



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

**CHATBOT POSTAVENÝ NA UMĚLÝCH
NEURONOVÝCH SÍTÍCH**

CHATBOT BASED ON ARTIFICIAL NEURAL NETWORKS

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE

AUTHOR

JIŘÍ ČECHÁK

VEDOUCÍ PRÁCE

SUPERVISOR

Ing. IGOR SZÓKE, Ph.D.

BRNO 2018

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií

Akademický rok 2017/2018

Zadání bakalářské práce

Řešitel: **Čechák Jiří**

Obor: Informační technologie

Téma: **Chatbot postavený na umělých neuronových sítích**

Chatbot Based on Artificial Neural Networks

Kategorie: Zpracování řeči a přirozeného jazyka

Pokyny:

1. Seznamte se se základy chatbotů postavených na generativních modelech. Nastudujte základy strojového učení.
2. Najděte vhodná a veřejně dostupná data. Navrhněte jednoduchého chatbota (vstupem a výstupem bude text). Otestujte kvalitu jeho odpovědí pomocí vhodné zvolené metriky.
3. Zdokonalte chatbota (např. více dat, lepší machine learning techniky) a průběžně sledujte jeho úspěšnost.
4. Otestujte chatbota na několika uživateli. Diskutujte dosažené cíle a navrhněte směry dalšího vývoje.
5. Vyrobte A2 plakátek a cca 30 vteřinové video prezentující výsledky Vaší práce.

Literatura:

- Dle pokynů vedoucího
- <http://www.wildml.com/2016/04/deep-learning-for-chatbots-part-1-introduction/>

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 a 2 ze zadání.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese <http://www.fit.vutbr.cz/info/szz/>

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Szóke Igor, Ing., Ph.D.,** UPGM FIT VUT

Datum zadání: 1. listopadu 2017

Datum odevzdání: 16. května 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
L. Štábl 66 Brno, Božetěchova 2



doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

Bakalářská práce popisuje implementaci a způsob fungování generativního chatbotu. Chatbot byl implementován v jazyce Python pomocí umělých neuronových sítí a je založen na sequence-to-sequence principu. Výsledný chatbot obsahuje tři modely, které je možné každý natrénovat a poté použít k vedení konverzace ve vytvořeném GUI. Chatbot byl, po natrénování všech modelů, testován pomocí metriky BLEU. Také byl testován několika uživateli, kteří porovnali kvalitu generovaných odpovědí s kvalitou odpovědí již existujícího chatbotu Cleverbot. Pro lepší pochopení dané problematiky se zde nachází jednoduchý popis základních pojmů, jako je umělá inteligence, umělá neuronová síť, rozdíl mezi uzavřenou a otevřenou doménou, word embedding a základní popis chatbotů a jejich dělení, včetně jejich výhod, nevýhod a použití.

Abstract

The thesis describes an implementation and the way generative chatbot operates. The chatbot was implemented in Python using artificial neural networks and is based on a sequence-to-sequence principle. The final chatbot contains three models, which can be trained and used for conversations in a created GUI. After training of all three models, the chatbot was then tested by using BLEU metric. It was also tested by some users who compared the quality of its generated answers with the quality of answers created by already an existing chatbot Cleverbot. For a better understanding of the given problematics, there is a simple description of the basic terms, such as artificial intelligence, artificial neural networks, the difference between closed and open domain, word embedding and a basic description of the chatbots and their types, including their advantages, disadvantages and usage.

Klíčová slova

chatbot, bot, chatterbot, talkbot, generativní chatbot, umělá neuronová síť, LSTM, strojové učení, umělá inteligence

Keywords

chatbot, bot, chatterbot, talkbot, generative chatbot, artificial neural network, LSTM, machine learning, artificial intelligence

Citace

ČECHÁK, Jiří. *Chatbot postavený na umělých neuronových sítích*. Brno, 2018. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Igor Szóke, Ph.D.

Chatbot postavený na umělých neuronových sítích

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Igora Szóke Ph.D. Další informace mi poskytl pan Ing. Karel Beneš. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....

Jiří Čechák
9. května 2018

Poděkování

Děkuji panu Ing. Igoru Szóke Ph.D. za vedení a rady při práci na této bakalářské práci. Také děkuji panu Ing. Karlu Benešovi za velmi užitečnou konzultaci.

Obsah

1	Úvod	3
2	Základní pojmy	5
2.1	Chatbot	5
2.2	Doména	7
2.3	Umělá inteligence	7
2.4	Neuronová síť	8
3	Základní dělení chatbotů	14
3.1	Retrieval based chatbot	14
3.2	Generativní chatbot	16
4	Implementace	19
4.1	Použité technologie	19
4.2	Volitelné parametry a módy chatbotu	19
4.3	Slovník	20
4.4	Trénovací data	20
4.5	Word embedding	23
4.6	Model	24
4.7	Trénink	25
4.8	Testování	27
4.9	Chatování	27
4.10	Základní verze chatbotu	28
4.11	Vylepšená verze chatbotu	29
5	Testování a výsledky	34
5.1	Základní verze chatbotu	34
5.2	Testování uživateli	37
5.3	Vylepšená verze chatbotu	38
5.4	Cleverbot	46
5.5	Shrnutí výsledků testování	47
6	Závěr	49
	Literatura	51
A	Obsah přiloženého CD	53
B	Dokumentace	54

B.1	Požadavky	54
B.2	Použití programu	54
B.3	Před používáním	55
B.4	Dokumentace zdrojových kódů	55

Kapitola 1

Úvod

Chatbot [2] je v dnešní době stále rozšířenějším způsobem, jak uživatelům ulehčovat práci s počítači nebo různými „chytrými“ zařízeními.

Chatbot může mít různá zaměření a plnit spoustu úkolů.

Kromě toho, že chatbot může usnadňovat ovládání počítačů a jiných „chytrých“ zařízení, slouží také někdy k zábavě, kdy simuluje komunikaci s člověkem a uživatel s ním může tedy konverzovat. Mimo to může být nápomocný při vyhledávání na internetu nebo bývá využit v zákaznické podpoře nebo v internetových obchodech. Zde nahrazuje živé operátory a například poskytuje uživatelům okamžité rady ohledně jejich problému se zařízením, dává doporučení na nabízené produkty nebo poskytuje odpovědi na dotazy na otevírací dobu kamenného obchodu.

Chatboty mohou být nápomocny nejen v práci s počítači, ale i v ostatních každodenních činnostech. Například mohou objednávat jídlo, rezervovat letenky, vyhledávat restaurace nebo obchody v okolí, pomáhat s organizací financí nebo plánováním schůzek, dávat informace o aktuálním počasí nebo pomoci s učením cizího jazyka.

Využití chatbotů je ještě mnohem širší, a proto jsou v tomto odvětví stále prováděny výzkumy, kdy je u některých snaha začlenit do chatbotů silnou umělou inteligenci [7]. Díky ní by chatbot byl schopen, při konverzaci na libovolné téma, vyvolat dojem, že uživatel komunikuje s jiným člověkem.

Základním úkolem chatbotu je na zadaný vstup od uživatele vytvořit relevantní výstup. Komunikace mezi uživatelem a chatbotem probíhá většinou přes textové nebo grafické rozhraní, kde chatbot dostává textový vstup a vrací odpovídající textový výstup. U složitějších typů lze s chatbotem komunikovat i pomocí různých hlasových příkazů.

Tato bakalářská práce se zabývá tvorbou generativního chatbotu, který je postavený na umělých neuronových sítích [4] (dále jen neuronová síť). Chatbot by měl být schopen, na uživatelem zadaný textový vstup, kterým mohou být věty obecné konverzace, vygenerovat odpovídající textový výstup a tím tvořit komunikaci s uživatelem v anglickém jazyce.

Chatbot je implementovaný v jazyce *Python*, v němž je také napsáno mnoho knihoven pro práci s neuronovými sítěmi a zpracování textových dat.

Všechny zdrojové kódy, včetně popisu použití a požadavků, jsou veřejně dostupné na adrese <https://github.com/jirkacechak/chatbot>.

Nejdříve byl vytvořen základní chatbot, který byl následně natrénován na získaných trénovacích datech. Tento chatbot byl otestován metrikou *BLEU* [1, 8], pomocí které je možné si udělat představu, jak kvalitní je natrénovaný model. Poté byl chatbot různými způsoby vylepšen. Například obsahuje celkem tři modely, které je možné trénovat a porovnat. Tento chatbot, respektive všechny tři modely zvlášť, byly také natrénovány a poté

otestovány znovu pomocí metriky *BLEU*. Následně byly jednotlivé modely testovány i uživateli, aby byla získána lepší představa, jak jsou jednotlivé modely kvalitní a jak dokáže chatbot reagovat na reálné vstupy od uživatelů.

Vytvořený chatbot se jmenuje *George*. Během vytváření chatbotu, proběhl pokus chatbot naučit se představit svým jménem, reagovat na pozdravy a loučení zahrnující jeho jméno a naopak upozornit uživatele, jak se jmenuje, pokud by ho například zdravil špatným jménem. Tento pokus byl jen částečně úspěšný, protože k lepším výsledkům by bylo potřeba více vhodných trénovacích dat a další rozsáhlé úpravy již existujících trénovacích dat nebo použít jiné techniky.

Kapitola 2

Základní pojmy

Kapitola se věnuje popisu pojmů, které souvisí s tvorbou chatbotu. Je zde jak obecný popis chatbotu, tak i popis domény, umělé inteligence a umělých neuronových sítí.

2.1 Chatbot

Chatbot¹ [2] (nazýván také bot, chatterbot nebo talkbot) je počítačový program určený k automatizované komunikaci s lidmi. Hlavním úkolem chatbotu je, pro zadaný textový vstup (kontext), vrátit relevantní textový výstup (odpověď). Bývá založený na určitých pravidlech, pomocí kterých vyhledává relevantní odpověď v databázi nebo může používat neuronové sítě, které mu umožňují vytvářet odpovědi na uživatelské vstupy.

S některými chatboty je možné, kromě používání textového nebo grafického rozhraní pro zadávání textu, komunikovat i pomocí zadávání hlasových příkazů.

U komplexnějších chatbotů se očekává i schopnost držet se určitého tématu během delší konverzace, než je jen jeden dotaz a odpověď na něj. Mohou také případně zahrnout do svých odpovědí i různé entity zmíněné v předchozích vstupech, jako jsou například jména lidí nebo názvy existujících míst.

Velkým cílem v tomto odvětví je také tvorba chatbotů, které, při komunikaci s uživatelem, u něj dokáží vyvolat dojem, že dostává odpovědi od jiného člověka a ne pouze od počítačového programu.

Inteligentní stroj

První zaznamenaná představa o systému, který by byl schopen myslet a byl inteligencí na úrovni člověka, vznikla v roce 1950. Tato myšlenka byla publikována v časopisu *Computing Machinery and Intelligence*² [12].

Autorem myšlenky byl britský matematik a logik Alan Mathison Turing, který se zabýval i kryptoanalýzou.

Ve svém článku se Alan Turing zabýval převážně otázkou, zdali někdy budou schopny stroje vůbec myslet jako lidé a dojde tedy ke vzniku skutečné umělé inteligence. Dále zde popsal metodu, pomocí které, bychom měli být schopni poznat, že se daný stroj chová inteligentně.

¹<http://www.wildml.com/2016/04/deep-learning-for-chatbots-part-1-introduction/>

²Výpočetní technika a inteligence (přeloženo z anglického originálu)

Turingův test

Pro ověření, že se konkrétní stroj skutečně chová inteligentně, vymyslel Alan Turing test, který je dnes známý pod názvem Turingův test [7, 12].

Jedná se o úlohu, která je nazývána „Imitation game“. V originální verzi této hry je jeden člověk v roli soudce a ten komunikuje pomocí textových zpráv se dvěma subjekty. Jedním subjektem je muž a druhým žena. Soudce má s oběma subjekty komunikovat a zjistit, který z nich je muž a který žena.

Tento původní test byl upraven tak, že jedním subjektem je člověk a druhým subjektem je stroj. Na začátku soudce neví, který ze subjektů je člověk a který je stroj. Během komunikace s oběma subjekty má identifikovat, který ze subjektů je člověk. Stroj úspěšně splní tento test a je prohlášen za inteligentní, pokud jej soudce označí jako člověka nebo se nedokáže mezi subjekty jednoznačně rozhodnout (oba subjekty se mu jeví jako člověk).

Základní dělení

Chatboty se v základu dělí na chatboty založené na vyhledávání (retrieval based chatbot, rule based chatbot) (viz. 3.1) a generativní chatboty (generative chatbot) (viz. 3.2) [13, 14].

Retrieval based chatboty získávají odpověď na vstup vyhledáním odpovídající odpovědi ve své databázi. U jednoduchých typů se může jednat o pouhé porovnávání vstupního řetězce s řetězci uloženými v databázi. U složitějších typů dochází k vyhledávání klíčových slov a různých vzorů v textu, případně je využita klasifikace pomocí strojového učení.

Generativní chatboty generují odpověď pomocí předem natrénované neuronové sítě (viz. 2.4). K tréninku této neuronové sítě je nutné použití trénovacích dat, podle kterých se chatbot naučí generovat odpověď na zadaný vstup.

Využití

V současnosti se chatboty používají ve všech možných aplikacích a různých platformách. Mezi nejvíce známé patří například chatbot Siri, který byl vytvořen společností Apple pro operační systém iOS. Dále Google Now od společnosti Google, Cortana od společnosti Microsoft, Alexa od společnosti Amazon, a také chatboty v celosvětově používaných aplikacích jako jsou Skype, Slack nebo Facebook Messenger.

Pro chatboty existuje velká škála využití. Používají se jako nákupní rádci a online technické podpory, kde nahrazují živé operátory a poskytují okamžité rady k uživatelským problémům a různá doporučení. Dále usnadňují vyhledávání na internetu a práci s počítačem, případně jinými „chytrými“ zařízeními. Také mohou pomáhat v každodenních činnostech, jako je objednávání jídla nebo letenek a v neposlední řadě umožňují simulovat komunikaci s člověkem.

Někdy se chatboty i zneužívají k nelegálním činnostem. Mohou být použity k šíření nebezpečných počítačových virů, ke zvýšení počtu shlédnutí videí nebo k získání většího počtu odběratelů na YouTube³.

³https://medium.com/@onlim_com/the-history-of-chatbots-2530dd3cdac5

2.2 Doména

Doména určuje, na jaké vstupy od uživatelů je schopen chatbot poskytnout odpovídající výstup a které vstupy naopak neumí vhodně nebo vůbec zpracovat. Chatboty mohou pracovat na uzavřené nebo otevřené doméně⁴.

Uzavřená doména

U uzavřené domény se určí spektrum možných vstupů, na které je model schopen reagovat a poskytovat vhodné odpovědi.

Chatbot, používající uzavřenou doménu, se většinou snaží dosáhnout specifického cíle. Díky tomu neumí takový systém zvládnout všechny možné případy vstupů. Většinou se takové systémy používají například jako technická podpora, kde je cílem poskytnout odpovědi na některý z předdefinovaných problémů. Na vstupy týkající se jiných témat není systém schopen najít nebo vytvořit relevantní odpověď.

Model pracující na uzavřené doméně je většinou tedy jednodušší na návrh, implementaci a případný trénink. Díky tomu se používá častěji než chatboty s otevřenou doménou.

Otevřená doména

Chatboty s otevřenou doménou jsou velice náročné na vytvoření, protože by měly být schopné relevantně reagovat na jakékoliv vstupy od uživatele. Uživatel tedy může vést konverzaci na jemu zvolené téma a nemusí se držet žádného konkrétního, jako je tomu u modelů s uzavřenou doménou.

Systémy s otevřenou doménou se hojně využívají na sociálních sítích, chatovacích klientech nebo webových fórech, kde s uživateli vedou konverzace na každodenní témata nebo reagují na jejich příspěvky.

2.3 Umělá inteligence

Umělá inteligence je simulace lidského myšlení počítačem. Procesy, které toto chování umožňují zahrnují učení, uvažování a sebeopravování, tedy poučení se z vlastních chyb.

S tvorbou chatbotů souvisí hned několik odvětví umělé inteligence. Těmito odvětvími jsou strojové učení, rozpoznávání vzorů a zpracování přirozeného jazyka⁵ [7].

Strojové učení

Strojové učení⁶ se zabývá vývojem programů, které jsou schopny se učit z dat, přizpůsobovat se změnám a chovat se způsobem, který nebyl dopředu programován.

Strojové učení se v dnešní době používá na všech možných místech a zařízeních. Používá se například při vytváření chatbotů, v internetových vyhledávacích jako je Google, v antispamových filtrech, v samořídících autech, při operacích na světových finančních trzích, při strojových překladech textů nebo také pomohly k lepšímu porozumění lidského genomu.

Strojové učení se dělí podle algoritmu, který používají. Dělí se na učení s učitelem, učení bez učitele a posilované učení [7].

⁴<http://www.wildml.com/2016/04/deep-learning-for-chatbots-part-1-introduction/>

⁵<http://searchcio.techtarget.com/definition/AI>

⁶<https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/>

U učení s učitelem máme trénovací data, která se skládají z dvojic vstupních dat a správných výstupních dat, které jsou porovnávány s výsledkem učení.

Při učení bez učitele jsou k dispozici pouze vstupní data. Správná výstupní data k dispozici nejsou a systém, který je založen na učení bez učitele, musí sám v datech objevit to, co je po něm požadováno.

Pokud jsou některá vstupní data bez správných výstupních dat a některá se správnými vstupními daty, tak lze použít zároveň techniky učení s učitelem i učení bez učitele.

Během posilovaného učení, neboli zpětnovazebného učení, dostává systém umělé inteligence, po provedení jedné nebo více akcí, zpětnou vazbu. Na základě této zpětné vazby si systém upraví své naučené informace, případně změní styl učení.

Rozpoznávání vzorů

Rozpoznávání vzorů je odvětvím strojového učení, které má za cíl vyhledávání a rozpoznání vzorů v různých datech. Těmito daty mohou být obrazová, zvuková nebo textová data. Rozpoznávání vzorů se někdy využívá i při programování chatbotů.

Zpracování přirozeného jazyka

Zpracování přirozeného jazyka (tzv. NLP⁷) je proces analýzy a zpracování přirozeného lidského jazyka pomocí specializovaného počítačového programu. Způsob provádění zpracování přirozeného jazyka je založen na strojovém učení (viz. 2.3).

Používá se k rozpoznávání řeči, překladu textu a analýze textu, což může být využito například k detekci emailů se spamem. V neposlední řadě se zpracování přirozeného jazyka může použít při vytváření chatbotů.

2.4 Neuronová síť

Neuronová síť⁸ (jinak také nazývána model) (Obrázek 2.1) je jistý druh výpočetního modelu, který se používá v systémech umělé inteligence při paralelním zpracování různých typů dat. Vstupní data převádí na výstupní data. Ve své podstatě je tento model určen k tomu, aby modeloval způsob provedení určitých úkolů tak, jak by to dělal lidský mozek [4].

Složení

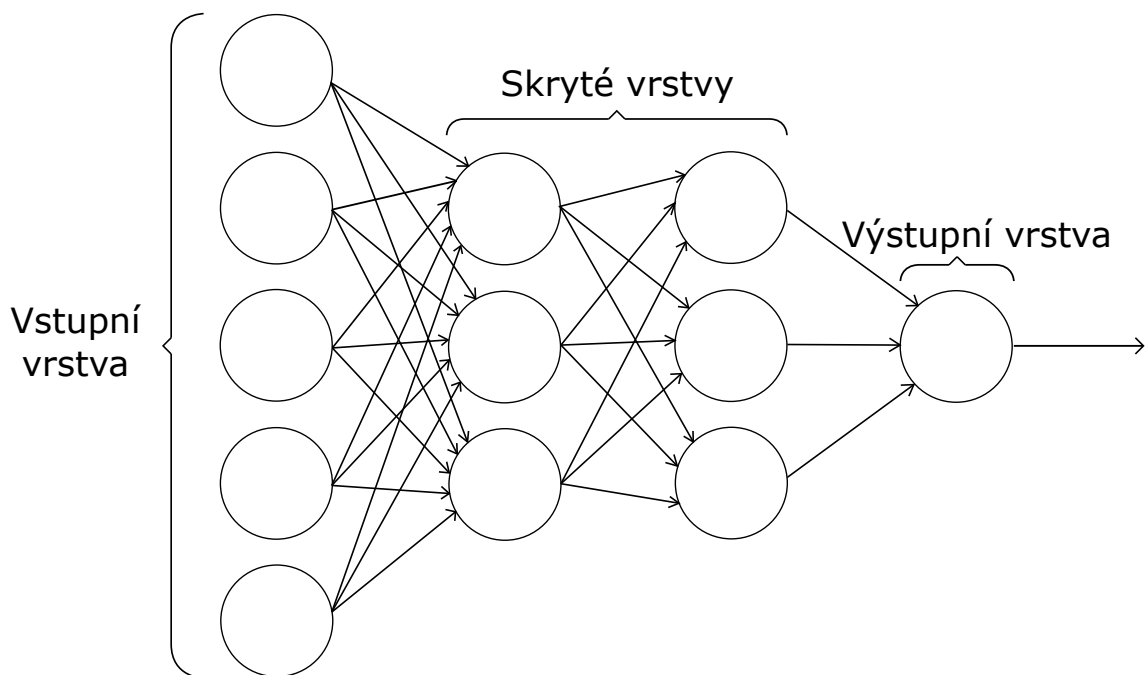
Neuronová síť (Obrázek 2.1) je složena z několika vrstev umělých neuronů (dále jen neuron) [4]. Minimálně musí mít neuronová síť jednu vstupní a jednu výstupní vrstvu. Pokud je to pro řešení daného problému potřeba, tak může neuronová síť obsahovat i jednu nebo více skrytých vrstev, které se všechny nachází mezi vstupní a výstupní vrstvou. Každá vrstva je složena z neuronů, které jsou všechny propojeny se všemi neurony předchozí a následující vrstvy.

