhé fázi se anologicky určí zdrojové slovo podle zdrojového kontextu. Ačkoli je využita stejná projekční vrstva pro obě fáze, nejsou jazyky zcela svázány a proto následují další dvě fáze, kde se odhadují slova z opačných kontextů: zdrojové slovo podle cílového kontextu a cílové slovo podle zdrojového kontextu.

Poslední prozkoumané rozšíření *Vztahy omezující model (Relation Constrained Model - RCM)*[48] se zabývá spojením *word2vec* s lexikálními databázemi (*WordNet* a *Paraphrase database*[11]). Definuje R jako množinu vztahů mezi slovy w a w' a tyto vztahy ohodnocuje podle jejich síly. Cílem tohoto modelu je maximalizovat pravděpodobnost všech vztahů přes všechna slova N ve slovníku:

$$\frac{1}{N}\sum_{i=1}^{N}\sum_{w\in R_{w_i}} log\ p(w|w_i), \tag{2.30}$$

kde R_w je podmnožina vztahů R mezi w a w_i. RCM společně s CBOW tvoří tzv. *Joint model*, kde tento výsledný model tvoří jejich vážená (C) lineární kombinace:

$$\frac{1}{T}\sum_{t=1}^{T} log\ p(w_t|w_{t-c}^{t+c}) + \frac{C}{N}\sum_{i=1}^{N}\sum_{w\in R_{w_i}} log\ p(w|w_i) \tag{2.31}$$

Z tohoto vztahu je možné vidět, že CBOW se trénuje přes všechna slova trénovacího korpusu T a RCM se trénuje přes všechna slova slovníku N.

Nejlepších výsledků bylo dosaženo v případě použití Joint modelu s náhodnými počátečními slovními vektory, které jsou následně použity jako počáteční vektory opětovného trénování RCM.

2.4 Korpus

Nejdříve je nutné upřesnit terminologii. V publikacích se často objevují pojmy *korpus* a *datová sada*, které se často zaměňují. V této práci jsou jako korpus označována trénovací data a jako testovací datová sada jsou označována testovací data.

Konečný model statistických metod je značně závislý na volbě trénovacích dat, čili trénovacího korpusu. Výsledky se liší na základě množství dat, ale také na základě domény, ze které data pochází. Je zřejmé, že rozložení slov a slovník, vytvořený na základě dat ze zpráv, budou jiné než v případě dat z Twitteru nebo diskuzí. Jestliže nemá být model zaměřen na cílovou doménu, je vhodné trénovat na obecném textu. K získání dostatečně velkého množství obecného textu lze použít právě výtažky ze zpravodajských webových stránek nebo offline zálohy wikipedie.

Trénovací data jsou sekvence po sobě jdoucích slov, která mohou být rozdělena do jednotlivých dokumentů. Tato data také mohou být předzpracovaná přidáním slovních druhů, stemováním nebo lematizací.

Pro ověření natrénovaného modelu se používají testovací datové sady. Pro testování statistických sémantických metod mapujících slov do vektorového

prostoru se používají testovací datové sady založené na podobnosti dvojic slov nebo nový přístup, předvedený Mikolov a kol.[33], využívající analogie slov.

Datové sady využívající analogie slov se skládají ze skupin otázek, kde každá skupina je zaměřena na jiný vztah mezi slovy. Každá otázka je tvořena čtyřmi slovy a vyhodnocuje se tak, že se pomocí vektorové aritmetiky odečte vektor druhého slova od vektoru prvního slova a přičte se vektor třetího slova. Aby otázka byla považována za úspěšnou, tak je nutné, aby nově vzniklý vektor byl nejblíže vektoru reprezentující čtvrté slovo otázky. Výsledná úspěšnost modelu je počítána jako poměr správně zodpovězených otázek ku celkovému počtu otázek.

Je možné měřit jak sémantickou přesnost, tak syntaktickou přesnost. Sémantické otázky jsou složeny ze čtyř odlišných slov, kde má být k třetímu slovu správně doplněno čtvrté slovo ve stejném vztahu jako je první slovo ke druhému (viz rovnice 2.32). Syntaktické otázky jsou zaměřené na jednotlivé slovní tvary. V těchto otázkách je třeba doplnit čtvrté slovo tak, aby modifikovalo třetí slovo stejně jako druhé slovo modifikuje slovo první (viz rovnice 2.33). Avšak nevýhodou těchto datových sad je závislost na vektorech. Tento přístup je možné použít pouze v případě, že metoda mapuje slova do vektorového prostoru, tudíž na nich není možné například vyhodnocovat dříve zmiňované metody založené na ontologiích.

$$vec(''\check{C}echy'') - vec(''Praha'') + vec(''Anglie'') = vec(''Londýn'') \quad (2.32)$$

$$vec(''p\acute{i}še'') - vec(''psal'') + vec(''čte'') = vec(''četl'') \quad (2.33)$$

Druhým typem testovacích datových sad jsou sady založené na podobnosti dvojic slov. Tímto způsobem je možné testovat všechny zmiňované přístupy sémantické podobnosti, na druhou stranu neudávají informaci o syntaktické přesnosti. Podobnost dvojic slov je určená lidskými anotátory. V následujících kapitolách budou jednotlivé datové sady probrány podrobně.

V případě, že skóre jsou určována lidskými anotáry, je vhodné určit konzistenci jejich rozhodnutí tzv. *vzájemnou shodu anotátorů*[6][46]. Pro tyto účely se používá *Cohenův kappa koeficient*(κ_{Cohen})[6] definovaný jako:

$$\kappa_{Cohen} = \frac{P(A) - P(E)}{1 - P(E)}, \quad (2.34)$$

kde $P(A)$ je poměr případů, kdy se anotátoři shodli a $P(E)$ je poměr případů, kde se předpokládá, že shoda anotátorů byla pouze náhoda. Tento

[6]Z anglického *inter-rater agreement*.

koeficient je definovaný pouze pro dva anotátory, pro určení shody mezi větším množstvím anotátorů je nutné vypočítat aritmetický průměr koeficientů mezi všemi anotátory nebo použít jeho modifikaci *Fleissův kappa koeficient*(κ_{Fleiss})[10]. *Fleissův kappa koeficient* je definovaný stejně jako *Cohenův kappa koeficient*:

$$\kappa_{Fleiss} = \frac{P(A) - P(E)}{1 - P(E)}, \tag{2.35}$$

kde jsou jednotlivé pravděpodobnosti určeny jako:

$$P(A) = \bar{P} = \frac{1}{n}\sum_{i=1}^{n} P_i = \frac{\sum_{i=1}^{n}\sum_{j=1}^{k} x_{ij}^2 - mn}{mn(m-1)} \tag{2.36}$$

$$P(E) = \frac{1}{nm}\sum_{i=1}^{n} x_{ij}, \tag{2.37}$$

kde n je počet hodnocených párů, m je počet anotátorů, x_{ij} je počet hodnocení i-tého páru skórem j (detailní odvození viz [10]).

K porovnávání úspěšnosti modelů na těchto datových sadách se používá korelace výsledků získaných modelem a datovou sadou. Obvykle se používá *Pearsonova korelace* nebo *Spearmanova korelace*[9].

Pearsonova korelace udává míru lineární závislosti dvou veličin a je definována jako:

$$\rho = \frac{cov(X,Y)}{\sigma(X) \times \sigma(Y)}, \tag{2.38}$$

kde $cov(X,Y)$ je kovariance definována jako:

$$cov(X,Y) = E(X \times Y) - E(X) \times E(Y), \tag{2.39}$$

kde $E(X)$ je střední hodnota X definována jako:

$$E(X) = \int_{-\infty}^{+\infty} x \times f(x)dx \tag{2.40}$$

a $\sigma(X)$ směrodatná odchylka X:

$$\sigma(X) = \sqrt{E(X^2) - E^2(X)} \tag{2.41}$$

Pro výpočet korelace mezi vektory lze použít výběrový korelační koeficient:

$$r_{Pearson}(\mathbf{x},\mathbf{y}) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=0}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=0}^{n}(y_i - \bar{y})^2}} \tag{2.42}$$

Spearmanova korelace udává míru lineární závislosti mezi pořadím jednotlivých položek vektorů:

$$r_{Spearman}(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{n}(r_{x_i} - \bar{r_x})(r_{y_i} - \bar{r_y})}{\sqrt{\sum_{i=0}^{n}(r_{x_i} - \bar{r_x})^2}\sqrt{\sum_{i=0}^{n}(r_{y_i} - \bar{r_y})^2}}, \tag{2.43}$$

kde r_x a r_y udávají pořadí jednotlivých hodnot X a Y, $\bar{r_x}$ a $\bar{r_y}$ je výběrový průměr r_x a r_y definovaný jako:

$$\bar{r_x} = \bar{r_y} = \frac{n+1}{2} \tag{2.44}$$

$$\sum_{i=0}^{n}(r_{x_i} - \bar{r_x})^2 = \sum_{i=0}^{n}(r_{y_i} - \bar{r_y})^2 = \frac{N^3 - N}{12} \tag{2.45}$$

Po úpravě (detailní odvození viz [9]):

$$r_{Spearman}(\mathbf{x}, \mathbf{y}) = 1 - \frac{6\sum_{i=0}^{n}(r_{x_i} - r_{y_i})^2}{n \times (n^2 - 1)} \tag{2.46}$$

Korelace je číslo mezi -1 a 1 takové, že v absolutní hodnotě udává lineární závislost dvou veličin, kde čím více se hodnota blíží jedné, tím jsou veličiny více lineárně závislé. To znamená, že čím větší hodnota, tím více se výsledky modelu shodují s anotátory testovací datové sady.

2.4.1 Rubenstein-Goodenough

Rubenstein-Goodenough (RG) je jeden z prvních datasetů svého druhu publikovaný již v roce 1965. Bylo vybráno 65 párů slov obsahující běžná anglická slova, která byla ohodnocena dvěma skupinami anotátorů. Každý anotátor dostal instrukce nejdříve seřadit dvojice podle podobnosti a až poté jim přiřadit hodnoty podobnosti 0-4.

První skupina prováděla hodnocení ve dvou fázích po dvou týdnech. V obou fázích se překrývalo 36 dvojic slov, aby bylo možné spočítat spolehlivost jednotlivých anotátorů. Jedná se o informaci, jak se anotátor shodne sám se sebou.

2.4.2 Miller

Tato datová sada je podmnožinou RG vybraná v [35].

2.4.3 Wordsim

Jelikož předchozí datové sady obsahovaly pouze malé množství dvojic, rozhodli se v [1] vytvořit obsáhlejší datovou sadu z 350 párů podstatných jmén. Mezi těmito páry je obsaženo i 30 párů z *Millerovi datové sady* a také 82 párů, kde alespoň jedno ze slov není obsaženo ve *WordNetu*. Všechny páry byly hodnoceny 16 anotátory na stupnici od 0 do 10.

2.4.4 MTurk

Tato datová sada byla hodnocena pomocí služby poskytované Amazonem, která se nazývá *Amazon's Mechanical Turk service (MTurk)*. 280 dvojic slov bylo vygenerováno automaticky ze slovníku všech slov článků *New York Times*, které se zároveň nachází v *DBPedii*. Dále byla odstraněna *stop words* slova a vzácná slova (slova s počtem výskytů menším než 1000).

Jednotlivé páry slov byly rozděleny do dávek po 50 párech. Každá dávka měla navíc přidáno také 10 párů z *Wordsim datové sady*, na základě kterých se ověřovala kvalita anotátorů. Každou dávku anotovalo 30 různých anotátorů od *MTurk*. Jelikož je možné, že se jeden člověk zaregistruje pod dvěma účty, není možné zaručit, že nebyli dva stejní anotátoři pro jednu dávku, stejně jako že byl splněn požadavek, že anotátor je rodilý anglický mluvčí[38].

2.4.5 Rare words

Datová sada *Rare words*[25] se skládá pouze z vzácných slov, která se v textu neobjevují zcela běžně. Výběr prvního slova páru byl založen na předponách a koncovkách, které dané slovo obsahovalo a také na frekvenci výskytů na anglické *Wikipedii*, kde se rozlišovalo 5 skupin podle četnosti: $(5, 10]$, $(10, 100]$, $(100, 1000]$, $(1000, 10000]$. Jelikož se v méně frekventovaných skupinách objevovala i cizí slova, která nebyla anglického původu, přidala se podmínka, že dané slovo musí být obsaženo v databázi *WordNet*. Bohužel kvůli této podmínce jsou značně zvýhodněny systémy založené na této znalostní databázi.

V druhém kroku se vytvářely páry. První slovo bylo vybráno náhodně z jedné zmiňované skupiny přípon, koncovek a frekvencí. Slovo do páru bylo náhodně vybráno ze slov, která měla k prvnímu slovu nějaký vztah ve *WordNetu* (např. hyponymum, hypernymum atd.). Tímto způsobem se vytvořilo 3145 párů slov.

K získání hodnocení od anotátorů byl opět použit *MTurk* od Amazonu. Anotátoři měli hodnotit na stupnici od 0 do 10 a navíc označovat slova, která

neznají. Slova s méně než 10 hodnotami podobnosti byla vyřazena a zbylo 2034 párů.

2.4.6 MEN

Datová sada *MEN*[4] obsahuje pouze slova, která se vyskytují jako značky v databázích označkovaných obrázků. Tato sada byla vytvořena především pro testování multimodálních modelů[7], ale je možné ji použít i pro testování pouze textových modelů.

Slovní páry byly náhodně vybrány tak, aby byly obsaženy jak v textovém korpusu (alespoň 700x), tak v databázi označkovaných obrázků (alespoň 50x). Aby se zabránilo výběru pouze slov s malou podobností, byly náhodné páry seřezeny pomocí kosínové vzdálenosti vypočtené pomocí textového modelu *HAL* s oknem o velikosti 20 na obě strany od zkoumaného slova. Konečná sada 3000 párů byla vybrána tak, že se vzalo prvních 1000 párů, dalších 1000 mezi 1001 a 3000 a posledních 1000 ze zbytku vygenerovaných párů.

K anotaci byli také použiti anotátoři od *MTurk*, avšak namísto hodnocení absolutním skórem, rozhodovali anotátoři binárně, která z poskytnutých dvojic je podobnější. Na základě těchto rozhodnutí bylo vygenerováno finální skóre dvojic.

2.5 Stemming

Stemming je proces sloužící k zredukování počtu skloňovaných nebo jinak upravených slov v závislosti na jejich syntaktické roli ve větě. Během tohoto procesu se nechávají ze slov pouze jejich *stemy*. Stem je v lingvistice definován různě. V některých publikacích je stem definován jako část slova, ze které se pomocí různých lingvistických procesů vytváří slova nová. Podle [16] je možné nová slova vytvářet pomocí metody *kompozice*, která kombinuje jednotlivé stemy (například *black–bird*) nebo metody *derivace*, která vytváří nová slova pomocí připojení affixů. Avšak v českém jazyce je ke kompozici dvou stemů nutný ještě *spojovací morfém (interfix)* (například *velk*–**o**–město). Stemy, které bez jakékoli modifikace, tvoří samostatné slovo se nazývají *volné* stemy[8], naopak od stemů *vázaných*[9], které samy netvoří slovo. Jiné publikace ([18]) definují stem jako část slova, která zůstává stejná

[7]Multimodální modely se zabývají více druhy (reprezentacemi) informace. V tomto případě textem a obrazem.

[8]Z anglického *free* stem

[9]Z anglického *bound* stem

pro všechny skloňované a časované tvary (např. kol–o, kol–a, kol–e.)

