VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

POČÍTAČ JAKO INTELIGENTNÍ SPOLUHRÁČ VE SLOVNĚ-ASOCIAČNÍ HŘE KRYCÍ JMÉNA

COMPUTER AS AN INTELLIGENT PARTNER IN THE WORD-ASSOCIATION GAME CODENAMES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **PETR JAREŠ**
AUTHOR

VEDOUCÍ PRÁCE **doc. RNDr. PAVEL SMRŽ, Ph.D.**
SUPERVISOR

BRNO 2019

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2018/2019

Zadání bakalářské práce

Student: **Jareš Petr**
Program: Informační technologie
Název: **Počítač jako inteligentní spoluhráč ve slovně-asociační hře Krycí jména**
Computer as an Intelligent Partner in the Word-Association Game Codenames
Kategorie: Umělá inteligence

Zadání:

1. Seznamte se s principem hry Krycí jména a se strategiemi hraní, prostudujte oblast automatického určování sémantické příbuznosti slov se zaměřením na vazby mezi skupinami slov.
2. Shromážděte data potřebná pro trénování automatických metod, která budou mít dostatečné pokrytí slov používaných ve hře.
3. Navrhněte a implementujte systém, který bude schopen hrát roli zadavatele asociovaných slov (hlavního agenta) i hadače (člena kontaktního týmu).
4. Ověřte kvalitu vytvořeného systému v reálných hrách a vyhodnoťte zvolenou strategii v porovnání s kvalitními hráči.
5. Vytvořte stručný plakát prezentující vytvořenou práci a její výsledky

Literatura:

- dle doporučení vedoucího

Pro udělení zápočtu za první semestr je požadováno:

- fFunkční prototyp řešení

Podrobné závazné pokyny pro vypracování práce viz http://www.fit.vutbr.cz/info/szz/

Vedoucí práce: **Smrž Pavel, doc. RNDr., Ph.D.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2018
Datum odevzdání: 15. května 2019
Datum schválení: 1. listopadu 2018

Abstrakt

Tato práce řeší určování sémantické podobnosti slov. K tomu je využita kombinace prediktivního modelu fastText a metody založené na počtu Pointwise Mutual Information. Je zde popsán systém, který s využitím sémantických modelů je schopen zastoupit hráče ve slovně-asociační hře Krycí jména. Systém má implementovanou herní strategii využívající informace z průběhu hry k prospěchu týmu, za který hraje. Systém je schopen plnit funkci hráče hádajícího asociovaná slova k nápovědě, tak i hráče vytvářejícího vlastní nápovědy.

Abstract

This thesis solves a determination of semantic similarity between words. For this task is used a combination of predictive model fastText and count based method Pointwise Mutual Information. Thesis describes a system which utilizes semantic models for ability to substitue a player in a word association game Codenames. The system has implemented game strategy enabling use of context information from the game progression to benefit his own team. The system is able to substitue a player in both team roles.

Klíčová slova

zpracování přirozeného jazyka, sémantická podobnost, Krycí jména, umělý hráč, fastText, Pointwise Mutual Information, Python

Keywords

natural language processing, semantic similarity, Codenames, artificial player, fastText, Pointwise Mutual Information, Python

Citace

JAREŠ, Petr. *Počítač jako inteligentní spoluhráč ve slovně-asociační hře Krycí jména.* Brno, 2019. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce doc. RNDr. Pavel Smrž, Ph.D.

Počítač jako inteligentní spoluhráč ve slovně-asociační hře Krycí jména

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana doc. RNDr. Pavla Smrže, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Petr Jareš
8. května 2019

Poděkování

Rád bych poděkoval svému vedoucímu panu doc. RNDr. Pavlovi Smržovi, Ph.D. za odborné vedení práce a také panu Ing. Martinovi Fajčíkovi za odborné rady a podněty.

Obsah

Kapitola 1

Úvod

Vytvoření umělého hráče ve slovně-asociační hře Krycí jména nabízí možnost ověřovat funkcionalitu technik zpracování přirozeného jazyka formou hry. Zpracování přirozeného jazyka (natural language processing, NLP) je obor počítačové vědy, který se zabývá komunikací mezi člověkem a počítačem pomocí přirozeného jazyka, jímž mluví lidé. Hlavním cílem NLP je umožnit počítačům porozumět lidskému jazyku stejně dobře, jako mu rozumí lidé. Výsledky výzkumu v oblasti NLP lze využít například při strojovém překladu textu, vyhledávání relevantních zdrojů a převodu textu na řeč (text-to-speech). [6]

Cílem této práce je vytvořit systém, který je schopen zastoupit hráče Krycích jmen v roli hádajícího hráče i hráče vytvářejícího nápovědy. Pomocí implementované herní strategie je cílem se co nejvíce dovednostně vyrovnat zkušeným hráčům. Z oblasti NLP jsou v rámci této práce trénovány sémantické modely, s jejichž použitím lze vyhodnocovat slovní asociace, které jsou základním principem hry Krycí jména. Vytvořený systém slouží nejen jako aplikační doména NLP, ale také provádí sběr dat, která mohou sloužit k vyhodnocování nebo případně k trénování sémantických modelů.

V rámci uvedené problematiky navazuji na bakalářskou práci *Shlukování slov podle významu* [7], která se zabývá metodami pro určování sémantické podobnosti slov. Dále navazuji na diplomovou práci *Počítač jako inteligentní spoluhráč ve slovně-asociační hře Krycí jména* [15], která sloužila jako referenční řešení. Tato diplomová práce je dále v textu označovaná jako „předchozí práce“. Obdobně jako v předchozí práci využívám sémantický model fastText, ale vytvořil jsem i model založený na počtu, který při testování dosahoval lepších výsledků (podkapitola 5.3.4).

Jádro předkládané práce kombinuje nejen sémantické modely, ale také implementovanou herní strategií, která například při napovídání ve hře definuje hodnotící funkci možných nápověd, která bere v potaz nejen míru asociace se slovy ve hře, ale také předchozí tahy a počet zbývajících slov každého týmu.

Výsledný systém obsahuje i webovou službu[1], kde je možno ve webovém prohlížeči testovat obě role implementovaných hráčů. Původně byl systém implementován jen pro český jazyk, dotatečně však byla implementována i podpora pro angličtinu. Ve spolupráci s Martinem Hurtou, který souběžně pracoval na bakalářské práci *Podpora hry Krycí jména na mobilním telefonu s OS Android* [8], bylo vytvořeno komunikační rozhraní mezi webovou službou a mobilní aplikací, které slouží jako rádce pro aktuální tah ve hře nebo kompletně zastoupí hráče v mobilní aplikaci.

[1] http://athena3.fit.vutbr.cz:8086/

Kapitola 2 obsahuje obecný úvod do strategie her společně s detailním popisem hry Krycí jména. Kapitola 3 se zabývá předzpracováním textu. Kapitola 4 se zabývá technikami, které vyhodnocují podobnosti slov na základě jejich kontextu v textu určenému k trénování. Kapitola 5 popisuje návrh a implementaci systému, společně se zvolenou herní strategií. V kapitole 6 jsou uvedené dosažené výsledky.

Kapitola 2

Strategie her

V této kapitole je představen letmý úvod pojmů teorie her, strategie a užitková funkce. Nejsou zde popsány exaktní definice, pojmy spojené se strategií byly v implementační části používané spíše na intuitivní úrovni.

Hry a umělá inteligence mají společnou dlouhou historii. Již od počátků vývoje umělé inteligence byl výzkum zaměřen na vytváření programů schopných hrát deskové hry. Především se jednalo o hry šachy a dáma. Časem bylo vyvinuto několik základních metod, které se dnes běžně používají. Výběr těchto metod je zavislý na principu dané hry. [23]

Teorie her je oblast studia zabývající se matematickými modely, které jsou v interakci s objekty, jež provádějí racionální rozhodování v závislosti na dané situaci. Tato oblast se využívá nejen u hraní her, ale také například v ekonomii nebo sociálních studiích. [14] Pojem strategie lze vysvětlit jako postup volící provedení jedné z možných akcí, kterou lze co nejlépe dosáhnout předem stanoveného cíle. U her se jedná o provedení takového tahu, který je vyhodnocen, jako nejlepší volba pro dosažení vítězství. Pro výběr optimálního tahu slouží například užitková funkce.

Užitkovost v teorii her představuje míru ohodnocení daného rozhodnutí. Jedná se o funkci, která ohodnotí dané rozhodnutí na základě informací z aktuálního stavu hry. Na základě vyhodnocení možných proveditelných rozhodnutí lze vybrat právě takové, které bude mít největší užitek pro dosažení výhry. [23]

2.1 Krycí jména

Krycí jména je původní česká hra autora Vlaadi Chvátila, která sbírá úspěchy po celém světě. Získala ocenění Spiles des Jahres 2016 a podle celosvětové herní databáze[1] je nejlepší párty hra historie [4]. Hra je založena na prostředí tajných agentů, kde cílem hry je kontaktovat všechny agenty svého týmu dříve než soupeř. Tito agenti jsou reprezentováni slovy vyloženými ve hře. Princip hry je vytváření jednoslovních asociací, které by měly spojovat slova daného týmu a umožnily tímto ostatním členům týmu kontaktovat své agenty.

[1] https://www.boardgamegeek.com/

Obrázek 2.1: Ukázka herních karet. Převzato z [4] a upraveno.

2.1.1 Příprava hry a pravidla

Na začátku hry se náhodně vybere 25 kartiček se slovy, které se poskádají do mřížky 5×5. Hráči se rozdělí do dvou týmů, kde z každého týmu se vybere jeden hráč, který bude hlavní špión. Zbytek týmu bude představovat členy operativy. Dále se náhodně vybere hrací plánek, který určí rozřazení slov. Začínající tým má 9 slov (agentů), druhý tým 8, jedno slovo bude nájemný vrah a zbytek náhodní kolemjdoucí. Podle plánku, na který vidí jenom hlavní špióni obou týmu, se snaží spojit co nejvíce svých slov do jedné slovní asociace, kterou řeknou členům svého týmu společně s počtem spojovaných slov. Ostatní členové týmu následně po jednom budou označovat slova, o kterých si myslí, že byla cílena. Tým, který jako první označí všechny své agenty a zároveň se vyhne nájmenému vrahovi, vyhrává.

Zde jsou upřesňující kritéria pro jednotlivé tahy:

- Nápověda musí být jenom jedno slovo.
- Nápověda nesmí obsahovat kořen žádného slova ze hry.
- Nápověda musí mít pouze sémantickou asociaci se slovy ve hře.[2]
- Hlavní špión nesmí kromě jedné asociace za tah používat jiné verbální i neverbální nápovědy.

Speciální čísla pro nápovědu mohou být 0 a nekonečno. Nápověda typu nekonečno znamená, že hráči můžou hádat kolik slov uznají za vhodné. Nápověda typu nula cílí slovo nebo slova, která právě nemají nic společného s danou nápovědou. Tato nápověda se většinou cílí na

[2] např: Nápověda „čtyři“ 2: nesmí cílit slova, která se skládají ze čtyř písmen.

nájemného vraha. Oba typy speciálních nápověd se vyplatí používat v případě, že týmu zbývá dohádat několik slov z předchozích tahů.

- Hráči musí na zadanou nápovědu označit alespoň jedno slovo.
- Celkově se postupně může označit o jedno slovo více, než kolik bylo zadáno u nápovědy.[3]
- Pokud při postupném označování slov narazí na náhodného kolemjdoucího nebo na agenta nepřátelského týmu, tah končí a pokračuje ve hře druhý tým.
- Pokud se označí nájemný vrah, hra končí a vyhrává protivník.

2.1.2 Princip strategie

Z pohledu strategie je hra Krycí jména odlišná od běžných deskových her. Při srovnání se hrou šachy mají obě hry herní pole, ale šachy fungují na principu války. Za použití vlastních zdrojů se hráč snaží likvidovat zdroje protivníka. U Krycích jmen operují oba týmy na stejném poli, ale nijak svými akcemi nezpůsobují škody nepříteli, pouze se snaží zužitkovat informace k odhalení vlastních karet, což znamená, že se jedná spíše o závod.

Hlavní princip, který ovlivňuje prováděný tah, je vytváření slovních asociací. Příklady základních slovních asociací jsou:

- **synonyma** – slova se stejným významem (dům–budova)
- **antonyma** – slova s opačným významem (zima–horko)
- **hyperonyma** – nadřazený pojem (ovoce–jablko)
- **homonyma** – slova se stejnou zvukovou nebo grafickou podobou avšak s odlišným významem (raketa jako tenisová pálka nebo létací stroj)
- **analogie** – přirovnání dvou slov na základě stejné vlastnosti (pták–letadlo)

Ze záznamů her však vyplývá, že asociace nemusí být jenom ze skupiny těchto základních typů. Může se například využívat obecná znalost světa, vztahy známých názvů nebo osobností s místy, vlastnostmi a událostmi. Například slovo „Alexandrovci“ může být použito jako nápověda pro slova „Moskva“ a „hudba“. Asociace se můžou přenášet i přes různé specifické vztahy. Pro slova „obr“ a „hodinky“ může být vytvořena asociace se slovem „Ben“, které vychází ze známého názvu hodinové věže „Big Ben“.

V roli hlavního špióna je výběr optimální nápovědy takový, aby nápověda spojovala co nejvíce slov a zároveň je její síla asociace s cílenými slovy dost silná, aby byla velká pravděpodobnost, že spoluhráč daná slova označí. Dále je potřeba brát ohled na počty zbývajících slov obou týmů. V případě že hráč výrazněji prohrává, vyplatí se zariskovat a vybrat nápovědu, která cílí více slov, ale jejich asociace s nápovědou nebude tak silná. Pro výběr optimální nápovědy se nabízí implementace užitkové funkce, která by v ideální míře váhovala sílu asociace nápovědy se slovy v kombinaci s počtem cílených slov.

V roli člena operativy je vhodné sledovat i nepřátelské nápovědy pro případ, že by protivník pokazil svůj tah a hráč by tak mohl mít podezření, která slova jsou nepřátelského týmu. Výběr slov k nápovědě může také ovlivnit úvaha nad tím, proč byla použita právě

[3] Hráč může zkusit uhádnout nějaké slovo, které nezvládl z předchozích kol.

daná nápověda a ve specifické mluvnické kategorii. Z pohledu obou rolí je užitečné si pamatovat předchozí tahy. Člen operativy tak může postupně v dalších kolech označit zbývající slova z předchozích tahů a hlavní špión potom s těmito slovy bude počítat a může se zaměřit na vytváření nápověd pro další slova.

Kapitola 3

Předzpracování textu

V této kapitole jsou popsány techniky používající se pro předzpracování textu, který se použije pro trénování sémantických modelů. Vhodnou úpravou lze docílit rychlejšího vytvoření modelu, který zároveň bude méně paměťově náročný a přesnější v rámci určování podobnosti slov. Toto se projeví zvláště u jazyků, které jsou morfologicky bohaté, jako je například čeština. Pro tento účel slouží techniky jako lemmatizace, stemming a vyřazování nepotřebných slov.

