LATVIJAS UNIVERSITĀTE

DATORIKAS FAKULTĀTE

DAUDZVALODĪGU JĒDZIENTELPU PIELIETOJUMS NODOMU NOTEIKŠANĀ

MAĢISTRA DARBS

Autors: **Viktorija Leimane**

Studenta apliecības Nr.: vl16047

Darba vadītājs: Dr.sc.comp. Kaspars Balodis

RĪGA 2023

ANOTĀCIJA

Daudzvalodīga lietotāja nodomu noteikšana ir nozīmīga virtuālo asistentu darbībā, un klientu apkalpošanas automatizācija kļūst arvien izdevīgāka un aktuālāka. Viens veids noteikt nodomu ir attēlojot ievades teksta virknes daudzdimensionālā vektoru telpā jeb jēdzientelpā, kuru izmanto nodomu klasifikācijas modeļi, lai piegādātu lietotājiem tiem nepieciešamo informāciju. Darbā tiks apmācīti dažādi mašīnmācīšanās modeļi un salīdzinātas dažādas pieejas: ievades teksta attēlojums uz daudzvalodīgu tekstu korpusu apmācītas jēdzientelpas un ievades teksta mašīntulkošana uz angļu valodu un attēlojums uz angļu valodas korpusa apmācītas jēdzientelpas.

Atslēgas vārdi: daudzvalodīgas jēdzientelpas, nodomu noteikšana

ABSTRACT

Multilingual intention detection is important for virtual assistants, and customer-service automation is becoming more cost-effective and relevant. One way of detecting intent is mapping input text strings to to multidimensional vector space, which is used by intent classification models to supply users with the information they need. This thesis focuses on training a variety of machine learning models and comparing different approaches, such as mapping input text to multingual word embeddings and machine translating input text to English and using English corpus-based word embeddings for intent detection.

Keywords: multilingual word embeddings, intent detection

SATURS

APZĪMĒJUMU SARAKSTS

NLP (*natural language processing*) – dabisko valodu apstrāde.

Jēdzientelpa (*word embeddings*) – vārdu vai frāžu attēlojums daudzdimensionālā vektoru telpā.

Word2Vec (*word to vector*) – jēdzientelpas implementācija, kurā individuālus vārdus aizstāj daudzimensionāli vektori.

PCA (*Principal Component Analysis*) – galveno komponentu analīze.

GPT (*Generative Pre-trained Transformer*) – dziļo neironu tīklu modelis, kas spēj producēt tekstu, kas līdzīgs cilvēka rakstītam.

Transformeris (*transformer*) – dziļās mācīšanās modelis ar uzmanības (attention) mehānismu, kas spēj novērtēt ievades daļas nozīmīgumu.

Pārpielāgošana (*overfitting*) – pārmērīga pielāgošanās kādam konkrētai datu kopai, zaudējot spēju ģeneralizēt uz citām datu kopām.

Pietrenēšana (*fine-tuning*) – metode, kurā iepriekš apmācīts modelis tiek pietrenēts jaunam uzdevumam.

XLM (*Cross-lingual Language Model*) – valodas modelis, kas apmācīts uz daudzvalodu datiem, lai apgūtu jēdzientelpas, ko var pielietot vairākām valodām.

Multilingual BERT (*Bidirectional Encoder Representations from Transformers*) –

XLM-R (*Cross-lingual Language Model pre-trained with XLM*) –

XLM-Roberta (*Robustly Optimized BERT Pretraining Approach*) –

IEVADS

Arvien lielāku daļu tirgus pārņem pakalpojumu industrija, un pakalpojumi arvien biežāk tiek piedāvāti starptautiski. Tam ir nepieciešams lietotāju dzimtās valodas atbalsts gan valstu valodu regulējumu, gan tirgus nišas ieņemšanas un tirgus konkurences dēļ.

Uzņēmumiem tas ir izdevīgi, jo ļauj samazināt personālizdevumus. Tas savukārt samazina barjeru iekļūšanai un dalībai starptautiskā tirgū, kas nozīmē lielāku konkurenci un piedāvāto pakalpojumu daudzveidību. Lietotājiem, kuru dzimto valodu pārvalda mazs cilvēku skaits kā tas ir, piemēram, latviešu valodā, kļūst pieejami pakalpojumi, kuru tulkojumus būtu ekonomiski nerentabli nodrošināt ar algotu profesionālu personālu.

Darbā apskatītā metode nodrošina automatizāciju divos veidos:

- daudzvalodīgs modelis aizvieto profesionālu tulkotāju;
- virtuālais asistents aizvieto klientu apkalpošanas speciālistu.

Darbs ir sadalīts teorētiskajā un praktiskajā daļā. Teoretiskajā daļā ir īsi aprakstīti mūsdienu modeļi un pieejas. Praktiskajā daļā ir veikti eksperimenti ar mērķi pielietot daudzvalodīgus modeļus un salīdzināt tos ar esošiem risinājumiem.

Pētījuma jautājums: Kādas ir efektīvākās metodes un daudzvalodīgi jēdzientelpu modeļi daudzvalodu nodomu noteikšanai?

1. KORPUSI

Dabiskā valodas apstrāde (NLP – *natural language processing*) ir starpdisciplināra datorlingvistikas un mākslīgā intelekta nozare, kas strādā pie tā, lai datori varētu saprast cilvēka dabiskās valodas ievadi. Dabiskās valodas pēc būtības ir sarežģītas, un daudzi NLP uzdevumi ir slikti piemēroti matemātiski precīziem algoritmiskajiem risinājumiem. Palielinoties korpusu (liela apjoma rakstītas vai runātas dabiskās valodas kolekcija) pieejamībai, NLP uzdevumi arvien biežāk un efektīvāk tiek risināti ar mašīnmācīšanās modeļiem [1]. Dabiskās valodas apstrādei ir liels biznesa potenciāls, jo tas ļauj uzņēmumiem palielināt peļņu samazinot izdevumus, no kuriem lielākais parasti ir darbs.

Viens no svarīgākajiem korpusiem tieši nodomu noteikšanā ir aviokompāniju ceļojumu informācijas sistēmu (ATIS – *Airline Travel Information Systems*) datu kopa. Tā ir audioierakstu un manuālu transkriptu datu kopa, kas sastāv no cilvēku sarunām ar automatizētām aviolīniju ceļojumu informācijas sistēmām. ATIS datu kopa nodrošina lielu ziņojumu un ar tiem saistīto nodomu skaitu, ko plaši izmanto kā novērtējuma (*benchmark*) datu kopu klasifikatoru apmācībai nodomu noteikšanā [2].

Lielo teksta korpusu un mašīnmācīšanās modeļu precizitātes vēsture ir cieši saistīta. Agrīnie mašīnmācīšanās algoritmi balstījās uz nelielām manuāli veidotām datu kopām, kas ierobežoja to efektivitāti. Viens no pirmajiem korpusiem bija *Standard Sample of Present-Day American English*, plašāk pazīstams kā *The Brown Corpus*, kas tika izdots 1964-1965. gadā un sastāvēja no apmēram viena miljona vārdu angļu teksta no dažādiem avotiem [3], tas ir mazs apjoms teksta salīdzinot ar mūsdienās pieejamo.

Procesoru jaudas palielināšanās kopā ar datoru un interneta savienojuma pieejamību plašākai sabiedrībai ir radījuši labvēlīgu vidi izveidot un uzglabāt lielu daudzumu digitālo datu, tostarp teksta formā. Lieliem teksta korpusiem ir bijusi izšķiroša loma efektīvu mašīnmācīšanās modeļu izstrādē. Mašīnmācīšanās modeļu efektivitāte ir proporcionāla tiem pieejamo apmācības datu lielumam un kvalitātei.

Pirms lielu teksta korpusu pieejamības mašīnmācīšanās modeļi aprobežojās ar mazām un salīdzinoši vienkāršām datu kopām, tādēļ bija grūti sasniegt augstu precizitāti dabiskās valodas apstrādes uzdevumos. Mūsdienās lielos teksta korpusos kā Common Crawl un Wikipedia ir miljardiem vārdu vairākās valodās, kas ļauj modeļiem iemācīties ģenerēt cilvēkiem līdzīgu valodu.

SNIPS – *Spoken Natural Language Interaction for Personal Assistant*) datu kopā ir 16 000 ievadi, kas sadalīti septiņos nodomos: SearchCreativeWork, GetWeather, BookRestaurant, PlayMusic, AddToPlaylist, RateBook, SearchScreeningEvent.

Three tasks are successively performed. Intent Classification consists in extracting the intent expressed in the query (e.g. SetTemperature or SwitchLightOn). Once the intent is known, Slot Filling aims to extract the slots, i.e. the values of the entities present in the query. Finally, Entity Resolution focuses on built-in entities, such as date and times, durations, temperatures, for which Snips provides an extra resolution step. It basically transforms entity values such as "tomorrow evening" into formatted values such as "2018-04-19 19:00:00 +00:00". Snippet 1 illustrates a typical output of the NLU component

Behind every chatbot and voice assistant lies a common piece of technology: Natural Language Understanding (NLU). Anytime a user interacts with an AI using natural language, their words need to be translated into a machine-readable description of what they meant. The NLU engine first detects what the intention of the user is (a.k.a. intent), then extracts the parameters (called slots) of the query. The developer can then use this to determine the appropriate action or response.