Problémy nastávají v případě, kdy dvě významově odlišná slova mají stejný stem. Po stemmingu se tato slova budou považovat za slova stejná. Například slovo *leden* a *led* jsou slova významově naprosto odlišná, ale mají stejný stem, tudíž se po stemmingu budou považovat za slova totožná.

Existuje několik algoritmů pro stemming, které se liší účinností ořezávání a způsobem přístupu. První algoritmy využívaly pravidlový přístup pro ořezávání přípon. Jsou založeny na základě nadefinovaných pravidel, která od slov ořezávají přípony. Některé příklady pravidel:

- jestliže slovo končí -atech, odstraň -atech
- jestliže slovo končí -ětem, odstraň -ětem
- jestliže slovo končí -ou, odstraň -ou

Tato pravidla se mohou aplikovat na jedno slovo několikrát. Podobně lze uzpůsobit pravidla i na předpony. Tento algoritmus je velice jednoduchý a dosahuje dobrých výsledků. Nevýhodou však je závislost na jazyce.

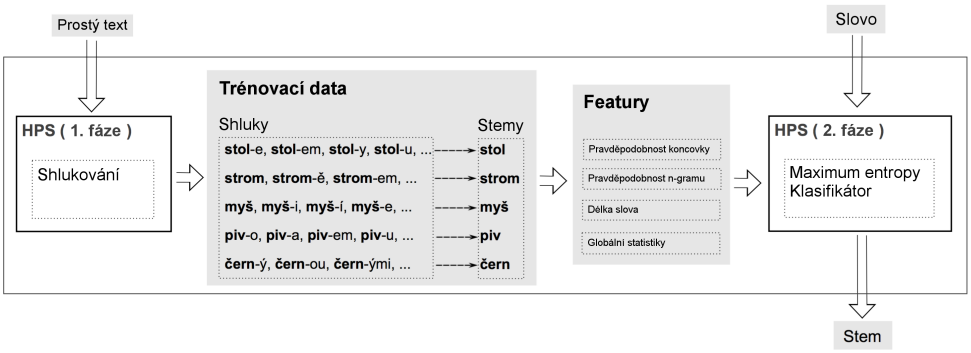
Se stemmingem je úzce spojen proces lemnatizace. Lemnatizace na rozdíl od stemmingu pracuje s kontextem a nehledá pouze kořeny slov, ale jejich základní tvary. Například slovo *horší* má základní tvar *špatný*, což stemmer nemůže bez kontextu poznat[8].

2.5.1 High Precision Stemmer

High Precision Stemmer (HPS)[5] je stemmer založený na statistickém přístupu a skládá se ze dvou fází. První fáze vychází z předpokladu, že stemming by měl zachovat sémantickou informaci a odstranit morfosyntaktickou informaci obsaženou ve slově. Proto se v této fázi provádí *shlukování (clustering)*, kde se shlukují slova, která se nacházejí ve stejném kontextu (sémanticky podobná slova – viz distribuční hypotéza v kapitole *2.2.3 Statistické metody*) a sdílí dostatečně dlouhý společný prefix (lexikálně podobná slova). Už výstup první fáze lze použít jako samostatný stemmer ořezáním všech slov v jedné skupině (clusteru) podle jejich nejdelšího společného prefixu. Avšak takto samostatný stemmer nedosahuje uspokojivých výsledků. Místo toho jsou takto ostemovaná slova použita jako trénovací data pro druhou fázi.

Druhá fáze využívá faktu, že na stemmování se používá mnoho pravidel (viz pravidlové přistupy). Proto má druhá fáze klasifikační charakter a pravidla jsou transformována do jednotlivých *featur*[10]. Jako klasifikátor je použit

[10]Často překládáno jako *příznak*, avšak tento překlad je sporný a proto je ponechán počeštěný anglický výraz *featura*.



Obrázek 2.7: Architektura HPS[5].

klasifikátor *Maximum Entropy* a k jeho natrénování jsou použita výstupní data první fáze.

Jelikož pravidla klasifikátoru jsou dostatečně obecná, podává stemmer velice dobré výsledky jak na známých (viděných) slovech, tak na neznámých (neviděných) slovech.

3 Analýza

3.1 Předzpracování

Jak již bylo řečeno, konečný model je značně závislý na volbě a tvorbě trénovacího korpusu a s tím je úzce spojeno také jeho předzpracování. Cílem předzpracování je odstranit šum a nežadoucí data (jakými jsou například HTML značky a podobně) z textu. Přitom je však nutné zvážit co všechno bude považováno za šum a nežádoucí. V případě původního *word2vec* je považováno za nežádoucí vše mimo interpunkci a slova. Při trénování na takto zpracovaném korpusu je zachována informace, jestli právě zkoumané okno obsahuje konec věty nebo jestli se slovo obvykle vyskytuje v okně s čárkou a podobně. Na druhou stranu interpunkce zabírá místo v okně, kde by jinak mohlo být významové slovo. V případě, že se tyto znaky odstraní, zmenší se také velikost trénovacího korpusu a trénování proběhne v kratším čase, tudíž je možné trénovat model na větších datech ve stejném čase. V práci budou ověřeny oba přístupy a očekává se, že ponechaná interpunkce je důležitá a není možné ji odstranit.

3.2 Známé rozšíření

Ani jedno ze známých rozšíření není zaměřeno na flektivní jazyky, mezi které patří čeština. Tyto jazyky kvůli skloňování, časování, předponám a příponám obsahují velké množství slovních tvarů. Navíc ani jeden z modelů nedosahuje značně lepších výsledků než původní *word2vec*. RCM model (viz kapitola *2.3.8 Známá rozšíření*) sice dosahuje lepších výsledků než původní *word2vec*, ale závislost na externí databázi je značné omezení, obzvlášť v případě českého jazyka.

V rámci vývoje architektur CWindow a Strukturovaného Skip-gramu bylo pravděpodobně vyvinuto také rozšíření *cngram2vec (Character n-gram)*[1], které však nebylo nikde publikováno. Jelikož bylo rozšíření vyvíjeno pro anglický jazyk, nepodporuje české znaky. Rozšíření není příliš optimalizované, n-gramy se vytváří opakovaně při trénování a značně prodlužují dobu trénování. Návrh tohoto modelu se shoduje s návrhem modelu *ngram2vec* vyvinutého v rámci této práce.

[1] https://github.com/wlin12/wang2vec/blob/master/cngram2vec.c

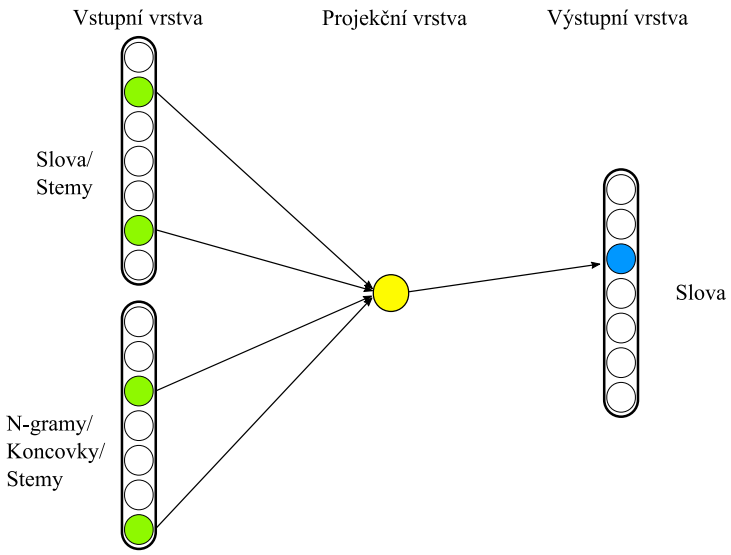
3.3 Rozšíření modelů

Tato práce se zabývá rozšířením nástroje *word2vec* se zaměřením na český jazyk. Jelikož je čeština flektivní jazyk, obsahuje velké množství slovních tvarů, které jsou v rámci *word2vec* modelu považovány za samostatná slova, i když můžou nést stejný sémantický význam. Například slovo *král* a *královna* mají téměř stejný sémantický význam, ale v modelu mají dva nezávislé samostatné vektory. Kdyby v trénovacím korpusu byla obě slova zastoupena ve stejné míře a slova byla reprezentovaná stejným vektorem, aktualizoval by se tento vektor dvakrát častěji a díky tomu by mohl vektor lépe modelovat sémantický význam. Na druhou stranu by tento vektor nenesl žadnou informaci o syntaktické podobnosti. Proto je nutné zachovat vlastnost původního modelu, že žádná dvě odlišná slova nesdílí stejný vektor. Z tohoto důvodu je nutné rozdělit slovo na část, která nese informaci o sémantické podobnosti, sdílenou pro sémanticky podobná slova, a syntaktickou část, která specifikuje dané slovo a zároveň je sdílena se syntakticky podobnými slovy.

K rozložení slov na menší části je možné použít stemmer nebo znakové n-gramy. V případě stemmeru je slovo rozděleno na dvě části: stem a koncovku, které by měly reprezentovat dvě části slova zmiňované v předchozím odstavci. V případě znakových n-gramů nemusí být slovo vůbec rozděleno i přesto, že zmiňované dvě části obsahuje, nebo rozděleno na více než dvě části. Tyto části jsou oproti částem vytvořených stemmerem stejně velké. Tento fakt může být výhodou, například v případě složených slov, ale také nevýhodou, jelikož slovo může být rozděleno do nevhodných částí. Pomocí znakových n-gramů může být také oddělena předpona, kterou většina stemmerů ponechává, jako v případě HPS stemmeru (viz kapitola *2.5.1 High Precision Stemmer*).

Při rozdělování pomocí znakových n-gramů je nutné zvolit jejich velikost. Jestliže bude velikost příliš malá, jednotlivé celky neponesou žádnou sémantickou ani syntaktickou informaci, na druhou stranu jestliže velikost bude příliš velká, n-gramy budou stejné jako samotná slova a rozdělení bude postrádat smysl. Také je nutné zvážit, zda rozlišovat mezi n-gramy na krajích slova a uprostřed slova. Například n-gram *pod* ve slově ***pod**chod* má jiný sémantický význam než ve slově *hos**pod**ářství*.

Aby bylo možné během trénování rozlišit a využít jednotlivé části slova, byla původní architektura rozšířena tak, aby podporovala ve vstupní vrstvě dva nezávislé vstupy. Tato *základní architektura* (viz obrázek 3.1) obsahuje dvě nezávislé vstupní vrstvy, které jsou promítány do jedné společné projekční vrstvy. Během trénování se pro každé slovo okna aktivují jeho odpovídající neurony v obou vstupních vrstvách a váhy se promítnou do projekční

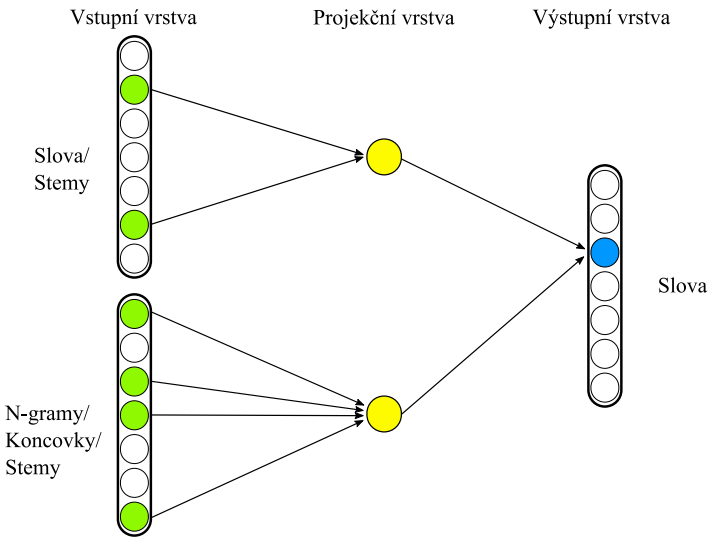


Obrázek 3.1: Ukázka *základního architektury.*

vrstvy. Výstupní vrstva zůstává stejná jako v případě původní architektury, kde se aktivuje právě trénované slovo, vypočítá se chyba a aktualizují se váhy výstupního neuronu. Nakonec jsou na základě vypočtené chyby aktualizovány všechny váhy aktivovaných neuronů ve vstupní vrstvě. V tomto případě se ovlivňují oba vstupy během celého trénování.

I přesto, že základní architektura má vstupní vrstvy oddělené, při trénování jsou promítnuty do jedné vrstvy a aktualizovány společně, tudíž je chyba prvního vstupu propagována i do druhého vstupu a opačně. Proto byla navržena *architektura s oddělenými projekčními vrstvami (separated)* (viz obrázek 3.2). V tomto případě se při trénování nejdříve aktivují neurony v první vrstvě, provede se aktualizace neuronové sítě a poté se proces zopakuje pro druhovou vrstvu. Tyto kroky je možné provádět po sobě pro každé slovo v okně nebo je možné provést první krok pro celý korpus a při druhém průchodu provézt druhý krok. Tento přístup je v práci nazýván *fázovým modelem (phase)*. V tomto modelu se oba vstupy ovliňují pouze pomocí sdílené výstupní vrstvy a jejích vah.

Dalším způsobem jak oddělit vstupy v projekční vrstvě je udělat dvojnásobně velkou projekční vrstvu (dimenzi) a vektory z obou vstupních vrstev složit za sebe. To znamená, že do první poloviny projekční vrstvy se promítá první vstupní vrstva a do druhé poloviny druhá vstupní vrstva (viz obrázek 3.3). Kvůli tomu má tato *projekční architektura* poloviční dimenzi vstupních vrstev oproti ostatním vrstvám. Zbytek trénovacího procesu zůstává nepozměněný a vstupní vrstvy jsou aktualizovány odpovídajícími chybami.



Obrázek 3.2: Ukázka *architektury s oddělenými projekčními vrstvami.*

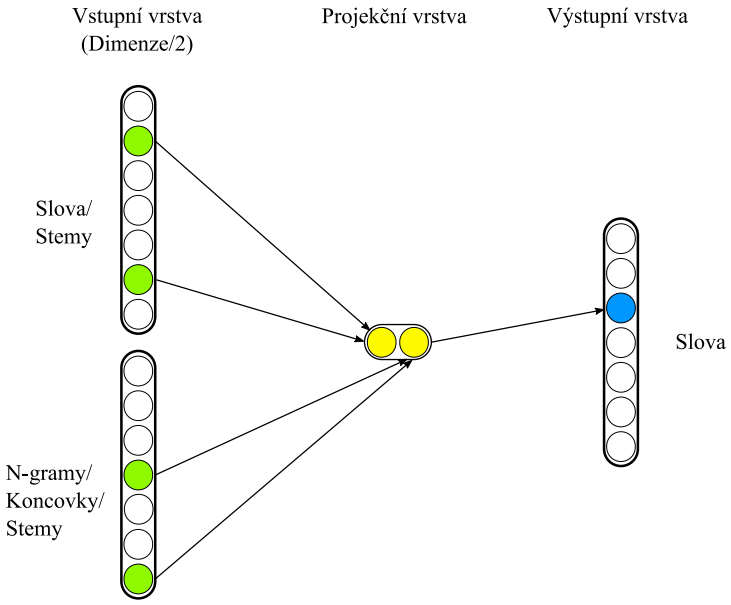
Aby byla zachována dimenze vektorů daná vstupním parametrem, je nutné vektory slov opět složit z odpovídajích vektorů z obou vstupních vrstev pro dané slovo. Jelikož je možné, že tato architektura bude doplácet na nedostatek parametrů v jednotlivých vstupních vrstvách, je vhodné otestovat tyto modely také s dvojnásobnou velikostí vektorů.

