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

Diplomová práce

Mezi-jazyčné transformace sémantických prostorů

Plzeň 2021 Bc. Adam Mištera

ZÁPADOČESKÁ UNIVERZITA V PLZNI

Fakulta aplikovaných věd

Akademický rok: 2020/2021

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Adam MIŠTERA**
Osobní číslo:	**A19N0038P**
Studijní program:	**N3902 Inženýrská informatika**
Studijní obor:	**Softwarové inženýrství**
Téma práce:	**Mezi-jazyčné transformace sémantických prostorů**
Zadávající katedra:	**Katedra informatiky a výpočetní techniky**

Zásady pro vypracování

1. Prostudujte state-of-the-art metody pro jazykově nezávislou reprezentaci významu.
2. Naimplementujte vybrané lineární transformace sémantických prostorů.
3. Prozkoumejte a naimplementujte vybrané nelineární transformace založené na neuronových sítích pro transformace sémantických prostorů.
4. Natrénujte mezi-jazyčné sémantické prostory pro reprezentaci jazyků z různých jazykových rodin.
5. Metody pro jazykově nezávislou reprezentaci významu otestujte na datasetech slovních podobností, slovních analogií a na strojovém překladu slov, a výsledky zhodnoťte.

Rozsah diplomové práce: **doporuč. 50 s. původního textu**
Rozsah grafických prací: **dle potřeby**
Forma zpracování diplomové práce: **tištěná**

Seznam doporučené literatury:

dodá vedoucí diplomové práce

Vedoucí diplomové práce: **Ing. Tomáš Brychcín, Ph.D.**
Nové technologie pro informační společnost

Datum zadání diplomové práce: **11. září 2020**
Termín odevzdání diplomové práce: **20. května 2021**

L.S.

Doc. Dr. Ing. Vlasta Radová
děkanka

Doc. Ing. Přemysl Brada, MSc., Ph.D.
vedoucí katedry

V Plzni dne 24. září 2020

Prohlášení

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

V Plzni dne 20. května 2021

Bc. Adam Mištera

Poděkování

Děkuji především Ing. Tomáši Brychcínovi, PhD. za odborné vedení, ochotu a cenné rady, které mi v průběhu zpracování této diplomové práce věnoval.

Výpočetní prostředky byly poskytnuty v rámci projektu e-Infrastruktura CZ (e-INFRA LM2018140) poskytovaného v rámci programu Velké infrastruktury pro výzkum, vývoj a inovace.

Abstract

Cross-lingual meaning representation is a current topic in the field of Natural Language Processing. Semantic spaces of various languages are transformed into a shared universal space. It allows to transfer knowledge between languages, especially between resource-rich and resource-poor languages. This Master Thesis aims to analyze, compare and implement available methods of language independent semantic representation. Most of related works focus on linear projections as they showed a very good performance. In this thesis, however, we study non-linear methods based on artificial neural networks and clustering. Both linear and non-linear methods were subsequently evaluated on different tasks such as datasets containing word similarities, word analogies and machine translation. The newly implemented non-linear transformations consistently outperformed state-of-the-art linear transformations in all evaluation criteria and on several languages within different language families.

Abstrakt

Mezi-jazyčná reprezentace významu je aktuální téma v oblasti zpracování přirozeného jazyka. Sémantické prostory z různých jazyků jsou transformovány do jednoho sdíleného univerzálního prostoru. Tento fakt umožňuje přenést znalosti z jazyků, které jsou velmi bohaté na zdroje, do jazyků, jež jsou na zdroje omezené. Cílem této diplomové práce bylo analyzovat, porovnat a implementovat dostupné metody pro jazykově nezávislou sémantickou reprezentaci. Většina souvisejících prací se zaměřuje na lineární transformace, jelikož vykazují velmi dobrý výkon. V této práci se však soustřeďujeme na zvolené nelineární transformace založené na umělých neuronových sítích a shlukování. Kvalita lineárních i nelineárních metod je následně vyhodnocena na různých úlohách, například datových sadách slovních podobností, slovních analogií a strojovém překladu. Nově implementované nelineární transformace v mnoha ohledech překonaly lineární transformace ve všech sledovaných kategoriích na rozdílných jazycích z odlišných jazykových rodin.

Obsah

1 Úvod

V druhém desetiletí 21. století zaznamenaly značný úspěch metody *Word2vec* a *FastText* založené na umělých neuronových sítích pro vytvoření vektorové reprezentace slov. Slovní vektory (anglicky *word embedding*) jsou v současné době hojně využívané především v oblasti zpracování přirozeného jazyka [13, 17, 19]. Výše zmíněné metody jsou schopné reprezentovat datový korpus předaný na vstupu jako vektorový prostor a následně s jednotlivými slovy pracovat jako s vektory. Výhodou tohoto přístupu je fakt, že výsledné slovní vektory reprezentují samotný význam slova. Díky tomu lze v systémech využívajících zpracování přirozeného jazyka zpracovat text podobného významu, přestože je napsán jinými slovy.

Dalším přirozeným krokem byl výzkum v oblasti vztahů mezi sémantickými prostory reprezentujícími rozdílné jazyky. Základní motivací je schopnost odvodit význam slova ve více jazycích, což skýtá mnoho různých využití především v oblastech vyhledávání informací (anglicky *information retrieval*) a strojového překladu (anglicky *machine translation*). Současně se nabízí možnost přenosu znalostí z jazyků, které jsou velmi bohaté na zdroje, jako je například globálně rozšířená angličtina, do jazyků, které jsou na zdroje velmi chudé, jako například islandština.

Cílem diplomové práce je průzkum nejmodernějších metod pro jazykově nezávislou reprezentaci významu a následný návrh a implementace nelineárních metod s využitím umělých neuronových sítí a metody shlukování určené pro transformaci sémantických prostorů. Implementované metody pro jazykově nezávislou reprezentaci významu jsou dále otestované na datasetech slovních podobností, slovních analogií a na strojovém překladu slov.

Kapitola 2 analytické části diplomové práce je věnována sémantickým prostorům a definici všech potřebných termínů, které se s nimi pojí, společně se zástupci modelů určených pro jejich reprezentaci. Kapitola 3 se zabývá třemi klíčovými metodami pro lineární transformace, které lze využít pro vytvoření transformační matice mezi dvěma sémantickými prostory. Kapitola 4 je věnována navržení nelineárních transformací s použitím neuronových sítí, které by byly schopné překonat sledované vlastnosti lineárních transformací. V kapitole 5 jsou následně představeny metody využívající shlukování, které jsou určené pro vylepšení vlastností základních lineárních transformací. Druhá část diplomové práce je zaměřena na detailní představení experimentů a všech naměřených hodnot společně s porovnáním všech implementovaných metod.

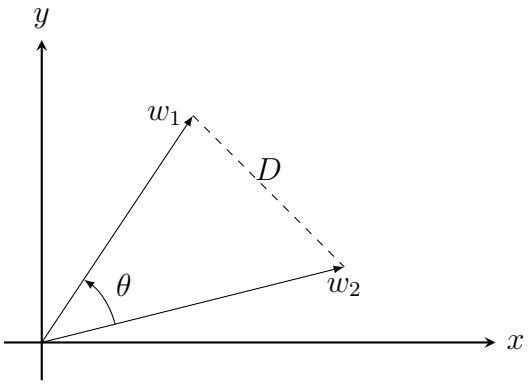
2 Sémantické prostory

Cílem této kapitoly je definice sémantických prostorů a představení jejich klíčových vlastností společně s modely distribuční sémantiky, které slouží pro vytvoření reprezentace sémantických prostorů. První část této kapitoly bude věnována distribuční hypotéze, na které jsou modely distribuční sémantiky založené. V druhé části této kapitoly se zaměříme na tři rozdílné modely, jmenovitě *Word2vec*, *Glove* a *fastText*, které lze využít pro vytvoření vektorové reprezentace slov. Poslední zmíněná metoda *fastText* je využita v rámci diplomové práce.

Označíme-li $w \in \mathbf{V}$ jako slovo, kde $\mathbf{V}$ představuje slovník daného jazyka, můžeme posléze definovat **sémantický prostor** jako funkci

$$\mathcal{S} : \mathbf{V} \mapsto \mathbb{R}^d \tag{2.1}$$

Jedná se tedy o matematickou funkci, která mapuje slovo w do Euklidovského prostoru o dimenzi d. Vybrané slovo je následně reprezentováno pomocí vektoru reálných čísel $\mathcal{S}(w)$. Reprezentace slov pomocí vektorů přináší řadu výhod. Reálná čísla jsou v počítačovém prostředí zpracovávána snadněji než textové řetězce, zároveň je poté možné se slovy provádět stejné operace jako s vektory, například sčítání či odčítání.



Obrázek 2.1: Zjednodušený sémantický prostor

Získaný vektor $\mathcal{S}(w)$ posléze v daném sémantickém prostoru reprezentuje samotný **význam** slova w. Tento fakt je obzvláště důležitý, neboť podobnost zvolených vektorů $sim(\mathcal{S}(w_1), \mathcal{S}(w_2))$, respektive slov w_1 a w_2 lze následně velmi snadno porovnat například vypočtením euklidovské vzdálenosti D mezi vektory, případně určením úhlu θ mezi těmito vektory s využitím kosinové podobnosti. Podrobný postup výpočtu těchto metrik je uveden v sekci

6.5.1, respektive 6.5.2. Na obrázku 2.1 lze vidět ukázku značně zjednodušeného dvourozměrného sémantického prostoru obsahujícího slova w_1 a w_2.

2.1 Distribuční hypotéza

Modely distribuční sémantiky jsou založeny na takzvané distribuční hypotéze. Podle této hypotézy mají slova, která se vyskytují v podobném kontextu, tendenci mít obdobné významy. Tento předpoklad vychází z tvrzení, které formuloval již v roce 1957 anglický lingvista John Rupert Firth:

„You shall know a word by the company it keeps.“ [9]

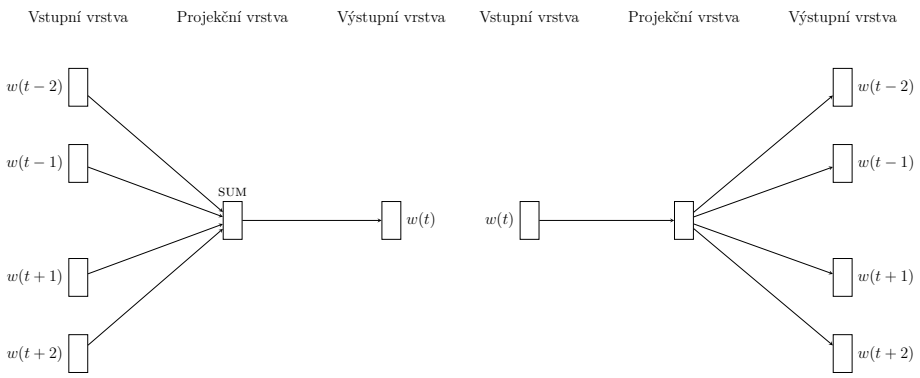
Z tohoto tvrzení vyplývá, že význam slova je v určitém slova smyslu předurčen jazykovým kontextem, v němž se vyskytuje. Jednoduchým příkladem demonstrujícím výše uvedený výrok je například věta: “Šel do cukrárny pro . . . ”, jejímž doplněním budou spíše slova *dort* nebo *zmrzlinu* než *auto* či *růže*. Význam tohoto faktu je ještě více patrný u *homonym* neboli slov souzvučných. Jedná se o slova, která mají stejnou zvukovou či grafickou podobu, ale zcela odlišný význam. Příkladem homonyma je slovo *pila*, které může označovat jak nástroj určený k řezání, tak příčestí minulé, jak lze vidět zejména ve větách “Každé ráno *pila* čaj.” a “*Pila* byla opřena o strom.”. V těchto případech je pro určení správného významu slova znalost jeho kontextu zcela nezbytná.

Modelů, které jsou založené na výše uvedené distribuční hypotéze, je v současné době značné množství. V následujících sekcích jsou představeny tři vybraní zástupci, konkrétně *Word2vec*, *GloVe* a *fastText*.

2.2 Word2vec

Model **Word2vec**, který byl představen v roce 2013 týmem vedeným Tomášem Mikolovem, slouží k vytvoření slovních vektorů pomocí umělých neuronových sítí [17, 19]. Model *Word2vec* vychází z výše uvedené distribuční hypotézy a lze jej dle architektury dále rozdělit na dva odlišné druhy. Konkrétně se jedná o *continuous bag-of-words* (*CBOW*) a *skip-gram*. *CBOW* se na základě okolí zvoleného slova snaží dané slovo předpovědět. Oproti tomu *skip-gram* se pokouší ze zvoleného slova předpovědět slova nacházející se v jeho okolí. Oba uvedené druhy jsou znázorněné na obrázku 2.2.

Pokud se blíže zaměříme na metodu *skip-gram*, můžeme definovat její cíl, kterým je pro každé slovo z posloupnosti $w_1, w_2, w_3, \ldots w_T$ určit slova,



(a) CBOW (b) Skip-gram

Obrázek 2.2: Druhy modelu *Word2vec*

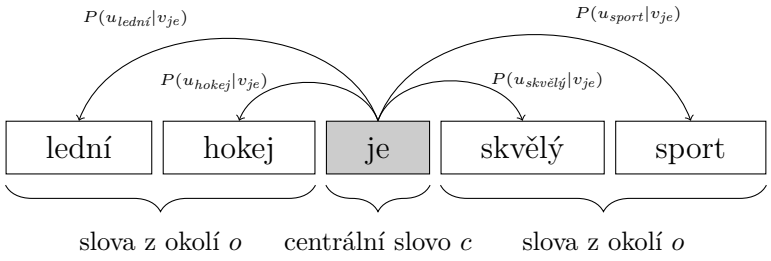
která se vyskytují v jeho okolí o předem zvolené maximální vzdálenosti m. Následně lze stanovit cenovou funkci pomocí vzorce:

$$J(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-m\leq j\leq m\\ j\neq 0}} \log P(w_{t+j}|w_t). \tag{2.2}$$

Cílem tohoto modelu je minimalizovat výše uvedenou cenovou funkci, čímž současně dochází k maximalizaci přesnosti předpovědi slov v okolí. Pravděpodobnost $P(w_{t+j}|w_t)$ lze určit využitím funkce *softmax* jako:

$$P(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w\in V}\exp(u_w^T v_c)}, \tag{2.3}$$

kde symbol o představuje slova z okolí, symbol c samotné centrální slovo a symbol V slovník zvoleného trénovacího korpusu. Pro lepší představu zná-



Obrázek 2.3: Pravděpodobnosti v modelu *skip-gram*

zorňuje obrázek 2.3 pravděpodobnosti jednotlivých slov v ukázkovém textu "lední hokej je skvělý sport". Centrální slovo *je* v tomto případě slouží pro

vypočtení pravděpodobností výskytu slov v jeho okolí, jmenovitě *lední*, *hokej*, *skvělý* a *sport*.

Výše definovaná pravděpodobnost je však značně nepraktická, jelikož cena výpočtu tohoto výrazu je úměrná počtu slov ve slovníku V, který se často pohybuje v hodnotách větších než 10^5. Z tohoto důvodu se pro zvýšenou efektivnost výpočtu používá pouze aproximace funkce *softmax* pomocí *hierarchical softmax* [21] nebo *negative sampling* [10, 20].

2.3 GloVe

Model **GloVe** neboli *Global Vectors* [23] představuje metodu učení bez učitele sloužící pro získání vektorové reprezentace slov. *GloVe* je open-source projekt vyvíjený od roku 2014 na Stanfordově univerzitě. Na rozdíl od modelu *Word2vec*, který pro vytvoření sémantické reprezentace slov využívá neuronovou síť, model *GloVe* je založen na snížení dimenze matice, která obsahuje počty vzájemných výskytů slov ve vstupním korpusu.

Matici, která ukládá jednotlivé vzájemné výskyty slov, označíme X. Poté prvky matice X_{ij} udávají počet, kolikrát se slovo j vyskytuje v okolí slova i. Současně $X_i = \sum_k X_{ik}$ představuje celkový počet výskytů všech slov v okolí slova i. Následně lze určit pravděpodobnost $P_{ij} = P(i|j) = X_{ij}/X_i$, která udává, jak často se slovo j vyskytuje v okolí slova i. Samotná matice však ještě nepředstavuje výslednou reprezentaci slov pomocí vektorů.

Výslednou cenovou funkci, která je založena na vážené metodě nejmenších čtverců, můžeme posléze definovat jako:

$$J(\theta) = f(X_{ij})(w_i^T \tilde{w}_j + b_i + \tilde{b_j} - \log X_{ij})^2, \tag{2.4}$$

kde symbol V označuje velikost slovníku a $f(X_{ij})$ takzvanou váhovou funkci (anglicky *weighting function*). Jedná se o spojitou neklesající funkci, která slouží pro vylepšení vlastností modelu, jelikož častějším slovům přiřazuje menší váhu a naopak slovům, která se vyskytují spíše zřídka, přiřazuje větší váhu. Dle výše uvedeného článku je nejlépe fungující volbou parametrizovatelná funkce definovaná jako:

$$f(x) = \begin{cases} (x/x_{max})^\alpha & \text{pokud } x < x_{max}. \\ 1 & \text{jinak.} \end{cases} \tag{2.5}$$

Výhodou modelu *GloVe* je především znatelně kratší doba nutná pro natrénování oproti předchozímu modelu *Word2vec*. Dle autorů článku [23] je model *GloVe* vhodný pro úlohy zabývající se slovní podobností, především však pro úlohu rozpoznávání pojmenovaných entit (anglicky *named entity recognition* nebo také zkráceně *NER*).

2.4 FastText

Třetí a poslední zde uvedený model nazvaný **fastText** byl publikován v roce 2017 a vychází z modelu *skip-gram* [3]. Vylepšením oproti původnímu modelu je reprezentace slov pomocí sumy takzvaných n-gram vektorů. Pojem n-gram označuje sled n prvků (jako například slova či písmena), které se objevují v delší posloupnosti. Pokud například zvolíme $n = 3$, můžeme následně slovo *hokej* vyjádřit *n-gramy*, respektive trigramy:

<ho, hok, oke, kej, ej>.

Speciální symboly < a > představují hranice daného slova, respektive jeho začátek a konec. Díky využití n-gramů je model *fastText* schopen na rozdíl od předchozích modelů lépe reprezentovat slova, která se ve vstupním korpusu vyskytují velmi zřídka nebo se tam nevyskytují vůbec, ale jsou si lexikálně podobná s jinými slovy nalézajícími se v korpusu.

Dle dostupné literatury [3] tento model překonává jak *Word2vec* tak *Glove*, a proto byl v rámci této diplomové práce použit pro všechny experimenty.

3 Lineární transformace

Problematika lineárních transformací je detailně popsána například v článku [4]. Označíme-li symbolem $\boldsymbol{L}$ množinu jazyků, poté slovo $w^x \in \boldsymbol{V}^x$ označuje slovo ze zvoleného jazyka $x \in \boldsymbol{L}$, kde symbol $\boldsymbol{V}^x$ představuje slovník daného jazyka. Lineární transformaci mezi sémantickými prostory můžeme posléze definovat jako:

$$\mathcal{S}^{x \to y}(w^x) = \mathbf{T}^{x \to y} \mathcal{S}^x(w^x) \tag{3.1}$$

Jedná se tedy o vynásobení transformační maticí $\mathbf{T}^{x \to y} \in \mathbb{R}^{d \times d}$. Pomocí násobení matic lze provést takzvanou *afinní transformaci*. Jde o zobrazení jednoho afinního prostoru do jiného pomocí operace posunutí, otočení, změny měřítka a zkosení [22]. Složením uvedených operací je maticové násobení, jehož výsledkem je opět matice v $\mathbb{R}^{d \times d}$.

