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

Bakalářská práce

Vícejazyčné rozpoznávání dialogových aktů

Plzeň 2022 Jiří Trefil

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Akademický rok: 2021/2022

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Jiří TREFIL**
Osobní číslo:	**A20B0260P**
Studijní program:	**B0613A140015 Informatika a výpočetní technika**
Specializace:	**Informatika**
Téma práce:	**Vícejazyčné rozpoznávání dialogových aktů**
Zadávající katedra:	**Katedra informatiky a výpočetní techniky**

Zásady pro vypracování

1. Seznamte se s dodanými datovými kolekcemi pro automatické rozpoznávání dialogových aktů (DA).
2. Prostudujte relevantní metody automatického rozpoznávání dialogových aktů založené na neuronových sítích.
3. Na základě předchozí studie navrhněte a implementujte prototyp systému pro automatické rozpoznávání dialogových aktů, které jsou ve více jazycích. V systému budou integrovány dvě různé klasifikační metody.
4. Funkčnost prototypu otestujte na dodaných datových kolekcích.
5. Výsledky diskutujte a navrhněte další možná rozšíření.

Rozsah bakalářské práce: **doporuč. 30 s. původního textu**
Rozsah grafických prací: **dle potřeby**
Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

Dodá vedoucí bakalářské práce.

Vedoucí bakalářské práce: **Doc. Ing. Pavel Král, Ph.D.**
Katedra informatiky a výpočetní techniky

Datum zadání bakalářské práce: **4. října 2021**
Termín odevzdání bakalářské práce: **5. května 2022**

L.S.

Doc. Ing. Miloš Železný, Ph.D.
děkan

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

V Plzni dne 14. října 2021

Prohlášení

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

V Plzni dne 5. května 2022

Jiří Trefil

Abstract

This bachelor thesis deals with multi-lingual dialogue act recognition. It explains problems associated with this task and describes neural network architectures of models that were used in experiments. Most of the current research on the topic of automatic dialogue act recognition is conducted in only one language. The aim of this bachelor thesis is to propose and implement system for multi-lingual dialogue act recognition.
The available data sets contain English, German and Spanish utterances with labeled dialogue acts. Four different classification techniques were used for the task of multi-lingual dialogue act recognition. Two topologies of convolutional neural networks, BILSTM neural network and transformer neural network. Experiments have shown that it is possible to implement this system and recognize dialogue acts with good accuracy.

Abstrakt

Tato bakalářská práce se zabývá vícejazyčným rozpoznáváním dialogových aktů. V práci je vysvětlena problematika této úlohy společně s architekturami neuronových sítí, které byly využity k její řešení.
Většina současných přístupů automatické rozpoznávání dialogových aktů se provádí pouze v jednom jazyce, vícejazyčným rozpoznáváním dialogových aktů se téměř nikdo nezabýval. Cílem práce je proto navrhnout a implementovat systém pro rozpoznávání dialogových aktů ve více jazycích. Dostupné datové sady obsahují anglický, německý a španělský jazyk.
K rozpoznávání dialogových aktů ve více jazycích byly použity čtyři různé klasifikační metody. Dvě topologie konvoluční neuronové sítě, síť typu BILSTM a síť typu transformer. Experimenty ukázaly, že je možné takovýto systém vytvořit a rozpoznávat dialogové akty s velmi dobrou přesností.

Poděkování

Rád bych poděkoval doc. Ing. Pavlu Královi, Ph.D za ochotu, dobré rady a hlavně velké množství času, které mi věnoval při vypracování bakalářské práce.

Obsah

1 Úvod

Dialog je rozhovor dvou či více osob, který vede k výměně informací. Jednou ze základních charakteristik dialogů je dialogový akt (DA), který reprezentuje funkci věty v dialogu. Například funkce věty "Co jsi měl dneska za tričko?" je získání informací, zatímco funkcí věty "Dneska mám modré tričko." je předání informací. Rozpoznávání dialogových aktů se v současné době realizuje ze zvukového signálu, případně z jeho textového přepisu. Většina současných přístupů automatického rozpoznávání dialogových aktů se provádí pouze v jednom jazyce. Vzhledem k velkému množství dat ve více jazycích roste důležitost vícejazyčného zpracování.
Cílem této práce je proto navrhnout a implementovat systém pro rozpoznávání dialogových aktů ve více jazycích. Vstupem systému bude text v podobě přepisů z audio nahrávek dialogů. Zaměříme se na angličtinu, němčinu a španělštinu, protože dostupná data jsou v těchto jazycích.
Pro rozpoznávání dialogových aktů budou použity hluboké neuronové sítě, protože dosahují v současné době nejlepších výsledků ve většině úloh z oblasti zpracování přirozeného jazyka. V práci budou prozkoumány a porovnány jejich hlavní topologie. Dále bude navrženo řešení problému vícejazyčného zpracování.
Struktura práce je následující. Nejdříve je popsán dialogový akt a dialogový systém, který je jednou z nejdůležitější aplikací dialogových aktů. Další kapitola obsahuje stručný popis neuronových sítí a procesu jejich učení. Dále je rozvedena architektura neuronových sítí, které chci použít pro rozpoznávání dialogových aktů ve více jazycích. V následující kapitole jsou popsány datové sady použité v této práci. Konkrétně se jedná o datasety DIHANA a Verbmobil, které jsou použity pro vytvoření vstupních dat. Další kapitola se zabývá analýzou problému rozpoznávání dialogových aktů, popisuje extrakci dat z DIHANA korpusu, vytvoření vícejazyčné datové sady, použité algoritmy pro vektorovou reprezentaci slov a detailní popis klasifikátorů, které budou využity pro rozpoznávání dialogových aktů ve více jazycích. V další kapitole je popsána implementace preprocesoru pro DIHANA korpus a následné vytvoření datasetu pro španělský jazyk. Dále je zde popsán a zobrazen proces předzpracování dat pro neuronové sítě a konečně samotná implementace klasifikátorů. V předposlední kapitole jsou popsány použité evaluační metriky, průběh trénování neuronových sítích a jejich výsledky při rozpoznávání dialogových aktů. V kapitole závěr shrnuji výsledky a navrhuji pokračování této práce.

2 Dialogový akt a dialogový systém

2.1 Dialogový akt

Austin [3] definuje dialogový akt (DA) jako význam výpovědi na úrovni ilukučního aktu. Jinými slovy, dialogový akt slouží jako funkce věty (či její části) v dialogu. Například funkcí otázky je žádost o informace, zatímco funkcí odpovědi je poskytnutí informací. Dialogové akty mohou být použity v kontextu porozumění mluvené řeči (angl. Spoken Language Understanding / SLU). V SLU systémech je dialogový akt definován přesněji, ale je také závislý na doméně použití systému. Jeong et al. [16] definují dialogový akt jako záměr závislý na oblasti použití, například 'Ukázat let' či 'Koupit letenku' v systému pro rezervaci letů.
V tabulce 2.1 je zobrazen fragment dialogu mezi osobou A a B, jednotlivé promluvy jsou anotovány dialogovými akty.

Mluvčí	**Dialogový akt**	**Dialog**
A	Konvenční zahájení konverzace	Haló?!
B	Konvenční zahájení konverzace	Ahoj!
B	Tvrzení	Tady Michal.
B	Otázka	Jak se ti daří?
A	Konvenční zahájení konverzace	Čau, Michale!
A	Tvrzení	Jde to.
A	Otázka	Co ty?
B	Tvrzení	Já se mám dobře.

Tabulka 2.1: Ukázka dialogu mezi osobou A a B s dialogovými akty.

Dialogové akty mají řadu aplikací z nichž nejdůležitější jsou dialogové systém, proto si je popíšeme dále.

2.2 Dialogový systém

Dialogový systém je počítačový program, jehož účelem je komunikovat s člověkem pomocí přirozené řeči. Architektur dialogových systému je celá řada, ale všechny sdílí základní komponenty:

- Rozpoznávač řeči (*ASR, Automatic Speech Recognition*)
- Modul pro porozumění přirozenému jazyku (*NLU, Natural Language Understanding*)
- Dialogový manažer (*DM,Dialogue Manager*)
- Generátor přirozeného jazyka (*NLG,Natural Language Generator*)
- Syntetizér řeči (*TTS,Text-to-speech Synthesis*)

Základní schéma dialogového systému je zobrazeno na obrázku 2.1. V následujících kapitolách příblížím funkčnost jednotlivých komponent.

2.2.1 Rozpoznávač řeči

Komponenta má za úkol převedení spontánní řeči do textové podoby. Model rozpoznávání řeči se dělí na model akustický, který modeluje, jak se jednotlivé hlásky slov vyslovují a na model jazykový, který modeluje, jak se řadí slova do vět [25].

2.2.2 Modul pro porozumění přirozenému jazyku

Komponenta převádí textový přepis řeči do sémantické reprezentace. Mezi používané formalismy reprezentace sémantické informace patří dialogové akty.

2.2.3 Dialogový manažer

Komponenta reaguje na vstup uživatele a generuje odpověď systému. Typicky se dělí na dvě části:

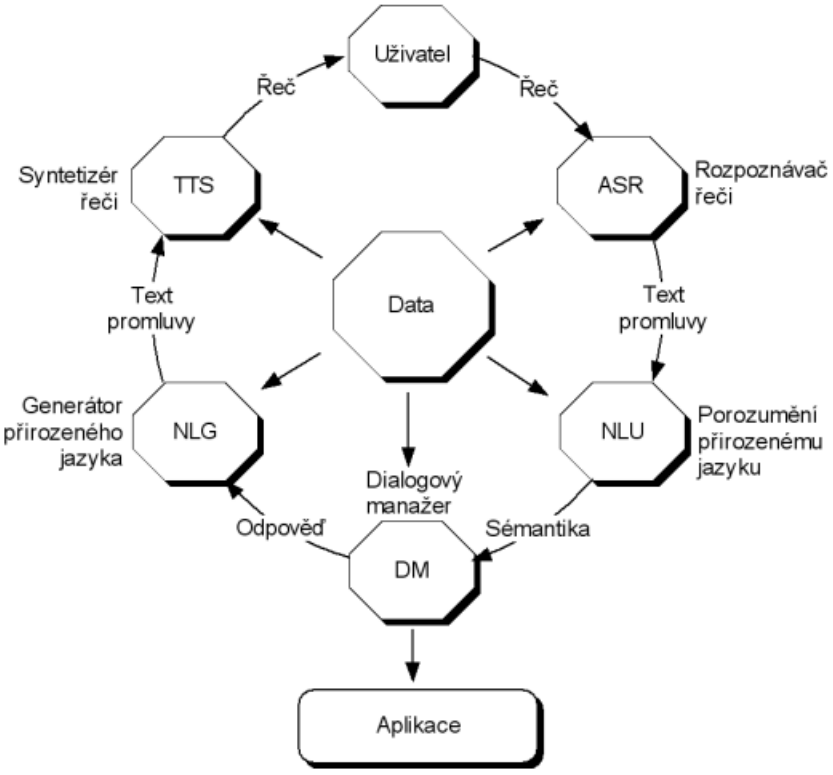
- Model dialogu - modeluje stav dialogu.
- Strategie řízení dialogu - určuje následující akci dialogového systému na základě odhadnutého stavu.

2.2.4 Generátor přirozeného jazyka

Komponenta z formální reprezentace odpovědi, získané z dialogového manažeru, generuje textovou podobu výsledné promluvy. Modul generování odpovědi je v praktických dialogových systémech většinou založen na šablonách [31]. Šablony jsou ručně napsané věty pro různé dialogové akty, ve kterých jsou některé části nahrazeny proměnnými. Při generování dialogového aktu vybere komponenta nejvhodnější šablonu a proměnné v šabloně se nahradí hodnotami atributů z dialogového aktu.

2.2.5 Syntetizér řeči

Komponenta převádí promluvu v textové podobě na akustický signál, který je následně přehrán uživateli. Proces syntézy se skládá z předzpracování textu a následné syntézy zvukového signálu.



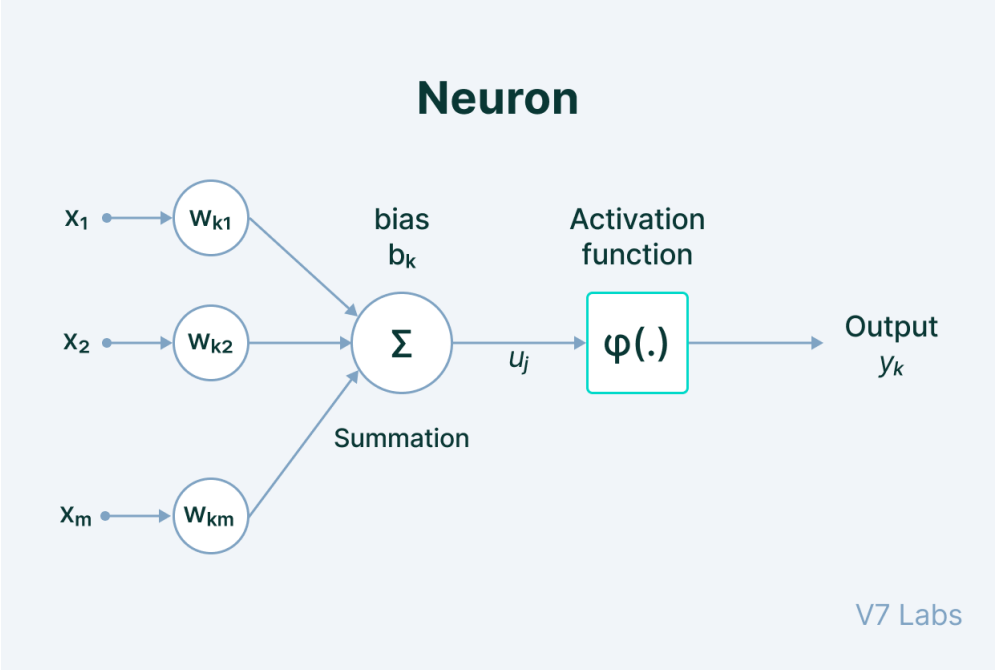
Obrázek 2.1: Schéma dialogového systému [23].

3 Neuronové sítě

V této práci budou pro rozpoznávání dialogových aktů použity umělé neuronové sítě. Proto si je stručně přiblížíme v dalším textu.

3.1 Umělé neuronové sítě

Umělá neuronová síť, obvykle nazývána pouze neuronová síť je matematický model inspirovaný neuronovou sítí biologickou. Neuronová síť je kolekce propojených bodů (neuronů). Každé propojení může přenášet informace dalšímu neuronu. Vstup neuronu je reálné číslo, například prvek vektoru. Vstupní hodnota je přepočítána pomocí nelineární funkce neuronu. Tato hodnota je výstupní hodnota neuronu.



Obrázek 3.1: Model neuronu [4].

Na obrázku 3.1 je zobrazen model jednoho neuronu, který obsahuje následující informace:

- $x_1 ... x_m$ - vstupy neuronu.
- $w_{k1} ... w_{km}$ - váhy neuronu.

- bias b_k - prahová konstanta slouží k posunu hodnoty výstupu aktivační funkce.
- Summation (součet) - transformační funkce, která vykoná váhový součet.
- Activation function (aktivační funkce) - nelineární funkce.

3.2 Učení neuronové sítě

Učení neuronové sítě je proces, při kterém jsou modelu opakovaně předávána data z trénovací množiny. Cílem tohoto procesu je najít optimální hodnotu vah neuronů. Metody a algoritmy používány při učení jsou rozvedeny níže.

3.2.1 Učení s učitelem

Učení s učitelem používá spočetnou konečnou trénovací množinu A dvojic x a y, které představují vstupy a odpovídající korektní výstupy řešené úlohy. V případě klasifikace DA se jedná o větu s anotovaným dialogovým aktem.