Všechny doposud zmiňované architektury měly pouze jednu výstupní vrstvu, která byla reprezentována vektory pro celá slova. To znamená, že se na základě dvou vstupů snaží předpovídat celá slova. *Kompletní architektura* (viz obrázek 3.4) obsahuje dvě výstupní vrstvy a snaží se předpovídat dva výstupy (například slova i n-gramy). Jelikož se mají oba vstupy a výstupy vzájemně ovlivňovat, je nutné, aby projekční vrstva byla sdílena. V opačném případě by trénování obou vstupů bylo nezávislé a bylo stejné jako trénování dvou samostatných původních modelů.

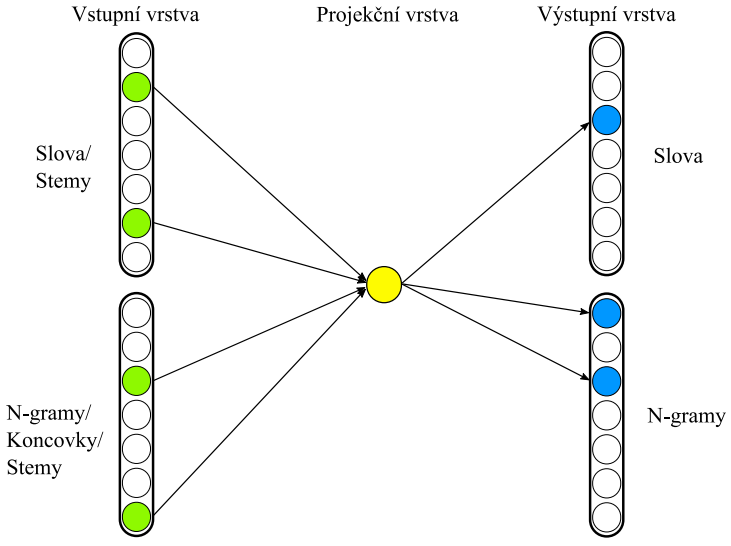
V případě, že by některý z modelů podával znatelně lepší výsledky pouze v jedné z kategorií, je možné tento model zkombinovat s jiným modelem, který podává dobré výsledky naopak v druhé kategorii. Vektory dvou různých modelů je možné sečíst nebo složit za sebe jako v případě promítané projekční vrstvy.

3.4 Evaluace

Při porovnávání modelů je nutné brát v úvahu jejich výpočetní složitost při trénování a srovnávat stejně náročné alternativy. Oddělený model se dvěma



Obrázek 3.3: Ukázka *projekční architektury.*



Obrázek 3.4: Ukázka *kompletní architektury.*

projekčními vrstvami při trénování provádí celý proces pro jedno okno dvakrát (pro každou projekční vrstvu zvlášť) a v případě fázového modelu prochází korpus dvakrát. To znamená, že při porovnávání s ostatními modely je nutné tyto modely trénovat na dvojnásobně větších datech nebo na dvou epochách. V případě zvolení dvojnásobně velkých trénovacích dat je riziko, že jeden z trénovacích korpusů bude více nebo méně zašuměný a výsledky tím budou ovlivněny. Proto je vhodnější používat k vyrovnání výpočetní náročnosti mezi modely větší počet epoch. Ve výsledcích bude u modelů vždy naměřena jedna epocha a také vybraný počet epoch, srovnatelný pro všechny porovnávané modely.

V případě kombinace dvou modelů je nutné brát v úvahu dobu trénování obou modelů. To znamená, že je potřeba tuto kombinaci porovnávat s modelem, který byl natrénován ve stejném počtu epoch jako oba modely dohromady. V případě kombinování modelů pomocí skládání vektorů je nutné zvážit, jestli je vhodnější vyrovnávat tuto výhodu pomocí epoch. Jelikož se vektory skládají za sebe, je jejich velikost dvojnásobná, takže je možné tuto kombinaci modelů porovnávat s modelem, který má dvojnásobnou velikost vektorů. Avšak výpočetní složitost natrénování dvou epoch neodpovídá výpočetní složitosti trénování dvojnásobně velkých vektorů. Na druhou stranu v případě stejné výpočetní složitosti zase neodpovídají velikosti vektorů. Budou provedy dva experimenty, jeden s dvěma epochami a druhý s dvojnásobnou velikostí vektorů.

Při evaluaci je možné, že se v datové sadě objeví slova, která nejsou obsažena ve slovníku. Je nutné zvážit, zda tyto záznamy považovat za špatně zodpovězené nebo je vynechat z evaluace. Tato chyba je více spojena s nedostatečně rozmanitým trénovacím korpusem než se samotným typem modelu. Je možné, že v případě dostatečného výskytu slova v korpusu by model otázku zodpověděl správně. Proto budou v práci tato slova při evaluaci vynechávána a bude vždy zmíněno procento neviděných dat (stejně jako v původním *word2vec* evaluačním skriptu).

4 Řešení

4.1 Jednotlivá rozšíření

V případě modelů založených na n-gramech není nutné žádné předzpracování textu. Slovník n-gramů se vytváří společně se slovníkem slov. Každé slovo obsahuje odkazy na n-gramy, ze kterých je složeno, aby se trénování nezpomalovalo kopírováním a vyhledáváním řetězců. Ve všech těchto rozšířeních je možné si zvolit velikost n-gramů a také možnost, jestli n-gramy na začátku a na konci slova mají být rozdílné od n-gramů uprostřed slov.

Pro použití stemovacích rozšíření je nutné trénovat model na předzpracovaném trénovacím korpusu (viz kapitola *4.2 Trénovací korpus*). Při trénování se vytváří se slovníkem slov také slovník pro stemy a koncovky. Každé slovo opět obsahuje odkazy na stem a koncovku, které toto slovo obsahuje, aby se co nejvíce urychlil trénovací proces. Tato rozšíření neobsahují žádné dílčí parametry.

V rámci práce byla vytvořena celá řada rozšíření kombinující různé přístupy a kombinace vstupních informací ve vstupní vrstvě. V případě n-gramových rozšíření se vůbec neosvědčila klasická architektura se sdílenou projekční vrstvou a model poskytoval výsledky v rámci jednoho procenta stejně jako kompletní architektura pro oba typy rozšíření. Proto nejsou modely založené na těchto architekturách v práci zmíněny. Pro stemovací rozšíření jsou v práci zmíněny všechny ostatní kombinace architektur a vstupních informací.

4.1.1 Ngramonly2vec

První vytvořené rozšíření je spíše experimentální a používá pouze n-gramy. Architektura modelu je shodná s původní *word2vec* architekturou, ale jak vstupní, tak výstupní vrstvu tvoří n-gramy. V případě *CBOW* se promítnou všechny n-gramy slov z okna do projekční vrstvy stejně tak jako v případě *Skip-gramu*, kde se do projekční vrstvy promítají pouze n-gramy jednoho vstupního slova.

Jelikož výstupní neurony jsou tvořeny také n-gramy, v případě *Hierarchického softmaxu* se vytváří Huffmanův strom pro n-gramy místo pro slova. Poté jsou pro každé předpovídané slovo aktualizovány výstupní váhy pro všechny jeho n-gramy, přesněji řečeno všechny rozhodovací uzly Huffmanova stromu vedoucí k danému n-gramu (viz kapitola *2.3.4 Hierarchický softmax*).

V případě *Negativního vzorkování* je situace poněkud jednodušší, protože se aktualizují výstupní váhy n-gramů předpovídaného slova a poté se náhodně vybere zvolený počet negativních vzorků, tedy n-gramů, které se aktualizují. Výsledné vektory slov jsou nakonec vytvořeny jako součet vektorů n-gramů, které obsahují.

4.1.2 Ngram2vec

V případě *ngram2vec* se na základě n-gramů předpovídají slova a je použita také původní *word2vec* architektura (shodné s modelem *cngram2vec*). To znamená, že vstupní neurony reprezentují n-gramy, stejně jako v případě *ngramOnly2vec*, ale výstupní neurony reprezentují slova. Do projekční vrstvy jsou opět promítány vektory n-gramů všech vstupních slov, ale výstupní váhy se aktualizují pouze pro jedno předpovídané slovo stejně jako v původním *word2vec*. Jelikož vstupní neurony reprezentují n-gramy, je opět nutné vytvořit konečné vektory slov jako součet vektorů všech n-gramů, které obsahují.

4.1.3 Wordgram2vec

Model *wordgram2vec* přidává do vstupní vrstvy navíc neurony reprezentující samotná slova a je založen na architektuře s oddělenými projekčními vrstvami. Do jedné projekční vrstvy se promítají slova a do druhé n-gramy.

Při trénování se pro každé okno promítají a aktualizují neurony slov i n-gramů. To znamená, že jedna epocha *wordgram2vec* výpočetně odpovídá přibližně dvěma epochám původního *word2vec*. Nicméně trénovací korpus se projde pouze jednou a učící koeficient se snižuje rovnoměrně v rámci jednoho průchodu na rozdíl od původního algoritmu, kde se korpus prochází dvakrát a při druhém průchodu už je učící koeficient zredukovaný. Jako výsledné vektory slov se použijí pouze vektory z neuronů reprezentující slova.

Vektory n-gramů se do výsledných vektorů slov nijak nezahrnují. Jelikož jsou projekční vrsty nezávislé a trénování probíhá pro obě vstupní vrstvy samostatně, není možné tyto vektory jednoduše sečíst, jelikož tyto vektory mají jiný význam. Je možné je spojit k sobě, ale výsledek bude identický s modelem *wordgramprojected2vec* s dvojnásobnou velikostí vektorů.

4.1.4 Phasewordgram2vec

Toto rozšíření modifikuje předchozí *wordgram2vec* tak, že trénování slov a n-gramů probíhá ve dvou fázích (fázový model). Nejdříve se projde celý korpus a natrénují se n-gramy a poté se korpus projde znovu a natrénují se slova.

Tento přístup se ještě více podobá dvěma epochám původního *word2vec*, avšak učící koeficient se pro oba průchody nastavuje na původní hodnotu. V případě spuštění tohoto modelu s větším počtem epoch se nejdříve provede daný počet epoch pro n-gramy a poté daný počet epoch pro slova. Učící koeficient se rovnoměrně snižuje při n-gramovém průchodu, před slovním průchodem se nastaví opět na původní hodnotu a je znovu rovnoměrně snižován během slovního průchodu.

Toto rozšíření nabízí možnost při druhém průchodu předinicializovat hodnoty vektorů slov na součty vektorů jejich n-gramů. V opačném případě se nastaví hodnoty náhodně jako u obyčejného *word2vec*.

4.1.5 Wordgramprojected2vec

Jedná se o n-gramový model s promítanou architekturou, kde do první části projekční vrstvy jsou promítány vektory slova a do druhé části součet vektorů n-gramů daného slova. Výsledné vektory slov jsou poté tvořeny stejnou projekcí.

4.1.6 Stemfix2vec

První stemmovací rozšíření využívá ve vstupní vrstvě kombinaci stemů a koncovek (název z anglického **Stem**Suf**fix**). Byly vytvořeny všechny varianty architektur: základní - *stemfix2vec*, s oddělenými projekčními vrstvami bez fázového trénování - *separatestemfix2vec*, s fázovým trénováním - *phasestemfix2vec* a promítaná - *stemfixprojected2vec*. Všechny tyto modely mají stejné vlastnosti jako v případě n-gramových modelů, pouze výsledné vektory jsou vždy tvořeny kombinací vstupních vektorů a to z důvodu, že ani jeden ze vstupů nenese kompletní sémantickou a syntaktickou informaci.

4.1.7 Wordfix2vec a Wordstem2vec

Wordfix2vec (z anglického **Word**Suf**fix**) a *Wordstem2vec* jsou téměř shodné s *Stemfix2vec* se stejnou jmennou konvencí pro jednotlivé modely, pouze ve vstupní vrstvě místo stemů a koncovek využívá slova a koncovky popřípadě stemy. Výsledné vektory není nutné vytvářet, jelikož vstupní vrstva je tvořena těmito vektory. Pouze v případě promítaného modelu je nutné opět spojit vektory z obou vstupních vrstev.

4.2 Trénovací korpus

Pro účely trénování byly vytvořeny dva nové neoznačkované trénovací korpusy. Prvním korpusem je zpracovaná offline záloha české *Wikipedie* z 10.5.2016 (*CZ Wiki* korpus). Text byl nejdříve očištěn od značkovacího jazyka *Wikipedie* a ostatních *HTML* značek pomocí nástroje *wiki-dump-parser*[1]. Druhý korpus (*Nový CZ* korpus) obsahuje navíc ještě stažené zpravodajské stránky mezi roky 2011 a 2015 z konference strojového překladu v Berlíně (WMT16[2]).

Na tato data byly použity dva způsoby předzpracování. V obou případech se provedou následující kroky (převzaté z původního *word2vec*):

1. Odstranění neobvyklých znaků (např. *$, ** atd.)
2. Odsazení zbylých znaků od slov tak, aby tvořily samostatná slova
3. Nahrazení mnohonásobných mezer jednou mezerou
4. Všechna velká písmena převedena na malá

Tyto kroky jsou v práci označovány jako *částečné předzpracování*. Tento způsob zachovává interpunkci a konečný text je o 6% větší než text zpracovaný *kompletním předzpracováním*, které pokračuje dalšími dvěma kroky:

5. Odstranění všech ne-alfanumerických znaků
6. Spojení řádek

Pro ověření nových modelů na anglickém jazyce bylo nutné vygenerovat anglický korpus. K tomuto účelu byl použit skript přiložený k nástroji *word2vec*. V publikaci [30] však není přesně zmíněno, z kterých zdrojů byl tento korpus vytvořen. Proto byly vybrány zdroje tak, aby se výsledný korpus počtem slov (861M slov) co nejvíce podobal publikovanému středně velkému korpusu (783M slov). Jedná se o část složenou ze zpravodajských stránek z roků 2012 a 2013 ze stejné konference v Berlíně jako v případě Nového CZ korpusu.

Jelikož stemovací rozšíření potřebují předzpracovaný trénovací korpus, byly vytvořeny i ostemované verze korpusů. Jako stemmer byl použit HPS (viz kapitola *2.5.1 High Precision Stemmer*). Každé slovo korpusu bylo ostemováno a ve tvaru *kořen|koncovka* vloženo do nově vytvořeného trénovacího

[1] https://blog.afterthedeadline.com/2009/12/04/generating-a-plain-text-corpus-from-wikipedia/

[2] http://www.statmt.org/wmt16/translation-task.html

Předzpracování	Částečné	Kompletní
Velikost	636 MB	596 MB
Počet znaků	607 462 703	567 402 817
Počet slov	105 974 949	85 544 542
Počet 4-gramů	419 128 695	397 993 311
Počet článků	697 249	697 249
Počet unikátních slov	505 289	506 707
Počet unikátních 4-gramů	320 901	265 103
Počet unikátních stemů	295 361	*
Počet unikátních koncovek	10 139	*

Tabulka 4.1: Informace o *CZ Wiki* korpusu. Počty unikátních slov (4-gramů atd.) jsou spočteny přímo programem a nezahrnují slova (4-gramy atd.) s počtem výskytů menším než 5. * Jelikož se částečné předzpracování projevilo lépe než kompletní (viz kapitola *5 Výsledky*), nebyl vytvořen ostemovaný korpus pro kompletní předzpracování.

Předzpracování	Částečné	Kompletní
Velikost	4,7 GB	4,4 GB
Počet znaků	4 571 144 225	4 284 316 283
Počet slov	802 720 192	681 766 383
Počet 4-gramů	3 184 779 327	2 928 522 376
Počet unikátních slov	1 158 870	1 115 946
Počet unikátních 4-gramů	484 717	390 570
Počet unikátních stemů	580 300	*
Počet unikátních koncovek	17 911	*

Tabulka 4.2: Informace o *Novém CZ* korpusu. * Jelikož se částečné předzpracování projevilo lépe než kompletní (viz kapitola *5 Výsledky*), nebyl vytvořen ostemovaný korpus pro kompletní předzpracování.

korpusu a to i v případě, že dané slovo nemá koncovku (např. tečka má následující formu: ./).

