LATVIJAS UNIVERSITĀTE
DATORIKAS FAKULTĀTE

PĒC LOĢIKAS LĪDZĪGU TEIKUMU MEKLĒŠANA, IZMANTOJOT MAŠĪNMĀCĪŠANĀS METODES

MAĢISTRA DARBS

Autors: **Henrijs Smelēns**
Studenta apliecības Nr.: hs11009
Darba vadītāja: Dr. dat. Inguna Skadiņa

RĪGA 2018

ANOTĀCIJA

Mūsdienās ļoti strauji pieaug mašīnmācīšanās popularitāte, kas rezultējas ar dažādu metožu izveidi valodas apstrādes jomā, bet lielākā daļa no šīm metodēm tiek izstrādātas priekš angļu valodas. Darbā tika izvirzīts mērķis aplūkot un salīdzināt populārākās metodes, kas ļauj salīdzināt teikumus pēc to loģikas un pārbaudīt tās uz latviešu valoda, lai gūtu priekšstatu par to, kuras no tām ir efektīvākas.

Darba ietvaros tika pētītas mašīnmācīšanās dabiskās valodas apstrādes (NLP) metodes, kas ļauj iemācīt datoram saprast teikumu loģiku. Tika sagatavota apmācāmo datu kopa, kas satur teikumus latviešu valodā. Daļa no aplūkotajiem risinājumi tika izmēģināti, izmantojot sagatavoto apmācāmo datu kopu, un pēc tam novērtēti un salīdzināti savā starpā. Novērtēšana notika salīdzinot risinājumu apmācīšanas laikus, kā arī tie tika testēti ar iepriekš sagatavotu testa kopu, kas sastāv no dažādiem teikumu pāriem, kas ir vai nu pēc loģikas līdzīgi, atšķirīgi vai arī neitrāli savā starpā.

Atslēgvārdi: dabiskās valodas apstrāde, mašīnmācīšanās, teikumu līdzības.

ABSTRACT

Search for similar sentences by their logic using machine learning techniques

Nowadays, the popularity of machine learning is growing very rapidly, which leads to the development of various methods in the field of language processing, but most of these methods are developed for the English. The aim of the work was to examine and compare the most popular methods that make it possible to compare the sentences according to their logic and test them in the Latvian language in order to get an idea of which ones are more effective.

Within the framework of the work, machine learning natural language processing (NLP) techniques were studied, which allows the computer to understand the logic of sentences. Prepared the learner's data set that contains the sentences in the Latvian language. Some of the solutions discussed were tested using a trained dataset, and then evaluated and compared with each other. The assessment was made by comparing the solution training periods and were tested with a pre-made test set consisting of different pairs of sentences, which are either logically similar, different or neutral to each other.

Keywords: natural language processing, machine learning, sentence similarities.

AUTOREFERĀTS

Darba ietvaros tika aplūkotas dažādas dabiskās valodas apstrādes metodes un tās salīdzinātas savā starpā, pēc tam daļa no šim metodēm (Sent2Vec, Skip Thought un FastSent) tika izmēģinātas un ar tām apmācīti modeļi uz latviešu valodas tekstu. Apmācāmā teksta kopa tika sagatavota aplūkojot dažādus avotus internetā, kur ir brīvi pieejai latviešu valodas teksti.

Pēc modeļu apmācīšanas tie tika salīdzināti ar testa kopu, kas tika izveidota no teikumu pāriem, kuri bija savā starpā vai nu pretējas, līdzīgas vai neitrālas nozīmes. Pēc iegūtajiem datiem tika izdarīti secinājumi, kura no metodēm strādā viss efektīvāk uz attiecīgo latviešu valodas datu kopu.

Darbā izvirzītais mērķis novērtēt labāko metodi priekš latviešu valodas tika daļēji izpildīts, jo tika noskaidrots kura no izvēlētajām metodēm ir labākā uz konkrēto datu kopu, kas ir latviešu valodā.

Darba ietvarā ietvaros pārsvarā tika aplūkotas publikācijas, kurās tika prezentēti izmantotie dabiskās valodas apstrādes modeļi, kā arī to implementāciju apraksti.

Darbs ir izstrādāts balstoties uz Latvijas Universitātes Datorikas fakultātes maģistra darba izstrādes metodiskajiem norādījumiem. Darba izstrādes gaitā izmantotās literatūras saraksts pieejams darba beigās. Darba izstrādē tika izmantoti pareizrakstības rīki, kā arī darbs tika vairākas reizes caurskatīts, lai novērstu kļūdas.

SATURS

Ievads 8

Apzīmējumu saraksts 10

1. Dabiskā valodas apstrāde 11

2. Populārākās tekstu attēlošanas bibliotēkas 12

2.1. Facebook fastText 12

2.2. Paragraph vectors 13

2.2.1. Metodes apraksts 13

2.2.2. Priekšrocības 13

2.2.3. Trūkumi 13

2.4. Sent2Vec 14

2.4.1. Metodes apraksts 14

2.4.2. Priekšrocības 14

2.4.3. Trūkumi 14

2.5. Skip Thought 15

2.5.1. Metodes apraksts 15

2.5.2. Priekšrocības 15

2.5.3. Trūkumi 15

2.6. InferSent 16

2.6.1. Metodes apraksts 16

2.6.2. Priekšrocības 16

2.6.3. Trūkumi 16

2.7. FastSent 17

2.7.1. Metodes apraksts 17

2.7.2. Priekšrocības 17

2.7.3. Trūkumi 17

3. Apmācīšanas datu avoti 18

3.1. Vikipēdija 18
3.2. Twitter 19
3.3. OPUS 19
3.4. Korpuss.lv 20
3.5. LNB digitālā bibliotēka 20
4. NLP risinājumu izmēģināšana 21
4.1. Datu kopas sagatavošana 21
4.1.1. Teksta dalīšana tekstvienībās 21
4.1.2. Lielo burtu pārveidošana par mazajiem 22
4.1.3. Simbolu apstrāde 22
4.1.4. Kļūdu labošana 23
4.2. Testa kopas izveide 23
4.3. Darbstacijas sagatavošana 24
4.3.1. Darbstacijas specifikācijas 24
4.3.2. Darbstacijas programmatūra 24
4.4. Sent2Vec 25
4.4.1. Datu sagatavošana 25
4.4.2. Modeļa apmācīšana 25
4.4.3. Rezultāti 26
4.4.4. Rezultātu izvērtējums 27
4.5. Skip Thought 28
4.5.1. Datu sagatavošana 28
4.5.2. Modeļa apmācīšana 28
4.5.3. Rezultāti 29
4.5.4. Rezultātu izvērtējums 30
4.6. FastSent 32
4.6.1. Datu sagatavošana 32
4.6.2. Modeļa apmācīšana 32

4.6.3. Rezultāti 32
4.6.4. Rezultātu izvērtējums 34
4.7. Rezultātu salīdzināšana 35
4.7.1. Apmācīšanas ilgums 35
4.7.2. Iegūtie rezultāti 35
4.7.3. Secinājumi 36
5. Tālākā darbība 37
6. Rezultāti 38
7. Secinājumi 39
Izmantotā literatūra un avoti 40
Pielikumi 41
1. pielikums. Testu kopa ar līdzīgas nozīmes teikumu pāriem 41
2. pielikums. Testu kopa ar pretējas nozīmes teikumu pāriem 42
3. pielikums. Testu kopa ar neitrālas nozīmes teikumu pāriem 43

IEVADS

Lai arī datori sākotnēji tika veidoti, lai galvenokārt veiktu sarežģītus matemātiskus aprēķinus, mūsdienās tos plaši izmanto ļoti dažādu uzdevumu veikšanai. Viens no tiem ir valodas apstrāde. Valodas apstrāde sevī ietver ļoti dažādas lietas, piemēram, tekstu nozīmes saprašanu, runas atpazīšanu, tulkošanu, tekstu kategorizēšanu, kļūdu labošu, u.c..

Sākotnēji valodas apstrādei pārsvarā izmantotās metodes bija manuāla likumu iekodēšana, pēc kā dators arī veica savus darbus. Šāda metodes ir labas, bet ir ļoti sarežģīti definēt visus valodu likumus, un tas prasa ļoti lielu darbu. Kā arī tādus uzdevumus kā teksta kategorizēšana ir ļoti grūti paveikt ar šādām metodēm, jo ieejas dati, var būt ļoti dažādi, kas prasa ļoti daudz gadījumu aplūkošanu.