Natural Language Understanding (NLU) is a fundamental technology that powers chatbots and voice assistants. When a user communicates with an AI using natural language, the words are translated into a machine-readable format to determine what the user intended to convey. The NLU engine identifies the user's intent, followed by the extraction of parameters or slots of the query. By utilizing this extracted information, developers can determine the most suitable response or action to take.

he main metric used in this benchmark is the average F1-score of intent classification and slot filling. The data consists in three corpora. Two of the corpora were extracted from StackExchange, one from a Telegram chatbot. The exact same splits as in the original paper were used for the Ubuntu and Web Applications corpora. At the date we ran the evaluation, the train and test splits were not explicit for the Chatbot dataset (although they were added later on). In that case, we ran a 5-fold cross-validation. T

Kad cilvēks ievada tekstu dabiskā valodā, tas tiek pārveidots jēdzientelpā. Pēc tam klasifi-

kators nosaka lietotāja nodomu.

Dabisko valodu saprašanas (Natural Language Understanding (NLU)) uzdevumi iedalās divās daļās: nodomu noteikšana un parametru (slots) iegūšana no ievadiem.

1.1. tabula

Add caption

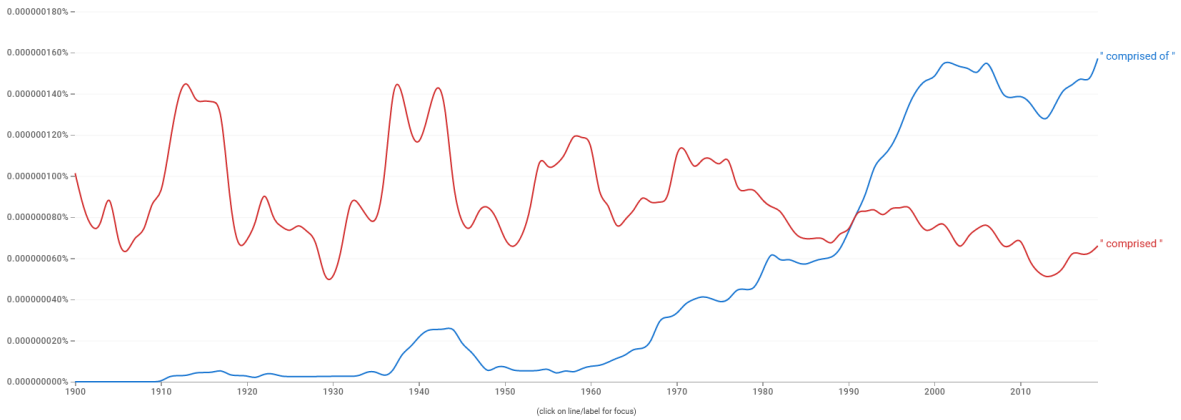
Intent Name	Slots	Samples
	region	Is it cloudy in Germany right now?
	country	Is South Carolina expected to be sunny in 2 hours?
ForecastCondition	datetime	Is there snow in Paris?
	locality	Should I expect a storm near Mount Rushmore?
	condition	
	point of interest	

Taču vai visas valodas ir līdzvērtīgi pārstāvētas korpusos? Viegli iztēloties, ka tādi lielie korpusi kā Common Crawl (kopš 2008. gada ievākti petabaiti datu no interneta mājaslapām, tostarp Vikipēdijas un Reddit) līdzvērtīgi pārstāv visu Zemes iedzīvotāju valodas. Taču valodu reprezentācija korpusos ir saistīta ar rakstītā teksta datu pieejamību šajā valodā, un neprecīzi atspoguļo cilvēku skaitu, kuri runā šajā valodā. Piemēram, valodai, kurā runā liels skaits cilvēku, korpusā var būt maz marķieru, ja šajā valodā ir maz digitāli pieejama rakstīta teksta.

Tomēr dažādi faktori traucē visiem rakstīt tekstus internetā, kas vēlāk nokļūst korpusos, piemēram, rakstītneprasme, nabadzība, ierīču un interneta nepieejamība, karš utml. Tā rezultātā korpusos ir disproporcionāli pārstāvēti gados jaunāku lietotāju no attīstītajām valstīm drukāti teksti, piemēram, GPT-2 apmācības dati tika ievākti no Reddit, un pēc Pew Internet Research pētījuma 67% Reddit lietotāju Amerikas Savienotajās Valstīs ir vīrieši un 64% vecumā no 18 līdz 29 gadiem.[4].

Līdzīgi 87% Vikipēdijas ierakstu veicēji ir vīrieši. Gandrīz puse dzīvo Eiropā un viena piektā daļa Ziemeļamerikā, salīdzinot ar 9.7% un 4.8% pasaules iedzīvotāju [5]. Analizējot labojumus Vikipēdijas rakstos no 2001. līdz 2010. gadam, 1% visbiežākie ierakstu veicēji uzrakstīja 77% satura [6].

Korpusos tam ir vairākas praktiskas implikācijas gan sintaksē, gan semantikā. Piemēram, Vikipēdijas autors Brians Hendersons (*Bryan Henderson*) 15 gados veica 90 tūkstošus labojumu, kur lielākā daļa izmaiņu ir no ”comprised of” uz ”comprised”, kaut gan abas formas tiek pieņemtas un citos rakstiskos avotos ”comprised of” ir izplatītāks (2.1 attēls). Tāpat BERT biežāk asociē cilvēkus ar invaliditāti ar negatīva sentimenta vārdiem un vairāki darbi to sasaista ar treniņu datu kopu īpašībām [4]. Pētīt negatīvu sentimentu valodu korpusos ir svarīgi, jo kompānijas reputācija ciestu, virtuālajam asistentam sniedzot atbildes ar negatīviem steriotipiem klientu apkalpošanas jomā.



1.1. att. **Uz x ass attēloti gadi, uz y ass – cik procentu no visiem vārdiem, kas ietverti angļu valodā rakstīto grāmatu korpusā (English 2019), ir ”comprised of” un ”comprised”? [7]**

Ar to tiek pierādīts, ka tekstu nav radījuši nejauši izvēlēta izlase cilvēku, tāpēc teksts nav neitrāls. Tiek paredzēts, ka virtuālos asistentus izmantos plašāks cilvēku loks nekā šobrīd internetā publicēto tekstu autori, tāpēc ir svarīgi, lai treniņdatos ir atbilstoši pārstāvēta potenciālo lietotāju valoda.

Latviešu marķieru daļa Common Crawl 100 korpusā ir atkarīga no daudziem faktoriem, tostarp latviešu satura daudzuma tīmeklī un korpusa konstruēšanā izmantotās izlases metodikas. Iespējams, ka latviešu teksta saturs korpusā ir pārāk vai nepietiekami pārstāvēts, salīdzinot ar tā izplatību tīmeklī vai proporcionāli latviešu valodā runājošo īpatsvaram.

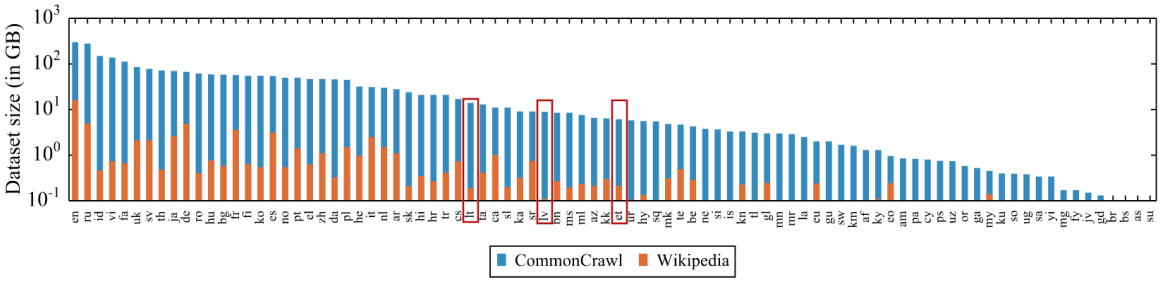
Kas padara valodu par maz-resursu? Mazāks skaits vārdu un teikumu datu kopās, tātad mazāks skaits tokenu uz kuriem trenēt daudzvalodu jēdzientelpu modeli. Piemēram, Common

Crawl-100 korpusā, uz kura trenēts XLM-R modelis, svahili un urdu valodās ir 275M un 730M tokenu attiecīgi, darbā izmantotajās - lietuviešu, latviešu, igauņu - ir 1835M, 1198M un 843M tokenu attiecīgi (1.2 tabula), tātad šīs datukopas kontekstā tās var uzskatīt par maz-resursu.

1.2. tabula

Common Crawl-100 valodas un statistika: valodu saraksts ar marķieru (*tokens*) skaitu (miljonos) un datu izmēru gibibaitos (GiB) katrai valodai

ISO kods	Valoda	Marķieri (M)	Izmērs (GiB)
en	angļu	55608	300.8
ru	krievu	23408	278.0
lt	lietuviešu	1835	13.7
lv	latviešu	1198	8.8
et	igauņu	843	6.1
ur	urdu	730	5.7
sw	svahili	275	1.6



1.2. att. **Datu apjoms GiB (logaritmiskā skalā) valodām Wiki-100 korpusā, ko izmanto mBERT un XLM-100, un Common Crawl-100, ko izmanto XLM-R. Common Crawl-100 palielina datu apjomu par vairākām kārtām, jo īpaši maz-resursu valodās (lietuviešu, latviešu, igauņu valodas ar sarkanu izdalīju es) [8]**

2. JĒDZIENTELPA

Jēdzientelpa ir vārdu vai frāžu attēlojums daudzdimensionālā vektoru telpā. Jēdzientelpu pamatā ir ideja, ka vārdiem, kuriem ir līdzīga nozīme un kurus lieto līdzīgos kontekstos, daudzdimensiju telpā jābūt savstarpēji tuvākiem, bet vārdiem ar atšķirīgu nozīmi un kontekstiem jābūt tālākiem, piemēram, vārds "suns" būs tuvāk vārdiem "kaķis" un "mājdzīvnieks" nekā vārds "koks".