3.2.2 Částečné učení s učitelem

Jedná se o speciální případ metody učení s učitelem. V množině trénovacích dat je malé množství dat označeno a větší množství neoznačeno. Tato metoda vychází alespoň z jednoho z předpokladů:

- Předpoklad kontinuity - body, které jsou blízko sebe sdílejí příslušnost ke stejné třídě.
- Shlukový předpoklad - data formují diskrétní shluky a jednotlivé body ve shlucích sdílejí příslušnost ke stejné třídě.

3.2.3 Učení bez učitele

V tomto případě data nejsou vůbec nijak označena a model musí samostatně objevit vznikající vzory - třídy. Používá shlukovací algoritmy [18].

3.2.4 Ztrátová funkce

V kontextu optimalizačního algoritmu se ztrátová funkce (angl. loss function) používá pro vyhodnocení daného řešení (množina vah neuronů). Tuto funkci také nazýváme *objektivní (účelovou) funkcí* [8], jejíž úkolem je minimalizovat

numerickou hodnotu ztráty. Při trénování modelu je snaha minimalizovat chybu modelu přes všechna trénovací data. Při výpočtu chybovosti modelu je klíčové zvolit takovou ztrátovou funkci, která zachytí všechny aspekty řešeného problému. V případě rozpoznávání dialogových aktů potřebujeme zařadit DA do jedné z n tříd. Pro tento problém je vhodné zvolit funkci *křížové entropie* [5], která vyjadřuje míru rozdílu mezi dvěma rozděleními pravděpodobnosti pro danou proměnnou.

3.2.5 Proces učení

V prvním kroku učení inicializuje váhy neuronů náhodně. Tato konfigurace samozřejmě přinese špatné výsledky, tj. vysoká ztrátová hodnota a nízká úspěšnost klasifikace. Při tomto procesu je chtěné **snižovat** hodnotu ztráty a **zvyšovat** úspěšnost klasifikace.

3.2.6 Hluboké učení a zpětná propagace

Hluboké učení (angl. deep learning) je typ algoritmu strojového účení, který využívá vícevrstvou architekturu neuronové sítě k postupné extrakci příznaků z nezpracovaného vstupu [19]. Modely využívající tento učící proces obecně fungují podle principu: Každá skrytá vrstva je určena k identifikaci objektů na jiné úrovni (například hrany či celá zvířata). Každá skrytá vrstva extrahuje příznaky, které přímo využívá vrstva následující.
Například skryté vrsty blíže vstupu mohou identifikovat například hrany. Tento výstup předají následující skryté vrstvě, která z hran umí identifikovat další vzory na vyšší úrovni abstrakce (např. ucho či tlamu).
Zpětná propagace chyby (backpropagation) je optimalizační algoritmus užívaný při učení neuronových sítí. Algoritmus je speciální případ metody gradientího sestupu (optimalizační algoritmus pro hledání lokálního minima funkce). Používá se pouze při metodě *učení s učitelem* , kdy známe správný výstup k aktuálnímu vstupu.
Při učení neuronové sítě:

1. Aplikují se vstupní data a postupně se směrem vpřed napočítá výstup sítě (vstupní signál se sítí šíří směrem dopředu).

2. Výstup ze sítě se porovná s požadovaným výstupem, tj. spočte se chyba.

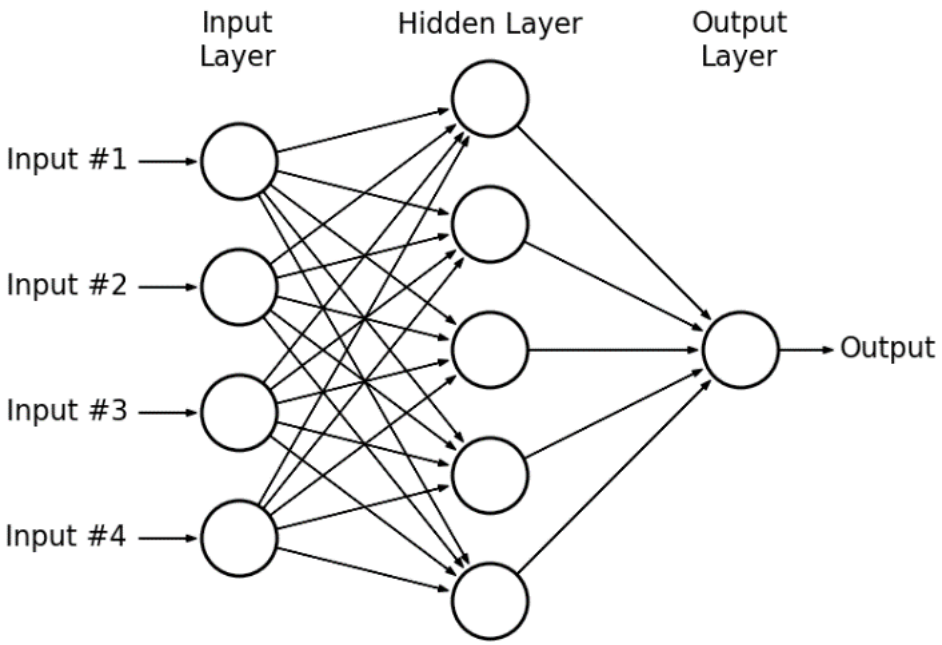
3. Spočítá se gradient chybové funkce, tj. závislost chyby na vahách neuronů. Výpočet postupuje zpět od výstupní vrstvy až po vstupní vrstvu. Váhy se zároveň mění podle jejich vlivu na chybu. To zaručuje, že pro stejný vstup bude chyba menší.

3.3 Architektury neuronových sítí

Architektur neuronových sítí existuje mnoho, stejně jako u dialogových systému, všechny však sdílí základní princip. Neuronová síť se skládá ze vstupní vrstvy, skryté vrstvy (není povinná, model může mít pouze vstupní a výstupní vrstvu. Slouží například pro extrakci příznaků) a výstupní vrstvy. Vstupní vrstva (angl. input layer) není tvořena neurony ve smyslu zavedené definice, jedná se pouze o uzly realizující identickou kopii hodnot vstupního vektoru na vstupy všech neuronů. Skrytá vrstva je vrstva mezi vrstvou vstupní a výstupní. V této vrstvě neurony ze vstupních dat aktivační funkcí vypočítají výstup a předají jej další vrstvě. Výstupní vrstva je poslední vrstva neuronové sítě, jejím výstup je výstup samotného programu, který neuronovu síť využívá.
V dalším textu popíši architekturu neuronových sítí, které byly použity k rozpoznávání dialogových aktů.

3.3.1 Vícevrstvý perceptron

Vícevrstvý perceptron umožňuje klasifikaci do tříd, které nejsou lineárně oddělitelné. Vícerstvý perceptron má **vstupní vrstvu**, alespoň jednu **skrytou vrstvu** a **výstupní vrstvu**.
Příklad architektury vícevrstvého perceptronu je zobrazen na obrázku 3.2.



Obrázek 3.2: Příklad vícevrstvého perceptronu využívajícího jednu skrytou vrstvu [12].

3.3.2 Konvoluční neuronová síť

Konvoluční neuronová síť (CNN) je architektura široce využívána zejména v problémech z oblasti počítačového vidění. Tato architektura používá tři typy vrstev:

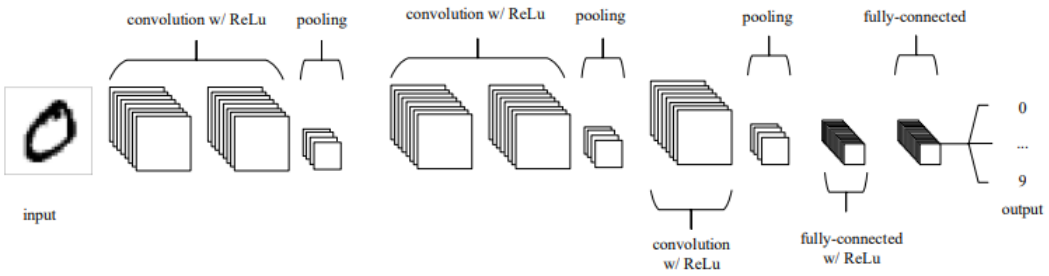
- Konvoluční vrstva (Convolutional layer)
- Sdružovací vrstva (Pooling layer)
- Plně propojená vrstva (Fully connected layer)

Konvoluční vrstva je hlavním stavebním blokem této architektury a je také výpočetně nejnáročnější. Vrstva se skládá z n filtrů (často také označováno jako jádro, kernel) o rozměrech $N \times M$, například $n = 40$, $N = 3$, $M = 3$. Samotný filtr se skládá z vah.
Vstupní data konvoluční vrstvy jsou ve tvaru matice či vektoru a na výstupu vrstvy je vektor či matice příznaků (feature vector/matrix), který vzniká skalárním součinem filtrů a vstupních dat. Počet filtrů ovlivňuje rozměr vektoru příznaků. 40 filtrů vygeneruje 40 různých vektorů příznaků a celková dimenze výstupu bude tedy $N \times M \times 40$ [14].
Pohyb filtru po vstupních datech udává parametr krok (stride), který určuje

o kolik prvků (sloupců matice či položek vektorů) se filtr pohybuje na vstupních datech.
Sdružovací vrstva (angl. pooling layer) slouží k redukci dimenzionality vstupních dat. Stejně jako konvoluční vrstva využívá i tato vrstva filtry. Podstatný rozdíl je ten, že tyto filtry nemají váhy, nýbrž aplikují agregační funkci. Existují dvě hlavní agregační funkce:

- Max pooling
- Average pooling

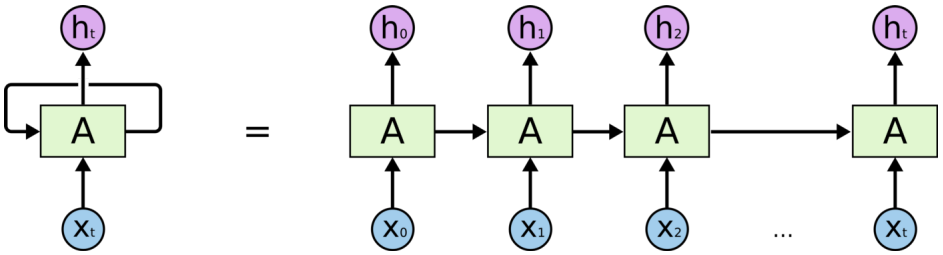
První funkce vybírá maximální hodnotu z aktuálního umístění filtru. Například z 3×3 filtru vybere nejvyšší hodnotu ze všech 9 možných a ta zůstane zachována, ostatní hodnoty budou ztraceny. Druhá funkce vypočítá průměrnou hodnotu ze všech 9 možných a zachová ji. Pomocí těchto hodnot je plněn výstupní vektor.
Plně propojená vrstva u CNN architektury reprezentuje vrstvu výstupní a její úkol je provést klasifikaci nad extrahovanými daty z předchozích vrstev. Ukázka konvoluční sítě je zobrazena na obrázku 3.3.



Obrázek 3.3: Konvoluční neuronová síť [22].

3.3.3 Rekurentní neuronové sítě

Rekurentní neuronové sítě (RNN) jsou typem neuronových sítí, které jsou navržené pro práci s daty sekvenčního charakteru, tj. hodnota na pozici x_i závisí na hodnotě na pozici x_{i-1}. RNN mají paměť, která jim pomáhá uchovávat stav (informaci) o předchozích vstupech a začlenit tuto informaci při výpočtu výstupu. Tato paměť se nazývá *skrytý stav* (angl. hidden state). Skrytý stav je vektor, který uchovává informaci o předešlých vstupech a propaguje se dalším vrstvám [6]. Na obrázku 3.4 je uvedena ukázka rekurentní sítě.



Obrázek 3.4: Rekurentní neuronová síť [9].

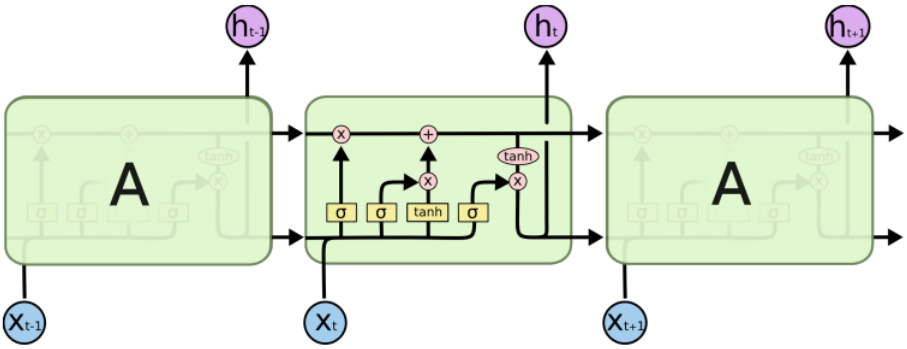
3.3.4 Long short-term memory

Long Short Term Memory (LSTM) [13] je speciálním typem rekurektní neuronové sítě. Tato architektura byla navržena, aby opravila nedostatky standartních RNN. Problémem těchto sítí je fakt, že si nedokáží zapamatovat dlouhodobé závislosti, tzv. *vanishing gradient problem* . LSTM toho schopna je.

Klíčem úspěchu LSTM sítí je pomocný vektor (angl. cell state), na obrázku 3.5 je zobrazen jako horizontální linie vedoucí skrze horní části LSTM buňky. LSTM sítě mají schopnost tento vektor modifikovat pomocí bran (angl. gates). Rozlišujeme tři typy bran: zapomínací, vstupní a výstupní. Brány jsou na obr. 3.5 označeny písmenem σ.

Každá brána je neuronová vrstva s aktivační funkcí sigmoid. Brána na základě vektoru x_t a h_{t-1} rozhodne, zda je důležité pamatovat si aktuální hodnotu na vstupu. Toto chování zajišťuje relevanci předchozí informace v dlouhodobé paměti.

Posledním krokem v LSTM vrstvě je vytvoření výstupního vektoru. Tento vektor se vytvoří kombinací aktuální dlouhodobé paměti (angl. cell state) a nově získané informace. V rámci této práce byla použita BI-LSTM, tj. oboustranně zřetězená LSTM síť.



Obrázek 3.5: Architektura LSTM sítě [9].