Mūsdienās attīstoties datoriem arvien populārākas kļūst mašīnmācīšanās metodes. Tās ļauj programmētājiem veidot modeļus, ko apmāca ar lielām datu kopām, tādējādi iemācot datoram veikt nepieciešamos uzdevumus. Tās ļauj nedefinēt ļoti specifiskus likumus, lai dators spētu izdarīt darāmo, bet paļaujas uz to, ka pareizi izveidots modelis spēs apmācīties, aplūkojot ļoti daudz piemērus, kas daudzos gadījumos tā arī ir, ko pierāda tas, ka ļoti daudzās jomās mašīnmācīšanās pārspēj klasiskās metodes. Kā arī ļoti bieži izmantotās mašīnmācīšanās metodes ir universālas. Piemēram, daudzas metodes, kas tiek izmantotas kādas valodas loģikas iemācīšanai, ir lietojamas dažādās valodās.

Mašīnmācīšanās ir kļuvusi arī ļoti populāra teksta apstrādē. Šīs metodes kā standarts jau ir kļuvis tādos uzdevumos, kā automātiska teksta klasificēšana, teksta loģikas saprašana, tulkošana, līdzīgu tekstu atpazīšana, kā arī daudzos citos uzdevumos.

Tā kā šīs metodes ir kļuvušas populāras tikai ļoti nesen, tad daudzu problēmu risinājumi tiek vēl joprojām aktīvi pētīti un veidoti. Kā vienu no šādām problēmām var minēt teikumu loģikas attēlošanu datorā. Lai gan ir jau lieli sasniegumi vārdu attēlošanā, sasniegumi teikumu un tekstu attēlošanā ir krietni mazāki.

Viena liela problēma, kas ir mašīnmācīšanās metodēm, ir tāda, ka, lai tās izmantotu ir nepieciešami ļoti lieli datu apjomi ar ko apmācīt datoru. Kā arī daudzas metodes paredz, ka šie dati ir uzraudzīti un kādā ziņā apzīmēti. Šī problēma ir īpaši aktuāla mazajām valodām, kurās nav publiski pieejami tik daudz resursu.

Līdz ar to, ka latviešu valodā arī ir nepieciešami rīki, lai apstrādātu tekstus automātiski, tad ir svarīgi izpētīt pašreiz efektīvākās metodes šajā jomā, un atrast labākos risinājumus tieši latviešu valodai. Darbā tiks meklēts labākais risinājums teikumu loģikas attēlošanai, aplūkojot vairākas populārākās metodes, kas tiek izmantotas citām valodām, kā arī apmācīti modeļi izmantojot tās. Apmācītie modeļi tiks salīdzināti savā starpā pēc to apmācīšanai prasītā laika,

kā arī pārbaudot, cik labi tie atpazīts teikumu loģikas līdzības, izmantojot sagatavotas testu datu kopas kas sastāv no teikumu pāriem, kas pēc loģikas ir vai nu līdzīgi, pretēji, vai arī neitrāli.

APZĪMĒJUMU SARAKSTS

BiLTSM – divvirzienu ilgas īslaicīgas atmiņas tīkls (Bi-directional LTSM network)

BOW – NLP modelis kurā teikums tiek attēlots ar tā vārdiem (bag of words)

C# – Microsoft izstrādāta programmēšanas valoda

GRU – RNN modelis (Gated Recurrent Unit)

GUI – grafiskā lietotāju saskarne

LNB – Latvijas Nacionālā bibliotēka

LTSM – ilgas īslaicīgas atmiņas tīkls, RNN modelis (Long Short Term Memory network)

NLI – dabiskās valodas saskarne

NLP – dabiskā valodas apstrāde (natural language processing)

RNN – atkārtotais neironu tīkls (recurrent neural network)

SNLI – Stenfordas dabiskās valodas saskarne

1. DABISKĀ VALODAS APSTRĀDE

Dabiskā valodas apstrāde (NLP) ir datorzinātnes joma, kas pēta iespējas ar datora palīdzību saprast un analizēt cilvēku valodu. Piemēram, dažādu tekstu tēmas saprašana, tulkošana, runas atpazīšanu, informācijas atrašanu tekstos, kā arī daudzas citas problēmas.

Pēdējos gados, līdz ar datoru jaudas un pieejamo datu pieaugumu, datorzinātnē arvien populārākās kļūst mašīnmācīšanās metodes. Izņēmums arī nav NLP, kur mašīnmācīšanās metodes dod ļoti labus rezultātus daudzās sfērās, piemēram, runas atpazīšanā, tekstu klasificēšanās, tulkošanā u.c.

Aplūkojot tekstu saprašanu, tad pašreiz populāri ir kļuvuši 2 virzieni. Viens no tiem ir dziļā mašīnmācīšanās, kurā tiek veidoti un apmācīti sarežģīti neironu tīkli, un otra izmanto vienkāršākas struktūras, ko apmācīt, piemērām kā teksta attēlošana vektoru telpā.

Dziļo neironu tīklu izmantošana ir ļoti sekmīga un tā uzrāda ļoti labus rezultātus, bet tās lielākais trūkums ir tāds, ka uz lieliem datu apjomiem tie ir ļoti ilgi apmācāmi, un to apmācīšana prasa lielus apstrādes resursus.

Turpretī vienkāršāku struktūru izmantošana dod iespēju ātrāk apmācīt modeļus izmantojot lielākus datu apjomus.

2. POPULĀRĀKĀS TEKSTU ATTĒLOŠANAS BIBLIOTĒKAS

Lai risinātu darbā izvirzīto problēmu, ir izveidotas vairākas bibliotēkas, kas uzrāda sekmīgus rādītājus, mācoties angļu valodā. Bet daudzas no tām apmācībai izmanto pārraudzītus datus, kas pie ievades satur sava veida pievienotu informāciju, kas ļauj algoritmam saprast, kādu papildu informāciju par datiem. Piemēram, tekstiem tā var būt tēma, lai pēc tam varētu to izmantot, lai saprastu kādā kategorijā ir teksts. Bet tādēļ, ka šādu datu sagatavošana prasa daudz manuāla darba, kas aizņem daudz laika, un latviešu valodā nav arī pieejami daudz sagatavoti dati, tad šajā nodaļā tiks aplūkotas populārākās metodes, neuzraudzītu teikumu apstrādei.

2.1. **Facebook fastText**

Viena no populārākajām atvērtā koda bibliotēkām priekš NLP ir Facebook fastText. To izstrādāja kompānija Facebook un tā ļauj lietotājiem risināt tādas problēmas, kā tekstu klasificēšana, kā arī vārdu attēlošana vektoru telpā.

Šī bibliotēka attēlo vārdu, izmantojot vektoru telpu un arī saglabā n-grammu virknes, lai varētu spriest par to, kādā secībā vārdi viens pēc otra atrodas. Šie dati par vārdiem tiek izmantoti, lai pēc tam analizētu tekstus un tos kategorizētu [1].

Kategorizēšanā fastText bibliotēka izmanto koka struktūru, lai glabātu kategorijas, kas ļauj uzlabot metodes ātrdarbību [1].

Viens no šīs metodes trūkumiem ir tas, ka tās apmācīšanai ir nepieciešami pārraudzīti dati, kur tekstiem jau ir noteikta kategorija, kurā tie ietilpst. Tas ir liels trūkums mazajām valodām, kāda ir latviešu valoda, tajās nav pieejami tik lieli resursi ar jau sagatavotiem apmācības datiem.

2.2. **Paragraph vectors**

Paragraph vectors modelis, jeb dēvēts arī par doc2vec un paragraph2vec ir izdomāts 2014. gadā. Tas ir balstīta uz word2vec metodes, un ir kā tās papildinājums. Metode izmanto tos pašus principus, kā word2vec, tikai paplašina to ideju, lai varētu arī vektoru telpā attēlot teikumus[2].

2.2.1. *Metodes apraksts*

Paragraph vectors modelis darbojās līdzīgi, kā word2vec, kas mēģina paredzēt nākamo vārdu sarakstā balstoties un iepriekšējiem vārdiem. Šajā modelī šai idejai papildus tiek pievienots tas, ka visi paragrāfi, kas var būt arī teikumi, tiek apzīmēti arī ar vektoriem un tiek ņemti vērā paredzot nākamos vārdus [2].

2.2.2. *Priekšrocības*

Priekšrocība šai metodei ir tāda, ka tā ņemt vērā vārdu secību teikumos. Vēl pie priekšrocībām var uzsvērt to, ka šī spēj darboties ne tikai ar teikumiem, bet gan ar dažāda garuma tekstiem.

2.2.3. *Trūkumi*

Pie trūkumiem var uzskaitīt to, ka metode neņem vērā kontekstu no blakus teikumiem. Dēļ tā, tās precizitāte uz izņēmuma gadījumiem, kur ir svarīgs konteksts no blakus teikumiem var būt mazāka salīdzinot ar metodēm, kas to ņem vērā.