3.1 Korpus

Termínem korpus se nejběžněji definuje tělo textu, které je možno zpracovat strojem. Přesněji korpus označuje konečnou kolekci textů zpracovatelnou strojem, která maximálně reprezentuje daný jazyk ideálně ve všech jeho oblastech, jako je například věda, kultura, a politika [5]. Pro hru Krycí jména může mít volba trénovacího textu zásadní vliv. Jak již bylo zmíněno v podkapitole 2.1.2, je dobré pracovat s co nejrozmanitějšími a aktuálními významnými informacemi ze světa. Proto by nebylo vhodné použít například jenom kolekci beletrie, ale také třeba různé novinové články nebo internetové diskuze.

3.2 Odstranění nepotřebných slov

Nejvíce frekventovaná slova v přirozeném jazyce velice často nenesou samy o sobě žádný sémantický význam. Těmito slovy bývají například předložky, spojky a zájmena. Filtrováním těchto slov můžeme značně snížit časovou náročnost vytváření sémantických modelů a paměťovou náročnost při jejich využívání. Při zpracování korpusu je můžeme filtrovat pomocí takzvaného *stoplistu*, kde jsou uvedená jednotlivá slova, která nechceme zpracovávat. [21]

3.3 Lemmatizace

Lemmatizace je technika, která se za použití slovníku a morfologické analýzy snaží odstranit sufixy skloňovaného slova [1]. Tímto procesem vznikne slovo v základním tvaru, takzvaná lemma. Například pokud se v korpusu vyskytnou slova „dělal“ a „dělá“, lemmatizací se obě slova upraví na slovo „dělat“. Díky této technice se sníží počet unikátních slov v daném modelu a zlepší se přesnost vyhodnocování podobnosti, protože různé derivace stejného slova pořád reprezentují stejný sémantický význam.

3.4 Stemming

Na rozdíl od lemmatizace se stemming snaží pomocí odstraňování prefixů a sufixů najít kmen slova. Stemming je algoritmicky obtížný u morfologicky bohatých jazyků. Může zde docházet k understemmingu, neboli k nedostatečnému odstranění prefixů a sufixů nebo k overstemmingu, což je opak understemmingu. Stemming lze využít jako zjednodušenou variantu úplné lemmatizace, hlavně v angličtině.

Kapitola 4

Distribuční sémantika

V této kapitole jsou popsány sémantické modely, které slouží pro vyhodnocování podobnosti slov. Tyto modely se dělí na modely založené na počtu a prediktivní modely. Princip sémantických modelů vychází z distribuční sémantiky.

Distribuční sémantika je studie, která se zabývá metodami pro vyhodnocování podobnosti jazykových útvarů na základě jejich vzájemné distribuce v kontextu. Jejím základem je distribuční hypotéza. Ta říká, že stupeň sémantické podobnosti dvou jazykových výrazů A a B je funkce podobnosti jazykových kontextů, ve kterých se výrazy A a B můžou vyskytnout [9]. Z této hypotézy vychází celá řada sémantických modelů. Nelze jednoznačně určit, který model nebo kategorie modelů přináší nejlepší výsledky. Pro různé úlohy a přirozené jazyky se může úspěšnost modelů lišit [2].

4.1 Modely založené na počtu

Mezi prvními sémantickými modely byly modely založené na počtu (count based models) [2]. Jejich základem je matice spoluvýskytů, kde jsou uloženy hodnoty pro dané výrazy, kolikrát se vzájemně vyskytly v určeném kontextu. Kontext může být věta, odstavec nebo i celý dokument. Matice vyjadřuje spoluvýskyt typicky [*slova* : *slova*]. Může se také používat [*slova* : *odstavce*] nebo [*slova* : *dokumenty*]. S použitím matice spoluvýskytů lze vyhodnotit podobnost daných dvou výrazů. Obecně lze uplatnit tvrzení, že čím častěji se spolu dvě slova vyskytnou v kontextu, tím více si jsou sémanticky podobná.

	auto	**motor**	**zeď**	**telefon**
auto	0	3	2	2
motor	3	0	1	0
zeď	2	1	0	1
telefon	2	0	1	0

Tabulka 4.1: Příklad matice spoluvýskytů slov

U modelů založených na počtu jsou jednotlivá slova reprezentována jako unikátní indexy matice. Pokud má matice typu [*slova* : *slova*] stejné indexy slov pro řádky i sloupce, tak je zákonitě symetrická. Při velkém rozsahu korpusu s velkým počtem unikátních slov je možnost matici uložit v řídkém formátu a „odseknout“ všechny prvky pod diagonálou. Potom by se ale musel zařídit správný přístup k hodnotám matice, kde by se používaly pouze hodnoty nad diagonálou. Nevýhodou modelů založených na počtu oproti predik-

tivním modelům je paměťová náročnost, kdy s přibývajícím počtem unikátních slov roste reprezentace každého slova v rámci spoluvýskytu s ostatními slovy. U prediktivních modelů je reprezentace každého slova dána N-dimenzionálním vektorem bez ohledu na celkový počet unikátních slov.

4.1.1 Mutual Information

Mutual Information (doslovně by se dalo přeložit jako společná informace) je kritérium, které se běžně používá ve statistickém modelování přirozeného jazyka. Z matematického pohledu Mutual Information ohodnocuje vztah mezi dvěmi náhodnými proměnnými z pravděpodobnostního rozložení. Ohodnotí, kolik informací výskyt jedné proměnné dává o výskytu druhé proměnné. [16]

Existuje zde rozdíl mezi Pointwise Mutual Information (PMI) a Mutual Information (MI). Pointwise Mutual Information udává společnou informaci, ale v rámci výskytu daného jednoho jevu z každé náhodné proměnné. Mutual Information udává průměrnou hodnotu PMI všech jevů z obou náhodných proměnných. I když se jedná o rozdílné věci, někdy dochází k jejich záměně. Uvažujme, že X a Y jsou dvě náhodné proměnné, x a y jsou jevy těchto náhodných proměnných. Potom vzorce PMI 4.1 a MI 4.2 jsou následující:

$$PMI(x, y) = \log \frac{P(x \wedge y)}{P(x) \times P(y)} \tag{4.1}$$

$$MI(X, Y) = \sum_{y \in Y} \sum_{x \in X} P(x \wedge y) \times \log \frac{P(x \wedge y)}{P(x) \times P(y)} \tag{4.2}$$

V rámci zpracování přirozeného jazyka při určování podobnosti slov lze uvažovat náhodnou proměnnou jako slovník sémantického modelu a jednotlivé jevy potom udávají výskyt daných slov. Podobnost dvou určitých slov je potom ohodnocena pomocí PMI, kde výsledná hodnota záleží na vztahu společné pravděpodobnosti výskytu obou slov v kontextu v poměru s pravděpodobností výskytu jednotlivých slov nezávisle. Matici s aproximovanými hodnotami PMI implicitně faktorizuje i prediktivní model Word2vec (kapitola 4.2.1).

Pointwise Mutual Information lze podle článku [22] vypočítat za použití následujícího vzorce 4.3. Do použité rovnice lze jednoduše doplnit data z matice spoluvýskytů, kde x a y jsou dvě daná slova, u kterých se zjišťuje podobnost.

$$PMI(x, y) \approx \log \frac{A \times N}{(A + C) \times (A + B)} \tag{4.3}$$

- A: počet kolikrát se spolu slova x a y vyskytnou
- B: počet kolikrát se slovo x vyskytne bez slova y
- C: počet kolikrát se slovo y vyskytne bez slova x
- Výskyty pro hodnoty A, B a C jsou vztaženy na daný kontext (věta, odstavec, dokument...). N potom bude počet všech kontextů v korpusu.

Rovnice je již upravená. V původním tvaru jsou pravděpodobnosti výskytů vyjádřeny takto:

$$P(x \wedge y) = \frac{A}{N} \quad P(x) = \frac{A + B}{N} \quad P(y) = \frac{A + C}{N} \tag{4.4}$$

Čím vyšší je vypočítaná hodnota pro dva výrazy, tím více jsou na sobě závislé, neboli by se dalo říci, že je mezi nimi sémantická asociace. Pro dva výrazy, které se spolu ani jednou nevyskytnou v kontextu, bude hodnota před logaritmizací 0. Protože logaritmus není definován v nulové hodnotě, nelze výpočet provést. Můžeme ovšem těmto dvěma nesouvisejícím výrazům přiřadit výsledek $-\infty$, jelikož funkce logaritmus, která má asymptotu osu y, se blíží k $-\infty$. Výsledek se logaritmizuje, aby některé hodnoty nebyly příliš vysoké a rozdíl mezi nižšími hodnotami byl znatelnější.

Positive Pointwise Mutual Information

Positive Pointwise Mutual Information (PPMI) je úprava původní PMI, kde se jednoduše záporné hodnoty nahradí nulou. Protože PMI má obor hodnot od $-\infty$ do ∞, nemusí být vždy lehké se zápornými hodnotami dále pracovat. Dojde zde ale ke ztrátě informací o podobnosti všech slov, které mají menší pravděpodobnost spoluvýskytu než kombinace nezávislého výskytu.

$$PPMI(x,y) = \begin{cases} PMI(x,y) & PMI(x,y) \geq 0 \\ 0 & PMI(x,y) < 0 \end{cases} \tag{4.5}$$

Normalized Pointwise Mutual Information

Rovnici pro výpočet PMI 4.1 lze postupným upravováním převést do tvaru:

$$PMI(x,y) = \log P(x \wedge y) - \log P(x) - \log P(y) \tag{4.6}$$

Pokud by se daná dvě slova vyskytla pouze spolu, pravděpodobnost výskytu (závislého nebo nezávislého) jednoho slova by se rovnala pravděpodobnosti výskytu druhého slova, což by se zároveň rovnalo i pravděpodobnosti spoluvýskytu obou slov. To znamená, že $P(x \wedge y) = P(x) = P(y)$. S použitím pro tento případ by se odvozená rovnice dala zkrátit a upravit na $PMI(x,y) = -\log P(x \wedge y)$. Toto nabízí několik způsobů normalizací hodnot. Normalizace by mohla i zredukovat tendenci příliš vysoké hodnoty PMI pro málo frekventovaná slova, jejichž pravděpodobnost výskytu je nízká, čímž se výrazně snižuje hodnota ve jmenovateli. [3]

$$NPMI(x,y) = \frac{\log \frac{P(x \wedge y)}{P(x) \times P(y)}}{-\log P(x \wedge y)} \tag{4.7}$$

Rovnice 4.7 normalizuje hodnoty PMI do intervalu $[-1, 1]$. Dvě slova, která se vyskytují pouze ve vzájemném kontextu, budou mít hodnotu 1. Slova, která se spolu ani jednou nevyskytnou, budou mít hodnotu -1. Výpočet NPMI s pomocí matice spoluvýskytu lze provést následovně:

$$NPMI(x,y) \approx \frac{\log \frac{A \times N}{(A+C) \times (A+B)}}{-\log \frac{A}{N}} \tag{4.8}$$

4.1.2 χ^2 statistic

Při získávání podobnosti slov lze použít χ^2 statistic (česky chí kvadrát), což je metoda, která měří míru nezávislosti dvou proměnných přirovnávaných k χ^2 pravděpodobnostnímu rozložení o jednom stupni volnosti. [22]

Výpočet podobnosti dvou slov x a y lze vypočítat tímto vzorcem:

$$\chi^2(x,y) = \frac{N \times (A \times D - C \times B)^2}{(A+C) \times (B+D) \times (A+B) \times (C+D)} \tag{4.9}$$

- A: počet kolikrát se spolu slova x a y vyskytnou
- B: počet kolikrát se slovo x vyskytne bez slova y
- C: počet kolikrát se slovo y vyskytne bez slova x
- N: počet všech kontextů
- D: počet kolikrát se ani jedno slovo nevyskytne. D lze odvodit: $D = N - A - B - C$ [11]

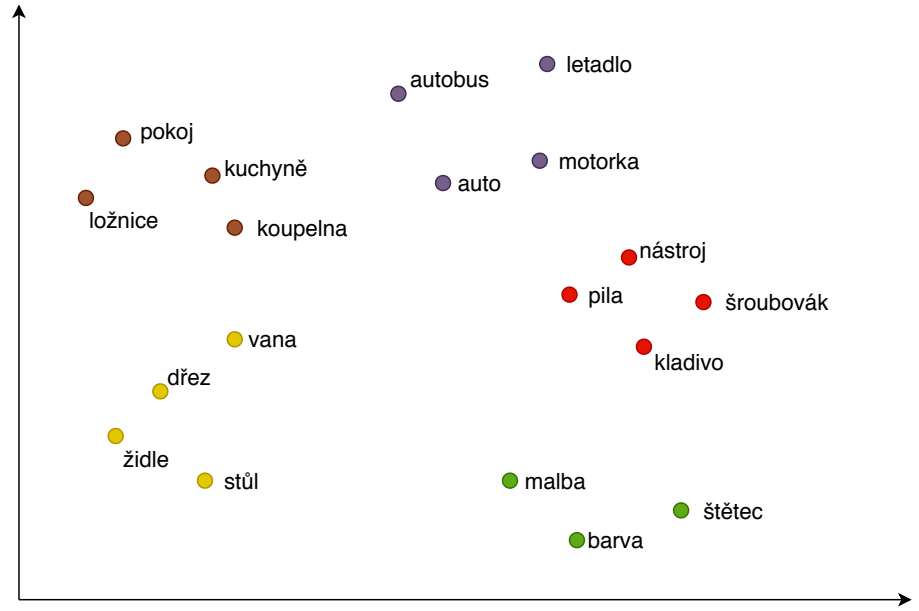
Kde opět platí, čím vyšší hodnota, tím více si jsou dvě slova podobná. Hlavní rozdíl mezi PMI a χ^2 je, že pokud se nějaké slovo vyskytuje v korpusu málokrát oproti jiným slovům, tak výskyt slov už nelze tak dobře přirovnat k χ^2 pravděpodobnostnímu rozložení. Proto χ^2 statictics není spolehlivé pro slova s malým počtem výskytů. [22]

4.2 Prediktivní modely

Prediktivní modely jsou relativně nové. Na rozdíl od modelů založených na počtu, jsou prediktivní modely založeny na principu trénování neuronové sítě, kde jejich původní účel je predikovat slovo nebo slova na základě ostatních slov v kontextu. Zjištění, že schopnost vyhodnocení podobnosti slov je bráno jako vedlejší efekt. Slova jsou v modelu reprezentována jako N-dimenzionální hustý vektor. Proto se často pro označení používá anglický název *word embeddings*, neboli že jsou jednotlivá slova vkládána jako vektor do N-dimenzionálního prostoru.

auto	[0,54; 0,2; -1,41; . . .]
motorka	[0,67; 0,31; -0,8; . . .]
postel	[-0,1; 0,35; 0,71; . . .]