4.3 Testovací datová sada

V rámci této práce byla také vytvořena obsáhlá testovací datová sada pro češtinu založená na podobnosti dvojic slov. Tato sada je složena z podmnožin všech prozkoumaných anglických sad popsaných v kapitole *2.4 Korpus* a navíc přidává nově vybraná slova podle tří různých kritérií (viz tabulka 4.3).

První přidaná kategorie *MT*[3] pochází z tabulek pro strojový překlad z anglického jazyka do češtiny. České překlady byly nejprve zlemmatizovány, aby se vybíraly pouze základní tvary slov. Poté se náhodně vybíraly dvojice slov z možných alternativ překladu pro náhodně vybrané anglické slovo. Například bylo vybráno anglické slovo *bag*, které je možné přeložit jako *taška*, *pytel*, *sáček* nebo *batoh* a z těchto alternativ se náhodně vybere dvojice *taška–sáček*. Tento přístup byl převzat z *ParaphraseDB*[11].

Další katogorie *SCIO* pochází z testů Obecně studijních předpokladů pro přijímací zkoušky na střední školy od společnosti *SCIO*[4], kde se nachází úlohy zabývající se podobností slov.

Poslední přidanou kategorií *Vlastní* jsou nově vymyšlené dvojice slov dvěma zkušenými pedagogy zabývajícími se českým jazykem. Každý pedagog na základě zhlédnutí předchozích dvojic, navrhl okolo nových 100 párů slov.

Anotace prováděl nezávisle na sobě 5 různých anotátorů. Mezi nimi byli i dva zmiňovaní padagogové. Každý dostal podrobné anotátorské instrukce (viz příloha *A Instrukce pro tvorbu datové sady*). Ke každému páru byly zadávány následující informace:

- Podobnost - Hodnoceno na stupnici od 0 do 5 (inspirováno *SemEval 2016 Task 2*[2]).
- Souvislost - Hodnoceno také na stupnici od 0 do 5.
- Kontext jednotlivých slov - Kontext, ve kterém se vyskytuje dané slovo ve správném sémantickém významu. Například ve dvojici *slepice–kohout*: *Kohout běhal po dvoře.*

[3] Z anglického **M**achine **T**ranslation.

[4] Společnost *SCIO* se zabývá tvorbou profesionálních testů pro přijímací zkoušky již od roku 1995. V současné době spolupracuje s velkým množstvím škol a s tvorbou těchto testů má dlouholeté zkušenosti. Použití těchto testů pro účely diplomové práce bylo schváleno po telefonickém rozhovoru s odpovědnou osobou společnosti.

- Víceznačnost - V případě, že je slovo víceznačné, byl získán kontext jiného sémantického významu. Například ve dvojici *slepice–kohout*: *Při odběru plynu, kdy je otevřen odběrový kohout.*

- Společný kontext obou slov - Kontext, ve kterém se objevují obě slova ve správném sémantickém významu. Maximální vzdálenost mezi slovy musí být menší než 20 slov. Například ve dvojici *slepice–kohout*: *Slepice a kohout běhali po dvoře.*

Pro zlepšení kvality kontextů a jejich rozmanitosti nebyly kontexty vymýšleny, ale byly získány pomocí nástroje *Korpus.cz* a korpusu *SYN*[15].

V první řadě bylo nutné u převzatých dvojic přeložit slova z angličtiny do češtiny. Jelikož slova lze přeložit často mnoha způsoby, byli anototáři instruováni, aby překládali pokud možno jako jednoslovné podstatné jméno v jednotném čísle a používali základní tvary slov (lemma). V případě, že je možné přeložit slovo mnoha významy, tak vybrat význam nejvíce podobný druhému slovu v páru. I přesto se v celé řadě případů anotátoři neshodli a bylo nutné tato slova vyřadit. Jelikož testovací datová sada porovnává pouze dvojice samostatných slov a nezábývá se frázemi, tak další problém nastal v případě, že slovo není možné jednoslovně přeložit do češtiny (např. *seafood - plody moře*, *postcode - poštovní směrovací číslo* nebo *hyperlink - hypertextový odkaz* atd.). Tato slova byla také vyřazena společně se slovy, která nebyli anototáři schopni přeložit ani za pomocí slovníků (např. *housefull*, *caesarism* nebo *nonconscious* atd.). Jednalo se především o slova z *Rare Words*.

V druhém kroku se určovala shoda a spolehlivost anotátorů. Vzájemná shoda anotátorů byla určena pomocí *Fleissovova kappa koeficientu* a pro podobnost vyšla $\kappa_{Fleiss} = 0,21$. Tato hodnota je definována jako uspokojivá (fair) shoda, ale v tomto případě je to relativně velká hodnota. Pro porovnání byla vypočtena tato míra i u nejpoužívanější dostupné anglické sady *Wordsim*. $\kappa_{FleissWordsim} = 0,12$, což symbolizuje špatnou (poor) shodu. Tato míra není pro tento případ užití příliš vhodná, jelikož skóre 4 a 5 jsou považovány za neshodu stejně velkou, jako v případě 0 a 5. Proto byla shoda anotátorů měřena také pomocí *Pearsonovi korelace*, která byla pro tento účel použita i v případě RG datové sady. Celková korelace byla vypočtena jako aritmetický průměr korelací všech dvojic anotátorů a vyšla $r = 0,71$. Avšak v porovnání s ostatními sadami tato hodnota není příliš vysoká $r_{Wordsim} = 0,72$ a $r_{RG} = 0,85$. Samozřejmě je nutné brát v úvahu velikosit datový sad a počty anotátorů.

V následujícím kroku byly vyřazeny sporné páry z důvodu zvýšení shody anotátorů a tím i kvality celé sady. Sporný pár byl definován následovně:

Datová sada	Vybrané	Shodně přeložené	Použité
Miller	30	27	20
RG	36	31	26
Wordsim	322	288	205
MTurk	150	123	97
MEN	150	141	121
Rare Words	150	119	85
MT	150	150	108
Scio	150	150	118
Vlastní	227	227	173
Celkem	1365	1256	**953**

Tabulka 4.3: Složení datové sady. Jedná o počty párů z jednotlivých sad. Poslední tři kategorie (nově přidané kategorie) již obsahovaly české páry, a proto je počet vybraných párů shodný s počtem shodně přeložených párů.

- pár ohodnocen více než 3 různými skóre
- pár obsahující zároveň skóre 0 nebo 1 a 4 nebo 5

Datová sada obsahovala 303 sporných párů a po jejich vyřazení je konečný počet párů v sadě 953 (konečné rozložení použitých sad viz tabulka 4.3). Díky tomuto kroku se povedlo navýšit jak kappa koeficient $\kappa_{Fleiss} = 0,29$, tak i korelaci $r = 0,81$, která už je v této podobě srovnatelná s korelací dosáhnutou v případě RG datové sady.

V případě souvislosti je situace horší, jelikož souvislost je více závislá na uvážení anotátora. Jednotlivé hodnoty vyšly $\kappa_{Fleiss} = 0,16$ a $r = 0,66$. V tomto případě se sporné páry nechaly v datové sadě, jelikož tato sada je primárně určena na podobnost mezi slovy.

Na závěr byla ještě určena spolehlivost jednotlivých anotátorů. Jedná se o případ, kdy se anotátor shodne sám se sebou. Každý anotátor obdržel 14 dní po hodnocení 100 náhodně vybraných párů z datové sady, které již jednou hodnotil a měl doplnit znovu hodnocení. Kappa byla počítána pomocí *Cohenovo kappa koeficientu* a zprůměrována přes všechny anotátory stejně jako *Pearsonova korelace*. Pro podobnost vyšly hodnoty $\bar{\kappa}_{Cohen} = 0,41$ a $\bar{r} = 0,80$ a pro souvislost $\bar{\kappa}_{Fleiss} = 0,32$ a $\bar{r} = 0,67$. Je vidět, že v případě kappa koeficientu se anotátor shodne sám se sebou více než s ostatními anotátory, avšak tato změna není příliš markantní a v případě korelace je tato hodnota téměř stejná.

Skóre	Podobnost	Souvislost
0	263	13
1	202	17
2	127	112
3	175	134
4	123	342
5	63	275

Tabulka 4.4: Rozložení hodnot podobnosti a souvislosti.

Anotátoři	1	2	3	4	5
Podobnost					
1	0,73	0,80	0,78	0,78	0,81
2	0,80	0,92	0,83	0,82	0,81
3	0,78	0,83	0,79	0,81	0,82
4	0,78	0,82	0,81	0,76	0,81
5	0,81	0,81	0,82	0,81	0,83
Souvislost					
1	0,32	0,62	0,53	0,62	0,69
2	0,62	0,85	0,59	0,71	0,75
3	0,53	0,59	0,60	0,65	0,67
4	0,62	0,71	0,65	0,75	0,81
5	0,69	0,75	0,67	0,81	0,81

Tabulka 4.5: Vzájemná shoda mezi jednotlivými anotátory určena pomocí *Pearsonovy korelace*. Hodnoty na diagonále určují, jak se anotátoři shodli sami se sebou.

4.4 Evaluační skripty

K evaluaci je používán původní nástroj dodávaný s *word2vec* (*compute-accuracy*). Byly provedy pouze malé změny pro podporu české datové sady. Nyní je možné specifikovat počet kategorií sémantických otázek, který byl původně fixně nastaven na hodnotu 5. Aby nemusely být prováděny žádné další konverze během evaluace, předpokládá se, že vstupní otázky budou obsahovat pouze malá písmena.

Dále byla vytvořena modifikace tohoto skriptu s podporou dvou nacvičených modelů, která umožnuje kombinovat jednotlivé modely (*models-accuracy)*. Vektory je možné kombinovat dvěma způsoby. Buďto sečtením nebo spojením za sebe dvou odpovídajících vektorů pro stejné slovo v obou modelech. V případě, že slovo chybí ve slovníku jednoho z nich, je toto slovo vynecháno z evaluace a otázka považována za neviděnou.

Pro evaluaci nové české datové sady založené na podobnosti slov byl vytvořen skript *compute-sim*. Tento skript počítá kosínové vzdálenosti, které nakonec pomocí *Pearsonovi* a *Spearmanovi* korelace porovná s hodnotami určenými anotátory. Tyto korelace jsou počítány pomocí knihovny *ALGLIB*[5].

[5]http://www.alglib.net

5 Výsledky

K trénování testovaných modelů se používaly následující trénovací korpusy (podrobné informace v tabulce 5.1):

- CZ Wiki - Nově vytvořený korpus z české Wikipedie
- Nový CZ - Nový velký český korpus
- Obsáhlý CZ - Velice obsáhlý český korpus převzatý z [43]
- Mikolov EN - Anglický korpus používaný v [30]. Použit k porovnání rozšíření na anglickém jazyce.

Z důvodů rychlejšího trénování se obvykle omezuje používaný slovník na jeden milion nejfrekventovanějších slov v korpusu. Ostatní slova jsou často překlepy a velice vzácná slova nebo cizí slova, která nejsou příliš důležitá k běžnému použití. V tabulce 5.1 je vidět, že je takto omezen Obsáhlý CZ korpus. Nový CZ korpus také přesahuje hranici jednoho milionu slov, ale pouze v malé míře a urychlení by bylo zanedbatelné, proto byl ponechán slovník nepozměněn. K ověřování modelů natrénovaných na těchto korpusech byly použity následující testovací datové sady:

- CZ Analogy - Česká datová sada založená na analogii slov z [43]
- CZ Sim - Nově vytvořená datová sada založená na podobnosti dvojic slov
- EN Analogy - Původní anglický *word2vec* testovací korpus založený na analogii slov[30]

CZ Analogy sada byla upravena pro účely této práce. V původní formě sada obsahuje slova spojená do frází, které nejsou předmětem této práce, proto byla sada rozdělena na část se slovy a frázemi podobně jako *EN Analogy* sada. Navíc tato sada v původní podobě obsahuje fráze, které se ani v trénovacím korpusu nemohou vyskytnout. Např. fráze *jindřich_i._lucemburský* se nemůže vyskytnout v korpusu, protože na trénovací korpus použitý ve zmiňované práci byl spuštěn nástroj *word2phrase* pouze jednou, tudíž není možné vytvořit fráze o třech slovech a navíc je korpus předzpracován tak, že jsou tečky odděleny od slov mezerou. To znamená, že fráze by musela vypadat *jindřich_i_._lucemburský* a nástroj *word2phrase* by se musel na korpus pustit třikrát.

Název korpusu	Velikost (GB)	Počet slov (M)	Velikost slovníku (M)
CZ Wiki	0,6	106	0,5
Nový CZ	4,7	803	1,1
Obsáhlý CZ	35	4 888	1,0*
Mikolov EN	4,2	861	0,4

*Manuálně omezeno z 1,6M

Tabulka 5.1: Podrobné informace o jednotlivých korpusech.

Při testování je nutné brát v úvahu fakt, že nástroj *word2vec* běží obvykle v několika vláknech a neřeší synchronizaci mezi vlákny. Chyby vzniklé špatnou synchronizací jsou považovány za šum a nemají příliš velký vliv na konečný výsledek, protože hodnoty jsou modifikovány často, ale jde pouze o malé změny. Předpokládá se, že chyba vektoru bude opravena další modifikací. Díky tomu, že není nutné řešit synchronizaci, je algoritmus rychlejší a je možné použít více trénovacích dat[27].

Tento předpoklad v souvislosti s používáním generátoru náhodných čísel způsobuje, že natrénovaný model může dávat různé výsledky při stejné konfiguraci, a to až v rámci 1% (viz tabulka 5.2). Je možné tomu zabránit v případě použití pouze jednoho vlákna a pevného semínka[1] generátoru náhodných čísel. Avšak testování za těchto podmínek je časově nepřijatelné.

5.1 Porovnání předzpracování

Z tabulky 5.3 je vidět, že ponechaná interpunkce (viz kapitola *3.1 Předzpracování*) má v češtině velký vliv. Naopak v angličtině jsou výsledky srovnatelné. Na základě těchto výsledků jsou veškerá následující měření prováděna na trénovacích korpusech připravených pomocí částečného předzpracování.

Z tabulky je také vidět, že *Obsáhlý CZ* korpus z [43] není vhodný pro trénování sémantických modelů. Kvalita textu byla obětována za cenu velkého množství dat a obsahuje velké množství zašuměného textu (opakující se odkazy z wikipedie atd.). Z tohoto důvodu na něm nebyla prováděna žádná další měření.

V tabulce je také uveden poměr viděných otázek ke všem otázkám. Je zřejmé, že anglická datová sada obsahuje spíše obvyklá slova, jelikož při velikosti slovníku 0,4M je pokrytí 99,70% datové sady. Narozdíl od české datové sady, kde je při velikosti slovníku 1,1M pokrytí 98,52%. Také je možné vidět,

[1] Z anglického *seed*.

Číslo běhu	Sém. p.	Syn. p.	Celková p.
Word2vec			
1.	26,88	**58,65**	**49,63**
2.	27,75	58,22	49,63
3.	27,40	57,64	49,11
4.	**27,94**	57,99	49,52
5.	26,61	57,88	49,06
Wordgramprojected2vec			
1.	20,59	73,71	58,74
2.	21,08	73,23	58,52
3.	21,30	**74,60**	**59,58**
4.	21,84	72,29	58,07
5.	**23,06**	72,85	58,81

Tabulka 5.2: Různé výsledky pro různé běhy programů při stejné konfiguraci (viz *Vybraná konfigurace* v kapitole *5.2 Konfigurace parametrů*). Trénováno na Novém CZ korpusu pro dvě epochy. Naměřeno na původním modelu *word2vec* a nově navrženém modelu s nejlepšími výsledky *wordgramprojected2vec*.