2.4. **Sent2Vec**

Sent2Vec metode ir izveidota 2017. gada jūlijā, ņemot par pamatu iepriekš minēto Facebook fastText bibliotēku. Metode ir papildināta, pievienojot iespēju to apmācīt uz neuzraudzītiem datiem, tādējādi ļaujot iekodēt ne tikai vārdu nozīmi bet arī teikumu loģiku.

2.4.1. *Metodes apraksts*

Sent2Vec apmācīšana uz tekstiem notiek, izmantojot metodi, kurā teikumi tiek atspoguļoti, kā vidējā tā vārdu vērtība, kā arī tiek ņemta vērā vārdu secība izmantojot n-grammu virknes. [3]

2.4.2. *Priekšrocības*

Sent2Vec modeļa priekšrocība ir tāda, ka tā uzbūve ir vienkārša, salīdzinot ar modeļiem, kas izmanto neironu tīklus. Šī vienkāršība ļauj modeli apmācīt uz lielām datu kopām daudz ātrāk un ar mazākiem resursiem.

2.4.3. *Trūkumi*

Pie šīs metodes mīnusiem varētu pieskaitīt to, ka tā ņem vērā tikai konkrētā teikuma vārdus, un nemācās kontekstu no blakus esošajiem teikumiem. Tas rada problēmas, ja ir nepieciešams atrast līdzīgus teikumus, piemēram, idiomātiskiem izteicieniem, kas var pilnīgi atšķirties ar vārdu uzbūvi, bet būt līdzīgi pēc nozīmes.

2.5. Skip Thought

Skip Thought modelis ir izveidots 2015. gadā un tas teikumus attēlo ar vārdu vektoriem, kā arī izmanto dziļos neironu tīklus, ko apmāca uzminēt iepriekšējo un nākamo teikumu no dotā. Tā kā šai metodei ir nepieciešams uzminēt iepriekšējo un nākamo teikumu, tad apmācāmā datu kopa nevar sastāvēt tikai no nejauši izvēlētiem teikumiem, tiem ir jābūt loģiski saistītiem.

2.5.1. *Metodes apraksts*

Skip Thought metode izmanto kodētāju un dekodētāju, kas izmanto RNN un GRU. Kodētājs savieno teikuma vārdu vektorus ar teikuma vektoru, bet dekodētājs cenšas uzminēt blakus esošos teikumus, tādejādi gūstot priekšstatu par teikuma kontekstu.

Vārdu kodēšanai tiek izmantots word2vec modelis[4].

2.5.2. *Priekšrocības*

Šis metodes priekšrocība ir ka, ņemot vērā blakus teikumus, var tikt iemācītas sarežģītākas līdzības starp teikumiem, jo, piemēram, mēdz būt teikumi, kas sastāv no ļoti dažādiem vārdiem, bet kuru nozīmē ir ļoti līdzīga, piemēram, idiomātiski izteicieni. Šādas sakritības var piefiksēt, tika tad, ja tiek ņemts vērā teksta konteksts, ko var izdarīt tikai aplūkojot vairākus pēc kārtas sekojošus teikumus[4].

2.5.3. *Trūkumi*

Kā mīnusu šai metodei var minēt to, ka apmācīšanas laiks ir ilgs, jo tiek apmācīts neironu tīkls, kas patērē daudz resursus. Vēl kā mīnusu var minēt, ka lai šo metodi apmācītu ir nepieciešami secīgi teikumi, kas šajā gadījumā apgrūtina tādu avotu izmantošanu, kā Twitter, jo tur šie teksti ir ļoti īsi [4].

2.6. **InferSent**

InterSent 2017. gadā ir Facebook radīta teikumu loģikas attēlošanas metode, kas balstās uz uzraudzītiem datiem. [5]

2.6.1. *Metodes apraksts*

InterSent metode kā ievades datus izmanto pārraudzītus teikumu pārus, kas savā starpā ir apzīmēti ar vienu no 3 kategorijām:

- Implikācija
- Pretruna
- Neitrāli

Pēc tam katrs šis teikumu pāri tiek pārkodēts par vektoru, kam tiek izmantota BiLSTM neironu tīkla veids. Pēc tam šie vektori tiek apvienoti un klasificēti. [5]

Šajā metodē kā vārdu kodēšanas metode tiek izmantota GloVe metode, kas vārdus pārkodē par vektoriem. [5]

2.6.2. *Priekšrocības*

Pie metodes priekšrocībām var uzskatīt iegūstamos rezultātus, ja ir pieejami dati ar ko tos apmācīt, jo apmācot metodi uz SNLI datu kopas, kas sastāv no 570 tūkstošiem teikumu pāru, metodes izstrādātājiem bija izdevies iegūt vispārīgi labāku rezultātu nekā SkipThought metodei, kas apmācīta uz daudz lielāku datu apjomu. [5]

Vēl pie priekšrocībām var uzskatīt apmācīšanas ātrumu, kas ir ievērojami mazāks, jo ir nepieciešams apstrādāt mazāku datu apjomu, lai iegūtu labu rezultātu. Bet tas ir tikai iespējams, ja ir pieejama apmācāmo datu kopa. [5]

2.6.3. *Trūkumi*

Pie lielākā trūkuma var uzskaitīt to, ka šī metode strādā ar uzraudzītiem datiem, kas mazām valodām ļoti bieži nav publiski pieejami, un ko izveide prasa daudz laika.

2.7. FastSent

FastSent metode ir izveidota 2016. gadā. Tas tika veidots ņemot vērā Skip Thought modeļa priekšrocības un trūkumus, un cenšoties novērst trūkumus.

2.7.1. *Metodes apraksts*

FastSent metode līdzīgi kā Skip Thought metode apmācīšanas procesā izmanto teikumam esošos blakus teikumus, līdz ar to spējot mācīšanas procesā aplūkot teikuma kontekstu, ko metodes, kas neņem vērā blakus teikumus nespēj [6].

Skip Thought lielākais trūkums ir tas, ka lai to apmācītu ir nepieciešams ļoti liels laiks. Šajā modelī tas tiek novērsts attēlojot teikumus ar BOW metodi, kas neņem vērā vārdu secību teikumā. Šāda pieeja ļauj algoritmam strādāt ievērojami ātrāk [6].

2.7.2. *Priekšrocības*

FastSent metodes priekšrocības ir tādas, ka tā līdzīgi, kā Skip Thought metode, ņem vērā teikumam esošos blakus teikumus, tādejādi tas dod iespēju labāk saprast kontekstu kādā teikums tiek izmantots. Papildus šī metode neizmanto sarežģītus neironu tīklus, kas dod priekšrocību apmācīšanas laikā, kas ir ievērojami mazāks par Skip Thought metodi.

2.7.3. *Trūkumi*

Galvenais metodes trūkums ir tas, ka šajā metodē teikumi tiek attēloti, kā to vārdu summa, bet netiek ņemta vērā vārdu secība. Šāda lieta tik darīta, lai uzlabotu algoritma apmācīšanas ātrumu, bet mēdz būt gadījumi, kur vārdu secība teikumā ir svarīga, līdz ar to šādos gadījumos šis modelis strādās nepilnīgāk.

3. APMĀCĪŠANAS DATU AVOTI

Lai veiktu darbā paredzētos modeļu apmācīšanas darbus un to salīdzināšanu, ir nepieciešams iegūt teksta datu kopu ar ko šos uzdevumus veikt.

Lai gan pirmajā mirklī var šķist, ka internetā ir pieejami ļoti daudz tekstu visās valodās, ko varētu izmantot, lai veiktu uzdevumu, ļoti daudzus datus nevar izmantot, jo tie ir aizsargāti ar autortiesībām, vai arī nav ērta veida, kā šos datus iegūt no avotiem un apkopot.

Šajā nodaļā tiks aplūkoti dažādi datu teksta datu avoti, kā arī uzskaitīti to plusi un mīnusi priekš to izmantošanas darba izstrādei.

3.1. **Vikipēdija**

Viens no vieglāk pieejamajiem teksta datu avotiem latviešu valodā ir Vikipēdija. Tā sniedz iespēju lejuplādēt rakstu kopiju failus, kurus, pēc tam apstrādājot, ir iespējams iegūt tekstus nepieciešamajā formā [7].

Šis avots ir labs, jo tas satur daudz rakstu, un tie ir viegli pieejami.

Pie avota mīnusiem var uzskaitīt to, ka Vikipēdijā pieejamā informācija parasti ir zinātniska rakstura, tādēļ tajā pārsvarā tiek izmantota zinātniska valoda un specifiski termini. Kā arī raksti aptver ļoti daudz un dažādas tēmas, kas sagādā grūtības pa vienu tēmu atrast līdzīgus teikumus.

Vikipēdijas datu faili ir xml formātā, un šādi formatēti dati nav tieši izmantojami tālākām darbībām. Lai no šiem datiem iegūtu vienkāršu tekstu bez papildus informācijas vai formatējuma, tika izmantota wiki2text bibliotēka, kas ļāva no xml failiem iegūt vienkāršu tekstu [8].