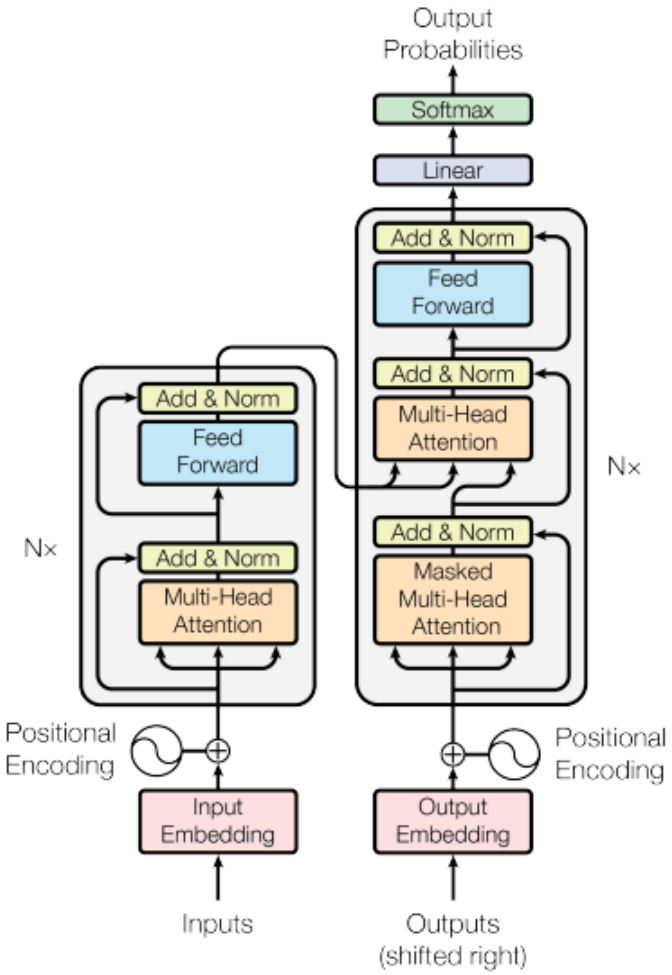
3.3.5 Enkodér/dekodér architektura

Tato architektura se skládá ze dvou hlavních komponent, enkodéru a dekodéru. Komponenty je možné použít spolu či nezávisle na sobě. Architektura se hojně používá v tzv. sequence-to-sequence problémech, jako je například překlad vět z jazyka A do jazyku B. Enkodér i dekodér mohou být například rekurentní sítě. Enkodér na vstupu přijme sekvenci dat proměnlivé délky (například větu) a vytvoří vektor příznaků (embedding vector) konstantní délky. Dekodér má funkci inverzní, na vstupu přijme vektor konstantní délky a vytvoří sekvenční data proměnlivé délky.
Například při překladu z angličtiny do češtiny: enkodér přijme větu *Hi, how are you?* do vektoru příznaků zakóduje sematické hodnoty slov a vytvoří vektor délky K, kde K je přirozené číslo, například 128. Dekodér tento vektor přijme na vstupu a podle semántické hodnoty jednotlivých položek vektoru vygeneruje příslušná slova v cíleném jazyce, tedy *Ahoj, jak se máš?*
Hlavní výhodou této architektury je její flexibilita. Pro vstup libovolné délky enkodér vygeneruje vektor konstantní délky.

3.3.6 Transformer architektura

Transformer architektura se řídí enkodér/dekodér architekturou. Enkodér mapuje sekvenci vstupních symbolů $(i_1, ..., i_n)$ na sekvence $o = (o_1, ..., o_n)$. Ze sekvence o poté dekodér generuje sekvenci výstupních symbolů.
V článku [27] navržený transformer model obsahuje za sebou spojené bloky obsahující tzv. self-attention blok a plně propojenou dopřednou neuronovou síť. Tyto bloky jsou použity pro enkodér i dekodér. Tato architektura je zobrazena na obrázku 3.6.
V původním článku se enkodér skládá ze šesti identických vrstev. Každá

vrstva má dvě parciální vrstvy. První parciální vrstvou je tzv. multi-head self-attention mechanismus. Druhou vrstvou je obyčejná dopředná plně propojená neuronová síť.
Dekodér je stejně jako enkodér složen ze šesti identických vrstev. Oproti enkodéru ale dekodér obsahuje dvě vrstvy multi-head self-attention. Vstupem této přidané vrstvy je výstup z enkodér bloku. Původní struktura attention vrstvy je pozměněna, aby se předešlo k přístupu k následujícím datům, tzn. odhad pro pozici i závisí pouze na známých výstupech na pozici menší než i [27].
Funkcí Attention vrstvy je zachycení vazeb mezi jednotlivými vstupními tokeny. Například ve větě: *Ta liška to být nemohla. Myslivec ji dneska ráno střelil.* bude mít token 'ji' nejsilnější vazbu k tokenu 'liška'.



Obrázek 3.6: Transfomer architektura [27].

4 Datové sady

V rámci této bakalářské práce budou použity následující datové sady, které byly dodány vedoucím bakalářské práce.

4.1 DIHANA

4.1.1 Představení projektu

Rozvoj dialogových systémů [2] umožnil řešení problémů ve specifických oblastech, například pro rezervaci jízdenek či konzultaci jízdních řádů. Projekt DIHANA (DH) vznikl s cílem vytvořit modulární dialogový systém, který zpracovává požadavky na rezervační služby. Pro omezení domény zpracovává systém pouze požadavky na zjištění ceny celostátních vlakových jízdenek a dotazy na jízdní řád.

4.1.2 Korpus

Projekt DIHANA [17] měl za úkol získat informace o španělských celostátních vlacích za použití telefonu. V projektu bylo definováno 300 různých scénářů za účelem kontroly dialogu s uživatelem. Scénář je definován:

- Cílem: informace, kterou chce uživatel.
- Situací: specifické okolnosti v relaci s žádostí o cestě.
- Specifické požadavky cesty: typ cesty, město odjezdu, město příjezdu a jedna či více restrikcí.

Příklad scénáře:

- Cíl: získání ceny jízdenky
- Specifické požadavky:

  Typ cesty: jednosměrná cesta

  Situace: cesta na svatbu
- Město odjezdu: Valencie
- Město příjezdu: Barcelona
- Restrikce: odjezd v pátek, příjezd v sobotu, cestovat se společností Euromed

4.1.3 Ukázka dialogu

Následující tabulka 4.1 ukazuje fragment dialogu přeložený do angličtiny a češtiny mezi počítačem a uživatelem. V prvním sloupci je identifikátor, zda mluví stroj (M) či uživatel (U).

M	Bienvenido al servicio de información de trenes . ¿ En qué puedo ayudarle? .
U	Me gustaría saber los horarios del tren Euromed de Barcelona a Valencia.
M	¿Desea viajar de Barcelona a Valencia?
U	Sí.
M	¿Tiene que viajar hoy?
U	No, next thursday.
M	Estoy buscando horarios de Barcelona a Valencia para el 15 de julio. Un momento, por favor.
	Anglický překlad
M	Welcome to the information system for nationwide trains, what information would you like ?
U	I would like to know the timetables of the Euromed train from Barcelona to Valencia.
M	Do you wish to travel from Barcelona to Valencia?
U	Yes.
M	Do you with to travel today?
U	No, next Thursday.
M	I am looking for timetables from Barcelona to Valencia for the 15th of July. One moment, please.
	Český překlad
M	Vítejte v informačním systému celostátních vlaků, jakou informaci si přejete?
U	Chtěl bych znát jízdní řády vlaku Euromed z Barcelony do Valencie.
M	Přejete si cestovat z Barcelony do Valencie?
U	Ano.
M	Přejete si jet dnes?
U	Ne, příští týden ve čtvrtek.
M	Vyhledávám jízdní řády z Barcelony do Valencie v den 15. července. Chvíli strpení, prosím.

Tabulka 4.1: Dialog mezi uživatelem a strojem při konzultaci jízdních řádů.

4.1.4 Wizard of Oz metoda

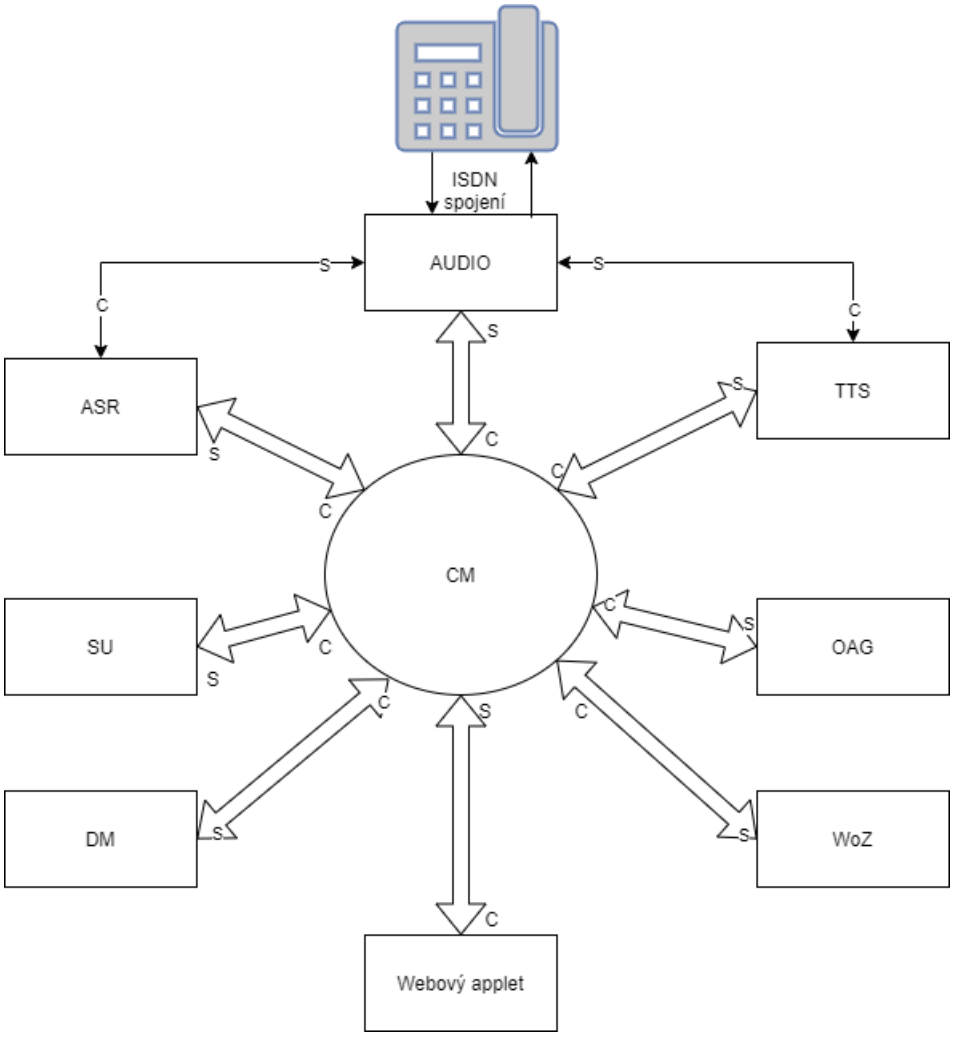
Wizard of Oz (WoZ) je technika v oboru interakce mezi člověkem a počítačem (angl. human-computer interaction). Princip této techniky spočívá v tom, že uživatel komunikuje se systémem o kterém si myslí, že je automní. Systém je ale ve skutečnosti řízen operátorem. V případě projektu DIHANA je operátorem umělá inteligence, která komunikuje s uživatelem pomocí generátoru ústních odpovědí (GRO) a pomocí serveru pro převod textu na řeč (CTV). Umělá inteligence s uživatelem komunikuje v případě nutnosti doplnění informací, potvrzení či objasnění požadavku a validace informací. Odpověď systému je generována na základě informací poskytnutých uživatelem a pomocí připravených scénářů v databázi.

4.1.5 Akvizice korpusu

Architektura platformy pro akvizici korpusu se skládá z osmi komponent, které jsou zobrazeny na obrázku 4.1.

- Audio server (AUDIO) - server zasílá zprávu WoZ serveru když uživatel zavolá. Zasláním této zprávy dojde k zahájení konverzace mezi uživatelem a systémem.
- Server pro automatické rozpoznávání řeči (RAH) - server zasílá rozpoznané věty WoZ serveru a serveru pro porozumění řeči s využitím komunikačního manažera.
- Server pro porozumění řeči (CH) - server využívá akustické modely pomocí kterých zasílá zprávy WoZ serveru.
- Wizard of Oz server (MO) - server naslouchá promluvám uživatele a používá rozpoznané věty společně se zprávami z akustických modelů pro řízení dialogu. Výsledný dialog je zaslán serveru pro převod textu na řeč.
- Server pro správu dialogů (GD) - server byl při akvizici korpusu nahrazen WoZ serverem, který simuluje jeho funkcionalitu.
- Generátor ústních odpovědí (GRO) - generuje odpověď v přirozeném jazyce.
- Server pro převod textu na řeč (CTV) - server provádí konverzi textu na zvukové signály.
- Klient pro správu komunikace (GC)

Komunikace mezi moduly je realizovaná prostřednictvím paketů za použití TCP/IP protokolu. Veškerá komunikace mezi moduly s vyjímkou audio souborů probíhá v XML formátu s využitím komunikačního manažera, který je kontrolován appletem spuštěným v prohlížeči.



Obrázek 4.1: Architektura platformy u WoZ (s=server,c=client).

4.1.6 Analýza korpusu

Komunikací mezi 225 lidmi a WoZ telefoním informačním systémem bylo získáno 900 dialogů. Dialogy jsou ručně anotovány a rozloženy do třech úrovní. První úroveň představuje obecný záměr dialogu nezávisle na kontextu úkolu, zatímco ostatní dvě úrovně jsou vázany ke kontextu. Tyto dvě úrovně mohou nabírat hodnoty NIL, která informuje o absenci úrovně. První úroveň anotace jsou dialogové akty, které se klasifikují do 11 tříd. Jejich distribuce spolu s anglickým a českým překladem je zobrazena v tabulce 4.2. Dále je uveden příklad dialogových aktů, viz tabulka 4.3.

DA španělsky	**DA anglicky**	**DA česky**	**Celkem**	%
Pregunta	Question	Otázka	6338	27
Respuesta	Answer	Odpověď	4285	18
Confirmacion	Confirmation	Potvrzení	3629	15
Nueva consulta	New consult	Nová konzultace	2474	11
Espera	Waiting	Čekání	1948	8
Cierre	Closing	Ukončení konz.	1827	8
Afirmacion	Acceptance	Potvrzení	990	4
Apertura	Opening	Úvodní výrok	900	4
No entendido	Not understood	Neporozumění dotazu	657	3
Indefinida	Indefinite	Neurčité	159	1
Negacion	Negative	Negativní odpověď	340	1

Tabulka 4.2: Rozložení dialogových aktů v DIHANA korpusu.

DA	**Příklad**	**Překlad**
Question	hola mira que quería ir a Málaga.	Zdravím, chci jet do Malagy.
Feedback	Hay varios tipos de trenes.	Existuje několik typů vlaků.
Confirmation	¿ Quiere viajar a Gijón?.	Chcete cestovat do Gijónu?
New consult	Desea algo más?	Chcete ještě něco dalšího?
Waiting	Un momento por favor.	Chvíli počkejte, prosím.
Acceptance	Sí.	Ano.
Opening	¿ En qué puedo ayudarle?	Jak mohu pomoci?
Not understood	¿ Puede repetir su consulta?.	Můžete zopakovat Váš dotaz?
Indefinite	Quiero saber.	Chci to vědět.
Negative	No.	Ne.
Closing	No, muchas gracias.	Ne, mnohokrát děkuji.

Tabulka 4.3: Ukázky dialogových aktů z DIHANA korpusu.

4.1.7 Trénovací a testovací množina

Rozdělení datové sady na trénovací množinu a na testovací množinu je důležitý krok při učení neuronové sítě. Trénovací množina slouží pro učení modelu. Testovací množina slouží k evaluaci modelu. Tyto dvě množiny musí být disjunktní. Kdyby nebyly, model by byl evaluován na datech, která se již naučil a výsledky evaluace modelu by byly zavádějící.
Rozdělení trénovací a testovací množiny je provedeno podle článku [17]. Trénovací množina obsahuje 70% dat a testovací množina obsahuje zbylých 30% dat.

4.1.8 Struktura dat v datasetu

Z audio nahrávek dialogů mezi uživateli a systémem byla vytvořena databáze dialogů v textové podobě. Databáze se skládá z 225 adresářů. Každý tento adresář reprezentuje jednoho uživatele systému. Ke každému uživateli jsou dostupné 4 dialogy, které vedl se systémem.
Každý dialog je popsán v několika souborech, pro rozpoznávání dialogových aktů je nejdůležitější soubor typu dia. Tento soubor obsahuje ruční anotaci dialogových aktů ke každé promluvě uživatele či systému. Více o tomto souboru v kapitole 5.1.