že při poloviční velikosti slovníku (0,5M v případě CZ Wiki korpus) je stále dost vysoké pokrytí 92,66%.

5.2 Konfigurace parametrů

Vybrané parametry (konfigurace) programu byly zvoleny na základě doporučení přímo z oficiálních stránek *word2vec*[2] není-li uvedeno jinak:

- podvzorkování (sample) - 10^{-3}
- velikost okna (window) - 10
- počet negativiních vzorků (negative) - 10 ([32])
- model - Skip-gram
- velikost vektorů - 300 ([43])
- míra učení - 0.025 pro Skip-gram a 0.05 pro CBOW
- velikost n-gramu - 4 (zvoleno na základě odhadu a poté ověřeno)

[2]https://code.google.com/p/word2vec/

Název korpusu	Předzpracování	Sém. p.	Syn. p.	Celková p.	Viděno
Nový CZ	Částečné	25,53	55,47	**46,75**	98,52
Nový CZ	Kompletní	23,70	51,44	43,62	98,52
CZ Wiki	Částečné	13,10	27,81	**23,67**	92,66
CZ Wiki	Kompletní	12,24	25,58	21,83	92,66
Mikolov EN	Částečné	59,01	60,61	59,88	99,70
Mikolov EN	Kompletní	59,58	60,20	**59,92**	99,70
Obsáhlý CZ	Částečné	3,83	4,65	**4,42**	97,00
Obsáhlý CZ	Kompletní	1,11	10,41	3,36	97,00

Tabulka 5.3: Porovnání předzpracování na všech korpusech. Jedná se o obyčejný *word2vec* při jedné epoše trénování (viz *Vybraná konfigurace* v kapitole *5.2 Konfigurace parametrů*).

- ohraničení slov k rozpoznání krajních ngramů - ano (zvoleno na základě odhadu a poté ověřeno)

V tabulce 5.4 je tato konfigurace ověřena na českém jazyce a porovnávána s různými změnami parametrů. Pro ověření n-gramových rožšíření je použit model *wordgramprojected2vec* poskytující nejlepší výsledky. Je vidět, že *Hierarchický softmax* poskytuje lepší výsledky na sémantické podobnosti, ale syntaktická podobnost je značně horší, proto byl vybrán způsob trénování pomocí negativního vzorkování. Větší počet negativních vzorků poskytuje lehce vyšší syntaktickou přesnost, ale naopak horší sémantickou přesnost. Větší okno poskytuje zase lepší sémantickou přesnost a horší syntaktickou přesnost. I přesto, že větší počet negativních vzorků má lepší celkovou přesnost, tento nárůst není tak znatelný a časový nárůst je příliš vysoký, proto byla nakonec zvolena *vybraná konfigurace*.

V případě n-gramů se ověřily zvolené předpoklady a to jak jejich velikost, tak rozlišování krajních n-gramů (viz kapitola *3.3 Rozšíření modelů*). Větší velikost n-gramů sice podává vyšší sémantickou přesnost, ale syntaktická přesnost je znatelně horší, proto byla nakonec vybrána velikost čtyři. V dalších výsledcích je použita tato *vybraná konfigurace*, není-li určeno jinak.

5.3 Porovnání rozšíření

Všechna naimplementovaná rozšíření byla nejdříve otestována a odlazena na menším *CZ Wiki* korpusu a až poté otestována na velkém českém *Novém CZ korpusu* a anglickém *Mikolov EN korpusu*.

N-gramová rozšíření se osvědčila v případě menšího množství dat a syn-

Verze	Sém. p.	Syn. p.	Celková p.	Čas
Word2vec				
Vybraná konfigurace	13,10	27,81	23,67	22m
Negativní vzorkování 5	11,35	25,03	21,18	15m
Negativní vzorkování 15	12,51	**29,21**	**24,51**	30m
Okno 5	10,53	27,75	22,90	12m
Okno 15	13,88	25,26	22,06	28m
Dimenze 100	10,10	28,53	23,34	**9m**
Dimenze 500	10,78	25,41	21,29	34m
Hierarchický softmax	**14,15**	24,71	21,74	25m
Wordgramprojected2vec				
Vybraná konfigurace	13,89	**67,02**	**52,07**	35m
3-gramy	8,50	63,75	48,20	47m
5-gramy	**14,98**	58,88	46,52	49m
Bez krajních n-gramů	11,58	52,65	41,09	**32m**

Tabulka 5.4: Konfigurace parametrů pro český jazyk na CZ Wiki korpusu a jedné epoše trénování. Jednotlivé výsledky obsahují zmíněnou změnu v konfiguraci oproti *Vybrané konfiguraci*.

taktické přesnosti. Z tabulky 5.5 je vidět, že jeden průchod *ngram2vec* podává více než dvojnásobně lepší výsledky na syntaktických otázkách než obyčejný *word2vec*. Bohužel sémantická přesnost se na úkor syntaktické lehce zmenšila, ale celková úspěšnost je díky syntaktické přesnosti také dvojnásobná. Naopak *phasewordgram2vec* s předinicializovanými vektory slov pomocí n-gramových vektorů lehce převyšuje dvě epochy obyčejného *word2vec*. Bohužel tento nárůst není příliš znatelný. Na druhou stranu lepší výsledky podává i v případě syntaktické přesnosti a tudíž celková úspěšnost je okolo 2% vyšší než u *word2vec*. Nejlépší výsledky však podává *wordgramprojected2vec*, který obsahuje ve výsledných vektorech jak slovní vektory, tak kombinaci n-gramových vektorů, díky tomu výsledky obsahují nejlepší hodnoty jak sémantické přesnosti, tak v syntaktické přesnosti. Celkovou úspěšnost se díky tomuto modelu povedlo navýšit až o 22%.

Zajímavé jsou výsledky rozšíření *ngramonly2vec*, které i přesto, že během trénování nepoužívá informace o celém slově, dosahuje srovatelných výsledků s ostatními metodami. Bohužel rozšíření *wordgram2vec* se moc neosvědčilo a dosahuje nejhorších výsledků. Jelikož *phasewordgram2vec* poskytuje lepší výsledky s předinicializovanými vektory, není v dalších výsledcích zahrnuta jeho verze bez této inicializace.

Druhá sada rozšíření založených na stemmeru (tabulka 5.6) se osvědčila

Verze	Sém. p.	Syn. p.	Celková p.	Epochy
word2vec - 1 epocha	13,10	27,81	23,76	1
word2vec - 2 epochy	16,65	38,23	32,16	2
ngramonly2vec	8,43	62,47	47,26	1
ngramonly2vec - 2 epochy	10,89	66,82	51,08	2
ngram2vec	10,32	66,51	50,70	1
ngram2vec - 2 epochy	13,88	69,35	53,74	2
wordgramprojected2vec	13,80	67,02	52,07	1
wordgrampr...2vec - 2 ep.	**16,99**	**69,45**	**54,69**	2
wordgram2vec	13,58	29,41	24,95	2
phasewordgram2vec	15,73	35,52	29,95	2
phase...2vec init-vectors	16,75	41,26	34,36	2

Tabulka 5.5: Výsledky n-ngramových rozšíření na CZ Wiki korpusu.

pouze v případě *wordfix2vec*, které podává lepší výsledky na sémantické podobnosti v rámci 1%. Tento výsledek však degraduje fakt, že syntaktická podobnost je nižší a celková úspěšnost je kvůli tomu také nižší. Rozšíření *separatestemfix2vec* podává lepší výsledky pro syntaktickou přesnost, ale bohužel velice špatné výsledky v případě sémantické přesnosti. Také tyto výsledky na syntaktické přesnosti nepřevyšují n-gramové rozšíření. Rozšíření *wordstem2vec* se nejvíce osvědčilo ve verzi *wordstemprojected2vec* s promítanou projekční vrstvou. Poskytuje lepší celkovou úspěšnost o necelé 2% než původní *word2vec*.

Bohužel ani jedno ze stemmovacích rozšíření není tak úspěšné jako v případě n-gramových rozšíření. Nicméně rozšíření *wordfix2vec* poskytuje nejlepší sémantickou přesnost ze všech rozšíření. Do dalších měření je ještě zahrnuto rozšíření *stemfixprojected2vec*, které podává srovnatelné výsledky s ostatními rozšířeními.

Jelikož promítané modely mají ve vstupní vrstvě pouze poloviční dimenzi vektorů (viz kapitola *3.3 Rozšíření modelů*), je možné, že tyto modely jsou tím penalizované, a proto byly naměřeny výsledky v tabulce 5.7 s dostatečně velkou vstupní a projekční vrstvou. V tabulce je vidět, že tato domněnka je mylná a modely s větší projekční vrstvou se chovají stejně jako původní *word2vec*, tudíž úspěšnost ve všech směrech klesá.

V případě většího korpusu (viz tabulka 5.8) šla úspěšnost podle očekávání nahoru, avšak nárůst u *word2vec* je znatelnější než u n-gramových rozšíření. Syntaktická přesnost vzrostla u *word2vec* o 20% v porovnání s *wordgramprojected2vec* pouze o 5%. Avšak na větších datech *wordgramprojected2vec* modeluje znatelně lépe syntaxi než *ngram2vec*. Naopak sémantická přesnost

Verze	Sém. p.	Syn. p.	Celková p.	Epochy
word2vec - 1 epocha	13,10	27,81	23,76	1
word2vec - 2 epochy	16,65	38,23	32,16	2
stemfix2vec	1,34	21,13	15,56	1
stemfix2vec - 2 epochy	1,53	20,79	15,37	2
wordfix2vec	12,13	28,82	24,12	1
wordfix2vec - 2 epochy	**17,46**	36,65	31,25	2
wordstem2vec	3,39	4,22	3,98	1
wordstem2vec - 2 epochy	5,26	5,54	5,46	2
separatestemfix2vec	5,82	**58,05**	**43,35**	2
separatewordfix2vec	12,96	26,45	22,65	2
separatewordstem2vec	14,32	30,58	26,00	2
phasestemword2vec	8,19	23,31	19,05	2
phasestemfix2vec	3,39	53,33	39,44	2
phasewordfix2vec	6,33	17,51	14,36	2
wordstemprojected2vec	8,72	35,48	27,95	1
wordstemproj..2 epochy	10,93	43,00	33,98	2
stemfixprojected2vec	4,97	52,41	39,06	1
stemfixproj...2 epochy	5,67	53,78	40,24	2
wordfixprojected2vec	8,85	27,81	22,47	1
wordfixproj...2 epochy	12,26	38,47	30,26	2

Tabulka 5.6: Výsledky všech stemmovacích rozšíření na CZ Wiki korpusu.

Verze	Sém. p.	Syn. p.	Celková p.	Dimenze
word2vec	13,10	27,81	23,67	300
stemfixprojected2vec	4,97	52,41	39,06	300
wordfixprojected2vec	8,85	27,81	22,47	300
wordstemprojected2vec	8,72	35,48	27,95	300
wordgramprojected2vec	**13,80**	**67,02**	**52,07**	300
word2vec	11,58	24,61	20,94	600
stemfixprojected2vec	3,51	52,10	38,43	600
wordfixprojected2vec	8,08	23,54	19,19	600
wordstemprojected2vec	4,97	27,09	20,87	600
wordgramprojected2vec	10,82	64,81	49,62	600

Tabulka 5.7: Porovnání stemmovacích rozšíření s rozdělenou projekční vrstvou (*projected* verze rozšíření) s dostatečně velkou projekční vrstvou o velikosti 600. Pro porovnání jsou zobrazeny i výsledky s velikostí projekční vrstvy 300. Všechna měření byla provedena na CZ Wiki korpusu a jedné epoše trénování.

už není srovnatelná s původním *word2vec* a je znatelně horší. *Phasewordgram2vec* je na vetších datech bohužel o 3% horší než *word2vec*. *Wordfix2vec* stále poskytuje lepší výsledky než *word2vec*, ale stále pouze v rámci 1%. Ostatní stemovací rozšíření neposkytují výrazné zlepšení ani v případě velkého korpusu.

V tabulce 5.8 jsou také porovnány rozdílné přístupy trénování *Skip-gram* a *CBOW*. Největší rozdíl je v případě *ngramonly2vec*, který poskytuje lepší syntaktickou přesnost než *ngram2vec* i *wordgramprojected2vec* a v porovnání pouze s *ngram2vec* i v případě sémantické přesnosti. Dále se příliš neosvědčil *phasewordgram2vec* s inicializovanými vektory v kombinaci s *CBOW*. Obecně mají modely natrénované pomocí *CBOW* horší sémantickou přesnost a syntaktická přesnost se odvíjí od jednotlivých modelů, ale pro n-gramové rozšíření je bohužel značně nižší.

V tabulce 5.9 jsou naměřeny kombinace jednotlivých modelů. První kombinace je složena z modelu s nejlepší sémantickou přesností (*wordfix2vec*) a s nejlepší syntaktickou přesností (*wordgramprojected2vec*). Jelikož *wordfix2vec* převyšuje v sémantické přesnosti původní model pouze v malé míře, další kombinace obsahuje místo *wordfix2vec* obyčejný *word2vec*. Poslední kombinace je složena z *ngram2vec* a *word2vec*, aby bylo možné porovnat nezávislé trénování n-gramů a slov se společným trénováním v případě *wordgramprojected2vec*. Jelikož modely s dvojnásobnou dimenzí vektorů podávají horší výsledky (viz tabulky 5.4 a 5.7), nemá smysl při porovnávání zdvojnásobovat velikost obyčejných modelů, ale raději je trénovat ve dvou epochách (viz kapitola *3.4 Evaluace*).

Ačkoli samostatně dávají nejlepší výsledky modely *wordfix2vec* a *wordgramprojected2vec*, dohromady dává nejlepší výsledky kombinace *ngram2vec* a *word2vec*. Pouze sémantická přesnost je v případě první kombinace vyšší. Z porovnání *wordgramprojected2vec* s kombinovaným modelem *ngram+word2vec* je možné usoudit, že nezávislé trénování slov a n-gramů podává lepší výsledky, než když se během trénování vzájemně ovlivňují.