To, cik tuvu ir vārdi jēdzientelpā, var noteikt, izmantojot kosinusa līdzības (*cosine similarity*) metriku [9]. Ja vienam vārdam atbilst vektors $\vec{a}$, bet otram – vektors $\vec{b}$, tad kosinusa līdzību $K.L.$ var atrast šādā veidā:

$$K.L. = \cos(\varphi) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|},$$

kur φ ir leņķis starp vektoriem $\vec{a}$ un $\vec{b}$, bet $||$ apzīmē vektora garumu jeb moduli: $|\vec{a}| = \sqrt{\vec{a} \cdot \vec{a}}$.

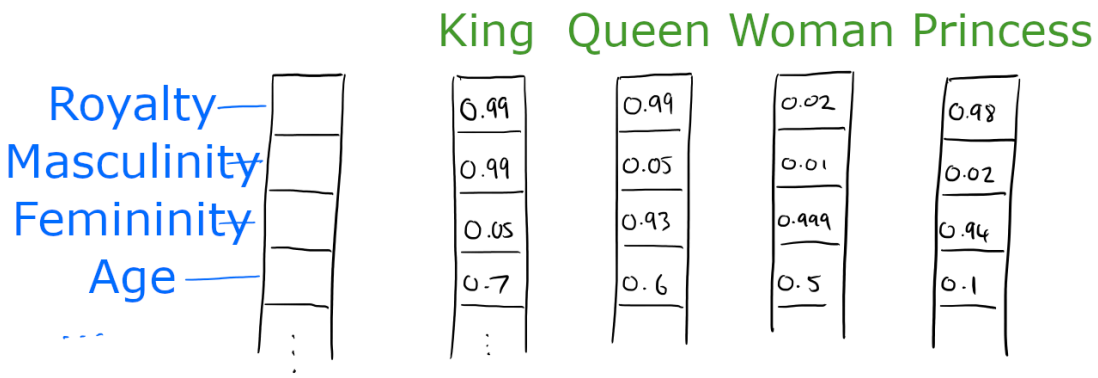
Noderīgs sākumpunkts izpratnei par vārdu attēlošanu ar skaitļu vektoriem un šo attēlojumu pielietojamības robežām ir vienizcēluma kodējums (*one hot encoding*). Vienizcēluma kodējums dabiskās valodas apstrādē ir vektors, kurā katrs vektora elements ir sasaistīts ar vārdu krājuma elementu. Līdz ar to katrs vārds ir vektors, kurā atbilstošais elements ir 1 un visi pārējie elemeni ir 0. Piemēram, ja vārdu krājumā ir četri vārdi: karalis, karaliene, sieviete, vīrietis, karaliene tiktu kodēta kā [0, 1, 0, 0] [10].

Taču vektora garuma sasaiste ar vārdu krājuma izmēru ir trūkums, jo vārdu vektori ir cieši savienoti (*coupled*) ar korpusu un statiski, piemēram, pievienot jaunu vārdu nozīmē katram esošajam vārdu vektoram pievienot papildus nulli, tātad nāktos pārtrenēt visu modeli. Tāpat palielinoties dimensiju skaitam telpa pieaug tik strauji, ka daudzdimensiju telpām raksturīgs neblīvums/izretinātība (*sparsity*): vienizcēluma kodējuma vektorā ir tikai viens nenulles elements un korpusos mēdz būt miljardiem vārdu. Visbeidzot vienizcēluma kodējums nesatur kontekstuālu vārdu nozīmi, nav korelācijas starp vārdiem ar līdzīgu nozīmi un lietojumu [10].

Atšķirībā no dabisko valodu apstrādes metodēm, kas katru vārdu uztver kā vienu atsevišķu vienību un tādēļ vienīgā iespējamā darbība ar vārdiem ir pārbaudīt vienādību, katras jēdzientelpas vektora vērtības ietekmē vārdi tiem apkārt jeb reprezentācija ir izkliedēta (*distributed representation*) un būtībā jēdzientelpas uztver attiecības starp vārdiem. Rezultātā vārdam atbilstošais vektors satur semantisku un sintaktisku informāciju par vārdu. No tā izriet praktiskā implikācija – ar vektoriem var veikt lineārās algebras operācijas, piemēram, saskaitīt un atņemt [10].

Vārdus ir daudz grūtāk salīdzināt nekā skaitļus, tādēļ mēs piešķiram vārdiem vektorus. To-

mēr vārdi apraksta objektus ar noteiktām kvantificējamām īpašībām, piemēram, vieglāks/smagāks (svars), lētāks/dārgāks (cena). Šādai reprezentācijai ir jēga, jo dažādus objektus var salīdzināt savā starpā pēc īpašību vērtības jeb izteiktības pakāpes, piemēram, velosipēds ir vieglāks nekā mašīna. Tādā veidā vārda attēlojums tiek sadalīts pa visiem vektora elementiem, un katrs elements pievieno nozīmi daudziem vārdiem (2.1 attēls). Zinot, ka objektu īpašību skaitliska reprezentācija palīdz tos salīdzināt, atklājas jēga kvantitatīvi izteikt semantiku, tādējādi vārdi tiek attēloti veidā, kas izsaka to nozīmi caur kontekstu.



2.1. att. **Vārdu vektoru piemērs, kur katra dimensija ir novērtēta ar svariem un atbilst hipotētiskai vārda nozīmes niansei [10].**

Cilvēkiem uztverama jēdzientelpu analoģija ir krāsas nosaukums un tam atbilstošais vektors RGB krāsu modelī ar R, G un B koordinātēm no 0 līdz 255, piemēram, red = (255, 0, 0). Ar krāsu jēdzientelpām ir iespējams veikt saskaitīšanu un atņemšanu, kam ir fizikāla nozīme [11].

Atrast tuvākās krāsas sarkanam.

```
closest(colors, colors['red'])
# red (229, 0, 0)
# fire engine red (254, 0, 2)
# bright red (255, 0, 13)
# tomato red (236, 45, 1)
# cherry red (247, 2, 42)
```

Operācijas ar vektoriem darbojas gan krāsu nosaukumiem semantiski, gan skaitliskiem vektoriem krāsu telpā. Piemēram, tuvākais vektors violeta un sarkana starpībai ir zils, kas atbilst

cilvēku intuīcijai par RGB krāsām.

$$purple - red = blue$$

$$(126, 30, 156) - (229, 0, 0) = (-103, 30, 156)$$

```
closest(colors, subtractv(colors['purple'], colors['red']))
# cobalt blue (3, 10, 167)
# royal blue (5, 4, 170)
# darkish blue (1, 65, 130)
# true blue (1, 15, 204)
# royal (12, 23, 147)
```

Tā saskaitot zaļu un zilu rodas kaut kas pa vidu – tirkīzs.

$$blue + green = turquoise$$

$$(3, 67, 223) + (21, 176, 26) = (24, 243, 249)$$

```
closest(colors, addv(colors['blue'], colors['green']))
# bright turquoise (15, 254, 249)
# bright light blue (38, 247, 253)
# bright aqua (11, 249, 234)
# cyan (0, 255, 255)
# neon blue (4, 217, 255)
```

No vektoru operācijām var nolasīt secinājumus par semantiskajām attiecībām starp vārdiem, piemēram, rozā sarkanam ir tas pats, kas gaiši zils zilam.

$$pink - red + blue = lightblue$$

$$(255, 129, 192) - (229, 0, 0) + (3, 67, 223) = (29, 196, 415)$$

```
closest(colors, addv(subtractv(colors['pink'], colors['red']), colors['blue
                                                                        ']))
# neon blue (4, 217, 255)
# bright sky blue (2, 204, 254)
# bright light blue (38, 247, 253)
# cyan (0, 255, 255)
# bright cyan (65, 253, 254)
```

Kā analoģiju izkliedētai reprezentācijai var apsvērt arī ģeogrāfiskā platuma un garuma koordinātas kā vektora attēlojumu vietu nosaukumiem. Divu ģeogrāfisku punktu tuvums koordinātēs var norādīt uz līdzīgu klimatu, vēsturi, kultūru un citiem faktoriem. Piemēram, Rīga (56°57′N 24°6′E) ir līdzīgāka Viļņai (54°41′N 25°19′E) nekā Riodežaneiro (22°54′40″S 43°12′20″W). Tāpat jēdzientelpas ir veids, kā attēlot vārdus kā vektorus daudzdimensiju telpā, kur vārdi ar līdzīgu nozīmi vai kontekstu atrodas tuvāk viens otram. Tāpat kā krāsas var attēlot kā vektorus RGB telpā un vietas var attēlot kā vektorus platuma-garuma telpā, vārdus var attēlot kā vektorus semantiskā telpā, kas atspoguļo to attiecības ar citiem vārdiem.

Izrādās, tādas pašas sakarības, kādas ir krāsu nosaukumiem un to attēlojumiem krāsu telpā, ir spēkā jebkuram vārdam. Vārdi, kuri bieži atrodas līdzīgos kontekstos, ir tuvāki pēc nozīmes. Jēdzientelpas ietver gan sintaktiskas (2.2 tabula), gan semantiskas (2.1 tabula) attiecības starp vārdiem. Jāuzsver, ka tādas semantiskas attiecības kā valsts–galvaspilsēta (2.2) nav uzdotas tiešā veidā, jēdzientelpu modelis tās ir novērojis tikai balstoties uz vārdu atrašanās vietām teksta korpusā. Iespēja trenēt modeli uz neanotētiem datiem kā šajā gadījumā samazina modeļa trenēšanas izmaksas valodām, kurās anotēti dati ir mazāk pieejami, un daudzkārt palielina potenciālās treniņu kopas apjomu, kas parasti ļauj sasniegt lielāku precizitāti.