Pro sestavení transformační matice $\mathbf{T}^{x \to y}$ mezi dvěma zvolenými jazyky $x, y \in \boldsymbol{L}$, například *angličtinou* a *italštinou*, je nezbytné nejdříve vytvořit dvojjazyčný slovník $\boldsymbol{D}^{\boldsymbol{x \to y}}$. Konkrétně se jedná o množinu m předem zvolených slovních dvojic $(w^x, w^y) \in \boldsymbol{D}^{\boldsymbol{x \to y}}$, kde $\boldsymbol{D}^{x \to y} \subset \boldsymbol{V}^x \times \boldsymbol{V}^y$. Velikost výsledného slovníku je $|\boldsymbol{D}^{x \to y}| = m$. Pro sestavení matic $\mathbf{X} \in \mathbb{R}^{m \times d}$ a $\mathbf{Y} \in \mathbb{R}^{m \times d}$, které jsou nezbytné pro určení výsledné transformace, využijeme m uspořádaných výše uvedených slovních párů (w^x, w^y), přesněji jejich odpovídajících slovních vektorů $(\mathcal{S}^x(w^x), \mathcal{S}^y(w^y))$.

Obsahem této kapitoly je představení a analýza aktuálně dostupných metod lineární transformace mezi sémantickými prostory, které reprezentují rozličné jazyky. Celkem budou představeny tři odlišné postupy. Konkrétně se jedná o *metodu nejmenších čtverců*, *ortogonální transformaci* a dále také transformaci pomocí *kanonické korelace*. Každá z výše uvedených metod poskytuje jisté výhody i nevýhody, které budou dále podrobněji popsány.

3.1 Metoda nejmenších čtverců

Metoda nejmenších čtverců známá již v 18. století[1] stále patří k základním metodám regresní analýzy v oblasti statistiky. Cílem metod regresní analýzy je odhad hodnoty náhodné veličiny z využitím znalosti jiné veličiny. Zmiňovanou metodu nejmenších čtverců lze využít pro nalezení přibližného

[1] První použití metody je připisováno německému matematikovi a fyzikovi *Carlu Friedrichovi Gaussovi* již v roce 1795.

řešení přeurčené soustavy lineárních rovnic 3.2.

$$\mathbf{Ax} = \mathbf{b} \tag{3.2}$$

Využití metody nejmenších čtverců pro nalezení transformační matice $\mathrm{T}^{x \to y}$ ve své práci navrhuje Mikolov et al. [18]. Vzorec pro nalezení transformační matice můžeme vidět v rovnici 3.3.

$$\hat{\mathbf{T}}^{x \to y} = \underset{\mathbf{T}^{\mathbf{x} \to \mathbf{y}}}{\arg\min} \|\mathbf{Y} - \mathbf{X}\mathbf{T}^{\mathbf{x} \to \mathbf{y}}\|_2^2 \tag{3.3}$$

Jedná se o optimalizační problém, který lze vyřešit využitím algoritmu gradientního sestupu, který je podrobněji představen v sekci 4.2. Pro nalezení transformační matice lze také využít analytického řešení metody nejmenších čtverců, jak lze vidět v rovnic 3.4.

$$\hat{\mathbf{T}}^{x \to y} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{Y}^\top \mathbf{X} \tag{3.4}$$

Analytické řešení je vzhledem k nutnosti násobit a invertovat matice vhodné spíše pro matice menších rozměrů, oproti gradientnímu sestupu se nicméně nejedná o iterativní algoritmus a není nutné volit dodatečné parametry. Výše uvedený výraz $(\mathbf{X}^\top \mathbf{X})^{-1}$ však nemusí být vždy invertibilní. Z tohoto důvodu je ve většině případů pro vypočtení inverzní matice nutné využít takzvanou **Moore–Penroseovu pseudoinverzi**.

3.2 Ortogonální transformace

Druhou zmiňovanou transformací je takzvaná **ortogonální transformace**. Tato transformace byla představena v článku [1]. Autoři práce zdůrazňují, že transformační matice by měla být ortogonální, jelikož zachovává úhly mezi jednotlivými body (slovy) v prostoru. Ortogonální transformace je tedy transformace pomocí metody nejmenších čtverců s omezením, že transformační matice $\mathbf{T}^{x \to y}$ musí být ortogonální.

Řádky i sloupce ortogonální matice jsou ortonormální vektory. Z tohoto faktu plyne velmi významná vlastnost ortogonální matice, což znamená, že její transponovaná matice je zároveň i její inverzní maticí, jak můžeme vidět v následující rovnici:

$$Q^\top = Q^{-1} \tag{3.5}$$

Optimální transformační matice $\mathbf{T}^{x \to y}$ je v případě ortogonální transformace určena pomocí vzorce:

$$\hat{\mathbf{T}}^{x \to y} = \mathbf{V}\mathbf{U}^\top. \tag{3.6}$$

Uvedené matice $\mathbf{U}$ a $\mathbf{V}$ jsou získány pomocí *singulárního rozkladu* neboli **SVD** (viz sekce 3.2.1) matic $\mathbf{Y}^\top\mathbf{X}$.

Nejvýznamnější vlastností ortogonální transformace je fakt, že zachovává hodnoty skalárního součinu, respektive zachovává úhly mezi vektory i po provedení transformace daného sémantického prostoru. Tento fakt je umožněn právě zásluhou ortogonality výsledné transformační matice $\mathbf{T}^{x\to y}$. Díky tomu je zabráněno degradaci kvality výsledného sémantického prostoru.

3.2.1 Singulární rozklad

Singulární rozklad či **SVD** matice je faktorizace reálné či komplexní matice $\mathbf{M}$ o velikosti $m \times n$ na maticový součin:

$$\mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*,$$

kde matice $\mathbf{U}$ je reálná nebo komplexní unitární[2] matice o rozměrech $m \times n$, matice $\mathbf{V}$ je reálná nebo komplexní unitární matice $n \times n$ a $\mathbf{\Sigma}$ je obdélníková matice $m \times n$, která má nezáporná čísla na diagonále. Tato čísla jsou označována jako **singulární hodnoty** matice $\mathbf{M}$. Symbol hvězdičky, který lze vidět u matice $\mathbf{V}^*$, označuje takzvanou *konjugovanou* matici. Jedná se o transponovanou matici s komplexně sdruženými čísly. Singulární rozklad představuje klíčový teorém lineární algebry, jelikož jej lze aplikovat na všechny matice a daná faktorizace vždy existuje [7].

3.3 Transformace pomocí kanonické korelace

Třetí uvedená transformace, která byla představena v článku [8], využívá pro nalezení transformační matice **kanonickou korelační analýzu**. Kanonická korelační analýza je metoda, kterou lze využít při výzkumu vztahů a závislostí mezi dvěma vícerozměrnými proměnnými, například vektory. Cílem této metody je nalezení bázových vektorů pro každou dvojici proměnných, tak aby korelace projekce těchto proměnných na bázové vektory byla vzájemně maximální. Pro nalezení hledaných bázových vektorů lze využít algoritmus prezentovaný v článku [11].

Proces nalezení transformační matice se skládá z několika kroků. Prvním krokem je nalezení dvojice vektorů $\mathbf{c}_1^x \in \mathbb{R}^d$ a $\mathbf{c}_1^y \in \mathbb{R}^d$. Je zvolena taková dvojice vektorů, jejíž projekce $\mathbf{X}\mathbf{c}_1^x$, $\mathbf{Y}\mathbf{c}_1^y$ dosahuje nejvyšší hodnoty *Pearsonova korelačního koeficientu*, který je detailně popsán v sekci 6.5.3. Po nalezení prvního nejlepšího páru je dalším krokem nalezení druhé nejlepší

[2]Unitární maticí rozumíme matici, pro kterou platí $\mathbf{U}^*\mathbf{U} = \mathbf{U}\mathbf{U}^* = \mathbf{I}$.

dvojici vektorů $(\mathbf{c}_2^x, \mathbf{c}_2^y)$. Avšak druhou dvojici hledáme pouze v podprostoru, který je ortogonální k první dvojici nalezených vektorů $(\mathbf{c}_1^x, \mathbf{c}_1^y)$. Cílem tohoto procesu je nalézt celkem d párů vektorů. Obecně lze k-tou dvojici vektorů nalézt pomocí vzorce:

$$\mathbf{c}_k^x, \mathbf{c}_k^y = \arg\min_{\mathbf{c}^x, \mathbf{c}^y} corr(\mathbf{X}\mathbf{c}^x, \mathbf{Y}\mathbf{c}^y), \tag{3.7}$$

kdy zároveň platí, že:

$$(\mathbf{X}\mathbf{c}^x) \cdot (\mathbf{X}\mathbf{c}_i^x) = 0 \text{ a } (\mathbf{Y}\mathbf{c}^y) \cdot (\mathbf{Y}\mathbf{c}_i^y) = 0 \tag{3.8}$$

pro každé $1 \leq i < k$. Nalezené vektory lze posléze reprezentovat jako dvě matice $\mathbf{C}^x \in \mathbb{R}^{d \times d}$ a $\mathbf{C}^y \in \mathbb{R}^{d \times d}$. Sloupce těchto matic odpovídají nalezeným vektorům $\mathbf{c}_k^x$ a $\mathbf{c}_k^y$. Výslednou transformační matici $\hat{\mathbf{T}}^{x \to y}$ lze následně získat pomocí rovnice:

$$\hat{\mathbf{T}}^{x \to y} = \mathbf{C}^x (\mathbf{C}^y)^{-1} \tag{3.9}$$

Obsahem následující kapitoly bude představení nelineárních transformací, respektive transformací, které využívají neuronové sítě. Do těchto transformací je oproti výše představeným lineárním transformacím přidán nelineární prvek v podobě aktivační funkce.

4 Nelineární transformace

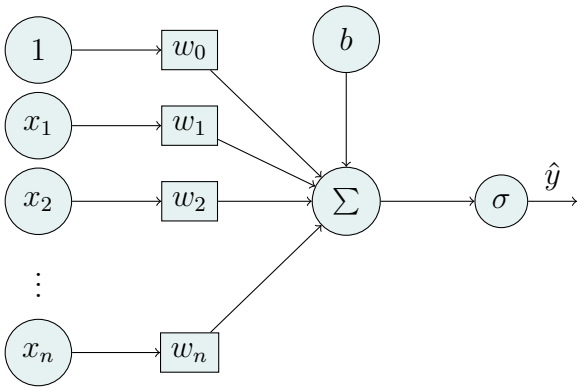
V této kapitole si detailně představíme a analyzujeme první skupinu metod pro vytvoření transformací mezi sémantickými prostory pomocí neuronových sítí, tedy takzvané nelineární transformace. Začátek kapitoly bude věnován krátkému představení neuronových sítí a běžně používaných aktivačních funkcí, které reprezentují nelineární prvek. Následně se zaměříme na navržené neuronové sítě, které byly použité v rámci diplomové práce.

4.1 Neuronové sítě

Přirozeným krokem při návrhu metod transformace bylo využití umělých neuronových sítí, které zejména v posledních letech zažívají mnoho úspěchů v rozmanitých technologických i humanitních oborech. Umělé neuronové sítě se svým návrhem a funkcemi snaží přiblížit samotné struktuře lidského mozku. Základní prvek umělé neuronové sítě je stejně jako v případě mozku neuron, respektive umělý neuron. Jednotlivé umělé neurony jsou mezi sebou propojeny. Tato propojení, představující synapse mozku, jsou schopná přenášet vzruch ostatním neuronům.

Historie neuronových sítí sahá až do 40. let minulého století, kdy byl Warrenem McCullochem a Walterem Pittsem představen *umělý neuron* [16]. Výstup tohoto neuronu byl 0 či 1 v závislosti na tom, zda vážená suma vstupních signálů daného neuronu překročila prahovou hodnotu. Pro tento neuron však ještě nebyla navržena žádná metoda pro jeho natrénování. Značného pokroku na teoretickém poli bylo dosaženo na konci 50. let minulého století, kdy Frank Rosenblatt navrhl *perceptron* [25]. Jednalo se o základní model umělého neuronu, který již bylo možné učit dle daných pravidel. Avšak jejich použití bylo limitované pouze na lineárně separovatelné úlohy. Z tohoto důvodu byl zájem o neuronové sítě utlumen až do příchodu vícevrstevnatých perceptronů a popularizace dodnes využívané metody učení takzvané *backpropagation* v 80. letech. V této době bylo prokázáno, že lze uvedenou metodu využít pro naučení neuronových sítí [26].

Backpropagation je proces zpětného šíření chyby neuronovou sítí, který spočívá v úpravě vah, respektive hodnot propojení jednotlivých neuronů. Jedná se o specifický případ gradientního sestupu, který je dodnes používán pro naučení neuronových sítí a detailně je představen v sekci 4.2. Proces učení je výkonnostně velmi náročný, takže především z tohoto důvodu se popularita neuronových sítí začala zvyšovat až v posledních letech s příchodem



Obrázek 4.1: Jednoduchý perceptron

moderního hardwaru, především výkonných grafických karet, které dokážou tento proces několikanásobně urychlit.

Neuronové sítě dále oproti lineárním transformacím probraným v minulé kapitole obsahují nelineární prvek, který jim umožňuje řešit i problémy, které nejsou lineárně separovatelné. Jedná se o aktivační funkce. Nejznámější zástupci aktivačních sítí používaných v neuronových sítích budou podrobněji představeny v sekci 4.1.2.

4.1.1 Perceptron

Schéma jednoduchého perceptronu můžeme vidět na obrázku 4.1. Ve své podstatě se jedná o analogii k neuronu ve skutečném lidském mozku. Matematicky můžeme perceptron popsat pomocí rovnice 4.1.

$$h_\theta(x) = \sigma(\sum w_i x_i + b) \tag{4.1}$$

Symbol σ zde představuje aktivační funkci daného neuronu. Dále symbol w_i reprezentuje váhu spojení mezi daným neuronem a neuronem, který zasílá signál x_i. Písmeno b označuje práh. Perceptron představuje základní stavební prvek neuronové sítě.

4.1.2 Aktivační funkce

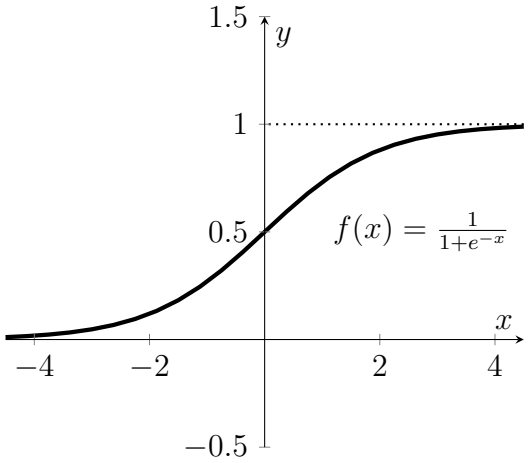
V této části si představíme tři běžně používané aktivační funkce. Konkrétně se jedná o funkci *sigmoid*, *hyperbolický tangens* a *ReLU* (*Rectified Linear Unit*). Poslední dvě zmíněné byly využity při samotném návrhu umělých neuronových sítí sloužících pro transformaci sémantických prostorů.

Sigmoid

Aktivační funkce **sigmoid**, která je významným příkladem *logistické* funkce, je patrně nejznámější aktivační funkcí, se kterou se můžeme v oblasti umělé inteligence setkat. Předpisem funkce sigmoid je následující rovnice:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{4.2}$$

Graf této funkce můžeme vidět na obrázku 4.2. V grafu si lze povšimnout, že obor hodnot funkce sigmoid je roven intervalu $(0, 1)$. Jedná se jednoduchou



Obrázek 4.2: Aktivační funkce *sigmoid*

funkci, která je vhodná spíše pro počáteční experimenty, kdy je přednější pochopení dané problematiky. Funkce sigmoid není ve většině případů příliš často využívána v reálných modelech, neboť se s jejím použitím pojí několik problémů.

Patrně nejvýraznějším nedostatkem funkce sigmoid je takzvaný problém *mizejícího gradientu*. K tomuto problému dochází v případě, že se výstup aktivace neuronu blíží hodnotě 0 či 1, tedy okrajovým hodnotám oboru dané funkce. Gradient je v této oblasti téměř nulový, čímž je velmi znesnadněn klíčový proces učení neuronové sítě neboli *backpropagation*. Další problém, který se pojí s použitím této funkce, je, že její výstup není vycentrován okolo nuly.

Z výše uvedených důvodů nebyla funkce sigmoid použita pro návrh neuronových sítí použitých v rámci diplomové práce.

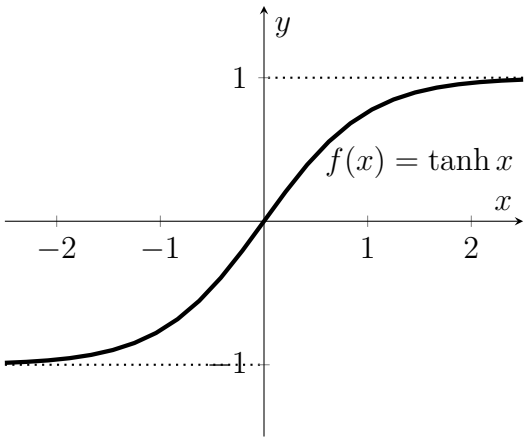
Hyperbolický tangens

Druhou aktivační funkcí, která již byla pro návrh neuronových sítí použita, je **hyperbolický tangens** nebo také krátce *tanh*. Předpis této funkce je

uveden v rovnici 4.3.

$$f(x) = \tanh x = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{4.3}$$

Graf této funkce můžeme vidět na obrázku 4.3. Z grafu je vidno, že obor hodnot je oproti funkci sigmoid roven intervalu $(-1, 1)$. Je tedy patrné, že vy-



Obrázek 4.3: Aktivační funkce *hyperbolický tangens*

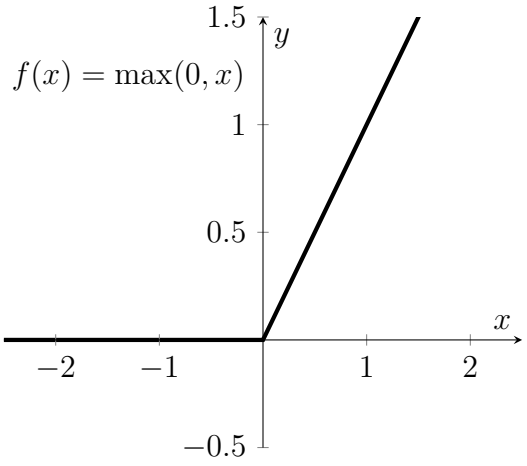
užitím funkce hyperbolický tangens lze odstranit druhý zmiňovaný problém funkce sigmoid, jelikož je vycentrován okolo nuly. Z tohoto důvodu je v praxi aktivační funkce hyperbolický tangens preferovaná před aktivační funkcí sigmoid. Avšak stejně jako v předchozím případě i zde může nastat problém mizejícího gradientu, pokud se aktivační hodnota neuronu blíží okrajovým hodnotám oboru hodnot funkce.