V tabulce 5.10 je možné vidět výsledky na jednotlivých typech otázek: Sémantické kategorie:

- AN - Antonyms-nouns - Protiklady podstatných jmen
- AD - Antonyms-adjectives - Protiklady přídavných jmen
- AV - Antonyms-verbs - Protiklady sloves
- SC - States-cities - Státy a města
- FR - Family-relations - Rodinné vztahy

Verze	Sém. p.	Syn. p.	Celková p.	Epochy
Skip-gram				
word2vec - 1 epocha	24,53	55,47	46,75	1
word2vec - 2 epochy	26,68	58,65	49,63	2
ngramonly2vec	11,80	69,38	53,15	1
ngramonly2vec - 2 epochy	12,82	70,90	54,53	2
ngram2vec	14,62	70,25	54,56	1
ngram2vec - 2 epochy	15,29	71,64	55,75	2
wordgram2vec	22,96	54,35	45,50	2
wordgramprojected2vec	20,51	72,68	57,97	1
wordgramproj... 2 ep.	21,30	**74,60**	**59,58**	2
phase...2vec init-vectors	24,95	54,81	46,39	2
separatestemfix2vec	5,27	59,07	43,90	2
wordfix2vec	24,81	55,74	47,02	1
wordfix2vec - 2 epochy	**27,06**	58,32	49,51	2
wordstemprojected2vec	17,31	55,68	44,87	1
wordstemproj... 2 epochy	19,75	59,27	48,13	2
stemfixprojected2vec	5,15	52,09	38,86	1
stemfixproj... 2 epochy	5,01	51,30	38,25	2
CBOW				
word2vec - 1 epocha	21,41	58,68	48,17	1
word2vec - 2 epochy	23,46	60,91	50,36	2
ngramonly2vec	7,50	68,88	51,58	1
ngramonly2vec - 2 epochy	8,83	**71,42**	53,77	2
ngram2vec	6,51	67,85	50,56	1
ngram2vec - 2 epochy	8,10	70,78	53,11	2
wordgram2vec	19,49	56,37	45,97	2
wordgramprojected2vec	7,52	69,64	52,13	1
wordgramproj... 2 ep.	9,97	71,18	**53,93**	2
phase...2vec init-vectors	13,98	56,61	44,59	2
separatestemfix2vec	6,82	48,75	36,93	2
wordfix2vec	21,04	57,87	47,49	1
wordfix2vec - 2 epochy	**23,76**	60,07	49,83	2
wordstemprojected2vec	12,92	61,08	47,50	1
wordstemproj... 2 epochy	17,00	64,55	51,14	2
stemfixprojected2vec	3,90	40,10	29,90	1
stemfixproj... 2 epochy	4,80	43,35	32,48	2

Tabulka 5.8: Výsledky analogické datové sady na Novém CZ korpusu.

Verze	Sém. p.	Syn. p.	Celková p.	Epochy
word	26,68	58,65	49,63	2
wordgramprojected	21,30	74,60	59,58	2
wordfix+wordgramproj...	**28,46**	70,43	58,60	1+1
wordgramproj+word	27,99	70,32	58,38	1+1
ngram+word	26,74	**75,57**	**61,81**	1+1

Tabulka 5.9: Kombinace modelů na Novém CZ korpusu. U modelů není zmíněna koncovka *2vec*.

Syntaktické kategorie:

- NP - Nouns-plural - Množná čísla podstatných jmen
- J - Jobs - Povolání
- VP - Verb-past - Minulé časy sloves
- P - Pronouns - Zájmena
- AAG - Antonyms-adjectives,gradation - Stupňování přídavných jmen
- N - Nationalities - Národnosti

Je vidět, že na každou kategorii se hodí více jiné rozšíření, v sémantické přesnosti dominují ve stejné míře *word2vec* a *wordfix2vec* a v syntaktické přesnosti dominuje hlavně *wordgramprojected2vec*. V případě *wordgram2vec*, který se v celkovém hodnocení příliš neosvědčil, dominuje v kategorií *Protiklady sloves*. Zajímavý je také fakt, že *wordfix2vec* podává v jedné syntaktické kategorii nejlepší výsledky, kde jinak dominují jednoznačně n-gramová rozšíření. Původní *word2vec* se povedlo překonat ve všech kategoriích kromě *Státy a města* a *Národnosti*.

Výsledky na nové testovací datové sadě potvrdily výsledky z analogické datové sady v sémantické přesnosti. V případě *Skip-gramu* vychází nejlépe *word2vec*, ale *wordfix2vec* a *phasewordgram2vec* podávají srovnatelné výsledky v rámci 0,25%. Pro *CBOW* jsou výsledky také stejné. Avšak tyto výsledky vyvracují domněnku, že *phasewordgram2vec* v kombinaci s *CBOW* nefunguje.

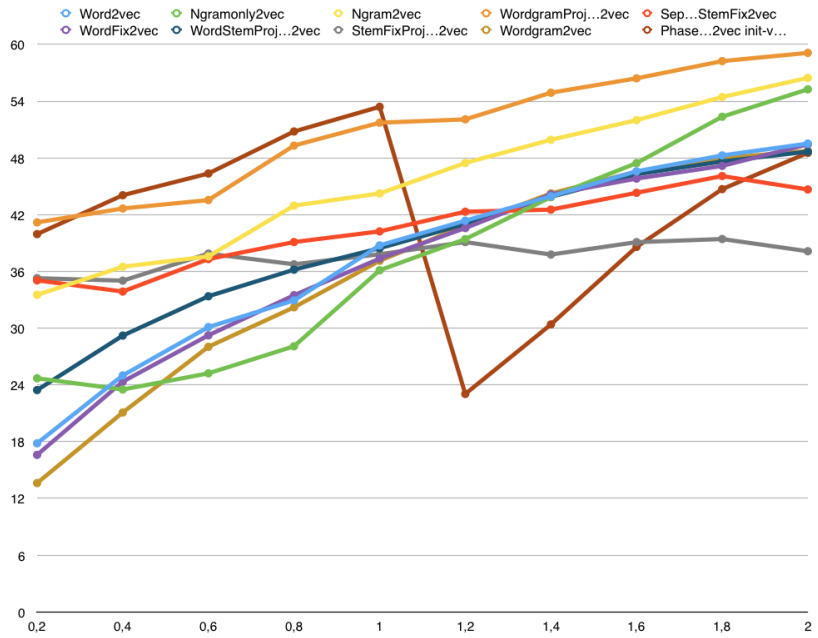
Na obrázku 5.1 (větší náhled v příloze na obrázku C.1) je zobrazen průběh trénování jednotlivých modelů na dvou epochách. Potvrdil se předpoklad, že v případě menšího počtu slovních tvarů je sémantická podobnost modelována lépe než v případě celých slov a to zejména při malém množství trénovacích dat. *Wordgramprojected2vec* má úspěšnost 42% už po první pětině trénovacího korpusu. Naopak původní model během počátku trénování

Verze							Epochy
Sémantické kategorie	AN	AD	AV	SC	FR		
word2vec - 1 e.	20,55	21,08	6,36	44,13	42,73		1
word2vec - 2 e.	**20,84**	22,42	6,74	**52,08**	44,36		2
ngramonly2vec	13,80	8,13	3,32	17,99	22,55		1
ngramonly2vec - 2 e.	16,15	9,81	3,98	21,40	25,27		2
ngram2vec	16,71	13,24	3,42	20,55	23,64		1
ngram2vec - 2 e.	17,14	11,32	3,04	24,24	29,27		2
wordgram2vec	17,71	20,27	**7,21**	42,05	38,36		2
wordgramprojected2vec	20,55	17,25	3,89	35,98	32,73		1
wordgramproj... 2 ep.	18,63	14,81	4,84	44,32	35,82		2
phase...2vec init.	20,34	17,89	7,02	47,44	40,36		2
separatestemfix2vec	4,05	4,53	0,95	7,20	15,27		2
wordfix2vec	20,34	23,05	5,79	42,99	43,27		1
wordfix2vec - 2 e.	20,77	**23,23**	6,45	51,89	**46,91**		2
wordstemprojected2vec	10,31	10,57	2,47	42,61	36,18		1
wordstemproj... 2 epochy	11,52	12,78	3,61	46,40	42,36		2
stemfixprojected2vec	2,42	3,72	0,57	11,27	13,64		1
stemfixproj... 2 e.	3,20	4,07	1,71	9,85	9,64		2
Syntaktické kategorie	NP	J	VP	P	AAG	N	
word2vec - 1 e.	65,69	70,03	60,99	10,85	50,00	19,22	1
word2vec - 2 e.	68,24	77,69	62,06	11,24	60,53	27,56	2
ngramonly2vec	61,56	64,14	81,54	15,34	71,05	18,66	1
ngramonly2vec - 2 e.	65,69	71,80	82,54	**21,03**	71,05	21,78	2
ngram2vec	58,18	76,60	82,65	14,02	65,79	19,98	1
ngram2vec - 2 e.	59,68	78,96	**84,04**	19,18	63,16	23,11	2
wordgram2vec	66,67	69,87	57,89	11,24	52,63	24,72	2
wordgramprojected2vec	67,42	**88,64**	80,81	14,68	**76,32**	**28,98**	1
wordgramproj... 2 ep.	40,54	37,29	65,12	7,80	36,84	15,25	2
phase...2vec init.	62,24	72,56	56,79	11,64	57,89	26,70	2
separatestemfix2vec	57,73	46,63	75,40	9,39	28,95	14,49	2
wordfix2vec	66,97	71,46	60,50	11,51	52,63	19,70	1
wordfix2vec - 2 e.	**70,05**	78,79	61,54	13,10	57,89	26,33	2
wordstemprojected2vec	53,98	67,68	64,00	19,18	39,47	22,92	1
wordstemproj... 2 epochy	58,26	74,33	66,16	16,67	50,00	28,88	2
stemfixprojected2vec	48,12	39,90	64,83	10,45	34,21	17,99	1
stemfixproj... 2 e.	40,54	37,29	65,12	7,80	36,84	15,25	2

Tabulka 5.10: Výsledky samostatných kategorií analogické datové sady na Novém CZ korpusu.

Verze	Pearson. kor.	Spearman kor.	Epochy
Skip-gram			
word2vec - 1 epocha	0,6466	0,6652	1
word2vec - 2 epochy	**0,6583**	**0,6818**	2
ngramonly2vec	0,5824	0,6098	1
ngramonly2vec - 2 epochy	0,5939	0,6227	2
ngram2vec	0,6060	0,6316	1
ngram2vec - 2 epochy	0,6183	0,6457	2
wordgram2vec	0,6335	0,6577	2
wordgramprojected2vec	0,6288	0,6542	1
wordgramproj... 2 ep.	0,6381	0,6625	2
phase...2vec init-vectors	0,6560	0,6810	2
separatestemfix2vec	0,5004	0,5165	2
wordfix2vec	0,6440	0,6642	1
wordfix2vec - 2 epochy	0,6558	0,6800	2
wordstemprojected2vec	0,6339	0,6555	1
wordstemproj... 2 epochy	0,6466	0,6712	2
stemfixprojected2vec	0,6104	0,6288	1
stemfixproj... 2 epochy	0,5525	0,5678	2
CBOW			
word2vec - 1 epocha	0,5892	0,6084	1
word2vec - 2 epochy	0,6132	0,6337	2
ngramonly2vec	0,5827	0,6071	1
ngramonly2vec - 2 epochy	0,5917	0,6119	2
ngram2vec	0,5542	0,5694	1
ngram2vec - 2 epochy	0,5613	0,5758	2
wordgram2vec	0,5742	0,6010	2
wordgramprojected2vec	0,5576	0,5711	1
wordgramproj... 2 ep.	0,5791	0,5944	2
phase...2vec init-vectors	0,6069	0,6265	2
separatestemfix2vec	0,3445	0,3468	2
wordfix2vec	0,5776	0,5961	1
wordfix2vec - 2 epochy	0,6147	0,6321	2
wordstemprojected2vec	0,6143	0,6335	1
wordstemproj... 2 epochy	**0,6259**	**0,6443**	2
stemfixprojected2vec	0,5863	0,6040	1
stemfixproj... 2 epochy	0,5995	0,6180	2
Viděno dvojic			98,74%

Tabulka 5.11: Výsledky nové datové sady na Novém CZ korpusu.



Obrázek 5.1: Průběh učení jednotlivých modelů. K trénování byl použit Nový CZ korpus.Hodnoty na ose y jsou celková úspěšnost na analogické datové sadě a osa x znázorňuje průběh dvou epoch.

podává nejhorší výsledky a to pouze 18% (společně s *wordgram2vec*, který se příliš neosvědčil). Avšak úspěšnost původního modelu roste v porovnání s ostatními modely rychleji. Tento jev je pravděpodobně způsoben tím, že modely využívající menších slovních tvarů, jsou schopné modelovat vektory neobvyklých slov na základě jejich částí, které jsou viděny častěji. Na konci trénování jsou všechny tvary viděny dostatečně často, a proto se původní model vyrovnává nebo v některých případech modeluje lépe sémantiku slov než n-gramové a stemovací modely. Rapidní pokles *phasewordgram2vec* je způsoben změnou trénování z n-gramů na slova.

Na závěr byla rozšíření otestována také pro anglický jazyk (viz tabulka 5.12). Bohužel v tomto případě se nepovedlo překonat celkovou přesnost původního modelu, ale jednotlivé dílčí přesnosti byly překonány stejně jako v případě Nového CZ korpusu. Sémantická přesnost díky *wordfix2vec* a syntaktická přesnost pomocí *wordgramprojected2vec*. Všechny ostatní modely podávají horší výsledky v obou přesnostech než původní model. Jak již bylo dříve řečeno, nová rozšíření byla navržena pro český jazyk a zaměřují se především na jeho ohebnost a velké množství slovních tvarů, proto v případě anglického jazyka nepodávají stejně dobré výsledky.

Pomocí kombinovaných modelů (viz tabulka 5.13) se povedlo překonat o

Verze	Sém. p.	Syn. p.	Celková p.	Epochy
word2vec - 1 epocha	59,01	60,61	59,88	1
word2vec - 2 epochy	64,28	63,33	**63,76**	2
ngramonly2vec	19,53	45,11	33,55	1
ngramonly2vec - 2 epochy	25,38	47,90	37,71	2
ngram2vec	23,45	46,51	36,08	1
ngram2vec - 2 epochy	27,87	50,22	40,12	2
wordgram2vec	59,48	56,29	57,73	2
wordgramprojected2vec	40,55	71,10	57,29	1
wordgramproj... 2 ep.	48,03	**72,92**	61,66	2
phase...2vec init-vectors	63,91	57,33	60,30	2
separatestemfix2vec	9,86	36,86	24,65	2
wordfix2vec	58,73	60,42	59,66	1
wordfix2vec - 2 epochy	**64,51**	62,24	63,27	2
wordstemprojected2vec	52,31	60,39	57,03	1
wordstemproj... 2 epochy	57,68	62,93	60,56	2
stemfixprojected2vec	18,07	36,83	28,35	1
stemfixproj... 2 epochy	20,93	37,85	30,20	2

Tabulka 5.12: Výsledky na Mikolov EN korpusu.

necelé 3% celkovou přesnost původního modelu pomocí kombinace modelů *wordfix2vec* a *worgramprojected2vec*, a to díky velké syntaktické přesnosti. Sémantická přesnost v této kombinaci lehce klesla na rozdíl od Nového CZ korpusu a kvůli tomu má původní model větší sémantickou přesnost. Kombinace *word2vec* a *ngram2vec* má největší syntaktickou přesnost i přesto, že samotný *ngram2vec* měl znatelně menší přesnost v porovnání s obyčejným *word2vec* modelem.

Verze	Sém. p.	Syn. p.	Celková p.	Epochy
word	**64,28**	63,33	63,76	2
wordfix+wordgramproj...	60,31	71,70	**66,55**	1+1
wordgramproj+word	59,94	71,48	66,26	1+1
wordfix+word	60,58	61,59	61,14	1+1
ngram+word	55,98	**72,88**	65,24	1+1

Tabulka 5.13: Kombinace modelů na Mikolov EN korpusu. U modelů není zmíněna koncovka *2vec*.

6 Závěr

V rámci práce byly prozkoumány různé přístupy k úloze statistické sémantické podobnosti slov a podrobně nastudována metoda *word2vec*. Pro tuto metodu bylo navrženo a implementováno velké množství rozšíření zaměřených pro český jazyk. Pro účely trénování modelů byly vytvořeny dva nové trénovací korpusy a pro účely testování jedna obsáhlá testovací datová sada založená na podobnosti dvojic slov.

Cíle této práce i navržené modely byly zaměřené na sémantickou podobnost, kterou se povedlo navýšit pouze o 2% a pouze pro český jazyk. Avšak hlavním přínosem této práce je značný nárůst syntaktické přesnosti, díky které je i celková přesnost navržených modelů výrazně vyšší. Tato značně vyšší syntaktická přesnost by měla mít pozitivní vliv na ostatní NLP úlohy, které této přesnosti využívají.