Spēja noteikt sintaktiskas un semantiskas vārdu attiecības ir īpaši būtiska virtuālo asistentu jomā, jo, pirmkārt, semantiski līdzīgiem nodomiem ir līdzīgi vektori, tātad tie tiks vienādi klasificēti, otrkārt, informācija par sintakses attiecībām noder, jo lietotāji ievada jautājumus brīvā formā un tas ir it īpaši svarīgi fleksīvām valodām kā latviešu.

2.1. tabula

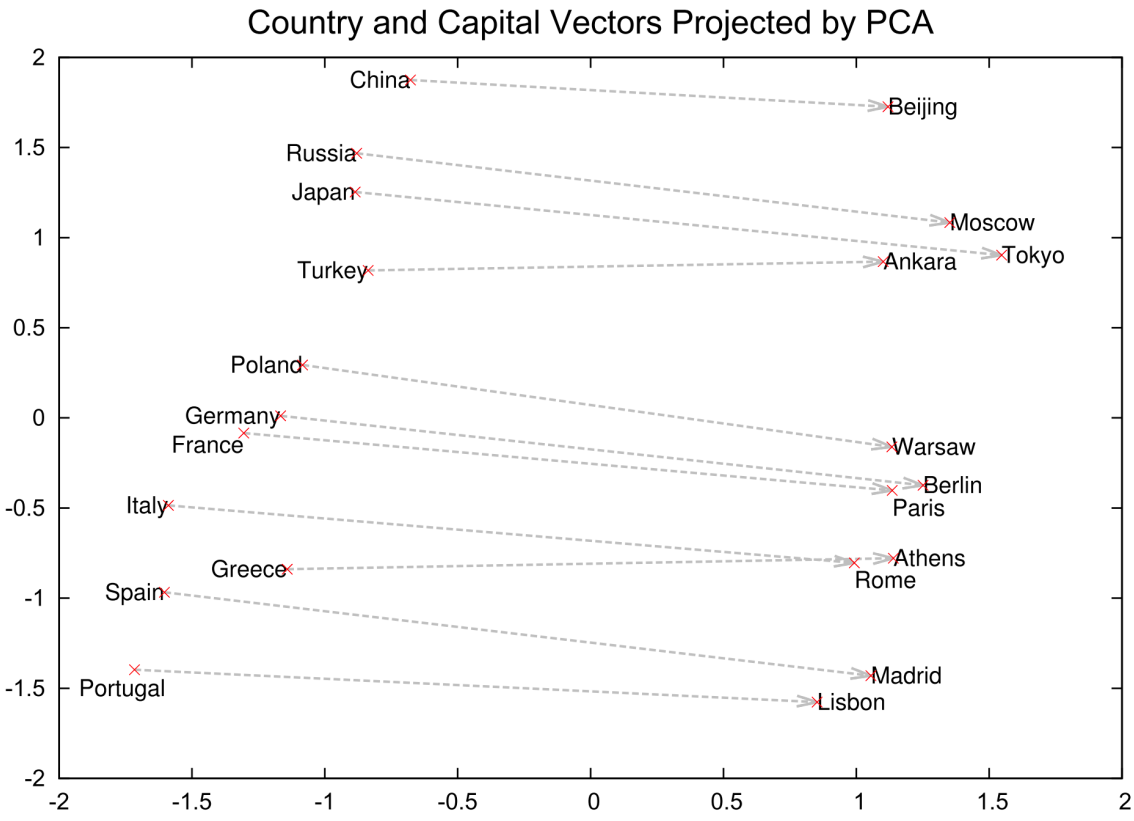
Semantisko attiecību piemēri [12]

attiecība	piemērs
valsts–galvaspilsēta	Parīze - Francija + Itālija = Roma
valsts–valūta	dolāri - ASV + Latvija = eiro
vīrietis–sieviete	karalis - vīrietis + sieviete = karaliene

2.2. tabula

Sintaktisko attiecību piemēri [12]

attiecība	piemērs
daudzskaitlis	pele - peles
pagātne	staigā - staigāja
salīdzināmā pakāpe	labs - labāks



2.2. att. **Divdimensionāla PCA projekcija uzrāda attiecības starp valstu un galvaspilsētu jēdzientelpām [10]**

2.1. Daudzvalodīga jēdzientelpa

Daudzvalodīgas jēdzientelpas no vienvalodīgām jēdzientelpām atšķiras ar to, ka uztver attiecības starp vārdiem no dažādām valodām.

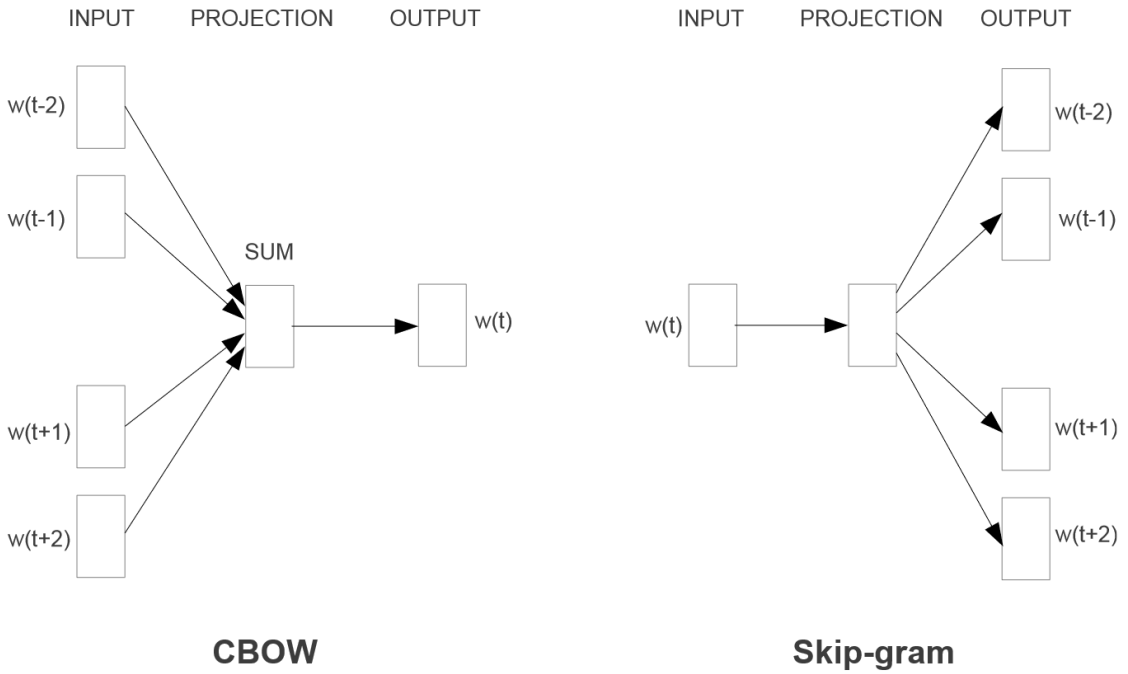
Tā kā vienvalodīgas jēdzientelpas tiek trenētas tikai uz vienas valodas, tās nespēj notvert attiecības starp vārdiem dažādās valodās, un spēj raksturot tikai attiecības starp vārdiem vienā valodā, piemēram, vārds ”suns” tiek reprezentēts kā vektors kas jēdzientelpā ir tuvs citiem ar suņiem saistītiem vārdiem kā ”kucēns” un ”riet”, bet nav sasaistīts ar suņiem saistītiem vektoriem citās valodās.

Turpretī daudzvalodīgas jēdzientelpas tiek trenētas uz paralēliem datiem - vienādas nozīmes tekstiem dažādās valodās. Tas ļauj notvert starpvalodu (*cross-lingual*) sakarības starp līdzīgas nozīmes vārdiem kopējā jēdzientelpā, piemēram, vārdi ”suns” un ”dog” (”suns” angļu valodā) tiek reprezentēti kā tuvi vektori kopējā jēdzientelpā, kas norāda uz līdzīgu nozīmi. Daudzvalodīgas jēdzientelpas īpaši noder valodām ar mazākām treniņdatu kopām, jo palīdz tulkot un atgriezt informāciju starp valodām (*cross-language information retrieval*) – piemēram atgriezt kādām vaicājumam angļu Vikipēdijas lapu, ja tai nav latviešu Vikipēdijas ekvivalenta.

3. JĒDZIENTELPU MODEĻU APMĀCĪBA

Jēdzientelpas no teksta korpusa iegūst ar neironu tīkliem, kuri uztver kontekstu no tuvākajiem vārdiem tekstā.

Continuous Bag-of-Words (CBOW) un *Continuous Skip-gram Model* ir divas neironu tīklu modeļu arhitektūras jēdzientelpu izveidei balstoties uz teksta korpusa. Metožu priekšrocība ir tajā, ka nav nepieciešama anotēta treniņu datu kopa, trenēšanai izmanto lielus teksta korpusus. CBOW modelī apkārt esošos vārdus izmanto vidū esošā vārda paredzēšanai. Skip-gram modelī vārda vektoru izmanto konteksta paredzēšanai (3.1 attēls).