3.2. Twitter

Mūsdienās par plaša teksta datu avotu var arī uzskaitīt sociālos tīklus, jo tajos cilvēki ik dienas ieraksta ļoti daudz informācijas, kā arī šī informācija ir publiski pieejama. Viens no populārākajiem šāda veida sociālajiem tīkliem ir Twitter.

Pie šī avota plusiem var pieskaitīt to, ka pieejamie dati atspoguļo sarunvalodu, kā arī dažādus dialogus, kas var rezultēties modeļa izveidē, kas saprot ikdienā lietotu valodu.

Bet par mīnusu var uzskatīt to, ka šis avots cilvēkiem ļauj publicēt tekstus ierobežotā garumā, kas ir 280 simboli, bet iepriekš bija 140 simboli. Šādi ierobežojumi rezultējas tajā, ka šie teksti tiek rakstīti ar dažādiem saīsinājumiem, lai varētu iekļaut visu nepieciešamo informāciju. Kā arī šie teksti bieži vien nav gramatiski pareizi.

3.3. OPUS

OPUS, jeb atvērtais paralēlais korpuss ir atvērta tekstu kolekcija, kas sastāv no tulkotiem tekstiem daudzās valodās. Šajā korpusā ir iespējams lejuplādēt tekstus dažādos formātos, gan failā kur ir apkopoti orģinālā un tulkotā valoda, gan arī atsevišķi, kur ir tikai viena valoda. Iespējams iegūt tekstus, kas ir neapstrādāti, gan arī tādus, kas ir jau sadalīti pa teksta vienībām, kā arī formatēti, piemēram xml formātā [9].

Šajā avotā pārsvarā ir apkopoti tulkoti teksti no dažādām sistēmām (Ubuntu, GNOME), kā arī teksti no iestādēm, piemēram Eiropas Parlamenta.

Mīnuss šiem tekstiem ir tāds, ka sistēmas teksti pārsvarā sastāv no vārdu salikumiem, vai īsiem tekstiem bez konteksta, kas neatbilst darbā nepieciešamajam mērķim. Kā arī teksti, kas satur dažādus noteikumus vai likumus, nav labākais variants priekš vienkārša valodas modeļa apmācīšanas, jo tie satur dažādus specifiskus vārdus, kas ikdienā bieži netiek izmantoti, kā arī šo teikumu struktūra ir visai sarežģīta.

No šī avota tika izvēlēta “Europarl” tekstu kopa, kas satur sarunu tekstus no Eiropas Parlamenta. Līdz ar to, ka šie teksti satur sarunvalodu, tad tie nav tik sarežģīti, kā likumi un ir labāk pielietojami, darba mērķim.

3.4. **Korpuss.lv**

Korpuss.lv ir LU LII Mākslīgā intelekta laboratorijas izveidots korpuss, kas satur latviešu valodas tekstus, kas paredzēti valodas analīzei. Šajā vietnē ir pieejama "Latviešu teksti datortīklā" elektroniskā bibliotēka, kas satur dažādus latviešu daiļliteratūras darbus [10].

Šī avota pluss ir ka tas satur dažādus latviešu daiļliteratūras darbus, kas parasti sastāv no vienkāršiem teikumiem, kas ir labi priekš darba mērķa.

Pie mīnusa var uzskaitīt to, ka šie teksti ir skenēti no grāmatām un automātiski pārveidoti digitālā formātā, līdz ar to tekstos ir drukas kļūdas.

3.5. **LNB digitālā bibliotēka**

Latvijas Nacionālā bibliotēka ir izveidojusi digitālu tekstu krātuvi, kur ir pieejami dažādi latviešu literatūras darbi.

Šī avota pluss ir tāds, ka tajā ir pieejams liels darbu apjoms.

Mīnusi ir tāds, ka šie darbi ir digitāli ieskenēti un ar rīka palīdzību teksts tiek pārveidots digitālā formātā, līdz ar to, tas var saturēt dažādas kļūdas. Vēl pie mīnusiem uzskaitāms, tas, ka grāmatu saturs nav lejuplādējams apkopoti teksta formātā, bet tekstus ir iespējams tikai aplūkot pa lapaspusei, kas sarežģi tekstu apkopošanas procesu.

4. NLP RISINĀJUMU IZMĒĢINĀŠANA

Lai varētu salīdzināt iepriekš minētos risinājumus, ir nepieciešams tos izmēģināt latviešu valodai. Lai izmēģinātu metodes darbs tika iedalīts 4 posmos:

- Datu kopas sagatavošana;
- Testa kopas izveide;
- Darbstacijas sagatavošana;
- Metošu apmācīšana;
- Rezultātu salīdzināšana;
- Secinājumu veikšana.

4.1. Datu kopas sagatavošana

Dažādiem risinājumiem ir nepieciešamas dažādi formatētas datu kopas, kas atbilst katras metodes specifikai. Bet ir arī datu apstrāde, kas nepieciešama vairākām metodēm vienāda. Uz datu kopām tika veiktas sekojoša lietas:

- Teksta dalīšana tekstvienībās;
- Lielo burtu pārveidošana par mazajiem;
- Simbolu apstrāde;
- Kļūdu labošana.

4.1.1. *Teksta dalīšana tekstvienībās*

Lai apmācītu metodes uz teikumu loģikas atpazīšanu, datu kopai ir nepieciešams būt sadalītai pa teikumiem un vārdiem, ko sauc arī par dalīšanu pa tekstvienībām (tokenization). Mazām datu kopām to ir iespējams paveikt manuāli, bet lielām tas prasa daudz laika. Viens no veidiem, kā to paveikt automātiski, ir dalīt teikumus tur, kur atrodas punkta simbols, bet šī metode nestrādā labi, jo punkts tiek lietots arī citur, piemēram, datumos, kārtas skaitļos utt. Vārdus turpretī varētu dalīt pēc atstarpēm, bet šāda metode arī nestrādā pilnīgi labi, jo teikumos pie vārdiem var būt klāt, piemēram, komati. Šādas specifiskas var būt katrai valodai savas, tādēļ tika atrasta C# bibliotēka “Latvian” [11], kas šo problēmu palīdzēja risināt un bija radīta tieši latviešu valodai. Izmantojot šo bibliotēku tika izveidots kods, kas sadalīja tekstu pa teikumiem un vārdiem.

4.1.2. *Lielo burtu pārveidošana par mazajiem*

Lielo burtu pārvēršana par mazajiem tika darīta, jo tekstā vārdiem ar lielajiem burtiem un mazajiem, lielākajā daļā gadījumu ir vienāda nozīme, un tā nemainās, ja visi vārdi pārveidoti uz mazajiem burtiem. Bet, ja tiek atstāti vārdi ar lielajiem burtiem, tad tie tiek uztverti kā atšķirīgi vārdi un apgrūtina modeļu apmācīšanas procesu. Šis uzdevums tika risināts izmantojot Vim programmu un regulāro izteiksmi.

4.1.3. *Simbolu apstrāde*

Teikumos loģika pārsvarā slēpjas vārdos, no kā tie sastāv, kā arī kontekstā. Bet tekstos bieži vien ir atrodami dažādi simboli, kuru nozīme ir mazāka loģikas attēlošanai un kuri sarežģi apmācīšanas procesu un kurus ir nepieciešams izņemt no teksta, lai uzlabotu modeļu apmācīšanas ātrumu un kvalitāti.
Šajā gadījumā no teksta tika atlasīti sekojoši simboli:

- Pēdiņas – visa veida pēdiņas tika izņemtas no teksta, jo tās pārsvarā attēloja tiešās runas sākumu un beigas, vai arī ietvēra kādus nosaukumus;
- @ simboli – tekstā bija sastopami vairāki šādi simboli;
- Domu zīmes teikumu sākumos – tās bija sastopamas pie dažādiem uzskaitījumiem.

Izmēģinājumos iegūtā datu kopa sastāvēja no Vikipēdijas iegūtajiem datiem, kas bija apmēram 1 miljons teikumu liela, latviešu literatūras tekstiem, un Eiropas Parlamenta tekstiem. Visi teksti kopā veidoja ap 1800000 teikumu lielu kopu.

4.1.4. *Kļūdu labošana*

Pēc iepriekš minētajām darbībām caurskatot tekstu, bija ievērojamas vairākas nepilnības, kuras vajadzēja maksimāli novērst. Lielākā daļa no tām tika labotas, izmantojot regulārās izteiksmes, meklējot tās tekstā un novēršot.

Tās bija:

- Saīsinājumi un daži gada skaitļi tika uzskatīti pa teikuma beigām un pārdalīti uz jaunām rindiņām;
- Vairākas sekojošas atstarpes;
- Vārdi, kuru burti atdalīti ar atstarpēm, kas rezultējās pēc kārtas sekojošos burtos, kas atdalīti ar atstarpēm;
- Daudzpunktu nomaiņa uz vienu punktu;
- Tukšo rindiņu izdzēšana.