Neuron

Každý neuron (Obrázek 2.2) má váhy [4] a bias (sklon) [5], které ovlivňují jeho výstup. Dochází k tomu tak, že vstupy neuronu jsou vynásobeny jeho vahami a dojde k sumarizaci výsledků spolu s bias. Následně je tento výsledek vstupem aktivační funkce [5] daného

⁷Natural Language Processing

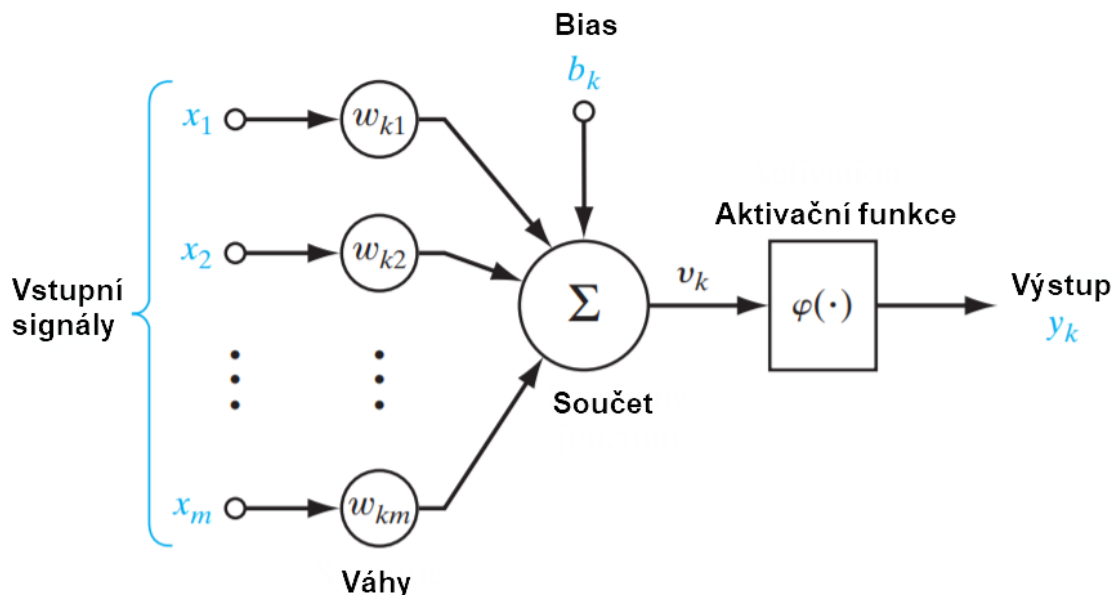
⁸<https://chatbotslife.com/how-neural-networks-work-ff4c7ad371f7>



Obrázek 2.1: Příklad umělé neuronové sítě (konkrétně feed-forward neuronové sítě), která se skládá ze vstupní, výstupní a dvou skrytých vrstev.

Zdroj: <http://neuralnetworksanddeeplearning.com/chap1.html>

neuronu. Dle typu aktivační funkce je provedena transformace jejího vstupu na výstup. Výstup aktivační funkce je i výstupem neuronu a tento výstup může být použit jako vstup dalšího neuronu nebo neuronů.



Obrázek 2.2: Umělý neuron, z kterých se skládá umělá neuronová síť. [4]

Trénink

Aby mohla neuronová síť řešit konkrétní problém, bývá nejdříve natrénována na vhodných trénovacích datech.

Při tréninku neuronové sítě dochází k úpravě vah [4] v neuronech tak, aby se zmenšovala hodnota výstupu ztrátové funkce [5] na trénovacích datech, které se říká ztráta. Ztráta určuje, jak velký je nesoulad mezi očekávaným výstupem a skutečným výstupem neuronové sítě. Pomocí výsledku ztrátové funkce lze sledovat, jestli se natrénovaný model zlepšuje.

K úpravě vah se používá optimalizátor⁹, který má nastaven jako jeden z parametrů learning rate¹⁰ (míra učení). Learning rate určuje rychlost úpravy vah během jednotlivých epoch tréninku. Pokud je nastaven nízký learning rate, tak je optimalizace pomalejší, ale bývá většinou přesnější. Při použití vyššího learning rate dochází k větším úpravám vah natrénovaného modelu, ale změny nemusí být dostatečně přesné.

Přetrénování modelu

Pokud jsou zvoleny u neuronové sítě nevhodné parametry, jako je například moc velký počet skrytých vrstev neuronové sítě, případně trénink neuronové sítě trvá příliš dlouho nebo je prováděn na malém množství trénovacích dat, tak může u neuronové sítě dojít k přetrénování.

Přetrénování znamená, že si neuronová síť zapamatuje trénovací data a je tedy dokonale schopna vytvořit odpovídající výstup na vstup z trénovacích dat. Na druhou stranu ale kvůli přetrénování ztrácí neuronová síť schopnost generalizovat [4] a tedy vygenerovat odpověď na data, která nebyla použita při tréninku.

Většinou se přetrénování dá předejít sledováním přesnosti neuronové sítě při testování neuronové sítě na validačních datech, což jsou vzorky, které nebyly použity při trénování neuronové sítě. Neuronová síť je tedy musí generalizovat, což je schopna, jen pokud není přetrénována.

Použití

Neuronové sítě se využívají v systémech umělé inteligence (viz. 2.3). Používají se například k rozpoznávání různých vzorů ve zvukových nebo obrazových datech, jako je například rozpoznávání předmětů nebo lidí na fotografiích. Také se používají při filtrování spamu mezi emailovými zprávami, vytváření chatbotů a při strojových překladech.

Feed-forward neuronová síť

U feed-forward neuronových sítí¹¹ [6] (Obrázek 2.1) prochází informace neuronovou sítí pouze jedním směrem a žádným neuronem, v neuronové síti, neprojde dvakrát. Neexistuje zde tedy smyčka u žádného neuronu. Vstupní vzorky jsou v síti transformovány na výstupy, mapují vstupní data do kategorií a umí také rozpoznat vzory v datech.

⁹<https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/>

¹⁰<https://towardsdatascience.com/understanding-learning-rates-and-how-it-improves-performance-in-deep-learning-d0d4059c1c10>

¹¹<https://deeplearning4j.org/lstm.html>

Dense

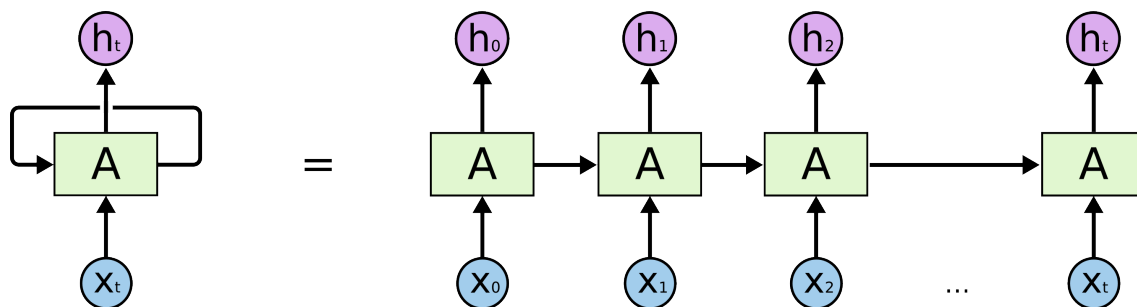
Typickým představitelem feed-forward neuronové sítě nebo vrstvy je v knihovně *Keras* (viz. 4.1) vrstva *Dense*¹². Tato vrstva slouží k transformaci vstupních dat na data výstupní pomocí některé aktivační funkce.

Rekurentní neuronová síť

Rekurentní neuronová síť¹³ (*RNN*¹⁴) [3, 6] je typ neuronové sítě, která byla vytvořena za účelem hledání souvislostí a vzorů v určitém kontextu nebo sekvenci dat, kterými může být prostý text, mluvené slovo, číselné řady, obrázky a mnoho dalších.

U rekurentních neuronových sítí, na rozdíl od feed-forward sítí, je rozhodnutí, které rekurentní síť udělá v čase t ovlivněno rozhodnutím, které udělala v čase $t - 1$, tedy v předchozím kroku.

Rekurentní síť má tedy dva vstupy, současný a vstup získaný naposledy, a oba dva určují, jaká data budou na výstupu. Od klasických neuronových sítí se tedy liší smyčkou (Obrázek 2.3), která představuje zpětnou vazbu a napojuje výstup na vstup v dalším kroku. Díky tomu ovlivňuje výstup nejen daný vstup, ale také předchozí vstupy a výstupy.



Obrázek 2.3: Rekurentní neuronová síť po odrolování (jednotlivé kroky v čase), kde x jsou jednotlivé vstupy a h jsou výstupy.

Zdroj: <https://medium.com/@jianqiangma/all-about-recurrent-neural-networks-9e5ae2936f6e>

Výhody

Díky přidání paměti mohou rekurentní sítě využít informace ze sekvence dat k provedení úkolů, které feed-forward síť nezvládnou.

Informace získaná ze sekvence dat je uložena ve skrytém stavu neuronové sítě a ovlivňuje výsledek v mnoha krocích. Vyhledává korelace mezi událostmi, které mohou být odděleny mnoha kroky. Tyto korelace se nazývají dlouhodobé závislosti (long-term dependencies), protože událost, která se nachází v určitém kroku, závisí na jedné nebo více událostech, které jí předcházely.

¹²<https://keras.io/layers/core/>

¹³<https://deeplearning4j.org/lstm.html>

¹⁴Recurrent Neural Networks

Nevýhody

Největší nevýhodou rekurentních neuronových sítí je, že oproti feed-forward sítím je velice náročné je natrénovat. Převážně je velice těžké je naučit efektivně využívat jejich schopnosti učít se z dlouhých sekvencí a získávat závislosti mezi událostmi oddělenými mnoha kroky.

Dále je také nevýhodou, že u nich může docházet k naučení věcí, které jsme neočekávali a většinou ani nechtěli. Může být tedy velice těžké donutit rekurentní neuronovou síť, aby se učila věci, které od ní očekáváme a někdy je tedy nutné s ní experimentovat a zjistit, co funguje a co ne.

LSTM

*LSTM*¹⁵ blok nebo síť je druh rekurentní neuronové sítě¹⁶.

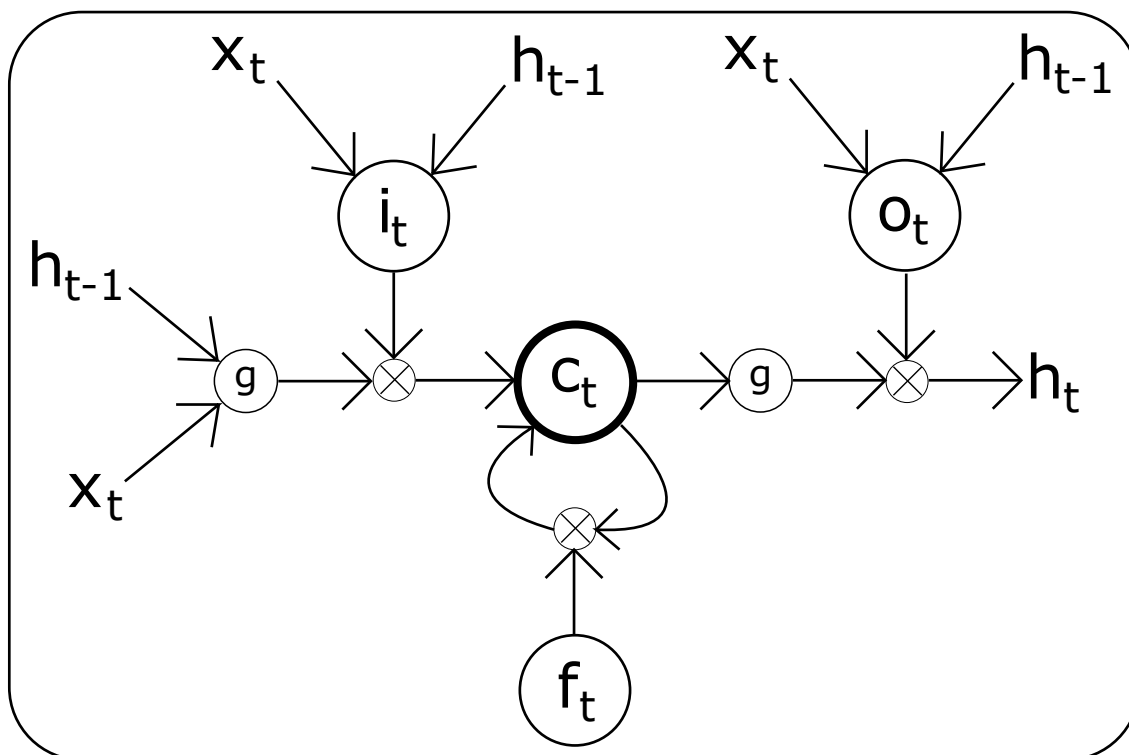
LSTM bloku (Obrázek 2.4) obsahuje, mimo hlavní tok neuronové sítě, informace, které jsou uloženy v buňce, do které lze data zapisovat a číst z ní. *LSTM* blok se rozhoduje, kdy umožní čtení, zápis (zapamatování) nebo vymazání (zapomenutí) dat pomocí otevírání a zavírání příslušných bran. Tyto brány jsou analogové a jsou složeny z vrstvy *sigmoid* v rozsahu 0 až 1. Tyto brány se, stejně jako buňky neuronové sítě, na základě signálů, které dostávají, rozhodují, zdali budou průchod informace blokovat nebo ho povolí. Toto filtrují pomocí svých vlastních vah, které se upravují při učení neuronové sítě. Blok se tedy učí, kdy umožní vstup, výstup nebo vymazání informace na základě iterativního procesu hádání, zpětné propagace chyby a příslušného upravení vah.

Díky možnosti uložení informací *LSTM* umožňuje uchovat chybu, kterou lze následně zpětně propagovat jednotlivými vrstvami neuronové sítě.

LSTM neuronové sítě by měly být schopny vyhnout se některým problémům klasických rekurentních neuronových sítí a mohou být tedy vhodnější pro strojové překlady, klasifikaci textu, tvorbu chatbotů a různé další úlohy.

¹⁵Long-short term memory

¹⁶<http://colah.github.io/posts/2015-08-Understanding-LSTMs/>



Obrázek 2.4: Diagram složení *LSTM* bloku, kde c_t je buňka, i_t je vstupní brána, o_t je výstupní brána a f_t je brána k zapomínání. Buňka i jednotlivé brány jsou ve stavu v kroku t . Vstup *LSTM* bloku v kroku t je označen jako x_t a výstup *LSTM* bloku v kroku t je označen jako h_t . Na vstup *LSTM* bloku, vstupní brány a výstupní brány jde kromě vstupu x_t i výstup *LSTM* bloku v kroku $t - 1$, který je označen jako h_{t-1} . Vstup a výstup *LSTM* bloku v předchozím kroku jdou na vstup aktivační funkce g . Výstup aktivační funkce, která je na vstupu *LSTM* bloku, je násoben výstupem vstupní brány i_t . Výsledek jde do buňky c_t . Stav této buňky může být ovlivněn vynásobením výstupem brány f_t , která slouží k zapomínání. Výstup buňky c_t jde na vstup aktivační funkce, jejíž výstup je vynásoben výstupem výstupní brány o_t . Výsledek je výstupem *LSTM* bloku.

Zdroj:

https://www.researchgate.net/figure/Diagram-of-LSTM-block_fig1_307889445

Kapitola 3

Základní dělení chatbotů

Chatboty se v základu dělí na dva typy a to podle způsobu, jakým získávají odpověď na zadaný textový vstup. Jedná se o retrieval based chatboty [13, 14] a generativní chatboty [13, 14].

Retrieval based chatbot v reálných případech pracuje pouze na uzavřené doméně (viz. 2.2). Jedná se o chatbot, který je, oproti ostatním, nejjednodušší na implementaci a vytvoření (Obrázek 3.1).

Generativní chatboty většinou také pracují na uzavřené doméně, ale bývají náročnější na vytvoření, jelikož je většinou k jejich vytvoření potřeba využití vhodné neuronové sítě (viz. 2.4), která je trénována na trénovacích datech (Obrázek 3.1).

Složitější generativní chatboty by měly být schopné pracovat i na otevřené doméně (viz. 2.2), kdy by měly být schopné adekvátně reagovat na jakékoliv vstupy od uživatele. Tento typ chatbotu je nejnáročnější na vytvoření, jelikož se v ideálním případě jedná o silnou umělou inteligenci. Takový chatbot by měl být schopen se vypořádat s každou situací, která může nastat (Obrázek 3.1).

Poslední možností, která se nabízí, je retrieval based chatbot, který pracuje na otevřené doméně, což ale není možné. Takový chatbot by musel mít nekonečnou databázi vstupů a možných výstupů, aby pokryl jakoukoliv situaci, která by při konverzaci s uživatelem mohla nastat (Obrázek 3.1).

3.1 Retrieval based chatbot

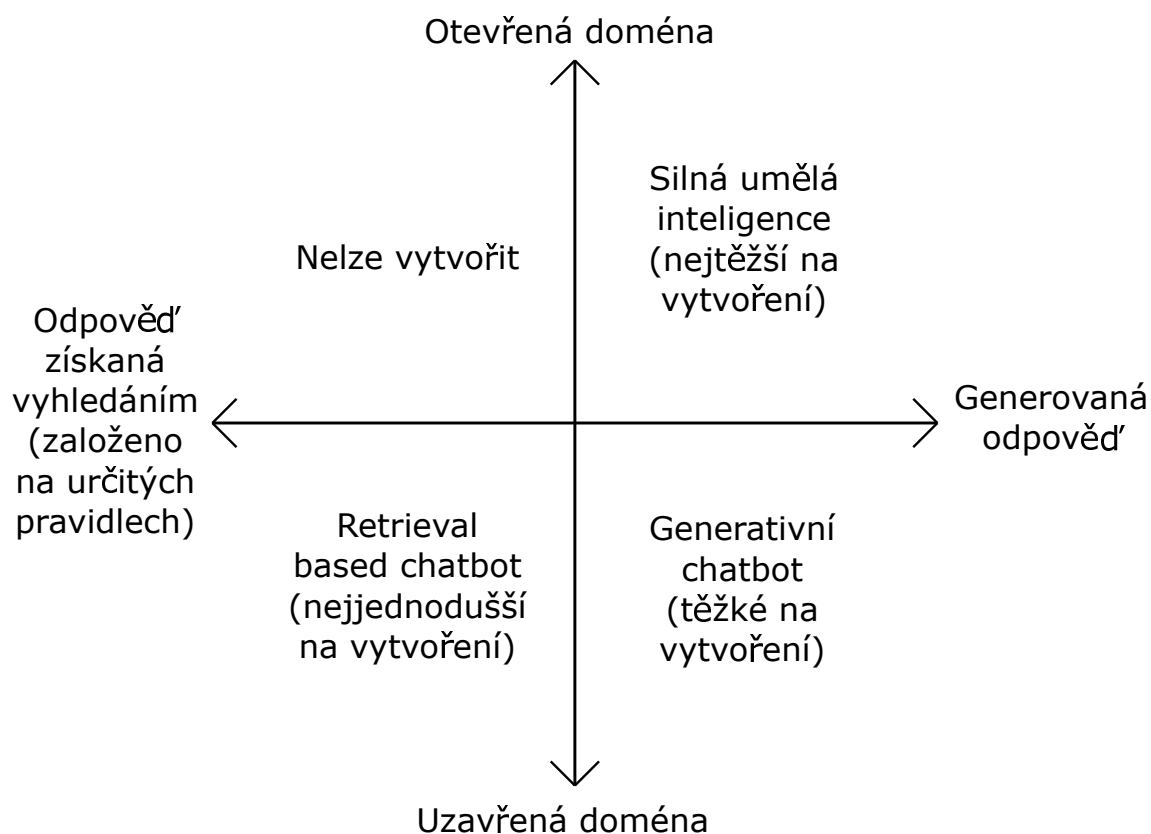
Retrieval based¹ (rule based) [13, 14] chatbot je mnohem jednodušší na návrh a implementaci, protože používá určitou databázi již předdefinovaných dotazů a odpovědí na ně.

Chatboty mohou používat různé databáze. Některé chatboty mohou mít databázi vět vlastní, která bývá u jednodušších typů pevně dána tím, co do ní bylo vloženo před začátkem používání chatbotu.

U složitějších typů může být chatbot schopen, během svého fungování a komunikace s člověkem, svou databázi kontextů a odpovědí na ně automaticky rozšiřovat podle nových vstupů od uživatele. V tomto případě, ale může docházet i k ukládání nevhodných vět a informací, když například uživatel zadá úplně nesmyslnou odpověď na otázku od chatbotu.

Pokud chatbot nemůže mít vlastní databázi, tak může být v dnešní době využito vyhledávání relevantní odpovědi na internetu. Odpověď může být vyhledávána na konkrétních

¹<http://www.wildml.com/2016/07/deep-learning-for-chatbots-2-retrieval-based-model-tensorflow/>



Obrázek 3.1: Druhy chatbotů v závislosti na způsobu získání odpovědi a doméně.

webových stránkách nebo pomocí internetových vyhledávačů. V ideálním případě, by takový chatbot mohl pracovat i na otevřené doméně, jelikož by díky internetu měl přístup k velké části lidských znalostí.

Samotný chatbot funguje tak, že používá určitý typ heuristiky a pravidel pro výběr vhodné odpovědi na zadaný vstup od uživatele. V nejjednodušších případech se může jednat pouze o porovnání řetězců vstupu a jednotlivých kontextů uložených v databázi chatbotu a následně se vybere vhodná odpověď. U složitějších typů dochází k vyhledávání klíčových slov a různých vzorů v textu. Také lze ale tyto chatboty udělat mnohem komplexnější a to například využitím klasifikace pomocí strojového učení.

