RĪGAS TEHNISKĀ UNIVERSITĀTE
Datorzinātnes un informācijas tehnoloģijas fakultāte
Lietišķo datorsistēmu institūts

Rinalds Vīksna
maģistra akadēmisko studiju programmas „Datorsistēmas”
students, stud. apl. nr. 011RDB353

Emocionālās ekspresijas noteikšana sīkziņās latviešu valodā

Maģistra darbs

Zinātniskais vadītājs
Dr.sc.ing., docents
G.JĒKABSONS

Rīga 2018

RĪGAS TEHNISKĀ UNIVERSITĀTE
Datorzinātnes un informācijas tehnoloģijas fakultāte
Lietišķo datorsistēmu institūts
Programmatūras inženierijas katedra

APSTIPRINU
PI katedras vadītājs
Dr.sc.ing., as. profesors

_____ A. Jurenoks

_____. gada ___._____

MAĢISTRA DARBA UZDEVUMS
maģistra akadēmisko studiju programmas „Datorsistēmas”
studentam Rinaldam VĪKSNAM

Maģistra darba tēma: Emocionālās ekspresijas noteikšana sīkziņās latviešu valodā

Darba vadītājs: *Gints Jēkabsons*, Dr.sc.ing., docents,

Darba nodošanas termiņš _____. gada ____. _____

Uzdevuma izsniegšanas datums: _____. gada ____. _____

_____
(darba vadītāja paraksts)

Uzdevums pieņemts izpildīšanai: _____. gada ____. _____

_____
(studenta paraksts)

Maģistra darbs izstrādāts *Programmatūras inženierijas katedrā.*

Darba autors: *stud. R. Vīksna* _____
(paraksts, datums)

Darba vadītājs: *Dr.sc.ing. G. Jēkabsons* _____
(paraksts, datums)

Maģistra darbs ieteikts aizstāvēšanai:

PI katedras vadītājs.: *Dr.sc.ing., as. profesors A. Jurenoks* _____
(paraksts, datums)

Maģistra darbs aizstāvēts *maģistra akadēmisko studiju programmas „Datorsistēmas" Lietišķo datorsistēmu programmatūras virziena gala pārbaudījuma komisijas* 2018. gada ___. _____ sēdē un novērtēts ar atzīmi _____ (____).

LDP virziena gala pārbaudījuma komisijas

sekretārs: _____ /_____/

ANOTĀCIJA

Noskaņojuma analīze ir pētījumu virziens, kas, pielietojot dažādas dabīgās valodas apstrādes metodes, pēta un analizē cilvēku viedokļus un subjektīvos izteikumus, lai noteiktu vērtējumus, attieksmes un emocijas pret kādu notikumu vai objektu pasaulē. Noskaņojuma analīze tiek lietota, lai atklātu agresīvu viedokļu paušanu tīmeklī, sabiedrības viedokļa pētīšanai dažādu reklāmas un politisko kampaņu ietvaros.

Maģistra darba gaitā tika veikts pētījums par iespējām pielietot noskaņojuma analīzes metodes latviešu valodā rakstītu sīkziņu klasificēšanai. Latviešu valoda atšķirībā no angļu valodas, kurā ir veikta lielākā daļa pētījumu noskaņojuma analīzes jomā, ir morfoloģiski bagāta valoda un nepieciešamas teksta apstrādes metodes, kas šo īpatnību ņem vērā. Darba gaitā tika izpētītas līdz šim pielietotās metodes, izmantotie datu avoti un iegūtie rezultāti.

Tika identificētas dažādas teksta apstrādes metodes, kuru lietderīgums tika novērtēts izmantojot standartizētu datu kopu un novērtēšanas metodes. Pēc dažādu teksta apstrādes metožu novērtēšanas tika izstrādāta priekšapstrādes metožu secība, kuru izmantojot, tika apstrādāti apmācības dati un izveidots modelis. Izveidotais modelis tika pārbaudīts izmantojot šķērsvalidēšanas metodi.

Darba pamattekstā ir 48 lappuses, 16 attēli, 20 tabulas, 47 nosaukumu informācijas avoti un 0 pielikumi, taču izejas kods un izmantotie dati ir izlikti Github.

ABSTRACT

Sentiment analysis is a field of study, which using various natural language processing tools analyzes opinions, emotions and subjective expressions of people against some other entity. Sentiment analysis may be used to find out aggressive or abusing comments or to research effectiveness of political campaigns or mood of consumers about some product or service.