ReLU

Třetí aktivační funkcí, v současné době hojně rozšířenou, je takzvaná **ReLU**, jejíž název vychází z anglického výrazu **rectified linear unit**, což lze přeložit jako *usměrněná lineární funkce*. Předpis této funkce lze vidět v rovnici 4.4.

$$f(x) = \max(0, x) \tag{4.4}$$

Jak můžeme vidět v grafu 4.4, obor hodnot funkce ReLU leží v intervalu $[0, \infty)$. Značnou výhodou aktivační funkce ReLU je její výrazná jednoduchost oproti výše uvedeným aktivačním funkcím sigmoid a hyperbolický tangens, které provádí komplexní matematické operace. Jednoduchost této funkce umožňuje výrazným způsobem urychlit učení neuronové sítě. Díky



Obrázek 4.4: Aktivační funkce *ReLU*

svým vlastnostem je vhodná především jako aktivační funkce pro skryté vrstvy umělé neuronové sítě.

Avšak ani použití funkce ReLU není zcela bezproblémové. V případě, že je hodnota gradientu procházejícího přes neuron příliš vysoká, může nastat situace, kdy daný neuron „odumře“ a již se nikdy pro žádný trénovací vzorek neaktivuje. Řešením tohoto problému je využití upravené funkce ReLu, například takzvané *Leaky ReLU*, jejíž předpis lze vidět v rovnici 4.5.

$$f(x) = \begin{cases} x & \text{pokud } x > 0 \\ 0.01x & \text{jinak.} \end{cases} \tag{4.5}$$

V následující kapitole si blíže představíme návrh jednoduché neuronové sítě použité v diplomové práci, která ve své struktuře neobsahuje skrytou vrstvu.

4.2 Gradientní sestup

Gradientní sestup (anglicky *gradient descent*) je iterativní optimalizační algoritmus prvního řádu sloužící pro nalezení lokálního minima zvolené funkce. Nutnou podmínkou pro použití tohoto algoritmu je diferencovatelnost dané funkce, musí se tedy jednat o funkci, která má v určitém bodě diferenciál. Matematicky můžeme gradientní sestup vyjádřit pomocí rovnice 4.6. Symbol α reprezentuje rychlost učení a J cenovou funkci. Vizualizaci minimalizace cenové funkce lze vidět v grafu na obrázku 4.5. Symbol $\hat{\theta}$ zde označuje hle-

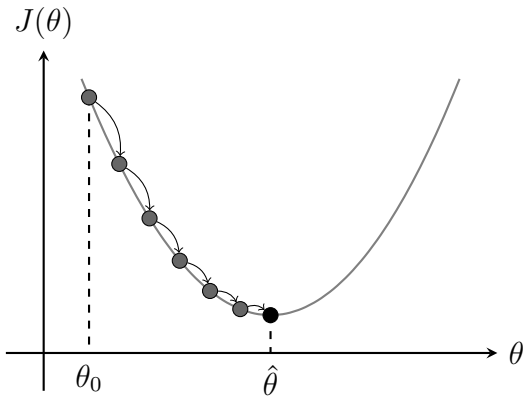
dané lokální (v tomto případě současně globální) minimum funkce $J(\theta)$.

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial t} J(\theta) \tag{4.6}$$

Gradientní sestup můžeme dále rozdělit na tři základní druhy, jmenovitě:

- **dávkový** gradientní sestup,
- **mini-dávkový** gradientní sestup a
- **stochastický** gradientní sestup.

Dávkový gradientní sestup v každé iteraci algoritmu pracuje s m trénovacími vzorky $(x^{(i)}, y^{(i)})$ z trénovací množiny, kde m označuje velikost trénovací množiny. Oproti tomu stochastický gradientní sestup v každém kroku mini-



Obrázek 4.5: Vizualizace gradientního sestupu

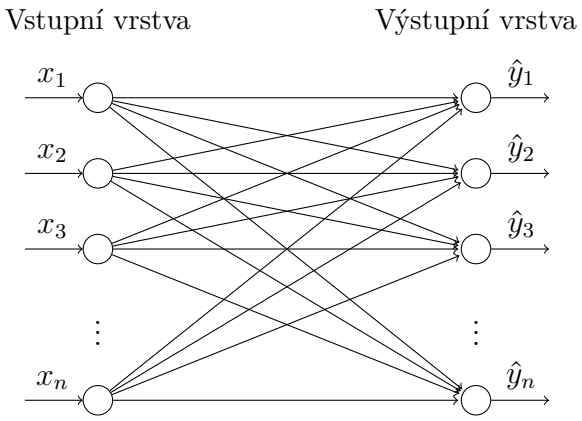
malizuje cenovou funkci pouze s jedním trénovacím vzorkem. Díky tomuto faktu je stochastický gradientní sestup schopen lépe předcházet uváznutí v lokálním minimu. Mini-dávkový gradientní sestup kombinuje vlastnosti obou předchozích druhů. Nepracuje s celou trénovací množinou, ale pouze s její podmnožinou o předem zvolené velikosti. Varianty gradientního sestupu jsou též využity pro trénování neuronových sítí v sekci 4.3 a 4.4.

4.3 Neuronové sítě bez skryté vrstvy

Cílem prvních experimentů s neuronovými sítěmi bylo pokusit se spojit již fungující lineární transformace a obohatit je o nelineární prvek, který přináší právě neuronové sítě. Z tohoto důvodu prvotní návrh neuronových sítí v rámci diplomové práce obsahoval pouze vstupní a výstupní vrstvu. Jednalo se tak ve své podstatě o metodu nejmenších čtverců. Na rozdíl od

této metody však byla k výstupní vrstvě přidána aktivační funkce, která představovala samotný nelineární prvek. Jako aktivační funkce byl zvolen hyperbolický tangens díky svým vhodným vlastnostem pro tento problém, konkrétně rozsah intervalu a centrování kolem nuly.

Strukturu takovéto neuronové sítě můžeme vidět na obrázku 4.6. Vstupem neuronové sítě je vektor $\mathbf{x} = (x_1, x_2, x_3, \ldots, x_n)$. V našem případě je počet složek vektoru n roven d, tedy dimenzi sémantického prostoru $\mathbf{X}$. Každý neuron ze vstupní vrstvy je následně propojen se všemi neurony z výstupní vrstvy. Výstupem neuronové sítě je poté vektor $\hat{\mathbf{y}} = (\hat{y}_1, \hat{y}_2, \hat{y}_3, \ldots, \hat{y}_n)$, který již představuje slovní vektor transformovaný do sémantického prostoru $\mathbf{Y}$.

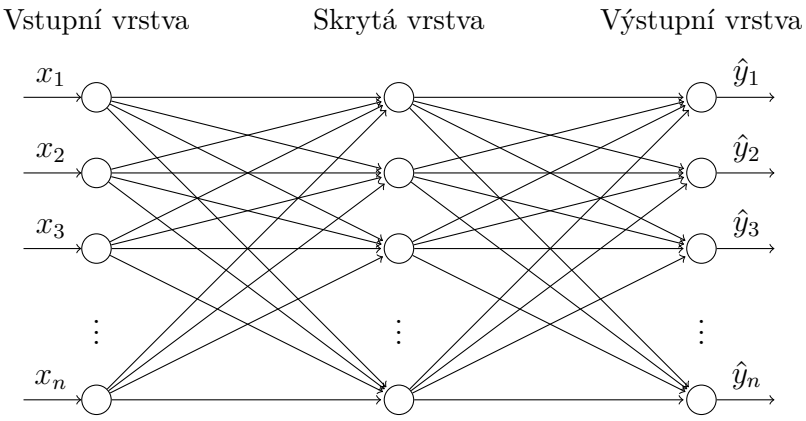


Obrázek 4.6: Struktura neuronové sítě bez skryté vrstvy

4.4 Neuronové sítě se skrytou vrstvou

Druhým přístupem k návrhu neuronových sítí, který byl použit v rámci diplomové práce, bylo přidání skrytých vrstev do jejich struktury. Využití neuronových sítí s velkým počtem vrstev se souhrnně označuje termínem hluboké učení (anglicky *deep learning*). Vyšší počet skrytých vrstev umožňuje neuronovým sítím naučit se řešit velmi komplexní problémy, jako například klasifikace obrázků, hraní stolních či počítačových her nebo stále populárnější řízení autonomních vozidel [27, 28]. Vyšší počet skrytých vrstev u jednoduchých problémů však může vést k přetrénování neuronové sítě na trénovacích datech. Přetrénování sítě výrazným způsobem omezuje její možnost generalizovat naučené poznatky a aplikovat je na data mimo trénovací množinu. Tento problém se samozřejmě může vyskytnout také během trénování neuronových sítí pro transformace sémantických prostorů. Z tohoto důvodu bylo při návrhu neuronových sítí dbáno na omezení jejich nadměrné složitosti.

Ukázku navržené neuronové sítě se skrytou vrstvou lze vidět na obrázku 4.7. Stejně jako v předchozím případě nově navržená neuronová síť obsahuje vstupní i výstupní vrstvu, která se skládá ze 300 neuronů, což odpovídá počtu složek slovních vektorů. Oproti neuronovým sítím z předchozí kapitoly byla mezi vstupní a výstupní vrstvu přidána skrytá vrstva, která také obsahovala 300 neuronů. Jako kritérium pro minimalizaci cenové funkce neuronové sítě byla stejně jako v předchozím případě použita kosinová podobnost a střední kvadratická chyba.



Obrázek 4.7: Struktura neuronové sítě se skrytou vrstvou

4.5 Cenové funkce

4.5.1 Střední kvadratická chyba

Jako první cenová funkce, která slouží jako kritérium pro minimalizace chyb neuronové sítě, byla využita střední kvadratická chyba (anglicky *mean squared error* nebo MSE), jejíž vzorec můžeme vidět v rovnici 4.7.

$$\text{MSE} = \frac{1}{m}\sum_{i=1}^{m}(Y_i - \hat{Y}_i)^2 \tag{4.7}$$

Symbol Y_i zde představuje vzorek pozorované veličiny a $\hat{Y}_i$ predikci daného modelu pro tento vzorek. Jedná se tedy o sumu kvadrátů chyb, nebo také odchylek predikovaných hodnot od naměřených. Střední kvadratickou chybu může výrazně ovlivnit výskyt extrémních hodnot neboli takzvaných *outlierů*. Alternativou je střední absolutní chyba, která kvadratickou funkci nahrazuje absolutní hodnotou. Absolutní hodnota však není diferencovatelná po celém definičním oboru, což znesnadňuje optimalizaci pomocí gradientního sestupu, kde je směr gradientu určen pomocí derivace.

4.5.2 Kosinová podobnost

Druhou cenovou funkcí, která byla použita v rámci diplomové práce, je kosinová podobnost. Současně se jedná o často využívanou metriku udávající podobnost vektorů, která je detailněji popsána v kapitole 6.5.2. Avšak na rozdíl od střední kvadratické chyby hodnota kosinové podobnosti se vzrůstající podobností vektorů roste. Aby bylo možné hodnotu této cenové funkce minimalizovat pomocí gradientního sestupu, využívá se **záporná** hodnota kosinové podobnosti. Minimální hodnota této cenové funkce je posléze rovna -1, kdy je zároveň podobnost porovnávaných vektorů nejvyšší.

4.5.3 *Hinge-loss*

Poslední a současně velmi důležitá a specifická cenová funkce, která byla využita v rámci diplomové práce, je inspirovaná speciální lineární transformací z článku [15]. Cílem metody navržené ve výše uvedeném textu je snížení *hubness* (viz sekce 6.4) výsledného transformovaného sémantického prostoru.

Cílem této metody je zařadit správný překlad y_i slova x_i výše než všechny ostatní možné překlady y_j z cílového sémantického prostoru. Stanovením **minimální vzdálenosti**, kterou má predikovaný vektor y_i od všech ostatních vektorů y_j udržovat, lze následně vylepšit vlastnosti transformace. Primárním účelem je samozřejmě snížení *hubness* transformovaného sémantického prostoru, které v důsledku může vylepšit výsledky i v ostatních sledovaných kategoriích, jež jsou detailně představeny v sekci 6. Výslednou cenovou funkci výše uvedené transformace lze definovat jako:

$$J(\theta) = \sum_{n \in N} \max\{0, \gamma + D(\hat{\mathbf{y}}, \mathbf{y}) - D(\mathbf{y}, \mathbf{n})\}, \tag{4.8}$$

kde symbol $\mathbf{N} = \mathcal{N}(\hat{\mathbf{y}}, \mathbf{y}, \mathbf{Y})$ označuje funkci, jejímž výstupem je množina negativních příkladů $\mathbf{N}$. V tomto případě množina $\mathbf{N}$ zahrnuje všechny slovní vektory z $\mathbf{Y}$ vyjma vektoru y_i. Symbol D zde označuje zvolenou metriku vzdálenosti, která slouží k určení podobnosti slovních vektorů, respektive samotných slov. Velmi důležitý je především symbol γ, jenž udává minimální vzdálenost, kterou má mít správný vektor $\mathbf{y}$ od negativních příkladů $\mathbf{n}$.

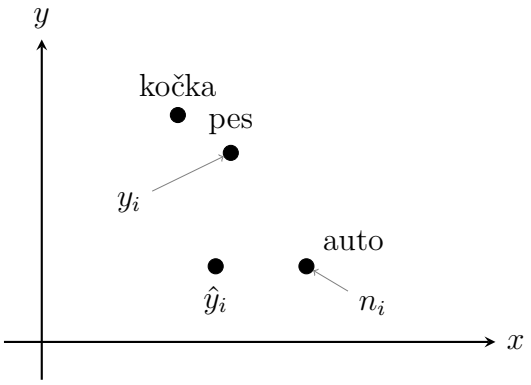
Z výpočetního hlediska v praxi není možné do množiny negativních příkladů zařadit všechny slovní vektory ze sémantického prostoru $\mathbf{Y}$ (vyjma správného vektoru y_i), jelikož by cenová funkce musela být spočtena jako suma celkem $m \times (m-1)$ položek, kde symbol m označuje počet vektorů v daném sémantickém prostoru. V sémantických prostorech je mnohdy běžné, že hodnota $m > 10^5$. Lazaridou et al. [15] však ve své práci uvádí, že tento

problém lze vyřešit využitím pouze jednoho negativního příkladu, který lze získat pomocí následující rovnice:

$$\mathcal{N}(\hat{\mathbf{y}}, \mathbf{y}, \mathbf{Y}) = \underset{1 \leq i \leq m, i \neq j}{\arg\min} \{D(\hat{\mathbf{y}}_i, \mathbf{y}_j) - D(\mathbf{y}_i, \mathbf{n}_j)\} \tag{4.9}$$

Pomocí vzorce 4.9 lze nalézt negativní příklad $\mathbf{n}_j$. Tento negativní příklad představuje slovo, které je svým významem co nejblíže predikovanému vektoru $\hat{\mathbf{y}}_i$ a současně co nejdále od správného překladu y_i slova x_i. Stejně jako v předchozí rovnici 4.8 i zde představuje symbol D zvolenou metriku vzdálenosti. Negativní příklad n_i tedy představuje závažnou chybu, která z našeho pohledu přináší nejvíce informací.

Ukázku nalezení negativního příkladu můžeme vidět na obrázku 4.8. Pro přehlednost je tento sémantický prostor vyobrazen ve dvou dimenzích a slovní vektor je v tomto případě reprezentován pouze pomocí bodu. Z ob-



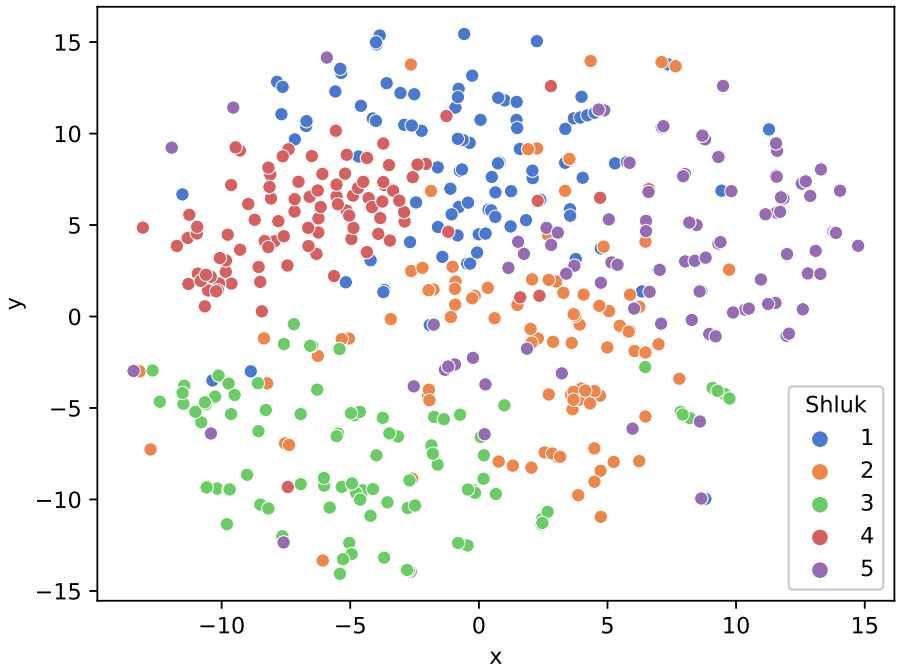
Obrázek 4.8: Nalezení negativního příkladu

rázku je patrné, že se jedná o český sémantický prostor. Do tohoto sémantického prostoru se například pokoušíme z anglického jazyka transformovat slovo *dog*, které odpovídá českému ekvivalentu *pes*. Slovo *pes* v tomto případě představuje správný překlad, tedy konkrétně vektor y_i. Symbol $\hat{y}_i$ označuje námi predikovaný vektor, tedy již transformovaný vektor s využitím předem zvolené transformace. V prostoru se současně vyskytují další dvě slova, konkrétně *kočka* a *auto*. Na první pohled je zřejmé, že slovo *kočka* je k hledanému překladu *pes* významově mnohem blíže než slovo *auto*. Z tohoto důvodu není *kočka* vhodným negativním příkladem, jelikož jsou si tato slova významově příliš blízká. Naproti tomu slovo **auto** je od hledaného překladu významově zcela odlišné, a v tomto případě tedy představuje námi hledaný negativní příklad, který lze označit symbolem n_i.

Následující kapitola bude věnována nově navržené metodě, která dále rozšiřuje myšlenku lineární transformace a s pomocí shlukování do ní přidává nový samostatný prvek.

5 Transformace s využitím shlukování

Jak naznačuje předchozí analýza, v současnosti je pro transformaci sémantických prostorů stále využíváno především lineárních transformací. Vzhledem ke komplexnosti této úlohy jsem se rozhodl aplikovat odlišný přístup, respektive transformaci s využitím shlukování, jíž dosud nebyla věnována pozornost. Klíčovou myšlenkou třetího přístupu pro vytvoření transformace v rámci diplomové práce, která je obsahem této kapitoly, bylo využití shlukové analýzy pro roztřídění slov sémantického prostoru do shluků[1], které sdílejí stejný či podobný význam. Cílem této metody bylo pokusit se s po-



Obrázek 5.1: Vizualizace sémantického prostoru po provedení shlukování

mocí lokálních transformací vytvořených pro jednotlivé shluky vylepšit vlastnosti základní lineární transformace. Hlavní přínos tohoto přístupu spočívá zejména v zanesení nové lokální informace do procesu transformace, díky níž

[1]Často se můžeme setkat také s anglickým pojmem *cluster*.

by byla výsledná transformace schopna tento proces zpřesnit pomocí údaje o tom, ke kterému shluku se transformované slovo řadí.