Typickými představiteli tohoto typu jsou například první vzniklý chatbot ELIZA [10], který vytvořil v roce 1966 německý profesor matematiky na MIT Joseph Weizenbaum nebo také chatbot A.L.I.C.E.² [10].

Chatbot A.L.I.C.E vycházel z chatbotu ELIZA. Textový vstup zpracovával pomocí heuristických vzorců a tím vytvářel konverzaci. Pravidla konverzace tohoto chatbotu udává jazyk AIML³, který je založen na schématu jazyka XML⁴.

V dnešní době existuje také velmi oblíbený a na internetu dostupný chatbot Cleverbot⁵.

²Artificial Linguistic Internet Computer Entity

³Artificial Intelligence Markup Language

⁴eXtensible Markup Language

⁵<http://www.cleverbot.com>

Výhody

Určitě velkou výhodou tohoto modelu je, že na rozdíl od generativních chatbotů nedělá gramatické chyby, pokud se tedy chyby nenachází v již předdefinovaných odpovědích. Další výhodou je, že je možné chatbot většinou okamžitě používat po implementaci, protože jej není nutné trénovat, jako generativní chatbot. Pokud má chatbot za úkol pouze řešit určitý problém, jako je například odpovídání na jednoduché dotazy zákazníků v internetovém obchodě, tak není nutné velké množství dat.

Nevýhody

Na rozdíl od generativních chatbotů tento typ negeneruje žádný nový text. Díky tomu je největší nevýhodou tohoto typu chatbotu, že většinou nedokáže najít vhodnou odpověď na neznámý vstup od uživatele, protože ho nemá uložený ve své databázi. Pokud má být tento typ chatbotu schopen vést konverzaci na více témat, tak je nutné, aby databáze obsahovala velké množství kontextů a odpovědí.

Použití

I přes zmíněné nedostatky tohoto typu, dochází k jeho hojnému využívání jako online technické podpory nebo nákupního poradce v online obchodech. Zde pracuje chatbot v uzavřené doméně a má předdefinované odpovědi na dotazy týkající se daného problému, který má řešit. Jeho jedinou úlohou je totiž dostat se ke svému cíli, kterým je vyřešení uživatelského problému. V případě online obchodu se může jednat o odpovědi na dotazy na otevírací dobu kamenného obchodu, kontakty nebo na vyhledávání různých druhů zboží. Na tyto dotazy dokáže vhodně odpovědět, ale například na uživatelské dotazy na počasí nebo politiku již ne.

Použití tohoto typu chatbotu na otevřené doméně není možné (Obrázek 3.1), jelikož by musel mít ve své databázi předdefinované odpovědi na všechny možné vstupy od uživatele.

Metrika

K evaluaci modelu, tedy zjištění, jak je dobře retrieval based chatbot schopen odpovídat na zadané textové vstupy, se využívají různé metriky. Nejčastěji využívanou metrikou je $recall@k$ ⁶.

Při používání této metriky necháme model vybrat k odpovědí z 10 předem určených, kde je vždy pouze jedna odpověď správná a zbylých 9 jsou špatné odpovědi. Těchto 9 špatných odpovědí jsou tzv. distraktory, které se vybírají náhodně. Pokud je správná odpověď mezi vybranými, je tento test označen za úspěšný.

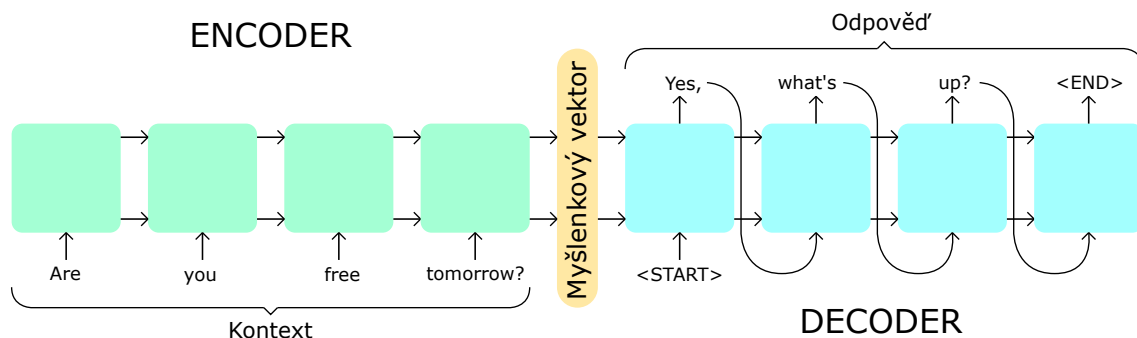
Při zvolení většího k je pravděpodobnost správného výběru větší a naopak, pokud je za k dosazeno číslo 1, je výběr odpovědi nejnáročnější, protože musí být vybrána pouze správná odpověď.

⁶<http://www.wildml.com/2016/07/deep-learning-for-chatbots-2-retrieval-based-model-tensorflow/>

3.2 Generativní chatbot

Generativní chatbot [13, 14], na rozdíl od retrieval based chatbotu (viz. 3.1), nepoužívá předdefinované odpovědi. Na základě zadaného vstupu od uživatele generují úplně novou odpověď.

Většinou je tento typ založen na principu, který se nazývá *sequence-to-sequence* [6, 11]. Vstup uživatele jde na vstup *encoderu* [6, 11], který jej zpracuje do vektoru určité délky, který poté *decoder* [6, 11] převede na výstup, kterým je odpověď pro uživatele (Obrázek 3.2).



Obrázek 3.2: *Sequence-to-sequence* princip, kde je kontext, který jde na vstup *encoderu*, převeden do vektoru, který poté *decoder* převede na výstup, kterým je odpověď.

Zdroj:

<http://www.wildml.com/2016/04/deep-learning-for-chatbots-part-1-introduction/>

V praxi se to provádí tak, že se na vstup vloží kontext a část odpovědi (na začátku pouze token, který představuje začátek odpovědi). Model vygeneruje další token v odpovědi, který se přidá k odpovědi. Nová odpověď se dá zase na vstup modelu. Takto se pokračuje, dokud nemá vytvořená odpověď maximální možnou délku nebo v některých případech, dokud není dosažen token ukončující odpověď.

Oproti retrieval based chatbotům je tento typ chatbotu náročnější na návrh a implementaci.

Výhody

Nespornou výhodou tohoto typu je schopnost generování zcela nové odpovědi bez předdefinovaných kontextů a odpovědí. Díky tomu dokáží vytvořit celkem relevantní odpověď i na vstup, pro který nebyl chatbot trénován. Odpověď je relevantnější pokud chatbot pracuje na uzavřené doméně (viz. 2.2), jelikož v tomto případě pro trénink stačí méně trénovacích dat, zaměřující se na konkrétní téma nebo související s určitým problémem.

Nevýhody

Kvůli generování zcela nových odpovědí mají tendenci chatboty tohoto typu dělat gramatické chyby. Gramatické chyby se více objevují s rostoucí délkou věty. Dále je při dlouhém vstupu od uživatele většinou mnohem náročnější vygenerovat vhodnou odpověď tak, aby dávala smysl. Někdy se totiž může stát, že se chatbot začne opakovat nebo v polovině dlouhé věty změni její význam. Pro trénink je také potřeba velké množství kvalitních trénovacích dat a samotný trénink je velmi časově a výpočetně náročný.

Použití

I přes výše zmíněné nevýhody má tento typ díky svým výhodám velký potenciál. Stejně jako retrieval based chatboty (viz. 3.1) mohou pracovat na uzavřených doménách (viz. 2.2) a využívat se jako technická nebo zákaznická podpora, nákupní rádce nebo online asistent při řešení různých problémů a poskytování informací.

Často se však díky velkým možnostem ve vytváření odpovědí na kontext, pro který nebyly trénovány, využívají ke konverzacím s uživateli na různá témata například na sociálních sítích, online diskuzních fórech a v chatovacích klientech. V těchto případech se někdy snaží fungovat i na otevřené doméně (viz. 2.2) a jejich odpovědi nejsou vázány na konkrétní téma nebo řešení pouze jednoho problému.

Metrika

Stejně jako u retrieval based chatbotu (viz. 3.1), je i u generativního chatbotu možné použít různé metriky k evaluaci kvality jeho odpovědí. Jednou z nich je i metrika *BLEU*⁷ [1,8].

Tato metrika se nejčastěji používá pro strojové překlady textů do jiných jazyků, ale je možné ji použít i pro zjištění kvality vygenerovaných odpovědí u generativního chatbotu.

Jedná se o algoritmus, který je schopen určit kvalitu textu, který byl vygenerován (tzv. kandidát), oproti dodanému jednomu nebo více referenčním textům. Vygenerovaný text je tím kvalitnější, čím víc je podobný referenčnímu textu. Nejčastěji se vygenerovaný text porovnává s textem, který byl vytvořen člověkem.

Metrika nebere v potaz případné gramatické chyby a tedy, zdali je celá sekvence gramaticky správně. Není posuzována ani smysluplnost textové sekvence.

Výstupem algoritmu je reálné číslo představující skóre v rozmezí 0 až 1. Pokud je skóre 0, tak si vygenerovaná a referenční textová sekvence nejsou vůbec podobné. Naopak, pokud je skóre rovno přesně číslu 1, tak jsou dané textové sekvence totožné.

Při výpočtu skóre se pro každý token (slovo, interpunkční znaménko apod.) v referenčním textu získá počet jeho výskytů m_{max} . Pro výpočet skóre P pro konkrétní token se pak použije vzorec $P = \frac{m}{w_t}$, kde m je počet výskytů daného tokenu z kandidáta v referenčním textu, který je ořezán na m_{max} a w_t je celkový počet tokenů v kandidátovi.

Tabulka 3.1: Příklad kandidáta a referenčního textu

Kandidát	the	the	the	the	the	the	the
Referenční text	the	cat	is	on	the	mat	

Ve výše uvedeném příkladu se nám vyskytuje slovo *the*, konkrétně $m_w = 7$. Počet výskytů slova *the* v referenčním textu je $m_{max} = 2$. Po ořezání je tedy $m = 2$. Kandidát má celkem 7 slov. Po dosazení do vzorečku vychází tedy $P = \frac{m}{w_t} = \frac{2}{7}$.

BLEU ale počítá skóre, kvůli zvýšení přesnosti, i pomocí výskytu *n-gramů* [6], což je sekvence n sousedících položek v textu, případně řeči. Položky mohou být, dle potřeby různá slova, slabiky nebo jen písmena a další znaky. V našem případě se jedná obecně o tokeny, ze kterých se skládá textová sekvence. V algoritmu *BLEU* se obvykle používají *n-gramy* o velikosti 1 až 4. Jedná se tedy o *n-gram* o velikosti 1, což je tzv. *unigram*, *n-gram* o velikosti 2 je tzv. *bigram*, *n-gram* o velikosti 3 je tzv. *trigram* a *n-gram* o velikosti 4 je tzv. *four-gram*.

⁷Bilingual Evaluation Understudy

Nevýhodou *BLEU* algoritmu může být, že dochází k upřednostňování krátkých textových sekvencí, u kterých bývá získáno mnohem vyšší celkové skóre.

Kapitola 4

Implementace

Tato kapitola popisuje krátce použité technologie při implementaci, a také jednotlivé části, ze kterých se chatbot skládá, data se kterými pracuje a jakým způsobem. Je zde jak obecný popis, tak popis rozdílných částí pro základní verzi chatbotu (viz. 4.10) a vylepšenou verzi chatbotu (viz. 4.11), která obsahuje již tři modely.

4.1 Použité technologie

Chatbot je implementován v jazyce *Python 3.6.3* s použitím vhodných knihoven pro práci s neuronovými sítěmi (viz. 2.4), textem a poli, jako jsou *Keras*¹, *nlTK* nebo *numpy*.

Keras

Pro implementaci chatbotu je použita knihovna *Keras*, která je implementována v jazyce *Python* a běží na knihovnách *TensorFlow*, *CNTK* a *Theano*. Knihovna je kompatibilní s jazykem *Python 2.7-3.6*. Výpočty mohou být prováděny pomocí procesoru nebo i pomocí grafické karty.

Jedná se o vysokoúrovňové rozhraní pro programování aplikací, které slouží pro práci s neuronovými sítěmi, které mohou být typu feed-forward (viz. 2.4) nebo i rekurentní (viz. 2.4). Díky větší abstrakci stačí pro vytvoření neuronové sítě mnohem méně kódu než je tomu u jiných knihoven se stejným nebo podobným zaměřením.

Jedná se tedy o vhodný nástroj pro vytvoření chatbotu postaveného na neuronových sítích.

4.2 Volitelné parametry a módy chatbotu

Chatbot má tři módy, kterými jsou trénink (viz. 4.7), testování (viz. 4.8) a chatování (viz. 4.9). Běh programu v módech testování a chatování je umožněn až po natrénování neuronové sítě a uložení takto získaných vah do souboru, které jsou poté před testováním nebo chatováním načteny do modelu.

Pokud je chatbot spuštěn v módu trénink, tak dochází k tréninku chatbotu na trénovacích datech. V módu testování je chatbot testován pomocí metriky *BLEU* (viz. 3.2) na testovacích datech a v módu chatování je uživateli umožněno vést konverzaci s chatbotem v konzole nebo v GUI.

¹<https://keras.io>

Některé parametry, jako je maximální množství trénovacích dat nebo výběr módu, lze volit pomocí odpovídajících argumentů příkazové řádky. Ostatní parametry lze většinou upravit v souboru `constants.py`, ve kterém se nachází všechny konstanty použité v kódu programu.

4.3 Slovník

Slovník slouží k jednodušší manipulaci s tokeny a k omezení počtu tokenů, které se mohou vyskytnout v trénovacích datech, případně při generování odpovědi. Slovník je uložen v příslušném souboru a používá se během tréninku, testování i pro chatování s chatbotem.

Slovník obsahuje tokeny, kterými jsou v první řadě anglická slova, ale také interpunkční znaménka a v neposlední řadě dva speciální tokeny, kterými jsou *BOS*², představující začátek odpovědi a *EOS*³, představující konec odpovědi.

Slovník byl získán z již existujícího zdroje⁴, kde obsahoval 6999 tokenů.

Po načtení slovníku je k němu přidán zvolený token, nahrazující neznámé tokeny, které se vyskytují v trénovacích datech nebo v kontextech získaných jako vstup od uživatele během konverzace s chatbotem. Tímto tokenem je slovo *something*.

4.4 Trénovací data

Data pro trénink se nachází ve dvou souborech. Jeden soubor obsahuje na jednotlivých řádcích kontext a druhý soubor obsahuje na odpovídajících řádcích odpověď na kontext v prvním souboru. Jedná se o dialogy získané z filmových titulků, sociální sítě apod. Jsou zde jak krátké, někdy i jednoslovné, sekvence, tak textové sekvence obsahující několik desítek slov.

Výše zmíněná data pro trénink chatbotu byla získána z již existujícího chatbotu s názvem Victor⁵. Konkrétně se jedná o dva páry souborů tvořené souborem s kontexty a souborem s příslušnými odpověďmi.

Později bylo více dat získáno z dalšího existujícího chatbotu s názvem ChatLearner⁶. Zde se jednalo o soubory s kontexty i příslušnými odpověďmi v jednom souboru. Data z těchto souborů musela být upravena do příslušné podoby, aby mohla být použita při trénování chatbotu.

Jelikož se jedná, mimo jiné, o data z filmových titulků, tak se zde nachází různé nevhodné tokeny, jako jsou tagy označující vzhled textu apod. Tyto přebytečné tokeny byly, pomocí funkcí využívajících regulární výrazy, alespoň částečně odstraněny a textové sekvence byly takto ještě před tréninkem chatbotu upraveny do vhodné podoby.

Poté byla data promíchána po jednotlivých párech tvořících kontext a odpověď a znovu uložena do souborů.

Příklady trénovacích dat

Kontext: There.

Odpověď: Where?

²Beginning Of Sentence

³End Of Sentence

⁴https://github.com/oswaldoludwig/Seq2seq-Chatbot-for-Keras/blob/master/vocabulary_movie

⁵<https://github.com/codedak/chatbot>

⁶<https://github.com/bshao001/ChatLearner>

Kontext: Hi.
Odpověď: Looks like things worked out tonight, huh?
Kontext: She okay?
Odpověď: I hope so.
Kontext: Did you change your hair?
Odpověď: No.
Kontext: Where did he go? He was just here.
Odpověď: Who?
Kontext: How many people were in your old school?
Odpověď: Thirty-two.
Kontext: But she can't go out with you because her sister is this insane head case and no one will go out with her. right?
Odpověď: Does this conversation have a purpose?
Kontext: He's pretty!
Odpověď: Okay! I wasn't sure
Kontext: What've you got for me?
Odpověď: I've retrieved certain pieces of information on Miss Katarina Stratford I think you'll find helpful.
Kontext: What about him?
Odpověď: You wanna go out with him?
Kontext: Where ya goin?
Odpověď: Away.
Kontext: How'd you do it?
Odpověď: Do what?
Kontext: Hey.
Odpověď: Are you lost?
Kontext: Nope - just came by to chat
Odpověď: We don't chat.
Kontext: But she can't go out with you because her sister is this insane head case and no one will go out with her. right?
Odpověď: Does this conversation have a purpose?
Kontext: You think this'll work?
Odpověď: No fear.
Kontext: What'd he say?
Odpověď: Who cares?
Kontext: I appreciate your efforts toward a speedy death, but I'm consuming. Do you mind?
Odpověď: Does it matter?
Kontext: Pick you up Friday, then
Odpověď: Oh, right. Friday.
Kontext: Okay?
Odpověď: I'm fine. I'm
Kontext: You're not okay.
Odpověď: I just need to lie down for awhile
Kontext: Why 're you doing this?
Odpověď: I told you
Kontext: No.
Odpověď: No what?

Kontext: It's just something I had. You know

Odpověď: Oh huh

Generování dat pro naučení chatbot se představit

Kromě dostupných dat z internetu, byla použita data, která byla vygenerována pomocí vytvořeného skriptu.

Vygenerovaná data byla, mimo jiné, použita při snaze naučit chatbot se představit, odpovědět na pozdravy a loučení obsahující jeho jméno *George*. Také byla snaha naučit chatbot, pokud byl osloven jiným jménem, než svým vlastním, aby uživatele upozornil, že se jmenuje jinak, případně se znovu představil.

Kromě generování dat, bylo součástí tohoto pokusu i, za pomoci vytvořeného skriptu, odstranění některých kontextů a odpovědí, které by mohli negativně ovlivnit tuto snahu, z již používaných trénovacích dat. Jednalo se především o některé věty s představením, dotazy na jméno apod. Vět bylo odstraněno relativně hodně, ale určitě ne všechny tohoto typu, které se v trénovacích datech vyskytují. Také bylo u některých vět nahrazeno jméno, které se zde vyskytovalo, jménem *George*.

Při generování dat bylo použito prvních tisíc křestních jmen, která se nacházela v použitém seznamu anglických křestních jmen⁷. Dále byly použité různé způsoby pozdravů, loučení, představení apod. v anglickém jazyce, které byly náhodně voleny během generování dat.

Pro generování odpovědí bylo použito, jako u vygenerovaných kontextů, jméno chatbotu, a také náhodně zvolené pozdravy, loučení apod.

Celkem bylo takto vygenerováno více než 2000 různých kontextů a odpovědí, které zahrnují pozdravy, loučení, představení apod.

Tato snaha byla jen částečně úspěšná u všech modelů. Chatbot se umí někdy představit a reaguje na pozdravy a loučení, ale při oslovení jiným jménem, než je jeho, nedokáže většinou uživatele upozornit, že se jmenuje jinak nebo vypíše upozornění na jiný vstup, než by měl.

Pro získání lepších výsledků by bylo potřeba více upravit trénovací data a získat větší množství vhodných trénovacích dat nebo použít jiné techniky.

Příklady vygenerovaných trénovacích dat

Kontext: good afternoon! my name is daniel

Odpověď: it is nice to meet you, call me George.

Kontext: good evening, alfred, how are you doing today?

Odpověď: hi, but i'm George

Kontext: what is your name?

Odpověď: my name is George

Kontext: hey, George

Odpověď: hello!

Kontext: see you later, George

Odpověď: goodbye

Kontext: hi, edmund

Odpověď: hey hey! please, call me George

Kontext: bye, gregg

⁷<http://deron.meranda.us/data/census-derived-all-first.txt>

Odpověď: have a good day! but for the next time... please, call me George

Kontext: is your name George?

Odpověď: it's me. can you tell me your name?

Kontext: are you terry?

Odpověď: no, call me George

4.5 Word embedding

Některé neuronové sítě dokáží pracovat i s klasickou podobou textu, ale pro lepší zachycení významu a snadnější práci, je vhodné převést jednotlivá slova a další tokeny (což mohou být například interpunkční znaménka) textových sekvencí do nějaké číselné reprezentace.

K číselné reprezentaci tokenů slouží například word embedding⁸, který umožňuje vektorovou reprezentaci slov. Jedná se tedy o kódování slov a dalších tokenů na vstup neuronové sítě.

Word embedding dále také umožňuje určovat, jak jsou si jednotlivá slova a další tokeny významově blízká. K tomuto účelu je nutné reprezentovat slova pomocí číselných vektorů, protože spolu skoro vždy bývají každá dvě slova příbuzná vícero způsoby. Například slova *man* a *woman* jsou vzájemně významově podobná, jelikož se v obou případech jedná o lidské bytosti. Na druhou stranu se jedná o slova, která spolu stojí v protikladu, jelikož jsou to zástupci opačných pohlaví lidské rasy. Z toho vyplývá, že přesnost určení významově blízkých slov závisí mimo jiné i na velikosti dimenze vektoru. Čím vyšší dimenze vektoru, tím je i určení významově podobných slov přesnější. Čím jsou si slova významově bližší, tím je Euklidovská vzdálenost vektorů menší (kosinová podobnost).