4.2 VERBMOBIL

4.2.1 Představení projektu

VERBMOBIL (VM) byl dlouhodobý mezioborový výzkumný projekt z oblasti strojového překladu. Cílem bylo vytvořit systém, který dokáže rozpoznat, přeložit a generovat výroky v přirozeném jazyce. Systém dokázal oboustranně překládat Německý/Anglický jazyk a Německý/Japonský jazyk. Výzkum trval od roku 1993 do roku 2000 a byl financován Německým federálním ministerstvem pro výzkum a technologie (Bundesministerium für Forschung und Technologie) [28].

4.2.2 Použití dialogových aktů

Dialogové akty jsou v systému použity různými komponentami [15]. Ačkoliv tyto komponenty mezi sebou sdílí základní porozumění dialogových aktů, tak mají různou perspektivu na detaily. V systému lze rozlišovat mezi třemi hlavními aplikacemi DA.

1. stanovení DA pro anglické výroky založené na klíčových slovech.
2. stanovení DA pro německé výroky založené na mikro a makro strukturální infomaci.
3. použití dialogové aktu v dialogové komponentě.

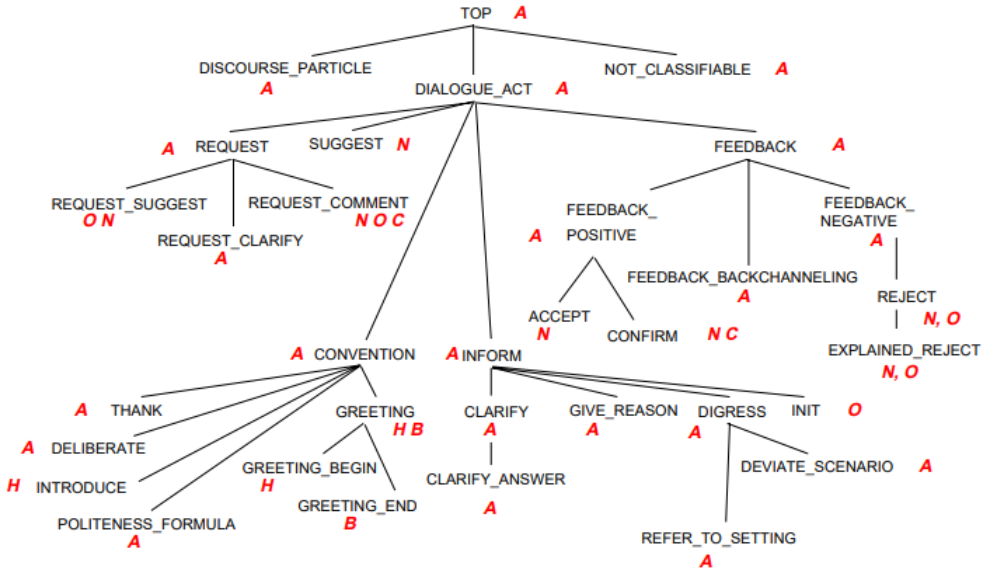
4.2.3 Korpus

V rámci projektu byl proveden empirický sběr dat sedmi univerzitami v Německu, Japonsku a Americe [29]. Hlavní účel tohoto sběru bylo vytvoření robustního korpusu spontánních mluvených dialogů a jejich anotace pro natrénování akustických modelů [26], které vytvářejí slovník pro překladač.

4.2.4 Dialogové akty ve VM

Dialogové akty jsou hierarchicky strukturované. K anotaci dialogových aktů korpusu se používá rozhodovací strom, viz obrázek 4.2. Při anotaci se postupně prochází strom od kořene směrem k listům. Listy stromu reprezentují klasifikační třídy (dialogové akty). V každém uzlu stromu je položena otázka. Podle odpovědi na otázku rozhodne, kam se ve stromu dále vydat.
První otázkou zjistíme, zda je věta pochopitelná či ne. Pokud není pochopitelná, tak je zařazena do kategorie *NOT_CLASSIFIABLE* a proces končí.

Kategorie *DISCOURSE_PARTICLE* nebyla při anotaci použita. V případě srozumitelnosti věty nastane rozhodovací proces, který rozhodne o dialogovém aktu.



Obrázek 4.2: Rozhodovací strom pro anotaci dialogových aktů [1].

4.2.5 Příklad dialogového aktu

Tato sekce obsahuje ukázky dialogových aktu z korpusu Verbmobil pro německý a anglický jazyk.
V tabulkách 4.4 a 4.5 jsou ukázány příklady dialogových aktů pro německý a anglický jazyk.

DA	**Promluva**	**DA česky**	**Překlad promluvy**
INTRODUCE	Hello mein name ist Gurtner.	Představení se	Zdravím, mé jméno je Gurtner.
INIT	ich rufe an um mit Then einen termin fur ein funft Arbeitstreffen.	Zahájení konverzace	Volám, abych si s vámi domluvil pátou pracovní schůzku.

Tabulka 4.4: Příklad dialogových aktů pro německý jazyk.

DA	**Promluva**	**DA česky**	**Překlad promluvy**
SUGGEST	Hello. I would like to make an appointment with you in the next week.	Návrh	Zdravím, chtěl bych si s Vámi domluvit schůzku na příští týden.
FEEDBACK	Well, that is a pretty rough week. I am pretty busy.	Zpetná vazba	No, to je rušný týden. Nemám moc čas.

Tabulka 4.5: Příklad dialogových aktů pro německý jazyk.

4.2.6 Analýza korpusu

Pro trénování modelů je k dispozici VB korpus anglických a německých výroků anotovaných příslušnými dialogovými akty. Rozdělení na trénovací a testovací množiny spolu s četnostmi dialogů a dialogových aktů je popsáno v tabulkách 4.6 a 4.7.

počet	trénovací množina	testovací množina
Dialog	6 485	940
Dialogové akty	9 599	1 420

Tabulka 4.6: Statistické informace o VM korpusu pro Anglický jazyk.

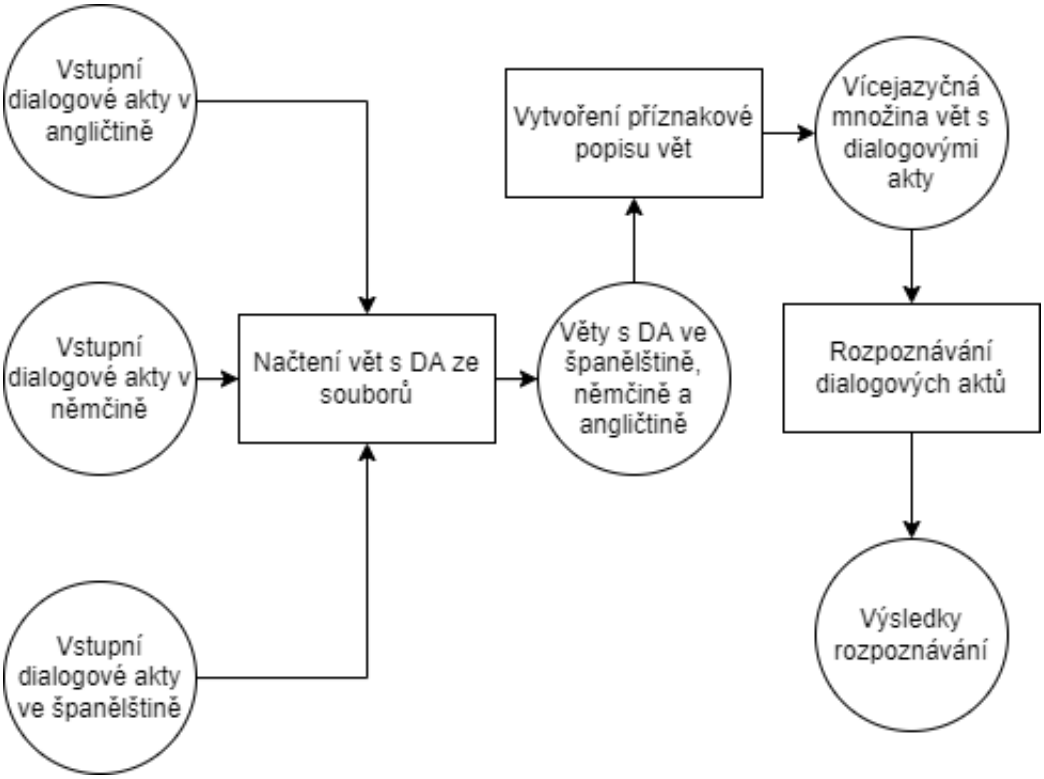
počet	trénovací množina	testovací množina
Dialog	15 513	622
Dialogové akty	32 269	1 460

Tabulka 4.7: Statistické informace o VM korpusu pro Německý jazyk.

V anotovaném korpusu se vyskytuje 16 tříd dialogových aktů: *FEEDBACK, GREET, INFORM, SUGGEST, INIT, CLOSE, REQUEST, DELIBERATE, BYE, COMMIT, THANK, POLITENESS_FORMULA, BACKCHANNEL, INTRODUCE, DEFER, OFFER* . Rozložení dialogových aktů v korpusu je nevyvážené, primárně se vyskytují čtyři dominantní DA (jmenovitě *feedback, suggest, inform, request*), které reprezentují téměř 80% korpusu.

5 Analýza problému

Jak již bylo uvedeno, cílem práce je navrhnout a implementovat prototyp systému pro rozpoznávání dialogových aktů ve více jazycích. Takovýto vícejazyčný systém umožní rozpoznávat dialogové akty z promluv ve více jazycích.
Nejdříve bylo nutné z DIHANA korpusu vytvořit datovou sadu a slovník. Datové sady pro jednotlivé jazyky bylo potřeba sloučit dohromady pro vytvoření vícejazyčné datové sady. Stejná operace byla provedena pro jednotlivé slovníky, které slouží pro příznakový popis vět. Blokové schéma navrženého systému je znázorněno na obrázku 5.1.



Obrázek 5.1: Diagram prototypu systému.

- Vstupní dialogové akty v angličtině (němčine, španělštině) - datový soubor formátu conll obsahující promluvy a dialogové akty.
- Načtení vět s DA ze souborů - zpracování conll souboru, tj. nahrání jednotlivých promluv a dialogových aktů.

- Vytvoření vektorové reprezentace vět - transformace promluv a dialogových aktů do číselné podoby.
- Rozpoznávání dialogových aktů - samotné rozpoznávání dialogových aktů ve více jazycích.

5.1 Parsování DIHANA korpusu

Jak bylo zmíněno v kapitole 4.1 o projektu DIHANA, projektu se za použití různých scénářů podařilo získat celkem 900 dialogů vedených mezi uživateli a systémem.
Jednotlivé dialogy jsou přepsané a ručně anotované v .dia souborech. Příklad jedné promluvy v dialogu je uveden v tabulce 5.1.

U1: hola mira que quería ir a Málaga desde aquí desde Bilbao y queria saber los horarios de los trenes.
hola mira que queria ir a Málaga (U:Pregunta:Hora_salida:Destino) pal: 1-7 desde aquí desde Bilbao y quería saber los horaris de los trenes. (U:Pregunta:Hora_salida:Origen) pal: 8-20

Tabulka 5.1: Ukázka uložení dat v .dia souboru

V tabulce je zobrazen formát anotace. Nejdříve je zaznamenán mluvčí (U) a následně je promluva očíslována (1). Po dvojtečce následuje samotná promluva. Promluva je následně rozdělena na jednotlivé věty, které jsou již anotovány příslušnými dialogovými akty.
Formát této anotace je možné popsat pomocí regulárního výrazu a věty společně s dialogovými akty z korpusu extrahovat.

5.2 Uložení textu do .conll souborů a vytvoření slovníku

Extrahovaná data z DIHANA korpusu jsem následně uložil do .conll souboru. Rozhodl jsem se data uložit v tomto formátu hlavně z důvodu konzistence, protože VM korpus pro anglický i německý jazyk je uložen právě v tomto formátu souboru.
Soubor .conll je používám pro reprezentaci korpusu jedním slovem na řádek a tabulátorem jsou odděleny informace o slovu, viz reprezentace věty *Bienvenido al servicio de información de trenes.* v tabulce 5.2.

Již bylo zmíněno, že slovník je použit k příznakovému popisu vět, je proto nutné jej vytvořit. Slovník ukládá četnost jednotlivých slov, ukázka souboru je zobrazeno v tabulce 5.3.

Slovo	**DA**	**Jazyk**
Bienvenido	Apertura	ESP
al	Apertura	ESP
servicio	Apertura	ESP
de	Apertura	ESP
información	Apertura	ESP
de	Apertura	ESP
trenes	Apertura	ESP
.	Apertura	ESP

Tabulka 5.2: Reprezentace věty v CONLL souboru.

Četnost	**Slovo**
1800	Bienvenido
1890	al
3648	servicio
32306	de

Tabulka 5.3: Ukázka části slovníku pro španělský jazyk.

5.3 Sjednocení datových sad pro vícejazyčnost

Po úspěšném vytvoření slovníku a .conll souboru pro španělský jazyk je nutné sjednotit všechny tři jazyky do jedné vícejazyčné množiny. Touto množinou bude model učen a evaluován.
Anglický a německý jazyk je snadné sloučit, oba jazyky totiž sdílí totožné třídy dialogových aktů. Španělský jazyk má třídy odlišné. Tento problém jsem vyřešil mapováním tříd dialogových aktů DIHANA korpusu na třídy dialogových aktů Verbmobil korpusu.

5.3.1 DIHANA-Verbmobil mapování dialogových aktů

V této sekci je popsán mapovací proces, který byl nezbytný pro vytvoření vícejazyčného modelu. Mapování tříd bylo provedeno na základě analýzy obsahu promluv příslušících k jednotlivým dialogovým aktům. Tento proces proběhl úspěšně, podařilo se namapovat všechny třídy dialogových aktů z DIHANY na Verbmobil, tudíž nedojde k žádné ztrátě informace. Kompletní mapování je zobrazeno v tabulce 5.4. Mapováním došlo k redukci tříd dialogových aktů pro španělský jazyk z 11 na 6.

DH DA	VM DA	DH příklad	VM příklad
Respuesta	FEEDBACK	Hay varios tipos de trenes. Pasado mañana	okay that works for me that is okay
Espera	FEEDBACK	Un momento por favor.	
Afirmation	FEEDBACK	Sí.	
Negacion	FEEDBACK	No.	
Pregunta	REQUEST	hola mira que quería ir a Málaga desde aquí desde Bilbao y quería saber los horarios de los trenes .	okay was it good What are the options
Nueva_consulta	REQUEST	Desea algo más? Desea cambiar alguna de las características solicitadas?	
No_entendido	REQUEST	Puede repetir su consulta?	
Apertura	INTRODUCE	¿ En qué puedo ayudarle? Bienvenido al servicio de información de trenes	This is Barb My name is Kieveta
Indefinida	INFORM	Quiero saber. Hola, quería un billete de tren para ir a alguna ciudad que celebre carnavales.	the fun thing and then the six hour n yeah so we will get there at midnight
Confirmacion	OFFER	¿ Quiere viajar a gijón? ¿ Desea salir el martes, día 1 de junio de 2004?.	so shall I just call and see what they have then shall I make the call for the reservation
Cierre	CLOSE	No, gracias. Gracias por utilizar este servicio.	alright then so that is settled

Tabulka 5.4: Mapování DIHANY na VM.