Výše uvedený přístup bylo možné využít díky faktu, že se slova stejného či podobného významu vyskytují v daném sémantickém prostoru blízko u sebe na rozdíl od slov s odlišným významem. Je tedy pravděpodobné, že slova jako *univerzita*, *profesor* či *student* se budou nacházet nedaleko od sebe v českém sémantickém prostoru, stejně tak jako slova *university*, *professor* nebo *student* v sémantickém prostoru reprezentujícím angličtinu.

Na obrázku 5.1 lze vidět vizualizaci předem zvolené podmnožiny sémantického prostoru po provedení shlukování do celkem pěti shluků pomocí metody *k-means* (viz sekce 5.2). Jedná se o konkrétní sémantický prostor použitý v rámci diplomové práce. Slovní vektory tohoto sémantického prostoru byly pomocí metody *t-SNE* neboli *t-distributed stochastic neighbor embedding* [29] určené pro nelineární snížení dimenzionality dat projektovány do dvoudimenzionálního prostoru. Shluky slovních vektorů jsou na obrázku na první pohled patrné. V některých případech si lze povšimnout, že se vizualizované shluky mírně prolínají, je však nutné si uvědomit, že snížením dimenze sémantického prostoru současně došlo k velmi výrazné ztrátě původní informace. Ve výsledku tedy jednotlivé shluky obsahují slova s podobným významem, a to díky faktu, že jsou slovní vektory schopné reprezentovat samotný význam slova.

Z výše uvedených důvodů do výpočtu výsledné transformace přidáváme nový prvek, kterým je transformační matice pro cluster k. Formálněji bychom mohli transformaci pro zvolené slovo w_k^x, které náleží do clusteru k, definovat pomocí rovnice 5.1.

$$\mathbf{T}_k^{x \to y} = \mathbf{T}_b \odot \mathbf{T}_k \tag{5.1}$$

Symbol $\mathbf{T}_b$ zde představuje základní transformační matici, jež vznikla libovolně zvolenou metodou pro lineární transformaci z kapitoly 3. Pro sestavení této matice se využijí všechny slovní dvojice (w^x, w^y) ze slovníku $\boldsymbol{D}^{x \to y}$. Oproti tomu transformační matice $\mathbf{T}_k$ označuje matici, která slouží pro transformaci pouze slov náležejících do shluku k. Pro její sestavení bylo využito pouze podmnožiny slovníku $\boldsymbol{D}$, která byla nejdříve určena pomocí zvoleného shlukovacího algoritmu. Symbol $\odot$ označuje blíže neurčenou základní matematickou operaci mezi danými maticemi. Pomocí provedených experimentů bylo následně ověřeno, že nejlepší výsledky jsou dosahovány využitím lineární kombinace obou matic.

5.1 Vážené shlukování

Využití principu shlukování pro zpřesnění lineární transformace jsem se rozhodl dále rozšířit zavedením koeficientů pro obě matice. Cílem specifického nastavení koeficientů, respektive vah těchto matic bylo další zpřesnění transformace. Výslednou transformaci můžeme posléze definovat jako:

$$\mathbf{T}_k^{x \to y} = (a \cdot \mathbf{T}_b) + (b \cdot \mathbf{T}_k), \tag{5.2}$$

kde koeficienty $a, b \in \mathbb{R}$. Rovnice 5.2 představuje specifický případ, kdy jsou oba koeficienty a a b rovny jedné, tedy obě matice mají během transformace stejnou váhu. Přesný postup, jakým byla stanovena hodnota obou koeficientů během experimentů, je stanoven v kapitole 7.5.2.

V následujících kapitolách si představíme dva významné algoritmy sloužící pro vytvoření shluků, které byly použity v diplomové práci. Konkrétně se jedná o poměrně populární algoritmus *k-means* a soubor metod souhrnně označovaných jako *hierarchické shlukování*.

5.2 Algoritmus k-means

Jako první možnost pro určení shluků v sémantickém prostoru byl využit iterační algoritmus *k-means*. S využitím tohoto algoritmu je n trénovacích vzorků, v našem případě slovních vektorů, rozděleno do k shluků, kde každý slovní vektor náleží shluku, jehož středovému bodu neboli *centroidu* je nejblíže. Samotný algoritmus se skládá ze dvou základních kroků, kterými jsou výpočet shluků a přesun centroidů.

Pokud symbolem $\mu_{c^{(i)}}$ označíme centroid shluku c, kterému byl přiřazen trénovací vzorek $x^{(i)}$, můžeme posléze definovat optimalizační kritérium pomocí vzorce 5.3.

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \left\| x^{(i)} - \mu_{c^{(i)}} \right\|^2 \tag{5.3}$$

Cílem tohoto algoritmu je tedy minimalizovat kvadráty vzdálenosti vzorků přiřazených k danému clusteru od odpovídajícího centroidu. Počáteční umístění centroidů jednotlivých clusterů lze volit libovolně, často se pro ně volí náhodné vzorky z trénovací množiny. Případně lze využít specializované inicializační algoritmy, jako je například často využívaný `k-means++`.[2] Algoritmus *k-means* vždy konverguje, není však zaručeno, že nalezne globální optimum.

Po dokončení počáteční inicializace k centroidů jednotlivých shluků označených $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$ se algoritmus *k-means* skládá ze dvou základních

kroků. Prvním krokem je nalezení nejbližšího shluku pro každý vzorek $x^{(i)}$, jak můžeme vidět v rovnici 5.4. Vzorek $x^{(i)}$ je poté přiřazen odpovídajícímu nejbližšímu shluku.

$$c^{(i)} = \arg\min_j \left\| x^{(i)} - \mu_j \right\|^2 \tag{5.4}$$

Druhým nezbytným krokem je vypočtení nové pozice pro centroid shluku po předchozím přiřazením nových vzorků do shluku. Tento krok lze vidět v rovnici 5.5.

$$\mu_j = \frac{1}{|c_k|} \sum_{i \in c_k} x^{(i)} \tag{5.5}$$

Algoritmus dosáhne konvergence v případě, že se již vzorky přiřazené k jednotlivých shlukům dále nemění. Další možností je ukončení algoritmu po provedení předem daného množství iterací, případně je změna cenové funkce mezi dvěma po sobě jdoucími iteracemi menší než určená tolerance ϵ.

5.3 Hierarchické shlukování

Druhou metodou, která byla zvolena pro vytvoření shluků, je takzvané *hierarchické shlukování*. Na rozdíl od algoritmu *k-means*, který byl detailně představen v předchozí kapitole, tato metoda neshlukuje do předem zadaného počtu k shluků. Oproti tomu provádí shlukování postupným spojováním menších shluků dohromady, tento postup nazýváme *aglomerativní*, popřípadě využívá takzvaných *divizních* metod, kdy probíhá dělení větších shluků na menší.

Aglomerativní přístup spočívá v postupném slučování menších shluků do větších, z nichž je následně vytvořena hierarchie všech shluků. Na začátku algoritmu je tedy každý slovní vektor považován za samostatný shluk. Hlavní nevýhodou algoritmu pro aglomerativní hierarchické shlukování je jeho poměrně vysoká časová náročnost $\mathcal{O}(n^3)$ a paměťová náročnost $\mathcal{O}(n^2)$, kde symbol n označuje počet shlukovaných bodů. Tento fakt velmi znesnadňuje jeho použití pro rozsáhlé datasety.

Z tohoto důvodu byl pro tvorbu diplomové práce zvolen *divizní* přístup, kdy jsou nejdříve všechna slova umístěna do jednoho shluku, který je následně rozdělován na menší. Vykonávání algoritmu lze zastavit v momentě, kdy již existuje požadovaný počet shluků k. V opačném případě je možné pokračovat až do chvíle, kdy jsou všechna slova v samostatném shluku. Pro tvorbu shluků lze v tomto případě využít rekurzivního volání již dříve představeného algoritmu *k-means*.

V následující kapitole si představíme jednotlivé datové sady a metriky, které byly použity k evaluaci výkonnosti všech dosud představených metod pro lineární i nelineární transformace.

6 Evaluace

V této kapitole si blíže představíme datové sady a metriky, které byly v rámci diplomové práce použité k objektivnímu posouzení funkčnosti a kvality všech posuzovaných transformačních metod. Výkonnost metod byla změřena za pomoci čtyř hlavních kategorií. Konkrétně se jednalo o *slovní podobnost* (sekce 6.1), *slovní analogie* (sekce 6.2), *strojový překlad* (sekce 6.3) a v neposlední řadě také *hubness* (sekce 6.4).

V následujících sekcích si detailně projdeme jednotlivé kategorie a uvedeme, co je jejich obsahem.

6.1 Slovní podobnost

První sledovanou kategorií, která sloužila pro ověření funkčnosti a kvality transformace, byla slovní podobnost. V experimentech byl využit dataset SemEval 2017 Task 2 [6]. Cílem experimentu bylo ověřit, do jaké míry lze s využitím zvolených transformací sémantického prostoru určit podobnost slov z odlišných jazyků, tedy kvalitu mezi-jazyčné transformace.

Tento dataset je tvořen soubory, které obsahují páry slov (w^x, w^y) pro dané dvojice jazyků x a y. Každému páru byla anotátory manuálně určena slovní podobnost. Tato podobnost je vyjádřena jako hodnocení z předem určené pětibodové škály od 0 do 4. Toto hodnocení je inspirované takzvanou Likertovou škálou [12], která se řadí mezi nejspolehlivější techniky měření postojů. Hodnota 0 zde označuje zcela odlišná slova z rozdílných oblastí, jako například dvojice *PlayStation* – herní konzole a *monarchy* – forma vlády. Dvojice, které byly označené hodnotou 4, představují synonyma. Příkladem synonym je v tomto případě například dvojice *motherboard* a *Mainboard*, kdy se v anglickém i německém jazyce jedná o stejný výraz, konkrétně *základní deska*.

Pro vypočtení podobnosti jednotlivých dvojic byla využita *kosinová podobnost*. Tato metrika je podrobně představena v kapitole 6.5.2. Výsledná hodnota pro slovní podobnost byla následně změřena jako průměr mezi hodnotami *Pearsonova korelačního koeficientu* (viz sekce 6.5.3) a *Spearmanova korelačního koeficientu* (viz sekce 6.5.4) mezi naměřenou podobností slov $sim(w^x, w^y)$ a vzorovými hodnotami určenými anotátory. Výsledné vyšší hodnoty pro slovní podobnost tedy naznačují kvalitnější transformaci.

6.2 Slovní analogie

Druhou neméně důležitou kategorií, která určuje kvalitu transformace, jsou slovní analogie. Realizace tohoto experimentu je inspirována článkem [5]. Pro jednoduchost si tuto kategorii můžeme představit jako tvrzení ve formě: slovo w_1 se má ke slovu w_2 jako slovo w_3 se má ke slovu w_4. Cílem tohoto experimentu je pokusit se předpovědět slovo w_4 za předpokladu, že mezi slovy v těchto dvojicích se jedná o totožné vztahy. Konkrétním příkladem v českém jazyce může být dvojice *Paříž* a *Francie*, u které se jedná o stejný vztah jako u dvojice z anglického jazyka *Madrid* a *Spain*, respektive vztah hlavní město a odpovídající stát.

Formálně lze slovní dvojici ze zdrojového jazyka x označit jako (w_1^x, w_2^x) a slovo z cílového jazyka y jako w_3^y. Pro nalezení hledaného slova w_4^y v cílovém sémantickém prostoru $\mathcal{S}^y$ je nejprve nutné sestavit hledaný vektor $\mathbf{v}$, který lze získat dle rovnice:

$$\mathbf{v} = \mathcal{S}^{x \to y}(w_2^x) - \mathcal{S}^{x \to y}(w_1^x) + \mathcal{S}^y(w_3^y) \tag{6.1}$$

Poté je nezbytné postupně projít všechna slova w^y, která jsou součástí slovníku V^y daného jazyka y, kromě slova w_3^x, které je z vyhledávání vyřazeno. Výsledný vektor w_4^y je nalezen dle následující rovnice:

$$\hat{w}_4^y = \arg\max_{w^y} \frac{\mathcal{S}^y(w^y) \cdot \mathbf{v}}{\|\mathcal{S}^y(w^y)\|_2 \cdot \|\mathbf{v}\|_2}. \tag{6.2}$$

Pokud platí, že je nalezený vektor $\hat{w}_4^y$ roven vektoru w_4^y, je následně odpověď zaznamenána jako správná.

Použité analogie se dělí do dvou základních druhů, konkrétně sémantické a syntaktické, které obsahují celkem 9 odlišných kategorií. Pro sémantické analogie je klíčový význam slova. Datová sada obsahuje celkem tři sémantické kategorie, jmenovitě:

- **rodina**: tato skupina obsahuje rodinné vztahy založené na rozdílných pohlavích. Příklad může být dvojice *syn* a *dcera*.
- **stát – měna**: jednotlivé páry této kategorie obsahují stát a měnu, která je platidlem v dané zemi, například *Mexiko* a *peso*.
- **hlavní město – stát**: poslední sémantická kategorie obsahuje dvojice představující hlavní město a odpovídající stát, například *Paříž* a *Francie*.

Pro syntaktické kategorie je přednější tvar slova. Dataset obsahuje celkem 6 různých syntaktických kategorií, konkrétně se jedná o tyto:

- **stát – přídavné jméno**: první kategorie reprezentuje vztah slova představujícího stát jako podstatného jména a odpovídajícího přídavného jména, například *Španělsko* a *španělský*.

- **pozitiv – komparativ**: tato kategorie obsahuje relace zkoumající první stupeň přídavného jména (*pozitiv*) a druhý stupeň (*komparativ*). Jedním z příkladů může být například dvojice *malý* a *menší*.

- **pozitiv – superlativ**: tato kategorie je velmi podobná předchozí, avšak zkoumá vztah prvního stupně přídavného jména se třetím stupněm, takzvaným *superlativem*. Příkladem může být dvojice *velký* a *největší*.

- **antonyma**: jednotlivé páry této kategorie obsahují přídavné jméno a slovo opačného významu (*antonymum*), například *vhodný* a *nevhodný*.

- **singulár – plurál**: tato kategorie zkoumá relace podstatných jmen uvedených v jednotném (*singulár*) a množném čísle (*plurál*). Příkladem může být dvojice *pes* a *psi*.

- **prézens – préteritum**: poslední kategorie obsahuje dvojice sloves uvedené v přítomném (*prézens*) a minulém čase (*préteritum*), například *být* a *byl*.

6.3 Strojový překlad

Cílem třetí sledované kategorie, jak již její název napovídá, je ověřit schopnost zvolené transformace nalézt správný překlad slova w^x ze zdrojového jazyka x do cílového jazyka y. Slovo w^x, respektive odpovídající slovní vektor $\mathcal{S}^x(w^x)$ je nejdříve transformován pomocí zvolené lineární či nelineární transformace, čímž je vytvořen odhad slovního vektoru překladu $\hat{w}^y$ v cílovém sémantickém prostoru $\mathcal{S}^y$. Následně je v cílovém sémantickém prostoru nalezeno n nejbližších slov k vytvořenému překladu. V případě, že výsledný seznam nejbližších slov obsahuje správný výraz w^y, je překlad zaznamenán jako správný.

Pokud zvolíme hodnotu $n = 1$, zajímá nás pouze, zda odhadovaný slovní vektor odpovídá nejbližšímu nalezenému slovu v cílovém prostoru. Avšak tento přístup nebere v úvahu fakt, že první nalezené slovo může být pouze synonymem hledaného výrazu. Z tohoto důvodu je žádoucí provádět také experimenty, během kterých je zvoleno $n > 1$, například $n = 5$, kdy je výraz hledán mezi pěti nejbližšími vektory cílového prostoru. Použití strojového

překladu pro testování kvality mezi-jazyčné lineární transformace sémantického prostoru je detailně popsáno například v článku *Exploiting Similarities among Languages for Machine Translation* ([18]).

6.4 Hubness

Jak je uvedeno v článku [24], závažným problémem, který postihuje datové prostory s mnoha dimenzemi, je takzvaná *hubness*. Pokud budeme uvažovat, že $D \subset \mathbb{R}^d$ je množina bodů z d-dimenzionálního prostoru, můžeme definovat hodnotu $N_k(\mathbf{w})$ jako počet, kolikrát se zvolené slovo $\mathbf{w}$ vyskytuje mezi k nejbližšími sousedy všech ostatních bodů v množině D dle předem zvolené metriky vzdálenosti. Pojem *hub* poté v našem případě reprezentuje slova, která se nejčastěji vyskytují v okolí ostatních.

Radovanović ve své práci prokázal, že distribuce náhodné veličiny N_k je se zvyšujícím se počtem dimenzí výrazně nesymetrická [24]. Z tohoto důvodu je pro jednotlivé sémantické prostory spočten koeficient šikmosti (anglicky *skewness*), respektive míra asymetrie před transformací a po jejím provedení pro všechny použité jazykové páry. Koeficient šikmosti náhodné veličiny N_k můžeme spočítat pomocí vzorce uvedeného v rovnici 6.3.

$$S_{N_k} = \frac{E(N_k - \mu_{N_k})^3}{\sigma_{N_k}^3}, \tag{6.3}$$

kde symboly μ a σ jsou střední hodnota náhodné veličiny, respektive její směrodatná odchylka. Čím vyšší je hodnota koeficientu šikmosti, tím vyšší je poté *hubness* výsledného sémantického prostoru.

Na první pohled se lze domnívat, že tento fenomén nedosahuje stejné důležitosti jako tři předchozí sledované kategorie. Avšak jak bude možné dále vidět v provedených experimentech v kapitole 7, určité transformace způsobují ve výsledném sémantickém prostoru výrazně vyšší *hubness* než jiné. V článku *Linear Transformations for Cross-lingual Semantic Textual Similarity* [4] autor poukazuje na klíčový fakt, že problém *hubness* mezi různými sémantickými prostory nabývá ještě větší důležitosti, neboť významným způsobem ovlivňuje kvalitu výsledné transformace.

Na rozdíl od předchozích sledovaných vlastností transformace byla při výpočtu *hubness* pro určení vzdálenosti slovních vektorů využita *euklidovská* metrika, jak bylo doporučeno v článku [4]. Tato metrika bude blíže představena v kapitole 6.5.1.

6.5 Metriky

Obsahem této sekce bude představení metrik použitých při tvorbě diplomové práce během evaluace experimentů uvedených výše.

6.5.1 Euklidovská vzdálenost

Euklidovskou vzdálenost d mezi dvěma body $\mathbf{p}$ a $\mathbf{q}$ v n-rozměrném euklidovském prostoru o souřadnicích $\mathbf{p} = (p_1, p_2, \ldots, p_n)$ a $\mathbf{q} = (q_1, q_2, \ldots, q_n)$ můžeme určit pomocí následující rovnice:

$$\begin{aligned} d(\mathbf{p}, \mathbf{q}) &= (p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2 \\ &= \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2} \end{aligned} \tag{6.4}$$

6.5.2 Kosinová podobnost

Druhou využitou metrikou pro vypočtení podobnosti mezi dvěma vektory byla **kosinová podobnost**. Oproti výše uvedené euklidovské vzdálenosti kosinová podobnost nepočítá rozdíl mezi jednotlivými složkami vektoru, ale určuje úhel, který dané vektory svírají. Čím menší úhel dva vektory svírají, tím jsou podobnější. Kosinovou podobnost mezi dvěma vektory A a B můžeme vypočíst podle vzorce 6.5

$$\textit{podobnost} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \, \|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}, \tag{6.5}$$

kde A_i a B_i reprezentují jednotlivé složky vektoru A, respektive B.