Pokud jsou slova významově úplně stejná, tak je to vyjádřeno úhlem, který má přesně 0 stupňů, což odpovídá hodnotě 1 a znamená to, že se vektory překrývají. Pro úhly v rozmezí 0 až 360 stupňů mezi dvěma vektory je hodnota menší než 1. Úhel o velikosti 90 stupňů odpovídá hodnotě 0, což znamená, že dané dva tokeny, které jsou reprezentovány příslušnými dvěma vektory, si navzájem nejsou vůbec podobné. Úhel 180 stupňů odpovídá hodnotě -1. Tato hodnota pro dané dva vektory znamená, že tokeny, které tyto vektory reprezentují, jsou spolu v protikladu.

Vektorové reprezentace slov bývají nejčastěji uloženy v binárním nebo textovém slovníku. Každý vektor je tvořen stejným počtem reálných čísel. Počet slov ve slovníku závisí na zvoleném jazyku. Čím je slovník obsáhlejší, tím je i větší pravděpodobnost nalezení významově podobného slova. U kvalitních algoritmů pro tvorbu vektorových reprezentací slov, může někdy metrika určit jako významově podobné slovo danému slovu i takové, které většina lidí ani nezná nebo ho vůbec nepoužívá, ale i tak se jedná o relevantní příbuzné slovo.

Jedny z nejčastěji používaných modelů, pro tvorbu vektorové reprezentace slov, jsou *Word2Vec* a *GloVe*.

Word2Vec

*Word2Vec*⁹ je dvouvrstvá neuronová síť, která bere jako vstup celý textový korpus a na výstupu vrátí vektorové reprezentace slov z textového korpusu, který byl na jejím vstupu.

⁸<https://deeplearning4j.org/word2vec.html>

⁹<https://deeplearning4j.org/word2vec.html>

GloVe

*GloVe*¹⁰ [9] je algoritmus pro získávání vektorových reprezentací slov, který je založený na učení bez učitele. Vektory jsou pomocí algoritmu získávány na základě podobnosti jednotlivých slov, která se určuje podle souhrnných globálních statistik jejich vzájemného výskytu v různých textových korpusech.

Implementovaný chatbot používá ke kódování slov předtrénované vektory vytvořené právě pomocí *GloVe*.

Vektory jsou uloženy v textových souborech. K dispozici je volba mezi reprezentacemi vektory s dimenzí 50, 100, 200 a 300. V souboru se na každém řádku nachází anglické slovo, interpunkční znaménko, případně jiný token a za ním je mezerou oddělený vektor. Vektor obsahuje 50, 100, 200 nebo 300 reálných čísel (záleží na souboru), které jsou jedno od druhého také odděleny mezerami. Tokeny jsou v souboru seřazeny sestupně podle jejich frekvence výskytu v textu.

4.6 Model

Model v základní verzi chatbotu i modely ve vylepšené verzi chatbotu jsou založené na *sequence-to-sequence* principu (viz. 3.2). Všechny modely je možné natrénovat a následně použít pro predikci odpovědí.

Jako ztrátová funkce je použita funkce *categorical crossentropy*¹¹ (viz. 2.4), která se používá pro klasifikaci do více tříd. Ztráta s hodnotou 0 znamená, že není rozdíl mezi skutečným a očekávaným výstupem neuronové sítě.

Jako optimalizační nástroj je použit optimalizátor (viz. 2.4) *Adam*¹², který je nenáročný na paměť, výpočetně efektivní, vhodný pro trénink na velkém množství dat a obecně se často používá při tvorbě generativních chatbotů.

Learning rate (viz. 2.4) v optimalizátoru je nastaven na 0.00005, aby bylo trénování přesnější. Na druhou stranu je ale trénink o něco pomalejší, než kdyby se použil vyšší learning rate.

Během tréninku se počítá i přesnost na trénovacích datech pomocí metriky *accuracy*, která oproti ztrátě určuje, jak moc je skutečný výstup neuronové sítě podobný očekávanému výstupu, kde hodnota 1 znamená, že jsou si rovny.

Základní složení modelu

Základem každého modelu (Obrázky 4.1, 4.2, 4.3 a 4.4) je vstupní vrstva pro *encoder* a vstupní vrstva pro *decoder*, které berou jako vstup vektory s dimenzí o velikosti 30, což je 30 indexů, které představují pozice konkrétních tokenů ve slovníku. Na vstup vrstvy pro *encoder* jde kontext a na vstup vrstvy pro *decoder* jde odpověď.

Výstupy obou vrstev jdou na vstup vrstvě *Embedding*, která nahradí každý index ve vstupním vektoru vektorem s dimenzí o velikosti word embedding, která je pro základní (viz. 4.10) i vylepšenou verzi (viz. 4.11) rozdílná. Tento vektor tvoří embedding vždy pro konkrétní token, který se nachází na daném indexu ve slovníku (viz. 4.3).

Dále se modely již v něčem liší (viz. 4.10 a 4.11).

¹⁰Global Vectors For Word Representation

¹¹<https://keras.io/losses/>

¹²<https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/>

Aktivační funkce

V modelu jsou používány v některých vrstvách aktivační funkce (viz. 2.4) *RELU*¹³ a *softmax*¹⁴ k transformaci vstupů na výstup. Ve skrytých vrstvách je používána aktivační funkce *RELU*, která převádí záporná vstupní reálná čísla na číslo 0 a kladná čísla nechává stejná. Dále je zde, například ve výstupní vrstvě použita aktivační funkce *softmax*, která transformuje vstupní reálná čísla na reálná čísla v rozsahu 0 až 1. Tato aktivační funkce se často používá při strojových překladech při klasifikaci do několika tříd.

4.7 Trénink

Před tréninkem modelu je nejdříve potřeba načíst a zpracovat trénovací data, a poté je rozdělit na trénovací a validační.

Zpracování trénovacích dat

Oba soubory s trénovacími daty (viz. 4.4) jsou postupně zpracovány. Z každého souboru jsou načítány jednotlivé textové sekvence. Zároveň je načten vždy kontext z prvního souboru a příslušná odpověď z druhého souboru.

Jednotlivé načtené sekvence jsou rozděleny na tokeny pomocí funkce `word_tokenize` z knihovny *nlk* jazyka *Python*.

Po převedení sekvencí na tokeny, je kontrolováno, že kontext ani odpověď nejsou prázdné. Dále musí být počet tokenů kontextu i odpovědi menší nebo roven maximálnímu povolenému počtu tokenů. Toto je nutné kvůli pozdější potřebě zarovnání všech sekvencí na stejnou délku a také aby nemuseli být moc dlouhé sekvence kontextu nebo odpovědi ořezány a přebývající tokeny zahozeny. Tím by mohlo dojít ke ztrátě celkového významu nebo ke změně smyslu obou sekvencí.

Následně se projdou obě sekvence tokenů a kontroluje se, kolik tokenů v sekvencích se nenachází ve slovníku. Pokud se v alespoň jedné sekvenci nachází více neznámých tokenů než je povolený limit, který je nastaven na maximálně jeden neznámý token v sekvenci, tak jsou obě sekvence zahozeny a pokračuje se v načítání dalších dat. Pokud obě sekvence tokenů tento limit splňují (neobsahují žádný nebo maximálně jeden neznámý token), tak je případný neznámý token nahrazen tokenem k tomu určeným, kterým je slovo *something*. Tato kontrola je prováděna, aby došlo k co nejlepšímu zachování celkového významu sekvencí. Pokud by sekvence obsahovala velké množství neznámých tokenů, tak by došlo ke ztrátě nebo změně jejího významu, protože neznámé tokeny chatbot neumí zpracovat.

Po provedení výše zmíněných kontrol, je na začátek sekvence tokenů odpovědi přidán token *BOS*, představující začátek odpovědi, a na konec této sekvence je přidán token *EOS*, představující konec odpovědi. Kontrola na maximální možnou délku odpovědi zahrnuje tedy i tyto dva tokeny.

Pokud je v příslušném argumentu programu zadán maximální počet trénovacích dat, tak je při překročení tohoto limitu ukončeno načítání. Pokud limit zadán není, tak jsou načtena všechna trénovací data.

Po tomto zpracování jsou všechny tokeny nahrazeny jejich indexem ve slovníku. Následně jsou všechny takto vzniklé sekvence zarovnány na délku maximální možné sekvence.

¹³Rectified Linear Units

¹⁴<https://github.com/Kulbear/deep-learning-nano-foundation/wiki/ReLU-and-Softmax-Activation-Functions>

Jako výplň (padding) je použito číslo 0, které odpovídá tokenu . (tečka) ve slovníku. Výplň je přidávána ke kontextu od začátku a k odpovědi od konce.

Rozdělení dat na trénovací a validační

Vstupní a odpovídající výstupní data jsou náhodně rozdělena na trénovací data, která budou v každé epoše použita k tréninku modelu a na data validační, která jsou používána k validování modelu po každé epoše tréninku. Na těchto datech nebyl model trénován, tudíž musí generalizovat. Validační data tvoří 5% z celkového množství trénovacích dat, ale maximální počet vzorků může být omezen kvůli urychlení tréninku. Omezení je nastaveno na 1000 vzorků dat. Validační data jsou vybrána z trénovacích dat ještě před tréninkem a model na nich tedy není trénován.

Rozdělení dat je provedeno pomocí funkce `train_test_split` z knihovny `sklearn` (konkrétně z balíčku `sklearn.model_selection`), která jako parametry přijímá vstupní a výstupní data a desetinné číslo, které procentuálně určuje výslednou část validačních dat pro vstupní i výstupní data nebo celé číslo, které určuje přesný počet validačních vzorků.

Poté je stejným způsobem ještě vybráno stejné množství trénovacích dat, které budou také v každé epoše použity k měření kvality modelu. Tato data jsou ale použita i k tréninku a jedná se tedy o schopnost modelu tvořit odpověď na kontexty z trénovacích dat. Data jsou také vybrána ještě před tréninkem a po každé epoše jsou tedy použita ta samá.

Trénink modelu

Pokud je trénovacích dat větší množství, konkrétně více než 7000 vzorků, tak dojde, kvůli snížení paměťové náročnosti, k jejich rozdělení na dvě podmnožiny, které se pak použijí k tréninku v každé epoše zvlášť. Podmnožin je vždy o jednu více pro každých dalších 7000 vzorků trénovacích dat.

Při trénování je v jednotlivých epochách modelu dán na vstup neuronové sítě kontext, který je reprezentován sekvencí indexů, určující token, podle slovníku. Jako druhý vstup je stejným způsobem předána částečná odpověď. Jako očekávaný výstup je předána odpověď, která je stejná jako odpověď na vstupu a navíc je zde jeden nový index, představující token, který by měl být model schopný vygenerovat jako následující, po získání částečné odpovědi a kontextu na vstupu. Takto jsou modelu předány postupně všechny kombinace částečné odpovědi a kontextu na vstup a na výstup odpověď tvořená částečnou odpovědí na vstupu a následujícím indexem v odpovědi.

Trénink probíhá, dokud nedojde k dosažení předdefinovaného počtu epoch nebo dokud není přerušen uživatelem.

Po každé epoše tréninku dochází k testování kvality natrénovaného modelu na trénovacích a validačních datech pomocí metriky *BLEU* (viz. 3.2).

Testování kvality modelu pomocí metriky během tréninku

Během tréninku je možné orientačně sledovat kvalitu natrénovaného modelu na zobrazeném skóre, získaném pomocí metriky *BLEU*, které se počítá pro validační i trénovací data.

K dispozici jsou tedy, po každé epoše tréninku, dvě *BLEU* skóre v rozmezí 0 až 1. Skóre rovno číslu 0 znamená kompletně špatnou odpověď a skóre o velikosti 1 totožnou

odpověď s referenční odpovědí z validačních dat nebo trénovacích dat, která byla vybrána pro testování.

4.8 Testování

Aby mohl být model chatbotu (viz. 4.6) testován za pomoci metriky *BLEU* (viz. 3.2) na testovacích datech, musí být nejdříve model natrénován (viz. 4.7).

Pro testování modelu, bylo ze všech dat, které byly k dispozici, odebráno určité množství ještě před tréninkem. Tato data nebyla použita během tréninku ani validování modelu, ale lze je použít během testování vybraného modelu po tréninku. Data jsou uložena stejným způsobem, jako trénovací data (viz. 4.4).

Po spuštění chatbotu v tomto módu, dojde k načtení dat z příslušných souborů. Pokud není zadán limit, tak jsou načtena všechna data.

Poté probíhá výpočet *BLEU* skóre na jednotlivých vzorcích, je vypočten jejich průměr a výsledek je uložen do příslušného souboru.

Výsledkem je skóre v rozmezí 0 až 1. Skóre rovno číslu 0 znamená, že model nebyl schopen vygenerovat alespoň částečně relevantní odpověď ani pro jeden kontext z testovacích dat. Naopak skóre o velikosti 1 znamená, že chatbot vygeneroval pro všechny kontexty z testovacích dat totožné odpovědi, jako jsou příslušné odpovědi v testovacích datech.

Díky tomuto výslednému skóre můžeme získat částečnou představu, jak kvalitně pracuje zvolený model na reálných datech. Pro dokonalejší ověření kvality modelu, je nutné nechat model otestovat uživateli.

4.9 Chatování

Pro umožnění uživateli konverzovat s chatbotem, je nejdříve potřeba natrénovat (viz. 4.7) model (viz. 4.6) a uložit takto upravené váhy do souboru, ze kterého jsou pak načteny a použity v modelu pro predikci odpovědi.

Chatování s chatbotem je umožněno buď v konzole, kde prázdný vstup znamená ukončení programu nebo ve vytvořeném GUI.

GUI obsahuje textové pole pro zadání uživatelského vstupu, který je poslán ke zpracování při stisknutí tlačítka *enter* na klávesnici nebo příslušného tlačítka vedle textového vstupu. Nad tímto textovým polem se nachází výpis celé konverzace, kterou uživatel vedl s chatbotem, kde poslední odpověď chatbotu se nachází na konci textového pole.

Model

Nejdříve jsou z odpovídajícího souboru načteny uložené váhy z tréninku. Pokud nejsou váhy uloženy pro zvolený model, dojde k ukončení programu s chybou. Následně je ze souboru načten slovník (viz. 4.3), který byl použit při tréninku. Vektorová reprezentace není již potřeba, jelikož váhy už jsou k dispozici po načtení ze souboru a není potřeba vytvářet nové. Poté je vytvořen model jako u tréninku, ale rovnou jsou do něj načteny váhy získané ze souboru.

Generování odpovědi

Získaná vstupní textová sekvence od uživatele je rozdělena na jednotlivé tokeny. Pokud je překročen limit na maximální délku textové sekvence, kterým je 30 tokenů, tak je textový vstup od konce ořezán na maximální délku textové sekvence.

Následně jsou všechny tokeny nahrazeny jejich indexem ve slovníku. Když se některý token ve slovníku (viz. 4.3) nenachází, tak je substituován k tomu určených tokenem, kterým je slovo *something*.

Poté je takto získaná sekvence indexů, představující tokeny kontextu, použita k vytvoření odpovědi. Na začátku je odpověď tvořena pouze indexem tokenu *BOS*, který je uložen na poslední index v sekvenci. Zbytek sekvence, představující odpověď, je vyplněn nulami, které tvoří padding.

Vstup i současný stav částečné odpovědi jsou předány modelu do funkce `predict` jako parametry a funkce vrátí vygenerovaný index následujícího tokenu. Tento index se přidá před index tokenu *BOS*. Poté jsou opět předány funkci `predict` vstupní sekvence a nově upravená sekvence, představující odpověď. Takto se pokračuje, dokud nejsou vygenerovány všechny indexy do požadované délky sekvence (maximální možná délka sekvence).

Následně jsou všechny indexy nahrazeny odpovídajícím tokenem ve slovníku, převedeny do textové podoby a je odstraněn token *BOS* a první výskyt tokenu *EOS*, společně se vším, co následuje po něm v dané sekvenci.

Všechny tokeny, kromě specifických tokenů, jsou odděleny mezerou. Tokeny . (tečka), .. (dvě tečky), ... (tři tečky), , (čárka), ! (vykřičník), ? (otazník), ; (mezerník), : (dvojtečka), ' (apostrof), " (uvozovky),) (pravá závorka), / (pravá hranatá závorka), } (pravá složená závorka) nepředchází mezera a po tokenech ' (apostrof), " (uvozovky), ((levá závorka), [(levá hranatá závorka), { (levá složená závorka) nenásleduje mezera.

Dále dojde k nahrazení počátečních písmen vět za velká písmena. Také jsou velkými písmeny nahrazena první písmena u křestních jmen, které se nachází v seznamu křestních jmen, který obsahuje 1000 anglických křestních jmen. Seznam obsahuje pouze 1000 jmen, protože originální seznam jmen¹⁵ obsahoval i jména, která mají i jiný význam než jen křestní jméno.

Pokud není takto vzniklá textová sekvence prázdná, tak je vytištěna na standardní výstup, jinak chatbot odpoví, že nerozumí. Následně se opět čeká na další uživatelský vstup.

4.10 Základní verze chatbotu

Základní verze chatbotu byla implementována na základě existujícího generativního chatbotu¹⁶.

Slovník

Základní verze chatbotu používala slovník (viz. 4.3) o velikosti 6999 tokenů, ke kterému byl přidán token, sloužící pro nahrazení neznámých tokenů. Výsledná velikost slovníku tedy byla 7000 tokenů.

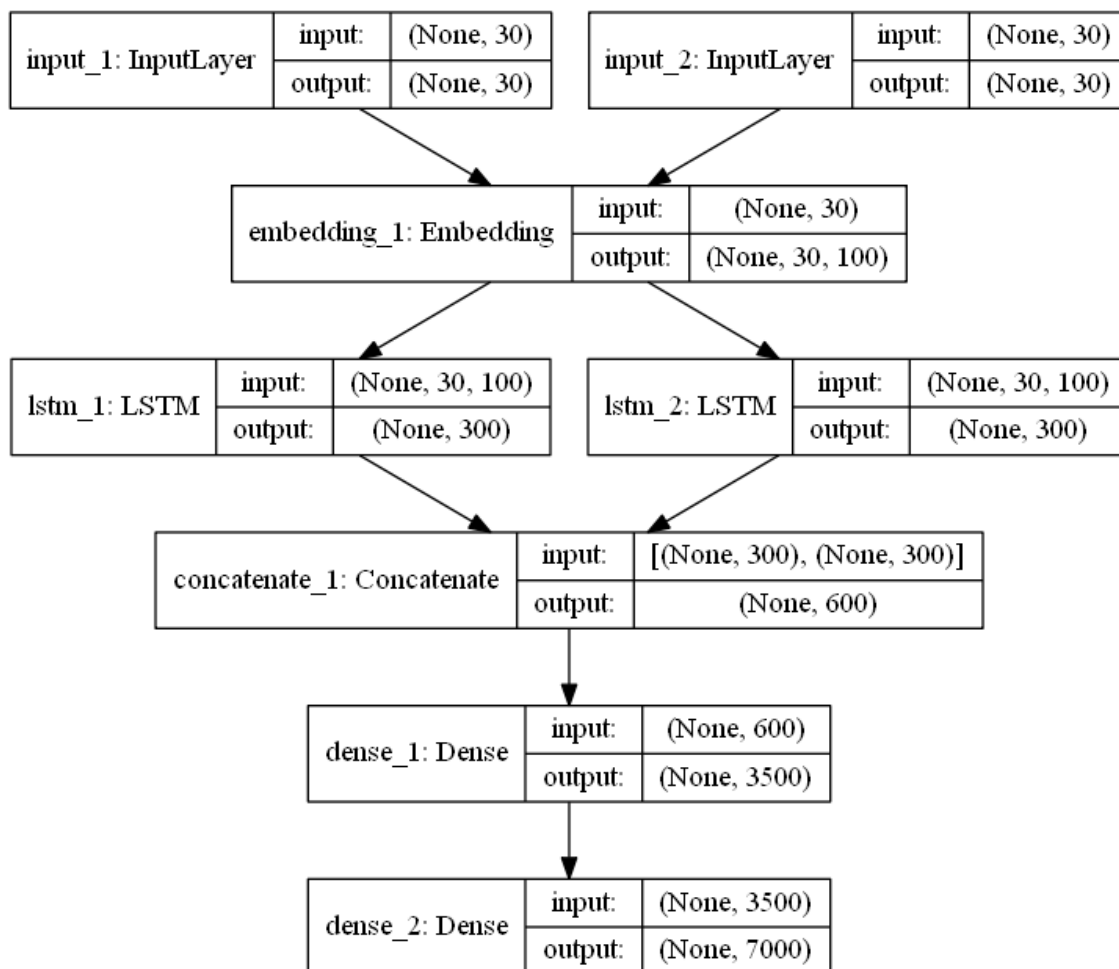
¹⁵<http://deron.meranda.us/data/census-derived-all-first.txt>

¹⁶<https://github.com/oswaldoludwig/Seq2seq-Chatbot-for-Keras>

Model

Model (Obrázek 4.1) pracoval s word embedding s dimenzí o velikosti 100 a sentence embedding s dimenzí o velikosti 300.

Sekvence vektorů, které jsou na výstupu vrstvy *Embedding* (viz. 4.6), jdou každá na vstup *LSTM* vrstvě (viz. 2.4), která vrací vektor s dimenzí o velikosti sentence embedding, který je zde 300. Tyto vektory jsou následně spojeny do jednoho vektoru o velikosti 600. Následuje *Dense* vrstva (viz. 2.4) s aktivační funkcí *RELU* (viz. 4.6), která transformuje tento vektor na vektor s dimenzí o velikosti poloviny slovníku, a poté je výstupní vrstva neuronové sítě s aktivační funkcí *softmax* (viz. 4.6), která transformuje výstupní vektor předchozí vrstvy na vektor s dimenzí o velikosti celého slovníku.