The goal of study in the Thesis "Sentiment analysis in Latvian tweets" is to review and test various text preprocessing methods suitable for texts written in Latvian.

Author has gathered tweet corpus in Latvian from Twitter and part of gathered corpus was annotated using crowd-sourcing website developed by author. Annotated corpus was used to test and evaluate text preprocessing methods using four different machine learning models. During work most useful preprocessing methods were identified and combined to develop flow of preprocessing methods which produce best results. Preprocessed data was used to train classifier, which was tested using cross-validation and external data corpus.

The thesis contains 48 pages, 16 figures, 20 tables, 47 information sources and no appendixes.

SATURS

IEVADS

Tīmeklī pieejams milzīgs informācijas daudzums, kas tiek nemitīgi papildināts. Dažādās vietnēs tiek publicētas dažādas ziņas, produktu apskati, lietotāju pieredze un pārdzīvojumi. Šī informācija satur ne tikai faktus par aprakstīto notikumu, cilvēku vai lietu, bet arī autora subjektīvo attieksmi. Emocionālās ekspresijas noteikšanai pielieto dabīgās valodas apstrādes metodes un teksta analīzes metodes lai noteiktu autora subjektīvo viedokli tekstā. Latviešu valodā šo metožu kopu, sauc „Noskaņojuma analīze" [1], bet literatūrā angļu valodā - *Sentiment Analysis* vai arī *Opinion Mining* [1][2]. Emocionālās ekspresijas noteikšana ļauj secināt lietotāju noskaņojumu saistībā ar kādu nosaukto vienumu (produktu, pakalpojumu, zīmolu u.tml.), kas ir biznesā ļoti nozīmīga informācija, jo parāda klientu viedoklis tieši iespaido to rīcību.

Noskaņojuma analīze tiek definēta [2] kā pētījumu virziens, kas analizē cilvēku viedokļus, noskaņojumus, vērtējumus, attieksmes un emocijas attiecībā pret kādu citu vienumu, kā, piemēram, produktiem, pakalpojumiem, organizācijām, indivīdiem, notikumiem vai to atribūtiem. Rakstītajā tekstā noskaņojums tiek izpausts ar vārdiem, piemēram, izvēloties vārdus vai frāzes, kas izsaka rakstītāja noskaņojumu, vai arī izmantojot neverbālos signālus, piemēram, emocijzīmes, pieturzīmes vai rakstīšanas stilu [3].

Teksta noskaņojuma noteikšana ir noderīga dažādās jomās, piemēram, lai iegūtu atgriezenisko saiti par kādu zīmolu vai produktu [4], agregētu un apkopotu viedokļus no ieteikšanas sistēmām (*recommender systems*) [5], noskaidrotu politisko kampaņu efektivitāti [6] u.c.

Angļu valodā publicēto tekstu sentimenta klasificēšana ir literatūrā plaši pētīta joma [7],[8],[9],[10], kamēr citām valodām ir pievērsts mazāk uzmanības[11],[12],[8], turklāt latviešu valodā noskaņojuma analīzes jomā pēdējo 4 gadu laikā ir tikai 3 publikācijas [13],[14],[15]. Noskaņojuma analīzes veikšanai nepieciešams iegūt kvalitatīvu anotētu datu kopu un balstoties uz to jāizveido modelis klasificēšanai. Tādēļ darba ietvaros tiek apzinātas jau esošās datu kopas un iespējas iegūt anotētu datu kopu. Tiek izpētītas tekstu priekšapstrādes metodes, izmantojamie faktoru (*features*) vektori un to veidošanas metodes, kā arī klasificēšanas metodes. Darba praktiskajā daļā ir veikta sīkziņu korpusa iegūšana no sociālā tīkla Twitter[1] un daļa no iegūtā korpusa tiek anotēta izmantojot trīs klases: pozitīvs,

[1] https://twitter.com/

negatīvs un neitrāls sentiments. Izmantojot iegūto anotēto korpusu, ir veikts dažādu faktoru vektora iegūšanas metožu salīdzinājums un novērtēta un salīdzināta vairāku klasificēšanas metožu darbība.

Maģistra darba mērķis ir izpētīt emocionālā noskaņojuma noteikšanas metodes, kā arī sistematizēt un salīdzināt emocionālā noskaņojuma analīzes metodes latviešu valodā rakstītām sīkziņām.