Lai arī tika veiktas daudzas darbības, lai uzlabotu datu kvalitāti, tik un tā, tekstā vēl ir atrodamas kādas nepilnības.

4.2. **Testa kopas izveide**

Lai salīdzinātu dažāda metodes, ir nepieciešams izveidot testu, kas pārbauda kura no metodēm labāk izprot teikumu loģiku.

Lai pārbaudītu modeļus tika izveidots tests, kas sastāv no teikumu pāriem. Teikumu pāri ir paredzēti, lai metodes noteiktu cik tie ir līdzīgi viens otram. Līdz ar to, kad ir zināms, vai šis pāris ir līdzīgs pēc nozīmes, vai pretējas nozīmes, vai arī teikumi ir neitrāli, tad ir iespējams spriest par to, vai apmācītais modelis pareizi attēlo teikumus un salīdzināt metodes vienu ar otru.

Tika sagatavotas sekojoši teikumu pāri:

- 30 teikumu pāri, kuri pēc loģikas ir ļoti līdzīgi, bet izteikti ar citiem vārdiem;
- 10 teikumi, kuri pēc nozīmes ir pretēji, respektīvi, viens teikums noliedz otru;
- 10 teikumu pāri, kuri savā starpā pēc nozīmes ir neitrāli.

4.3. Darbstacijas sagatavošana

Lai veiktu darbā paredzēto modeļu apmācīšanu bija nepieciešams uzstādīt darbstaciju ar pietiekošiem datu apstrādes resursiem. Apmācot neironu tīklus tiek veiktas kalkulācijas, kuras iespējams veikt daudz ātrāk uz grafiskā procesora, kā parastā, tādēļ izmantotajai darbstacijai bija nepieciešama iespējami labāka video karte. Papildus arī bija nepieciešams uzstādīt atbilstošu programmatūru un draiverus, lai būtu iespējams veikt darbā paredzētos uzdevumus.

4.3.1. *Darbstacijas specifikācijas*

Lai veiktu darbu tika nokomplektēta darbstacija ar sekojošiem parametriem :

- Procesors: Intel Core i7-8700k 3.7GHz 6 kodoli, 12 pavedieni
- Pamatplate: Asus ROG STRIX Z-370-E GAMING ATX
- Operatīvā atmiņa: G.Skill Trident Z 32GB (2 x 16GB) DDR4-3200 CL14
- Video karte: Asus GeForce GTX 1080 Ti 11GB STRIX GAMING
- Atmiņa: Samsung 960 EVO 500GB M.2-2280 Solid State Drive
- Operētājsistēma: Windows 10 64bit

4.3.2. *Darbstacijas programmatūra*

Aplūkojot dažādas metodes, ko darbā bija paredzēts izmēģināt tika ievērots, ka daļa no tām ir izstrādātas priekš Linux operētājsistēmas. Tā kā uz datora pamata operētājsistēma ir Windows 10, tad tika izlemts uzstādīt virtuālo mašīnu. Tas tika paveikts izmantojot windows piedāvāto Hyper-V virtuālo mašīnu. Kā Linux Operētājsistēma tika izvēlēta Ubuntu 17.10 64bit.

Daudzas no aplūkotajām metodēm bija veidotas izmantojot Python programmēšanas valodu. Līdz ar to bija nepieciešams uzstādīt uz ierīces to. Kā koda labošanas lietotne tika izvēlēta JetBrains piedāvātā PyCharm.

Zinot, ka dažādām metodēm var būt nepieciešami dažādi uzstādījumi un bibliotēkas, tad tika izmantota Ananaconda pakešu pārvaldnieks.

4.4. Sent2Vec

Sent2Vec bija pirmā metode, kas tika izmēģināta. Kā pirmā tā tika izvēlēta, jo tā bija viena no jaunākajām, kas tika atrasta, kā arī tā neizmantoja dziļos neironu tīklus, kuru apmācīšana prasītu ievērojami lielāku laiku.

4.4.1. *Datu sagatavošana*

Priekš Sent2Vec risinājuma bija nepieciešams sagatavot datus tā, ka katrs teksta teikums ir jaunā rindiņā, un katrs vārds ir atdalīts ar atstarpi.

4.4.2. *Modeļa apmācīšana*

Šī metode tika darbināta uz Hyper-V virtuālās mašīnas ar Ubuntu operētājsistēmu, jo, darbinot uz Windows operētājsistēmas radās vairākas problēmas ar draiveru instalēšanu. Tā kā šī metode nebija resursu prasīga, tad virtuālās mašīnas izmantošana daudz neietekmēja darba ātrumu. Modeļa apmācīšana sagatavotajai datu kopai norisinājās apmēram 20 minūtes[3]. Modelis tika apmācītas ar sekojošiem iestatījumiem:

- minCount = 8
- dim = 700
- epoch = 9
- lr = 0.2
- wordNgrams = 2
- loss = ns
- neg = 10
- thread = 20
- t = 0.000005
- dropoutK = 4
- minCountLabel = 20
- bucket = 4000000

4.4.3. *Rezultāti*

Apmācītais modelis tika pārbaudīts uz sagatavotajām testa kopām un tika iegūti sekojoši rezultāti. Rezultāti no testa kopas ar līdzīgas nozīmes teikumu pāriem redzami tabulā 4.1. Teikumu pāru atšifrējumi ir redzami 1. pielikumā.

4.1. tabula

Sent2Vec rezultāti uz līdzīgas nozīmes testu kopu

Nr.p.k.	Sent2Vec	Nr.p.k.	Sent2Vec	Nr.p.k.	Sent2Vec
1	0.11795	**11**	0.43396	**21**	0.43104
2	0.30983	**12**	0.09261	**22**	0.58827
3	0.57579	**13**	0.47574	**23**	0.40822
4	0.58681	**14**	0.19970	**24**	0.55044
5	0.14260	**15**	0.32796	**25**	0.61458
6	0.03878	**16**	0.20270	**26**	0.49153
7	0.42267	**17**	0.04628	**27**	0.35819
8	0.26018	**18**	0.57809	**28**	0.51147
9	0.43547	**19**	0.58263	**29**	0.53587
10	0.37582	**20**	0.26738	**30**	0.63576

Rezultāti testa kopas ar pretējas nozīmes teikumu pāriem redzami tabulā 4.2. Teikumu pāru atšifrējumi ir redzami 2. pielikumā.

4.2. tabula

Sent2Vec rezultāti uz pretējas nozīmes testu kopu

Nr.p.k.	Sent2Vec
1	0.98446
2	0.55883
3	0.58818
4	0.07711
5	0.54190
6	0.48494
7	0.12384
8	0.48651
9	0.01396
10	0.43147

Rezultāti testa kopas ar neitrālas nozīmes teikumu pāriem redzami tabulā 4.3. Teikumu pāru atšifrējumi ir redzami 3. pielikumā.

4.3. tabula

Sent2Vec rezultāti uz neitrālas nozīmes testu kopu

Nr.p.k.	Sent2Vec
1	0.09495
2	0.06577
3	0.14900
4	0.08052
5	0.10687
6	0.20154
7	0.00422
8	0.15288
9	0.05754
10	0.09978

Iegūtajos rezultātos, jo skaitlis ir tuvāk 1 vērtībai, jo modelis uzskata, ka teikumi ir līdzīgāki.

4.4.4. *Rezultātu izvērtējums*

Aplūkojot rezultātus var ievērot, ka vidējā līdzīgu teikumu pāru vērtība (0.38661) ir lielāka, par vidējo vērtību starp neitrāliem teikumiem (0.10130). No tā var secināt, ka šim modelim ir izdevies iemācīties kādas valoda loģiku.

Ja aplūko pretējo teikumu testa vērtības, tad var ievērot, ka to vidējā vērtība ir viss augstākā (0.42912). Tas varētu būt skaidrojams ar to, ka šie teikumi sastāvēja no līdzīgiem vārdiem, un modelis neuzrādīja labus rezultātus un nolieguma piefiksēšanu.

Aplūkojot cik no teikuma pāriem no līdzīgo teikumu testu kopas vērtība ir lielāka par vidējo neitrālo teikumu pāru vērtību, var ievērot, ka tie ir 27 pāri no 30.

Ja to pašu aplūko uz pretējajiem rezultātiem, tikai kuru vērtība ir mazāka, tad rezultāts ir 2 no 10.

4.5. Skip Thought

Skip Thought metode ir angļu valodās sasniedz vienus no labākajiem rezultātiem [12] dažādos NLP uzdevumos, tādēļ tā arī ir populāra. Bet tā izmanto neironu tīklus, līdz ar to šī modeļa apmācīšana prasīja daudz laika.