Obrázek 4.1: Složení modelu (jednotlivé vrstvy neuronové sítě a jejich vzájemné propojení) v základní verzi chatbotu (vygenerováno pomocí funkce `plot_model` z balíčku `keras.utils`)

4.11 Vylepšená verze chatbotu

Při dalších úpravách chatbotu bylo použito více trénovacích dat (viz. 4.4), a to konkrétně místo původních 10000 bylo použito 172362 vzorků trénovacích dat. Také byla použita vyšší

dimenze word embedding (viz. 4.5), která byla 100 v základní verzi chatbotu a je 300 ve vylepšené verzi chatbotu. Dále byl rozšířen slovník (viz. 4.3) z původních 6999 tokenů na 7815 tokenů, a také byly přidány další dva modely, které je možné také natrénovat.

Slovník

Ve vylepšené verzi chatbotu byl původní slovník (viz. 4.10), který obsahoval 6999 tokenů, rozšířen o tokeny, které se vyskytovaly alespoň 50krát v trénovacích datech o maximální délce 30 tokenů na výslednou velikost 7815 tokenů, ke kterým se přidá token, nahrazující neznámé tokeny.

Modely

Chatbot obsahuje celkem tři modely (Obrázky 4.2, 4.3 a 4.4), které pracují s word embedding s dimenzí o velikosti 300 a sentence embedding s dimenzí o velikosti 300.

První model

První model (Obrázek 4.2) je skoro stejný, jako model v základní verzi chatbotu (viz. 4.10). Liší se pouze ve velikosti word embedding, který zde tvoří vektory s dimenzí o velikosti 300 a použití rozšířeného slovníku.

Druhý model

Druhý model (Obrázek 4.3) se od prvního modelu (Obrázek 4.2) liší počtem *Dense* vrstev (viz. 2.4). Tyto vrstvy transformují vektor, získaný spojením výstupních vektorů z *LSTM* vrstev (viz. 2.4), s dimenzí o velikosti 600 na vektor s dimenzí o velikosti rovné velikosti slovníku, která zde je 7816 tokenů. Nachází se zde na rozdíl od prvního modelu celkem čtyři *Dense* vrstvy. První tři vrstvy používají k transformaci aktivační funkci *RELU* (viz. 4.6) a poslední *softmax* (viz. 4.6). První má na výstupu vektor s dimenzí o velikosti 1/4 velikosti slovníku, druhá vrstva 1/2 velikosti slovníku, třetí vrstva 3/4 velikosti slovníku a čtvrtá vrstva, která tvoří zároveň výstupní vrstvu modelu, má na výstupu vektor s dimenzí o velikosti celého slovníku.

Třetí model

Třetí model (Obrázek 4.4) používá několik vrstev zajišťujících attention mechanismus¹⁷ [6] u *encoderu*, který byl implementován podle existujícího návrhu¹⁸.

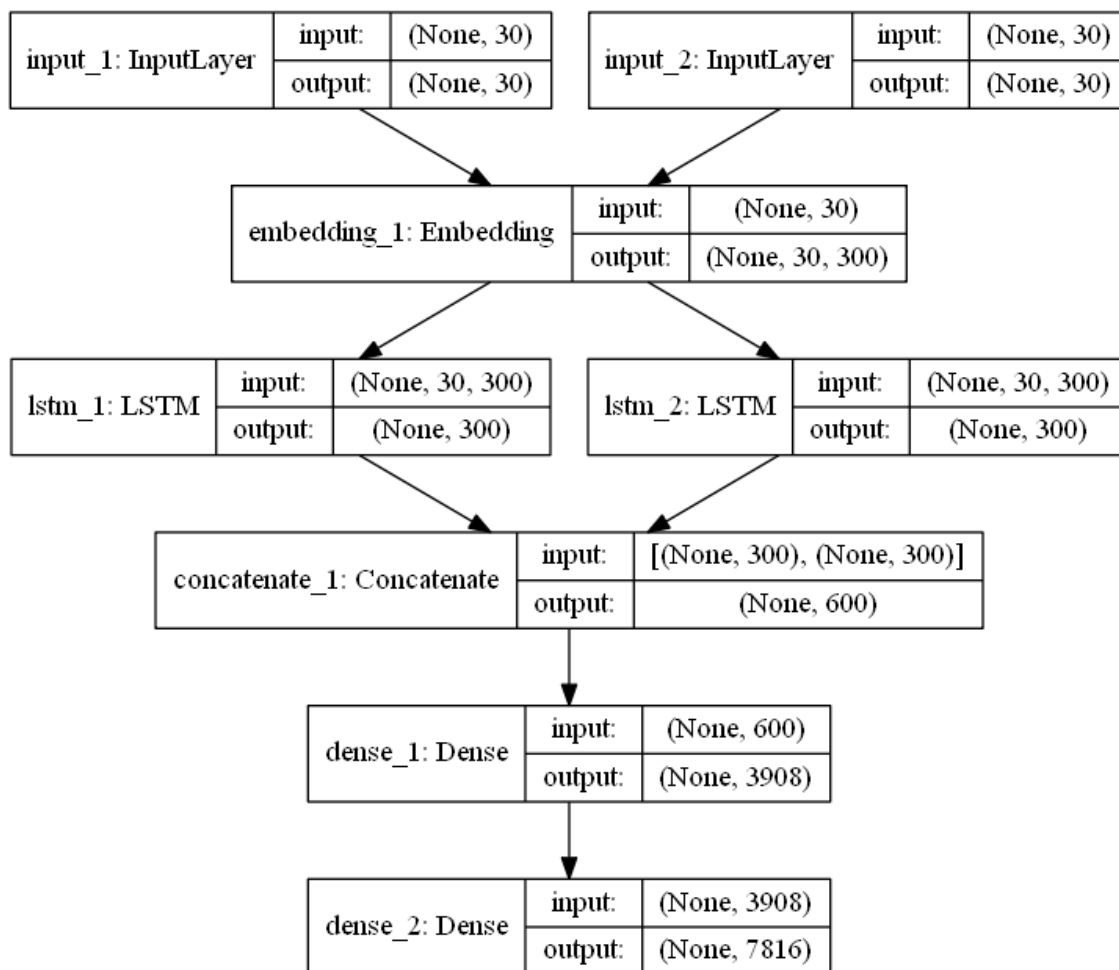
Attention mechanismus by měl částečně odstranit klasický problém *encoderu*, který musí převést celý vstupní kontext do jednoho vektoru určité velikosti. Tím nemusí být zachycen celkový význam a sémantika vstupního kontextu. U attention mechanismu by mělo dojít k vytvoření výstupního vektoru *encoderu*, ve kterém bude obsažen každý krok transformace, tedy význam každého tokenu v původním vstupním kontextu.

Attention mechanismus je ve třetím modelu proveden tak, že výstup z *Embedding* vrstvy (viz. 4.6), jde na vstup *Dense* vrstvě (viz. 2.4) používající aktivační funkci (viz. 2.4) *tanh*¹⁹. Tato aktivační funkce transformuje reálná čísla na čísla v rozsahu od -1 až 1. Záporná čísla jsou tedy transformována na záporná a kladná na kladná v tomto rozsahu. Výstup z této

¹⁷<https://medium.com/@Synced/a-brief-overview-of-attention-mechanism-13c578ba9129>

¹⁸<https://github.com/keras-team/keras/issues/4962>

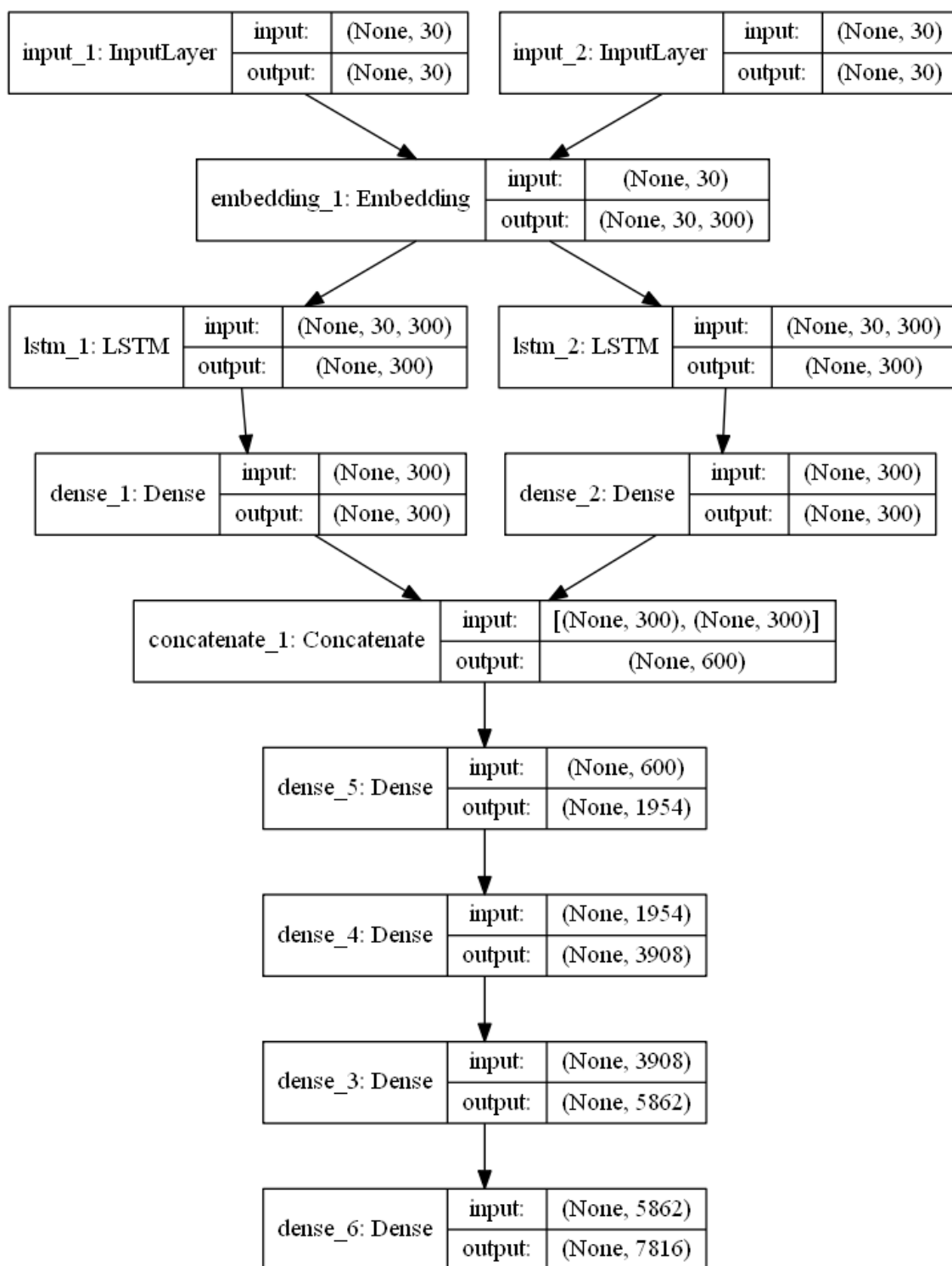
¹⁹<https://theclevermachine.wordpress.com/tag/tanh-function/>



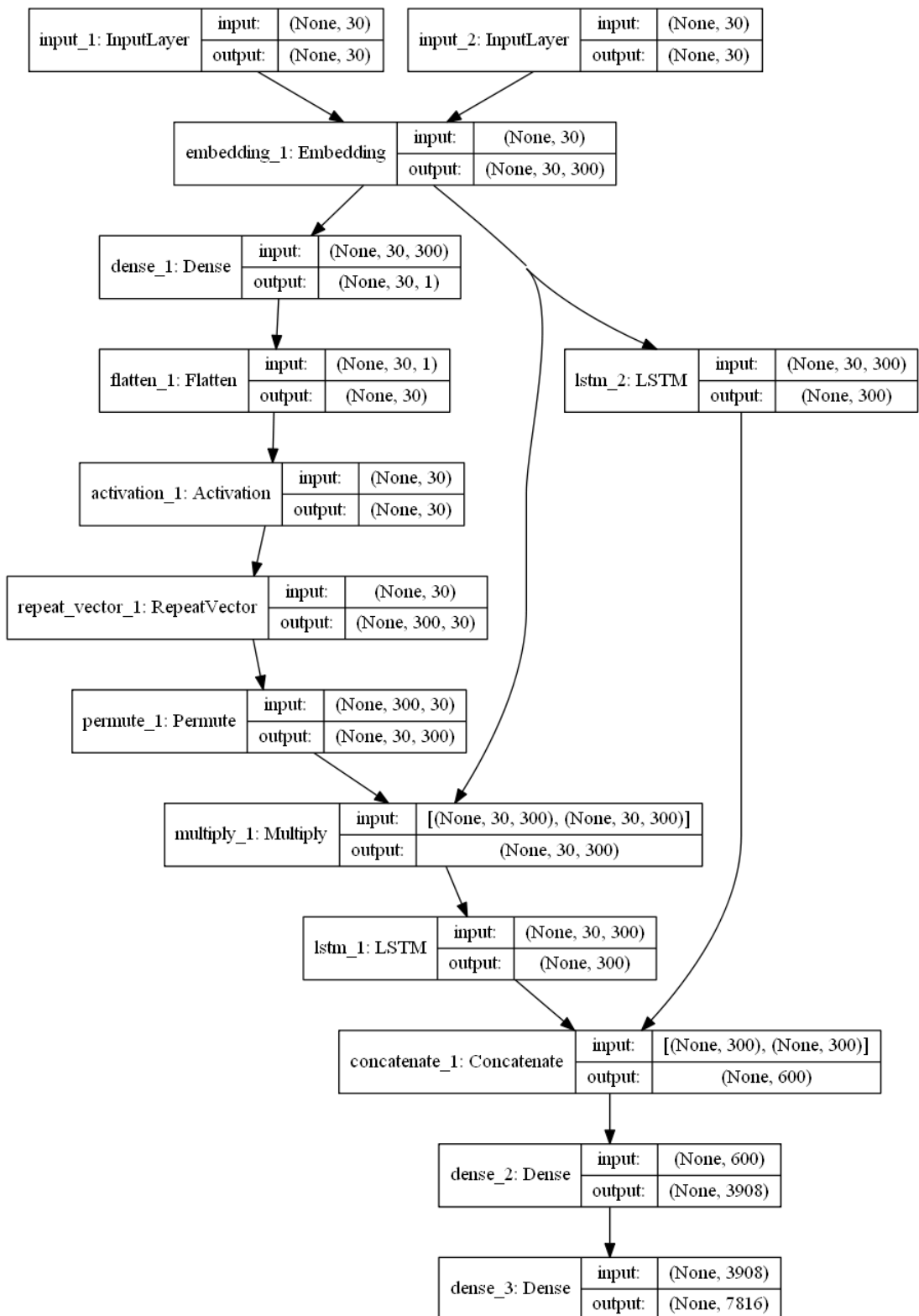
Obrázek 4.2: Složení prvního modelu (jednotlivé vrstvy neuronové sítě a jejich vzájemné propojení) ve vylepšené verzi chatbotu (vygenerováno pomocí funkce `plot_model` z balíčku `keras.utils`)

vrstvy je srovnán na vektor s dimenzí o velikosti maximálního počtu tokenů v sekvenci, který je nastaven na 30. Tento vektor je transformován pomocí aktivační funkce *softmax* (viz. 4.6), jejíž výstup je transformován zpět na vektor, obsahující 30 vektorů s dimenzí o velikosti word embedding, který je 300. Tento výsledek je vynásoben vektorem, který byl na výstupu z *Embedding* vrstvy a tedy byl i poté použit na vstupu vrstev tvořících attention mechanismus.

Dále už je model stejný jako první model (Obrázek 4.2).



Obrázek 4.3: Složení druhého modelu (jednotlivé vrstvy neuronové sítě a jejich vzájemné propojení) ve vylepšené verzi chatbotu (vygenerováno pomocí funkce `plot_model` z balíčku `keras.utils`)



Obrázek 4.4: Složení třetího modelu (jednotlivé vrstvy neuronové sítě a jejich vzájemné propojení) ve vylepšené verzi chatbotu, který využívá *attention mechanismus* (vygenerováno pomocí funkce `plot_model` z balíčku `keras.utils`)

Kapitola 5

Testování a výsledky

Základní i vylepšená verze chatbotu, obsahující tři modely, byly testovány pomocí metriky *BLEU* (viz. 3.2). Vylepšená verze chatbotu byla testována i uživateli.

V rámci testování vylepšené verze chatbotu uživateli, byl chatbot napojen na API existujícího chatbotu *Cleverbot*¹ (viz. 5.4). Uživatelé vedli konverzaci s chatbotem, který používal postupně všechny natrénované modely, a také byl napojen na chatbot *Cleverbot*. Celkem tedy uživatelé hodnotili čtyři konverzace s chatbotem. Uživatelé nevěděli, který chatbot je který, protože vždy používali stejné grafické uživatelské rozhraní pro chatování. Následně uživatelé ohodnotili kvalitu odpovědí chatbotu používajícího jednotlivé modely i chatbotu *Cleverbot*.

5.1 Základní verze chatbotu

Pro natrénování modelu bylo použito 10000 vzorků dat, ze kterých se odebralo 500 pro validování pomocí metriky *BLEU*. Z ostatních vzorků bylo náhodně vybráno 500 vzorků, na kterých byla potom testována schopnost modelu vytvářet odpovědi na data použitá při tréninku. Trénovacích dat bylo tedy 9500 vzorků a celý model se trénoval 65 epoch (Obrázky 5.1 a 5.2).

Pro vybraný model, po 65 epochách tréninku, byla ztráta, získaná pomocí ztrátové funkce (viz. 4.6), rovna 0,581952 a přesnost na trénovacích datech 0,79359 (Obrázek 5.1). Při testování na trénovacích datech, pomocí metriky *BLEU*, získal model skóre 0,335994 a na validačních datech skóre 0,103834 (Obrázek 5.2).

Testování pomocí metriky

Po zvolení výsledného modelu, byl tento model testován (viz. 4.8) pomocí metriky *BLEU* na 1000 vzorcích testovacích dat, které nebyly použity k tréninku ani validování modelu během tréninku. Model získal, při testování na testovacích datech, skóre 0,086986.

Příklad krátké konverzace se základní verzí chatbotu

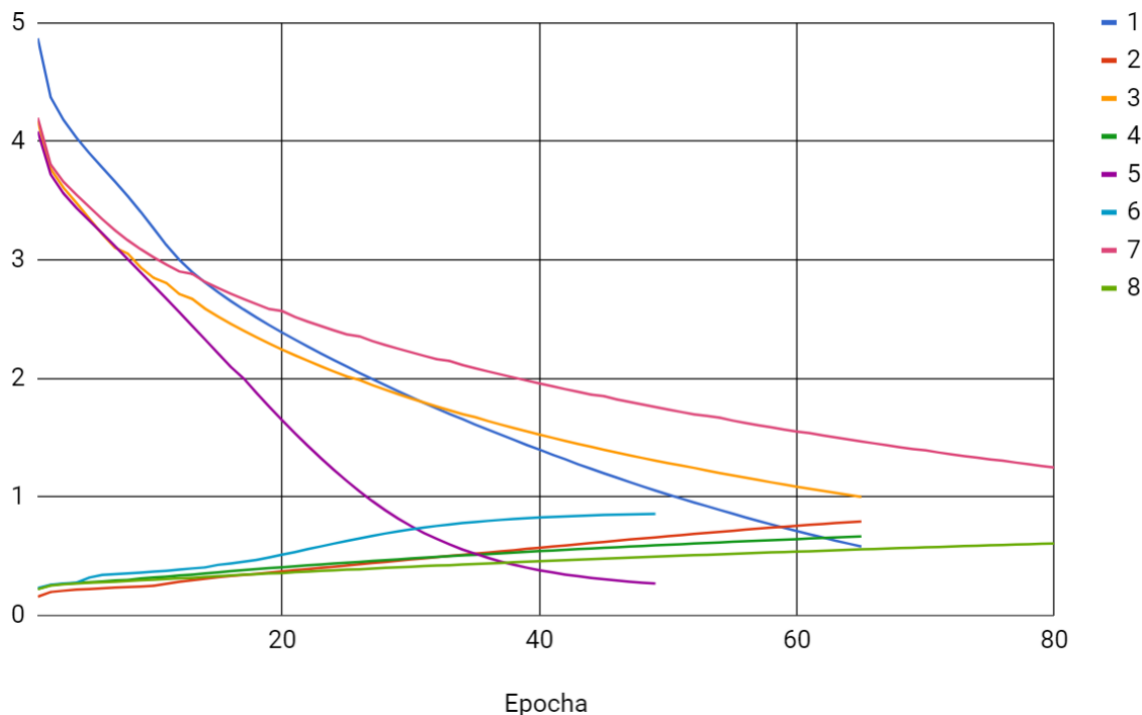
Kontext: Hello

Odpověď: Bye.

Kontext: How are you?

Odpověď: I got to make you to.

¹<https://www.cleverbot.com/api/>



Obrázek 5.1: Graf vypočtené ztráty a přesnosti na trénovacích datech základní verze chatbotu a jednotlivých modelů ve vylepšené verzi chatbotu v každé epoše tréninku.

- | | |
|--|--|
| 1: Ztráta získaná během tréninku základní verze chatbotu | 5: Ztráta získaná během tréninku druhého modelu vylepšené verze chatbotu |
| 2: Přesnost získaná během tréninku základní verze chatbotu | 6: Přesnost získaná během tréninku druhého modelu vylepšené verze chatbotu |
| 3: Ztráta získaná během tréninku prvního modelu vylepšené verze chatbotu | 7: Ztráta získaná během tréninku třetího modelu vylepšené verze chatbotu |
| 4: Přesnost získaná během tréninku prvního modelu vylepšené verze chatbotu | 8: Přesnost získaná během tréninku třetího modelu vylepšené verze chatbotu |

Kontext: What is your name?

Odpověď: Everybody calls me George

Kontext: How old are you?

Odpověď: I do know.

Kontext: Where do you live?

Odpověď: I be here.

Kontext: Where are you from?

Odpověď: I not to talk about you.

Kontext: Are you student?

Odpověď: I do know.

Kontext: Where do you work?