6.5.3 Pearsonův korelační koeficient

Pearsonův korelační koeficient ρ pro danou dvojici náhodných veličin (X, Y) můžeme spočítat pomocí následující rovnice:

$$\rho_{X,Y} = \frac{\operatorname{cov}(X, Y)}{\sigma_X \sigma_Y}, \tag{6.6}$$

kde výraz cov označuje kovarianci daných náhodných veličin, σ_X představuje směrodatnou odchylku náhodné veličiny X a σ_Y směrodatnou odchylku Y.

6.5.4 Spearmanův korelační koeficient

Spearmanův korelační koeficient slouží pro vypočtení statistické závislosti mezi dvěma veličinami. Jedná se o neparametrickou metodu, která využívá pořadí hodnot sledovaných veličin. Spearmanův korelační koeficient mezi dvěma náhodnými veličinami X a Y lze definovat jako:

$$r_s = \rho_{\mathrm{rg}_X, \mathrm{rg}_Y} = \frac{\mathrm{cov}(\mathrm{rg}_X, \mathrm{rg}_Y)}{\sigma_{\mathrm{rg}_X} \sigma_{\mathrm{rg}_Y}}, \tag{6.7}$$

kde ρ představuje Pearsonův korelační koeficient, $\mathrm{cov}(\mathrm{rg}_X, \mathrm{rg}_Y)$ označuje kovarianci ohodnocených veličin a σ_{rg_X}, respektive σ_{rg_Y} směrodatnou odchylku ohodnocených veličin. V případě, že je všech n ohodnocení určeno pomocí unikátních celočíselných hodnot, lze pro vypočtení Spearmanova korelačního koeficientu využít zjednodušený vzorec

$$r_s = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}, \tag{6.8}$$

kde $d_i = \mathrm{rg}(X_i) - \mathrm{rg}(Y_i)$ je rozdíl mezi dvěma pořadími v každém pozorování a n je celkový počet pozorování.

7 Experimenty

Obsahem této kapitoly jsou všechny testy, které byly provedeny v průběhu tvorby diplomové práce. V následující sekci si podrobně představíme výsledky naměřené pro jednotlivé lineární transformace uvedené v kapitole 3. Lineární transformace jsou v současné době velmi hojně využívány a současně také tvoří velmi solidní základ pro porovnání s navrženými metodami. Výsledky pro lineární transformace se tedy staly určitým výchozím bodem a cílem nově navržených metod bylo pokusit se tyto výsledky překonat a dále vylepšit. I přes jednoduchost metod využívajících lineární transformace, zejména metody nejmenších čtverců, která byla podrobněji představena v kapitole 3.1, se ukázal tento problém jako poměrně netriviální.

Veškeré tabulky umístěné v této kapitole, které uvádějí výsledky experimentů, používají v záhlaví následující výrazy: **SemEval** - odkazuje na slovní podobnost představenou v sekci 6.1, **Analogie** - v tomto případě se jedná o výsledky v kategorii slovních analogií, které jsou detailněji popsané v sekci 6.2, **MT@1** - tento sloupec obsahuje výsledky pro strojový překlad uvedený v sekci 6.3, **MT@5** - stejně jako v předchozím případě sloupec obsahuje výsledky dosažené při strojovém překladu, avšak v tomto případě je zkoumáno pět nejbližších nalezených slov, a **Skewness** - hodnoty v tomto sloupci uvádí koeficient šikmosti, který udává míru *hubness*, jak je definováno v sekci 6.4. Poslední řádek každé tabulky pro přehlednost uvádí průměr daných kategorií.

7.1 Trénovací data

Všechny experimenty mezi-jazyčných transformací sémantických prostorů uskutečněné v rámci diplomové práce byly provedeny s celkem **šesti** rozdílnými jazyky. Konkrétně se jednalo o *angličtinu*, *němčinu*, *italštinu*, *španělštinu*, *češtinu* a *chorvatštinu*. Jedná se o zástupce tří rozdílných větví indoevropských jazyků. Přesněji řečeno, jazyky angličtina a němčina se řadí mezi germánské jazyky, italština a španělština oproti tomu náleží mezi románské jazyky, které vznikly z latiny. Poslední skupinu tvoří čeština a chorvatština, které se řadí mezi jazyky slovanské. Slovanské jazyky lze dále dělit na západoslovanské, mezi které náleží také čeština, východoslovanské (například ruština) a jihoslovanské jako je chorvatština. Všechny slovanské jazyky vycházejí z *praslovanštiny* [14]. Na první pohled je tedy patrné, že se nejen z lingvistického hlediska jedná o velmi odlišné jazyky. Tento rozdíl se

dále prohloubil především u použitých slovanských jazyků, které jsou oproti ostatním použitým jazykům mnohem méně rozšířené. Tento fakt do jisté míry také ovlivňuje kvalitu dostupných slovních vektorů.

Pro všechny uvedené jazyky byly využity slovní vektory předtrénované s použitím modelu fastText[1] a metody skip-gram, které byly natrénovány na mnohojazyčné on-line encyklopedii Wikipedia[2]. Výjimku tvořil pouze jazyk chorvatština. Chorvatská Wikipedie v porovnání s ostatními obsahuje mnohem méně dat, což ve výsledku zapříčinilo výrazné snížení kvality výsledného sémantického prostoru. Z tohoto důvodu jsem se posléze rozhodl natrénovat slovní vektory pro tento jazyk na větším množství dat. Vstupní data pro trénink modelu *fastText* jsem vytvořil sloučením nejnovější dostupné verze chorvatské Wikipedie a rozsáhlého korpusu hrWaC[3], jehož data byla shromážděna z chorvatských webových stránek a který celkem obsahuje téměř 2 miliardy tokenů. Normalizace vstupních dat a nastavení parametrů použitého modelu *fastText* bylo pro zachování porovnatelnosti experimentů provedeno dle článku [3].

Tabulka 7.1: Statistika využitých slovních vektorů

Jazyk	**Velikost (MB)**	**Unikátních slov**
angličtina	6 292	2 519 371
němčina	5 690	2 275 233
španělština	2 475	985 667
italština	2 183	871 053
čeština	1 571	627 841
chorvatština	1 124	451 637
*chorvatština**	21 879	8 817 416

Tabulka 7.1 udává základní statistiky použitých slovních vektorů. Jazyk chorvatština označený hvězdičkou byl natrénován na výše uvedené kombinaci Wikipedie a korpusu hrWaC. Díky použití tohoto rozsáhlého korpusu došlo k více než 20-ti násobnému navýšení velikosti výsledného souboru i počtu unikátních slov.

Pro všechny uvedené experimenty bylo následně zvoleno **100 000** nejčastějších slov z každého jazyka. Každé slovo bylo v sémantickém prostoru reprezentováno slovním vektorem o dimenzi $d = \mathbf{300}$. Všechny vektory byly

[1]Předtrénované vektory jsou dostupné na adrese `https://fasttext.cc/docs/en/pretrained-vectors.html`

[2]Encyklopedie je dostupná na adrese `https://www.wikipedia.org/`.

[3]Korpus je dostupný na adrese `http://nlp.ffzg.hr/resources/corpora/hrwac/`.

po načtení normalizovány, tedy každý vektor byl vydělen svojí délkou, čímž byl získán jednotkový vektor o délce právě jedna. Velikost použitého slovníku pro vytvoření transformace byla dle článku [4] stanovena na **20 000** slov.

7.2 Testovací data

Tabulka 7.2 udává počet slovních párů pro jednotlivé jazyky dvojice, které jsou součástí datového setu *SemEval2017-Task2*[4] [6]. Tento datový set byl použit pro testování slovních podobností z kapitoly 6.1. Uvedený datový

Tabulka 7.2: Přehled datové sady *SemEval2017-Task2*

Transformace	**Počet párů**
de – es	956
de – it	912
en – de	914
en – es	978
en – it	970
es – it	967
en – cs	–
en – hr	–

set však neobsahuje data pro češtinu ani chorvatštinu, a z tohoto důvodu nejsou výsledky pro slovní podobnost uvedeny u dvojic obsahujících tyto jazyky. Počty použitých párů pro jednotlivé jazyky jsou poměrně vyrovnané a pro každý z nich jich existuje více než **900**.

Tabulka 7.3 uvádí počty dvojic slovních analogií v různých kategoriích pro všechny využité jazyky. Stejně jako v kapitole 6.2 jsou v této tabulce použité kategorie pro větší přehlednost rozděleny na sémantické a syntaktické. Počty párů se pro různé kategorie i jazyky značně liší, avšak na rozdíl od předchozí kategorie jsou dostupné pro všechny využité jazykové páry. Pro testování jednotlivých druhů analogií jsou použity všechny kombinace dvojic mezi zvolenými jazyky x a y. Například pro kategorii *stát – měna* při transformaci z angličtiny do němčiny máme celkem $29 \times 28 = 812$ testovaných párů. Výsledná přesnost pro slovní analogie je následně vypočtena jako průměr přesností jednotlivých kategorií.

[4]Podrobnosti k tomuto datasetu jsou dostupné na adrese `https://alt.qcri.org/semeval2017/task2/`.

Tabulka 7.3: Přehled datové sady slovních analogií

Analogie	**en**	**de**	**es**	**it**	**cs**	**hr**
rodina	24	24	20	20	26	41
stát – měna	29	29	28	29	29	21
hlavní město – stát	23	23	21	23	23	23
stát – přídavné jméno	41	41	40	41	41	41
pozitiv – komparativ	23	37	5	10	40	77
pozitiv – superlativ	20	34	40	29	40	77
antonyma	29	29	20	24	27	29
singulár – plurál	112	111	37	36	74	46
prézens – préteritum	38	40	39	33	95	40

Strojový překlad představený v kapitole 6.3 byl testován na předem určené podmnožině sémantického prostoru obsahující přesně **10 000** slov. Jednalo se o slova, která nebyla součástí trénovací množiny. Pro každé slovo x_i byl však odpovídající překlad y_i umístěn ve slovníku, a proto bylo možné správnost nalezeného překladu jednoduše ověřit. Tento test byl rovněž proveden na všech jazykových párech.

Z implementačního hlediska se výpočet hodnoty *hubness* (viz kapitola 6.4) posléze ukázal jako výpočetně nejnáročnější část programu, jelikož je zapotřebí spočítat vzdálenost mezi všemi páry slovních vektorů. Z tohoto důvodu je časová složitost výpočtu $\mathcal{O}(n^2)$, kde n představuje počet vektorů v sémantickém prostoru. Hodnota n se v experimentech pohybovala v řádu 10^5, také z tohoto důvodu byly následně všechny testy spouštěny v prostředí *MetaCentra*[5].

7.3 Lineární transformace

V této kapitole si postupně představíme výsledky pro tři metody lineární transformace testované v diplomové práci. Konkrétně se jedná o metodu nejmenších čtverců (kapitola 3.1), ortogonální transformaci (kapitola 3.2) a transformaci s použitím kanonické korelace (kapitola 3.3).

[5]Jedná se o platformu provozující a spravující distribuovanou výpočetní infrastrukturu. Prostředí MetaCetra lze využít pro úlohy, které vyžadují vysoký výpočetní výkon. Bližší informace jsou dostupné na oficiálních stránkách `https://metavo.metacentrum.cz/`.

7.3.1 Metoda nejmenších čtverců

Tabulka 7.4 uvádí naměřené výsledky pro transformaci s použitím metody nejmenších čtverců, která byla detailně popsána v sekci 3.1. Můžeme si povšimnout, že nejvyšších hodnot bylo dosaženo pro jazykové páry *angličtina - španělština* a *angličtina - italština*. Především první zmiňovaný jazykový

Tabulka 7.4: Naměřené výsledky - *metoda nejmenších čtverců*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.621	0.404	0.301	0.458	13.5
de – it	0.613	0.399	0.287	0.446	12.1
en – de	0.646	0.313	0.533	0.692	4.9
en – es	0.647	0.421	0.680	0.836	5.1
en – it	0.642	0.427	0.491	0.665	4.6
es – it	0.659	0.434	0.434	0.608	6.6
en – cs	–	0.328	0.278	0.469	4.1
en – hr	–	0.246	0.285	0.442	4.5
Průměr	**0.638**	**0.372**	**0.411**	**0.577**	**6.9**

pár dosáhl velmi vysokých hodnot pro strojový překlad. Na druhé straně výsledky pro transformace z anglického jazyka do češtiny a chorvatštiny dosáhly v porovnání s ostatními jazykovými páry mnohem nižších hodnot. Toto snížení však není způsobeno samotnou transformací, ale velmi pravděpodobně bohatší a komplexnější morfologií slovanských jazyků oproti germánským či románským jazykům.

Nevýhodou této metody jsou především její značně vysoké hodnoty *skewness*, kterých si můžeme povšimnout především u prvních dvou jazykových párů. Tato metoda tedy během transformace určitým způsobem deformuje a pokřivuje transformovaný sémantický prostor, čímž znemožňuje dosažení lepších výsledků. Přesto tato metoda velmi dobře posloužila jako základ, se kterým bylo možné porovnat nejen ostatní dostupné metody pro lineární transformaci, ale také nově navržené metody.

7.3.2 Ortogonální transformace

V tabulce 7.5 lze vidět výsledky pro ortogonální transformaci, která byla představena v sekci 3.2. Při porovnání s předchozí transformací jsou patrné především výrazně nižší hodnoty pro *skewness*. Tento významný fakt je způsobem tím, že ortogonální transformace nedeformuje transformovaný

sémantický prostor, jelikož zachovává úhly mezi jednotlivými body v daném prostoru. Díky zmíněné vlastnosti dosahovala ortogonální transformace v této kategorii po dlouhou dobu nejlepších výsledků.

Tabulka 7.5: Naměřené výsledky - *ortogonální transformace*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.608	0.423	0.318	0.482	5.6
de – it	0.595	0.425	0.327	0.479	4.1
en – de	0.630	0.352	0.582	0.727	2.8
en – es	0.659	0.432	0.732	0.876	3.1
en – it	0.649	0.451	0.553	0.709	2.7
es – it	0.652	0.493	0.435	0.596	3.5
en – cs	–	0.414	0.335	0.529	3.3
en – hr	–	0.352	0.289	0.456	5.2
Průměr	**0.632**	**0.418**	**0.446**	**0.607**	**3.8**

Oproti metodě nejmenších čtverců dosáhla ortogonální transformace, vyjma nižších hodnot pro *skewness*, znatelně vyšších hodnot pro testy analogií a strojového překladu. Avšak na druhé straně došlo k jistému snížení výsledků v případě slovních podobností, především u prvních tří jazykových párů. V průměru se však jednalo pouze o rozdíl lehce přesahující půl procenta, ze statistického hlediska se jedná o zanedbatelný rozdíl.

7.3.3 Transformace pomocí kanonické korelace

Třetí a poslední lineární transformací, jejíž výsledky zde budou představeny, je transformace pomocí kanonické korelace. Naměřené hodnoty pro tuto transformaci můžeme vidět v tabulce 7.6. Při porovnání této transformace s metodou nejmenších čtverců lze vidět, že si podobně jako ortogonální transformace vedla znatelně lépe ve všech sledovaných kategoriích s výjimkou slovních podobností, kde došlo k snížení přibližně o jedno procento.

Jak lze vidět, transformace pomocí kanonické korelace dosáhla oproti ortogonální transformaci vyšších hodnot ve sledovaných kategoriích analogie a strojového překladu. Ortogonální transformace však naopak dosahovala lepších hodnot pro *skewness*. Díky transformaci pomocí kanonické korelace je tedy možné vylepšit výsledky pro zmíněné kategorie, avšak za cenu zvýšené implementační a výpočetní složitosti společně s vyšší deformací výsledného transformovaného prostoru.

Tabulka 7.6: Naměřené výsledky - *transformace pomocí kanonické korelace*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.604	0.438	0.341	0.508	5.3
de – it	0.589	0.457	0.340	0.503	4.1
en – de	0.636	0.378	0.587	0.734	2.6
en – es	0.652	0.429	0.748	0.882	6.1
en – it	0.644	0.469	0.555	0.716	4.1
es – it	0.649	0.499	0.458	0.637	7.8
en – cs	–	0.425	0.339	0.539	3.4
en – hr	–	0.389	0.288	0.467	2.9
Průměr	**0.629**	**0.436**	**0.457**	**0.623**	**4.5**

V následující kapitole se zaměříme na výsledky transformací, které využívají neuronové sítě, tedy takzvané nelineární transformace sémantických prostorů.

7.4 Nelineární transformace

V této sekci si představíme naměřené výsledky pro nelineární transformace. Konkrétně jsou výsledky rozděleny do tří sekcí pro modely bez skryté vrstvy, se skrytou vrstvou a modely využívající *hinge-loss*.

7.4.1 Neuronové sítě bez skryté vrstvy

Výsledky pro první navrženou neuronovou síť v rámci diplomové práce můžeme vidět v tabulce 7.7. Struktura této neuronové sítě byla poměrně jednoduchá a samotná síť neobsahovala žádnou skrytou vrstvu. Jako aktivační funkce byl v tomto případě využit hyperbolický tangens a pro cenovou funkci byla zvolena střední kvadratická chyba.

Na první pohled je patrné, že výsledky této neuronové sítě jsou velmi podobné výsledkům pro metodu nejmenších čtverců uvedeným v tabulce 7.4. Tato neuronová síť však oproti metodě nejmenších čtverců dosahuje ve všech sledovaných kategoriích přibližně o jedno procento horších výsledků. Samotné přidání nelineárního prvku (v podobě aktivační funkce) do procesu transformace sémantických prostorů v tomto případě nevedlo k překonání výsledků lineárních transformací uvedených v předchozí kapitole. Zmíněné zhoršení výsledků je patrně způsobeno faktem, že je řešení v případě neuronové sítě pouze odhadováno pomocí gradientního sestupu na rozdíl od

Tabulka 7.7: Naměřené výsledky - *neuronová síť bez skryté vrstvy 1*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.617	0.398	0.289	0.448	13.7
de – it	0.611	0.389	0.277	0.434	13.4
en – de	0.644	0.303	0.519	0.683	4.5
en – es	0.646	0.411	0.674	0.833	5.3
en – it	0.638	0.421	0.472	0.651	4.1
es – it	0.661	0.419	0.429	0.603	6.7
en – cs	–	0.319	0.268	0.455	3.9
en – hr	–	0.243	0.272	0.432	8.0
Průměr	**0.636**	**0.363**	**0.400**	**0.567**	**7.5**

metody nejmenších čtverců, kde existuje přesné analytické řešení (viz rovnice 3.3). Vezmeme-li v potaz ještě přidání nelineárního prvku, je velmi pravděpodobné, že nebylo nalezeno dobré řešení.

Tabulka 7.8: Naměřené výsledky - *neuronová síť bez skryté vrstvy 2*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.617	0.449	0.308	0.462	15.0
de – it	0.610	0.456	0.294	0.467	13.9
en – de	0.644	0.393	0.543	0.696	6.2
en – es	0.645	0.450	0.681	0.836	5.9
en – it	0.638	0.493	0.498	0.667	5.6
es – it	0.659	0.506	0.436	0.604	7.9
en – cs	–	0.447	0.281	0.447	6.3
en – hr	–	0.437	0.262	0.425	8.2
Průměr	**0.636**	**0.454**	**0.413**	**0.576**	**8.6**

Výsledky pro druhý navržený model lze vidět v tabulce 7.8. Jednalo se o velmi podobný návrh jako v předchozím případě, kdy byla použita aktivační funkce hyperbolický tangens. Rozdílem však bylo použití cenové funkce kosinové vzdálenosti namísto střední kvadratické chyby. Při porovnání s předchozím modelem je na první pohled patrné značné vylepšení výsledků, především pro kategorii analogie. V této kategorii došlo oproti předchozímu modelu k velmi značnému nárůstu o téměř 10 procent. Tento výsledek je dokonce ještě o více než tři procenta lepší než dosavadní nejvyšší

výsledek **42.9** % v případě transformace pomocí kanonické korelace.