3.1. att. **CBOW un Skip-gram modeļu arhitektūra [12].**

3.1. Continuous Bag-of-Words

Bag-of-Words (BOW) apzīmē vārdu grupu nesaglabājot kārību. Vienā izlasē (bag) vārda tuvums mērķa vārdam konkrētā izlasē nav tik svarīgs, atkārtojot procesu uz korpusa no konteksta tāpat tiks sīkāk (granulētāk) izšķirti svari tuvākajiem vārdiem, piemēram, Rīga un Latvija būs tuvumā 1000 reizes biežāk nekā Rīga un sniegs.

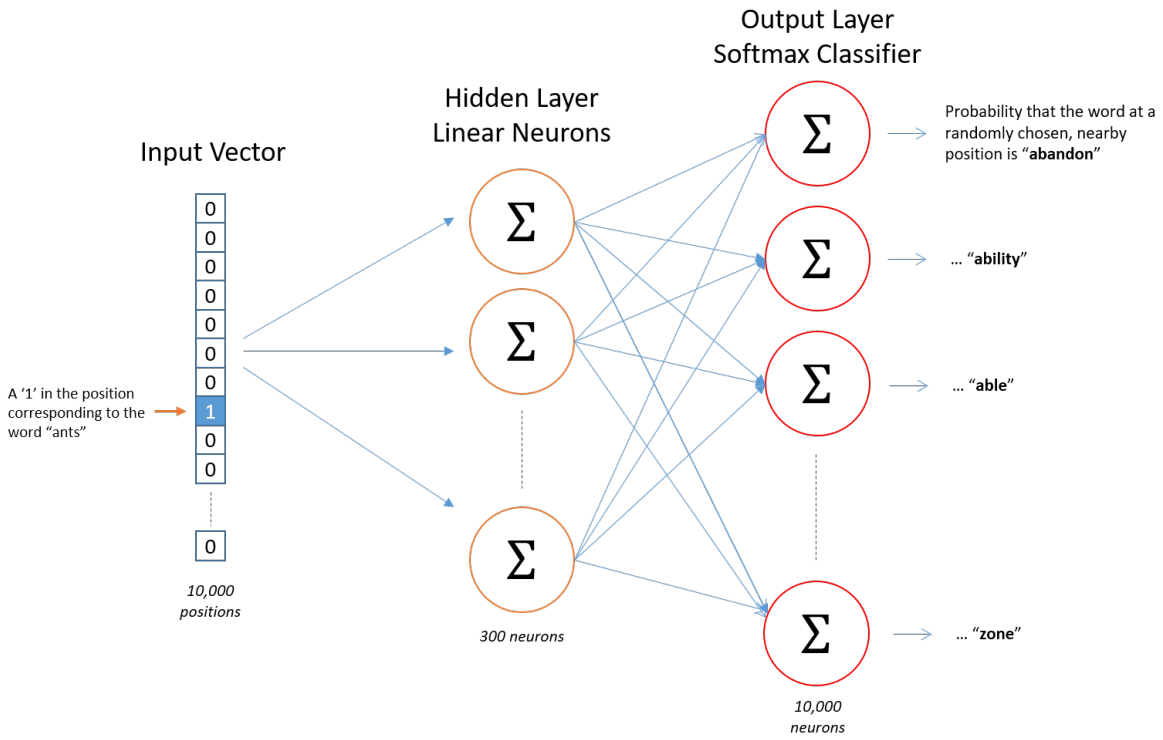
CBOW (Continuous Bag-of-Words) metodē neironu tīkls mēģina uzminēt esošo (vidējo)

vārdu no n iepriekšējiem un n nākošajiem vārdiem. Procesu atkārtojot, vārdiem, kas bieži parādās vienā kontekstā, būs līdzīgi vektori. Pēc izkliedētības (*distributional*) hipotēzes vārdi, kas atrodas līdzīgos kontekstos, ir ar līdzīgu nozīmi [12]. Tāpat kā BOW modelī, CBOW vārdu secība neietekmē projekciju. Nepārtrauktība (*continuuity*) modelī rodas no tā, ka izmanto nepārtrauktu izkliedētu konteksta reprezentāciju jeb svari starp ievades un projekcijas slāņiem tiek lietoti visiem vārdiem. [12]. CBOW neironu tīkla galvenās daļas ir kartējuma (*embedding*) slānis un tam sekojošs blīvais (*dense*) slānis, kurā katrs slāņa mezgls ir pilnībā savienots ar citu mezglu nākamajā slānī (3.1 attēls).

3.2. Continuous Skip-gram Model

Continuous Skip-gram metode ir līdzīga CBOW, tikai tās uzdevums ir paredzēt nevis vienu vārdu no apkārt esošajiem, bet otrādi – no ievades vārdu pa vārdam paredzēt apkārt esošos vārdus. Šīs metodes ideja ir uztrenēt neironu tīklu ar slēpto slāni (*hidden layer*) un iegūt slēptā slāņa svarus, kas patiesībā arī ir vārdu vektori. Vērā ņemamu kaimiņu vārdu skaits – loga izmērs (*window size*) – ir hiperparametrs (3.2 attēls). Modeļa trenēšanas laikā kaimiņu vārdu svara koeficienti nav vienādi: jo tālāk tie atrodas no aplūkotā ievades vārda, jo mazāk ietekmē tā nozīmi, tāpēc tālāk esošajiem vārdiem tiek piešķirti mazāki svara koeficienti [12].

Daudzvalodu valodu modeļi, piemēram, mBERT un XLM, tiek apmācīti uz vairākām valodām, un tas nodrošina efektīvu starp-valodu zināšanu pārnesi. Tie ir ievērojami uzlabojuši veikumu starpvalodu izpratnes (cross-lingual understanding) (*cross-lingual understanding*) uzdevumos, tostarp starpvalodu dabiskās valodas izvedumos (*cross-lingual language inference*), kas ietver sevī semantiskās līdzības noteikšanu starp teikumiem dažādās valodās [8]. Piemēram, salīdzinot teikumu ”the dog is sleeping on the mat” angļu valodā un teikumu ”suns guļ uz paklāja” latviešu valodā, var noteikt, ka šie divi teikumi ir semantiski līdzvērtīgi, neskatoties uz to ka tie ir rakstīti dažādās valodās.



3.2. att. **Skip-gram modeļa arhitektūra. Ievades vektors – vārda vienizcēluma kodējums** $1xn$ **– kur** n **ir vārdu skaits; slēptais slānis –** nxl**, kur** l **ir loga izmērs; Softmax slānis –** $1xl$**; Izvades slānis –** $1xn$ **[13].**

4. NODOMU NOTEIKŠANA

Nodoms ir mērķis, kas lietotājam ir padomā, rakstot jautājumu. Nodomu noteikšana ir lietotāja ievades teksta klasifikācija tam piešķirot visvarbūtīgāko nodomu no iepriekš definētu nodomu kopas [14]. Piemēram, klasificējot lietotāja nodomu kā vilciena atiešanas laiks, čatbots var sniegt lietotājam nepieciešamo atbildi no vilcienu grafika (4.1 tabula). Dabiskā valodā ir vairāki veidi kā izteikt vienu un to pašu nodomu (4.2 tabula).

4.1. tabula

Lietotāja ievada un nodoma piemērs

Ievads	Cikos ir nākošais vilciens no Rīgas uz Siguldu?
Nodoms	vilciena atiešanas laiks

4.2. tabula

Lietotāja dažādi ievadi ar vienu nodomu [15]

Nodoms	Ievads
switchLightOn	Ieslēdz gaismu
switchLightOn	Istabā ir pārāk tumši, vai vari to izlabot?

Pirms mašīnmācīšanās nodomi tika noteikti ar šabloniem (*pattern-based recognition*), bet izveidot un uzturēt lielu skaitu šablonu ir darbietilpīgi. Advancētāka pieeja nodomu noteikšanai ir apmācīt neironu tīklu klasifikatoru uz anotētas datu kopas – lietotāju ievades tekstiem un atbilstošajiem klientu apkalpošanas speciālista identificētajiem lietotāja nodomiem. Ierobežotās apmācību kopas dēļ dialogsistēmas/virtuālie asistenti var atbildēt uz ierobežotu jautājumu klāstu, piemēram, aptverot bieži uzdotos jautājumus (FAQ – *Frequently Asked Questions*) [14].

Lai arī jaunāko valodas modeļu, piemēram, GPT-3, izvades teksti lietotājam rada iespaidu par tekošu valodu, pastāv neparastās ielejas (*uncanny valley*) efekts, kurā novērotā plūstošā atbildes valoda rada ekspektācijas, kuras virtuālie asistenti nevar attaisnot un izraisa neapmierinātību [16]. Tāpēc klienta nodoma noteikšana ir svarīga, lai nodrošinātu patīkamu lietotāja pieredzi.

Jāpiebilst, ka labuma gūšanai no nodomu noteikšanas automatizācijas nav nepieciešams pārklāt 100% lietotāju pieprasījumu. Veiksmīgas izmantošanas piemērs telekomunikāciju industrijā

validācijā izmantoja 1732 klientu pieprasījumu datu kopu anotētu ar attiecīgajiem nolūkiem. Šajā gadījumā divi visbiežākie nodomi ir rēķina atlikšana (356 pieprasījumi; 21% datu kopas) un nokavēta rēķina maksājuma apstiprināšana (207 pieprasījumi; 12% datu kopas). Trīs mēnešus ilgā eksperimentālā pētījuma tika apstrādāti 14000 lietotāju pieprasījumi. Sākotnējos testos nodomu noteikšana un izvēlētā atbildes veidne bija precīza 90% gadījumu, eksperimenta gaitā iegūtie dati ļāva uzlabot nodomu noteikšanu par 2%, tātad klientu apkalpošanas speciālistiem bija jāveic izmaiņas tikai 8% pieprasījumu rēķinu kategorijā [16].