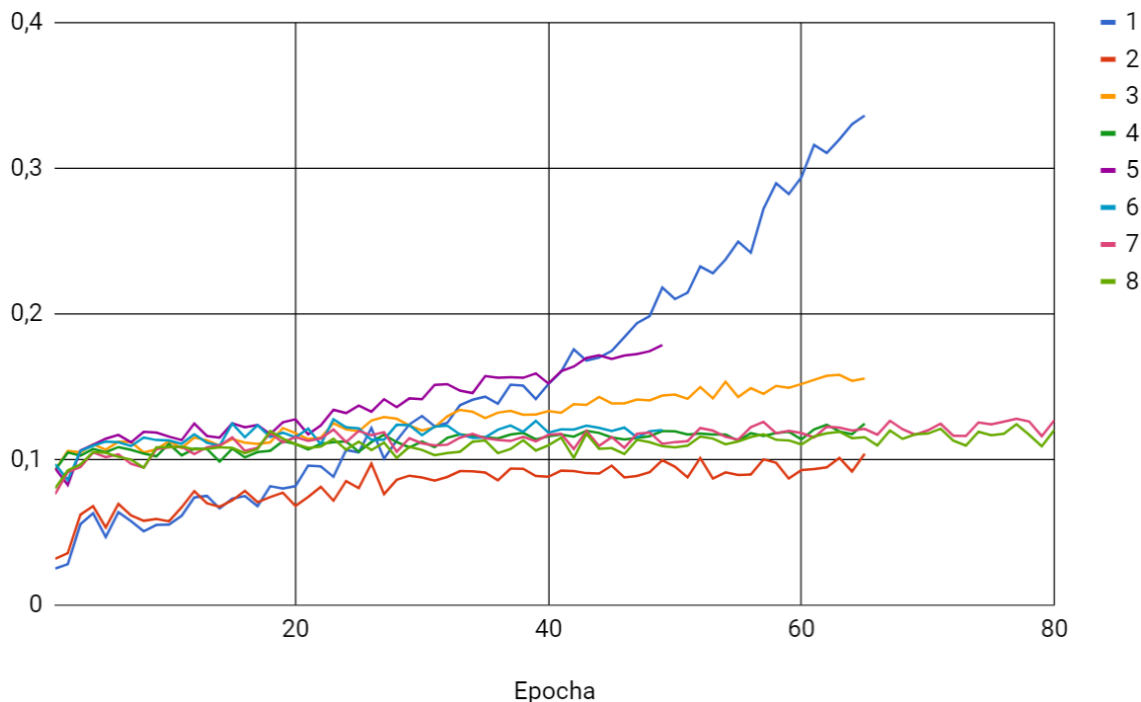
Odpověď: I be here, to feel about it.

Kontext: Do you have any brothers or sisters?

Odpověď: Yes.

Kontext: I like to eat. The best food is pizza. What do you like to eat?

Odpověď: I thought you will. If you want me to see you at it. I not one of the.



Obrázek 5.2: Graf vypočteného skóre pomocí metriky *BLEU*, které získala základní verze chatbotu a jednotlivé modely ve vylepšené verzi chatbotu, v každé epoše tréninku.

- | | |
|---|---|
| 1: <i>BLEU</i> skóre získané na trénovacích datech během tréninku základní verze chatbotu | 5: <i>BLEU</i> skóre získané na trénovacích datech během tréninku druhého modelu vylepšené verze chatbotu |
| 2: <i>BLEU</i> skóre získané na validačních datech během tréninku základní verze chatbotu | 6: <i>BLEU</i> skóre získané na validačních datech během tréninku druhého modelu vylepšené verze chatbotu |
| 3: <i>BLEU</i> skóre získané na trénovacích datech během tréninku prvního modelu vylepšené verze chatbotu | 7: <i>BLEU</i> skóre získané na trénovacích datech během tréninku třetího modelu vylepšené verze chatbotu |
| 4: <i>BLEU</i> skóre získané na validačních datech během tréninku prvního modelu vylepšené verze chatbotu | 8: <i>BLEU</i> skóre získané na validačních datech během tréninku třetího modelu vylepšené verze chatbotu |

Kontext: Do you have any hobbies?

Odpověď: Yes.

Kontext: I was snowboarding in Italy this winter. Do you do any sports?

Odpověď: No, it me!

Kontext: We can also go to cinema. Would you like to watch a movie with me?

Odpověď: I take it up to you.

Kontext: Can we be friends?

Odpověď: Yes, it

Kontext: Bye

Odpověď: Bye bye.

5.2 Testování uživatelů

Při testování jednotlivých modelů ve vylepšené verzi chatbotu a chatbotu *Cleverbot*, dostali testující uživatelé dotazník, který obsahoval vždy 6 stejných otázek pro zjištění kvality odpovědí jednotlivých modelů a kvality odpovědí chatbotu *Cleverbot*. Po každé konverzaci tedy vyplnili uživatelé odpovědi na 6 příslušných otázek.

Otázky pro testující uživatele

1. *Jaká je kvalita odpovědí na věty obsahující 15 a méně slov (nebo interpunkčních znamének)?*

Otázka zjišťuje kvalitu odpovědí na vstupy od uživatele, které obsahují **15 a méně tokenů**. Odpověď s hodnotou 1 znamená, že odpovědi chatbotu neměly vůbec žádný smysl ani nesouvisely se vstupem. Odpověď s hodnotou 10 znamená, že se uživateli odpovědi chatbotu jevily, jako odpovědi od člověka.

2. *Jaká je kvalita odpovědí na věty obsahující více než 15 slov (nebo interpunkčních znamének)?*

Otázka zjišťuje kvalitu odpovědí na vstupy od uživatele, které obsahují **více než 15 tokenů**. Odpověď s hodnotou 1 znamená, že odpovědi chatbotu neměly vůbec žádný smysl ani nesouvisely se vstupem. Odpověď s hodnotou 10 znamená, že se uživateli odpovědi chatbotu jevily, jako odpovědi od člověka.

3. *Bylo více odpovědí obsahujících 15 a méně slov (nebo interpunkčních znamének) a nebo odpovědí obsahujících více než 15 slov (nebo interpunkčních znamének)?*

Otázka zjišťuje, jaký byl poměr mezi odpověďmi obsahujícími **15 a méně tokenů** a odpověďmi obsahujícími **více než 15 tokenů**.

Odpovědi na otázku:

- (a) *Samé odpovědi obsahující 15 a méně slov (nebo interpunkčních znamének)* - Všechny odpovědi v konverzaci obsahovaly **15 a méně tokenů**.
- (b) *Více odpovědí obsahujících 15 a méně slov (nebo interpunkčních znamének)* - Více odpovědí v konverzaci obsahovalo **15 a méně tokenů**.
- (c) *Stejné množství odpovědí obou délek* - V konverzaci se nacházelo přibližně stejné množství odpovědí obsahujících **15 a méně tokenů** a odpovědí obsahujících **více než 15 tokenů**.
- (d) *Více odpovědí obsahujících více než 15 slov (nebo interpunkčních znamének)* - Více odpovědí v konverzaci obsahovalo **více než 15 tokenů**.
- (e) *Samé odpovědi obsahující více než 15 slov (nebo interpunkčních znamének)* - Všechny odpovědi v konverzaci obsahovaly **více než 15 tokenů**.

4. *Jaká je kvalita odpovědí obsahujících 15 a méně slov (nebo interpunkčních znamének), pokud se takové objevily?*

Otázka zjišťuje kvalitu odpovědí, které obsahují **15 a méně tokenů**. Odpověď s hodnotou 1 znamená, že odpovědi chatbotu neměly vůbec žádný smysl ani nesouvisely se vstupem. Odpověď s hodnotou 10 znamená, že se uživatelům odpovědi chatbotu jevily, jako odpovědi od člověka. Odpověď na tuto otázku byla nepovinná, jelikož nebylo zaručeno, že chatbot vytvoří vhodně dlouhou odpověď.

5. *Jaká je kvalita odpovědí obsahujících více než 15 slov (nebo interpunkčních znamének), pokud se takové objevily?*

Otázka zjišťuje kvalitu odpovědí, které obsahují **více než 15 tokenů**. Odpověď s hodnotou 1 znamená, že odpovědi chatbotu neměly vůbec žádný smysl ani nesouvisely se vstupem. Odpověď s hodnotou 10 znamená, že se uživatelům odpovědi chatbotu jevily, jako odpovědi od člověka. Odpověď na tuto otázku byla nepovinná, jelikož nebylo zaručeno, že chatbot vytvoří vhodně dlouhou odpověď.

6. *Kolik chyb se objevilo v odpovědích?*

Otázka zjišťuje, jaké množství gramatických chyb uživatel našel v odpovědích během konverzace.

Odpovědi na otázku:

- (a) *Žádné chyby*
- (b) *Maximálně jedna chyba*
- (c) *Malé množství chyb*
- (d) *Velké množství chyb*
- (e) *Odpovědi byly kvůli chybám nečitelné*

5.3 Vylepšená verze chatbotu

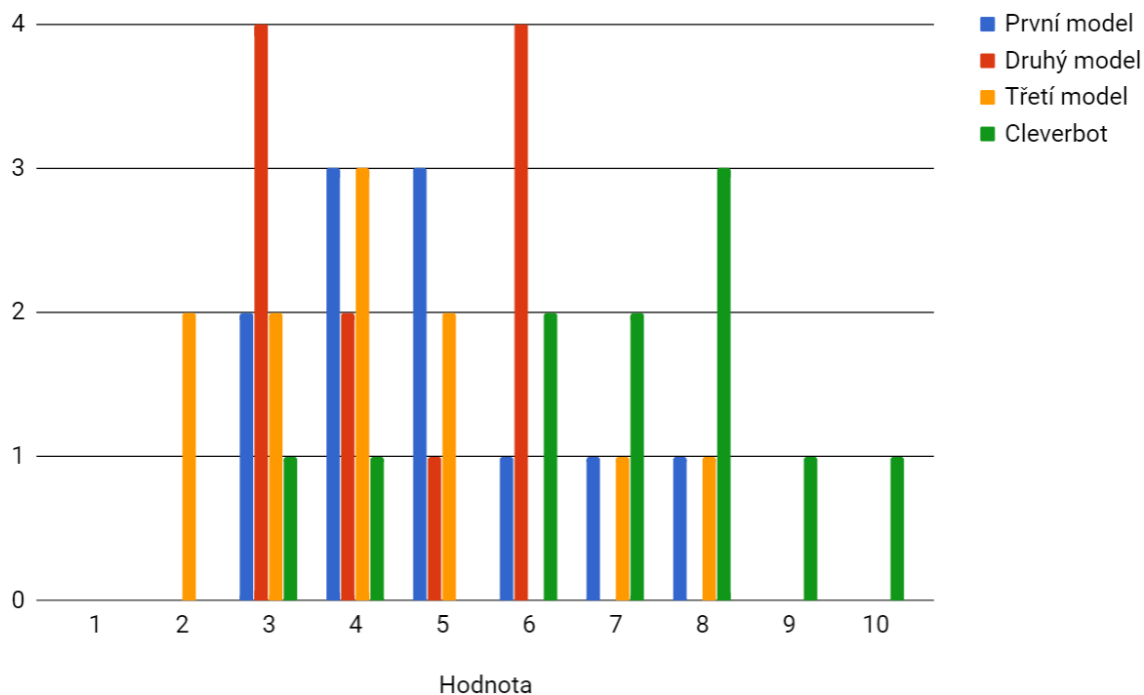
Pro trénink všech modelů bylo použito 172362 vzorků dat. Z těchto dat bylo ještě před začátkem tréninku odebráno přesně 1000 vzorků k validování modelu za použití metriky *BLEU*. Tyto vzorky nebyly následně použity při tréninku. Z ostatních vzorků bylo náhodně vybráno 1000 vzorků, na kterých byla potom testována schopnost modelu vytvářet odpovědi na data použitá při tréninku.

Po zvolení výsledného modelu byly všechny tři modely testovány, za pomoci *BLEU* metriky, na 10000 vzorcích dat, které nebyly použity k tréninku ani validování modelu během tréninku.

První model

Model se trénoval celkem 65 epoch (Obrázky 5.1 a 5.2).

Pro vybraný model, po 65 epochách tréninku, byla ztráta, získaná pomocí ztrátové funkce (viz. 4.6), rovna 1,000275 a přesnost na trénovacích datech 0,665844 (Obrázek 5.1). Při testování na trénovacích datech, pomocí *BLEU* metriky, získal model skóre 0,155484 a na validačních datech skóre 0,124420 (Obrázek 5.2).



Obrázek 5.3: Souhrn odpovědí od testujících uživatelů na otázku „*Jaká je kvalita odpovědi na věty obsahující 15 a méně slov (nebo interpunkčních znamének)?*“. Otázka zjišťuje kvalitu odpovědí na vstupy od uživatele, které obsahují **15 a méně tokenů**. Hodnota 1 znamená, že odpovědi chatbotu neměly vůbec žádný smysl ani nesouvisely se vstupem. Hodnota 10 znamená, že se uživatelé odpovědi chatbotu jeví, jako odpovědi od člověka.

Testování prvního modelu pomocí metriky

Po zvolení výsledného modelu, byl tento model testován (viz. 4.8) pomocí metriky *BLEU* na 10000 vzorcích testovacích dat, které nebyly použity k tréninku ani validování modelu během tréninku. Model získal, při testování na testovacích datech, skóre 0,110598. Model tedy získal vyšší skóre než základní verze chatbotu (viz. 5.1).

Testování prvního modelu uživateli

Z testování uživateli vyplynulo, že na kratší vstupy od uživatelů (Obrázek 5.3), chatbot generuje kvalitnější odpovědi než na delší vstupy od uživatelů (Obrázek 5.4). Chatbot pak většinou generuje kratší odpovědi a někdy také pouze odpovědi, které obsahují 15 a méně tokenů (Obrázek 5.5). Kratší vygenerované odpovědi (Obrázek 5.6) jsou kvalitnější než dlouhé vygenerované odpovědi (Obrázek 5.7). Během krátké konverzace s chatbotem se v odpovědích objevila jedna chyba nebo i malé množství chyb (Obrázek 5.8).

Příklad krátké konverzace s chatbotem, používající první model

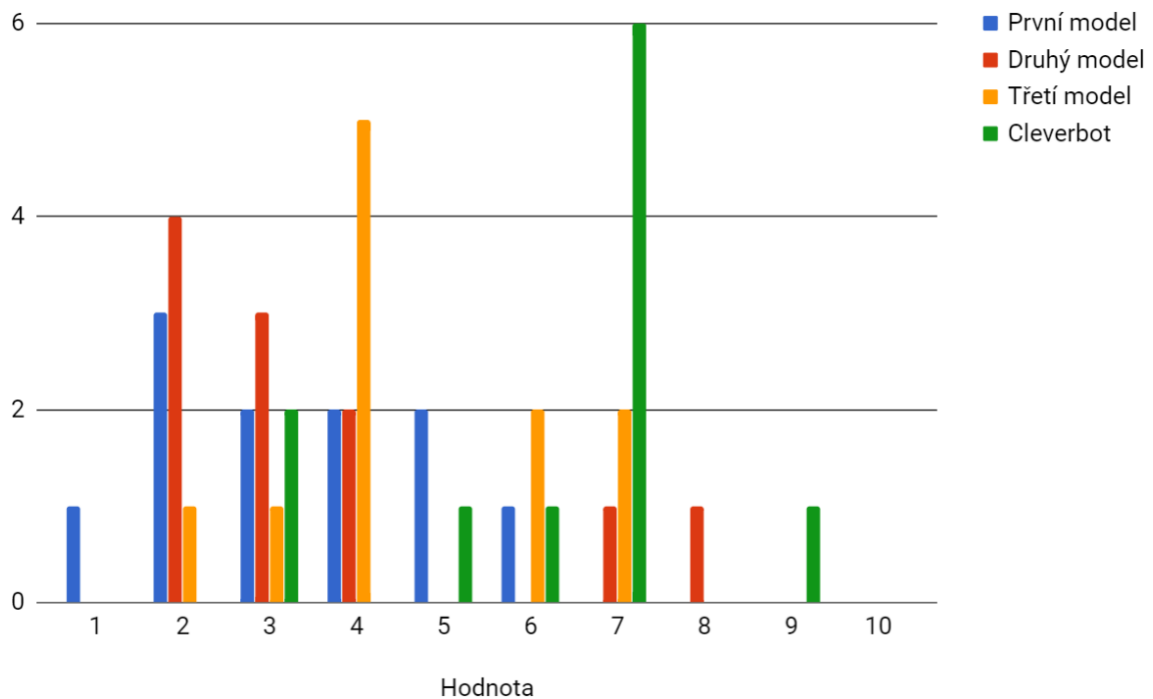
Kontext: Hello

Odpověď: Hello.

Kontext: How are you?

Odpověď: I am fine, thank you.

Kontext: What is your name?



Obrázek 5.4: Souhrn odpovědí od testujících uživatelů na otázku „*Jaká je kvalita odpovědí na věty obsahující více než 15 slov (nebo interpunkčních znamének)?*“. Otázka zjišťuje kvalitu odpovědí na vstupy od uživatele, které obsahují **více než 15 tokenů**. Hodnota 1 znamená, že odpovědi chatbotu neměly vůbec žádný smysl ani nesouvisely se vstupem. Hodnota 10 znamená, že se uživatelé odpovědi chatbotu jevily, jako odpovědi od člověka.

Odpověď: Please, call me George

Kontext: How old are you?

Odpověď: I'm eleven.

Kontext: Where do you live?

Odpověď: I live in a.

Kontext: Where are you from?

Odpověď: I am from czech republic.

Kontext: Are you student?

Odpověď: No.

Kontext: Where do you work?

Odpověď: I'm in the.

Kontext: Do you have any brothers or sisters?

Odpověď: No, i am only the only one and my sister.

Kontext: I like to eat. The best food is pizza. What do you like to eat?

Odpověď: I like the chicken. I have my mom in so long in. I think i May think the same way for myself.

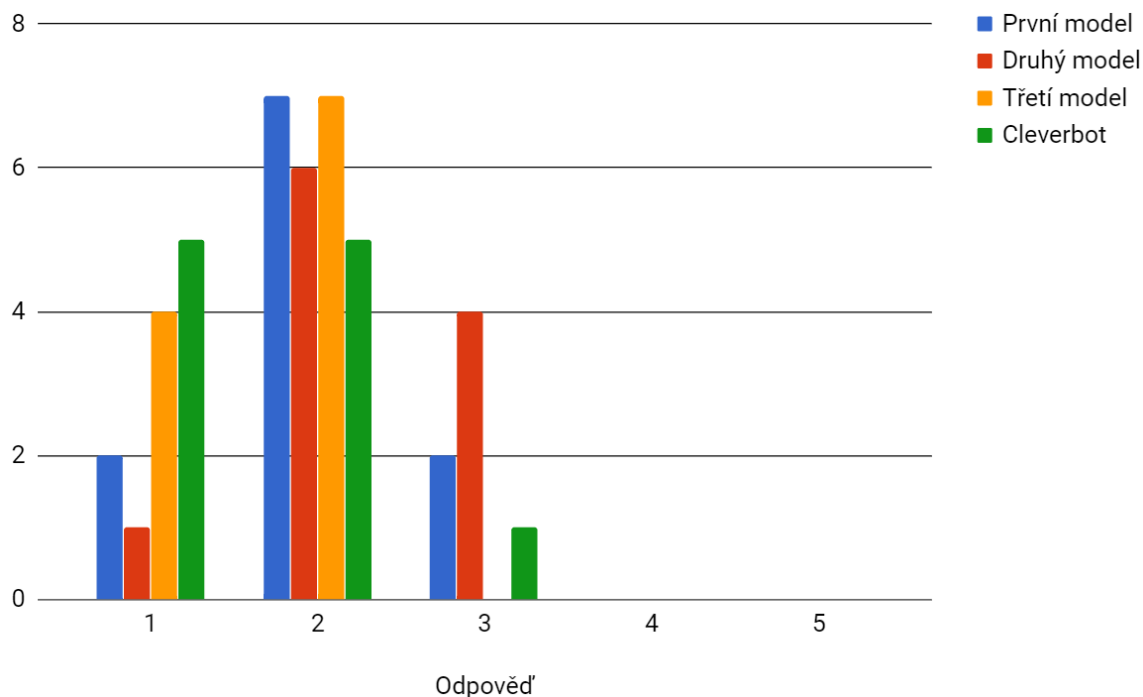
Kontext: Do you have any hobbies?

Odpověď: Yes, i do.

Kontext: I was snowboarding in Italy this winter. Do you do any sports?

Odpověď: Working games and, so no one

Kontext: We can also go to cinema. Would you like to watch a movie with me?



Obrázek 5.5: Souhrn odpovědí od testujících uživatelů na otázku „Bylo více odpovědí obsahujících **15 a méně slov** (nebo interpunkčních znamének) a nebo odpovědí obsahujících **více než 15 slov** (nebo interpunkčních znamének)?“. Otázka zjišťuje, jaký byl poměr mezi odpověďmi obsahujícími **15 a méně tokenů** a odpověďmi obsahujícími **více než 15 tokenů**.

Odpověď 1: *Samé odpovědi obsahující 15 a méně slov (nebo interpunkčních znamének)*
- Všechny odpovědi v konverzaci obsahovaly **15 a méně tokenů**.

Odpověď 2: *Více odpovědí obsahujících 15 a méně slov (nebo interpunkčních znamének)*
- Více odpovědí v konverzaci obsahovalo **15 a méně tokenů**.

Odpověď 3: *Stejné množství odpovědí obou délek* - V konverzaci se nacházelo přibližně stejné množství odpovědí obsahujících **15 a méně tokenů** a odpovědí obsahujících **více než 15 tokenů**.

Odpověď 4: *Více odpovědí obsahujících více než 15 slov (nebo interpunkčních znamének)* - Více odpovědí v konverzaci obsahovalo **více než 15 tokenů**.