5.4 Vektorová reprezentace slov

Při úlohách strojového zpracování přirozeného jazyka je důležité převést text, se kterým je operováno, do prezentace, která vyhovuje i výpočetním zařízením, tzn. konvertovat text do číselné podoby.
K řešeního tohoto úkolu existuje mnoho algoritmů, například Bag-of-words (BoW), one-hot vektor či tzv. vnoření slov (angl. word embedding).
V rámci této práce byla využita reprezentace pomocí one-hot vektoru, word embedding přístupu a Bag-of-words algoritmu. Metody popíši
v následujícím textu. Vytvoření vektorové reprezentace není nutné pro model ESEC, jeho primárním úkolem totiž je vytvářet word embedding.
V následujícím textu popíši jednotlivé algoritmy pro vytváření vektorové reprezentace a jejich aplikaci v této bakalářské práci.

5.4.1 One-hot vektor

Jedná se o jednu z nejjednodušších forem reprezentace slov. One-hot vektor reprezentuje jedno slovo jediným vektorem dimenze N, který obsahuje $N-1$ prvků s hodnotou 0 a jeden prvek s hodnotou jedna (tzv. hot pozice).

Tato metoda je v této práci použita pro reprezentaci dialogových aktů. Unikátních tříd dialogových aktů pro vícejazyčnou datovou sadu je celkem 16, one-hot vektor bude mít 16 prvků. Jednotlivé prvky vektorů odpovídají třídám zmíněných v sekci 4.2.6. Například dialogový akt: FEEDBACK bude reprezentován vektorem:
$[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$.

5.4.2 Bag-of-words

Bag-of-words slouží pro reprezentaci věty pomocí vektoru, který je naplněn četností výskytu slova ve větě. Napříkald pro větu:
Já jsem to neudělal, já jsem byl doma.
bude vzniklá vektorová reprezentace ve následující: $[2, 2, 1, 1, 1, 1]$. K vytvoření takové reprezentace je nutné mít slovník, který obsahuje četnost každého slova, které se v textu může vyskytnout.

5.4.3 Word embedding

Zmíněná One-hot vektor reprezentace používá řídké vektory (angl. sparse vector), které obsahují málo nenulových prvků. Zároveň tyto vektory postrádají informaci o vztazích mezi slovy, například sémantickou závislost.

Například slova želva, křeček, morče označují domácí zvířata, tento fakt ale one-hot vektor reprezentace neumí zachytit.
Použité word embedding vektory reprezentují slova jako multidimenzionální vektory s čísly. Výhodou těchto vektorů je fakt, že sémanticky podobná slova jsou v N-dimenzionálním prostoru umístěna blízko sebe.
V této práci byla využita Word2Vec (w2v) technika [21], která generuje vektorovou reprezentaci slov za použití dvouvrstvých neuronových sítí. Ke španělskému, anglickému i německému jazyku mi byly vedoucím práce dodány w2v soubory.

5.4.4 Vektorová reprezentace vět pro použité modely

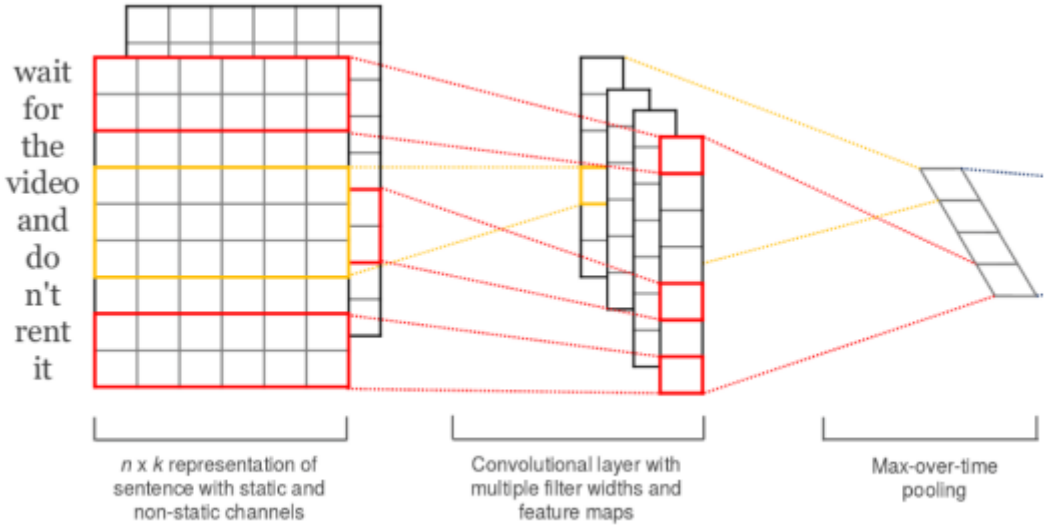
Jak již bylo zmíněno, model ESEC sám vytváří word embeddingu. Model proto nepotřebuje na vstupu slovník ani w2v vektory. Model používá preprocesor, který konvertuje věty v textové podobě do formátu vhodného pro enkodér, viz původní článek [11].
Pro ostatní modely je situace složitější, ke každému jazyku modely potřebují slovník. Bez něj není možné vytvořit vektorovou reprezentaci vět. Word2Vec vektory nejsou nutné pro funkčnost modelu, ale jejich využití by mělo vést k lepším výsledkům rozpoznávání.

5.5 Rozpoznávání dialogových aktů

V rámci bakalářské práce jsou použity dvě konvoluční neuronové sítě, jedna síť typu BILSTM a enkodér (ESEC), který je postaven nad BERT [11] transformer architekturou. Popis jejich architektury následuje v dalším textu.

5.5.1 $\mathbf{CNN}_1$

Tato konvoluční síť vychází z modelu, který byl navržen v práci [20]. Síť používá tři různé velikosti konvolučních jader - $3 \times EMB$, $4 \times EMB$, $5 \times EMB$, kde EMB představuje velikost slovního vektoru (angl. embedding vector). 100 konvolučních jader každé velikosti se počítá simuntálně a jejich výstup je sloučen a předán plně propojené vrstvě, která obsahuje 256 neuronů. Počet neuronů ve výstupní vrstvě odpovídá počtu klasifikovaných tříd. Jako chybová funkce se využívá Categorical Cross-Entropy a aktivační funkce výstupní vrsty je funkce softmax. Architektura sítě je zobrazena obrázku 5.2.

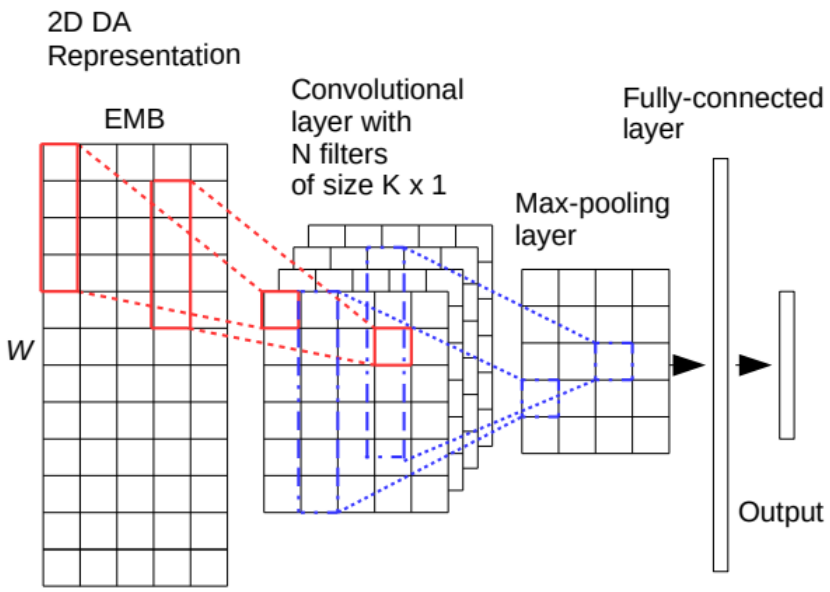


Obrázek 5.2: Architektura CNN_1 [20].

5.5.2 CNN_2

Tato konvoluční síť byla původně používána ke klasifikaci dokumentů. Konvoluční jádro bylo upraveno, aby se síť přizpůsobila úloze rozpoznávání dialogových aktů oproti původní úloze klasifikace dokumentů. Na dialogové akty lze nahlížet jako na velmi krátké dokumenty. Na obrázku 5.3 je vyzobrazena architektura této sítě.

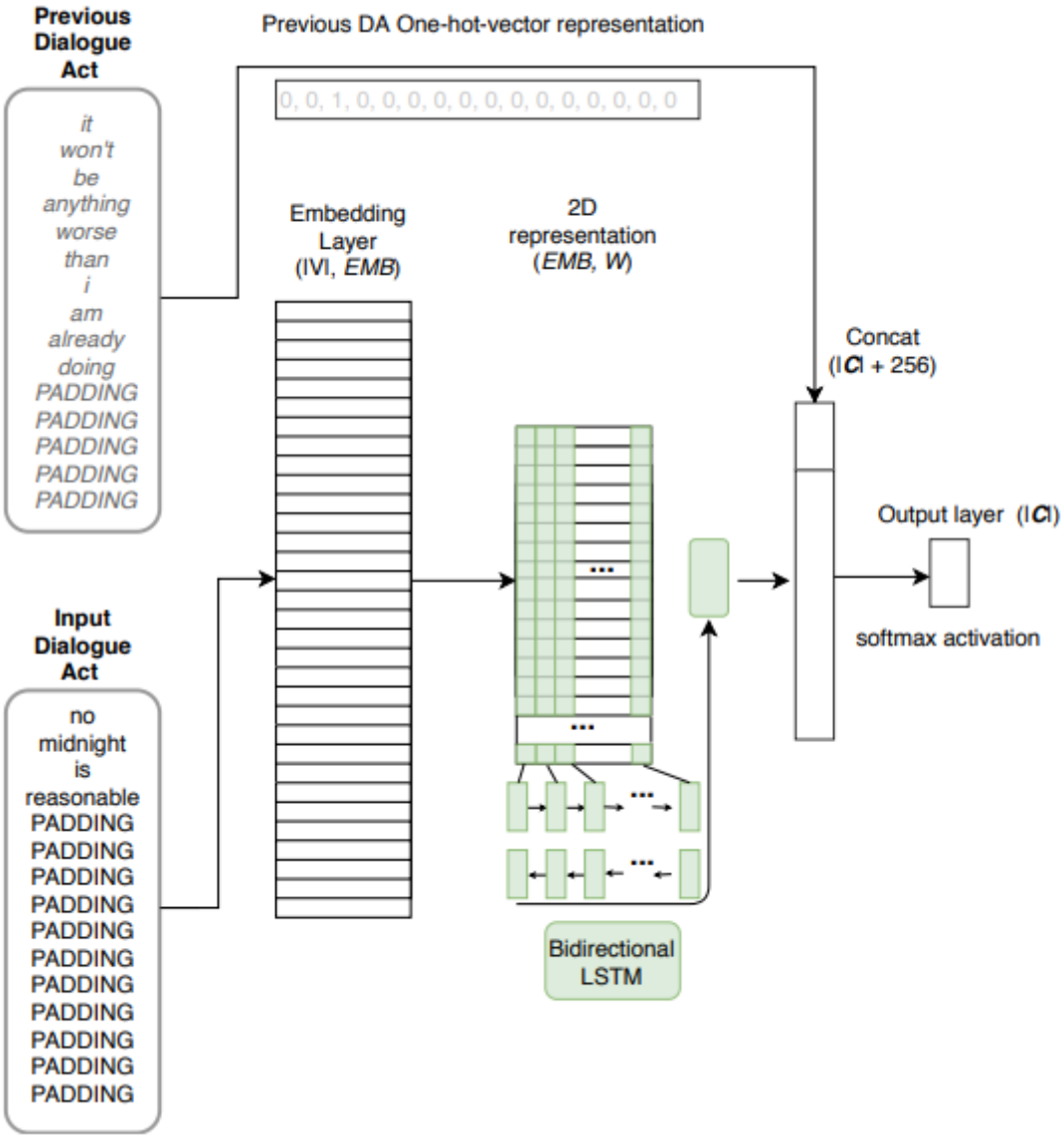
V síti se využívá 40 konvolučních jader s velikostí (4×1) s *relu* aktivační funkcí. Po konvoluční vrstvě následuje pooling vrstva,popis vrstev viz sekce 3.3.2. Výstup této vrstvy je předán plně propojené vrstvě jako v případě CNN_1.



Obrázek 5.3: Architektura sítě CNN_2 [20].

5.5.3 BILSTM

Tento model využívá obousměrnou LSTM síť. Vstupní a embedding vrstva jsou stejné jako u zmíněných CNN. Jádrem tohoto modelu je obousměrná LSTM vrstva, jejíž výstupní vektor se spojí s reprezentací předchozího dialogového aktu (one hot vector). Výstupní vrstva je stejná jako u výše zmíněných CNN. Na obrázku 5.4 je zobrazena architektura sítě.



Obrázek 5.4: Architektura BILSTM sítě [20].

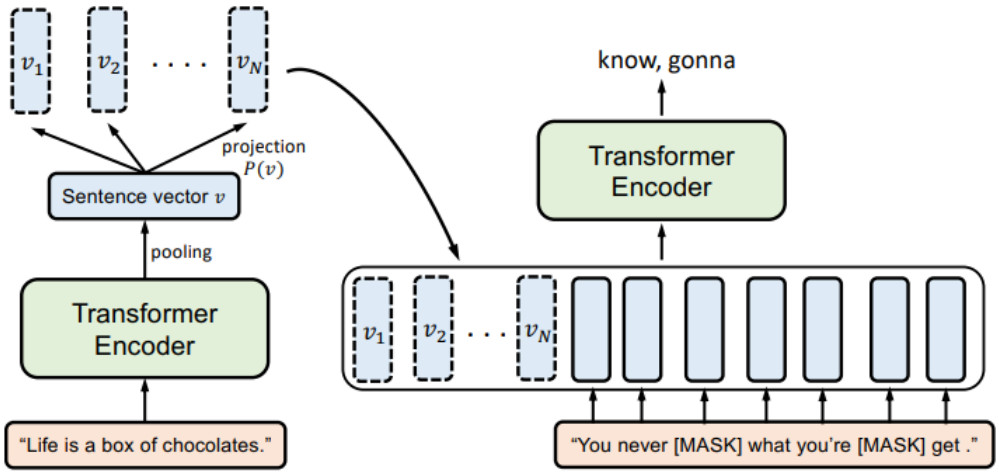
5.5.4 ESEC

Universal Sentence Encoder CMLM (ESEC) je model, který používá novou metodu trénování neuronových sítí - Conditional Masked Language Modeling (CMLM) [11] k efektivnímu učení reprezentace vět. Masked language modeling (MLM) je tzv. fill-in-the-blank task (vyplň prázdné políčko). Vstupem do modelu je neúplná věta (některé slovo je skryté) a očekávaným výstupem je věta plně doplněná. Příkladem věty na vstupu může být: "Včera jsem se díval v <maska> na fotbal". Očekáváným výstupem bude

věta "Včera jsem se díval v televizi na fotbal". Pokud je slov zakryto více, každá maska je nahrazena slovem s nejvyšší pravděpodobností výskytu. Model se skládá z enkodéru a dekodéru. Enkodér vytvoří vektorovou reprezentaci věty a za masku doplní vhodné slovo podle okolních slov. Dekóder následně vektorovou reprezentaci konvertuje zpátky na větu. CMLM rozšiřuje schopnost obyčejného MLM o zakódování informací o sousedních větách do embedding vektoru.