V ostatních kategoriích jsou však výsledky téměř totožné s výsledky pro metodu nejmenších čtverců. K výraznému zlepšení tedy došlo pouze u jedné ze sledovaných kategorií. Současně však také nastalo určité zvýšení *skewness* výsledného sémantického prostoru, který byl ještě více zdeformován oproti předchozímu modelu. Tento model by tedy ve výsledku mohl být použitelný jako alternativa metody nejmenších čtverců, kdy je prioritou především vysoká přesnost v kategorii analogií.

V následující sekci si představíme výsledky pro další dva modely neuronových sítí navržených v rámci diplomové práce, které oproti výše zmíněným modelům obsahovaly skrytou vrstvu.

7.4.2 Neuronové sítě se skrytou vrstvou

V této sekci budou představeny výsledky pro neuronové sítě obsahující skrytou vrstvu. Klíčovou otázkou pro tento model byla volba aktivačních funkcí pro jednotlivé vrstvy neuronové sítě. Celkem byly navrženy tři odlišné přístupy k jejich výběru.

1. Pro první případ byla použita aktivační funkce **tanh** z kapitoly 4.1.2, kterou pro aktivaci neuronů využívala skrytá vrstva i výstupní vrstva neuronové sítě.

2. V druhém případě byla využita kombinace aktivačních funkcí. Pro skrytou vrstvu byla použita aktivační funkce **ReLU** z kapitoly 4.1.2, která je pro tento typ vrstvy vhodná a může vést k urychlení tréninku neuronové sítě. Pro výstupní vrstvu byla ponechána aktivační funkce **tanh**. Jako cenová funkce byla v tomto případě zvolena kosinová podobnost.

3. Třetí a poslední případ opět využíval pro skrytou vrstvu aktivační funkci **ReLU**, avšak výstupní vrstva byla ponechána bez aktivace. Výstup sítě v tomto případě není omezen určitým intervalem, z tohoto důvodu je tento přístup často používán pro řešení regresních problémů.

Výsledky pro první výše uvedený model neuronové sítě se skrytou vrstvou lze vidět v tabulce 7.9. Tato neuronová síť obsahovala jednu skrytou vrstvu s aktivační funkcí hyperbolický tangens, který byl současně aktivační funkcí vrstvy výstupní. Jako cenová funkce zde byla zvolena střední kvadratická chyba stejně jako v případě prvního představeného modelu v předchozí sekci.

Při porovnání s výsledky z tabulky 7.7, která obsahuje obdobný model neuronové sítě pouze bez skryté vrstvy, je patrné, že došlo ve všech sledo-

Tabulka 7.9: Naměřené výsledky - *neuronová síť se skrytou vrstvou 1*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.614	0.399	0.277	0.430	12.6
de – it	0.601	0.385	0.259	0.417	13.8
en – de	0.648	0.300	0.518	0.683	5.1
en – es	0.643	0.406	0.647	0.815	5.5
en – it	0.639	0.416	0.468	0.651	4.5
es – it	0.659	0.418	0.417	0.597	7.1
en – cs	–	0.313	0.259	0.453	6.7
en – hr	–	0.237	0.267	0.429	7.9
Průměr	**0.634**	**0.359**	**0.389**	**0.559**	**7.9**

vaných kategoriích k drobnému zhoršení výsledků. Současně se také zvýšila *skewness* výsledného sémantického prostoru. Z naměřených výsledků lze dále usoudit, že pouhé zvýšení komplexity modelu neuronové sítě nevede ke zlepšení výsledků, ale spíše pouze ke ztížení a zpomalení procesu trénování společně se zvýšenou deformací výsledného prostoru.

Výsledky pro druhý model lze vidět v tabulce 7.10. Tento model využíval kombinaci aktivačních funkcí ReLU a hyperbolický tangens. První z uvedených funkcí byla použita jako aktivační funkce skryté vrstvy. Hyperbolický tangens v tomto případě posloužil pouze jako aktivace výstupní vrstvy. Dále byla pro cenovou funkci použita kosinová vzdálenost, která v modelu bez skryté vrstvy pomohla značně vylepšit výsledky v kategorii analogií.

Tabulka 7.10: Naměřené výsledky - *neuronová síť se skrytou vrstvou 2*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.616	0.450	0.284	0.439	15.5
de – it	0.609	0.476	0.281	0.435	14.9
en – de	0.645	0.405	0.520	0.680	6.4
en – es	0.643	0.448	0.621	0.797	6.2
en – it	0.643	0.487	0.474	0.641	5.1
es – it	0.658	0.500	0.425	0.595	7.2
en – cs	–	0.441	0.261	0.453	6.4
en – hr	–	0.446	0.264	0.427	8.3
Průměr	**0.636**	**0.457**	**0.391**	**0.558**	**8.8**

Podobně jako neuronová síť bez skryté vrstvy s cenovou funkcí kosinová podobnost, jejíž výsledky jsou uvedeny v tabulce 7.8, dosahuje i tento model velmi vysokých hodnot v kategorii slovních analogií, konkrétně **45.7** %. Model bez skryté vrstvy však dosahuje výrazně lepších hodnot pro strojový překlad. Současně jsou zřejmé velmi vysoké hodnoty pro *skewness*. Tato transformace tedy významným způsobem snižuje kvalitu výsledného transformovaného sémantického prostoru.

V případě třetího modelu neuronové sítě se skrytou vrstvou, jehož výsledky můžeme vidět v tabulce 7.11, byl použit odlišný návrh než v případě předchozích modelů. Jednalo se především o ponechání výstupní vrstvy bez aktivace aktivační Pro skrytou vrstvu byla zvolena aktivační funkce ReLU. Jako cenová funkce byla stejně jako v předchozím případě zvolena kosinová podobnost.

Tabulka 7.11: Naměřené výsledky - *neuronová síť se skrytou vrstvou 3*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.621	0.318	0.226	0.381	13.9
de – it	0.612	0.315	0.259	0.431	14.0
en – de	0.647	0.321	0.528	0.690	7.4
en – es	0.643	0.374	0.613	0.799	5.5
en – it	0.640	0.373	0.479	0.646	5.5
es – it	0.661	0.372	0.349	0.525	8.9
en – cs	–	0.333	0.277	0.463	5.8
en – hr	–	0.437	0.251	0.415	7.7
Průměr	**0.637**	**0.355**	**0.373**	**0.544**	**8.6**

Z tabulky je však na první pohled patrné, že se nejedná o kvalitní metodu a její výsledky jsou mezi neuronovými sítěmi nejhorší. Jedinou kategorií, která dosahovala porovnatelných výsledků s předchozími modely, byla pouze kategorie slovních podobností.

V následující sekci si představíme poslední skupinu neuronových sítí testovaných v diplomové práci, které používaly speciální cenovou funkci založenou na metodě *hinge-loss*.

7.4.3 Neuronové sítě s cenovou funkcí *hinge-loss*

Obsahem této sekce jsou výsledky pro navržené neuronové sítě, které jsou založené na metodě *hinge-loss*. Výsledky pro první experiment s touto struk-

turou lze vidět v tabulce 7.12. Daný model v tomto případě využívá pro vypočtení cenové funkce pouze **jeden** negativní příklad.

Tabulka 7.12: Naměřené výsledky - *neuronová síť s hinge-loss - 1 negativní příklad*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.603	0.442	0.349	0.516	4.7
de – it	0.597	0.452	0.348	0.510	4.5
en – de	0.638	0.405	0.595	0.740	2.3
en – es	0.666	0.439	0.749	0.887	2.5
en – it	0.662	0.470	0.569	0.726	1.8
es – it	0.658	0.473	0.475	0.643	3.3
en – cs	–	0.438	0.354	0.548	3.0
en – hr	–	0.347	0.217	0.374	3.2
Průměr	**0.637**	**0.433**	**0.457**	**0.618**	**3.2**

Na první pohled je patrná velmi nízká hodnota **3.2** pro *skewness*. Tato hodnota je dokonce nižší než v případě ortogonální transformace, kde je rovna **3.6**. Jedná se o velmi podstatný výsledek a současně praktické ověření myšlenky z článku [15], že lze tuto metodu využít pro snížení *hubness* výsledného transformovaného sémantického prostoru, a to dokonce v kombinaci s neuronovou sítí.

Při porovnání s nejlepší lineární transformací, respektive transformací pomocí kanonické korelace uvedené v tabulce 7.6, lze vidět, že i v ostatních sledovaných kategoriích si tato transformace vede velmi dobře. Kromě výsledků pro slovní podobnost, kde obě transformace dosáhly identických výsledků, je transformace pomocí neuronové sítě využívající metodu *hinge-loss* lepší ve všech kategoriích.

Tato transformace tedy jako první zde uvedená kombinuje nejlepší vlastnosti ortogonální transformace a transformace pomocí kanonické korelace, respektive velmi nízké hodnoty *skewness* a vysoké hodnoty pro ostatní sledované kategorie. Obě zmíněné transformace svými výsledky dokonce překonává. Dle všech sledovaných měřítek se tedy jedná o velmi kvalitní transformaci překonávající běžné lineární transformace. Vzhledem k tomuto faktu jsem se v následujících experimentech pokusil vylepšit výsledky této metody pomocí rozličných úprav jak v počtu negativních příkladů, tak i parametrů neuronové sítě.

Tabulka 7.13 udává výsledky pro neuronovou síť jako v předchozím případě s tím rozdílem, že tato síť využívá pro vypočtení cenové funkce **dva**

Tabulka 7.13: Naměřené výsledky - *neuronová síť s hinge-loss - 2 negativní příklady*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.604	0.436	0.352	0.514	4.3
de – it	0.598	0.446	0.352	0.514	4.4
en – de	0.644	0.395	0.600	0.741	2.3
en – es	0.662	0.440	0.754	0.886	2.4
en – it	0.658	0.474	0.572	0.729	1,8
es – it	0.659	0.469	0.476	0.643	2,9
en – cs	–	0.440	0.363	0.555	3,1
en – hr	–	0.355	0.214	0.380	3,3
Průměr	**0.638**	**0.432**	**0.460**	**0.620**	**3.1**

negativní příklady. Cílem tohoto experimentu bylo zjistit, zda postupné navyšování počtu negativních příkladů povede ke změně naměření výsledků, ať již ke zvýšení či snížení. Srovnáme-li však výsledky pro dva negativní příklady s výsledky předchozími, zjistíme, že nedochází k výrazné změně. Všechny sledované kategorie se liší pouze o desetiny procenta. Dané odchylky jsou velmi malé a může se tak například jednat o rozdíl způsobený odlišnou náhodnou inicializací vah během tréninku neuronových sítí.

Tabulka 7.14: Naměřené výsledky - *neuronová síť s hinge-loss - 5 negativních příkladů*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.608	0.427	0.350	0.515	4.1
de – it	0.591	0.437	0.351	0.512	3.9
en – de	0.645	0.392	0.597	0.739	2,3
en – es	0.666	0.429	0.750	0.886	2,4
en – it	0.652	0.466	0.570	0.725	1,7
es – it	0.662	0.467	0.474	0.643	2,9
en – cs	–	0.432	0.357	0.548	3.0
en – hr	–	0.363	0.219	0.378	3.0
Průměr	**0.637**	**0.427**	**0.459**	**0.618**	**2.9**

Na druhé straně však došlo k dalšímu snížení hodnoty *skewness* navzdory zvýšené komplexnosti cenové funkce přidáním druhého negativního příkladu.

Tento fakt tedy může naznačovat, že přidání dalších negativních příkladů by mohlo ještě více napomoci řešení problému *hubness*.

Z tohoto důvodu jsem dále pokračoval ve zvyšování počtu negativních příkladů za účelem ověření výše uvedené teorie. Výsledky pro neuronovou síť, která využívá celkem **pět** negativních příkladů v cenové funkci, můžeme vidět v tabulce 7.14. V tomto případě došlo k drobnému poklesu naměřených výsledků, ale opět se jednalo pouze o desetiny procenta. Současně došlo k dalšímu již znatelnějšímu poklesu hodnot pro *skewness*. Nelze zcela vyloučit, že se jedná pouze o statistickou odchylku, avšak při využití více negativních příkladů, od kterých má mít zvolené transformované slovo předem danou vzdálenost, lze patrně očekávat rovnoměrnější distribuci slov v prostoru. Tím je současně zamezeno nadměrné asymetrii distribuce nejbližších sousedů slov.

Ve výsledku tento model oproti předchozím dvěma zmíněným v této sekci nepřináší kromě snížení *skewness* žádné zlepšení. Nelze však opomenout, že s rostoucím počtem negativních příkladů úměrně roste také výpočetní náročnost metody gradientního sestupu, čímž se samozřejmě zpomaluje proces trénování neuronové sítě.

Tabulka 7.15: Naměřené výsledky - *neuronová síť s hinge-loss - 10 negativních příkladů*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.599	0.421	0.349	0.512	4.0
de – it	0.600	0.428	0.350	0.512	3.7
en – de	0.642	0.386	0.592	0.737	2.3
en – es	0.663	0.434	0.752	0.885	2.4
en – it	0.658	0.461	0.565	0.723	1.7
es – it	0.659	0.455	0.472	0.641	2.8
en – cs	–	0.424	0.350	0.542	2.9
en – hr	–	0.362	0.214	0.382	3.1
Průměr	**0.637**	**0.421**	**0.456**	**0.617**	**2.9**

Výsledky pro čtvrtý a zároveň poslední model, ve kterém došlo ke zvýšení počtu negativních příkladů, lze vidět v tabulce 7.15. V tomto případě bylo použito celkem **deset** negativních příkladů.

Při porovnání s výsledky pro předchozí model došlo k dalšímu nepatrnému snížení naměřených hodnot. Z uvedených hodnot lze usoudit, že pouhé navyšování počtu negativních příkladů zřejmě nevede ke zlepšení výsledků,

ale spíše ke zhoršení. Z tohoto důvodu jsem se rozhodl experimenty s vyšším počtem negativních příkladů již neprovádět. Na druhé straně však nedošlo ke zvýšení hodnoty *skewness*, dokonce u určitých jazykových párů, například *němčina-španělština* a *němčina-italština*, nastalo další snížení.

Pro následující transformaci jsem se rozhodl regularizovat váhy neuronové sítě s cílem zlepšit schopnost neuronové sítě generalizovat, a tím dále vylepšit transformaci pro data mimo trénovací množinu. Velikosti parametrů vah neuronové sítě jsou tedy v tomto případě penalizovány. Výsledky pro tuto transformaci můžeme vidět v tabulce 7.16.

Tabulka 7.16: Naměřené výsledky - *neuronová síť s hinge-loss a regularizací*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.620	0.450	0.346	0.509	5.5
de – it	0.613	0.472	0.342	0.503	5.0
en – de	0.650	0.402	0.590	0.732	2.7
en – es	0.667	0.453	0.751	0.890	2.9
en – it	0.660	0.500	0.559	0.717	2.2
es – it	0.665	0.501	0.467	0.636	3.8
en – cs	–	0.448	0.350	0.541	3.4
en – hr	–	0.409	0.303	0.469	
Průměr	**0.646**	**0.454**	**0.464**	**0.625**	**3.6**

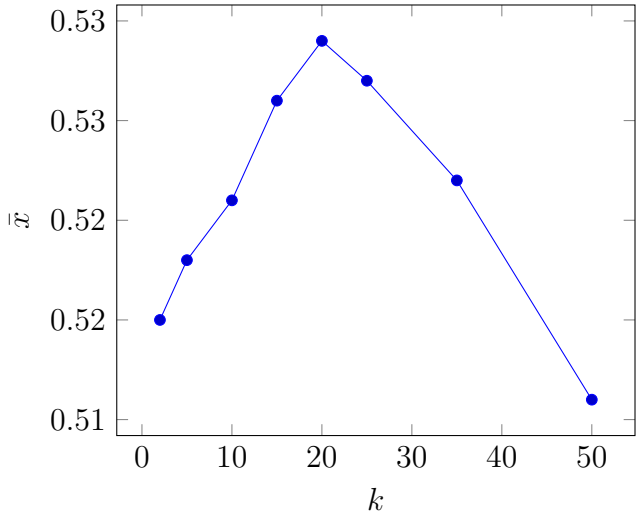
U všech sledovaných kategorií nastalo zvýšení naměřených hodnot. Ve výsledku tato transformace dosahuje velmi dobrých výsledků za cenu mírného zvýšení hodnoty *skewness*, která však nepřesahuje výsledek ortogonální transformace. Toto zvýšení bylo očekávané a nelze se mu zcela vyhnout, jelikož jej ovlivňuje nově přidaný prvek regularizace vah neuronové sítě. Regularizace vah samozřejmě jistým způsobem mění výstup sítě, čímž se může měnit i distribuce slov v transformovaném sémantickém prostoru.

V následující kapitole si představíme výsledky pro transformace, které využívaly princip shlukování.

7.5 Transformace s využitím shlukování

V této kapitole si představíme výsledky metod, které využívaly techniku shlukování pro vylepšení kvality základní transformace a detailněji byly přestaveny v kapitole 5.

7.5.1 Kombinace lineárních transformací



Obrázek 7.1: Výsledky pro rozdílné nastavení počtu shluků

Tato kapitola obsahuje výsledky metod, které pro transformaci sémantického prostoru využily techniku shlukování podrobněji popsanou v kapitole 5. V tomto případě se jednalo o lineární kombinaci základní matice a matice přiřazené k danému shluku, kdy každá z matic měla totožnou váhu rovnu jedné. Obě matice byly získány pomocí stejné, předem zvolené lineární transformace z kapitoly 3. Pro rozdělení slovních vektorů ze slovníku, respektive podmnožiny sémantického prostoru daného jazyka do jednotlivých shluků je použit algoritmus k-means představený v kapitole 5.2. Graf 7.1 vykresluje zprůměrovaný výsledek přes všechny sledované kategorie pro různá nastavení počtu shluků. Cílem bylo nalézt ideální počet shluků, který bude dosahovat nejlepších výsledků. Z grafu je patrné, že nejlepšího výsledku bylo dosaženo pro počet shluků $k = \mathbf{20}$. Tato hodnota byla také následně zvolena.

Kombinace metod nejmenších čtverců

Tabulka 7.17 uvádí výsledky pro první experiment využívající pro vylepšení vlastností transformace metodu shlukování. V tomto případě byla základní transformační matice i matice náležející shluku vytvořena pomocí metody nejmenších čtverců. Cílem bylo pokusit se vylepšit vlastnosti této lineární metody transformace sémantických prostorů. Při porovnání s výsledky pro metodu nejmenších čtverců v tabulce 7.4 je na první pohled patrné výrazné zvýšení hodnoty v kategorii analogie. Současně došlo ke snížení hodnoty *skewness* především pro první dva jazykové páry. Zároveň si však lze povšimnout také mírného snížení hodnot pro strojový překlad. Navržená metoda si

Tabulka 7.17: Naměřené výsledky - *shlukování a metoda nejmenších čtverců*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.617	0.440	0.313	0.488	11.2
de – it	0.610	0.449	0.305	0.470	10.8
en – de	0.655	0.416	0.519	0.690	4.4
en – es	0.645	0.413	0.584	0.769	4.1
en – it	0.655	0.469	0.475	0.663	4.5
es – it	0.665	0.477	0.441	0.616	5.7
en – cs	–	0.405	0.249	0.446	3.9
en – hr	–	0.415	0.257	0.425	3.4
Průměr	**0.641**	**0.436**	**0.393**	**0.571**	**6.0**

tedy celkově vedla lépe než základní lineární transformace, avšak nedošlo ke zlepšení ve všech sledovaných kategoriích.