Odpověď 5: *Samé odpovědi obsahující více než 15 slov (nebo interpunkčních znamének)*
- Všechny odpovědi v konverzaci obsahovaly **více než 15 tokenů**.

Odpověď: I can work on the channel, i can work on the channel, although i'd be quite well.

Kontext: Can we be friends?

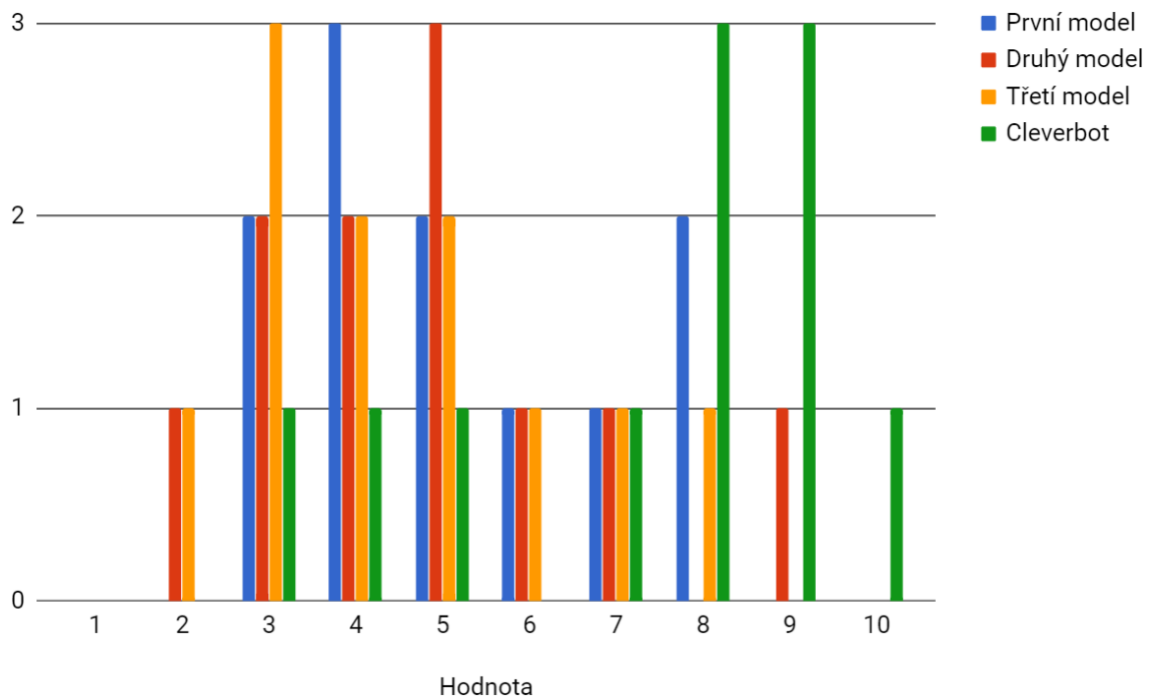
Odpověď: Well, i feel so very excited.

Kontext: Bye

Odpověď: Bye.

Druhý model

Model se trénoval celkem 49 epoch (Obrázky 5.1 a 5.2).



Obrázek 5.6: Souhrn odpovědí od testujících uživatelů na otázku „*Jaká je kvalita odpovědí obsahujících 15 a méně slov (nebo interpunkčních znamének), pokud se takové objevily?*“. Otázka zjišťuje kvalitu odpovědí, které obsahují **15 a méně tokenů**. Hodnota 1 znamená, že odpovědi chatbotu neměly vůbec žádný smysl ani nesouvisely se vstupem. Hodnota 10 znamená, že se uživatelé odpovědi chatbotu jeví, jako odpovědi od člověka. Odpověď na tuto otázku byla nepovinná, jelikož nebylo zaručeno, že chatbot vytvoří vhodně dlouhou odpověď.

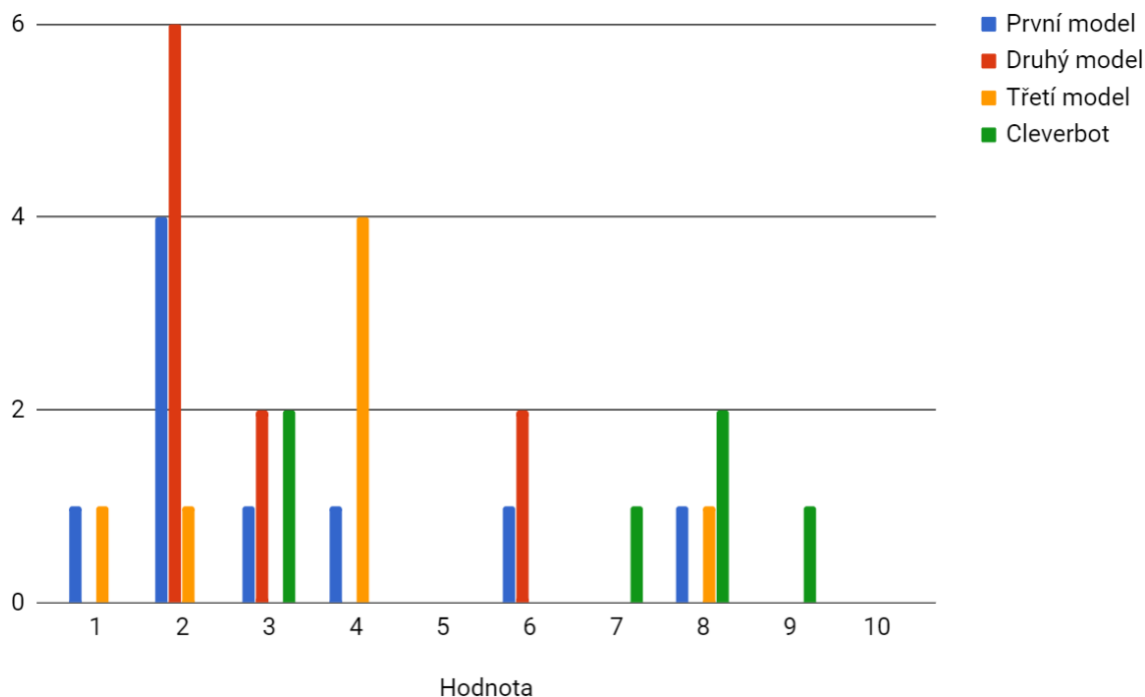
Pro vybraný model, po 49 epochách tréninku, byla ztráta, získaná pomocí ztrátové funkce (viz. 4.6), rovna 0,269027 a přesnost na trénovacích datech 0,857768 (Obrázek 5.1). Při testování na trénovacích datech, pomocí metriky *BLEU*, získal model skóre 0,178382 a na validačních datech skóre 0,120113 (Obrázek 5.2).

Testování druhého modelu pomocí metriky

Po zvolení výsledného modelu, byl tento model testován (viz. 4.8) pomocí metriky *BLEU* na 10000 vzorcích testovacích dat, které nebyly použity k tréninku ani validování modelu během tréninku. Model získal, při testování na testovacích datech, skóre 0,110568. Model tedy získal jen nepatrně menší skóre než první model (viz. 5.3).

Testování druhého modelu uživateli

Z výsledků testování je vidět, že na kratší vstupy od uživatelů (Obrázek 5.3), chatbot generuje kvalitnější odpovědi než na delší vstupy od uživatelů (Obrázek 5.4). Chatbot častěji generuje kratší odpovědi, které obsahují 15 a méně tokenů (Obrázek 5.5). Kratší vygenerované odpovědi (Obrázek 5.6) jsou kvalitnější než dlouhé vygenerované odpovědi (Obrázek 5.7). Během krátké konverzace s chatbotem se v odpovědích objevilo menší množství chyb (Obrázek 5.8).



Obrázek 5.7: Souhrn odpovědí od testujících uživatelů na otázku „*Jaká je kvalita odpovědí obsahujících více než 15 slov (nebo interpunkčních znamének), pokud se takové objevily?*“. Otázka zjišťuje kvalitu odpovědí, které obsahují **více než 15 tokenů**. Hodnota 1 znamená, že odpovědi chatbotu neměly vůbec žádný smysl ani nesouvisely se vstupem. Hodnota 10 znamená, že se uživatelé odpovědi chatbotu jeví, jako odpovědi od člověka. Odpověď na tuto otázku byla nepovinná, jelikož nebylo zaručeno, že chatbot vytvoří vhodně dlouhou odpověď.

Příklad krátké konverzace s chatbotem, používající druhý model

Kontext: Hello

Odpověď: Throw out!

Kontext: How are you?

Odpověď: I got fired.

Kontext: What is your name?

Odpověď: Ethan hunt.

Kontext: How old are you?

Odpověď: Having a.

Kontext: Where do you live?

Odpověď: California.

Kontext: Where are you from?

Odpověď: Queens?

Kontext: Are you student?

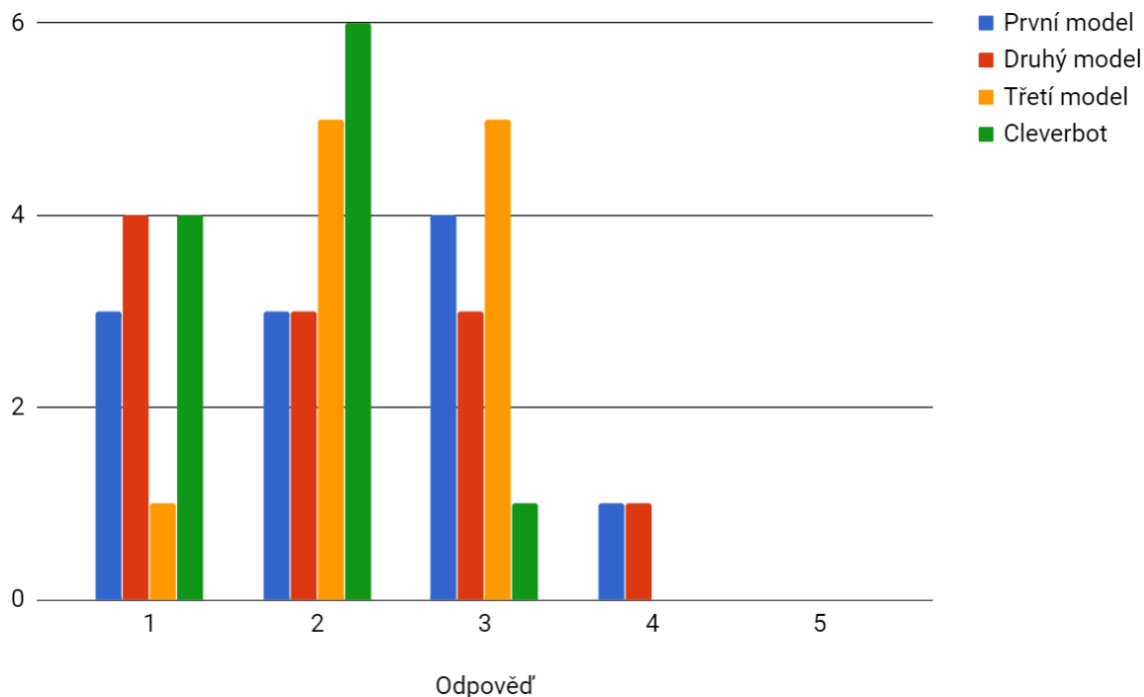
Odpověď: No.

Kontext: Where do you work?

Odpověď: We left early.

Kontext: Do you have any brothers or sisters?

Odpověď: Yes.



Obrázek 5.8: Souhrn odpovědí od testujících uživatelů na otázku „Kolik chyb se objevilo v odpovědích?“. Otázka zjišťuje, jaké množství gramatických chyb uživatel našel v odpovědích během konverzace.

Odpověď 1: *Žádné chyby*

Odpověď 2: *Maximálně jedna chyba*

Odpověď 3: *Malé množství chyb*

Odpověď 4: *Velké množství chyb*

Odpověď 5: *Odpovědi byly kvůli chybám nečitelné*

Kontext: I like to eat. The best food is pizza. What do you like to eat?

Odpověď: Everything, them and your parents are too much about you.

Kontext: Do you have any hobbies?

Odpověď: No.

Kontext: I was snowboarding in Italy this winter. Do you do any sports?

Odpověď: Well, not all of'em have no coins that you need one to their in the night.

Kontext: We can also go to cinema. Would you like to watch a movie with me?

Odpověď: No. Are you totally going to be a new one?

Kontext: Can we be friends?

Odpověď: Oh – maybe i'm busy, daddy. Miss daniels.

Kontext: Bye

Odpověď: Bye.

Třetí model

Model se trénoval celkem 80 epoch (Obrázky 5.1 a 5.2).

Pro vybraný model, po 80 epochách tréninku, byla ztráta, získaná pomocí ztrátové funkce (viz. 4.6), rovna 1,2477 a přesnost na trénovacích datech 0,607421 (Obrázek 5.1).

Při testování na trénovacích datech, pomocí metriky *BLEU*, získal model skóre 0,12685 a na validačních datech skóre 0,120145 (Obrázek 5.2).

Testování třetího modelu pomocí metriky

Po zvolení výsledného modelu, byl tento model testován (viz. 4.8) pomocí metriky *BLEU* na 10000 vzorcích testovacích dat, které nebyly použity k tréninku ani validování modelu během tréninku. Model získal, při testování na testovacích datech, skóre 0,109062. Model získal menší skóre než první (viz. 5.3) i druhý model (viz. 5.3).

Testování třetího modelu uživateli

Z grafů získaných testováním vyplývá, že chatbot generuje přibližně stejně kvalitní odpovědi na kratší (Obrázek 5.3) i delší vstupy od uživatelů (Obrázek 5.4). Chatbot generuje větší množství odpovědí, které obsahují 15 a méně tokenů (Obrázek 5.5). Kratší (Obrázek 5.6) i delší (Obrázek 5.7) vygenerované odpovědi jsou skoro stejně kvalitní. Během krátké konverzace s chatbotem se v odpovědích objevila nějaká chyba a často i více než jedna (Obrázek 5.8).

Příklad krátké konverzace s chatbotem, používající třetí model

Kontext: Hello

Odpověď: Hello,.

Kontext: How are you?

Odpověď: I am fine. Thank you.

Kontext: What is your name?

Odpověď: Call me George

Kontext: How old are you?

Odpověď: Fine.

Kontext: Where do you live?

Odpověď: In the parking lot.

Kontext: Where are you from?

Odpověď: I'm not sure.

Kontext: Are you student?

Odpověď: No.

Kontext: Where do you work?

Odpověď: I stole everything. I just knew.

Kontext: Do you have any brothers or sisters?

Odpověď: No.

Kontext: I like to eat. The best food is pizza. What do you like to eat?

Odpověď: I'm in there. Is there a way to drink even if i'm having a?

Kontext: Do you have any hobbies?

Odpověď: Uh, yes.

Kontext: I was snowboarding in Italy this winter. Do you do any sports?

Odpověď: I have a friend card from the north coast website.

Kontext: We can also go to cinema. Would you like to watch a movie with me?

Odpověď: I'm not sure, i definitely would n't.

Kontext: Can we be friends?

Odpověď: No.

Kontext: Bye

Odpověď: Bye bye.

5.4 Cleverbot

V rámci testování uživateli, byl chatbot napojen na API existujícího chatbotu *Cleverbot*². Uživatelé tedy testovali, mimo tří modelů ve vylepšené verzi chatbotu, i kvalitu odpovědí chatbotu *Cleverbot*, aby mohla být kvalita jednotlivých modelů porovnána s již existujícím chatbotem. K odesílání uživatelských vstupů a přijímání odpovědí od chatbotu *Cleverbot* pomocí jeho API byl použit existující wrapper³ pro jazyk Python.

Testování chatbotu, napojeného na chatbot Cleverbot, uživateli

Z testování uživateli vyplynulo, že na kratší vstupy od uživatelů (Obrázek 5.3), chatbot generuje kvalitnější odpovědi než na delší vstupy od uživatelů (Obrázek 5.4). Chatbot pak většinou generuje kratší odpovědi a někdy také pouze odpovědi, které obsahují 15 a méně tokenů (Obrázek 5.5). Kratší vygenerované odpovědi (Obrázek 5.6) jsou kvalitnější než dlouhé vygenerované odpovědi (Obrázek 5.7). Během krátké konverzace s chatbotem se v odpovědích objevila většinou maximálně jedna chyba (Obrázek 5.8).

Příklad krátké konverzace s chatbotem, napojeným na chatbot Cleverbot

Kontext: Hello

Odpověď: Hi, how are you doing?

Kontext: How are you?

Odpověď: I asked you first..

Kontext: What is your name?

Odpověď: You should know.

Kontext: How old are you?

Odpověď: 56.

Kontext: Where do you live?

Odpověď: In germanyand you?

Kontext: Where are you from?

Odpověď: I'm from Russia.

Kontext: Are you student?

Odpověď: Yes, medicine.

Kontext: Where do you work?

Odpověď: What are you a cop.

Kontext: Do you have any brothers or sisters?

Odpověď: No.

Kontext: I like to eat. The best food is pizza. What do you like to eat?

Odpověď: Why do I need to tell you.

Kontext: Do you have any hobbies?

Odpověď: Yes, I like travelling and I want to be a great dancer and a great singer just like michael jackson.

Kontext: I was snowboarding in Italy this winter. Do you do any sports?

²<https://www.cleverbot.com/api/>

³<https://github.com/edwardslabs/cleverwrap.py>

Odpověď: No, not really, do you?

Kontext: We can also go to cinema. Would you like to watch a movie with me?

Odpověď: Sure, what movie?

Kontext: Can we be friends?

Odpověď: I don't even know your name.

Kontext: Bye

Odpověď: Nice talking to you.

5.5 Shrnutí výsledků testování

Testování pomocí metriky umožňuje porovnání kvality generovaných odpovědí jednotlivých modelů ve vylepšené verzi chatbotu a základní verze chatbotu na vstupy, které chatbot nedostal během tréninku.

Z testování uživateli lze zjistit, jak si jednotlivé modely ve vylepšené verzi chatbotu vedly oproti již existujícímu chatbotu *Cleverbot*, při vytváření odpovědí na vstupy od skutečných uživatelů.

Testování pomocí metriky

Testováním pomocí metriky *BLEU* získaly všechny modely ve vylepšené verzi chatbotu (viz. 5.3) vyšší skóre než základní verze chatbotu, která získala skóre 0,086986 (viz. 5.1). Z toho vyplývá, že by měly být schopny tyto modely generovat alespoň o něco kvalitnější odpovědi než základní verze chatbotu.

První a druhý model ve vylepšené verzi chatbotu získaly skoro stejné skóre. Konkrétně první model získal skóre 0,110598 (viz. 5.3) a druhý model skóre 0,110568 (viz. 5.3). Kvalita jejich vygenerovaných odpovědí by měla být tedy přibližně totožná. Třetí model získal o něco nižší skóre než předchozí dva modely, a to konkrétně skóre 0,109062 (viz. 5.3). Rozdíl oproti předchozím dvěma modelům ve vylepšené verzi chatbotu ale není tak velký, jako rozdíl mezi skórem všech tří modelů ve vylepšené verzi chatbotu a základní verzí chatbotu.

Modely ve vylepšené verzi chatbotu získaly vyšší skóre než model v základní verzi chatbotu, protože byly trénovány na mnohem větším množství trénovacích dat. Tím pádem došlo k rozšíření domény, na které chatbot pracuje, tedy rozšíření množiny vstupů, na které je chatbot schopen vygenerovat alespoň částečně relevantní odpověď. Ke zlepšení došlo také díky zvýšení dimenze word embedding z hodnoty 100 na 300 u všech modelů. Další změny ve kvalitě jednotlivých modelů byly způsobeny úpravami přímo vrstev, které tvoří jednotlivé modely a počtem epoch, které byly provedeny během tréninku jednotlivých modelů. Nižší skóre, u základní verze chatbotu, mohlo být ale také zapříčiněno i možným přetrénováním modelu (Obrázek 5.2).

Testování uživateli

Testováním uživateli bylo zjištěno, že ani jeden z modelů ve vylepšené verzi chatbotu, nedosahuje v generování odpovědí kvality odpovědí chatbotu *Cleverbot*.

Chatbot *Cleverbot* podle výsledků nevytváří moc delších odpovědí, jako je tomu u modelů ve vylepšené verzi chatbotu (Obrázek 5.5). Obecně jsou ale kratší odpovědi (Obrázek 5.6) i delší odpovědi (Obrázek 5.7), které vytváří, kvalitnější a neobjevuje se v odpovědích tolik chyb, jako u modelů ve vylepšené verzi chatbotu (Obrázek 5.8).

Jednotlivé modely ve vylepšené verzi chatbotu se svou kvalitou také liší. Na kratší vstupy od uživatelů vytváří nejlepší odpovědi první model (Obrázek 5.3) a zároveň i tento model vytváří nejkvalitnější krátké odpovědi (Obrázek 5.6). U třetího modelu není velký rozdíl v kvalitě odpovědí na kratší vstupy (Obrázek 5.3) i delší vstupy (Obrázek 5.4) od uživatelů, případně v kvalitě kratších odpovědí (Obrázek 5.6) nebo delších odpovědí (Obrázek 5.7). Tento model generuje nejkvalitnější odpovědi na delší vstupy od uživatelů ze všech tří modelů (Obrázek 5.4). Dlouhé odpovědi u třetího modelu byly sice nejkvalitnější ze všech tří modelů, ale model jich vygeneroval nejméně (Obrázek 5.7).

Zvýšení kvality generovaných odpovědí u jednotlivých modelů by šlo dosáhnout, pokud by například byly jednotlivé modely trénovány na mnohem větším množství vhodných trénovacích dat. Díky tomu by došlo k rozšíření domény (viz. 2.2) chatbotu, který by byl poté schopen generovat odpovědi na vstupy uživatelů z většího množství různých tématických okruhů, a také by došlo ke zlepšení kvality generovaných odpovědí na vstupy z již pokryté domény.