Tabulka 4.2: Příklad vektorové reprezentace slov



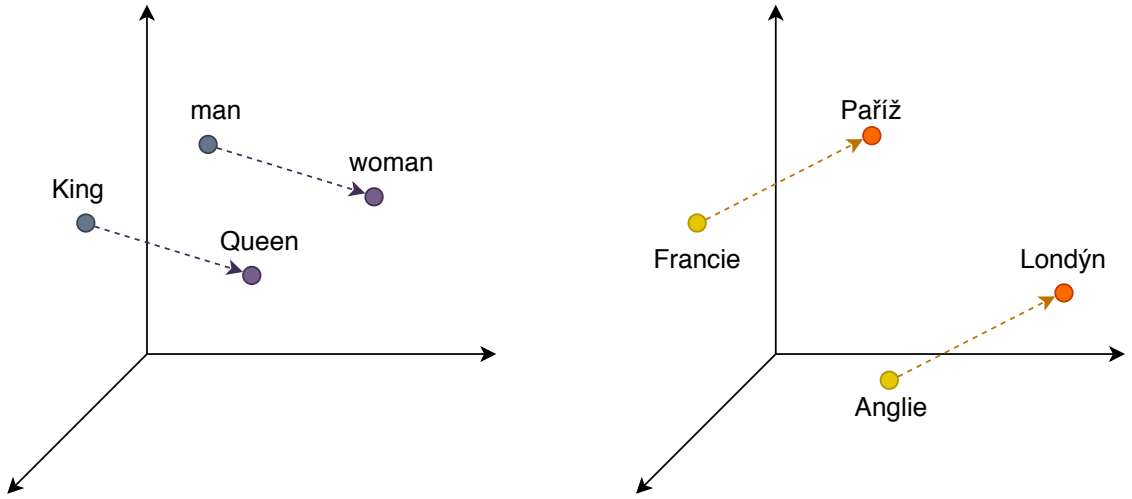
Obrázek 4.1: Příklad vektorové reprezentace slov ve dvou dimenzích

Jednotlivé dimenze těchto vektorů nemají žádný člověku rozpoznatelný význam. Neplatí, že by jedna konkrétní dimenze odpovídala samostatné kategorii, jako je jídlo, město nebo pohlaví. Určování takovýchto vlastností lze docílit použitím vektorové aritmetiky na celé vektory slov. Preditkivní modely jsou známé rovnicí [12]:

$$vektor(\text{„King“}) - vektor(\text{„man“}) + vektor(\text{„woman“}) \approx vektor(\text{„Queen“}) \tag{4.10}$$

Když se vezme vektor slova „King“, odečte se vektor slova „man“ a přičte vektor slova „woman“, výsledek by měl být hodně podobný vektoru slova „Queen“. Platí zde, že rozdíl vektorů slov „King“ a „man“ je velmi podobný jako rozdíl vektorů „Queen“ a „woman“. Díky této vlastnosti natrénovaných vektorů lze vektorově vyjádřit například vztah mezi zemí a jejím hlavním městem.

$$\begin{aligned} vektor(\text{„Londýn“}) - vektor(\text{„Anglie“}) &= X \\ vektor(\text{„Francie“}) + X &\approx vektor(\text{„Paříž“}) \end{aligned} \tag{4.11}$$



Obrázek 4.2: Příklad vektorové reprezentace slov ve třech dimenzích

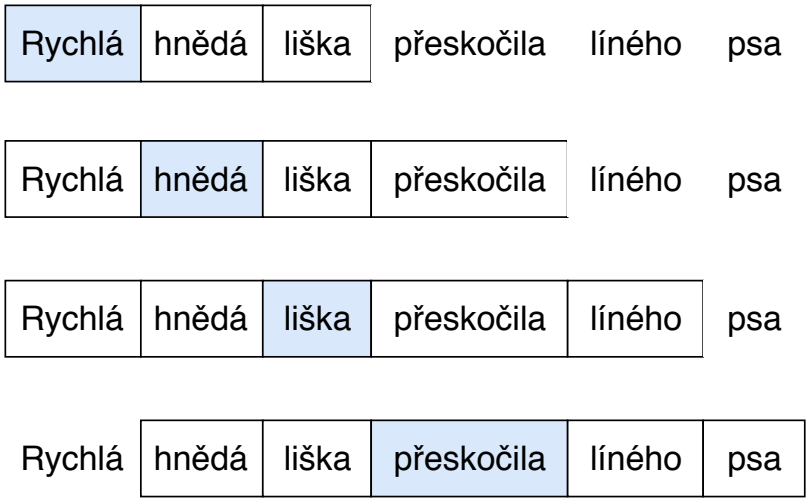
Pro vyhodnocení podobnosti dvou vektorů slov se nejčastěji používá kosinová vzdálenost. Ta definuje kosinus úhlu, který svírají dané dva vektory, z čehož vyplývá, že obor hodnot podobnosti je interval od -1 do 1. Čím více jsou si dvě slova podobná, tím více budou mít jejich vektory tendenci směřovat stejným směrem a jejich kosinová vzdálenost se bude blížit k 1. Naopak vektory dvou naprosto odlišných slov budou směřovat opačnými směry a jejich kosinová vzdálenost se bude blížit k -1. Pro vektory x a y, kde n je počet dimenzí vektorů, je výpočet následující:

$$\cos\theta = \frac{\vec{x}\cdot\vec{y}}{\|\vec{x}\|\|\vec{y}\|} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}} \tag{4.12}$$

4.2.1 Word2vec

Word2vec je skupina prediktivních modelů založena na trénování neuronové sítě. Trénování funguje na principu predikce slova nebo slov v daném kontextovém okně. Natrénovaná síť se již dále na predikci nepoužívá, pouze se využijí natrénované váhy projekční vrstvy sítě. Tyto váhy dokáží reprezentovat jednotlivá slova jako vektory, které nesou sémantický význam daných slov. Architektura modelů Word2vec původně výchází z jazykového modelu dopředné neuronové síte (NNLM, Feedforward Neural Net Language Model) [12]. Architektura NNLM se skládá ze vstupní, projekční, skryté a výstupní vrstvy. Word2vec ale nepoužívá skrytou vrstvu, takže se skládá pouze ze vstupní, projekční a výstupní vrstvy.

Velmi zjednonušeně řečeno, při trénování se prochází vstupním textovým korpusem, kde každá věta je zpracována slovo po slově s předem určenou velikostí kontextového okna. Středem okna je zpracovávané slovo, které se označuje termínem pivot. Ostatní slova v okně jsou označována jako kontextová slova. V závislosti na slovech vyskytujících se v daném kontexu, se upraví jejich vektorové váhy, tak aby výstup neuronové sítě odpovídal pravděpodobností s jakou se všechna slova vyskytnou v kontextu se vstupním slovem. Při každé iteraci se kontextové okno posune o jedno slovo. Celým textovým korpusem se prochází několikrát.



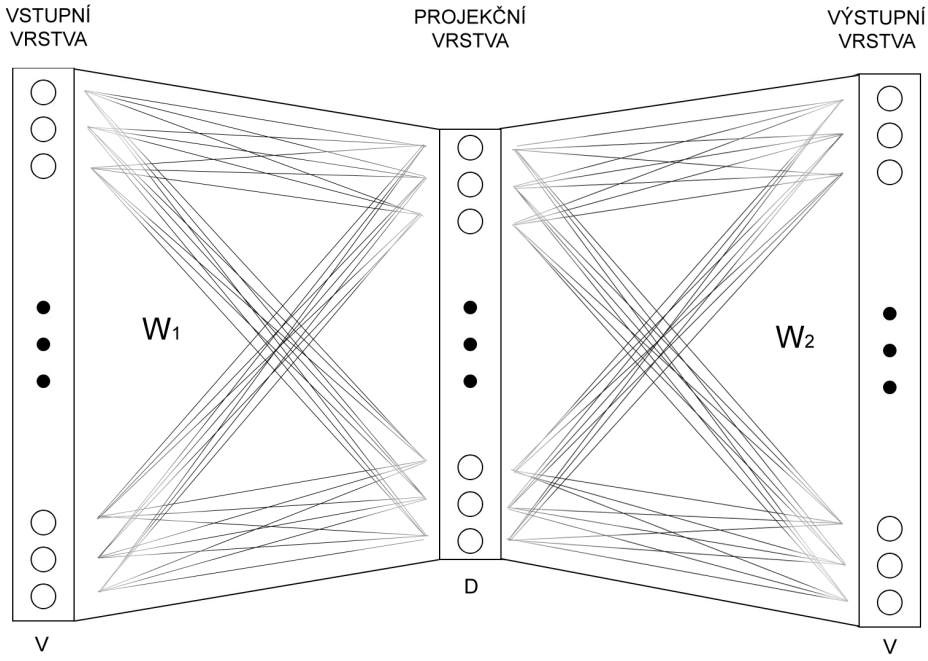
Obrázek 4.3: Příklad posouvání kontextového okna o velikosti 5 slov

Vstupní vrstva má velikost slovníku, kde kažný neuron reprezentuje právě jedno dané vstupní slovo. Vstup je tedy vektor o velikosti slovníku, kde pro konktrétní vstupní slovo je jeho index vektoru nastaven na hodnotu 1. Zbytek hodnot je nastaven na 0. Anglicky se tomuto vektoru říká *one-hot encoding* [18].

i_1	i_2	i_3	...	i_V
0	1	0	...	0

Tabulka 4.3: Demonstrace vstupní vrstvy neuronové sítě

Projekční vrstva má velikost rovnu počtu dimenzí jednotlivých vektorů. Tento parametr se nastaví dopředu před vlastním trénováním. Mějme vstupní vrstvu o velikosti slovníku V a počet dimenzí D. Potom matice $W_1 = V \times D$ představuje matici vektorových vah projekční vrstvy, kde jednotlivé vektory reprezentují sémantickou podobnost slov. Při použití one-hot encoding vektoru, který se vynásobí maticí W_1, bude výsledkem vektor v reprezentující právě jedno dané vstupní slovo.



Obrázek 4.4: Zjednodušené schéma architektury neuronové sítě

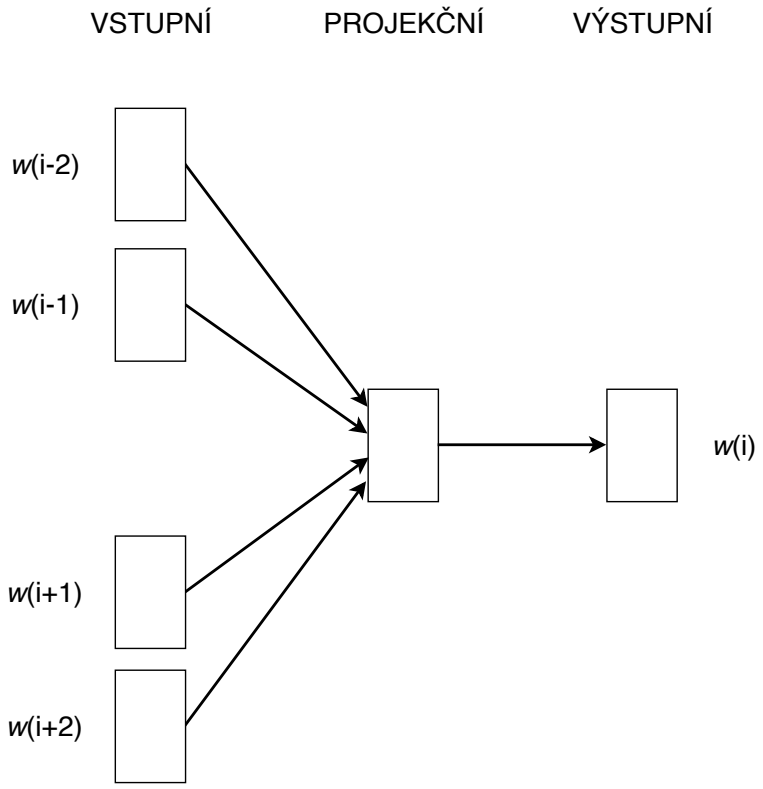
Výstupní vrstva má opět velikost slovníku. V matici W_2 o velikosti $D \times V$ potom jednotlivé sloupce představují vektory kontextových slov. Uvažujme příklad, kdy se zpracovává jedno vstupní slovo i a jedno kontextové slovo j. Z matice W_1 je vybrán i-tý řádek, což je vektor v. Vektor v' potom bude j-tý sloupec z matice W_2. Skalárním součinem ${v'}^T v$ získámé hodnotu spoluvýskytu v kontextu. Tato hodnota ale není pravděpodobností. Pokud tedy chceme vypočítat pravděpodobnost, že se slovo j vyskytne v kontextu slova i: $p(j|i)$, je potřeba výslednou hodnotu převést do pravděpodobnostního rozložení. K tomu slouží funkce softmax. K určení pravděpodobnosti dané hodnoty je ale potřeba znát soubor všech ostatních hodnot. Proto se skalární součin musí realizovat se všemi vektory z matice W_2.

$$p(j|i) = \frac{\exp({v'_j}^T v_i)}{\sum_{k=1}^{V} \exp({v'_k}^T v_i)} \tag{4.13}$$

Výstupem neuronové sítě je tedy vektor o velikosti slovníku, kde jednotlivé hodnoty uvádí pravděpodobnost výskytu všech slov ze slovníku v rámci kontextu vstupního slova. V posledním kroku se spočítá chyba oproti cíli a pomocí zpětné propagace se upraví váhy matic W_1 a W_2. Před začátkem trénování jsou tyto váhy nastavené na malé náhodné hodnoty. Jak již bylo zmíněno, výstup neuronové sítě se po natrénování modelu dále nepoužívá. Pro sémantickou reprezentaci slov slouží výsledné váhy matice W_1. Použití matice W_1 pro vstupní slova a matice W_2 pro kontextová slova při trénování, místo použití jedné matice pro vše, umožňuje lépe určovat pravděpodobnost spoluvýskytu. V případě $i = j$ by použití stejné matice pro výpočet neodpovídalo náležité pravděpodobnosti. Protože dané slovo se zpravidla nevyskytuje v kontextu sebe samotného, tak potom budou výsledné vahy matic W_1 a W_2 upraveny tak, aby se pravěpodobnost spoluvýskytu stejného slova blížila k 0.