Tipiski soļi nodomu noteikšanas pielietojumam uzņēmējdarbībā:

1. Atrast visbiežākos pieprasījumu tipus;
2. Sagatavot atbildes veidni (*template*);
3. Nodomu noteikšanas sistēma identificē, vai lietotāja pieprasījums pieder iepriekš definētajiem tipiem un izdod potenciālo atbildi;
4. Klientu apkalpošanas speciālists izvērtē un koriģē atbildi pirms nosūtīšanas;
5. Automātiski uzlabot nodomu noteikšanas sistēmu, balstoties uz speciālista veiktajām korekcijām [16].

5. EKSPERIMENTU DIZAINS/METADOLOĢIJA/APRAKSTS

Daudzvalodu nodomu noteikšana ir lietotāja vaicājumu nolūka identificēšana dažādās valodās. Šajā sadaļā tiks dots ieskats trīs pieejās daudzvalodu nodomu noteikšanas modeļu apmācībai un testēšanai, tās pamatojot ar pieejamo teorētisko literatūru par šo tēmu:

1. apmācība vienā valodā, un testēšana tajā pašā valodā, piemēram, apmācība latviešu valodā, un testēšana arī latviešu valodā;

2. apmācība visās valodās kopā, testēšana vienā valodā, piemēram, apmācībā izmantojot datu kopu, kurā angļu, latviešu, krievu, igauņu, lietuviešu datu kopas ir apvienotas vienā, testēšana latviešu valodā;

3. apmācība angļu valodā, testēšana ne-angļu valodā.

Katrai no trijām pieejām ir savas priekšrocības un ierobežojumi, un pieejas izvēle ir atkarīga no konkrētajām uzdevuma prasībām. Apmācība un testēšana vienā un tajā pašā valodā var nodrošināt augstāku precizitāti, savukārt, apmācot visas valodas kopā, var izveidot vienu modeli vairākām valodām, taču dažās valodās var būt zemāka precizitāte. Apmācība angļu valodā un testēšana valodās, kas nav angļu valoda, ir noderīgas, ja ir ierobežoti resursi citām valodām un ir sagaidāms, ka modelis labi darbosies angļu valodā. Pieejas izvēlei jābūt balstītai uz uzdevuma īpašajām prasībām un resursiem.

5.1. Apmācība un testēšana vienā valodā

Modeļa apmācība un testēšana vienā un tajā pašā valodā ir piemērota, ja paredzams, ka nodomu noteikšanas modelis konkrētajā valodā darbosies ar labi precizitāti. Vairāki pētījumi ir parādījuši, ka apmācība un testēšana vienā un tajā pašā valodā var nodrošināt lielāku nodomu noteikšanas modeļu precizitāti.

Pētījumā uz XTREME datu kopas [17] tika parādīts, ka, apmācot mBERT modeli [18] ne-angļu valodas nodomu noteikšanai uz datiem tajā pašā valodā var sasniegt par 17–20% augstāku precizitāti, nekā apmācot uz datiem angļu valodā. Pētījumā aplūkotas 40 valodas no 12 saimēm, un secināts, ka rezultāti ir labāki indo-eiropiešu valodu saimei, tāpēc ka citām saimēm var pastāvēt tokenizācijas grūtības.

H. Li un citi [19] izmantoja savā pētījumā paštaisītu datu kopu ar izteikumiem sešās valodās (angļu, spāņu, franču, vācu, hindi un taju). Vidējā precizitāte ne-angļu valodām sasniedza 78% gadījumā, kad apmācība notika vienā valodā, 80% – gadījumā, kad apmācība notika visās valodās, un 66% – kad apmācība notika tikai angļu valodā. Tika parādīts, ka XLM modelis [8] uz šīs datu kopas ļauj sasniegt par 10–11% augstāku precizitāti, nekā XLU modelis [20].

Citā pētījumā autori izmantoja daudzvalodu (angļu, japāņu) modeli un parādīja, ka tas ir efektīvāks, ja tikai daļa no modeļa tiek izmantota abās valodās, bet otra daļa ir specifiska katrai valodai. Apmācot un testējot šo modeli vienā un tajā pašā valodā, nodomu klasifikācijas precizitāte uzlabojās par 0.5–2%, salīdzinot ar modeļa apmācību uz visām valodām [21]. Pētījumā izmantotā datu kopa sastāvēja no dialoga replikām un jautājumiem angļu un japāņu valodās [22].

Priekšrocības šādai pieejai ir iespēja ieviest nepārtrauktu attīstību (*continuous integration*), kurā ienākošie lietotāju ievades teksti un nodomi tiek izmantoti papildus apmācībai, izolējot efektus vienā valodā. Tomēr ļoti daudzās valodās datu kopas, ko var izmantot neironu tīkla apmācībā, ir salīdzinoši mazas, kā parādīts 1.2 att.. Izplatīta stratēģija nepietiekama apjoma treniņdatu problēmas risināšanai ir ievākt vairāk datu un apmācīt katru vienvalodu nodomu noteikšanas modeli atsevišķi, taču tas ir dārgi un resursietilpīgi. Bet izmantojot vienu daudzvalodu modeli (pieeja, kas aprakstīta nākamajā apakšnodaļā) zināšanas no liel-resursu valodas tiek pārnestas uz maz-resursu mērķvalodu [23].

5.2. Apmācība uz visām valodām, testēšana vienā valodā

Otrā pieeja ir modeļa apmācība daudzvalodu datu kopā, kas ietver visas interesējošās valodas, un tā testēšana konkrētā valodā. Šī pieeja ir noderīga, ja paredzams, ka modelis dažādās valodās darbosies pietiekami labi un mērķis ir izveidot vienu nodomu noteikšanas modeli, kas spēj apstrādāt vairākas valodas. Cilvēki sagaida precīzu mijiedarbību ar virtuālajiem asistentiem neatkarīgi no viņu lietotās valodas, tomēr mērogot (*scaling*) nodomu noteikšanu uz vairākām valodām ir izaicinājums. Tipisks daudzvalodu nodomu noteikšanas risinājums ir transformeru daudzvalodu modeļu izmantošana, piemēram, mBERT un XLM-RoBERTa. Atšķirībā no monolingvāliem modeļiem, daudzvalodu modeļi tiek apmācīti uz daudzvalodu datu kopām.

2022. gada pētījumā [24] tika parādīts, ka modeļu precizitāti var uzlabot, mākslīgi palielinot

treniņkopas izmēru. Tomēr treniņkopas palielināšana palielināja arī pārklājumu ar testa kopu, kas pārvērtēja (*overestimating*) patieso precizitāti. Izveidotais daudzvalodu modelis sasniedza 93.4% vidējo precizitāti nodomu noteikšanā uz MASSIVE datu kopas, kas satur paralēlus anotētus datus (angļu izteikumi ar tulkojumiem 51 valodā) [25].

Citā pētījumā tika secināts, ka pie nemainīgas modeļa arhitektūras palielinot apmācībā izmantoto valodu skaitu līdz pat 100 modeļa efektivitāte mazāk populārām valodām sākumā pieaug, bet pēc tam sāk samazināties. Tas tiek dēvēts par "daudzvalodības lāstu" (*curse of multilinguality*), ko var novērst, palielinot kopējo neironu skaitu. Šā pētījuma ietvaros salīdzinot mBERT un XLM-RoBERTa jēdzientelpu daudzvalodu klasifikāciju, tika secināts, ka XLM-RoBERTa ir precīzāks līdz pat 23% maz-resursu valodās (svahili un urdu) [8].

Vēl vienā pētījumā parādīts, ka, izmantojot kopīgu modeli angļu, hindi un bengali valodu mBERT jēdzientelpām precizitāte palielinās par $\sim 2\%$, salīdzinot ar atsevišķu apmācību katrā valodā, t.i., individuāliem nodomu noteikšanas modeļiem [26].

Šīs pieejas (apmācība uz visām valodām) priekšrocības ir mazākas modeļa apmācības izmaksas (viens modelis visiem datiem), kā arī maz-resursu valodas var gūt labumu no starp-valodu zināšanu pārneses (*cross-lingual knowledge transfer*), kas raksturīga kopīgam modelim [24]. Taču var parādīties vairākas negatīvas sekas, ja daudzvalodu jēdzientelpu modeli apmāca uz datu kopas, kurā kāda valoda, piemēram, angļu, ir disproporcionāli pārstāvēta (*over represented*) un testējot uz maz-resursu (*low-resource*) valodas, piemēram, latviešu. Pirmkārt, modelim var būt zemāka precizitāte latviešu valodā, kas nozīmē nepareizi klasificētus lietotāja nodomus un lietotāju neapmierinātību ar virtuālo asistentu. Otrkārt, modelis, kas apmācīts uz disbalansētas datu kopas var ciest no katastrofiskas aizmiršanas (*catastrophic interference*) fenomena, kurā modelis "aizmirst" maz-resursu valodu kad tiek iepazīstināts ar jauniem datiem citās valodās, kas noved pie zemas precizitātes un nepieciešamības apmācīt modeli no jauna.