Při rozpoznávání DA stačí použít enkodér pro vytvoření vektorové reprezentace věty. Použití toho modelu společně s historií by mělo vést ke zlepšení výsledků, jelikož vygenerovaný vektor příznaků obsahuje více informací o větě než w2vec vektor (ESEC vektor obsahuje 768 prvků, použitý w2v pouze 300. Sémantická informace by tedy měla být lepší). Model je tvořen preprocesorem a enkoderém, za enkodérem následuje vrstva výstupní. Model rozšiřuje schopnosti modelu BERT [11].

ESEC model oproti ostatním nevyužívá na vstupu word embedding vrstvu, sám totiž word embedding vytváří. Architektura je zobrazena na obrázku 5.5.



Obrázek 5.5: Architektura esec modelu [30].

6 Implementace

V následující kapitole popíši postup řešení jednotlivých kroků. Nejprve popíši vytváření conll souborů z DIHANA korpusu. Následně vysvětlím postup zpracování korpusu do .conll souborů. Poté popíši vytvoření vektorového popisu vět a vytvoření klasifikátorů.
Klasifikátory byly vytvořeny jednojazyčné i vícejazyčné pro porovnání úspěšnosti rozpoznávání.

6.1 Parsování DIHANA korpusu

První částí samotné realizace bylo parsování DIHANA corpus a následné vytvoření .conll souborů a slovníku. Dialogy s anotovanými dialogovými akty jsou uloženy v .dia souborech.
Formát anotace je následující:
<věta> (<mluvčí>:<DA>:<specifické_žádosti>) pal: <délka_věty>

- <věta> - promluva uživatele či systému v dialogu, například: hola mira que queria ir a Málaga.
- <mluvčí> - označení, zda mluví uživatel (U) či stroj (M).
- <DA> - dialogový akt příslušné věty, například: Pregunta.
- <specifické_žádosti> - viz kapitola 4.1.2 o korpusu.
- <délka_věty> - počet slov dané věty, například: 1-7.

Tato anotace obsahuje řadu informací, z nichž pro vytvoření .conll souborů je pouze věta a dialogový akt.
Pro parsování .dia souborů jsem použil programovací jazyk python a s pomocí regulárního výrazu ve tvaru: `(.*)(\((M|U)\:\w+\:.*\))` jsem získal jednotlivé věty s příslušnými dialogovými akty.

6.2 Vytvoření .conll souborů a slovníku

Po úspěšném parsování korpusu je nutné věty s dialogovými akty uložit pro trénování neuronových sítí.
Rozhodl jsem se data uložit v .conll souboru hlavně z důvodu konzistence. Data z VM korpusu pro anglický i německý jazyk jsou uložena právě v tomto

formátu.
V .conll souboru je každé slovo uloženo na samostatném řádku a tabulátorem je oddělena informace o slovu (viz reprezentace věty *Bienvenido al servicio de información de trenes.* v tabulce 6.2). Tento formát používá dva oddělovače: ';' a '\n'. Středník slouží pro oddělení dialogů, nový řádek slouží pro oddělení promluv rámci stejného dialogu.

Slovo	**DA**	**Jazyk**
Bienvenido	Apertura	ESP
al	Apertura	ESP
servicio	Apertura	ESP
de	Apertura	ESP
información	Apertura	ESP
de	Apertura	ESP
trenes	Apertura	ESP
.	Apertura	ESP

Tabulka 6.1: Reprezentace věty v CONLL souboru.

Slovník reprezentuje čenost unikátních slov v korpusu. Slovník je používán u modelů CNN_1, CNN_2 a BI-LSTM pro vytvoření vektorového popisu vět. Tabulka 6.2 zobrazuje formát slovníku pro první tři slova ve španělském jazyce. Formát je stejný pro anglický i německý slovník.

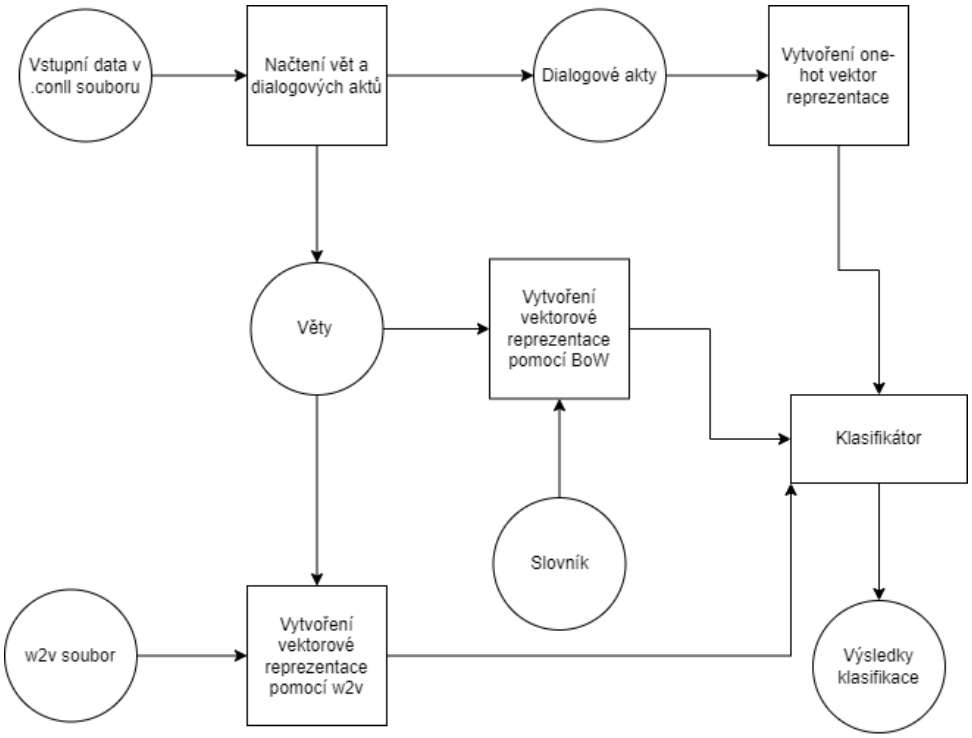
Četnost	**Slovo**
1800	bienvenido
1890	al
3648	servicio

Tabulka 6.2: Ukázka slovníku pro španělský jazyk.

6.3 Předzpracování dat pro neuronové sítě

Po úspěšném vytvoření .conll souborů z DIHANA korpusu je nutné data v tomto formátu přezpracovat pro použití jednotlivých modelů. Fáze předzpracování je zásadně odlišná pro modely CNN_1, CNN_2, BILSTM a pro model ESEC, jak již bylo zmíněno v sekci 5.4.4. Tato sekce je zaměřena na předzpracování dat pro první tři zmíněné modely.
Po načtení .conll souboru, který původní věty rozdělil na jednotlivá slova,

dojde k sestavení původních vět a vytvoření one-hot vector reprezentace dialogových aktů ke každé větě.
Pro vytvoření vektorové reprezentace se každé slovo ve větě nahradí příslušným indexem v seřazeném slovníku. Dále se načte samotný w2v soubor pro daný jazyk. Následně se prochází všechna dostupná slova v načteném vektoru, pokud slovo existuje ve slovníku daného jazyka, je jeho w2v reprezentace uložena do embedding matice.
One hot vektory mají vždy délku 16, tato délka reprezentuje počet tříd dialogových aktů. Vektorová reprezentace vět pomocí BoW má vždy délku 15. Může nastat situace, kdy věta neobsahuje přesně patnáct slov. Pokud věta obsahuje slov méně, je provedeno zarovnání (angl. padding). Zarovnání je vidět na obrázku 5.4. Pokud věta obsahuje více slov než 15, je do vektoru uloženo prvních 14 slov a slovo poslední. Na obrázku 6.1 je zobrazen celý proces rozpoznávání dialogových aktů, včetně vytváření vektorové reprezentace vět.



Obrázek 6.1: Proces rozpoznávání dialogových aktů.

6.4 Implementace modelů neuronových sítí

Princip vytváření modelů je stejný pro jednojazyčné i vícejazyčné modely. U vícejazyčných modelů se zvýší výpočetní složitost fáze předzpracování dat, je totiž nutné načítat slovníky i w2v vektory pro všechny jazyky. Oproti tomu k vytvoření jednojazyčných modelů stačí načíst w2v vektor a slovník pouze k danému jazyku, například angličtině.
Nejdříve jsem implementoval jednojazyčné modely, abych získal přehled o úspěšnosti klasifikace pro jednotlivé jazyky.
Samotné modely jsem implementoval v jazyce Python, konkrétně verze 3.7 s využitím knihovny *Tensorflow*. Tato knihovna umožňuje přístup ke Keras API, které obsahuje již naprogramované vrstvy (například konvoluční) a je možné jim pouze upravit hyperparametry. Pro modely CNN_1, CNN_2, BILSTM jsem použil nastavení hyperparametrů dle [20].
Model ESEC jsem stáhl již naučený a veškeré hyperparametry jsem ponechal původní z práce [30].

6.4.1 Vytvoření vícejazyčných modelů

V rámci této práce jsou použity tři jazyky: angličtina, němčina a španělština. K vytvoření vícejazyčného modelu CNN_1,CNN_2 nebo BILSTM je nutné mít slovník. Pro rozšíření modelu o španělský jazyk bylo nutné provést mapování tříd DA korpusu DIHANA na třídy DA korpusu VM, viz sekce 5.3. Vícejazyčný model umožňuje rozpoznávání dialogových aktů pro tři výše zmíněné jazyky. Pro vytvoření vícejazyčného modelu je nutné pozměnit fázi předzpracování.
Fáze předzpracování pro vícejazyčný model probíhá následovně. Načtou se datové (.conll) soubory pro každý jazyk. Následně se z těchto souborů načtou dialogové akty a příslušné věty. K vytvoření BoW vektorové reprezentace je nutné vytvořit jeden velký slovník sloučením třech slovníků k jednotlivým jazykům.
Samotný proces vytvoření reprezentace zůstává stejný jako pro jednojazyčný model. K načtení w2v embeddingu je nutné načíst všechny soubory s vektory a sloučit je do jednoho velkého souboru. Proces vytváření vektorové reprezentace je potom stejný jako u jednojazyčného modelu.
Pro rozšíření některého z těchto modelů o další jazyk je nutné dodat výše požadavané soubory. Model ESEC ale používá vlastní preprocesor a vytváří vlastní word embedding. Modelu stačí dodat one-hot vektor reprezentaci dialogových aktů a textovou podobu jednotlivých výroků, které k nim přísluší.

Pro mé vícejazyčné experimenty jsem jako klasifikátor zvolil model CNN_1 a model ESEC.
Model CNN_1 jsem v této práci použil, protože jej v podobné práci [20] využil Martínek, Král, Lenc k rozpoznávání dialogových aktů. Model ESEC vytváří vícejazyčný word embedding, proto jsem jej využil.

7 Experimenty

V následujícím textu jsou představeny metriky použité pro evaluaci modelů. Dále je zde popsán učící proces modelů a jsou zde prezentovány dosažené výsledky rozpoznávání dialogových aktů.

7.1 Evaluační metody

V rámci této práce jsou použity následující evaluační metriky. Proto si je blíže popíšeme v dalším textu.

7.1.1 Matice záměn

První použíta metrika je matice záměn (confusion matrix). Tato metrika poskytuje pohled nejenom na úspěšnost modelu, ale také uchovává statistiky o jednotlivých třídách, které model klasifikuje nejlépe či nejhůře a k jakým chybám nejčastěji dochází [7].
V tabulce 7.1 příklad matice záměn pro dvě třídy: Pozitivní a Negativní.

Třída	**Pozitivní predikce**	**Negativní predikce**
Pozitivní	True positive (TP)	False negative (FN)
Negativní	False positive (FP)	True negative (TN)

Tabulka 7.1: Matice záměn pro třídy A, B.

- True positive (TP) - správně odhadnuté pozitivní příklady
- False negative (FN) - špatně odhadnuté pozitivní příklady
- False positive (FP) - špatně odhadnuté pozitivní příklady
- True negative (TN) - správně odhadnuté negativní příklady

7.1.2 F-míra

Další metrika je F-míra (F-score), je definována jako vážený harmonický průměr přesnosti a úplnosti [24].

$$recall = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{7.1}$$

Úplnost (recall) je definována jako poměr korektně určených pozitivních hodnot, ke všem pozitivním hodnotám [24]. Úplnost je dána vztahem:

$$recall = \frac{TP}{TP + FN} \tag{7.2}$$

Přesnost (precision), je definována jako poměr korektně určených pozitivních hodnot ke všem klasifikací získaným hodnotám [24]. Přesnost je dána vztahem:

$$precision = \frac{TP}{TP + FP} \tag{7.3}$$

7.1.3 Přesnost (accuracy)

Poslední použitá metrika je přesnost (accuracy). Tato metrika se obecně počítá jako počet správně predikovaných výsledků ku počtu všech výsledků, tedy podle klasické definice pravděpodobnosti:

$$ACC = \frac{OK}{ALL} \tag{7.4}$$

.

- OK - množina všech správných predikcí.
- ALL - množina všech predikcí.

7.2 Učení modelů

Tato kapitola popisuje proces učení vícejazyčných modelů a je zde zmíněno učení modelů pro anglický dataset. Pro německý a španělský jazyk je tento průběh učení obdobný, nejsou proto dále rozvedeny. U všech modelů byla využita vstupní vrstva reprezentující historii a validační množina.
V následujícím textu popíši vliv historie při učení, důvod užití validační množiny a samotný proces učení modelů. V rámci učení modelů bylo vyzkoušeno i tzv. transfer learning. Tato metoda nevedla k jakémukoliv zlepšení modelů, byla proto dále zavržena.
Proces učení je velmi podobný u modelů CNN_1,CNN_2 a BILSTM, proto jsem zvolil model CNN_1 jako reprezentanta této skupiny a popsal jej. Následně je tento proces popsán pro model ESEC.

7.2.1 Vliv historie při učení modelů

Historie dodává modelu kontext o aktuální větě na vstupu. Vstupem této vrstvy je one-hot vector, ve kterém je uložena informace o třídě věty předcházející, tedy příslušný DA (pro první větu dialogu je tento vektor nulový). Implementace sekundární vstupní vrstvy by měla zlepšit úspěšnost klasifikace, protože po větě, ke které náleží DA například 'OTÁZKA' s vysokou pravděpodobností bude následovat věta, jejíž DA bude 'ODPOVĚĎ' a tento vzor se modely naučí.

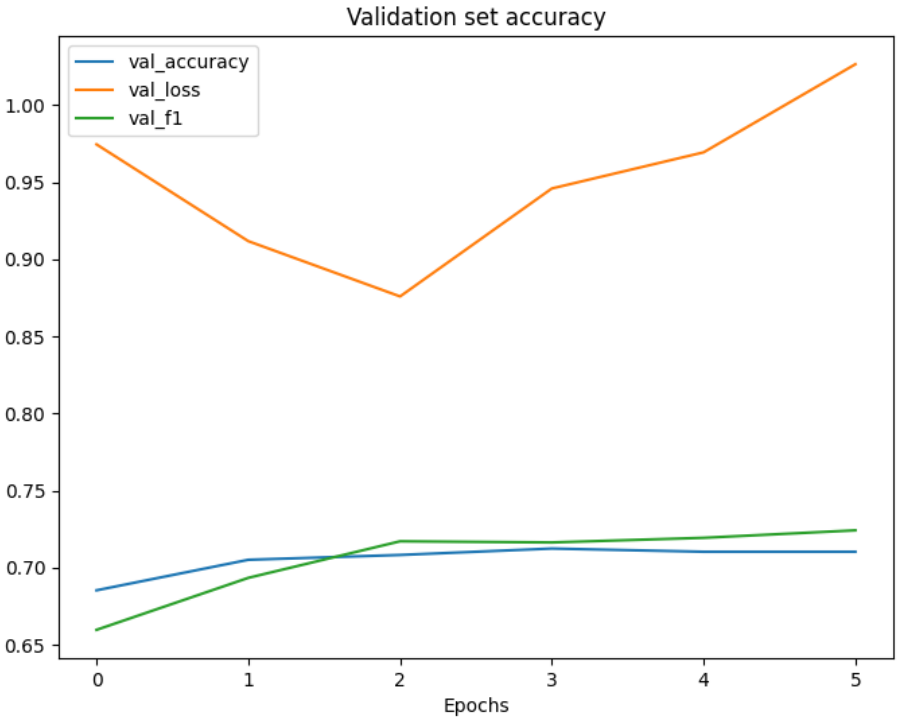
7.2.2 Validační množina

Validační množina slouží pro validaci modelu ve fázi učení. Množinu používám kvůli mechanismu early stopback, který je schopen najít trénovací optimum. Díky tomuto mechanismu nedojde k přeučení modelu. Jediným parametrem tohoto mechanismu je tzv. patience - tento parametr udává po jaké době se zastaví trénování, pokud se zvyšuje chybovost modelu. V mých experimentech jsem použil hodnotu parametru = 3.
V rámci této práce validační množina, obsahuje 10% dat z trénovací množiny.