Continuous Bag-of-Words Model

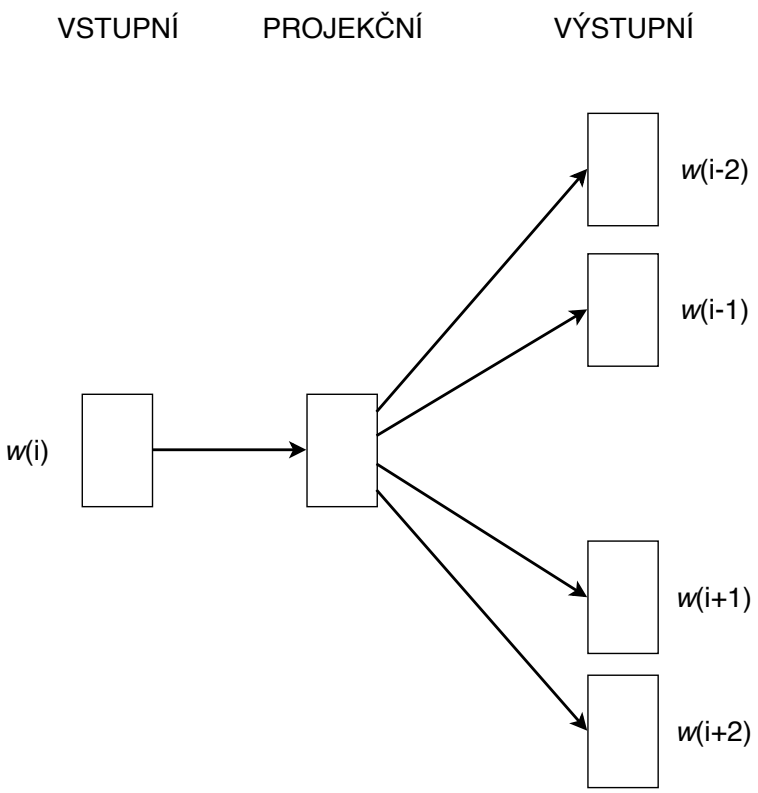
Continuous Bag-of-Words Model (CBOW) je první typ architektury modelu Word2vec. Princip spočívá v tom, že všechna kontextová slova jsou brána jako vstupní a predikuje se zpracovávané slovo. Místo ale použití jednotlivých vektorů vstupních slov pro predikci se vstupní vektory z matice W_1 zprůměrují. Výstupem projekční vrstvy je tedy jeden zprůměrovaný vektor.



Obrázek 4.5: Schéma architektury CBOW

Skip-gram Model

Architektura Skip-gram je přesným opakem CBOW. Zpracovávané slovo je vstupem do neuronové sítě a predikují se všechna kontextová slova. Výpočet pravěpodobnosti spoluvýskytu se provede pro každou dvojici vstupní slovo a kontextové slovo zvlášť. Při následném upravování váh se jako celková chyba oproti předpokladu použije součet predikčních chyb každé dvojice.



Obrázek 4.6: Schéma architektury Skip-gram

Hierarchický softmax

Výpočet funkce softmax, kde se při každé iteraci provádí skalární součin se všemi vektory slovníku, je výpočetně velice náročný. Proto je potřeba použít metodu, která aproximuje funkci softmax a je výpočetně mnohem méně náročná. Model Word2vec má implementovány dvě metody: hierarchický softmax a negativní vzorkování.

Hierarchický softmax využívá binární strom, kde koncové uzly představují jednotlivá slova ze slovníku. Ke každému koncovému uzlu existuje unikátní cesta z kořene stromu. Tato cesta představuje pravěpodobnost predikce daného slova reprezentovaného koncovým uzlem. Celková pravěpodobnost je potom součin pravděpodobností každého uzlu, že se vydá doleva nebo doprava. Matice W_2 bude místo vah všech kontextových slov představovat váhy všech vnitřních uzlů stormu. Složitost výpočtu se tímto sníží z $O(V)$ na $O(\log(V))$. [18]

Negativní vzorkování

Při trénování neuronové sítě se lehce upravují všechny váhy matice W_2. Negativní vzorkování (negative sampling) je postup, při kterém se upravuje pouze malé procento všech váh. Dvojice vstupní slovo a kontextové slovo se použijí jako pozitivní vzorky. Dále se vybere N náhodných slov ze slovníku. Tato vybraná slova budou použita jako negativní vzorky. Při upravování váh se váhy pozitivních vzorků upraví, aby výsledná pravděpodobnost se blížila k 1. Váhy negativních slov ze naopak upraví tak, aby se pravděpodobnost blížila k 0. Tímto způsobem se tedy upraví pouze váhy použitých vzorků oproti běžnému upravování všech váh, což výrazně sníží časovou náročnost trénování.

Aproximace PMI

Podle článku [10] model Word2vec s architekturou Skip-gram a použitím negativního vzorkování implicitně aproximuje hodnoty Pointwise Mutual Information (rovnice 4.1). Vynásobením matic W_1 a W_2 získáme matici M, která má rozměry $V \times V$. Prvky této matice udávají hodnotu PMI pro všechny dvojice slov. Jelikož jsou reprezentace slov v maticích W_1 a W_2 omezené na D dimenzích, tak hodnota PMI nebude přesná, pouze se jí bude blížit. Z toho vyplývá, že matice W_1 a W_2 faktorizují[1] matici s aproximovanými hodnotami PMI.

4.2.2 FastText

FastText je upravená verze modelu Word2vec. Liší se v tom, že nebere slova jako atomické jednotky, ale vytváří si reprezentace unikátních podřetězců slov. Díky tomuto může lépe zachytit vazby mezi různými lexikálními derivacemi (odvození tvaru slova). Proto by měl být FastText obecně úspěšnější pro morfologicky bohaté jazyky.

Jednotlivá slova jsou rozdělená do n-tic (n-grams) písmen, které mají délku 3 až 6 znaků. K nim se následně přidá i samotné slovo. Každá n-tice je reprezentována vlastním vektorem. Suma všech vektorů n-tic potom představuje výslednou reprezentaci daného slova. Díky této vlastnosti je FastText schopen odhadnout vektorovou reprezentaci slova, které nemá ve slovníku. Výsledné n-tice slova „papír“ by vypadaly následnovně:

$$< pap,\ api,\ pír,\ papí,\ apír >< papír >$$

[1] Faktorizace matice znamená rozklad matice na dvě matice s menším počtem dimenzí. Výnasobením těchto matic potom znovu vznikne původní matice.

Kapitola 5

Návrh a implementace

V této kapitole je popsán návrh a implementace jednotlivých částí výsledného sytému. Jsou zde popsány sémantické modely, implementovaný člen operativy a hlavní špión a také webová služba zprostředkující rozhraní hry. Dále jsou zde uvedeny použité knihovny a jejich účel.

5.1 Použité technologie

Výsledný systém byl vytvořen v jazyce Python. Python je vysokoúrovňový interpretovaný programovací jazyk. Python poskytuje dynamické typování a automatickou správu paměti. Podporuje několik programovacích paradigmat, včetně objektově orientovaného programování.[1] Jeho výhodou je velké množství dostupných knihoven i v rámci zpracování přirozeného jazyka.

Morfologická analýza českého jazyka je prováděna pomocí knihovny MorphoDiTa[2]. Tato knihovna pracuje s morfologickým slovníkem MorfFlex CZ [20]. Konkrétněji je využita pro lemmatizaci textového korpusu a použitých nápověd v rámci hry. Při nejednoznačném lemmatu nápovědy je dále knihovna využita pro part-of-speech označení. Toto označení poskytuje lexikální informace o vstupním slově. Především jsou využity informace o slovním druhu a jeho mluvnickém pádu. Pro podporu anglického jazyka byly použity knihovny NLTK[3] a spaCy[4], které poskytují nástroje pro stemming, lemmatizaci a další.

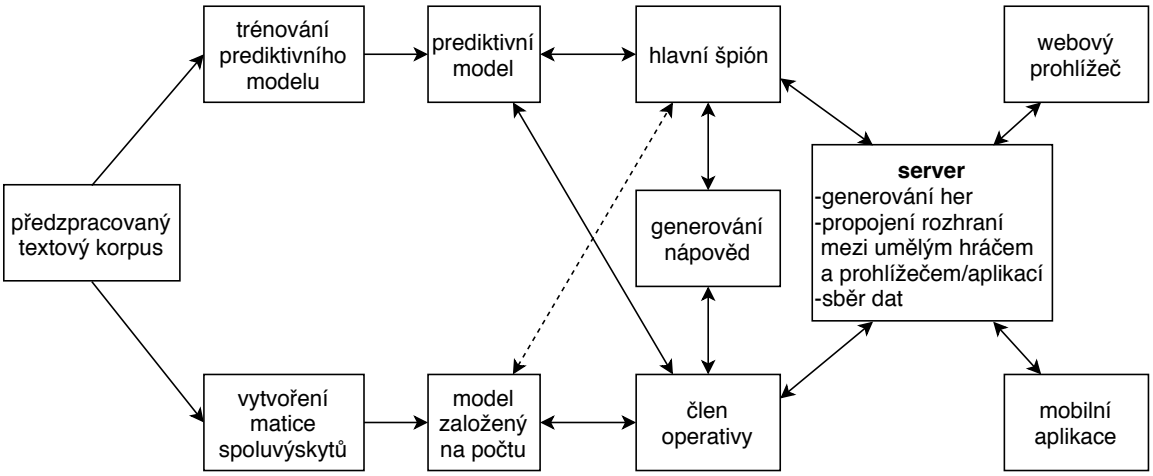
Pro práci s prediktivními modely je použita knihovna gensim [17]. Jedná se o komplexní nástroj umožňující zpracovávání podobnosti slov a celých dokumentů. Obsahuje implementaci řady modelů, jako word2vec, fastText, latent Dirichlet allocation a další. Mezi jednu z výhod patří i schopnost zpracovávat velké kolekce textových dat. Další použitou knihovnou je fastText[5]. Jedná se o rozhraní v pythonu oficiální implementace modelu fastText v jazyce C++ vytvořený skupinou Facebook Research.

Dokumentace zdrojových kódů byla vygenerována nástrojem sphinx[6].

[1] https://en.wikipedia.org/wiki/Python_(programming_language)
[2] http://ufal.mff.cuni.cz/morphodita
[3] https://www.nltk.org/
[4] https://spacy.io/
[5] https://github.com/salestock/fastText.py
[6] http://www.sphinx-doc.org/en/master/

5.2 Architektura systému

Výsledný systém lze rozdělit na tři hlavní části. V první části se natrénují sémantické modely. Ve výsledném systému jsou použity dva modely, jeden založený na počtu a jeden prediktivní. Kombinace těchto modelů je následně využita v implementaci člena operativy a hlavního špióna. Nakonec serverová aplikace propojuje rozhraní mezi umělým hráčem a webovým prohlížečem nebo mobilní aplikací.



Obrázek 5.1: Architektura vytovřeného systému

5.3 Sémantické modely

Před vytvořením sémantických modelů bylo potřeba předzpracovat textový korpus. K tomu je vytvořen skript *preprocess.py*, který pomocí knihovy MorphoDiTa provede lemmatizaci každého slova, odfiltruje interpunkci a nepotřebná slova. Lemmatizované věty jsou postupně zapsány do souboru, každá na vlastní řádek. Jelikož je lemmatizace na větším korpusu časově náročná, použil jsem již lemmatizované korpusy z předchozích prací.

5.3.1 Použité korpusy

V rámci této práce byly použity 3 korpusy. Prvním je CWC-2011 [19]. Jedná se o kolekci zpráv, článků, blogů a dalších literárních celků. Dalším korpusem je seznam2017, který je složen z velké kolekce webových stránek. Jeho výhodou je, že obsahuje mnohem větší množství informací, které jsou zároveň aktuálnější, než v korpusu CWC-2011. Naopak nevýhodu je, že není příliš vyvážený, protože obsahuje například stránky různých e-shopů, kde vytažený text neudává celé smysluplné věty. Pro další váhování podobnosti slov je použitý Slovník spisovné češtiny, který obsahuje definice více jak 28 tisíc pojmů.

korpus	**unikátní slova**	**počet vět**
CWC-2011	957 728	218 653 428
seznam2017	8 697 736	3 570 553 790

Tabulka 5.1: Velikost korpusů po předzpracování

5.3.2 Model založený na počtu

Jádrem modelů založených na počtu je matice spoluvýskytů, jejíž vytvoření je provedeno skriptem *create_cooccurrence.py*. Nejprve se ze vstupního korpusu pomocí knihovny gensim vytvoří slovník unikátních slov, která podléhají pravidlům jako minimální počet výskytů nebo výskyt v seznamu nepotřebných slov. Následně se objektem CountVectorizer z knihovny scikit-learn[7] vytvoří řídká matice, kde řádky představují jednotlivé věty a sloupce unikátní slova z vytvořeného slovníku. Daný prvek matice tedy udáva kolikrát se v dané větě vyskytlo určité slovo. Nechť matice $M_{v\times s}$ představuje vytvořenou matici [věty : slova]. Provedením následující operace

$$M^T M = M'_{s\times s} \tag{5.1}$$

vznikne matice [*slova* : *slova*], kde daný prvek udává, kolikrát se daná dvě slova vyskytla ve stejné větě. Diagonálu matice lze nastavit na nulu, jelikož informace spoluvýskytu stejného slova je zbytečná. Po vytvoření matice spoluvýskytu z korpusu CWC-2011 měl největší prvek 24 bitovou hodnotu. V rámci paměťové optimalizace byly prvky matice nastaveny na 32 bitová celá čísla. Výsledná matice je společně s dalšími podpůrnými informacemi serializována do souboru. Velikost tohoto souboru je 13 GB.

Kvůli velkému počtu unikátních slov a omezení operační paměti nebylo možné vytvořit tímto způsobem matici spoluvýskytů z korpusu seznam2017. Místo vytvoření matice [věty : slova] jsem zkusil rovnou inicializovat prázdnou řídkou matici [slova : slova] a postupně s procházením korupusu inkrementovat jednotlivé hodnoty. Přesto i tak došlo k nedostatku operační paměti. I kdyby se podařilo matici vytvořit, byla by příliš velká pro rozumné použití. Navíc prediktivní model natrénovaný na korpusu seznam2017 měl horší úspěšnost než na CWC-2011.

Objekt třídy *CountModel*, představující model založený na počtu, je instanciován s parametry: jméno souboru uložené matice a vybraná metoda pro vyhodnocování podobnosti dvou slov. Vyhodnocení sémantické podobnosti dvou slov je realizováno zavoláním metody *similarity*().

První implementovanou metodou pro vyhodnocení podobnosti je jednoduchý základní přístup, který vyhodnotí podobnost dvou slov následujícím vzorcem.

$$\text{podobnost} = \frac{\text{počet spoluvýskytů}}{\text{počet výskytů jednoho slova}} \tag{5.2}$$

Tento přísup udává, jak moc je jedno slovo podobné druhému v poměru s ostatními slovy. Pro lepší přehlednost výsledků je hodnota následně logaritmizována. Ačkoliv se jedná o primitivní postup, jeho výsledky nad testovací sadou nebyly vůbec špatné, což ukázalo, že pro Krycí jména by mohly dobře fungovat modely založené na počtu.

Další dvě implementované metody jsou chí kvadrát (rovnice 4.9) a Normalized Pointwise Mutual Information (rovnice 4.7), která dosahovala lepší výsledky než klasická Pointwise Mutual Information.

5.3.3 Prediktivní model

U prediktivních modelů jsem vycházel z práce na téma shlukování slov podle významu [7], kde jsem natrénované modely vyhodnotil na testovací sadě. V tabulce 5.2 jsou uvedeny výsledky, kde výsledek každého modelu představuje jeho nejúspěšnější kombinaci parametrů.