4.5.1. *Datu sagatavošana*

Lai apmācītu Skip Thought modeli bija nepieciešams sagatavot datus tā, ka teikumi ir sakārtoti grupās pa 3 teikumiem, kas ir secīgi viens aiz otra tekstā. Tas ir nepieciešams, jo modeļa apmācīšana notiek cenšoties uzminēt teikumam iepriekšējo un nākam teikumu. Teikumu sadalīšana notika automātiski izmantojot Tensorflow piedāvātās skip thought bibliotēkas funkciju [13].

4.5.2. *Modeļa apmācīšana*

Lai arī šai metodei ir pieejamas vairākas implementācijas, tika izvēlēta Tensorflow vidē pieejamā [13]. Šāda izvēle tika veikta, jo zinot, ka modeļa apmācīšana būs relatīvi ilga, kā arī procesā būs nepieciešams, lai apmācīšana notiek izmantojot datora grafisko procesoru, tika izvēlēta vide, kurai bija zināms, ka darbojās nepieciešamie draiveri, tādejādi samazinot risku, ka kaut kas var neiet.

Modelis tika apmācīts izmantojot sekojošu konfigurāciju:

- input_queue_capacity=640000,
- num_input_reader_threads=1,
- shuffle_input_data=True,
- uniform_init_scale=0.1,
- vocab_size=20000,
- batch_size=128,
- word_embedding_dim=620,
- bidirectional_encoder=False,
- encoder_dim=2400
- learning_rate=0.0008,
- learning_rate_decay_factor=0.5,
- learning_rate_decay_steps=400000,
- number_of_steps=500000,

- clip_gradient_norm=5.0,
- save_model_secs=600,
- save_summaries_secs=600

Modeļa apmācīšana notika apmēram 30 stundas ilgi.

4.5.3. *Rezultāti*

Apmācītais modelis tika pārbaudīts uz sagatavotajām testa kopām un tika iegūti sekojoši rezultāti. Rezultāti no testa kopas ar līdzīgas nozīmes teikumu pāriem redzami tabulā 4.4. Teikumu pāru atšifrējumi ir redzami 1. pielikumā.

4.4. tabula

Skip Thought rezultāti uz līdzīgas nozīmes testu kopu

Nr.p.k.	Skip Thought	Nr.p.k.	Skip Thought	Nr.p.k.	Skip Thought
1	0.15349	**11**	0.17315	**21**	0.16712
2	0.11229	**12**	0.00515	**22**	0.22263
3	0.20704	**13**	0.15006	**23**	0.23890
4	0.42449	**14**	0.31948	**24**	0.24994
5	0.10452	**15**	0.13710	**25**	0.35913
6	-0.00337	**16**	0.18028	**26**	0.27755
7	0.08366	**17**	0.15012	**27**	0.12827
8	0.14301	**18**	0.55997	**28**	0.12530
9	0.15774	**19**	0.84443	**29**	0.18800
10	0.15035	**20**	0.14201	**30**	0.12374

Rezultāti testa kopas ar pretējas nozīmes teikumu pāriem redzami tabulā 4.5. Teikumu pāru atšifrējumi ir redzami 2. pielikumā.

4.5. tabula

Skip Thought rezultāti uz pretējas nozīmes testu kopu

Nr.p.k.	Skip Thought
1	0.72146
2	0.24188
3	0.26571
4	0.06400
5	0.17396
6	0.13611
7	0.03362
8	0.62423
9	0.00651
10	0.24477

Rezultāti testa kopas ar neitrālas nozīmes teikumu pāriem redzami tabulā 4.6. Teikumu pāru atšifrējumi ir redzami 3. pielikumā.

4.6. tabula

Skip Thought rezultāti uz neitrālas nozīmes testu kopu

Nr.p.k.	Skip Thought
1	0.05787
2	0.49768
3	-0.02508
4	0.00910
5	0.00716
6	0.05854
7	0.04330
8	-0.01385
9	0.05569
10	0.05763

Iegūtajos rezultātos, jo skaitlis ir tuvāk 1 vērtībai, jo modelis uzskata, ka teikumi ir līdzīgāki.

4.5.4. *Rezultātu izvērtējums*

Līdzīgi kā Sent2Vec modeļa rezultātiem, arī šajos var ievērot to, ka vidējais neitrālo teikumu pāru rezultāts (0.07480) ir mazāks par līdzīgo teikumu pāru vidējo rezultātu (0.20919), kas liecina, ka modelis kaut nedaudz atpazīst līdzīgus teikumus.

Ja paskatās uz pretējo teikumu rezultātiem, tas arī šajā gadījumā, tajos neattēlojas noliegumi, un to vidējā vērtība ir viss augstākā (0.25123), jo tajos ir līdzīgi vārdi.

Aplūkojot cik no līdzīgo teikumu vērtībām ir lielākas par neitrālo teikumu vidējo vērtību, tad rezultāts ir 28 no 30 teikumu pāros. Kas liecina, ka lielākajā daļā teikumu kāda līdzība ir atrasta.

Līdzīgi aplūkojot šo vērtību uz pretējās nozīmes teikumu pāriem, tikai skatoties kuriem tā ir mazāka pa neitrālo teikumu vidējo vērtību, var ievērot, ka mazāka tā ir tikai 3 pāriem no 10.

4.6. FastSent

FastSent metodei netika atrasta implementācija iekš Tensorflow, tādēļ tika izvēlēta implementācijas versija, kurai bija viegli saprotama dokumentācija. Šī metode darbojās uz Linux operētājsistēmas.

4.6.1. *Datu sagatavošana*

Priekš FastSent risinājuma bija nepieciešams sagatavot datus tā, ka katrs teksta teikums ir jaunā rindiņā, un katrs vārds ir atdalīts ar atstarpi, līdzīgi kā tas bija priekš Sent2Vec metodē.

4.6.2. *Modeļa apmācīšana*

Šīs metodes apmācīšana notika izmantojot Linux operētājsistēmu (Ubuntu), kas tika darbināta uz Hyper-V virtuālās mašīnas. Vispirms tika lejuplādēta bibliotēka SentenceRepresentation, kas saturēja šī modeļa implementāciju. Bibliotēka ir pieejama iekš github repozitorija [14].

4.6.3. *Rezultāti*

Apmācītais modelis tika pārbaudīts uz sagatavotajām testa kopām un tika iegūti sekojoši rezultāti. Rezultāti no testa kopas ar līdzīgas nozīmes teikumu pāriem redzami tabulā 4.7. Teikumu pāru atšifrējumi ir redzami 1. pielikumā.

4.7. tabula

FastSent rezultāti uz līdzīgas nozīmes testu kopu

Nr.p.k.	FastSent	Nr.p.k.	FastSent	Nr.p.k.	FastSent
1	0.94575	**11**	-0.12671	**21**	0.89359
2	-0.60949	**12**	0.49051	**22**	0.91050
3	-0.46239	**13**	0.93218	**23**	0.85602
4	0.86271	**14**	-0.85064	**24**	0.75052
5	0.975048	**15**	-0.19399	**25**	-0.36326
6	-0.91182	**16**	0.96142	**26**	-0.44106
7	0.47480	**17**	-0.60427	**27**	-0.96631
8	-0.14153	**18**	-0.98719	**28**	-0.52337
9	-0.10067	**19**	0.99587	**29**	-0.83354
10	-0.91755	**20**	-0.54526	**30**	-0.84863

Rezultāti testa kopas ar pretējas nozīmes teikumu pāriem redzami tabulā 4.8. Teikumu pāru atšifrējumi ir redzami 2. pielikumā.

4.8. tabula

FastSent rezultāti uz pretējas nozīmes testu kopu

Nr.p.k.	FastSent
1	-0.39289
2	0.98380
3	-0.38241
4	0.61392
5	0.94973
6	-0.88440
7	0.84906
8	-0.57387
9	0.16047
10	0.14391

Rezultāti testa kopas ar neitrālas nozīmes teikumu pāriem redzami tabulā 4.9. Teikumu pāru atšifrējumi ir redzami 3. pielikumā.

4.9. tabula

FastSent rezultāti uz neitrālas nozīmes testu kopu

Nr.p.k.	FastSent
1	0.72260
2	0.11624
3	0.35752
4	-0.68377
5	0.18596
6	-0.51708
7	-0.98309
8	-0.97690
9	0.61986
10	-0.39262

Iegūtajos rezultātos, jo skaitlis ir tuvāk 1 vērtībai, jo modelis uzskata, ka teikumi ir līdzīgāki.