Výsledky této práce překonávají v současné době velmi populární nástroj *word2vec*. Model *wordgramprojected2vec* je téměř o 10% přesnější na českém jazyce než původní *word2vec* model. Navíc tento model poskytuje relativně dobré výsledky i pro velice malé množství trénovacích dat. Konkrétně už při velikosti korpusu okolo 160 milionů slov funguje s přesností přes 40%. Pomocí kombinovaného modelu *ngram2vec* a *word2vec* se povedlo navýšit přesnost dokonce o 12%. V případě anglického jazyka se povedlo pomocí kombinovaného modelu *wordfix2vec* a *wordgramprojected2vec* navýšit úspěšnost o necelá 3%.

Budoucí rozšíření této práce by mohla spočívat v ověření předpokladu, že navýšení syntaktické přesnosti má kladný vliv na ostatní NLP úlohy, například určování slovních druhů (Part-of-speech tagging). Také je možné použití nástroje *word2phrase* na trénovací korpusy a odlazení modelů pro fráze slov. Nastavení prahu pro tvorbu frází, popřípadě chování modelů při různé hodnotě podvzorkování. Dále je také možné se zaměřit na větší množství trénovacích epoch a otestovat chování modelů v závislosti na epochách, učícím koeficientu a podobně.

6.1 Splnění bodů zadání

1. Prostudujte metody sémantické analýzy založené na distribuční hypotéze.

Tento bod je splněn v kapitole *2 Teorie*, kde jsou popsány okrajově metody HAL, LSA, Glove a podrobně metoda *Word2vec*, která je rozšiřována

v rámci této práce. V rámci této metody je popsán úvod do neuronových sítí, jednotlivé modely, způsoby trénování a jeho známá rozšíření.

2. Navrhněte a zajistěte tvorbu vhodného českého korpusu.

Tento bod je splněn v kapitole *4 Řešení* (zejména *4.2 Trénovací korpus* a *4.3 Testovací datová sada*), kde je popsaná tvorba dvou nových českých trénovacích korpusů a jedné nové české testovací datové sady založené na podobnosti dvojic slov. Trénovací korpusy a testovací datová sada byly vytvořeny na základě nastudovaných poznatků v kapitole *2.4 Korpus*.

3. Navrhněte rozšíření vhodná pro český jazyk.

Tento bod je splněn v kapitole *3 Analýza*. Byla navržena rozšíření zaměřená na flektivnost českého jazyka založená na znakových n-gramech a stemmování. V rámci těchto rozšíření byly navrženy také čtyři nové architektury a zvážen způsob jejich kombinování a evaluace.

4. Experimentálně ověřte navržené implementace.

Tento bod je splněn v kapitole *5 Výsledky*. Všechny možné kombinace modelů a architektur byly otestovány jak na českém jazyce, tak na angličtině. V českém jazyce byly modely trénovány na dvouch trénovacích korpusech a testovány na dvouch různých testovacích datových sadách.

5. Kriticky zhodnoťte dosažené výsledky.

Tento bod je splněn v kapitole *5.3 Porovnání rozšíření*. Jednotlivá rozšíření jsou porovnána s původním modelem *word2vec*. K ověření předpokladů byl zkonstruován graf s průběhem učení jednotlivých modelů s rostoucím množstvím trénovacích dat. Úspěšnost na sémantické podobnosti se povedlo navýšit pouze v malé míře, avšak hlavním přínosem je značný nárůst syntaktické přesnosti a tím i celkové přesnosti modelů.

Seznam zkratek

CBOW	Continuos Bag-Of-Words
FEMA	Feature Embeddings for domain Adaptation
GloVe	Globální vektory (Global Vectors)
HAL	Hyperspace Analogue to Language
HPS	High Precision Stemmer
LSA	Latent Semantic Analysis
MT	Strojový překlad (Machine Translation)
MTurk	Amazon's Mechanical Turk service
NLP	Zpracování přirozeného jazyka (Nature Language Processing)
NNLM	Jazykový model založený na neuronových sítích (Neural Network Language Model)
RCM	Vztahy omezující model (Relation Constrained Model
RG	Rubenstein-Goodenough datová sada
SVD	Rozklad na singulární čísla (Singular Value Decomposition)

Literatura

[1] . Placing Search in Context: The Concept Revisited. *ACM Trans. Inf. Syst.* January 2002, 20, 1, s. 116–131. ISSN 1046-8188. doi: 10.1145/503104.503110. Dostupné z: http://doi.acm.org/10.1145/503104.503110.

[2] Agirre Eneko, R. G. M. M. SEMEVAL-2016, Interpretable STS Annotation Guidelines. SemEval '16, San Diego, California, June 2016. Association for Computational Linguistics. Dostupné z: http://alt.qcri.org/semeval2016/task2/data/uploads/annotationguidelinesinterpretablests2016v2.2.pdf.

[3] Bengio, Y. – Ducharme, R. – Vincent, P. A Neural probabilistic language model. *Journal of Machine Learning Research.* 2003, 3, s. 1137–1155.

[4] Bruni, E. – Tran, N.-K. – Baroni, M. Multimodal Distributional Semantics. *J. Artif. Intell. Res.(JAIR).* 2014, 49, 1-47.

[5] Brychcín, T. – Konopík, M. HPS: High precision stemmer. *Information Processing & Management.* 2015, 51, 1, s. 68 – 91. ISSN 0306-4573. doi: http://dx.doi.org/10.1016/j.ipm.2014.08.006. Dostupné z: http://www.sciencedirect.com/science/article/pii/S0306457314000843.

[6] Carletta, J. Assessing Agreement on Classification Tasks: The Kappa Statistic. *Comput. Linguist.* June 1996, 22, 2, s. 249–254. ISSN 0891-2017. Dostupné z: http://dl.acm.org/citation.cfm?id=230386.230390.

[7] Demir, H. – Ozgur, A. Improving Named Entity Recognition for Morphologically Rich Languages Using Word Embeddings. In *13th International Conference on Machine Learning and Applications, ICMLA 2014, Detroit, MI, USA, December 3-6, 2014*, s. 117–122, 2014. doi: 10.1109/ICMLA.2014.24. Dostupné z: http://dx.doi.org/10.1109/ICMLA.2014.24.

[8] Dolamic, L. – Savoy, J. Advances in Multilingual and Multimodal Information Retrieval. Berlin, Heidelberg: Springer-Verlag, 2008. Stemming Approaches for East European Languages, s. 37–44. doi: 10.1007/978-3-540-85760-0_4. Dostupné z: http://dx.doi.org/10.1007/978-3-540-85760-0_4. ISBN 978-3-540-85759-4.

[9] Dowdy, S. – Wearden, S. – Chilko, D. *Statistics for Research*. Wiley Series in Probability and Statistics. Wiley, 2011. Dostupné z: https://books.google.cz/books?id=IIv1tchEN7MC. ISBN 9780471477426.

[10] Fleiss, J. L. Measuring nominal scale agreement among many raters. *Psychological bulletin*. 1971, 76, 5, s. 378.

[11] Ganitkevitch, J. – Van Durme, B. – Callison-Burch, C. PPDB: The Paraphrase Database. In *Proceedings of NAACL-HLT*, s. 758–764, Atlanta, Georgia, June 2013. Association for Computational Linguistics. Dostupné z: http://cs.jhu.edu/~ccb/publications/ppdb.pdf.

[12] Goldberg, Y. *On the importance of comparing apples to apples: a case study using the GloVe model* [online]. 2014. Citováno: 20. 5. 2016. Dostupné z: https://docs.google.com/document/u/1/d/1ydIujJ7ETSZ688RGfU5IMJJsbxAi-kRl8czSwpti15s/mobilebasic?pli=1.

[13] Goldberg, Y. – Levy, O. word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*. 2014.

[14] Harris, Z. Distributional structure. *Word*. 1954, 10, 23, s. 146–162.

[15] Hnátková, M. et al. The SYN-series Corpora of Written Czech. In Chair), N. C. C. et al. (Ed.) *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland, may 2014. European Language Resources Association (ELRA). ISBN 978-2-9517408-8-4.

[16] Huddleston, R. *English grammar : an outline / Rodney Huddleston*. Cambridge University Press Cambridge ; New York, 1988. Dostupné z: http://www.loc.gov/catdir/toc/cam023/87034124.html. ISBN 0521323118 0521311527.

[17] Krcmár, L. – Konopík, M. – Jezek, K. Exploration of Semantic Spaces Obtained from Czech Corpora. In *DATESO*, s. 97–107, 2011.

[18] Kroeger, P. *Analyzing grammar: an introduction*. Cambridge University Press, 2005. Dostupné z: http://books.google.com/books?id=rSglHbBaNyAC,/bib/kroeger/kroeger2005analyzing/Analyzing%20Grammar%20-%20An%20Introduction%20By%20PAUL%20R.%20KROEGER.pdf. ISBN 9780521816229.

[19] Landauer, T. – Foltz, P. – Laham, D. An introduction to latent semantic analysis. *Discourse processes*. 1998, 25, s. 259–284.

[20] Lehmann, J. et al. DBpedia - A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. *Semantic Web Journal*. 2014.

[21] Levy, O. – Goldberg, Y. Dependency-Based Word Embeddings. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. 2014, 2.

[22] Ling, W. et al. Two/Too Simple Adaptations of Word2Vec for Syntax Problems. In *NAACL*, 2015.

[23] Liu, B. *Web Data Mining*. 2007. Dostupné z: `http://dx.doi.org/10.1007/978-3-540-37882-2`. http://www.cs.uic.edu/ liub/WebMiningBook.html.

[24] Lund, K. – Burgess, C. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, & Computers*. 1996, 28, 2, s. 203–208.

[25] Luong, M.-T. – Socher, R. – Manning, C. D. Better Word Representations with Recursive Neural Networks for Morphology. In *CoNLL*, Sofia, Bulgaria, 2013.

[26] Mikolov, T. *Language Modeling for Speech Recognition in Czech*. PhD thesis, Masters thesis, Brno University of Technology, 2007.

[27] Mikolov, T. *Multithreading and Concurrency about word2vec* [online]. 2013. Citováno: 20. 5. 2016. Dostupné z: `https://groups.google.com/forum/#!topic/word2vec-toolkit/iDTvPE0gFD0`.

[28] Mikolov, T. et al. Neural network based language models for highly inflective languages. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, s. 4725–4728. IEEE, 2009.

[29] Mikolov, T. et al. Recurrent neural network based language model. *INTERSPEECH*. 2010, 2, s. 3.

[30] Mikolov, T. et al. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*. 2013.

[31] Mikolov, T. – Le, Q. V. – Sutskever, I. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*. 2013.

[32] Mikolov, T. et al. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, s. 3111–3119, 2013.

[33] Mikolov, T. – Yih, W.-t. – Zweig, G. Linguistic Regularities in Continuous Space Word Representations. In *HLT-NAACL*, s. 746–751, 2013.

[34] Miller, G. A. WordNet: A Lexical Database for English. *Commun. ACM*. November 1995, 38, 11, s. 39–41. ISSN 0001-0782. doi: 10.1145/219717.219748. Dostupné z: http://doi.acm.org/10.1145/219717.219748.

[35] Miller, G. A. – Charles, W. G. Contextual correlates of semantic similarity. *Language and Cognitive Processes*. 1991, 6, 1, s. 1–28. doi: 10.1080/01690969108406936. Dostupné z: http://dx.doi.org/10.1080/01690969108406936.

[36] Morin, F. – Bengio, Y. Hierarchical Probabilistic Neural Network Language Model. In Cowell, R. G. – Ghahramani, Z. (Ed.) *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, s. 246–252. Society for Artificial Intelligence and Statistics, 2005. Dostupné z: http://www.iro.umontreal.ca/~lisa/pointeurs/hierarchical-nnlm-aistats05.pdf.

[37] Pennington, J. – Socher, R. – Manning, C. D. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, s. 1532–1543, 2014. Dostupné z: http://www.aclweb.org/anthology/D14-1162.

[38] Radinsky, K. et al. A Word at a Time: Computing Word Relatedness Using Temporal Semantic Analysis. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, s. 337–346, New York, NY, USA, 2011. ACM. doi: 10.1145/1963405.1963455. Dostupné z: http://doi.acm.org/10.1145/1963405.1963455. ISBN 978-1-4503-0632-4.

[39] Rong, X. word2vec Parameter Learning Explained. *arXiv preprint arXiv:1411.2738*. 2014.

[40] Rubenstein, H. – Goodenough, J. B. Contextual Correlates of Synonymy. *Commun. ACM*. October 1965, 8, 10, s. 627–633. ISSN 0001-0782. doi: 10.1145/365628.365657. Dostupné z: http://doi.acm.org/10.1145/365628.365657.

[41] Sienčnik, S. K. Adapting word2vec to Named Entity Recognition. In *Proceedings of the 20th Nordic Conference of Computational Linguistics, NODALIDA 2015, May 11-13, 2015, Vilnius, Lithuania*, č. 109, s. 239–243. Linköping University Electronic Press, 2015.

[42] Suchanek, F. M. – Kasneci, G. – Weikum, G. Yago: A Core of Semantic Knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, s. 697–706, New York, NY, USA, 2007. ACM. doi: 10.1145/1242572.1242667. Dostupné z: http://doi.acm.org/10.1145/1242572.1242667. ISBN 978-1-59593-654-7.

[43] Svoboda, L. – Brychcín, T. *New word analogy corpus for exploring embeddings of Czech words*. Springer International Publishing, 2016.

[44] Turney, P. D. – Pantel, P. – others. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*. 2010, 37, 1, s. 141–188.

[45] Wolf, L. et al. Joint word2vec networks for bilingual semantic representations. In *In Proceedings of the 15th International Conference on Intelligent Text Processing and Computational Linguistics*, 2013.

[46] Wynne, M. *Developing Linguistic Corpora: a Guide to Good Practice.* [online]. 2005. Citováno: 20. 5. 2016. Dostupné z: http://ota.ox.ac.uk/documents/creating/dlc/.

[47] Yang, Y. – Eisenstein, J. Unsupervised multi-domain adaptation with feature embeddings. *Proc. of NAACL-HIT*. 2015.

[48] Yu, M. – Dredze, M. Improving Lexical Embeddings with Semantic Knowledge. In *ACL (2)*, s. 545–550, 2014.

[49] Řehůnek, R. *Making sense of word2vec* [online]. 2014. Citováno: 20. 5. 2016. Dostupné z: http://rare-technologies.com/making-sense-of-word2vec/.

A Instrukce pro tvorbu datové sady

Tvorba datové sady na podobnost slov.
==========

Následující tabulka obsahuje informace o dvojicích slov. Pro další postup je nutné registrovat se na stránkách korpus.cz.

Pro každou dvojici slov proveďte následující kroky:

1. Přeložit anglické názvy v případě, že jsou k dispozici. Překládejte pokud možno jako podstatná jména v jednotném čísle a používejte základní tvary slov. V případě, že je možné přeložit slovo mnoha významy, tak vyberte význam nejvíce podobný druhému slovu v páru.

2. Doplnit celočíselný stupeň podobnosti slov (0 - 5).

3. Vyhledat na korpus.cz tato dvě slova a zkopírovat kontext daných slov (pole Kontext - slovo 1, Kontext - slovo 2). V případě, že dotaz na korpus.cz nic nevyhledá, tak nechte pole prázdné.

4. Je-li slovo víceznačné, zkopírovat kontext s jiným významem slova (pole Víceznačnost - slovo 1, Víceznačnost - slovo 2).