[7] https://scikit-learn.org/...CountVectorizer.html

U modelů Word2vec a fastTest se jedná o architekturu Skip-gram s použitím negativního vzorkování. Modely byly trénovány na korpusech CWC-2011, All.vert a Wikipedie. Všechny modely měly nejvyšší úspěšnost na korpusu CWC-2011.

model	**úspěšnost**
fastText	74,14%
Word2vec	71,87%
GloVe	68,76%

Tabulka 5.2: Vyhodnocení modelů z referenční práce

V rámci této práce byl dále využitý nejúspěšnější model fastText. Pro natrénování fastTextu jsou vytvořeny dva skripty. Skript *trainFasttext.py* trénuje model pomocí knihovny Facebook knihovny fastText a skript *trainFasttext_gensim.py* trénuje s využitím knihovny gensim. Oba zpbůsoby trénování uloží natrénovaný model jak v binárním formátu, tak i vektory jednotlivých slov v textovém formátu. Oba typy souborů lze pro vyhodnocování podobnosti slov načíst knihovnou gensim. Výhodou textového formátu je menší paměťová náročnost. V textovém formátu ale nejsou uloženy vektory n-tic jednotlivých slov, proto s jeho použitím nelze aproximovat vektor neznámého slova. Nejdůležitější roli hraje metoda *similarity*(), která počítá kosinovou vzdálenost dvou vektorů slov a vrátí její normalizovanou hodnotu v intervalu $[0, 1]$.

5.3.4 Vyhodnocení modelů

Vyhodnocení modelů bylo provedeno pomocí skriptu dostupného v repoztáři github[8], který vyhodnocuje prediktivní modely. Skript jsem upravil pro možnost vyhodnocení modelů založených na počtu a případně jejich kombinací. Testovací sada se skládá ze 680 zaznamenaných tahů realných her s případnými zamýšlenými slovy pro nápovědu. Pokud model vyhodnotí, jako nejpodobnější slova ta, která byla ve hře hádána, nebo slova, jež byla zamýšlena nápovědou, je vyhodnocení pro daný tah 100%. Úspěšnost klesá s tím, jak moc jsou cílená slova byla daleko v pořadí vyhodnoceném daným modelem.

model	**úspěšnost**
fastText (Facebook) + aproximace OOV	74,99%
fastText (Facebook)	74,46%
fastText (Facebook) (korpus seznam2017)	74,00%
fastText (gensim)	72,98%
NPMI	76,79%
PMI modifikace	76,43%
PMI	76,08%
poměr spoluvýskytů	71,69%
vážený počet spoluvýskytů	68,70%
chí kvadrát	36,94%
optimální výběr z NPMI & fastText	81,33%

Tabulka 5.3: Vyhodnocení sémantických modelů

[8] https://github.com/MFajcik/NLP-FIT

V tabulce 5.3 jsou všechny výsledky, kromě jedné hodnoty, vyhodnoceny z modelů natrénovaných na korpusu CWC-2011. Aproximace OOV (out-of-vocabulary) znamená použití aproximovaného vektoru slova, které není ve slovníku modelu. Pokud není nápověda daného tahu ve slovníku, standardně je daný tah ohodnocen nulovou úspěšností. U modifikace PMI je pro celkový počet výskytu daného slova použita suma celého řádku matice místo skutečného počtu výskytů. Při vyhodnocení tento způsob dosáhl lepšího výsledku. Při normalizaci tento způsob už nefunguje tak dobře, protože neodpovídá přímo hodnotě PMI, proto je pro výpočet NPMI je proto použitý skutečný počet výskytů slova.

Vážený počet spoluvýskytů udává celkový počet spoluvýskytů daných dvou slov, kde každý spoluvýskyt je vážený vzdáleností (počet slov) mezi těmito slovy v kontextu (rovnice 5.3). Tato metoda byla v kombinaci v modelem fastText použita v předchozí práci [15].

$$\text{vážený počet spoluvýskytů}(x, y) = \sum \frac{1}{\text{vzdálenost}(x, y)} \tag{5.3}$$

Nejlepší výslekdy vykazují modely NPMI a fastText. Při následném výběru úspěšnějšího modelu pro danou nápovědu by teoreticky mohla být výsledná úspěšnost 81,33%. Metoda NPMI byla úspěšnější ve 40,75% případů, fastText ve 29,70% případů a u 29,55% případů měly modely stejnou úspěšnost. To znamená že vhodnou kombinací těchto modelů by se mohlo docílit celkově lepšího výsledku. Výsledná kombinace je popsána v podkapitole 5.4.2

5.4 Člen operativy

Jak u hlavního špióna, tak i člena operativy, je možno nastavit, zda má používat herní strategii nebo nikoliv. V případě použití herní strategie obě tyto role využívají kontextovou informaci z předchozích tahů k ovlivnění aktuálně hraného tahu. Bezkontextovou variantu hráče lze využít například k neovlivněnému vyhodnocování sémantických modelů nebo jako poradce pro tah v mobilní aplikaci.

Člen operativy je implementován třídou *Operative* ve skriptu *operative.py*. Jeho úkolem je vyhodnotit podobnost slov ve hře s příchozí nápovědou a vybrat slova, která se v daném tahu označí. K vyhodnocení využívá kombinaci prediktivního modelu fastText a modelu založeném na počtu NPMI. Bez použití strategie si seřadí slova ve hře podle vyhodnocené podobnosti s nápovědou a vybere k označení N prvních slov, kde N je číslo nápovědy.

5.4.1 Lemmatizace nápovědy

Pro vyšší úspěšnost vyhodnocení je prováděna lemmatizace nápovědy, neboť sémantické modely jsou natrénované z lemmat slov. Lemmatizace je prováděná pomocí knihovny MorphoDiTa, která u každého lemmatu vrací i Part-of-speech tag vstupního slova. Part-of-speech (POS) tagging je forma označení daného slova, kde je uvedeno v jakém je slovním tvaru. Jedná se o informace jako slovní druh, jednotné/množné číslo, čas, rod, pád a další. Lemmatizace ovšem nemusí být vždy jednoznačná. Například slovo „čísla“ může být množné číslo základního tvaru „číslo“, nebo se může jednat o sloveso v minulém času slova „čísnout“. Pro vyhodnocení podobnosti se slovy ve hře jsou proto použity všechny vyhodnocená lemmata kromě několika pravidel.

Pokud by nápověda v daném tvaru byla použita jak v prvním pádu i v jiném pádu, bude k vyhodnocení použita pouze lemma připadající pro první pád. Například slovo „triky“ je morfologickou analýzou vyhodoceno jako první pád množného čísla slova „trik“ a také jako

sedmý pád množného čísla slova „triko“. Po zavedení tohoto pravidla bude použita pouze lemma „trik“.

Pokud by se jednalo o sloveso a zároveň by mohlo jít i o jiný slovní druh, musí mít sloveso přiřazený čas. Například slovo „stroj“ může být podstatné jméno v základním tvaru, nebo se může jednat o sloveso v rozkazovacím způsobu slova „strojit“. Takto bude použita jenom lemma „stroj“.

Pokud se jedná o sloveso s přiřazeným časem, které opět má stejný tvar jako významově jiné slovo, použité pravidlo vyhodnotí, zda se lemma jiného slovního druhu je stejné jako vstupní slovo, což následně povede k rozhodnutí, která lemma se použije. Toto pravidlo demonstrují následující příklady. Například slovo „teče“ může být podstatené jméno „teč“ nebo sloveso „téci“. Protože „teče“ není stejné jako „teč“, bude použita lemma „téci“. Druhý příklad je slovo „zahraničí“, což může být podstatné jméno nebo sloveso v přítomném čase od slova „zahraničit“. Zde je lemma podstatného jména stejná jako vstupní slovo, takže je použita pouze lemma „zahraničí“.

Jelikož morfologická analýza nefunguje se stoprocentní úspěšnostní, může nastat případ, že vyhodnocená lemma nebude mít v českém jazyce významový smysl.

V případě, že se jedná o vlastnost chrakterizovanou podstatným jménem ženského rodu končící na „ost“, dojde k záměně koncovky „ost“ na „ý“. Například místo slova „blízkost“ se použije přídavné jméno „blízký“. Tato teze však neplatí pro určitá slova, například: „událost“, „kost“, „radost“... Tato slova jsou pevně dána, aby nebyla modifikována tímto pravidlem.

Jestliže těmito pravidly projde více lemmat jednoho vstupního slova, jsou pro vyhodnocení použity všechny. Konečnou podobnost nápovědy se slovem bude udávat jeho skóre s nejpodobnějším lemmatem. I když jsou všechna slova ve hře podstatná jména prvního pádu, je prováděná jejich lemmatizace. Například herní slovo „dinosaurus“ má vyhodnocenou lemmu „dinosaur“. Jelikož jsou sémantické modely natrénované na lemmatech slov, neznají slovo „dinosaurus“. Všechná uvedená pravidla nezávazně na sobě zlepšují celkovou úspěšnost nad testovacími daty.

5.4.2 Kombinace modelů

Cílem použití kombinace modelů fastText a NPMI bylo se co nejvíce přiblížit jejich teoreticky maximálnímu potenciálu s úspěšností 81,33%. V tabulce 5.4 jsou uvedeny testované kombinace. Jelikož model fastText vrací hodnotu v intervalu $[0, 1]$ a NPMI má obor hodnot $[-1, 1]$, je hodnota NPMI pomocí operace $(výsledek + 1)/2$ ještě v rámci modelu normalizována na interval $[0, 1]$. Díky této normalizaci lze pohodlněji pracovat s kombinacemi hodnot obou modelů.

kombinace	**úspěšnost**
aritmetický průměr nenulových hodnot	77,31%
aritmetický průměr hodnot	77,24%
postupný výběr nejlepšího slova	76,61%
geometrický průměr hodnot	76,60%
seřazení nenulových hodnot NPMI + zbytek pořadí fastText	76,54%
seřazení podle vyšší hodnoty z obou modelů	76,47%

Tabulka 5.4: Vyhodnocení kombinace modelů fastText a NPMI

V případě aritmetického průměru nenulových hodnot se provede průměr pouze tehdy, pokud je hodnota NPMI nenulová. Někdy se může jednat o podobná slova, která ovšem nemají žádný spoluvýskyt. Potom by zprůměrovaní těchto dvou modelů vedlo k velkému poklesu skóre podobnosti. U těchto případů je jako výsledné skóre použita hodnota modelu fastText. Použití výsledné kombinace záleží na tom, zda se lemma nápovědy vyskytuje ve slovnících obou modelů, nebo pouze v jednom z nich. Pokud je nápověda v obou modelech, použije se aritmetický průměr nenulových hodnot. Pokud se vyskytuje pouze v modelu NPMI, nejprve se seřadí nenulové hodnoty. Aby u nulových hodnot nedocházelo k náhodnému pořadí, je zbytek slov seřazen podle modelu fastText pomocí aproximovaného vektoru neznámého slova. V případě výskytu nápovědy jenom ve fastTextu, použijí se čitě jeho hodnoty. Pokud se nápověda nenachází ani v jednom z modelů, opět se použije fastText s aproximovaným vektorem. Tento výběr kombinací dosahuje celkové úspěšnosti 77,81%.

Pro další váhování podobnosti slov je využit definiční slovník, který je zpracovaný pomocí skriptu *parse_def_dictionary.py*. Ze vstupního textu jsou odstraněny nepotřebné znaky a jednotlivá slova jsou lemmatizována. Slovník je následně uložen do souboru, kde každému pojmu jsou přiřazena unikátní slova, která se vyskytují v jeho definici. Při samotném vyhodnocování podobnosti se zkontroluje, zda se jedno slovo nevyskytuje v definici druhého slova. V případě výskytu se skóre podobnosti zvýší o 0,25. Použití definičního slovníku vedlo k zlepšení úspěšnosti na 77,96%.

5.4.3 Strategie člena operativy

V rámci strategie se člen operativy zaměřuje na pamatování svých předchozích neúspěšných tahů a určování potencionálních nepřátelských slov. Dále se také snaží zvážit, zda by místo obdržené nápovědy nepoužil na cílená slova lepší nápovědu. Což by mohlo znamenat, že obdržená nápověda možná cílí případně jiné slovo.

Váhování možných slov nepřítele

V předchozí práci [15] bylo označování potencionálních nepřátelských slov pevné stylem ano/ne. Já jsem se rozhodl pro dynamičtější přístup pomocí váhování slov, což by mělo lépe reprezentovat míru podezření, že slovo patří nepřátelského týmu. Navíc se zabrání tomu, že by mohlo být slovo jednoznačně označeno jako nepřátelské, byť by patřilo do mého týmu. Na začátku hry jsou váhy slov nastaveny na nulu. Při zadání nepřátelské nápovědy se vyhodnotí podobnost slov stejným způsobem, jako při nápovědě od mého hlavního špióna. Pokud se jedná o běžnou nápovědu s číslem N, vybere se prvních N slov a upraví se jejich váha následujícím vzorcem:

$$w_i = w_i + \frac{\text{sim}(h, i)}{N \cdot \alpha} \tag{5.4}$$

Symbol w_i představuje váhu slova i, $\text{sim}(h, i)$ znamená vypočítaná hodnota podobnosti nápovědy h se slovem i. Parametr α byl nastaven na hodnotu 0,75 pro lepší korelaci mezi počtem cílených slov a jejich výslednou váhou. Váha slov klesá s větším počtem cílených slov, protože při snaze spojit více slov dochází k širší míře abstrakce a je větší šance, že nějaké slovo bude sémantickými modely špatně vyhodnoceno jako cílené. Navíc při zadání nápovědy s vyšším počtem slov může protivník, ať už vědomě nebo nevědomě, cílit jedno ze slov mého týmu, a proto u takovýchto nápověd není vhodné moc zvyšovat váhu potencionálních slov. Naopak třeba u nápovědy cílící jedno slovo je velice jednoduché vytvořit silnou asociaci, a proto se může s velkou jistotou silně zváhovat dané slovo.

Při hádání slov nepřátelským týmem se sleduje, zda protivník neoznačil slovo, které nebylo jejich týmu. Pokud ano, ze seznamu aktuálně váhovaných slov (seřazeného podle vypočítané podobnosti) se vybere tolik slov, kolik jich zbývalo protivníkovi na danou nápovědu ještě označit. Pro tato slova se následně aplikují vypočítané váhy. Při vyhodnocování slov v mém tahu se od vypočítané podobnosti každého slova odečte jeho aktuální váha. To umožní případné označení již váhovaných slov v mém tahu.

Na konci každého herního kola (oba týmy zahrají svůj tah) se sníží váhy všech slov, která nebyla pozitivně váhována v posledním kole, o hodnotu 0,05. Platí, že nejnižší možná hodnota váhy je 0. Pokud se na začátku hry váhováním označí nějaké slovo, ale protivníkův tým se k tomuto slovu už znova nevrací, je možné, že dané slovo není nepřátelského týmu a došlo k jeho chybnému označení. Tímto postupným snižováním váh se zvětší šance, že se dané slovo vezme do úvahy jako cílené v závěrečném stádiu hry.

Pamatování předchozích tahů

Při každém vybírání slov pro vlastní nápovědu se daný tah uloží ve formátu: N označovaných slov, další 2 v pořadí do zálohy a případně slovo, které bylo přidáno na dokončení předchozích tahů. Pokud při označování slov dojde k označení slova, které není z mého týmu a to slovo nebylo přidáno jako dokončování předchozího tahu, uloží se tah jako neúspěšný s určením počtu zbývajících slov. Při zavolání metody pro vrácení slov z nedokončených tahů se vyberou slova v následujícím pořadí:

1. původně zamýšlená slova ze všech nápověd, která nejsou váhována
2. záložní slova, která nejsou váhována
3. ostatní slova seřazená vzestupně podle váhy

Při každém kroku výběru se z každé nedokončené nápovědy bere maximálně tolik slov, kolik na tuto nápovědu zbývá označit. Nedokončená nápověda se odstraní, pokud v seznamu už nejsou zádná uložená slova, nebo zbývající počet slov k uhádnutí se snížil na 0.