4.6.4. *Rezultātu izvērtējums*

Aplūkojot vidējo neitrālo pāru rezultātu un salīdzinot to ar vidējo līdzīgo teikumu rezultātu var ievērot, ka līdzīgo teikumu vērtība ir vidēji lielāka, par neitrālajiem. Turpretī Pretējo teikumu vidējā vērtība ir vēl lielāka, kas liecina, ka šis modelis noliegumus labi neattēlo.

Aplūkojot cik no līdzīgo teikumu pāru rezultātiem ir lielāki par vidējo neitrālo teikumu rezultātiem, var ievērot, ka tā ir apmēram puse, jeb 16 no 30. Līdzīga situācija ir arī ar pretējās nozīmes teikumiem, respektīvi 4 pāriem no 10 vērtība ir mazāka par vidējo neitrālo teikumu vērtības.

No šiem iegūtajiem rezultātiem var spriest, ka šis modelis nav labi apmācījies un neattēlo teikumu loģisko līdzību labi, jo apmēram tikai puses no rezultātiem ir pareiza, kas var būt arī nejaušība.

4.7. Rezultātu salīdzināšana

Lai saprastu kura no pārbaudītajām metodēm vis labāk atbilst darbā izvirzītajam mērķim – līdzīgu teikumu pēc nozīmes meklēšana, tad ir nepieciešams salīdzināt iegūtos rezultātus.

4.7.1. *Apmācīšanas ilgums*

Viss īsākais apmācīšanas ilgums bija Sent2Vec metodei, kuras apmācīšana prasīja apmēram 20 minūtes, otrajā vietā bija FastSent metode, kuras apmācīšana prasīja apmēram 30 minūtes, turpretī Skip Thought metodes apmācīšanas laiks bija ievērojami ilgāks, respektīvi apmēram 30 stundas.

4.7.2. *Iegūtie rezultāti*

Aplūkojot iegūtos rezultātus var ievētos, ka Sent2Vec un Skip Thought metodes uzrāda ļoti līdzīgus rezultātus. Tur pretī FastSent šajā eksperimentā uzrādīja ievērojami sliktāku rezultātu.

Ja tiek aplūkots cik daudz teikumu pārus katra metode atpazina pareizi, skatoties un pāru ar līdzīgu un pretēju nozīmi vērtībām pret vidējo neitrālo teikumu vērtību, tad redzam, ka viss labākos rezultātus ir uzrādījusi Skip Thought metode, kurai rezultāts ir 31 no 40, jeb 28 no 30 līdzīgajos pāros un 3 no 10 pretējajos. Otrajā vietā ir Sent2Vec metode, kuras rezultāts ir 29 no 40, jeb 27 no 30 līdzīgajos pāros un 2 no 10 pretējajos. Viss sliktāko rezultātu uzrādīja FastSent metode, respektīvi 20 no 40 – 4 no 10 pretējos pāros un 16 no 30 līdzīgajos teikumu pāros.

Ja aplūko teikumu pārus, pretējo teikumu pāru kopā, kuru vērtības bija vis mazākās Sent2Vec un Skip Thought modeļos, tad var ievērot, ka šie pāri neizmanto līdzīgus vārdus viens ar otru, kās tas ir vairākumā citu teikumu.

4.10. tabula

Pretējas nozīmes teikumu pāri ar mazāko līdzību

Nr.p.k.	Teikums	Pretējs teikums
4	man ir tiesības braukt ar auto	es esmu gājējs
7	svarīgi ir tikt labā vietā	vietai nav nozīmes
9	novadā dzīvo daudz cilvēku	šī vieta ir neapdzīvota

Tas arī izskaidro, kāpēc šis līdzības koeficients tiem bija viss mazākais.

4.7.3. *Secinājumi*

Aplūkojot iegūtos rezultātus var secināt, ka Skip Thought metode uzrādīja viss labākos rezultātus, kaut gan Sent2Vec metodes rezultāti bija ļoti līdzīgi.

FastSent metode uzrādīja ļoti vājus rezultātus. Spriežot pēc pieejamajiem rezultātiem uz angļu valodas tekstiem, atšķirībai nevajadzēja būt tik lielai, kas liek domāt, ka iespējams modelis tika apmācīts nekorekti, vai arī bibliotēkā, kurā bija realizēta metodes implementācija, ir kāda nepilnība.

Neviena no metodēm nespēja atpazīt noliegumus, no tā var secināt, ka šis metodes neiemācījās tik lielu kontekstu, lai atpazītu noliegumu pēc dažiem nolieguma vārdiem, bet gan saskatīja vārdu līdzības teikumos un tos attēloja tuvu vienu otram.

5. TĀLĀKĀ DARBĪBA

Pirmkārt lai uzlabotu darba rezultātus ir nepieciešams uzlabot apmācāmo datu kopu, kas ir sastādīta no 3 visi dažādiem avotiem, līdz ar to tā aptver plašu tēmu klāstu, kas satur arī dažādas specifiskas lietas. Priekš modeļu salīdzināšanas labāk būtu specifiskāka kopa, kurai var izveidot piemērotākus testa piemērus, kas labāk ļautu spriest par modeļu kvalitāti.

Rezultātu uzlabošanai vairāk ir nepieciešams papētīt tekstu kopas apstrādi pirms metošu apmācīšanas, jo iespējams daudz labākus rezultātus var iegūt, ja tiek izņemti, kādi simboli, vai arī, piemēram vārdi pārvērsit pamatformās.

Ir arī nepieciešams palielināt testa kopas izmēru, lai varētu labāk spriest par apmācītajiem modeļiem.

Nav nepieciešams tikai palielināt esošā testa kopu ar vairāk teikumiem, bet arī izdomāt papildus testus, kā salīdzināt metodes. Tas dotu vispārīgāku skatījumu uz metožu efektivitāti.

Šajā darbā, laika trūkuma dēļ netika daudz variēts ar modeļu apmācīšanas konfigurācijām. To ir nepieciešams darīt, lai iegūtu labākus rezultātus jau no esošajām metodēm.

Ir nepieciešamas aplūkot arī citas metodes, jo tās ir diezgan daudz. Kā arī tā kā teksta analīze un mašīnmācīšanās ir ļoti populāras tēmas mūsdienu datorikā, tad arī nemitīgi veidojas kādas jaunas metodes, kā sasniegt labākus rezultātus. Šīm metodēm vajag sekot līdz un tās paŗ baudīt.

6. REZULTĀTI

Darba ietvaros tika aplūkotas dažādas dabiskās valodas apstrādes metodes, izvēlētas populārākās no tām, kas būtu pielietojamas uz neuzraudzītiem datiem, kā arī izpētīta šo metožu uzbūve un teorētiski tās salīdzinātas viena ar otru.

Papildus tika aplūkoti dažādi datu avoti kur tīmeklī būtu pieejami teksti latviešu valodā, lai varētu izveidot datu kopu, kas vēlāk tika izmantota iepriekš minēto metožu izmēģināšanai.

Darba ietvaros tika izvēlētas 3 mašīnmācīšanās naturālās valodas apstrādes metodes, kas tika izmēģinātas, apmācot modeļus, kas teikumus attēloja vektoru telpās, lai pēc tam varētu noteikt to līdzību vienam ar otru.

Tika izveidots tests, lai salīdzinātu apmācītās metodes. Tas sastāvēja no 50 teikumu pāriem, kuri vai nu bija pēc loģikas līdzīgi, atšķirīgi, vai neitrāli. Salīdzinot metodes tika iegūti rezultāti, kas parādīja, ka Skip Thought metode sasniedza viss labākos rezultātus no apmācītajiem modeļiem, kaut gan iegūtie rezultāti bija ļoti līdzīgi ar Sent2Vec modeli.

Testu rezultātos atklājās arī tas, ka šajā gadījumā FastSent metode sasniedza ļoti sliktus rezultātus, salīdzinot ar pārējām apmācītajām metodēm.

7. SECINĀJUMI

Pirms veic kādu mašīnmācīšanās modeļu apmācīšanu ir nepieciešams iepriekš visu kārtīgi apdomāt, jo šis process var ilgt ļoti ilgu laiku, it īpaši, ka tiek apmācīti neironu tīkli, un ja kaut kas noiet greizi, tad tiek pazaudētas ļoti daudz laika.

Ir sarežģīti izmēģināt daudzas dabiskās valodas apstrādes metodes, jo to implementācijas mēdz ievērojami atšķirties vienai no otras un paiet daudz laiks kamēr izdodas metodi palaist. Papildus, līdz ar to ka šīs metodes ir samērā nesen izveidotas, pieejamā dokumentācija par tām nav diez ko plaša.

Pēc iegūtajiem rezultātiem var secināt, ka uz šo datu kopu lielu nozīmi apmācīšanā nav atstājis, tas, ka metodes skatās uz blakus teikumiem, jo Sent2Vec neaplūko blakus teikumus apmācīšanas procesā, turpretī Skip Thought to dara, bet rezultāti šīm metodēm bija līdzīgi.