5.3. Apmācība angļu valodā, testēšana valodās, kas nav angļu

Trešā pieeja ietver modeļa apmācību angļu valodā un tā testēšanu valodā, kas nav angļu valoda. Lasītājam var rasties šaubas, kā modelis kaut ko var paredzēt valodā, uz kuras nav apmācīts. Daudzvalodu modeļi tika apmācīti uz 100 dažādām valodām (precīzāk mBERT – 104 valodas,

XLM-Roberta – 100 valodas), kas iemācīja tiem izveidot jēdzientelpas dažādās valodās. Šie modeļi izmanto no daudzvalodu korpusa iemācīto kopējo reprezentāciju (*shared representations*), lai reprezentētu tekstu kā vektoru, kas tālāk tiek izmantots klasifikācijai. Būtībā modelis iemācās reprezentēt tekstu veidā, kas vispārina valodai raksturīgās nianses, tādējādi ļaujot tam strādāt dažādās valodās.

Daudzvalodu modeļi spēj sasniegt pietiekami labus rezultātus, jo ir "iemācījušies" uztvert valodas lietojumu plašā valodu diapazonā. Tomēr klasificējot ne-angļu tekstu var samazināties precizitāte, jo apmācībā modelim nebija pieejami valodai raksturīgie dati. Piemēram, pētījumā ar MASSIVE datu kopu [25] autori apmācīja daudzvalodu modeļus XLM-Roberta un mT5 uz ļoti lielas (1 miljons rindiņu) datu kopas. Veikti dažādi eksperimenti: gan apmācot tikai uz angļu valodas (šajā gadījumā rezultātu variance bija ļoti liela), gan uz visām valodām (iegūti par 25-37% labāki rezultāti, nekā apmācot tikai uz angļu valodas). Arī citā pētījumā [27] parādīts, ka, neatkarīgi no izmantotā modeļa (mBERT, XLM-R, mT5), apmācība uz visām valodām ļauj sasniegt par 1–5% augstāku teikumu klasifikācijas precizitāti, nekā apmācība tikai uz angļu valodas.

Šī pieeja ir noderīga, ja ne-angļu valodām ir ierobežoti treniņkopas resursi un ja paredzams, ka modelis labi darbosies angļu valodā. Tomēr šī pieeja paredz, ka valodu struktūra ir pietiekami līdzīga, lai modelis varētu pārnest zināšanas no angļu valodas uz citām valodām.

5.4. Mašīntulkošana uz angļu valodu

Mašīntulkošana izvēlēta lai apietu šķērsli kurā maz-resursu valodās ir mazāk datu uz kā apmācīt modeli. Ideja ir nevis gaidīt līdz tiks savākti pietiekami daudz datu, bet nodrošināt iespēju ienākt maz-resursu valodas lietotāju tirgū un sākt nodomu noteikšanu jau no pirmās dienas, lietotāju ievadus mašīntulkojot angļu valodā un izmantojot jau eksistējošos klasificēšanas modeļus angļu valodās.

Pieļaujamā nodomu noteikšanas precizitāte ir katra uzņēmuma biznesa plāna ziņā. Nav obligāti, lai tā sasniegtu 100%, jo ir iespējams lietot human-in-the-loop pieeju, kurā noteikto nodomu pārbauda klientu apkalpošanas speciālists vēl pirms lietotājam tiek nosūtīta atbilde [16]. Arī nepilna automatizācija ir noderīga, jo strādniekam novērtēt vaicājuma atbilstību konkrētam nodomam ir vieglāk, nekā izvērtēt kuram no daudziem nodomiem tas atbilst.

Taču mašīntulkošana rada papildu trokšņus un kļūdas, kas var ietekmēt ievades datu kvalitāti un klasifikācijas modeļu veiktspēju. Turklāt mašīntulkošanas rezultātā var tikt zaudētas svarīgas nianses un katrai valodai raksturīgā semantiskā informācija, kas var vēl vairāk pasliktināt ievades datu kvalitāti un apgrūtināt precīzu nodomu noteikšanu.

Mani darbā interesē tieši noskaidrot pielietojamības robežas maz-resursu valodām un kādi kompromisi (*trade-offs*) uzlabo modeļa precizitāti; šajā gadījumā vai modeļa veiktspēja ar mašīntulkošanas troksni atsver nepietiekamos datos. Tāpēc ir svarīgi novērtēt modeļu veiktspēju gan ar oriģinālajiem, gan mašīntulkotajiem ievades datiem un salīdzināt rezultātus, lai labāk izprastu modeļu stiprās puses un ierobežojumus daudzvalodu nodomu noteikšanas uzdevumos.

5.5. Citas pieejas

Visbeidzot, būtu svarīgi izpētīt, kā virtuālais asistents tiek galā ar jauktu valodu vaicājumiem (*code switching*), kas ir izplatīti daudzvalodu vidēs, piemēram, kombinācija latviešu-angļu. Tas ietver pieeju izpēti vairāku valodu identificēšanai un atdalīšanai vienā izteikumā, kā arī efektīvi pielietotas daudzvalodu jēdzientelpas. Piemēram, pētījumā datu kopa, kas sastāv tikai no jauktu angļu un hindi valodu vaicājumiem, bija ar 2% zemāku precizitāti (F1 score) nekā angļu, hindi un jauktu valodu datu kopa, ar ELMO jēdzientelpām [28].

Vēl viens jauktu valodu paņēmiens ir aizvietot izvēlētus vārdus ar to tulkojumiem maz-resursu valodām. Pētījumā [23] tika salīdzināta (a) modeļa apmācība tikai uz angļu valodas datu kopas un testēšana uz datiem spāņu valodā un (b) modeļa apmācība uz jauktiem angļu un spāņu valodu vaicājumiem. Izmainot pieeju no (a) uz (b), nodomu noteikšanas precizitāte uzlabojās no 73.7% uz 86.5% ar multilingvālām BERT jēdzientelpām un no 60.8% uz 83.9% ar XLM jēdzientelpām. Jaukti vaicājumi tika ģenerēti automātiski aizvietojot vārdus, kas izvēlēti balstoties uz uzmanības slāņa (*attention layer*) aprēķinātajiem rādītājiem (*scores*) uz angļu valodas modeļa, ar to tulkojumiem bilingvālā vārdnīcā [23].

6. REZULTĀTI

TODO: apraksti grafikiem un tabula līdz ko būs final version.

6.1. Chatbot datu kopa

6.1. tabula

Unikālo nodomu skaits ”chatbot” treniņkopā un testa kopā.

Nodoms	Treniņkopā	Testa kopā	Σ
FindConnection	57	71	128
DepartureTime	43	35	78
Σ	100	106	206

6.2. Askubuntu datu kopa

6.2. tabula

Unikālo nodomu skaits ”askubuntu” treniņkopā un testa kopā.

Nodoms	Treniņkopā	Testa kopā	Σ
Software Recommendation	17	40	57
Make Update	10	37	47
Shutdown Computer	13	14	27
Setup Printer	10	13	23
None	3	5	8
Σ	53	109	162

6.3. Webapps datu kopa

Webapps datu kopā ir nodoms ar tikai vienu piemēru, kas izraisa ”ValueError: The least populated class in y has only 1 member, which is too few. The minimum number of groups for any class

cannot be less than 2.” Tāpēc nodomi ar mazāk nekā trīs piemēriem tika apvienoti vienā nodomā ”Other”. Tas atbilst reālam pielietojumam industrijā, kur nodomi nav vienlīdzīgi pārstāvēti – piemēram, starp 115 dažādiem nodomiem divi visbiežākie nodomi kopā pārstāv 33% datu kopas [16] – un ir svarīgi spēt atsijāt nodomus, kurus jāapstrādā klientu apkalpošanas speciālistam – cilvēkam.

6.3. tabula

Unikālo nodomu skaits ”webapps” treniņkopā un testa kopā. Ar treknrakstu iezīmētas nodomi, kuri ir pietiekami pārstāvētas, pārējie nodomi tika apvienoti vienā jaunā nodomā: ”Other”

Nodoms	Treniņkopā	Testa kopā	Σ
Find Alternative	**7**	**16**	23
Delete Account	**7**	**10**	17
Filter Spam	**6**	**14**	20
Sync Accounts	**3**	**6**	9
Change Password	2	6	8
None	2	4	6
Export Data	2	3	5
Download Video	1	0	1
Σ	30	59	89

SECINĀJUMI

Testējot multilingual BERT un XLM-RoBERTa daudzvalodu jēdzientelpas lietotāju nodomu noteikšanā piecās dažādās valodās tika konstatēts, ka (modelim) bija visaugstākā precizitāte (valodās) valodās ar ()%. (Modelis) arī darbojās labi ar kopējo precizitāti ()%. Rezultāti liecina, ka daudzvalodu vārdu iegulšanas izmantošana var būt efektīva nodomu noteikšanai vairākās valodās, un modeļa izvēle var būtiski ietekmēt klasifikācijas uzdevuma precizitāti.

Modeļu precizitāte katrai valodai bija atšķirīga, ar zemāko precizitāti (valodā) valodā, un augstāko precizitāti (valodā) valodā, kas bija sagaidāms ņemot vērā datu kopas uz kurām tika apmācīti multilingual BERT un XLM-RoBERTa modeļi.

Kopumā paredzams, ka nolūku klasifikācijas modeļu precizitāte būs augstāka oriģinālajiem ievades datiem latviešu, igauņu, krievu un lietuviešu valodā, salīdzinot ar to pašu ievades datu mašīnu, kas tulkota angļu valodā. Tas ir tāpēc, ka mašīntulkošana rada papildu trokšņus un kļūdas, kas var ietekmēt ievades datu kvalitāti un klasifikācijas modeļu veiktspēju. Turklāt mašīntulkošanas rezultātā dažkārt var tikt zaudētas svarīgas nianses un katrai valodai raksturīgā semantiskā informācija, kas var vēl vairāk pasliktināt ievades datu kvalitāti un apgrūtināt precīzu nolūku klasificēšanu.