U běžné nápovědy se k cíleným slovům přiřadí první slovo ze seřazeného seznamu nedokončených tahů. Pro dostatečné využití pamatovaných slov dochází u nápověd 0 a nekonečno. Nápověda typu 0 slouží k označení slova, kterému by se měl hráč vyhnout. Při obdržení této nápovědy se nejpodobnější slovo zváhuje tak, aby nemohlo dojít k jeho možnému označení, a to až do konce hry. Jako vybraná slova se právě použije zmíněný seřazený seznam slov z nedokončených tahů. Pokud nejsou uložená žádná slova z nedokončených tahů nebo není nastavené použití strategie, použije se jedno slovo, které je nejméně podobné dané nápovědě.

Nápověda typu nekončeno také dovoluje neomezený počet hádaných slov, ale navíc cílí další slovo nebo slova, která by měla být uhádnuta. Opět se použije seznam neuhádnutých slov, plus se navíc přidá jedno nejpodobnější slovo nápovědě (pokud už není v seznamu). Agresivnější přístup nastává, když protivníkovi zbývá uhádnout už jenom poslední dvě slova. V tom případě se místo pouze nejpodobnějšího slova postupně přidávají další slova, u kterých platí, že jejich skóre podobnosti je bližší k předchozímu slovu v pořadí, než ke slovu následujícímu. Pokud protivníkovi zbývá už jenom jedno slovo, je prakticky jasné, že dalším tahem vyhraje hru. Potom se k neuhádnutým slovům použije tolik slov, aby celkový počet odpovídal tomu, kolik jich zbývá uhádnout.

Úvaha nad vlastní nápovědou

Při hraní Krycích jmen často dochází k úvahám, proč byla vlastně použita daná nápověda místo nějaké jiné, která by více upřednostnila slova, nad kterými uvažuji. Mějme příklad, kdy se ve hře společně z dalšími nacházejí slova: „buňka“, „orgán“ a „počítač“. Zadaná napověda je „mozek 2“. Nejlépe nápověda sedí na slova „buňka“ a „orgán“. Ale proč byla použita nápověda „mozek“, která se dost podobá i slovu „počítač“? Kdybych chtěl více objasnit, že cílím slova „buňka“ a „orgán“, použil bych například nápovědu „tkáň 2“ nebo „plíce 2“. Tyto nápovědy by hráče nezaváděly k označení slova „počítač“. Takže pokud obdržím nápovědu „mozek“, hlavní špión měl nejspíše v úmyslu cílit právě slovo „počítač“. Proto se rozhodnu k označení slov „orgán“ a „počítač“.

Tato úvaha je implementována následnovně. Nejdříve se vyhodnotí podobnost nápovědy ke slovům již výše zmíněným postupem. Vybere se N nejpodobnějších slov, kde N je číslo nápovědy. K těmto slovům se vyhodnotí 1000 nejpodobnějších slov z modelu fastText pomocí funkce *most_similar*(), která je implementována v knihovně gensim. Pokud se lemma obdržené nápovědy nachází v těchto vyhodnocených slovech, nejspíše nápověda cílí původně zamýšlených N slov a žádné další úpravy nebudou provedeny. V případě, že se lemma nápovědy nenachází v nejpodobnějších slovech, přejde se ke generování nejlepší nápovědy pomocí těchto nejpodobnějších slov, která jsou filtrována seznamem kořenů (detailně popsané v podkapitole 5.5.1). Dalším pravidlem je, že se slovo použije pro vyhodnocení jako nápověda, pokud je jeho frekvence alepoň 180 výskytů v celém korpusu. Tento parametr dosahoval v rámci experimentování nejlepších výsledků na testovací sadě.

K vyhodnocení nápověd se použije kombinace obou modelů a za nejlepší nápovědu je považovaná ta, kde je největší rozdíl hodnot podobnosti mezi posledním cíleným slovem a prvním necíleným. Pro každé necílené slovo se spočítá rozdíl mezi průměrnou hodnotu podobnosti cílených slov a jeho hodnotou jak pro původní obdrženou nápovědu, tak i pro vygenerovanou nápovědu. U slova, kde dojde k největšímu vzrůstu rozdílu od cílených slov, se uvažuje, že nejspíše byla snaha jej zaměřit původní nápovědou. U uvedeného příkladu by pro nápovědu „tkáň“ podobnost u slova „počítač“ zřetelně klesla, než kdyby byla použitá nápověda „mozek“. Tento rozdíl mezi původní a aktuální vzdáleností od cílených slov se přičte ke hodnotě podobnosti vůči původní nápovědě. Tímto zvýšením hodnoty je snaha dané slovo posunout do popředí, aby bylo případně v rámci tahu označeno.

Aby nedocházelo k upřednostňování špatných slov, dojde ke zvýšení podobnosti pouze v případě, že původní hodnota podobnosti byla alespoň dvoutřetinová oproti průměru původně cílených slov. Tato podmínka by měla pomoci determinovat, že hlavní špión měl snahu cílit dané slovo původní nápovědou.

Parametr pro získání nejpodobnějších slov je nastaven relativně vysoko, protože v rámci sémantických modelů lze skoro vždy najít několik slov, které jsou vyhodnoceny jako podobnější, než původní nápověda. Při testování této strategie docházelo při nižším počtu hledaných nejpodobnějších slov velmi často k rozhodnutí, že použitá nápověda nebyla optimální a bylo špatně upřednostněno slovo, které by nemělo být cíleno. V těchto případech již původní vyhodnocení podobnosti slov správně udávalo cílená slova.

Tato úvaha je použita pouze pro případy, kdy je nápověda určená pro dvě nebo tři slova. Při vyšších čislech nápovědy už člověk podobné úvahy nemívá a spíše se snaží označit slova, která mají přijatelnou podobnost s nápovědou.

Při vyhodnocování této úvahy nad testovací sadou s uvedenými parametry došlo ke zvýšení úspěšnosti u nápověd mířených na dvě slova o 0,63%. U nápověd na tři slova se úspěšnost zvýšila o 1,44%. Výsledná úspěšnout nad celou sadou potom vzrostla na 78,68%.

Jelikož se nápověda generuje pouze pro jednu danou N-tici, doba trvání je jenom pár sekund, takže to nijak neovlivňuje plynulost hry.

5.5 Hlavní špión

Hlavní špión je implementován třídou *Spymaster* ve skriptu *spymaster.py*. Jeho úkolem je vytvářet nápovědy pro své spoluhráče. Výsledná nápověda by měla být ve vhodném poměru mezi počtem cílených slov a mírou jednoznačnosti, která udává, že daná nápověda spojuje právě zamýšlená slova. V rámci strategie hráče je vhodné tyto poměry dynamicky měnit v závislosti na průběžném výsledku hry. Pro tým, který prohrává, je výhodnější zariskovat a použít například méně jednoznačnou nápovědu spojující více slov.

5.5.1 Generování nápovědy

Při vytvoření nápovědy je potřeba vybrat takové vhodné slovo, které se co nejvíce podobá cíleným slovům a zároveň se co nejméně podobá nájemnému vrahovi a slovům nepřátelského týmu. Ideálně by se nemělo podobat ani slovům, která představují náhodné kolemjdoucí. První zvolený postup používá pro generování nápovědy všechna slova ze slovníku sémantického modelu. Tento postup měl ze začátku velkou časovou náročnost a generování jedné nápovědy trvalo i několik hodin. Postupnou optimalizací se podařilo snížit generování nápovědy na dobu průměrně 150 s při 25 slovech ve hře. Ke konci hry, kdy je počet slov ve hře výrazně nižší, pohybuje se doba generovnání kolem 50 s.

Validita nápovědy

Při výběru slov pro nápovědu je nejprve zkontrolováno, zda se jedná o platné české slovo. Toto filtrování je provedeno kontrolou POS značky daného slova[9]. Při dalším kroku je nutné ověřit, že možná nápověda nemá stejný kořen slova jako kterékoliv slovo, které je aktuálně ve hře. Při kontrole kořenů jsem použil stejný postup, jako u předchozí práce [15], kde se nejprve pomocí knihovny sumy[10] provede stemming potencionální nápovědy a slov ve hře. Jelikož stemming nefunguje se stoprocentní úspěšností, porovná se řetězcová vzdálenost kořenů slov. Pokud jsou si výsledné kořeny podobné na 85% a výše, u daných slov je usouzeno, že mají stejný kořen. Dalším pravidlem je, že jedno slovo nesmí být podřetězcem druhého slova. Díky tomu jsou zachyceny případy jako „les“ a „lesopark“, které nejsou stemmováním rozpoznány. Pro vyšší úspěšnost byly seznamy kořenů slov následně ručně upraveny.

Pro vyhodnocování generovaných nápověd je použit pouze model fastText. Za prvé kvůli časové náročnosti, a za druhé proto, že nenulové hodnoty NPMI se obecně pohybují ve vyšších hodnotách, než hodnoty fastTextu, což mělo tendenci při kombinaci modelů vybírat nápovědy s příliš velkými čisly, u kterých bylo dost nejednoznačené určování cílených slov.

Optimalizace generování

Pro snížení časové náročnosti byl trvale vytvořen seznam všech vhodných slov ze slovníku modelu fastText, která podléhají filtrováním POS značkou. Hra Krycí jména obsahuje pevně daných 400 herních slov, což umožňuje i předběžnou kontrolu kořenů slov. Pro každé

[9] Značka slovního druhu nesmí být z množiny {I,J,P,R,T,X,Z}. Seznam všech POS značek je dostupný na http://ufal.mff.cuni.cz/pdt/Morphology_and_Tagging/Doc/hmptagqr.html

[10] https://github.com/miso-belica/sumy/blob/dev/sumy/nlp/stemmers/czech.py

herní slovo je vytvořen seznam obsahující slova ze slovníku modelu, která mají stejný kořen. Vyhodnocení kořenů pro nápovědu potom probíhá negovanou kontrolou indentity těchto seznamů u aktuálních herních slov. Toto předzpracování snižuje časovou náročnost generování na třetinu.

Vyhodnocování všech vhodných slov modelu sice vždy najde nápovědu s optimálním výsledkem hodnotící funkce, ale je časově náročné. Pro užší výběr potencináních slov pro nápovědu je použita funkce *most_similar*(). Pro všechny N-tice slov se vyhodnotí 2000 nejpodobnějších slov, kde N je číslo 1 až 5, s tím že slovo představující nájemného vraha je použito jako parametr, kterému by se vyhodnocená slova neměla podobat. Maximální číslo nápovědy je nastaveno na 5, přestože výsledné vyhodnocení podobnosti by v určitých případech vycházelo i pro více slov. Při použití většího čísla nápovědy vznikají nesmyslné nápovědy, u kterých prakticky nelze vůbec určit, která slova cílí. V reálných hrách se nápověda i s číslem 5 objevuje ojediněle.

U výsledných nejpodobnějších slov všech N-tic je kontrolován výskyt v seznamu vhodných slov pro nápovědy a negativní výskyt v seznamech kořenů slov. Při této kontrole počet kandidátních slov výrazně klesne, díky čemuž je generování mnohem rychlejší. Slova, která vyhovují těmto kritériím, jsou následně použitá pro vyhodnocení, které udává užitkové skóre nápovědy. Pro rychlejší kontrolu identity jsou zmíněné seznamy uložené také jako množiny, které jsou implementovány hashovací tabulkou, která má složitost $O(1)$ v ideálním případě.

Hodnota 2000 pro hledání nejpodobnějších slov byla nastavena v rámci manuálního testování, kdy se zároveň provádělo generování se všemi slovy modelu. S hodnotou 2000 byla výsledná nápověda stejná, nebo alespoň téměř stejně kvalitní, jako u pomalejšího generování používající všechna slova modelu. Metoda *select_best_hint*() pro generování nápovědy má parametr $force\_all$, kterým lze vynutit pomalejší, ale zaručeně optimální generování se všemi slovy modelu. Rychlejší metoda trvá kolem 25 s při první generované nápovědě. Při dalších tazích už jsou nejpodobnější slova všech N-tic uložená, takže generování trvá pouze pár sekund.

Vyhodnocení nápovědy

Hodnotící funkce v předchozí práci [15] spočívala v sečtení všech modelem vyhodnocených podobností herních slov s potencinální nápovědou. Podobnosti slov vlastního týmu přispívaly pozitivní hodnotou k celkovému skóre nápovědy, podobnosti ostatní slov přispívaly negativně podle týmové příslušnosti (náhodný kolemjdoucí, nepřátelský agent a nájemný vrah). Já jsem se rozhodl pro vyhodnocení rozdílu podobnotí mezi cílenými slovy a prvním nejpodobnějším slovem, které nepatří k mému týmu. Při hraní Krycích jmen především závisí, jak moc se cílená slova odlišují od ostatních. Například u posledních 10 nejméně podobných slov vůbec nezáleží na týmové příslušnosti slova, neboť se očekává, že nebudou vůbec braná v úvahu.

Výsledné skóre nápovědy tedy hlavně závisí na počtu cílených slov a odlišností od dalšího slova, které nepatří k mému týmu. Funkce pro výpočet skóre je následující:

$$U_h = N \cdot (\mathrm{sim}(h, W_t) - \mathrm{sim}(h, W_{nt})) + c \tag{5.5}$$

N představuje počet cílených slov. Pokud se jedná o nápovědu cílenou na jedno slovo, pro hodnotu N se použije 0,5. Vytvořit silnou asociaci pro jedno slovo, tak aby se výrazně lišila od ostatních slov, je velmi jednoduché. Dávat pouze nápovědy na jedno slovo je nevýhodné, protože se jedná o velmi pomalou strategii hraní, která nemá prakticky žádnou šanci vítězství. Výraz $\mathrm{sim}(h, W_t)$ je podobnost nápovědy s posledním cílovým slovem v pořadí po-

dobnosti, $\text{sim}(h, W_{nt})$ je podobnost s prvním slovem, které nepatří k mému týmu. Hodnota c je 0,1, pokud vyhodnocované slovo má alespoň frekvenci výskytu 1000 v celém korpusu. Při začlenění frekvence výskytu jako váhovaného parametru, nebo rozdělení slov do kategorií podle tohoto parametru, byla výsledná nápověda právě tímto začleněním značně ovlivněna. Rozdíl podobností slov neměl tak velkou váhu a spíše byla vybírána co nejfrekventovanější slova. Bez použití hodnoty c ale často docházelo k výběru slov, která jsou většinou odvozená od běžnějšího slova, případně se jednalo o relativně neznámé slovo. Jednoduchým použitím hodnoty 0,1 má stále hlavní vliv rozdíl podobností slov, ale výsledné nápovědy se už více blíží slovům, která by použil člověk při hře. Můžou nastat případy, kdy málo frekventované slovo by bylo velmi vhodnou nápovědou. Proto nejsou málo frekventovaná slova natvrdo odfiltrována, ale musí tvořit velmi silnou asociaci s cílenými slovy, aby byla brána v úvahu.