Pēc rezultātiem var spriest, ka labākus rezultātus ir uzrādīja tās metodes, kas ņēma vērā vārdu kārtību teikumā.

Rezultāti arī parādīja to, ka aplūkotās metodes nespēja attēlot noliegumu teikumos, tie tā pat teikumus uzskatīja par līdzīgiem, ja tie saturēja līdzīgus vārdus.

Ne vienmēr sarežģītāki un ilgāk apmācāmi modeļi ir labāki, lai attēlotu teikumu loģiku, to var secināt no tā, ka Sent2Vec sasniedza ļoti līdzīgus rezultātus, kaut gan tika apmācīta daudz īsāku laiku.

Tā kā mašīnmācīšanās un valodas apstrāde ir ļoti aktuāla tēma, tad jaunas metodes tiek veidotas ļoti bieži, tādēļ ir nepieciešams nemitīgi sekot līdzi jaunumiem.

IZMANTOTĀ LITERATŪRA UN AVOTI

[1] fastText. [Tiešsaiste]. [atsauce 12.05.2018.] Pieejams: https://research.fb.com/fasttext/

[2] Quoc Le, Tomas Mikolov, "Distributed Representations of Sentences and Documents" 2014. Pieejams: https://arxiv.org/pdf/1405.4053v2.pdf

[3] sent2vec. [Tiešsaiste]. [atsauce 12.05.2018.] Pieejams: https://github.com/epfml/sent2vec

[4] Ryan Kiros , Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, Sanja Fidler, "Skip-Thought Vectors" 2015. Pieejams: https://arxiv.org/pdf/1506.06726.pdf

[5] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loıc Barrault, Antoine Bordes, "Supervised Learning of Universal Sentence Representations from Natural Language Inference Data" 2017. Pieejams: https://arxiv.org/pdf/1705.02364.pdf

[6] Felix Hill, Kyunghyun Cho, Anna Korhonen, "Learning Distributed Representations of Sentences from Unlabelled Data" 2016. Pieejams: https://arxiv.org/pdf/1602.03483.pdf

[7] Wikimedia Downloads. [Tiešsaiste]. [atsauce 12.05.2018.] Pieejams: https://dumps.wikimedia.org/lvwiki/latest/

[8] wiki2text. [Tiešsaiste]. [atsauce 12.05.2018.] Pieejams: https://github.com/rspeer/wiki2text

[9] The open parallel corpus. [Tiešsaiste]. [atsauce 12.05.2018.] Pieejams: http://opus.nlpl.eu/

[10] Latviešu teksti datortīklā. [Tiešsaiste]. [atsauce 13.05.2018.] Pieejams: http://www.korpuss.lv/klasika/

[11] Latvian. [Tiešsaiste]. [atsauce 24.01.2018.] Pieejams: https://github.com/pdonald/latvian

[12] skip-thoughts. [Tiešsaiste]. [atsauce 12.05.2018.] Pieejams: https://github.com/ryankiros/skip-thoughts

[13] Skip-Thought Vectors. [Tiešsaiste]. [atsauce 12.05.2018.] Pieejams: https://github.com/tensorflow/models/tree/master/research/skip_thoughts

[14] SentenceRepresentation. [Tiešsaiste]. [atsauce 12.05.2018.] Pieejams: https://github.com/fh295/SentenceRepresentation

[15] Akadēmiskā terminu datubāze AkadTerm. Čeka definīcija [Tiešsaiste]. [atsauce 12.05.2018.] Pieejams: http://termini.lza.lv/term.php?term=thread&lang=LV

PIELIKUMI

1. pielikums. Testu kopa ar līdzīgas nozīmes teikumu pāriem

Nr.p.k.	Teikums	Līdzīgs teikums
1	man ir 3 bērni	es esmu vecāks
2	iet gar jūru	pastaigāties pa pludmali
3	valstī notiek daudz pasākumu	republikas ir daudz notikumu
4	protams latvija uzvarēs hokejā	latvija ir labākā hokejā
5	man ir daudz problēmas	es dzīvoju ar grūtībām
6	kungs zina visu	karalis ir izglītots
7	čempionātā uzvar labākais	sacensībās nosaka uzvarētājus
8	labas ziņas	patīkami jaunumi
9	amerikas savienotās valstis ir liela valsts	asv teritorija ir plaša
10	eiropas savienībā ir daudz valstu	es dalībvalstu skaits ir liels
11	septembrī skolēni sāk iet skolā	rudenī atsākas mācības bērniem
12	latvijas izlase labi nospēlēja	valsts vienības sniegums bija atzīstams
13	šī sieviete ir glīta	dāma ir skaista
14	politiķi daudz melo	deputāti nestāsta patiesību
15	izglītības sistēmā ir daudz problēmas	mācības ir iespējams uzlabot
16	šī melodija ir lipīga	dziesma paliek atmiņā
17	vēsturē notika daudz kari	pagātnē bija neskaitāmi konflikti
18	salauzts telefons	bojāts tālrunis
19	mest nazi	sviest asmeni
20	krievijai ir liela teritorija	padomju savienība bija plaša
21	nhl ir stiprākā līga	hokejs amerikā ir augstā līmenī
22	zemnieki dzīvo laukos	lauksaimnieki uzturas ārpus pilsētām
23	uzņēmuma priekšsēdētājs	iestādes vadītājs
24	nato karaspēks	apvienoto nāciju armija
25	televīzijā rāda interesantus raidījumus	tv ekrānos pārraida labas filmas
26	latvija igaunija un lietuva	baltijas valstis
27	personas dati	informācija par cilvēku
28	literatūras darbs ir labi uzrakstīts	grāmata ir interesanta
29	londonā notiek festivāls	anglijā norit pasākums
30	nasa	kosmosa izpēte

2. pielikums. Testu kopa ar pretējas nozīmes teikumu pāriem

Nr.p.k.	Teikums	Pretējs teikums
1	es eju ēst	es neeju ēst
2	man ir viens kaķis	man nav mājdzīvnieku
3	latvija ir skaista valsts	latvija ir slikta
4	man ir tiesības braukt ar auto	es esmu gājējs
5	komisija izlēma , ka iedzīvotāji ir svarīgi	komisija noliedza cilvēku nozīmi
6	vakar bija saules aptumsums	naktī spīdēja mēness
7	svarīgi ir tikt labā vietā	vietai nav nozīmes
8	pēteris redzēja lielu ziloni	pēteris ir akls
9	novadā dzīvo daudz cilvēku	šī vieta ir neapdzīvota
10	šahs ir interesants sporta veids	šahs nav sports , tā ir galda spēle

3. pielikums. Testu kopa ar neitrālas nozīmes teikumu pāriem

Nr.p.k.	Teikums	Teikums
1	man patīk skriet	jānis skatījās tv
2	ēdiens ir ļoti gards	kosmosa kuģis lidoja pa kosmosu
3	vakar ieradās pēteris	lasīt ir interesanti
4	francija atrodas eiropā	zivis peld pa straumi
5	rotaļlietu ražošana	datorika ir visur
6	kristietība ir ļoti veca reliģija	dzert ūdeni ir svarīgi
7	kā tev iet ?	dzīve ir skaista
8	vakar nokrita zvaigzne	šis krājums satur daudz informācijas
9	parkā notiek brīnumi	vēsturē bija daudz karaļi
10	vakar bija karsts laiks	dārzeņi ir veselīgi

DOKUMENTĀRĀ LAPA

Maģistra darbs "**Pēc loģikas līdzīgu teikumu meklēšana, izmantojot mašīnmācīšanās metodes**" izstrādāts LU Datorikas fakultātē.

Darba teksta galīgā versija izgatavota 21.05.2018.

Ar savu parakstu apliecinu, ka pētījums veikts patstāvīgi, izmantoti tikai tajā norādītie informācijas avoti un iesniegtā darba elektroniskā kopija atbilst izdrukai.

Autors: **Henrijs Smelēns_____21.05.2018**

Ar savu parakstu apliecinu, ka esmu lasījis augstāk minēto maģistra darbu un atzīstu to par p i e m ē r o t u / n e p i e m ē r o t u (nevajadzīgo svītrot) aizstāvēšanai Latvijas Universitātes datorzinātņu maģistrantūrā.

Darba vadītāja: **Dr.dat. Inguna Skadiņa_____21.05.2018**

Darbs iesniegts **maģistratūras sekretariātā** 21.05.2018.
Ar šo es apliecinu, ka darba elektroniskā versija ir augšupielādēta LU informatīvajā sistēmā.

Studiju metodiķe: _____.

Recenzents: **Docents Normunds Grūzītis _____**

Darbs aizstāvēts maģistra gala pārbaudījuma komisijas sēdē
_____ prot. Nr. _____

Komisijas sekretārs: _____