Kombinace ortogonálních transformací

V tabulce 7.18 lze vidět výsledky pro transformaci využívající shlukování kombinované s metodou ortogonální transformace. Stejně jako v předchozím případě jsou základní matice i matice pro jednotlivé shluky vypočteny pomocí stejné lineární transformační metody, konkrétně ortogonální transformace.

Tabulka 7.18: Naměřené výsledky - *shlukování a ortogonální transformace*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.621	0.377	0.343	0.510	6.2
de – it	0.613	0.373	0.338	0.507	4.8
en – de	0.643	0.371	0.570	0.727	2.9
en – es	0.665	0.394	0.695	0.859	3.9
en – it	0.663	0.419	0.528	0.708	2.7
es – it	0.665	0.435	0.444	0.630	3.7
en – cs	–	0.396	0.315	0.524	4.4
en – hr	–	0.416	0.296	0.469	3.8
Průměr	**0.645**	**0.398**	**0.441**	**0.617**	**4.1**

Pokud porovnáme dosažené výsledky s výsledky pro ortogonální transformaci v tabulce 7.5, dochází ke zlepšení v kategorii slovních podobností.

U analogií a *skewness* dochází ke zhoršení výsledků. Ostatní kategorie jsou s drobnou odchylkou porovnatelné.

Kombinace transformací pomocí kanonické korelace

Třetí transformace využívající shlukování, jejíž výsledky jsou umístěny v tabulce 7.19, používá jako základní transformaci kanonickou korelaci.

Tabulka 7.19: Naměřené výsledky - *shlukování a kanonická korelace*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.604	0.360	0.353	0.523	5.8
de – it	0.594	0.366	0.341	0.509	4.3
en – de	0.642	0.359	0.559	0.718	6.1
en – es	0.648	0.365	0.642	0.817	8.7
en – it	0.656	0.399	0.528	0.704	2.5
es – it	0.659	0.411	0.473	0.643	5.8
en – cs	–	0.338	0.291	0.494	7.4
en – hr	–	0.384	0.278	0.440	3.7
Průměr	**0.634**	**0.373**	**0.433**	**0.606**	**5.5**

V tomto případě však nedochází ke zlepšení výsledků oproti základní lineární metodě, ale spíše ke zhoršení. Pokles hodnot je znatelný především u kategorie strojového překladu a analogií. Jedinou porovnatelnou kategorií s původní lineární transformací je slovní podobnost.

7.5.2 Vážené kombinace lineárních transformací

Tato kapitola obsahuje výsledky pro transformace, které využívaly metodu shlukování společně s nastavením koeficientů matic lineární kombinace, tak jak bylo představeno v kapitole 5.1. Cílem bylo najít takové koeficienty lineární kombinace, které maximalizují dosažené výsledky ve sledovaných kategoriích. Pro nalezení těchto koeficientů byly použity slovní vektory ze zvolené podmnožiny slovníku $\boldsymbol{D}^{x \to y}$.

Ideální koeficienty těchto transformačních matic jsem se nejdříve pokusil nalézt s využitím evolučních algoritmů. Evoluční algoritmy stanovily výchozí bod udávající velikosti obou hodnot a předpokládané rozložení vah u jednotlivých transformací. Vzhledem k poměrně úzkému rozsahu hodnot nalezených koeficientů, které se běžně pohybovaly v intervalu od 0 do 2, jsem se však posléze rozhodl nalézt tyto parametry hrubou silou. Z tohoto

důvodu byl následně předem určený interval hodnot prohledán po malých krocích dané velikosti, pohybující se nejčastěji v řádu 10^{-2}. V každém kroku byla poté vypočtena kosinová podobnost mezi transformovanými vektory $\mathbf{X}$ ze zdrojového sémantického prostoru a odpovídajícími cílovými vektory $\mathbf{Y}$. Následně byla vybrána dvojice parametrů, která dosáhla nejlepšího výsledku. Tabulka 7.20 uvádí nalezené hodnoty koeficientů pro použité druhy transformací.

Tabulka 7.20: Koeficienty transformačních matic

Metoda	a	b
Metoda nejmenších čtverců	1.50	0.47
Ortogonální transformace	1.45	0.95
Kanonická korelace	0.85	0.25

V případě vážené kombinace se v průběhu experimentů dále prokázalo, že je lepší využít větší počet shluků podobně jako v kapitole 7.5.1. Vážené kombinace s menším počtem shluků ani po nalezení nejlepších koeficientů nedosahovaly ve sledovaných kategoriích tak vysokých hodnot jako při volbě většího počtu shluků. Proto byla hodnota udávající počet shluků zvolena $k = \mathbf{20}$.

Vážené kombinace metod nejmenších čtverců

Tabulka 7.21 obsahuje výsledky pro váženou kombinaci transformačních matic, které byly vypočteny pomocí metody nejmenších čtverců. Koeficienty matic jsou uvedeny v prvním řádku tabulky 7.20.

V porovnání s ostatními metodami jsou na první pohled patrné především velmi vysoké hodnoty pro kategorii analogií. V tomto případě bylo dosaženo průměrné hodnoty téměř **46** %, což je téměř o deset procent lepší než v případě běžné lineární transformace získané metodou nejmenších čtverců. Tato hodnota je ještě o tři procenta vyšší než u nejlepší lineární transformace s použitím kanonické korelace, jedná se tedy o velmi výrazné zlepšení. Jisté zlepšení je dále patrné také pro hodnoty u kategorií slovní podobnosti a strojového překladu, nejde však o tak výrazný posun jako v případě slovních analogií.

Nevýhodou této transformace jsou poměrně vysoké hodnoty *skewness*. Tomuto faktu se však samozřejmě nelze vyhnout, jelikož se metoda nejmenších čtverců vysokou hodnotou *skewness* vyznačuje, což je patrné také při použití kombinace. Použitím metody shlukování společně s přesným nastave-

Tabulka 7.21: Naměřené výsledky - *vážená kombinace metody nejmenších čtverců*

Transformace	SemEval	Analogie	MT@1	MT@5	Skewness
de – es	0.627	0.458	0.323	0.486	13.0
de – it	0.619	0.476	0.309	0.473	12.1
en – de	0.655	0.414	0.537	0.701	5.1
en – es	0.652	0.442	0.657	0.827	4.5
en – it	0.653	0.498	0.496	0.676	4.6
es – it	0.666	0.502	0.450	0.624	6.4
en – cs	–	0.451	0.282	0.481	4.7
en – hr	–	0.433	0.289	0.455	3.1
Průměr	**0.645**	**0.459**	**0.418**	**0.590**	**7.6**

ním koeficientů transformačních matic pro zlepšení vlastností transformace lze přesto základní metodu nejmenších čtverců znatelně **vylepšit**.

Vážené kombinace ortogonálních transformací

Výsledky pro váženou kombinaci transformačních matic, které byly vypočteny pomocí metody využívající ortogonální transformaci, jsou uvedeny v tabulce 7.22. Koeficienty matic jsou uvedeny v druhém řádku tabulky 7.20. Tato transformace se vyznačuje vysokou hodnotou pro slovní podobnost.

Tabulka 7.22: Naměřené výsledky - *vážená kombinace ortogonální transformace*

Transformace	SemEval	Analogie	MT@1	MT@5	Skewness
de – es	0.623	0.331	0.340	0.504	5.9
de – it	0.614	0.319	0.339	0.506	4.6
en – de	0.643	0.321	0.577	0.730	2.9
en – es	0.666	0.352	0.713	0.867	4.0
en – it	0.662	0.369	0.539	0.712	2.8
es – it	0.665	0.376	0.440	0.625	3.5
en – cs	–	0.348	0.323	0.528	4.3
en – hr	–	0.401	0.302	0.471	4.0
Průměr	**0.646**	**0.352**	**0.447**	**0.618**	**4.0**

Hodnoty pro ostatní kategorie jsou však poměrně nízké, především pro ka-

tegorii slovních analogií. Z tohoto důvodu není využití této transformace příliš vhodné i přes žádané nízké hodnoty *skewness*.

Vážené kombinace transformace pomocí kanonické korelace

V tabulce 7.23 jsou uvedeny výsledky pro váženou kombinaci transformačních matic, které byly vypočteny pomocí metody využívající kanonickou korelaci. Koeficienty matic jsou uvedeny v třetím řádku tabulky 7.20. Tato

Tabulka 7.23: Naměřené výsledky - *vážená kombinace transformací pomocí kanonické korelace*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.615	0.442	0.365	0.527	5.9
de – it	0.600	0.461	0.358	0.522	4.4
en – de	0.641	0.399	0.588	0.729	6.3
en – es	0.657	0.430	0.731	0.878	8.9
en – it	0.655	0.475	0.555	0.722	2.6
es – it	0.659	0.502	0.474	0.651	5.7
en – cs	–	0.447	0.342	0.537	5.5
en – hr	–	0.404	0.301	0.478	2.2
Průměr	**0.638**	**0.445**	**0.464**	**0.631**	**5.2**

transformace se na první pohled prezentuje především velmi vysokými hodnotami naměřenými pro strojový překlad, konkrétně **46.4** % a **63.1** %. Avšak oproti základní transformaci s použitím kanonické korelace, jejíž hodnoty jsou uvedeny v tabulce 7.6, došlo k jednoprocentnímu zlepšení u všech sledovaných kategorií. Jedinou nevýhodou této metody jsou vyšší hodnoty *skewness* oproti metodám využívajícím ortogonální transformaci nebo neuronové sítě s cenovou funkcí *hinge-loss*.

Volbou této metody tedy lze **vylepšit** vlastnosti základní transformace. Svými vysokými výsledky se řadí na druhé místo hned za metodu využívající neuronové sítě s cenovou funkcí *hinge-loss* využívající regularizaci. Obsahem následující kapitoly je představení výsledků, které byly získány s pomocí hierarchického shlukování.

7.5.3 Kombinace s využitím hierarchického shlukování

Obsahem této sekce jsou výsledky pro transformace, které využívaly metody hierarchického shlukování podrobně představeného v kapitole 5.3. Pro

vytvoření jednotlivých shluků byl využit takzvaný *divizní* přístup, kdy je počáteční shluk obsahující všechny slovní vektory postupně rozdělován na menší shluky. Výsledný počet shluků byl pomocí experimentů zvolen $k = \mathbf{16}$. Stejně jako v předchozích případech také zde byla využita stejná kombinace lineárních transformací pro základní matici i matici přiřazenou danému clusteru.

Kombinace metod nejmenších čtverců

Tabulka 7.24 obsahuje naměřené výsledky pro transformaci využívající hierarchické shlukování, kdy byla pro vypočtení transformačních matic využita metoda nejmenších čtverců. Tato transformace, podobně jako v předchozích

Tabulka 7.24: Naměřené výsledky - *hierarchické shlukování v kombinaci s metodou nejmenších čtverců*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.614	0.443	0.315	0.478	10.4
de – it	0.565	0.448	0.303	0.467	9.7
en – de	0.646	0.404	0.499	0.668	3.2
en – es	0.639	0.430	0.633	0.815	4.5
en – it	0.647	0.483	0.467	0.660	3.5
es – it	0.655	0.485	0.417	0.596	5.5
en – cs	–	0.447	0.281	0.473	3.7
en – hr	–	0.418	0.278	0.447	2.6
Průměr	**0.641**	**0.436**	**0.393**	**0.571**	**5.4**

případech, které využívaly shlukování, dosáhla lepších výsledků než výchozí metoda nejmenších čtverců. Avšak s použitím hierarchického shlukování nebylo dosaženo tak výrazného zlepšení jako při použití běžné nebo vážené kombinace lineárních transformací.

Zajímavým faktem je dosažení nižších hodnot *skewness*. Snížení je patrné především u prvních dvou jazykových párů. Tento fakt platí také v případě využití nehierarchického shlukování. Přidání nové lokální informace během procesu transformace v tomto případě do jisté míry redukuje problém *hubness*.

Kombinace ortogonálních transformací

V tabulce 7.25 jsou uvedeny výsledky pro transformaci využívající hierarchické shlukování, kdy byla pro vypočtení transformačních matic využita

ortogonální transformace.

Tabulka 7.25: Naměřené výsledky - *hierarchické shlukování v kombinaci s ortogonální transformací*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.616	0.372	0.341	0.505	6.4
de – it	0.604	0.373	0.342	0.505	5.2
en – de	0.636	0.358	0.565	0.721	3.0
en – es	0.664	0.400	0.691	0.857	3.9
en – it	0.658	0.415	0.535	0.708	2.9
es – it	0.659	0.429	0.436	0.625	3.4
en – cs	–	0.392	0.323	0.519	4.4
en – hr	–	0.422	0.294	0.469	3.7
Průměr	**0.640**	**0.395**	**0.441**	**0.614**	**4.1**

Při porovnání s výsledky pro základní ortogonální transformaci, které jsou uvedeny v tabulce 7.5, lze vypozorovat drobný nárůst pro kategorii slovní podobnosti. V ostatních kategoriích však dochází ke znatelnému zhoršení výsledků. Použití lineární kombinace ortogonálních transformací v rámci transformačních metod využívajících hierarchické i nehierarchické shlukování se tedy ukázalo jako **nevhodné**. Lineární kombinací ortogonálních matic tedy nejspíše dochází k vyrušení příznivých vlastností základní ortogonální transformace, jelikož tato kombinace již nevytváří ortogonální matice.

Kombinace transformací pomocí kanonické korelace

Tabulka 7.26 obsahuje naměřené výsledky pro transformaci využívající hierarchické shlukování, kdy byla pro vypočtení transformačních matic využita kanonická korelace. U této transformace došlo jako v předchozím případě ke snížení naměřených hodnot jednotlivých kategorií oproti základní transformaci s využitím kanonické korelace. V tomto případě tedy není tato transformace vhodná pro použití v rámci metody využívající hierarchické shlukování.

V následující kapitole si představíme výsledky pro poslední kategorii využívající hierarchické nebo nehierarchické shlukování. Tato kombinace využívá neuronové sítě s cenovou funkcí *hinge-loss*, které se v kapitole 7.4.3 velmi dobře osvědčily.

Tabulka 7.26: Naměřené výsledky - *hierarchické shlukování v kombinaci s transformací pomocí kanonické korelace*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.607	0.380	0.337	0.504	5.5
de – it	0.598	0.370	0.345	0.504	4.1
en – de	0.637	0.370	0.568	0.722	5.7
en – es	0.640	0.403	0.693	0.861	7.7
en – it	0.638	0.417	0.535	0.708	1.9
es – it	0.657	0.428	0.434	0.625	4.1
en – cs	–	0.383	0.320	0.521	8.5
en – hr	–	0.396	0.288	0.462	1.9
Průměr	**0.630**	**0.393**	**0.440**	**0.613**	**5.2**

7.5.4 Kombinace neuronových sítí

Obsahem této kapitoly jsou výsledky pro metody, které využívaly metodu shlukování kombinované s neuronovými sítěmi. Vzhledem k výkonnosti neuronových sítí s cenovou funkcí *hinge-loss*, jejíž výsledky byly představeny v kapitole 7.4.3, byly pro jednotlivé transformace shluků použity právě tyto neuronové sítě. Vzhledem ke zvýšenému riziku přeučení neuronových sítí na tak malé množině trénovacích dat byl zvolen menší počet shluků, konkrétně $k = \mathbf{5}$. Přesto během prvních experimentů k přeučení docházelo, a proto byla doplněna regularizace vah neuronové sítě. Výsledný predikovaný slovní vektor byl v tomto případě vytvořen jako průměr vektoru získaného pomocí základní transformace a vektoru, který predikovala neuronová síť přiřazená k odpovídajícímu shluku.

Tabulka 7.27 uvádí naměřené výsledky pro výše uvedenou transformaci kombinující metodu shlukování a neuronové sítě využívající cenovou funkci *hinge-loss* a regularizaci. Při porovnání s výsledky se základní transformací využívající neuronové sítě s *hinge-loss* z tabulky 7.12 je na první pohled zjevné, že použití shluků v tomto případě nevede k dalšímu zlepšení vlastností transformace. K nejznatelnějšímu snížení výsledků došlo především u jazykového párů *angličtina – chorvatština*, kde patrně nastalo již zmiňované přetrénování na testovacích datech. Přesto tato metoda stále dokázala překonat základní metodu nejmenších čtverců o více než dvě procenta.

V následující sekci si představíme souhrnné výsledky všech doposud představených metod a porovnáme, které si ve sledovaných kategoriích vedly nejlépe.

Tabulka 7.27: Naměřené výsledky - *shlukování v kombinaci s neuronovými sítěmi*

Transformace	**SemEval**	**Analogie**	**MT@1**	**MT@5**	**Skewness**
de – es	0.609	0.448	0.353	0.513	5.5
de – it	0.611	0.438	0.353	0.513	3.9
en – de	0.648	0.415	0.570	0.725	4.1
en – es	0.662	0.453	0.691	0.857	4.0
en – it	0.660	0.490	0.545	0.714	3.4
es – it	0.619	0.431	0.471	0.639	3.2
en – cs	–	0.459	0.347	0.539	4.7
en – hr	–	0.244	0.114	0.238	5.4
Průměr	**0.635**	**0.393**	**0.440**	**0.613**	**4.3**

7.6 Zhodnocení výsledků

Obsahem této kapitoly je shrnutí všech výsledků všech využitých metod a jejich porovnání ve sledovaných kategoriích. Vzhledem k poměrně značnému množství provedených experimentů v rámci diplomové práce jsou pro lepší orientaci v této kapitole zavedeny následující zkratky: **LS** - metoda nejmenších čtverců, **OT** - ortogonální transformace, **CCA** - transformace pomocí kanonické korelace, **NN** - transformace s využitím neuronové sítě bez skryté vrstvy, **NN-H** - transformace s využitím neuronové sítě se skrytou vrstvou, **MSE** - cenová funkce střední kvadratická vzdálenost, **COS** - cenová funkce kosinová podobnost, **HL** - cenová funkce *hinge-loss*, **R** - regularizace, **CL** - transformace s využitím shlukování, **W-CL** vážená transformace s využitím shlukování a **HC** - transformace využívající hierarchického shlukování.

Tabulka 7.28 obsahuje souhrnné výsledky pro testy všech využitých metod uvedených v této kapitole. Výše uvedené zkratky jsou v této tabulce kombinované, tedy například **CL-OT** představuje transformaci s využitím shlukování, kdy byla pro výpočet transformačních matic využita ortogonální transformace. Záhlaví tabulky obsahuje oproti ostatním tabulkám z této kapitoly navíc nový sloupec *Průměr*. Zde je uvedena zprůměrovaná hodnota přes všechny sledované kategorie vyjma *skewness*, která má zcela odlišný rozsah hodnot.