Při vyhodnocování uvažované nápovědy se spočítá podobnost se všemi slovy ve hře a následně se vyhodnotí slovo jako nápověda pro 1 až N cílených slov, kde N je nepřerušovaná posloupnost od nejpodobnějšího po N-té nejpodobnější slovo ve hře, kde každé slovo z těchto slov musí náležet k mému týmu. Vyjímku tvoří nápovědy s číslem 3 a výše, kde je dovoleno, aby v posloupnosti bylo maximálně jedno slovo, které představuje náhodného kolemjdoucího. V případě, že by toto slovo spoluhráč označil, bude mít pořád dostatečnou informaci pro dodatečné označení slov v dalších tazích hry. Jak je vidět v tabulce 5.5, slovo „melounový“ uvažované pro nápovědu má vyhodnocenou posloupnost se 4 nejpodobnějšími slovy, které náleží vlastnímu týmu. Poslední dvě tato slova mají ovšem dost blízkou podobnost s dalšími nepřátelskými slovy ve hře. Proto v tomto případě by byla tato nápověda použitá pro dvě slova „banán“ a „limonáda“.

V případě, že vyhodnocená nápověda je přídavné jméno a všechna cílená slova jsou ženského rodu, zamění se koncovka slova „ý“ na „á“, což by mělo spoluhráči lépe napovědět, která slova jsou cílena.

nápověda: **melounový**			
herní slovo	**tým**	**podobnost**	**skóre nápovědy**
banán	můj tým	0,4752	$0{,}5 \cdot 0{,}2703 + 0{,}1 = 0{,}2352$
limonáda	můj tým	0,4697	$2 \cdot 0{,}2648 + 0{,}1 = 0{,}6296$
kino	můj tým	0,218	$3 \cdot 0{,}0131 + 0{,}1 = 0{,}1393$
klaun	můj tým	0,2068	$4 \cdot 0{,}0019 + 0{,}1 = 0{,}1076$
maso	nepřítel	0,2049	N/A
moře	nepřítel	0,1973	N/A
sněženka	můj tým	0,1883	N/A
dřevo	nepřítel	0,1787	N/A
ponožka	neutrální	0,1744	N/A
američan	vrah	0,0659	N/A
⋮	⋮	⋮	⋮

Tabulka 5.5: Vyhodnocení slova uvažovaného pro nápovědu

Hyperonyma

Použité sémantické modely nemají moc tendenci vytvářet silnou asociaci s nadpojmy (hyperonymy) slov. Při hraní Krycích jmen ovšem reální hráči kladou hyperonymům velkou

váhu. Například pokud má hlavní špión ve svých slovech 3 zvířata a jinde se žádné zvíře nevyskytuje, použije nápovědu „zvíře 3“, což bude jeho spoluhráčovi jasné, jaká slova cílí. Při vyhodnocení nápovědy sémantickými modely sice zpravidla budou tato slova nejpodobnější, ale nebývá u nich takový rozdíl podobnosti od ostatních slov ve hře. K tomuto účelu byly ručně vytvořeny seznamy herních slov, která mají stejné hyperonymum.

Po vygenerování nápovědy se zkontroluje, zda by šlo použít jedno z definovaných hyperonym, které by spojovalo více slov, než kolik spojuje vygenerovaná nápověda. Pro použití hyperonyma platí, že žádné z hyponym (opak hyperonyma) nesmí být nájemný vrah nebo z nepřátelského týmu. Pokud by hyperonymum cílilo alespoň 3 slova, platí opět, že jedno další slovo se může vyskytovat v náhodných kolemjdoucích.

5.5.2 Strategie hlavního špióna

Při použití strategie se hlavní špión zaměřuje na aktuální stav, zda jeho tým vyhrává nebo prohrává, čímž se ovlivňuje míra rizika, která upřednostní nápovědy s větším či menším počtem cílených slov. Dále si hlavní špión pamatuje nedohádaná slova svého spoluhráče z předchozích tahů, díky čemuž uzpůsobuje nápovědy tak, aby měl spoluhráč možnost daná slova efektivně „dohádat“.

Parametr rizika

V různých herních situacích se vyplatí uzpůsobit herní styl tak, aby nápověda, kterou hráč dává nebyla příliš riskantní. To se děje v případě, že tým přehledně vyhrává. Naopak když tým výrazněji prohrává, vyplatí se zariskovat s nápovědou, která spojuje více slov, ašvak nápověda nebude tak jednodnačná. Tento parametr rizika je v původní rovnici pro ohodnocení nápovědy implementován následnovně:

$$U_h = N^{\frac{M}{E}} \cdot (\mathrm{sim}(h, W_t) - \mathrm{sim}(h, W_{nt})) + c \tag{5.6}$$

Parametr M udává počet zbývajících slov ve hře mého týmu, E počet slov protivníka. Na základě poměru těchto hodnot je potom váhován parametr N, kdy v případě, že můj tým prohrává, bude mít počet cílených slov větší váhu. Naopak čím více můj tým vyhrává, tim spíše budou upřednosněny jednoznačnější nápovědy s bezpečným rozdílem podobnosti vůči necíleným slovům.

Pamatování předchozích tahů

Hlavní špión si pamatuje své použité nápovědy, takže během hry nepoužije opět stejnou nápovědu. Dále si ke každé nápovědě pamatuje slova, která cílil. Pokud při dalším generování jsou v jeho týmu alesoň 2 slova, která ještě necílil, generované nápovědy budou soustředěné pouze na tato slova. V opačném případě budou cílená kterákoliv slova z mého týmu. Při hádání slov svého spoluháře se sleduje, zda svůj tah ukončil dobrovolně, to znamená, že se nesnažil hádat $N + 1$ slov na nápovědu s číslem N. Potom si hlavní špión smaže seznam všech nedohádaných slov a při dalších tazích je bude znovu cílit. V případě, že spoluháč špatně označí slovo, které se vztahovalo k předchozím nápovědám (jedná se o označení $N + \textit{prvního}$ slova), dostane ještě jednu šanci tato nedohádaná slova označit v příštích tazích. Při další chybě jsou všechna nedohádaná slova opět smazána a budou cílená v dalších kolech.

Když nastane situace, že jsou nedohádaná alespoň 3 slova z předchozích tahů, je vhodné použít nápovědu „nekonečno“. V tomto případě je nápověda vybrána stejným způsobem

jako při generování se standardním číslem, pouze číslo se zamění za „nekonečno“. U této nápovědy se neuloží cílená slova a ke každému otáčenému slovu se přistupuje tak, že spoluhráč se snaží daná slova dohádat z předchozích tahů. Pro celý tento tah jsou použita stejná pravidla, jako při hádání $N + prvního$ slova u standardní nápovědy.

Nápověda „nekonečno“ se také použije v případě, že nepřátelskému týmu zbývá poslední slovo, takže je víceméně jasné, že dalším tahem vyhraje hru. Pokud vygenerovaná nápověda necílí všechna zbývající slova mého týmu, změní se číslo v „nekonečno“. I v reálných hrách se používá tento přístup. Jedná se spíše o náhodné tipování, jako poslední šanče zkusit zabránit jisté prohře.

Vrácení posledního tahu

Obě implementované role hráče mají funkci, která umožní vrátit zpět poslední provedenou akci tak, aby hráč byl v korektním stavu se všemi proměnlivými parametry strategie. Vracení tahů bylo implementováno především kvůli kompatibilitě umělého hráče s funkcemi, které poskytuje mobilní aplikace.

5.6 Podpora angličtiny

Výsledný systém byl dodatečně upraven tak, aby mohl podporovat i anglický jazyk ve hře Krycí jména.

Použité modely

Jako prediktivní model je použitý fastText, který byl už natrénovaný stažen z internetu[11] [13]. Byl zvolen větší model fastText, který obsahuje 2 milióny slov, protože menší s 1 miliónem slov neobsahoval herní slovo „loch ness“ v žádném tvaru. Pro model založený na počtu byla použitá anglická wikipedie. Skript *parse_en_wiki.py* přijímá na vstup soubor, který byl výstupem nástroje Wikipedia Extractor a provádí tokenizaci, lemmatizaci a odstranění nepotřebných slov. Matice spoluvýskytu byla potom vytovřena stejným způsobem jako u korpusu CWC-2011. Model založený na počtu obsahuje 1,3 miliónu slov.

Člen operativy

Třída *Operative_en* dědí všechny atributy a metody ze třídy *Operative* s tím, že má redefinované metody, které jsou vázané na specifický jazyk. Jedná se o metody, kde je prováděna lemmatizace slov. Lemmatizace v kombinaci s POS značkováním je prováděna pomocí knihovny NLTK.

Hlavní špión

Třída *Spymaster_en* také dědí ze třídy *Spymaster* a má redefinované jazykově vázané metody. Pro vytvoření slovníku vhodného pro nápovědy byla použita knihovna spaCy, která dokázala POS značkováním odfiltrovat více nevhodných slov, než knihovna NLTK. Pro stemmování je použit LancasterStemmer[12] z knihovny NLTK. I když používá argresivní algoritmus, nefunguje stoprocentně. V porovnání se stemmováním v českém jazyce dochází

[11] https://fasttext.cc/

[12] https://www.nltk.org/_modules/nltk/stem/lancaster.html

u angličtiny k lepším výsledkům, protože angličtina není tak morfologicky složitá jako čeština.

Kvůli tomu, že POS značkováním nelze u angličtiny moc dobře odfiltrovat nesmyslná slova, jako při použití knihovny MorphoDiTa u češtiny, je pro slova určen minimální počet výskytů 30 v korpusu, aby mohla být použita pro generování nápovědy. Jelikož fastText použitý pro angličtinu má větší slovník, než fastText pro češtinu, trvá generování prvního tahu zhruba dvakrát déle.

5.7 Webová služba

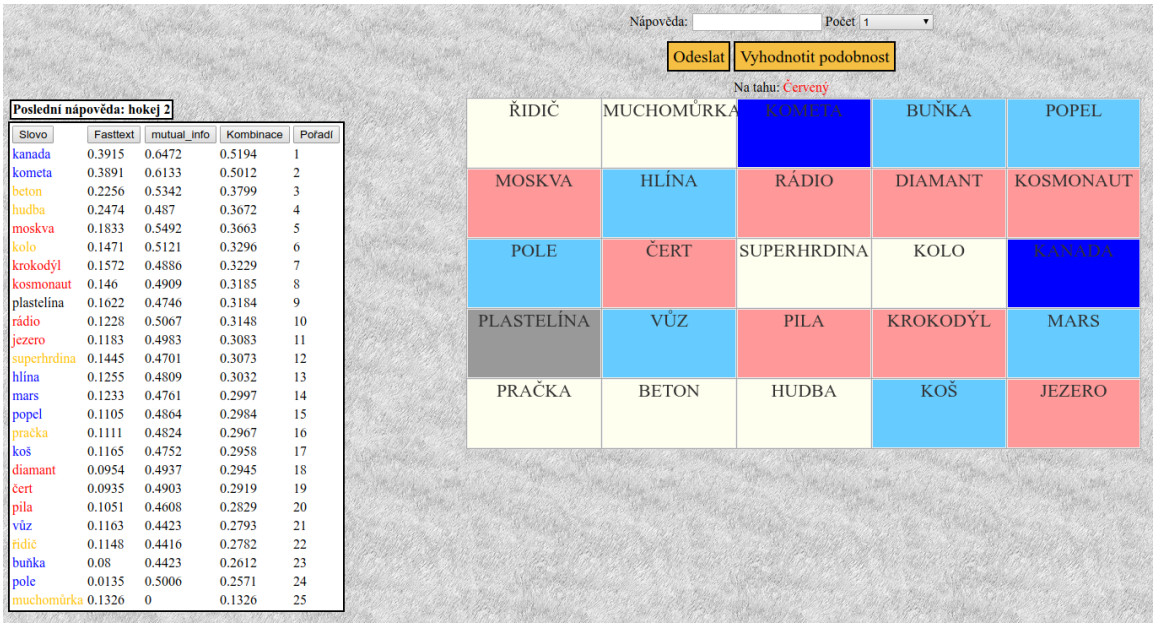
Při vytváření webové služby jsem vycházel z předchozí práce [15], kde jsem využil zdrojový kód jako vzorovou kostru programu. Webová služba běží na školním serveru athena3 a je přístupná na adrese `http://athena3.fit.vutbr.cz:8086/`. Jejím účelem je testování a možnost hraní Krycích jmen v internetovém prohlížeči, kde počítač zastupuje buď roli hlavního špióna nebo člena operativy. Součásti webové služby je rozhraní pro komunikaci s mobilní aplikací, kde počítač slouží jako poradce reálným hráčům, nebo kompletně zastoupí hráče a používá implementovanou strategii.

Webová služba je implementována pomocí knihovny Flask[13]. Jedná se o mikroframework založeném na Werkzeug[14] a Jinja2[15]. Komunikace na straně webového prohlížeče je implementována jazykem JavaScript.

Aplikace má pro webový prohlížeč dostupné tyto služby:

- */*, */index*
  Rozcestník zprostředkující odkazy na ostatní služby
- */hint_debug*
  Hra, kde počítač zastupuje roli člena operativy v obou týmech. Počítač zde nepoužívá implementovanou strategii. Při vytváření nápovědy lze označit zamýšlená slova, to umožní vytvoření záznamu pro zpětné testování sémantických modelů.
- */operative*
  Ekvivaletní služba s */hint_debug*. Počítač zde používá implementovanou strategii.
- */game_debug*
  Hra, kde počítač zastupuje hlavního špióna v obou týmech. Zde nepoužívá strategii.
- */spymaster*
  Ekvivaletní s */game_debug*, ale počítač využívá strategii.
- Všechna herní rozhraní jsou dostupné i v anglické verzi. Jejich jméno je totožné, pouze je přidána přípona „*_en*“.

[13] `http://flask.pocoo.org/`
[14] Knihovna umožnující komunikaci webového serveru s Python aplikací
[15] Jazyk pro vytváření HTML šablon `http://jinja.pocoo.org/docs/2.10/`



Obrázek 5.2: Ukázka webové aplikace, kde počítač zaujímá roli člena operativy. Snímek obsahuje i kontrolní tabulku vyhodnocení podobností slov.

Jelikož je znatelná časová odezva mezi prvním tahem hlavního špióna a jeho dalšími tahy, je nastaven plánovač, který každý den ve 2 ráno doplní do zásoby 100 předgenerovaných her s hlavními špióny. U hlavních špiónů v začínajícím týmu je generována první nápověda a u špiónů z druhého týmu jsou alespoň předpočítána nejpodobnější slova.

Pro mobilní aplikaci bylo vytvořeno následující rozhraní:

- */check_hint*
  Server obdrží použitou nápovědu a aktuální slova ve hře. V odpovědi vrátí slova, která by umělý hráč označil. Tato funkce slouží jako upozornění pro hlavního špióna, zda náhodou necílí nepřátelské slovo, nebo může sloužit pro člena operativy jako rada, co má označit.