5. Doplnit celočíselný stupeň souvislosti slov (0 - 5).

6. Vyhledat na korpus.cz tato dvě slova ve vzálenosti max. 10 nebo 20 slov (pole Kontext - souvislost). Nenajde-li se výskyt ve vzdálenosti 10, pokuste se je najít ve vzálenosti 20.

Vysvětlení pojmů:
Podobnost - Sémantická podobnost slov určuje, jak velký význam daná slova sdílejí. Čím větší je podobnost, tím větší je šance, že při záměně slov bude věta dávat stále stejný smysl.

Souvislost - Sémantická souvislost slov určuje, jak moc se daná slova mohou vyskytnout ve stejném kontextu. Čím větší je souvislost, tím větší je šance, že se slova objeví ve stejném kontextu.

Kontext - Libovolně velké okolí slova. Preferovaný kontext je slovo uprostřed věty. V případě, že je slovo na konci/začátku věty zkopírujte i větu následující/předchozí, je-li to možné.

Základní předpoklady:
$Podobnost = 5 \rightarrow Souvislost = 5$ - Neplatí opačně!
Antonyma ve stejné doméně. $Podobnost \pm 3, Souvilost = 5$
Kontexty nedopisujte ručně.

Příkazy na korpus.cz v CQL:
Pro jednotlivá slova v kontextu stačí zadat dané slovo. Jestliže dotaz nic nevyhledá použijte KonText nástroj na stránkách korpus.cz, kde je možné přepnout na větší korpus "syn".

Slovo "kolo"a "auto"ve vzdálenosti 10 slov:
$[lemma = "kolo"][]\{1, 10\}[lemma = "auto"]$

Slovo "kolo"a "auto"ve vzdálenosti 20 slov:
$[lemma = "kolo"][]\{1, 20\}[lemma = "auto"]$

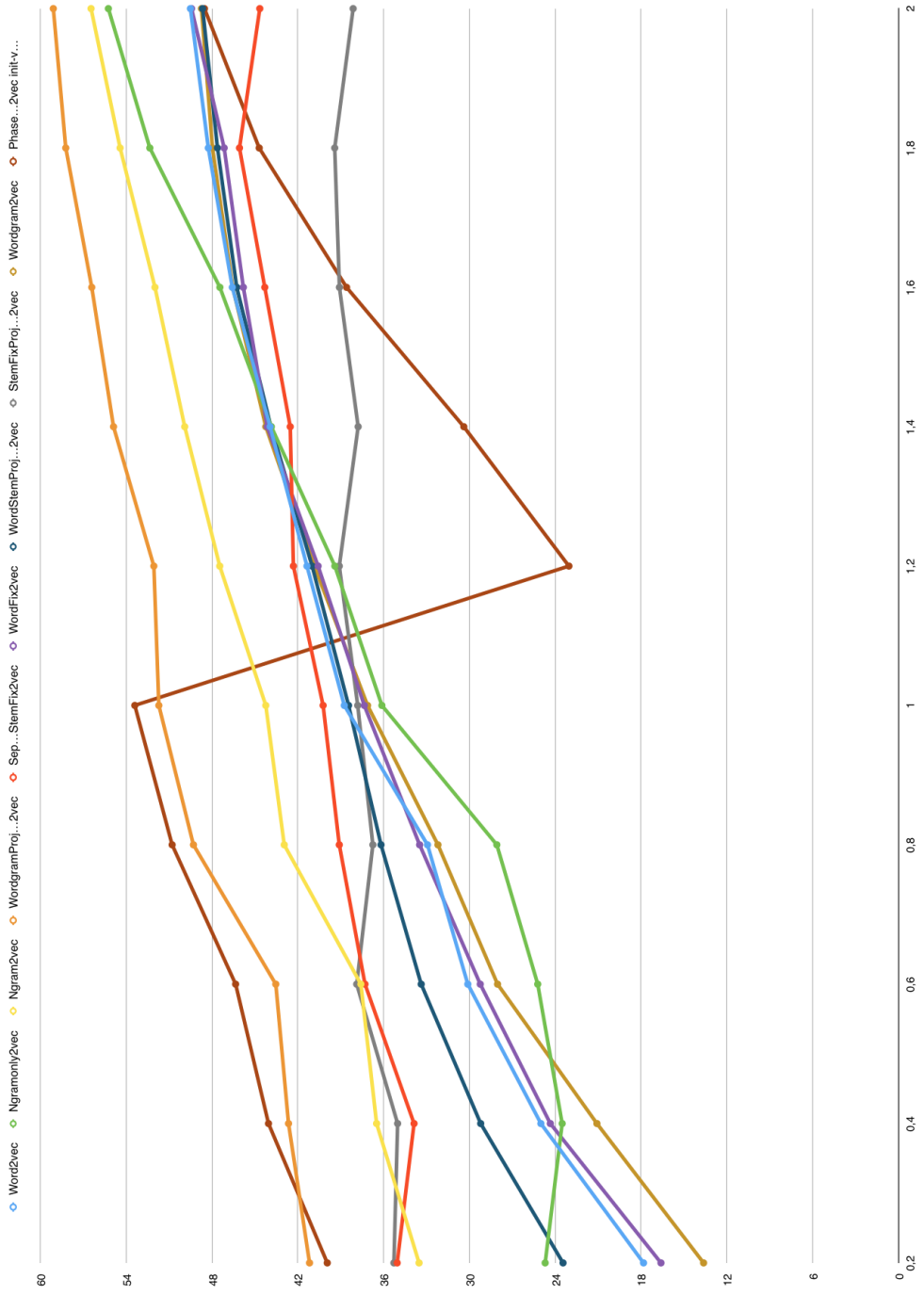
Pro složitější dotazy (např. specifikovat slovní druh) je možné využít opět KonText nástroj.

B Ukázka vytvořené datové sady

Slovo 1	Slovo 2	Podobnost	Souvislost
auto	automobil	5	5
klenot	šperk	5	5
kluk	chlapec	5	5
pobřeží	břeh	4	5
kouzelník	čaroděj	5	5
poledne	poledne	5	5
pec	kamna	4	5
jídlo	ovoce	3	5
pták	kohout	3	5
poduška	polštář	4	5
les	zálesí	4	4
autogram	podpis	5	5
nevolník	otrok	4	4
tygr	kočka	3	4
kniha	papír	2	5
počítač	klávesnice	2	5
droga	zneužívání	2	4
chléb	máslo	1	4
křižník	armáda	2	4
kronika	mládež	0	2
vědec	test	1	4
karanténa	choroba	2	5
trikolóra	vlajka	3	4
rozhněvanější	bouřlivý	3	4
potvrzení	přijetí	2	4
koncert	klavír	1	5
skupina	punk	2	4
provedení	realizace	5	5
výprava	expedice	5	5

Tabulka B.1: Ukázka nově vytvořené datové sady. Z důvodu nedostatku prostoru na stránce je zde zobrazena pouze zkrácená verze se skórem.

C Graf s průběhem trénování



Obrázek C.1: Průběh učení jednotlivých modelů.

D Návod k programu

D.1 Požadavky

Pro správné přeložení a spuštění vyžaduje program následující požadavky:

- Make[1] $\geq$ 3.81
- GCC[2] $\geq$ 4.7.2

Tyto nástroje jsou součástí *GNU*[3] a jsou obvykle běžnou součástí unixových operačních systémů. Knihovny třetích stran jsou přiloženy v adresáři *src/lib* (viz kapitola *E Obsah přiloženého DVD*) a není nutné s nimi nijak manipulovat.

D.2 Přeložení

Přeložení programu je možné provézt pomocí nástroje *Make*. Pro vytvoření spustitelných programů stačí otevřít adresář *src* s *makefile* a zadat do příkazové řádky následující příkaz:

```
make
```

Poté se vytvoří adresář *bin* s jednotlivými spustitelnými verzemi programu. Také je možné pomocí nadefinovaných cílů přeložit pouze část zdrojových kódů:

- `all` - Standardní cíl spouštěný v případě, že cíl není zadán. Překládá všechny verze a nástroje programu.
- `classic` - Přeloží původní verzi a nástroje *word2vec*.
- `ngrams` - Přeloží všechna ngramová rozšíření.
- `stemming` - Přeloží všechna stemmovačí rozšíření.
- `utils` - Přeloží všechny nově přidané nástroje

[1] https://www.gnu.org/software/make/
[2] https://gcc.gnu.org
[3] https://www.gnu.org/home.en.html

- Každý nástroj a verze programu má také svůj cíl, který se jmenuje podle názvu nástroje nebo modelu (např. `wordgramprojected2vec` atd.).

Jednotlivé cíle je možné spustit příkazem:

```
make <cíl>
```

D.3 Spuštění

Každý program, spuštěný bez argumentů, vypíše nápovědu k správnému spuštění.

D.3.1 Jednotlivé modely

```
<model> <argumenty>
```

Seznam argumentů (v závorkách jsou zmíněny standardní hodnoty jsou-li k dispozici - hodnoty převzaty z původního nástroje):

- `-train <cesta>` - Cesta k trénovacímu korpusu
- `-output <cesta>` - Cesta k výstupnímu souboru s modelem
- `-save-vocab <cesta>` - Uložit slovník na danou cestu
- `-read-vocab <cesta>` - Načíst slovník na dané cestě
- `-size <velikost>` - Velikost výsledných vektorů (100)
- `-binary <1/0>` - Binární výstup ANO/NE (NE)
- `-cbow <1/0>` - Výběr trénování CBOW/Skip-gram (CBOW)
- `-alpha <float>` - Počáteční míra učení (reálné číslo) (0.025)
- `-window <int>` - Velikost okna (celé číslo) (5)
- `-sample <float>` - Hodnota podvzorkování (1e-3)
- `-hs <1/0>` - Výběr trénování Hierarchický softmax/Negativní vzorkování (0)
- `-negative <int>` - Počet negativních vzorků (> 0 - zapnuto) (5)
- `-threads <int>` - Počet vláken (12)

- `-iter <int>` - Počet iterací (epoch) (5)
- `-min-count <int>` - Minimální počet slov ve slovníku (5)
- `-classes <int>` - Počet shluků při shlukování (není předmětem této práce) (0)
- `-debug <0/1/2/3>` - Stupeň ladících výstupů (2)
- `-ngrams <int>` - Velikost n-gramů (4)
- `-ngram-bounding <1/0>` - Rozlišovat krajní n-gramy ANO/NE (ANO)
- `-init-vectors <1/0>` - Předinicializovat vektory pomocí n-gramů ANO/NE (fázový model) (ANO)

V případě, že argument chybí, bere se standardní hodnota. V případě, že je argument navíc, je ignorován.

D.3.2 Ukázky

Přiložený *makefile* také obsahuje cíle pro ukázkové spuštění programu (trénováno na části Nového CZ korpusu přiloženého na DVD) s automatickou evaluací (testováno na české analogické testovací datové sadě). Jednotlivé ukázky je opět možné spustit pomocí příkazu:

```
make <ukázka>
```

- classicExample - Obyčejný *word2vec*
- ngramExample - *Wordgramprojected2vec*
- stemmingExample - *Wordfix2vec*

D.3.3 Compute–sim

Evaluační skript pro vyhodnocení testovací datové sady založené podobnosti dvojic slov.

```
compute-sim <model> < cz_sim.txt
```

D.3.4 Compute–accuracy

Evaluační skript pro vyhodnocení analogické testovací datové sady.

```
compute-accuracy <model> <počet_skupin> <limit_slovníku> \
 < <otázky>.txt
```

D.3.5 Models–accuracy

Evaluační skript pro vyhodnocení kombinace modelů na analogické datové sadě.

```
models-accuracy <model 1> <model 2> <počet_skupin> \
<limit_slovníku> < <otázky>.txt
```

D.3.6 Binary2text

Nástroj pro konverzi binárních modelů na textové[4].

```
binary2text <type> <zdroj> <cíl>
```

D.4 Výstup

Výstupem jednotlivých modelů je buď binární nebo textový soubor s výslednými vektory. Tento soubor má následující formát:

```
<velikost slovníku> <dimenze>
<slovo 1> <vektor 1>
<slovo 2> <vektor 2>
...
```

Během trénování programy informují o průběhu na základě zvolené hodnoty *debug* argumentu (v případě vyššího stupně jsou zahrnuty i nižší stupně):

- 0 - Žádné ladící výpisy
- 1 - Zobrazí velikosti trénovacího korpusu a slovníku
- 2 - Zobrazí průběh trénování
- 3 - Zobrazí mapování slov na n-gramy nebo slov na stem a koncovku

V případě evaluačního skriptu na testovací datové sady založené na podobnosti slov je výstup následující:

```
Loading model...
Model loaded...

Results:
-------------------------------------------------------
```

[4]Převzato z https://github.com/marekrei/convertvec

```
Pairs seen: <viděných> of <celkem>

Pearson correlation: <hodnota korelace>
Spearman correlation: <hodnota korelace>
-----------------------------------------------------------
```

V případě analogických testovacích datových sad je výstup komplikovanější:

```
<název skupiny otázek>:
ACCURACY TOP1: xx.xx %  (<správně> / <viděných> / <celkem>)
Total accuracy: xx.xx % Semantic accuracy: xx.xx % \
Syntactic accuracy: xx.xx %
```

Jednotlivé skupiny jsou vyhodnocovány samostatně (viz druhý řádek) a po každé skupině je vypsáno i celkové skóre (viz třetí řádek).

E Obsah přiloženého DVD

K této práci je přiložen DVD disk, na kterém jsou uloženy zdrojové kódy, knihovny a nástroje pro přeložení programu. Jeho struktura je následující:

- diploma-thesis - Elektronická verze (PDF) této práce.
- datasets - Testovací datové sady použité v této práci.
  - cz_sim.txt - Nová česká testovací datová sada založená na podobnosti dvojic slov pouze s hodnoty pro podobnosti k evaluaci pomocí skriptu *compute-sim*.
  - CZ-SIM-DATASET.txt - Kompletní česká testovací datová sada založená na podobnosti dvojic slov v textové podobě.
  - CZ-SIM-DATASET.xlxs - Kompletní Česká testovací datová sada založená na podobnosti dvojic slov ve formátu *Microsoft Excel*.
  - czech_emb_corpus_words_lower.txt - Část české analogické datové sady složená pouze ze slov. Sada je pouze malými písmeny.
  - czech_emh_corpus_phrases_lower.txt - Část české analogické datové sady složená pouze z frází slov. Sada je pouze malými písmeny.
  - czech_emb_corpus.txt - Původní česká analogická datová sada.
  - questions-phrases.txt - Anglická analogická datová sada složená pouze z frází slov.
  - questions-words.txt - Anglická analogická datová sada složená pouze ze slov.
- corpora - Krátké verze trénovacích korpusů.
  - cs_data.txt - Nový CZ korpus.
  - cs_data_clean.txt - Nový CZ korpus s kompletním předzpracováním.
  - large_cs_data.txt - Obsáhlý CZ korpus.
  - cs_wiki.txt - CZ Wiki korpus.
  - news.txt - Mikolov EN korpus.
  - cs_data_stemmed.txt - Ostemovaný Nový CZ korpus.

  - cs_wiki_stemmed.txt - Ostemovaný CZ Wiki korpus.
  - news_stemmed.txt - Ostemovaný Mikolov EN korpus.
- src - Zdrojové kódy a *makefile* pro přeložení programu.
  - lib - Zdrojové kódy knihovny *ALGLIB*.
- Poster - Poster prezentující výsledky práce.
- readme - Stručný popis DVD.
- bin - Složka pro binární soubory generovaná nástrojem *make*.
- models - Složka pro natrénované modely generovaná nástrojem *make*.
- results - Složka pro výsledky generovaná nástrojem *make*.