Tomēr precīza modeļu veiktspēja var atšķirties atkarībā no vairākiem faktoriem, piemēram, mašīntulkošanas kvalitātes, ievades vaicājumu sarežģītības un izmantoto daudzvalodu vārdu iegulšanas specifiskajām īpašībām. Tāpēc ir svarīgi novērtēt modeļu veiktspēju gan ar oriģinālajiem, gan mašīntulkotajiem ievades datiem un salīdzināt rezultātus, lai labāk izprastu modeļu stiprās puses un ierobežojumus daudzvalodu nolūku klasifikācijas uzdevumiem.

IZMANTOTĀ LITERATŪRA UN AVOTI

[1] Venkat N. Gudivada un Kamyar Arbabifard. „Chapter 3 - Open-Source Libraries, Application Frameworks, and Workflow Systems for NLP". *Computational Analysis and Understanding of Natural Languages: Principles, Methods and Applications*. Izdevis Venkat N. Gudivada un C.R. Rao. 38. sējums. Handbook of Statistics. Elsevier, 2018, 31.—50. lpp. DOI: https://doi.org/10.1016/bs.host.2018.07.007. URL: https://www.sciencedirect.com/science/article/pii/S0169716118300221.

[2] Charles T. Hemphill, John J. Godfrey un George R. Doddington. „The ATIS Spoken Language Systems Pilot Corpus". *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley*. 1990. URL: https://catalog.ldc.upenn.edu/docs/LDC93S4B/corpus.html.

[3] W. Nelson Francis un H Kucera. *Brown Corpus Manual: Manual of information to accompany a Standard Sample of Present-Day American English, for use with digital computers*. Brown University, 1964. URL: http://icame.uib.no/brown/bcm.html.

[4] Emily M. Bender u. c. „On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?": *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. FAccT '21. Virtual Event, Canada: Association for Computing Machinery, 2021, 610.—623. lpp. ISBN: 9781450383097. DOI: 10.1145/3442188.3445922. URL: https://doi.org/10.1145/3442188.3445922.

[5] Wikimedia Foundation. *Community Insights: Community Insights 2020 Report: Thriving Movement*. https://meta.wikimedia.org/wiki/Community_Insights/Community_Insights_2020_Report/Thriving_Movement#Community_and_Newcomer_Diversity. 2020.

[6] Brian C. Britt un Sorin Adam Matei. *Structural differentiation in social media: adhocracy, entropy, and the "1 % effect"*. Lecture notes in social networks. Springer, 2017. ISBN: 978-3-319-64425-7, 3319644254, 978-3-319-64424-0. URL: http://gen.lib.rus.ec/book/index.php?md5=416b9349cbf6cff824e540feb4228cb6.

[7] Google Ngram Viewer Team. *Google Books Ngram Viewer*. https://books.google.com/ngrams/. Aplūkots 2023-05-06.

[8] Alexis Conneau u. c. „Unsupervised Cross-lingual Representation Learning at Scale". *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, 2020. g. jūl., 8440.—8451. lpp. DOI: 10.18653/v1/2020.acl-main.747. URL: https://aclanthology.org/2020.acl-main.747.

[9] Pratap Dangeti. *Statistics for Machine Learning: Techniques for exploring supervised, unsupervised, and reinforcement learning models with Python and R*. Packt Publishing, 2017. ISBN: 9781788295758.

[10] Adrian Colyer. *The amazing power of word vectors*. 2016. URL: https://blog.acolyer.org/2016/04/21/the-amazing-power-of-word-vectors/.

[11] Allison Parrish. *Understanding word vectors*. 2017. URL: https://gist.github.com/aparrish/2f562e3737544cf29aaf1af30362f469.

[12] Tomás Mikolov u. c. „Efficient Estimation of Word Representations in Vector Space". (2013). arXiv: 1301.3781 [cs.CL]. URL: https://arxiv.org/abs/1301.3781v3.

[13] Chris McCormick. *Word2Vec Tutorial - The Skip-Gram Model*. 2016. URL: http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/.

[14] Kaspars Balodis un Daiga Deksne. „FastText-Based Intent Detection for Inflected Languages". *Information* 10.5:161 (2019). ISSN: 2078-2489. DOI: 10.3390/info10050161. URL: https://www.mdpi.com/2078-2489/10/5/161.

[15] Snips. *Snips Natural Language Understanding Documentation: Key Concepts & Data Model*. https://snips-nlu.readthedocs.io/en/latest/data_model.html#intent. Aplūkots 2023-05-10.

[16] Pēteris Paikens, Artūrs Znotiņš un Guntis Bārzdiņš. „Human-in-the-Loop Conversation Agent for Customer Service". *Natural Language Processing and Information Systems*. Izdevis Elisabeth Métais u. c. Cham: Springer International Publishing, 2020, 277.—284. lpp. ISBN: 978-3-030-51310-8. DOI: https://doi.org/10.1007/978-3-030-51310-8_25. URL: https://link.springer.com/chapter/10.1007/978-3-030-51310-8_25.

[17] Junjie Hu u. c. „XTREME: A Massively Multilingual Multi-task Benchmark for Evaluating Cross-lingual Generalization". *Proceedings of the 37th International Conference on Machine Learning*. 2020. URL: https://arxiv.org/abs/2003.11080.

[18] Jacob Devlin u. c. „BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". *NAACL-HLT 2019: Minneapolis, MN, USA*. 2019, 4171.—4186. lpp. URL: https://aclanthology.org/N19-1423.pdf.

[19] Haoran Li u. c. „MTOP: A Comprehensive Multilingual Task-Oriented Semantic Parsing Benchmark". *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. 2021, 2950.—2962. lpp. URL: https://aclanthology.org/2021.eacl-main.257.

[20] Sebastian Schuster u. c. „Cross-Lingual Transfer Learning for Multilingual Task Oriented Dialog". *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*. 2019, 3795.—3805. lpp. URL: https://doi.org/10.18653/v1/N19-1380.

[21] Ryo Masumura u. c. „Multi-task and Multi-lingual Joint Learning of Neural Lexical Utterance Classification based on Partially-shared Modeling". *Proceedings of the 27th International Conference on Computational Linguistics*. 2018, 1137.—1155. lpp. URL: https://aclanthology.org/C18-1304.pdf.

[22] Satoshi Sekine un Chikashi Nobata. „Definition, dictionaries and tagger for extended named entity hierarchy". *Proc. Language Resources and Evaluation Conference*. 2004.

[23] Zihan Liu u. c. „Attention-Informed Mixed-Language Training for Zero-Shot Cross-Lingual Task-Oriented Dialogue Systems". *Proceedings of the AAAI Conference on Artificial Intelligence* 34.05 (2020), 8433.—8440. lpp. DOI: 10.1609/aaai.v34i05.6362. URL: https://ojs.aaai.org/index.php/AAAI/article/view/6362.

[24] Maxime De bruyn u. c. „Machine Translation for Multilingual Intent Detection and Slots Filling". *Proceedings of the Massively Multilingual Natural Language Understanding Workshop (MMNLU-22)*. Abu Dhabi, United Arab Emirates (Hybrid): Association for Computatio-

nal Linguistics, 2022. g. dec., 69.—82. lpp. URL: https://aclanthology.org/2022.mmnlu-1.8.

[25] Jack FitzGerald u. c. „MASSIVE: A 1M-Example Multilingual Natural Language Understanding Dataset with 51 Typologically-Diverse Languages". *Arxiv.org* Jun (2022). URL: https://arxiv.org/abs/2204.08582.

[26] Mauajama Firdaus, Asif Ekbal un Erik Cambria. „Multitask learning for multilingual intent detection and slot filling in dialogue systems". *Information Fusion* 91 (2023), 299.—315. lpp. ISSN: 1566-2535. DOI: https://doi.org/10.1016/j.inffus.2022.09.029. URL: https://www.sciencedirect.com/science/article/pii/S1566253522001671.

[27] Linting Xue u. c. „mT5: A Massively Multilingual Pre-trained Text-to-Text Transformer". *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2021, 483.—498. lpp. URL: https://aclanthology.org/2021.naacl-main.41.

[28] Pratik Jayarao un Aman Srivastava. „Intent Detection for code-mix utterances in task oriented dialogue systems". *2018 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*. 2018, 583.—587. lpp. DOI: 10.1109/ICEECCOT43722.2018.9001577.

PIELIKUMS

Kods

Koda piemērs literatūras ievadā. Krāsu dati "xkcd.json" `https://github.com/dariusk/corpora/blob/master/data/colors/xkcd.json`.

Ideja un hex_to_int un closest funkcijas [11], pārējās pārrakstītas ātrdarbībai ar numpy.

```
import numpy as np
import json

def hex_to_int(s):
    s = s.lstrip("#")
    return int(s[:2], 16), int(s[2:4], 16), int(s[4:6], 16)



def distance(coord1, coord2):
  """Euclidean distance between two points
  """
  return np.sqrt(np.sum(np.subtract(coord1, coord2)**2))


def subtractv(coord1, coord2):
  """coord1 - coord2
  """
  return np.subtract(coord1, coord2)


def addv(coord1, coord2):
  """coord1 + coord2
  """
  return np.sum([coord1, coord2], axis=0)


def closest(space, coord, n=10):
  closest = []
  for key in sorted(space.keys(),
            key=lambda x: distance(coord, space[x]))[:n]:
```

```
    closest.append(key)
  return closest


color_data = json.loads(open("xkcd.json").read())

colors = dict()
for item in color_data['colors']:
    colors[item["color"]] = hex_to_int(item["hex"])
```