- */request_hint*
  Server obdrží slova s jejich týmovou příslušností. V odpovědi vráti nejlepších 5 nápověd cílících 2 až 5 slov, kde každá odeslaná nápověda cílí unikátní N-tici. Funkce slouží jako rada hlavnímu špiónovi, jaké nápovědy by mohl použít.

- */mobile_operative_* + $\{$*init*|*send_hint*|*send_enemy_hint*|*played_word*|*undo*$\}$
  Rozhraní umožňující implementovanému členovi operativy plně využít svojí strategii v mobilní aplikaci.

- */mobile_spymaster_* + $\{$*init*|*request_hint*|*played_word*|*end_turn*|*undo*$\}$
  Rozhraní pro hlavního špióna.

Kapitola 6

Vyhodnocení

V této kapitole jsou popsané výsledky, které byly získány během testování. Pro testování byly využity záznamy reálných her a data nasbíraná z webové aplikace. Vyhodnocení se vztahuje pouze pro českou verzi. Podpora angličtiny byla implementována dotatečně a nejsou nasbírána data pro automatické vyhodnocení.

6.1 Hádání slov

Úspěšnost hádání slov bylo vyhodnoceno na záznamech reálných her, skládajících se z 680 jednotlivých tahů. Úspěšnosti modelů a jejich kombinací jsou popsány v podkapitolách 5.3.4 Vyhodnocení modelů a 5.4.2 Kombinace modelů. Detailní úspěšnost výsledné použité kombinace je uvedena v tabulce 6.1. V případě nápověd typu „nekonečno“ jsou použita referenční slova, která se skutečně vztahují pro danou nápovědu, a nikoliv pro slova z minulých tahů. U nápovědy s číslem 0 je uvedeno slovo nebo slova, která by se neměla označit. Tento typ nápovědy je vyhodnocen stejně jako u běžného čísla tak, že se právě sleduje pozitivní podobnost s uvedenými slovy. Ve hře bývá účelem této nápovědy označení slov z minulých tahů, která v rámci vyhodnocení nejsou brána v potaz.

počet slov	**počet tahů**	**úspěšnost**
1	93	91,48%
2	394	76,54%
3	132	75,20%
4	23	77,68%
5	8	65,91%
6	1	74,60%
7	1	80,20%
0	3	61,85%
nekonečno	25	71,82%
celkem	680	77,96%

Tabulka 6.1: Detailní úspěšnost kombinace modelů

Vyhodnocení implementované úvahy nad vlastní nápovědou a její případné ovlivnění podobnosti slov je uvedeno v tabulce 6.2. Jedná se o případy nápověd s číslem 2 a 3.

	nápověda 2	**nápověda 3**	**agregace nápověd 2 a 3**
počet tahů	394	132	526
úspěšnost bez použití	76,54%	75,20%	76,15%
úspěšnost s použitím	77,17%	76,64%	77,09%
% případů s vyšší úspěšnosti	15,99%	16,67%	16,16%
% případů s nižší úspěšnosti	12,18%	4,55%	9,89%
% případů se stejnou úspěšnosti	71,83%	78,79%	73,95%

Tabulka 6.2: Vyhodnocení úvahy nad vlastní nápovědou

Celková úspěšnost člena operativy nad celou testovací sadou byla 78,68%. Systém z předchozí práce [15] měl úspěšnost 71,59% nad stejnou testovací sadou. Oproti předchozí práci došlo v rámci strategie k efektivnějšímu zpracování nepřátelských nápověd a využití informací z předchozích nepovedených tahů.

6.2 Generování nápověd

Pro vyhodnocení generovaných nápověd bylo vytvořeno rozhraní v rámci webové aplikace. Bylo vybráno 50 různých herních situací z testovací sady a pro každou z nich byly předem vygenerovány nápovědy. Nápovědy byly generovány systémem implementovaném v této práci a také systémem z předchozí práce [15] pro porovnání. Dále byly přidány i nápovědy, které byly použity člověkem v daných situacích v záznamu hry. Testovací rozhraní zobrazí uživatelovi náhodně jednu z 50 předpřipravených situací a postupně uživatel označuje slova pro jednotlivé vygenerované nápovědy. Záznam pro danou situaci se uloží pouze v případě, že uživatel označí slova pro všechny určené nápovědy. Tím je zajištěno, že jednotlivé metody generování nápověd budou vyhodnoceny na totožných datech stejnými uživateli.

Jelikož byly nápovědy použité člověkem přidány až v průběhu testování, bylo nasbíráno méně testovacích dat pro tuto skupinu nápověd. Celkem bylo zaznamenáno 148 tahů pro obě skupiny generované počítačem a 54 tahů pro nápovědy použité člověkem.

Vyhodnocení generovaných nápověd pomocí procentuální úspěšnosti hádaných slov by nebylo příliš objektivní, neboť postup, který by generoval nápovědy pouze na jedno cílené slovo, by sice měl vysokou procentuální úspěšnost, avšak neměl by skoro žádnou šanci vyhrát. U kvalitních nápověd, které cílí více slov, se může stát, že spoluhráč označí slovo, které není z jeho týmu. V tomto případě může mít dost dobrou informaci o jeho potencionálních slovech do dalších kol. Pro ohodnocení nápověd jsem zvolil následující kritérium:

- Je ohodnoceno každé označené slovo bez ohledu na pořadí, které by mohlo způsobit předčasný konec tahu.
- Každé označené slovo z mého týmu má hodnotu $+1$.
- Každé označené nepřátelské slovo z nepřátelského týmu má hodnotu -1.
- Každé slovo představující náhodného kolemjdoucího má hodnotu 0.
- Skóre nápovědy pro daný tah potom představuje sečtení všech hodnot označených slov.
- Pokud byl označen nájmený vrah, skóre nápovědy se automaticky rovná negativnímu počtu všech neodkrytých nepřátelských slov před začátkem tahu. Odkrytí nájemného

vraha je ekvivaletní s odkrytím všech nepřátelských slov. Navíc čím více nepřátelských slov se nachazí ve hře, tím je více trestáno odkrytí nájemného vraha. Naopak, čím méně slov zbývá nepřátelskému týmu, tím je logičtější zariskovat a použít nápovědu, která by mohla vést k odkrytí nájemného vraha.

Jelikož pro každou generovanou situaci byl nasbírán různý počet záznamů, bylo nejprve spočítáno průměrné skóre pro každou situaci. Nakonec z těchto hodnot byla vypočítána celková průměrná hodnota. Po vyhodnocení nasbíraných dat měl systém implementovaný v této práci průměrné skóre 0,989 na tah, systém z předchozí práce [15] 0,653 a nápovědy použité člověkem 1,061.

Při vyhodnocení, kde by označení nájemného vraha znamenalo automaticky skóre tahu 0, měl systém implementovaný v této práci průměrné skóre 1,221 na tah, systém z předchozí práce 1,063 a nápovědy použité člověkem 1,3.

Pro vyhodnocení nápověd byly generovány nápovědy bez použití definovaných hyperonym, které byly přidané až dodatačně. Díky tomu, že jsou tato hyperonyma definována manuálně, lze očekávat, že by generované nápovědy dostahovaly lepšího užitku.

V rámci vyhodnocení implementované strategie pro porovnání s reálnými hráči bylo zatím odehráno málo her. Objetivnější vyhodnocení bude možné provést, až webová služba nasbírá dostatek dat o odehraných hrách z mobilní aplikace.

Kapitola 7

Závěr

V rámci této práce byl v jazyce Python implementován systém, který je schopný zastoupit hráče ve hře Krycí jména v roli hádajícího hráče i hráče zadávajícího nápovědy. Systém podporuje český a anglický jazyk. Pro určování podobnosti slov byl natrénován prediktivní model fastText a vytvořena matice spoluvýskytů využívaná metodou Pointwise Mutual Information. Byla popsána herní strategie obou implementovaných rolí hráčů, která využívá kontextové informace z předchozích tahů.

Výsledný systém obsahuje webovou službu[1], která slouží pro testování implementovaných hráčů a sběr dat. Webová služba poskytuje i komunikační rozhraní, které poskytuje využití implementovaného systému v mobilní aplikaci podporujicí hru Krycí jména.

Úspěšnost hádání slov implementovaného systému byla 78,68%. Oproti předchozí práci se zvýšila úspěšnost o 7,09%. Generování nápověd dosahovalo výsledků, které se přiblížily k úspěšnosti nápověd použitých člověkem. Systém implementovaný v této práci měl průměrné skóre efektivity 0,989 na tah, systém z předchozí práce 0,653 a nápovědy použité člověkem 1,061. V rámci zpětné vazby sdělovali uživatelé, kteří systém testovali, že dosahovali vyšší úspěšnosti prováděných tahů poté, co zjistili, jaké typy asociací počítač vytváří. Tento princip platí i v reálných hrách, kdy se úspěšněji hraje s hráči, které osobně znáte a máte přehled o jejich znalostech a typu uvažování.

Možnost se přízpůsobit schopnostem jednotlivých hráčů by mohla být jedna variana rozšíření, kdy by si počítač pamatoval, s kým hraje a podle předchozích her by přizpůsoboval vytváření slovních asociací. Další možnost vývoje se naskýtá u sémantických modelů, kde by se před vlastním trénováním modelů určily jednotlivé významy homonymních slov v korpusu a k těmto významům by se přistupovalo jako k unikátním slovům. Jelikož sémantické modely nemají příliš tendenci upřednostňovat generalizaci pojmů, mohla by se tato problematika vyřešit vytvořením modelu pro automatické zpracování hyperonym.

[1] http://athena3.fit.vutbr.cz:8086/

Literatura

[1] Balakrishnan, V.; Ethel, L.-Y.: Stemming and Lemmatization: A Comparison of Retrieval Performances. *Lecture Notes on Software Engineering*, ročník 2, 01 2014: s. 262–267, doi:10.7763/LNSE.2014.V2.134.

[2] Baroni, M.; Dinu, G.; Kruszewski, G.: Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, 2014, s. 238–247, doi:10.3115/v1/P14-1023.
URL http://aclweb.org/anthology/P14-1023

[3] Bouma, G.: Normalized (Pointwise) Mutual Information in Collocation Extraction. *Proceedings of the Biennial GSCL Conference 2009*, 01 2009.

[4] Chvátil, V.: Krycí jména. [Online; navštívené 08.03.2019].
URL http://krycijmena.cz

[5] Dash, N.: *Corpus Linguistics: An Introduction*. 2008, ISBN 9788131716038.

[6] Donges, N.: Introduction to NLP. [Online; navštívené 31.01.2019].
URL https://towardsdatascience.com/introduction-to-nlp-5bff2b2a7170

[7] HOŠTÁK, V. S.: *Shlukování slov podle významu*. Bakalářská práce, Vysoké učení technické v Brně, Fakulta informačních technologií, Brno, 2017, vedoucí práce Doc. RNDr. Pavel Smrž, Ph.D.

[8] HURTA, M.: *Podpora hry Krycí jména na mobilním telefonu s OS Android*. Bakalářská práce, Vysoké učení technické v Brně, Fakulta informačních technologií, Brno, 2019, vedoucí práce Doc. RNDr. Pavel Smrž, Ph.D.

[9] Lenci, A.: Distributional semantics in linguistic and cognitive research. ročník 20, č. 1, 2008: s. 1–31.

[10] Levy, O.; Goldberg, Y.: Neural Word Embedding As Implicit Matrix Factorization. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, Cambridge, MA, USA: MIT Press, 2014, s. 2177–2185.
URL http://dl.acm.org/citation.cfm?id=2969033.2969070

[11] Mesleh, A.: Chi Square Feature Extraction Based Svms Arabic Language Text Categorization System. *Journal of Computer Science*, ročník 3, 06 2007, doi:10.3844/jcssp.2007.430.435.

[12] Mikolov, T.; Chen, K.; Corrado, G.; aj.: Efficient Estimation of Word Representations in Vector Space. *CoRR*, ročník abs/1301.3781, 2013, 1301.3781.
URL http://arxiv.org/abs/1301.3781

[13] Mikolov, T.; Grave, E.; Bojanowski, P.; aj.: Advances in Pre-Training Distributed Word Representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[14] Myerson, R. B.: *Game theory - Analysis of Conflict*. Harvard University Press, 1997, ISBN 978-0-674-34116-6.

[15] OBRTLÍK, P.: *Počítač jako inteligentní spoluhráč ve slovně-asociační hře Krycí jména*. Diplomová práce, Vysoké učení technické v Brně, Fakulta informačních technologií, Brno, 2018, vedoucí práce Doc. RNDr. Pavel Smrž, Ph.D.

[16] Recchia, G.; Jones, M. N.: More data trumps smarter algorithms: Comparing pointwise mutual information with latent semantic analysis. *Behavior Research Methods*, ročník 41, č. 3, Aug 2009: s. 647–656, ISSN 1554-3528, doi:10.3758/BRM.41.3.647.
URL https://doi.org/10.3758/BRM.41.3.647

[17] Řehůřek, R.; Sojka, P.: Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, Valletta, Malta: ELRA, Květen 2010, s. 45–50.

[18] Rong, X.: word2vec Parameter Learning Explained. *CoRR*, ročník abs/1411.2738, 2014, 1411.2738.
URL http://arxiv.org/abs/1411.2738

[19] Spoustová, J.; Spousta, M.: CWC2011. 2012, LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.
URL http://hdl.handle.net/11858/00-097C-0000-0006-B847-6

[20] Straka, M.; Straková, J.: Czech Models (MorfFlex CZ 161115 + PDT 3.0) for MorphoDiTa 161115. 2016, LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.
URL http://hdl.handle.net/11234/1-1836

[21] V. Srividhya, R. A.: Evaluating Preprocessing Techniques in Text Categorization. *International Journal of Computer Science and Application*, 2010, ISSN 0974-0767.

[22] Yang, Y.; Pedersen, J. O.: A Comparative Study on Feature Selection in Text Categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997, ISBN 1-55860-486-3, s. 412–420.
URL http://dl.acm.org/citation.cfm?id=645526.657137

[23] Yannakakis, G. N.; Togelius, J.: *Artificial Intelligence and Games*. Springer, 2018, http://gameaibook.org.

Příloha A

Obsah paměťového média

DVD nosič obsahuje:

- Bakalářskou práci ve formátu pdf
- Zdrojové soubory LaTeX – adresář: /bp_src
- Zdrojové soubory implementovaných modulů – adresář: /src
- Dokumentaci zdrojových kódů – adresář: /src/docs
- Virtuální prostředí s Python 3.6.7 a potřebnými knihovnami – adresář: /src/xjares00env
- Zdrojové soubory webové služby – adresář: /src/web
- Vytvořená statická data využívaná při běhu systému – adresář: /src/data
  Kvůli paměťové náročnosti nebyly přidány sémantické modely. Ty jsou dostupné v projektovém adresáři na školním serveru výzkumné skupiny KNOT.
- Skripty a data spojená s vyhodnocením úspěšnosti – adresář: /src/evaluation
- Vytvořený plakát ve formátu pdf