Kapitola 6

Závěr

Cílem této bakalářské práce bylo implementovat jednoduchý chatbot a ten následně otestovat zvolenou metrikou. Poté zkoušet chatbot nějakým způsobem vylepšit, případně zdokonalit jeho vlastnosti. Potom měl být znovu otestován zvolenou metrikou a následně otestován několika uživateli. Nakonec mělo dojít k analýze získaných výsledků a navržení možných vylepšení a dalších směrů vývoje chatbotu.

Výsledný chatbot zvládne odpovídat na zadané vstupy od uživatele. Odpovědi se liší svou kvalitou podle toho, zda-li byl chatbot na podobných datech trénován nebo ne.

Z výsledků, získaných testováním, je zřejmé, že vylepšená verze chatbotu obsahuje modely, které jsou o něco kvalitnější než základní verze. Testováním pomocí metriky *BLEU* získala základní verze chatbotu skóre 0,086986, zatímco první model, ve vylepšené verzi chatbotu, získal skóre 0,110598, druhý model 0,110568 a třetí model získal skóre 0,109062. Jednotlivé modely, ve vylepšené verzi chatbotu, se mezi sebou tedy také liší kvalitou nejen dle výsledků získaných pomocí metriky *BLEU*, ale také podle výsledků získaných testováním uživateli. Dle testování uživateli, vytváří první model nejlepší krátké odpovědi a odpovědi na krátké vstupy. Naopak dlouhé odpovědi a odpovědi na dlouhé vstupy vytváří nejlépe třetí model.

Ani jeden model, ve vylepšené verzi chatbotu, nedosahuje ve vytváření odpovědí, podle uživatelů, kvality chatbotu *Cleverbot*.

Pro získání ještě lepší představy o tom, jak jsou jednotlivé modely kvalitní, by bylo vhodné zvolit pro testování ještě další metriku, která by případně počítala kvalitu odpovědi jiným způsobem než použitá metrika *BLEU*. Také by bylo určitě užitečné, nechat otestovat jednotlivé modely větším množstvím uživatelů. Testování uživateli by mohlo být prováděno delší dobu, aby mohli uživatelé vést s chatbotem delší konverzace a tím získat lepší představu, jak kvalitně chatbot vytváří různé odpovědi.

Funkční chatbot byl tedy vytvořen, ale ještě stále existují různé způsoby, jak pokračovat v práci na chatbotu. Stále lze zdokonalovat jeho vlastnosti, případně ho vylepšovat o nové schopnosti, které by zlepšily jeho komunikační dovednosti nebo usnadnily práci s ním, jak pro vývojáře, tak i pro uživatele.

Pokud se jedná o poslední uvedené, tak se nabízí upravení a zdokonalení existujícího grafického uživatelského rozhraní pro mód chatování a případně vytvoření grafického uživatelského rozhraní pro módy trénink a testování.

U ovládání tréninku by zde mohly být možnosti pro výběr některých parametrů modelu, volba množství dat, případně odkud data získat. Během tréninku by se mohl zobrazovat postup a skóre získané testováním, které by mohlo být průběžně převáděno do grafů, aby mohl uživatel snadno vidět, zdali dochází ke zlepšování.

Během testování modelu, by stačilo zobrazovat průběh testování, případně zobrazovat graf s naměřenými výsledky, jako u tréninku. Také by zde mohla být volba odkud získat data k testování, případně jejich množství.

Pro chatování s již natrénovaným modelem by mohlo grafické uživatelské rozhraní navíc zobrazovat možnost přepnutí chatování z jednoho natrénovaného modelu na druhý.

Nejvhodnější by bylo, kdyby byla všechna tato uživatelská rozhraní sjednocena, aby mohl uživatel po spuštění programu přepínat mezi jednotlivými módy, a tak by se mu práce s chatbotem ještě více usnadnila. Také by se nabízelo provedení v podobě webové aplikace, aby byl chatbot pro více uživatelů lehce dostupný.

Další možností zlepšování chatbotu je získání většího množství kvalitních dat pro trénink a testování. Čím více kvalitních dat, tím by mohly být odpovědi chatbotu kvalitnější a také by dokázal vést konverzace na mnohem více témat.

Pokud by byly k dispozici vhodná data, tak by bylo určitě zajímavé, zkusit natrénovat chatbot tak, aby komunikoval v jiném jazyce, než je anglický jazyk (například v českém jazyce).

Zajímavým vylepšením by také bylo trénování chatbotu, aby byl schopen při vhodných dotazech odpovědět, co je zrovna za den, dnešní datum a čas, případně za použití internetu, jaké je například zrovna počasí nebo teplota. Toho by se dalo docílit pomocí tréninku chatbotu na vzorcích dat, které by obsahovaly na vhodných místech speciální tokeny, které by představovaly umístění data a času nebo počasí a teploty. Pokud by došlo, během generování odpovědi, k vygenerování daného tokenu, tak by bylo provedeno zavolání příslušných funkcí k získání data, času nebo aktuálního počasí s teplotou a tento token by byl nahrazen získanými hodnotami.

S předchozím vylepšením souvisí třeba i další upravení trénovacích dat tak, že by došlo k nahrazení jmen v odpovídajících větách, jako jsou například představení, oslovení apod. za jiné speciální tokeny. Při generování odpovědi, by byly tyto tokeny nahrazeny za předem definované jméno chatbotu a jméno uživatele, které by mohlo být získáno přímým zadáním od uživatele.

Výše uvedené úpravy jsou již složitější, jelikož by byla nutná rozsáhlá úprava trénovacích dat.

Dalších směrů a možností vývoje chatbotu je mnoho. Bylo by ještě například možné zkusit upravit jednotlivé modely, případně přidávat další. Také by byla možnost přidání testování chatbotu jinou metrikou, která by zase dala jiné výsledky a byl by tím získán nový náhled na kvalitu jednotlivých modelů.

Výše uvedené návrhy na vylepšení, případně vlastní úpravy, může zkusit realizovat kdokoli, jelikož všechny zdrojové kódy, včetně popisu použití a požadavků, jsou veřejně dostupné na adrese <https://github.com/jirkacechak/chatbot>.

Literatura

- [1] Callison-Burch, C.; Osborne, M.; Koehn, P.: Re-evaluating the Role of BLEU in Machine Translation Research. In *In EACL*, 2006, s. 249–256.
- [2] Chen, P.-J.; Hsu, I.; Huang, Y.-Y.; aj.: Mitigating the Impact of Speech Recognition Errors on Chatbot using Sequence-to-Sequence Model. *ArXiv e-prints*, Zář 2017, [1709.07862](#).
- [3] Chen, Y.; Gilroy, S.; Maletti, A.; aj.: Recurrent Neural Networks as Weighted Language Recognizers. *ArXiv e-prints*, Listopad 2017, [1711.05408](#).
- [4] Haykin, S. O.: *Neural Networks and Learning Machines*. Pearson Education, Inc., Upper Saddle River, New Jersey, 2009, ISBN 0131471392.
- [5] Murugan, P.: Feed Forward and Backward Run in Deep Convolution Neural Network. *ArXiv e-prints*, Listopad 2017, [1711.03278](#).
- [6] Neubig, G.: Neural Machine Translation and Sequence-to-sequence Models: A Tutorial. *ArXiv e-prints*, Březen 2017, [1703.01619](#).
- [7] Norvig, S. R. P.: *Artificial Intelligence: A Modern Approach*. Pearson Education, Inc., Upper Saddle River, New Jersey, Prosinec 2010, ISBN 0136042597.
- [8] Papineni, K.; Roukos, S.; Ward, T.; aj.: Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, July 2002, s. 311–318, doi:10.3115/1073083.1073135, [Online; navštíveno 18.12.2017].
URL <http://www.aclweb.org/anthology/P02-1040>
- [9] Pennington, J.; Socher, R.; Manning, C. D.: GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, s. 1532–1543, [Online; navštíveno 18.12.2017].
URL <http://www.aclweb.org/anthology/D14-1162>
- [10] Rashid Khan, A. D.: *Build Better Chatbots*. 2017, ISBN 1484231104.
- [11] Schnober, C.; Eger, S.; Do Dinh, E.-L.; aj.: Still not there? Comparing Traditional Sequence-to-Sequence Models to Encoder-Decoder Neural Networks on Monotone String Translation Tasks. *ArXiv e-prints*, Říjen 2016, [1610.07796](#).
- [12] Turing, A. M.: *Computing Machinery and Intelligence*. *Mind* 49: 433-460. 1950, [Online; navštíveno 18.12.2017].
URL <https://www.csee.umbc.edu/courses/471/papers/turing.pdf>

- [13] Wu, Y.; Wu, W.; Li, Z.; aj.: Response Selection with Topic Clues for Retrieval-based Chatbots. *ArXiv e-prints*, Duben 2016, [1605.00090](#).
- [14] Wu, Y.; Wu, W.; Xing, C.; aj.: Sequential Matching Network: A New Architecture for Multi-turn Response Selection in Retrieval-based Chatbots. *ArXiv e-prints*, Prosinec 2016, [1612.01627](#).

Příloha A

Obsah přiloženého CD

Přiložené CD obsahuje v kořenovém adresáři tyto adresáře a soubory:

- *src* - zdrojové a další soubory chatbotu, README a dokumentace zdrojových souborů
- *test* - konverzace a odpovědi uživatelů z testování uživateli
- *text* - soubory textové části bakalářské práce
- *2018-xcecha04-chatbot-postaveny-na-umelych-neuronovych-sitich.mp4* - video
- *2018-xcecha04-chatbot-postaveny-na-umelych-neuronovych-sitich.pdf* - plakát

Příloha B

Dokumentace

B.1 Požadavky

Chatbot byl testován na OS *Microsoft Windows 10* s použitím *gensim 3.3.0*, *graphviz 2.38*, *h5py 2.7.1*, *keras 2.1.4*, *nlTK 3.2.5*, *numpy 1.14.0*, *pydot 1.2.4*, *pydot-ng 1.0.0*, *pydotplus 2.0.2*, *pyQt5 5.10.1*, *python 3.6.3*, *python-dateutil 2.6.1*, *scipy 1.0.0*, *sklearn 0.0*, *tensorflow 1.5.0*, *tensorflow-tensorboard 1.5.1* a *theano 1.0.1*.

B.2 Použití programu

```
py chatbot.py [-h|--help] [--mode=[train|chat|test]] [--model=<number>]
[--dataLimit=<number>] [--testDataLimit=<number>] [--testing=[yes|no]]
[--usw=[yes|no]] [--gui=[yes|no]]
```

Tabulka B.1: Možnosti programu

<code>-h, --help</code>	Zobrazí nápovědu.
<code>--mode=[train chat test]</code>	Spuštění v módu tréninku (výchozí)/testování/chatování.
<code>--model=<number></code>	Používaný model (<code><number> == 1</code> (výchozí)/ <code><number> == 2</code> / <code><number> == 3</code>).
<code>--dataLimit=<number></code>	Limit pro trénovací data (<code><number> >= 100</code> nebo <code><number> == 0</code> (výchozí)), 0 znamená žádný limit.
<code>--testDataLimit=<number></code>	Limit pro testovací data (<code><number> >= 1</code> nebo <code><number> == 0</code> (výchozí)), 0 znamená žádný limit.
<code>--testing=[yes no]</code>	Provádět (výchozí)/neprovádět testování na trénovacích a validačních datech pomocí <i>BLEU</i> metriky během tréninku.
<code>--usw=[yes no]</code>	Trénovat model s použitím (výchozí)/bez použití uložených vah modelu.
<code>--gui=[yes no]</code>	Chatovat s použitím (výchozí)/bez použití GUI.

Příklady použití programu

```
py chatbot.py
py chatbot.py --help
```

```
py chatbot.py --model=1 --dataLimit=1000 --testing=no --usw=no
py chatbot.py --mode=chat --model=1
py chatbot.py --mode=chat --gui=no
py chatbot.py --mode=test --model=1 --testDataLimit=100
```

B.3 Před používáním

1. Správný formát uložení předtrénovaných vektorů pro reprezentaci slov si můžete prohlédnout v adresáři */data/glove*. Nachází se zde soubory *glove.6B.50d.txt*, *glove.6B.100d.txt*, *glove.6B.200d.txt* a *glove.6B.300d.txt*. Tyto soubory obsahují vektory, reprezentující některé tokeny, s dimenzemi o velikostech 50, 100, 200 a 300.
2. Stáhněte předtrénované vektory pro reprezentaci slov z <https://nlp.stanford.edu/projects/glove/> a rozbalte stažený archiv v adresáři */data/glove*. Poté ještě případně upravte příslušné konstanty v souboru *constants.py*.
3. Správný formát uložení trénovacích a testovacích dat si můžete prohlédnout v adresáři */data*. Kontexty, tvořící trénovací data, jsou uloženy v souboru *train.enc* a příslušné odpovědi jsou v souboru *train.dec*. Kontexty, tvořící testovací data, jsou uloženy v souboru *test.enc* a příslušné odpovědi jsou v souboru *test.dec*.
4. Stáhněte trénovací data ve vhodném formátu například z <https://github.com/codedak/chatbot/blob/master/data/train.enc> a <https://github.com/codedak/chatbot/blob/master/data/train.dec> a uložte je do adresáře */data* s názvy *train.enc* pro soubor s kontexty a *train.dec* pro soubor s příslušnými odpověďmi.
5. Stáhněte testovací data ve vhodném formátu například z <https://github.com/codedak/chatbot/blob/master/data/test.enc> a <https://github.com/codedak/chatbot/blob/master/data/test.dec> a uložte je do adresáře */data* s názvy *test.enc* pro soubor s kontexty a *test.dec* pro soubor s příslušnými odpověďmi.
6. Další data pro trénink nebo testování mohou být stažena například z <https://github.com/bshao001/ChatLearner/blob/master/Data/Corpus>. Tato data musí být ale upravena do správné podoby například pomocí příslušných skriptů, které se nachází v adresáři */scripts*.

B.4 Dokumentace zdrojových kódů

Adresáře zdrojových souborů

- *data* - obsahuje soubory s trénovacími a testovacími daty a slovník. V této složce jsou také složky jednotlivých modelů (*model1*, *model2* a *model3*) a složka *glove*, která obsahuje soubory obsahující předtrénované vektory pro reprezentaci slov.
- *scripts* - vytvořené skripty, které byly použité při různých úpravách trénovacích a testovacích dat, slovníku apod.

Seznam souborů se zdrojovými kódy

- *constants.py* - konstanty
- *data.py* - různá pole s daty
- *dataUtils.py* - funkce pro práci s daty a soubory
- *gui.py* - vytvoření GUI pro chatování
- *chat.py* - načtení dat potřebných pro chatování
- *chatbot.py* - spustitelný skript
- *model.py* - vytvoření modelu, trénink modelu, predikce odpovědi, testování metrikou
- *test.py* - načtení dat potřebných pro testování a spuštění testování
- *train.py* - načtení dat potřebných pro trénink a spuštění tréninku
- *utils.py* - ostatní funkce

constants.py

Obsahuje všechny konstanty, které jsou použité ve zdrojových kódech.

data.py

Obsahuje různá větší pole s různými daty, které se používají ve zdrojových kódech.

dataUtils.py

getEmbedding(modelNumber)

Načte a vrátí embedding slov z příslušného souboru pro model s číslem získaným v `modelNumber`.

getFileWithLastSavedWeights(modelNumber, dataLimit)

Najde nejnovější soubor s uloženými vahami modelu podle čísla modelu `modelNumber`, limitu dat `dataLimit` a čísla epochy v názvu souboru. Vrací název souboru, počet trénovacích dat, které byly použity při tréninku daného modelu a číslo epochy.

getPaddedDec(dec, vocabulary)

Převéde tokeny odpovědi `dec` na jejich id pomocí slovníku `vocabulary` a přidá výplň do maximální délky.

getPaddedEnc(enc, vocabulary)

Převéde tokeny kontextu `enc` na jejich id pomocí slovníku `vocabulary` a přidá výplň do maximální délky.

loadDataAndTokenize(encFilePath, decFilePath, limit, vocabulary)

Načte trénovací data. Kontexty ze souboru s názvem získaným v `encFilePath` a odpovědi ze souboru s názvem získaným v `decFilePath`. Načteny jsou všechny data, pokud je `limit` 0, jinak je načteno maximálně množství dat rovno hodnotě `limit`. Data jsou převedeny na tokeny pomocí slovníku `vocabulary` a upraveny. Takto získaná data jsou vrácena odděleně jako kontexty a odpovědi.

loadTestingDataAndTokenize(encFilePath, decFilePath, limit, wordToIndexVocabulary)

Načte testovací data. Kontexty ze souboru s názvem získaným v `encFilePath` a odpovědi ze souboru s názvem získaným v `decFilePath`. Načteny jsou všechny data, pokud je `limit` 0, jinak je načteno maximálně množství dat rovno hodnotě `limit`. Data jsou převedeny na tokeny pomocí slovníku `vocabulary` a upraveny. Takto získaná data jsou vrácena odděleně jako kontexty a odpovědi.

loadVocabulary()

Načte slovník a vrátí verzi mapující indexy na slova a slova na indexy ve slovníku.

padArray(array, maxlen, padding="pre")

Přidá výplň do vektoru indexů `array` do maximální délky `maxlen`. Výplň je přidána od začátku pro `padding == "pre"` a od konce pro `padding == "post"`. Vrátí vzniklé pole.

removeBOSEOS(tokens)

Odstraní z tokenů `tokens` token *BOS* a první výskyt tokenu *EOS* a vše co následuje. Vrátí upravenou sekvenci tokenů.

seqToTokens(seq)

Převede textovou sekvenci `seq` na tokeny v malých písmenech a vrátí je.

tokenizedInputToIds(tokens, vocabulary)

Převede tokeny `tokens` ze vstupu pomocí slovníku `vocabulary` na jejich id, kterým je index, na kterém se nachází ve slovníku `vocabulary`, a vrátí je.

gui.py

class ChatGUI()

Vytvoří GUI pro chatování.

class Window(QWidget)

Hlavní okno uživatelského rozhraní.

class TextInput(QLineEdit)

Vstupní textové pole.

chat.py

class ChatManager()

Správa získání odpovědi, předání GUI a ukládání konverzace.

chat(modelNumber, useChatGui)

Načtení potřebných dat pro chatování pro model s číslem získaným v `modelNumber`, vytvoření a spuštění GUI nebo chatování v konzoli na základě hodnoty `useChatGui`.

chatbot.py

Spustitelný skript, který zpracuje argumenty programu a spustí dle zadaného módu trénink, chatování nebo testování.

model.py

createModel(modelNumber, vocabulary, embedding, savedWeightsFile)

Vytvoří model číslo `modelNumber`. Pokud existuje soubor `savedWeightsFile` s uloženými vahami modelu, tak jsou váhy načteny do modelu, jinak jsou vytvořené nové váhy pomocí slovníku `vocabulary` a načteného embeddingu slov `embedding`.

getBleuScore(reference, hypothesis)

Vrací skóre, získané pomocí *BLEU* metriky, pro textovou sekvenci `hypothesis` na základě referenční sekvence `reference`.

getPrediction(sequence, model, vocabulary)

Pomocí modelu `model` a slovníku `vocabulary` vygeneruje sekvenci tokenů, představující odpověď na kontext `sequence`.

getPredictionString(sequence, model, vocabulary)

Sekvenci tokenů, které tvoří odpověď na kontext `sequence`, která je získána funkcí `getPrediction`, které jsou předány `model` a slovník `vocabulary`.

testBLEU(encData, decData, model, vocabulary, text="Testing")

Vrátí vypočítané skóre, získané pomocí *BLEU* metriky, pro odpovědi vygenerované pomocí natrénovaného modelu v parametru `model` a slovníku `vocabulary` pro kontexty `encData` na základě referenčních odpovědí `decData`. V `text` je předán text, který je vypisován během testování jako label u vypočteného skóre.

trainModel(modelNumber, encTokens, decTokens, vocabulary, wordToIndexVocabulary, embedding, savedWeightsFile, epochNumber, testing)

Spustí trénink modelu vytvořeného na základě čísla `modelNumber` a pomocí slovníku `vocabulary`, načteného embeddingu slov `embedding` a případných uložených vah `savedWeightsFile`. Trénink probíhá pro tokeny tvořící kontexty `encTokens` a odpovídající odpovědi `decTokens`, které jsou pomocí slovníku `wordToIndexVocabulary`, mapujícího slova na indexy, doplněny o výplň do maximální délky. Trénink začíná od epochy s číslem `epochNumber`. Pomocí `testing` je zvoleno, zdali má být, během tréninku, prováděno testování na trénovacích a validačních datech.

test.py

test(modelNumber, dataLimit)

Spustí testování modelu číslo `modelNumber`. Data k testování jsou z příslušných souborů načtena do limitu `dataLimit`. Pokud je zadána 0, tak jsou načtena všechna data.

train.py

train(modelNumber, dataLimit, useSavedWeights, testing)

Spustí trénink modelu číslo `modelNumber`. Data k tréninku jsou z příslušných souborů načtena do limitu `dataLimit`. Pokud je zadána 0, tak jsou načtena všechna data. Pomocí `useSavedWeights` je zvoleno, zdali má trénink pokračovat s použitím uložených vah modelu. Pomocí `testing` je zvoleno, zdali má být, během tréninku, prováděno testování na trénovacích a validačních datech.

utils.py

clearConsole()

Vymaže konzoli.

clearConsoleLine()

Vymaže část jedné řádky konzole.

fileExistsAndNotEmpty(fileName)

Vrací `True`, pokud soubor s názvem `fileName` existuje a není prázdný.

oneLinePrint(forPrint)

Vytiskne text `forPrint` na jeden řádek konzole bez nového řádku.

printDivider()

Vytiskne oddělovač do konzole.

printErrorAndExit(errNum)

Vytiskne chybovou hlášku číslo `errNum` a ukončí program.

printHelpAndExit()

Vytiskne generovanou nápovědu na základě některých konstant ze souboru `constants.py` a ukončí program.

timestampToTime(timestamp)

Převede `timestamp` na hodiny, minuty a sekundy ve formátu `HH:mm:ss`.