První tři řádky tabulky ukládají výsledky pro lineární transformace z kapitoly 7.3. Z těchto experimentů dosáhla nejlepší průměrné hodnoty transformace s pomocí kanonické korelace. Tato transformace dosáhla průměrné hodnoty **53.6** %. Přesto je však důležité poznamenat, že navzdory nejvyšší

Tabulka 7.28: Kompletní výsledky

Metoda	SemEval	Analogie	MT@1	MT@5	Skewness	Průměr
LS	0.638	0.372	0.411	0.577	6.9	0.499
OT	0.632	0.418	0.446	0.607	3.8	0.526
CCA	0.629	0.436	0.457	0.623	4.5	0.536
NN-MSE	0.636	0.363	0.400	0.567	7.5	0.492
NN-COS	0.636	0.454	0.413	0.576	8.6	0.519
NN-H-TANH	0.634	0.359	0.389	0.559	7.9	0.485
NN-H-RELU	0.636	0.457	0.391	0.558	8.8	0.510
NN-H-LIN	0.637	0.355	0.373	0.544	8.6	0.477
NN-HL-1	0.637	0.433	0.457	0.618	3.2	0.536
NN-HL-2	0.638	0.432	0.460	0.620	3.1	0.538
NN-HL-5	0.637	0.427	0.459	0.618	**2.9**	0.535
NN-HL-10	0.637	0.421	0.456	0.617	**2.9**	0.533
NN-HL-R	**0.646**	0.454	**0.464**	0.625	3.6	**0.547**
CL-LS	0.641	0.436	0.393	0.571	6.0	0.520
CL-OT	0.645	0.398	0.441	0.617	4.1	0.525
CL-CCA	0.634	0.373	0.433	0.606	5.5	0.511
W-CL-LS	0.645	**0.459**	0.418	0.590	7.6	0.528
W-CL-OT	0.640	0.395	0.441	0.614	4.0	0.516
W-CL-CCA	0.638	0.445	**0.464**	**0.631**	5.2	0.544
HC-LS	0.628	0.445	0.399	0.576	5.4	0.512
HC-OT	0.640	0.395	0.441	0.614	4.1	0.522
HC-CCA	0.630	0.393	0.440	0.613	5.2	0.519
CL-NN-HL	0.635	0.431	0.431	0.592	4.3	0.522

průměrné hodnotě tato transformace nedominovala ve všech sledovaných kategoriích. Tento fakt je nejvíce patrný u kategorie *skewness*, kde si díky svým vlastnostem vedla nejlépe ortogonální transformace. Kategorii slovní podobnosti ovládla nejjednodušší metoda nejmenších čtverců.

Další sekce tabulky uvádí souhrnné výsledky pro neuronové sítě představené v kapitole 7.4. Z této sekce stojí za zmínku především solidní výkon neuronové sítě bez skryté vrstvy využívající cenovou funkci kosinové podobnosti. Tato neuronová síť dosáhla velmi dobrého výsledku **45.4** % v kategorii slovních analogií. Jedná se o druhou nejlepší naměřenou hodnotu. Tato transformace současně dokázala překonat základní metodu nejmenších čtverců. Bohužel všechny metody využívající neuronové sítě z kapitoly 7.4 se vyznačují velmi vysokou hodnotou *skewness*. Při použití těchto metod tedy dochází ke značné deformaci transformovaného sémantického prostoru.

Třetí sekce této tabulky obsahuje výsledky pro neuronové sítě využívající speciální cenovou funkci *hinge-loss*, která byla představena v kapitole 7.4.3. Výsledky pro tyto neuronové sítě ostatním metodám dominují. Na první pohled jsou znatelné velmi nízké hodnoty *skewness*, které jsou dokonce ve všech případech nižší než hodnoty naměřené pro ortogonální transformaci, která díky svým vlastnostem tento problém oproti ostatním metodám značně redukuje. Neuronové sítě s *hinge-loss*, které využívaly jeden, případně dva negativní příklady, dosáhly srovnatelných výsledků s nejlepší lineární transformací s pomocí kanonické korelace, avšak navíc se znatelně nižší hodnotou *skewness*. Nejlepšího průměrného výsledku **54.7** % dosahuje neuronová síť využívající regularizaci. Tato neuronová síť překonává transformaci pomocí kanonické korelace ve **všech** uvedených kategoriích. Vyjma slovního překladu pro pět nejbližších slov se jedná o zlepšení o více než jedno procento. Ze všech testovaných metod v rámci diplomové práce byla tato metoda naprosto **nejlepší**. Jedinou kategorií, ve které tato neuronová síť nedosáhla maximálního výsledku, byla kategorie slovních analogií, kde byla naměřena nejvyšší hodnota u vážené kombinace metod nejmenších čtverců popsaná v kapitole 7.5.2.

Následující sekce tabulky uvádí výsledky pro transformace využívající metodu shlukování představené v kapitole 7.5.1. Všechny uvedené metody překonaly základní metodu nejmenších čtverců, avšak u kombinací, které využívaly ortogonální transformaci, respektive transformaci s použitím kanonické transformace nedošlo ke zlepšení výsledků oproti odpovídající lineární transformaci. Z těchto metod si nejlépe vedla kombinace využívající ortogonální transformaci, která dosáhla průměrného výsledku **52.5** %.

Pátá sekce tabulky obsahuje výsledky pro vážené kombinace transformací využívající metodu shlukování, jejíž výsledky byly prezentovány v sekci 7.5.2. Významného výsledku dosáhla především již dříve zmíněná transformace využívající váženou kombinaci nejmenších čtverců. Tato transformace zcela dominovala v kategorii slovních analogií, kde dosáhla hodnoty **45.9** %. Nejlepšího průměrného výsledku v této sekci a zároveň druhého nejlepšího výsledku ze všech provedených experimentů dosáhla s hodnotou **54.4** % metoda používající váženou kombinaci transformací využívající kanonickou korelaci. Tato metoda se vyznačovala především velmi vysokými hodnotami pro strojový překlad, konkrétně **46.4** % a **63.1** %. V případě druhého uvedeného výsledku pro pět nejbližších nalezených překladů dosáhla nejlepšího výsledku ze **všech** provedených experimentů.

Další sekce zahrnuje výsledky pro transformace využívající hierarchické shlukování, jehož experimenty jsou uvedeny v sekci 7.5.3. Výsledky těchto metod jsou do značné míry porovnatelné s výsledky pro shlukování využí-

vající metodu *k-means*. V této sekci si s průměrnou hodnotou **52.5** % nejlépe vedla kombinace využívající ortogonální transformaci. Tato hodnota je současně identická s hodnotou naměřenou v případě nehierarchického shlukování.

Sedmá a zároveň poslední část tabulky uvádí výsledky pro kombinaci metody shlukování a neuronových sítí využívajících cenovou funkci *hinge-loss*. Výsledky této metody jsou poměrně slibné, neboť překonává základní metodu nejmenších čtverců, avšak současně dochází k velmi výraznému poklesu hodnot pro slovní analogie, čímž je zároveň snížena také výsledná hodnota.

8 Závěr

Tato diplomová práce se zabývala projekcí sémantických prostorů do jednoho sdíleného univerzálního prostoru. V průběhu řešení byly implementovány aktuálně nejlepší existující lineární transformace, které sloužily jako výchozí bod práce. Testy byly provedeny na celkem šesti jazycích ze tří odlišných jazykových skupin, konkrétně germánské, románské a slovanské.

Nejprve jsem implementoval lineární transformace představované metodou nejmenších čtverců, ortogonální transformací a transformací s pomocí kanonické korelace. Lineární transformace sloužily jako východisko pro srovnání s mnou navrženými metodami. Dále jsem implementoval nelineární transformace využívající neuronové sítě bez skryté vrstvy a se skrytou vrstvou a současně jsem zkoumal různé cenové funkce. Jako nejlepší se ukázala cenová funkce *hinge-loss*. Pomocí shlukování jsem začal jednotlivé transformace kombinovat, čímž došlo k dalším zlepšením.

Hlavním přínosem této diplomové práce je především výzkum na poli nelineárních metod určených pro transformaci sémantických prostorů, zejména návrh zcela nových metod řešících tuto úlohu. Velmi slibné výsledky prokázaly především nelineární metody založené na umělých neuronových sítích využívajících cenovou funkci *hinge-loss*. Ve všech sledovaných kategoriích překonaly všechny analyzované lineární transformace. Konkrétně neuronová síť využívající regularizace dosáhla nejvyšší hodnoty pro slovní podobnost a strojový překlad. Oproti nejlepší lineární transformaci pomocí kanonické korelace došlo ke zlepšení o více než jedno procento absolutně a v průměru nad všemi jazyky a všemi aplikacemi.

Pro zlepšení vlastností základních lineárních transformací jsem využil metodu shlukování společně s lineární kombinací lineárních transformací. Cílem těchto metod bylo zpřesnění transformace zavedením nové lokální informace. Jako zvláště účinné se prokázaly vážené lineární kombinace, které dokázaly značným způsobem vylepšit vlastnosti použitých základních transformací. Vážené transformace využívající metodu nejmenších čtverců dosáhly nejlepšího výsledku v kategorii slovních analogií. Oproti samostatné transformaci došlo ke zlepšení o 9 %.

V kategorii slovních podobností dosáhla nejlepšího výsledku transformace využívající neuronové sítě s regularizací a cenovou funkcí *hinge-loss*, která byla nejkonzistentnější a současně dosáhla nejnižší hodnoty *skewness* a nejlepší průměrné hodnoty. V těsném sledu se umístila vážená kombinace využívající transformaci pomocí kanonické korelace, která zároveň ovládla

kategorii strojového překladu. Nejlepšího výsledku v kategorii slovních analogií dosáhla vážená kombinace využívající metodu nejmenších čtverců.

Závěrem lze tedy konstatovat, že transformace využívající neuronové sítě s regularizací a cenovou funkcí *hinge-loss* a vážená kombinace využívající transformaci pomocí kanonické korelace výrazným způsobem překonaly lineární transformace. Všechny experimenty byly prováděny v prostředí MetaCentra. Realizace výpočtů si vyžádala téměř tisíc hodin.

Seznam zkratek

CBOW Continuous Bag-of-Words – architektura modelu Word2vec

CCA Canonical Correlation Analysis – kanonická korelační analýza

CL Clustering – shlukování

COS Cosine Similarity – kosinová podobnost

CS Czech – čeština

DE German – němčina

EN English – angličtina

ES Spanish – španělština

HL Hinge-Loss – cenová funkce *hinge-loss*

HR Croatian – chorvatština

IT Italian – italština

LS Least Squares – metoda nejmenších čtverců

MSE Mean Squared Error – střední kvadratická chyba

MT Machine Translation – strojový překlad

NLP Natural Language Processing – zpracování přirozeného jazyka

NN Neural Network – neuronová síť

RELU Rectified Linear Unit – aktivační funkce ReLU

R Regularization – regularizace

TANH Hyperbolic Tangent – aktivační funkce hyperbolický tangens

Seznam obrázků

Seznam tabulek

Literatura

[1] Artetxe, M. – Labaka, G. – Agirre, E. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, s. 2289–2294, 2016.

[2] Arthur, D. – Vassilvitskii, S. k-means++: The advantages of careful seeding. Technical report, Stanford, 2006.

[3] Bojanowski, P. – Grave, E. – Joulin, A. – Mikolov, T. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*. 2017, 5, s. 135–146.

[4] Brychcín, T. Linear Transformations for Cross-lingual Semantic Textual Similarity. *Knowledge-Based Systems*. 2020, 187, 1, s. 1–9. ISSN 0950-7051.

[5] Brychcín, T. – Taylor, S. E. – Svoboda, L. Cross-lingual word analogies using linear transformations between semantic spaces. *Expert Systems with Applications*. 2019, 135, s. 287–295. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2019.06.021. Dostupné z: `https://www.sciencedirect.com/science/article/pii/S0957417419304191`.

[6] Camacho-Collados, J. – Pilehvar, M. T. – Collier, N. – Navigli, R. SemEval-2017 Task 2: Multilingual and Cross-lingual Semantic Word Similarity. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, s. 15–26, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/S17-2002. Dostupné z: `https://www.aclweb.org/anthology/S17-2002`.

[7] Deisenroth, M. – Faisal, A. – Ong, C. *Mathematics for Machine Learning*. Cambridge University Press, 2020. ISBN 9781108470049.

[8] Faruqui, M. – Dyer, C. Improving Vector Space Word Representations Using Multilingual Correlation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, s. 462–471, Gothenburg, Sweden, April 2014. Association for Computational Linguistics. doi: 10.3115/v1/E14-1049. Dostupné z: `https://www.aclweb.org/anthology/E14-1049`.

[9] Firth, J. R. A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*. 1957.

[10] Gutmann, M. U. – Hyvärinen, A. Noise-Contrastive Estimation of Unnormalized Statistical Models, with Applications to Natural Image Statistics. *Journal of Machine Learning Research.* 2012, 13, 2.

[11] Hardoon, D. R. – Szedmak, S. – Shawe-Taylor, J. Canonical Correlation Analysis: An Overview with Application to Learning Methods. *Neural Computation.* 2004, 16, 12, s. 2639–2664. doi: 10.1162/0899766042321814. Dostupné z: https://doi.org/10.1162/0899766042321814.

[12] Hayesová, N. *Základy sociální psychologie.* Praha: Portál, 2013. ISBN 80-7178-763-9.

[13] Joulin, A. – Grave, E. – Bojanowski, P. – Mikolov, T. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, s. 427–431, Valencia, Spain, April 2017. Association for Computational Linguistics. Dostupné z: https://www.aclweb.org/anthology/E17-2068.

[14] Lamprecht, A. *Praslovanština.* 1. vyd. Brno: Univerzita J.E. Purkyně, 1987.

[15] Lazaridou, A. – Dinu, G. – Baroni, M. Hubness and Pollution: Delving into Cross-Space Mapping for Zero-Shot Learning. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, s. 270–280, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1027. Dostupné z: https://www.aclweb.org/anthology/P15-1027.

[16] McCulloch, W. S. – Pitts, W. A Logical Calculus of the Ideas Immanent in Nervous Activity. *The Bulletin of Mathematical Biophysics.* 1943, 5. Dostupné z: http://dx.doi.org/10.1007/BF02478259.

[17] Mikolov, T. – Chen, K. – Corrado, G. – Dean, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781.* 2013.

[18] Mikolov, T. – Le, Q. V. – Sutskever, I. Exploiting Similarities among Languages for Machine Translation. *CoRR.* 2013, abs/1309.4168. Dostupné z: http://arxiv.org/abs/1309.4168.

[19] Mikolov, T. – Sutskever, I. – Chen, K. – Corrado, G. S. – Dean, J. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, s. 3111–3119, 2013.

[20] Mnih, A. – Teh, Y. W. A Fast and Simple Algorithm for Training Neural Probabilistic Language Models. In *International Conference on Machine Learning*, ICML'12, s. 419–426, Madison, WI, USA, 2012. Omnipress. ISBN 9781450312851.

[21] 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*, R5 / *Proceedings of Machine Learning Research*, s. 246–252. PMLR, 06–08 Jan 2005. Dostupné z: http://proceedings.mlr.press/r5/morin05a.html. Reissued by PMLR on 30 March 2021.

[22] Nomizu, K. – Katsumi, N. – Sasaki, T. – Bollobas, B. – Fulton, W. – Katok, A. – Kirwan, F. – Sarnak, P. – Simon, B. *Affine Differential Geometry: Geometry of Affine Immersions*. Cambridge Tracts in Mathematics. Cambridge University Press, 1994. ISBN 9780521441773.

[23] Pennington, J. – Socher, R. – Manning, C. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, s. 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. Dostupné z: https://www.aclweb.org/anthology/D14-1162.

[24] Radovanovic, M. – Nanopoulos, A. – Ivanovic, M. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research*. 2010, 11, sept, s. 2487–2531.

[25] Rosenblatt, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*. 1958, 65, 6, s. 386–408. Dostupné z: https://doi.org/10.1037/h0042519.

[26] Rumelhart, D. E. – Hinton, G. E. – Williams, R. J. Learning representations by back-propagating errors. *Nature*. 1986, 323, 6088, s. 533–536.

[27] Schmidhuber, J. Deep learning in neural networks: An overview. *Neural networks*. 2015, 61, s. 85–117.

[28] Silver, D. – Schrittwieser, J. – Simonyan, K. – Antonoglou, I. – Huang, A. – Guez, A. – Hubert, T. – Baker, L. – Lai, M. – Bolton, A. – others. Mastering the game of go without human knowledge. *Nature*. 2017, 550, 7676, s. 354–359.

[29] Maaten, L. – Hinton, G. Visualizing Data using t-SNE. *Journal of Machine Learning Research*. 2008, 9, 86, s. 2579–2605. Dostupné z: http://jmlr.org/papers/v9/vandermaaten08a.html.

A Uživatelská dokumentace

V přiloženém ZIP souboru jsou umístěny všechny zdrojové kódy sloužící pro otestování lineárních i nelineárních transformací společně s výsledným JAR souborem, který již obsahuje všechny požadované závislosti. Před samotným spuštěním aplikace je však nutné překopírovat data, která jsou umístěna ve složce `Vstupni_data`, k výše zmíněnému souboru JAR. Vzhledem k značné velikosti souborů s předtrénovanými slovními vektory modelu fastText jsou v archivu pro jednotlivé jazyky umístěny pouze zkrácené verze obsahující přesně 100 000 nejčastějších slov. Jedná se tedy o stejný počet, který byl využit pro provádění experimentů v rámci diplomové práce.

Po spuštění aplikace je vypsána nápověda a podrobná informace, jaké jazyky a transformace jsou uživateli dostupné. Samotné ovládání aplikace je velmi intuitivní a spočívá v doplnění požadovaných parametrů. Vybraná metoda je následně otestována na zvoleném jazykovém páru.

A.1 Obsah přiloženého ZIP

Přiložený soubor ZIP obsahuje následující složky:

- `Aplikace_a_knihovny`
  - `doc` - obsahuje dokumentaci projektu.
  - `jar` - složka obsahující výsledný jar soubor s pluginem.
  - `out` - obsahuje binární soubory, přeložené pomocí *Java* verze 8.
  - `src` - zde jsou umístěny zdrojové kódy knihovny.
  - `pom.xml` - umožňuje překlad aplikace pomocí systému *Maven*.
  - `Vectors.jar` - výsledný spustitelný soubor aplikace.
- `Poster` - obsahem této složky jsou oba soubory posteru.
- `Text_prace` - obsahuje zdrojové soubory LaTeX a PDF této práce.
- `Vstupni_data` - zde jsou uložena všechna data, která jsou nezbytná pro spuštění aplikace.
  - `analogy` - obsahuje datovou sadu slovních analogií.
  - `dictionaries` - obsahem této složky jsou slovníky pro jednotlivé jazykové páry.

  - `evaluation` - zde jsou umístěné anotované klíče datasetu slovních podobností.
  - `matrices` - obsahuje data natrénovaných neuronových sítí.
  - `native` - zde je umístěna nativní knihovna pro urychlení výpočtů.
  - `python` - obsahem této složky jsou zdrojové kódy pro trénování neuronových sítí v programovacím jazyce *Python*.
  - `tests` - obsahuje páry z datové sady slovních podobností.
  - `temp` - složka určená pro umístění dočasných výsledků.
  - `vectors` - zde jsou umístěny slovní vektory pro jednotlivé jazyky.
  - `task2-scorer.jar` - program z konference SemEval, který slouží pro vyhodnocení kategorie slovní podobnosti.
- `Vysledky` - obsahuje tabulku se všemi výsledky vypočtenými v průběhu tvorby práce.
- `Readme.txt` - tento soubor obsahuje podrobný popis obsahu archivu a procesu spuštění aplikace.