7.2.3 CNN_1

Na obrázku 7.1 je vizualizována úspěšnost klasifikace modelu pro data z validační množiny. Validační množina obsahuje data z anglického jazyka. Proces učení je stejný u ostatních jazyků jako u angličtiny, proto tento proces popíši pouze pro tento jazyk a pro vícejazyčnou variantu modelu.

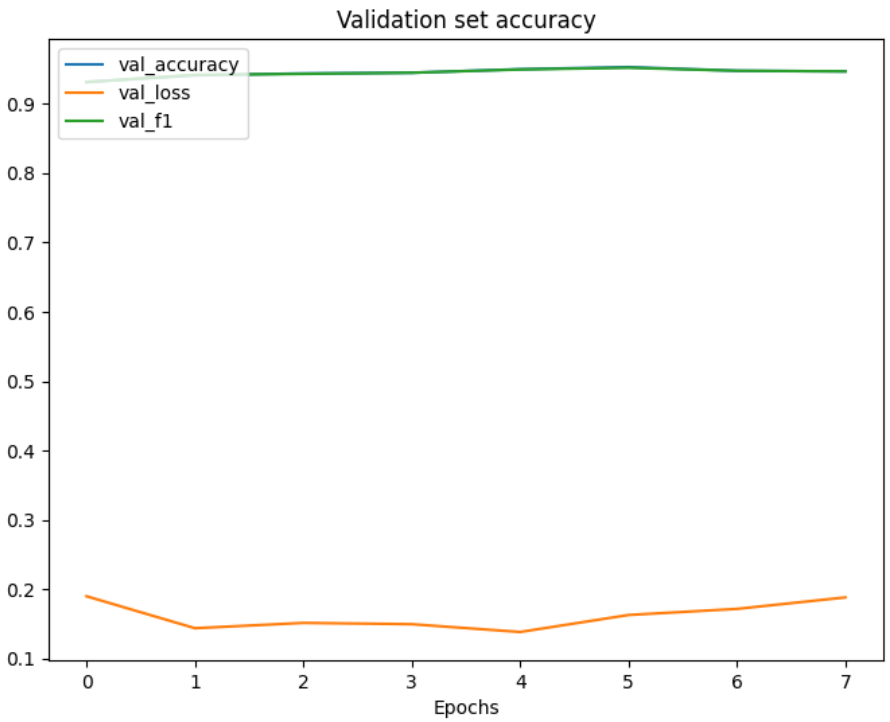
Obrázek znázorňuje, že je model natrénován již po 2. epoše. V dalších epochách nastane zvýšení loss hodnoty, která signalizuje zhoršení přesnosti klasifikace modelu. Trénovací optimum je v 2. epoše, váhy neuronů se obnoví na hodnoty právě z této epochy.



Obrázek 7.1: Vývoj úspěšnosti klasifikace DA po epochách pro data z validační množiny anglického jazyku.

7.2.4 CNN_1 vícejazyčné učení

Na obrázku 7.2 je vizualizována úspěšnost klasifikace modelu pro data z validační množiny. Validační množina obsahuje data ze všech dostupných datových sad.
Obrázek znázorňuje, že je model natrénován již ve 4. epoše. Přenost klasifikace DA z validační množiny je zde zavádějící, kvůli nevyváženosti datových sad. Validační množina z větší části obsahuje data ze španělské a německé datové sady.

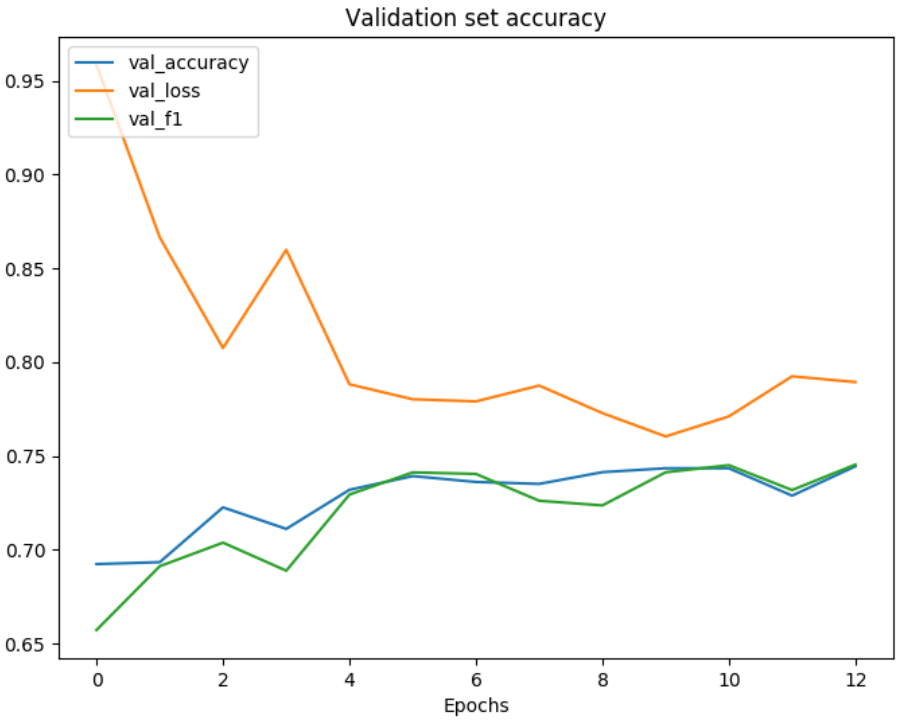


Obrázek 7.2: Vývoj úspěšnosti klasifikace DA po epochách pro data z vícejazyčné validační množiny.

7.2.5 ESEC

Na obrázku 7.3 je vizualizována úspěšnost klasifikace modelu pro data z validační množiny. Validační množina obsahuje data z anglického jazyka. Proces učení je stejný u ostatních jazyků jako u angličtiny, proto tento proces popíši pouze pro tento jazyk a pro vícejazyčnou variantu modelu.

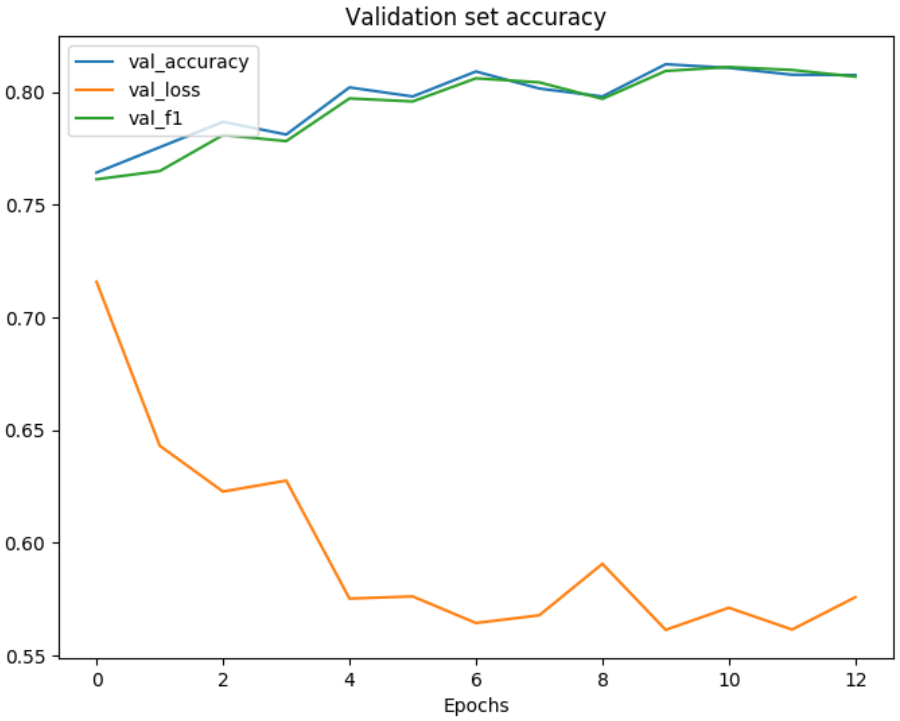
Obrázek znázorňuje, že je model natrénován v 9. epoše. V dalších epochách nastane zvýšení loss hodnoty, která signalizuje zhoršení přesnosti klasifikace modelu.



Obrázek 7.3: Vývoj úspěšnosti klasifikace DA po epochách pro data z validační množiny anglického jazyku.

7.2.6 ESEC vícejazyčné učení

Na obrázku 7.4 je vizualizována úspěšnost klasifikace modelu pro data z validační množiny. Validační množina obsahuje data ze všech dostupných datových sad.
Obrázek znázorňuje, že je model natrénován v 9. epoše. V následujích epochách nenastane snížení chybovosti modelu, proces trénování je zastaven tzv. early stop mechanismem.



Obrázek 7.4: Průběh učení nad validační množinou pro angličtinu, španělštinu a němčinu

7.3 Výsledky klasifikace DA na DIHANA korpusu

Tabulka 7.2 zobrazuje výsledky evaluace natrénovaných modelů na DIHANA korpusu.

Model	**w2v**	**Historie**	**ACC**	**Fmíra**	**Precision**	**Recall**
CNN_1	ano	ano	94,79	93,04	94,55	94,38
CNN_1	ne	ano	91,79	91,61	91,30	91,69
CNN_1	ano	ne	92,98	92,98	93,21	93,21
CNN_1	ne	ne	92,40	92,39	92,52	92,40
CNN_2	ano	ano	94,45	94,57	94,57	94,44
CNN_2	ne	ano	94,83	94,64	94,65	94,64
CNN_2	ano	ne	93,53	93,53	93,54	93,58
CNN_2	ne	ne	92,62	92,35	92,35	92,56
BILSTM	ano	ano	**95,43**	**95,57**	95,75	95,78
BILSTM	ne	ano	94,15	93,98	93,89	93,81
BILSTM	ano	ne	94,10	94,40	94,12	93,81
BILSTM	ne	ne	92,15	92,68	94,12	94,09
ESEC	-	ano	**95,66**	**95,56**	95,58	95,46
ESEC	-	ne	92	92	93,12	91,09

Tabulka 7.2: Úspěšnost klasifikace DA pro španělský jazyk [v %]; model ESEC vytváří embedding vrstvu interně.

Modely ESEC a BILSTM dosahují nejlepších výsledků. Z tabulky je vidět, že historie dialogových aktů má pozitivní vliv při rozpoznávání DA. Využití w2v embedding je pro klasifikaci DA také přínosné. Model ESEC si vytváří vlastní embedding a rozdíl mezi úspěšností BILSTM a tohoto modelu není statisticky signifikantní. Oba modely jsou navrženy pro práci s daty sekvenčního charakteru a jejich lepší výsledky oproti konvolučním sítím nejsou překvapivé.

7.4 Výsledky klasifikace DA na VM korpusu

V tabulce 7.3 jsou zobrazeny výsledky rozpoznávání dialogových aktů pro německý jazyk, v tabulce 7.4 jsou zobrazeny výsledky pro anglický jazyk.

Model	**w2v**	**Historie**	**ACC**	**Fmíra**	**Precision**	**Recall**
CNN_1	Ano	Ano	71,64	71,92	72,68	71,64
CNN_1	Ne	Ano	55,34	54,51	57,75	54,93
CNN_1	Ano	Ne	65,89	65,13	68,19	64,32
CNN_1	Ne	Ne	70,21	72,10	73,93	71,16
CNN_2	Ano	Ano	73,63	73,30	75,35	72,53
CNN_2	Ne	Ano	72,19	73,33	74,85	72,67
CNN_2	Ano	Ne	73,29	73,46	74,81	72,67
CNN_2	Ne	Ne	71,92	71,22	74,71	70,84
BILSTM	Ano	Ano	74,58	75,29	76,69	74,59
BILSTM	Ne	Ano	**75,27**	**76,03**	77,85	75,34
BILSTM	Ano	Ne	74,04	73,74	75,57	73,84
BILSTM	Ne	Ne	73,29	73,45	75,10	72,95
ESEC	-	Ano	72,01	71,05	74,03	71,04
ESEC	-	Ne	68,78	68,01	69,10	68,79

Tabulka 7.3: Úspěšnost klasifikace DA pro německý jazyk [v %]; model ESEC vytváří embedding vrstvu interně.

Datová sada pro německý jazyk je statisticky největší. I zde je vidět, že historie značně zlepšuje úspěšnost klasifikační úlohy. Nejvyšší úspěšnosti dosáhl model BILSTM.

Model	w2v	Historie	ACC	Fmíra	Precision	Recall
CNN_1	ano	ano	74,29	74,78	73,39	72,60
CNN_1	ne	ano	57,35	54,46	56,56	58,52
CNN_1	ano	ne	66,05	64,41	64,18	65,70
CNN_1	ne	ne	73,94	73,67	74,32	73,94
CNN_2	nno	ano	73,31	72,51	73,31	72,89
CNN_2	ne	ano	73,30	73,89	75,52	74,15
CNN_2	ano	ne	72,74	72,67	73,18	72,74
CNN_2	ne	ne	75,14	73,44	73,86	73,80
BILSTM	ano	ano	73,09	73,41	74,76	73,45
BILSTM	ne	ano	73,29	73,70	75,10	74,64
BILSTM	ano	ne	73,02	72,73	73,68	72,84
BILSTM	ne	ne	73,29	73,45	75,10	72,75
ESEC	-	ano	**75,01**	**76,05**	81,03	71,04
ESEC	-	ne	72,51	72,04	72,67	72,51

Tabulka 7.4: Úspěšnost klasifikace DA pro anglický jazyk [v %]; model ESEC vytváří embedding vrstvu interně.

Datová sada pro anglický jazyk je statisticky nejmenší. Nejlepších výsledků dosahuje model ESEC. Je zde vidět stejný vzor jako u předešlých tabulek.

7.5 Výsledky vícejazyčné klasifikace

Vícejazyčný model umí rozpoznávat dialogové akty v angličtině, němčine a španělštině. Model CNN_1 i ESEC byl naučen na sjednocené množině datových sad.
Pro oba modely byla jako vstup uvažována i historie dialogu. Pro model CNN_1 jsou slova na vstupu mapována na w2v vektory. Model byl testován na sjednocené testové sadě a na sadách pro jednotlivé jazyky.
V tabulce 7.5 jsou zobrazeny výsledky rozpoznávání dialogových aktů dosažené vícejazyčnými modely CNN_1, CNN_2 a BILSTM. V tabulce 7.6 jsou zobrazeny výsledky pro vícejazyčný model ESEC.

7.5.1 Vícejazyčné modely CNN_1, CNN_2 a BILSTM

Model	**ACC**	**Fmíra**	**Precision**	**Recall**	**Test**
CNN_1	87,27	87,34	87,19	87,27	en, ge, esp
CNN_1	67,18	66,87	67,40	67,18	en
CNN_1	70	70,41	70,75	70,76	ge
CNN_1	94,38	94,39	94,62	94,38	esp
CNN_2	89,21	89,14	90,38	88,07	en, ge, esp
CNN_2	68,66	68,34	76,23	62,08	en
CNN_2	73,01	73,11	74,91	71,45	ge
CNN_2	94,62	94,61	94,73	94,62	esp
BILSTM	88,95	88,75	89,30	88,97	en, ge, esp
BILSTM	68,80	68,19	76,31	61,82	en
BILSTM	72,53	72,89	76,98	70,10	ge
BILSTM	95,38	95,39	95,50	95,38	esp

Tabulka 7.5: Úspěšnost klasifikace DA pro anglický, německý a španělský jazyk [v %].

7.5.2 Vícejazyčný model ESEC

Model	ACC	Fmíra	Precision	Recall	Test
ESEC	89,51	89,22	91,01	87,76	en, ge, esp
ESEC	75,63	74,99	75,99	75,63	en
ESEC	73,97	72,72	76,30	73,97	ge
ESEC	94,64	94,60	94,77	94,64	esp

Tabulka 7.6: Úspěšnost klasifikace DA pro anglický, německý a španělský jazyk [v %].

Vícejazyčný model ESEC dosahuje lepších výsledků než ostatní vícejazyčné modely. Konkrétně pro anglický jazyk je rozdíl cca 7%. Kontextový model ESEC má více parametrů, je proto schopen dosáhnout lepších výsledků, když je učen na objemnější sjednocené datové sadě.

8 Závěr

V rámci této bakalářské práce jsem se věnoval problematice rozpoznávání dialogových aktů ve více jazycích.
Při řešení této úlohy jsem se musel seznámit s korpusy Verbmobil a DIHANA. Díky této zkušenosti jsem získal představu o tom, co a k čemu dialogové akty jsou a jak se anotují. Pro rozpoznávání DA bylo potřeba pochopit strukturu dodaného DIHANA korpusu a převést jej do formátu conll. K vytvoření datové sady pro španělský jazyk bylo nutné parsovat DIHANA korpus. S využitím regulárních výrazů se mi podařilo z korpusu extrahovat jednotlivé promluvy s dialogovými akty a vytvořit slovník pro španělský jazyk.
Aby bylo možné model natrénovat na sjednocené datové sadě všech dostupných jazyků, tj. angličtina, němčina a španělština, bylo nutné provést mapování dialogových aktů mezi korpusy. K rozpoznávání dialogových aktů byly zvoleny čtyři klasifikační metody, dvě topologie konvoluční neuronové sítě, BILSTM síť a síť typu transfomer. Nejdříve bylo nutné zvolit vhodnou reprezentaci promluv pro použité neuronové sítě. Pro vektorovou reprezentaci pro modely CNN_1, CNN_2 a BILSTM byl použita technika Word2Vec. Model ESEC vytváří embedding vnitřně, nebylo tedy nutné použít tuto techiku i zde.
Z výsledku experimentů vyplynulo, že použití historie dialogových aktů a w2v embedding vrstvy má pozitivní vliv při rozpoznávání. Pro jednotlivé jazyky dosahovaly nejlepších výsledků modely ESEC a BILSTM. V rozpoznávání dialogových aktů ve více jazycích se ukázal jako nejlepší model ESEC. Na sjednocené datové sadě rozpoznávaly DA všechny modely s podobnou úspěšností, ale vícejazyčné modely CNN_1, CNN_2 a BILSTM rozpoznávaly DA z jednojazyčné datové sady hůře.
Možná rozšíření této práce můžou obsahovat zahrnutí více jazyků. Model ESEC se ukázal jako velmi vhodný pro úlohu rozpoznávání dialogových aktů ve více jazycích. Model si sám vytváří vícejazyčný embedding, proto je snadné jej rozšířit o libovolný jazyk.

9 Přehled zkratek

- DA - dialogový akt
- DH - DIHANA
- VM - Verbmobil
- en - English (angličtina)
- ge - German (němčina)
- esp - Spanish (španělština)
- CNN - Convolutional Neural Network (konvoluční neuronová síť)
- LSTM - Long Short Term Memory
- RNN - Recurent neural network (rekurentní neuronová síť)
- w2v - Word2Vec
- ESEC - Universal Sentence Ecoder CMLM
- MLM - Masked Language Model
- CMLM - Conditional Masked Language Model
- BERT - Bidirectional Encoder Representations from Transformers
- NLP - Natural Language Processing (zpracování přirozeného jazyka)
- WoZ - Wizard of Oz

Literatura

[1] Alexandersson, J. et al. Dialogue acts in VERBMOBIL-2. 1997.

[2] Arora, e. a. Dialogue system: A brief review. *arXiv preprint arXiv:1306.4134*. 2013.

[3] Austin, J. L. *How to do things with words*. Oxford university press, 1975.

[4] Baheti, P. The Essential Guide to Neural Network Architectures. Dostupné z: `https://www.v7labs.com/blog/neural-network-architectures-guide`.

[5] Brownlee, J. A gentle introduction to cross-entropy for machine learning. *Machine learning mastery*. 2019, 20. Dostupné z: `https://machinelearningmastery.com/cross-entropy-for-machine-learning/`.

[6] Brownlee, J. *An Introduction To Recurrent Neural Networks And The Math That Powers Them* [online]. Machinelearningmastery, 2021. [cit. 2021/12/16]. Dostupné z: `https://machinelearningmastery.com/an-introduction-to-recurrent-neural-networks-and-the-math-that-powers-them/`.

[7] Brownlee, J. *Failure of Classification Accuracy for Imbalanced Class Distributions* [online]. Machinelearningmastery, 2019. [cit. 2021/11/19]. Dostupné z: `https://machinelearningmastery.com/failure-of-accuracy-for-imbalanced-class-distributions/`.

[8] Brownlee, J. *Loss and Loss Functions for Training Deep Learning Neural Networks* [online]. Machinelearningmastery, 2019. [cit. 2021/11/19]. Dostupné z: `https://machinelearningmastery.com/loss-and-loss-functions-for-training-deep-learning-neural-networks/`.

[9] Brownlee, J. Understanding LSTM Networks, 2019. Dostupné z: `https://machinelearningmastery.com/an-introduction-to-recurrent-neural-networks-and-the-math-that-powers-them/`.

[10] Chollet, F. [online]. keras, 2020. [cit. 2022/04/21]. Dostupné z: `https://keras.io/guides/functional_api/`.

[11] Devlin, J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*. 2018.

[12] Hassan, H. et al. ASSESSMENT OF ARTIFICIAL NEURAL NETWORK FOR BATHYMETRY ESTIMATION USING HIGH RESOLUTION

SATELLITE IMAGERY IN SHALLOW LAKES: CASE STUDY EL BURULLUS LAKE. *International Water Technology Journal*. 12 2015, 5.

[13] Hochreiter, S. – Schmidhuber, J. Long short-term memory. *Neural computation*. 1997, 9, 8, s. 1735–1780.

[14] IBM. *Convolutional Neural Networks* [online]. IBM, 2020. [cit. 2022/04/24]. Dostupné z: `https://www.ibm.com/cloud/learn/convolutional-neural-networks`.

[15] Jekat, e. a. Dialogue acts in VERBMOBIL. 1995.

[16] Jeong, M. – Lee, G. G. *Jointly predicting dialog act and named entity for spoken language understanding*. 2006.

[17] Jose-Miguel Benedí, e. a. Design and acquisition of a telephone spontaneous speech dialogue corpus in Spanish: DIHANA. 05 2022.

[18] Kaushik, S. [online]. Analytics Vidhya, 2016. [cit. 2021/11/21]. Dostupné z: `https://www.analyticsvidhya.com/blog/2016/11/an-introduction-to-clustering-and-different-methods-of-clustering/`.

[19] LeCun, Y. – Bengio, Y. – Hinton, G. Deep learning. *nature*. 2015, 521, 7553, s. 436–444.

[20] Martínek, J. et al. Multi-lingual dialogue act recognition with deep learning methods. *arXiv preprint arXiv:1904.05606*. 2019.

[21] Mikolov, T. et al. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*. 2013, 26.

[22] O'Shea, K. – Nash, R. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*. 2015.

[23] Pavel, C. Hybrid Dialogue Management in Frame-Based Dialogue Systems Exploiting VoiceXML, 2007. Dostupné z: `https://theses.cz/id/xow705/`.

[24] Powers, D. M. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061*. 2020.

[25] Psutka, J. et al. *Mluvíme s počítačem česky*. Academia, 2006. Dostupné z: `http://www.kky.zcu.cz/en/publications/PsutkaJ_2006_Mluvimes`. ISBN 80-200-1309-1.

[26] Sainath, T. N. et al. Improvements to deep convolutional neural networks for LVCSR. In *2013 IEEE workshop on automatic speech recognition and understanding*, s. 315–320. IEEE, 2013.

[27] Vaswani, A. et al. Attention is all you need. *Advances in neural information processing systems*. 2017, 30.

[28] Wahlster, W. Verbmobil. In *Grundlagen und anwendungen der künstlichen intelligenz*. Springer, 1993. s. 393–402.

[29] Weilhammer, K. – Reichel, U. – Schiel, F. Multi-Tier Annotations in the Verbmobil Corpus. 01 2002.

[30] Yang, Z. et al. Universal sentence representation learning with conditional masked language model. *arXiv preprint arXiv:2012.14388*. 2020.

[31] Young, S. . M. G. a. The Hidden Information Statemodel: A practical framework for POMDP-based spoken dialogue management. *Computer Speech and Language*. 2010, , 24.

Příloha č. 1 - Uživatelská příručka

Následující text popisuje použitou technologii a popis spuštění jednotlivých modulů.
Z licenčních důvodů jsou data z Verbmobil a DIHANA sady zredukována.

Obsah přiloženého zip

- Text_prace - zdrojové soubory LaTeXa PDF této práce.
- Aplikace_a_knihovny - zdrojové soubory.

  preprocesor - preprocesor k parsování DIHANA korpusu.

  neuronky - moduly pro vytvoření neuronových sítí.
- Vstupni_data - data pro preprocesor a neuronové sítě.
- Readme.txt - detailní popis adresářové struktury a příklad spuštění jednotlivých modolů.
- requirements.txt - výčet python balíčků.

Použitá technologie

Programovací jazyk a knihovny

K vytvoření preprocesoru i k implementaci byl využit jazyk Python, konkrétně python 3.7. Implementace neuronových sítí využívá knihovnu tensorflow, konkrétně verze 2.8.0.
Jazyk jsem kvůli tomu, že v doméně strojového učení je dominantní, zejména kvůli podporovaným knihovnám.

Spuštění preprocesoru pro DIHANA corpus

V kořenovém adresáři spustit příkaz
python3 (python pro windows) preprocessor.py –dihana_corpus_path <cesta_k_dialogum>

Modul provede parsování dihana korpusu a vytvoří conll soubory a slovník.

Spuštění trénování a evaluace neuronových sítí

V kořenovém adresáři se zdrojovými soubory s neuronovými sítěmi spustit příkaz
python3 (python pro windows) main.py <parametry>
<parametry>:

- --train_data - en, ge, esp, en+ge+esp. Argument určuje, který jazyk bude využit pro učení modelu. Výchozí hodnota = en.
- --test_data - en, ge, esp, en+ge+esp. Argument určuje, který jazyk bude využit pro evaluaci modelu. Výchozí hodnota = en.
- --epoch_count - počet epoch při učení modelu. Výchozí hodnota = 10.
- --model - kim, bilstm, cnn, esec. Argument určuje model, který se použije pro učení a evaluaci. Výchozí hodnota = kim.
- --word_emb_layer - yes, no. Argument určuje zda model využije word embedding vrstvu. Výchozí hodnota = yes.
- --history - yes, no. Argument určuje zda model využije historii jako druhou vstupní vrstvu. Výchozí hodnota = yes.
- --load_model - cesta k modelu, který bude načten a evaluován. Výchozí hodnota není žádná.
- --dihana_classes - yes, no. Příznak, zda se používají DIHANA třídy (je nutné explicitně zmínit, pokud se pouští nenamapovaný DIHANA dataset). Výchozí hodnota = no.
- --model_name - název vytvořeného modelu. Model je uložen ve formátu h5 Výchozí hodnota = model<dnešní_datum>.

Popis implementace preprocesoru

preprocesor.py

Vstupní modul programu. Využivá modul argparse pro přebrání argumentů od uživatele. Vytváří instanci třídy FileParser, která slouží pro parsování korpusu DIHANA.
Při spuštění preprocesoru dojde k parsování dialogů a k vytvoření adresáře *dihana_dataset* , který bude obsahovat soubory:

- classes_dihanana_ESP.txt - obsahuje výčet tříd DIHANA datasetu, viz 4.2.

- test-ESP.conll - testovací množinu dialogů s příslušnými DH DA, popis formátu souboru viz 6.2.
- train-ESP.conll - trénovací množinu dialogů s příslušnými DH DA.
- VM_classes_dihana_ESP.txt - výčet VM tříd po mapování, viz mapovací tabulka 5.4.
- VM_test-ESP.conll - testovací množinu dialogů s příslušnými VM DA.
- VM_train-ESP.conll - trénovací množinu dialogů s příslušnými VM DA.
- voc-esp.txt - slovník pro španělský jazyk.

FileParser.py

Třída v konstrukturu přebírá jediný parametr, cestu k DIHANA korpusu. Třída přečte všechny soubory s dialogy, pomocí regulárního výrazu rozdělí jednotlivé promluvy do listu, který má na indexu i = 0 větu a na indexu j = 1 DA. Následně vytvoří výstupní soubory, viz výčet 8.

Constants.py

Modul udržuje konstanty - primárně cesty k souborům, názvy vytvářených souborů, etc. Pokud je nutné něco měnit na původní konfigurace preprocesoru, je nutné zde upravit hodnoty.

Popis implementace neuronových sítí

main.py

Modul slouží stejně jako u preprocesoru k přebrání argumentů od uživatele za pomocí modulu argparse.

data_loader.py

Modul slouží k parsování conll souborů, načítání w2v embedding souborů a aplikování jej pro vytvoření vektorové reprezentace vět a vytváření one hot vector reprezentace tříd DA.
Funkce load_sentences_embeddings přebírá 4 argumenty: sentences, classes, vocabulary, we_matrix.

- sentences : list vět v přirozeném jazyce, například angličtině.
- classes : list one hot vectorů tříd jednotlivých vět. Například větě sentences[5] přísluší třída classes[5].
- vocabulary : slovník všech slov a jejich četností, které se vyskytují v datových souborech
- we_matrix : w2v embedding matice, mapují se na ní slova pomocí indexu ve slovníku. Například slovo 'Gracias' má ve slovníku index 300 => we_matrix[300] = w2v reprezentace slova 'Gracias'.

nn_model.py

Modul využívá Keras functional API [10] pro vytváření modelů neuronových sítí.
Funkce create_model_kim_with_emb_layer vytváří model CNN_1.
Funkce create_cnn_model_with_emb_layer_input_history vytváří model CNN_2.
Funkce create_lstm_bidirect_with_emb_layer_with_history vytváří model BILSTM.
Funkce create_esec_model_with_history vytváří model ESEC.

config.py

Konfigurační soubor, obsahuje konstanty jako jsou cesty k w2v souborům a třídy DA. V případě cest k souborům je nutné zde přepsat cesty.