Mērķa sasniegšanai tika izvirzīti šādi uzdevumi:

1. Izpētīt emocionālā noskaņojuma noteikšanas problēmu un tās iespējamos risinājumus.
2. Izpētīt populārākās klasificēšanas metodes, kas pielietojamas noskaņojuma analīzei.
3. Izpētīt metodes teksta transformēšanai faktoru vektorā un salīdzināt dažādu metožu ietekmi uz sasniegto rezultātu.
4. Realizēt teksta transformēšanas un klasificēšanas metodes programmatūrā un veikt to darbības salīdzināšanu.

Maģistra darba autors par veikto pētījumu rezultātiem ir ziņojis šādās zinātniskajās konferencēs:

- RTU 59. studentu zinātniskā un tehniskā konferencē ar prezentāciju „Emocionālās ekspresijas noteikšana sīkziņās latviešu valodā”

Pētījuma rezultāti ir publicēti šādos zinātnisko rakstu krājumos:

- Vīksna R., Jēkabsons G. Sentiment Analysis in Latvian and Russian: A Survey // Scientific journal „Applied Computer Systems” Vol. 23, 2018, 7 p. (in press) ISSN 2255-8683, e-ISSN 2255-8691 (iesniegts)

Darbs sastāv no 5 nodaļām: ievada, kas iepazīstina ar noskaņojuma analīzes problemātiku un risinājumiem; 2. nodaļa datu iegūšana īsi apskata dažādus datu avotus, ko pielieto noskaņojuma analīzei; 3. nodaļa datu priekšapstrāde un faktoru izvēle apskata metodes, ko pielieto lai transformētu tekstu faktoru vektorā; 4. nodaļā klasificēšanas metodes tiek apskatītas teksta klasificēšanas metodes un iespējas tās pielietot latviešu valodā rakstīto tekstu klasificēšanai; 5. nodaļā tiek veikti eksperimenti un iegūto rezultātu analīze; secinājumos tiek veikta iegūto rezultātu analīze un iezīmēti iespējamie tālākā darba virzieni.

1. NOSKAŅOJUMA ANALĪZE

Noskaņojuma analīze visbiežāk tiek veikta trīs dažādos detalizācijas līmeņos. Dokumenta līmenī ir iespējams noskaidrot dokumenta kopējo noskaņojumu, turklāt tiek pieņemts, ka veicot šo uzdevumu nav nepieciešams noteikt ne viedokļa izteicēju, ne entītiju, par kuru tiek izteikts viedoklis, ne arī noskaņojuma izteikšanas laiku – tie ir vai nu zināmi iepriekš vai arī nav svarīgi. Svarīgi ir noskaidrot tikai dokumenta kopējo noskaņojumu. Teikuma līmenī uzdevums ir noteikt konkrētajā teikumā esošo emocionālo ekspresiju, un novērtēt to kā pozitīvu, negatīvu vai neitrālu viedokli. Entītijas un aspekta līmenī tiek veikta sīkāka analīze, ar mērķi noskaidrot lietotāja viedokli un emocionālo ekspresiju attiecībā uz kādas mērķa entītijas īpašību vai pašu entītiju.[2]

B.Liu [2] viedokli definē kā vektoru (g,s,h,t), kur g – noskaņojuma mērķa entītija, s – noskaņojums, h – noskaņojuma izteicējs un t – laiks, kad noskaņojums ir ticis izteikts. Tā piemēram Twitter sīkziņā *„Lembergs par valsts budžetu: Pastāv liels risks nesavilkt galus kopā”* izteikts Lemberga (h) negatīvais (liels risks nesavilkt galus) viedoklis (s) par valsts budžetu (g). Laika brīdis sīkziņā nav dots, tomēr to var atrast sīkziņas meta datos. Lai varētu veikt viedokļu atrašanu brīvā tekstā, ir nepieciešams veikt vairākus apakš uzdevumus:

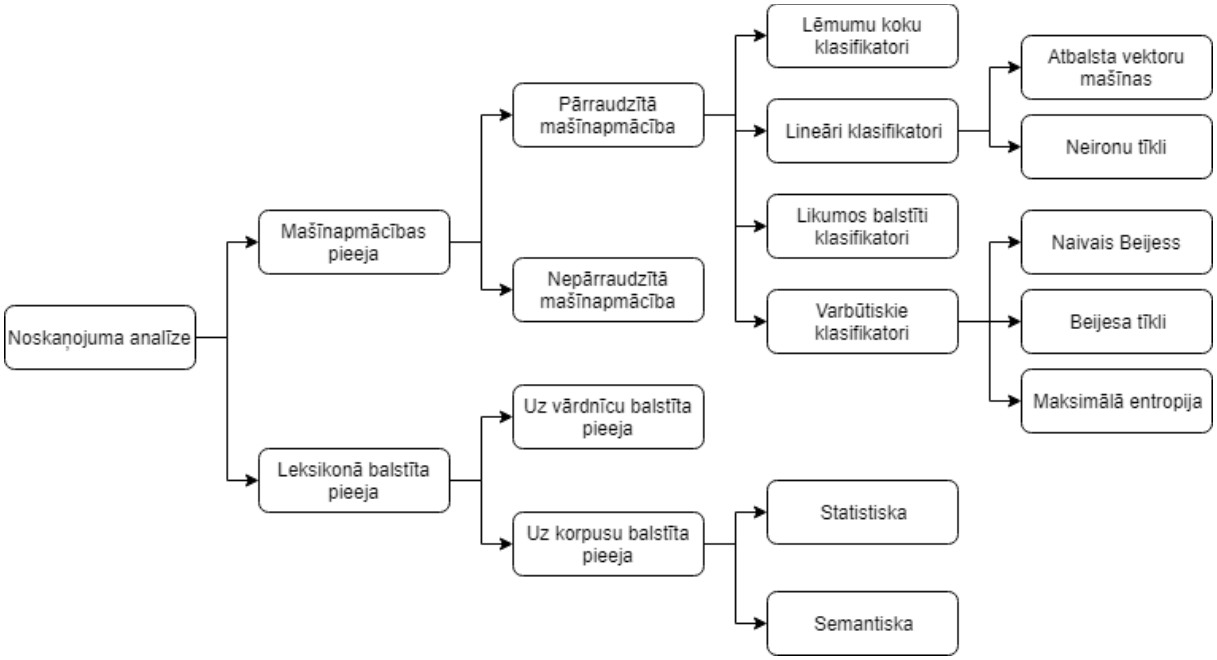
- Entītiju atrašana un klasificēšana identificē tekstā atrodamās entītijas un sagrupē tās;
- Aspektu atrašana un klasificēšana atrod katrai entītijai piesaistītās īpašības;
- Viedokļa izteicēja atrašana izgūst no teksta viedokļa autorus;
- Laika izgūšana – atrod laiku, kad viedoklis ticis izteikts;
- Emocionālās ekspresijas noteikšana – klasificē viedokļa noskaņu kā pozitīvu, negatīvu vai neitrālu, vai arī kādā citā skalā;
- Rezultāta apkopošana apkopo un sasaista iegūto informāciju kā vektoru (g,s,h,t).

Šajā darbā ar terminu „noskaņojuma analīze” tiks saprasta tieši emocionālās ekspresijas noteikšana, t.i. teksta (sīkziņas) klasificēšana kā saturoša pozitīvu, negatīvu vai neitrālu emocionālo ekspresiju.

1.1. Noskaņojuma analīzes metodes

Dokumenta līmeņa noskaņojuma analīzes metodes nosaka, kāds ir dokumenta kopējais noskaņojums. Veicot noskaņojuma analīzes uzdevumu, tiek pieņemts, ka nav nepieciešams

izgūt ne entītijas ne emocionālās ekspresijas autoru, ne arī laiku, kad šis noskaņojums bijis izteikts. Svarīgi ir tikai klasificēt teksta emocionālo noskaņojumu. Noskaņojumu var izteikt ar 2 klasēm (pozitīvs, negatīvs), 3 klasēm (pozitīvs, negatīvs, neitrāls) vai arī ar reitinga palīdzību (piemēram, no –3 līdz +3; no 0 līdz 5 utml.). Lai veiktu šo klasificēšanas uzdevumu iespējams pielietot dažādas pieejas un metodes. Walaa, Ahmed un Hoda [8] apskatot 54 darbus noskaņojuma analīzes jomā izdala divas galvenās pieejas: leksikonā balstītu pieeju un mašīnapmācības pieeju, kuras tālāk tiek sadalītas sīkāk (1.1 att.).



1.1. att. Walaa, Ahmed un Hoda piedāvātā noskaņojuma analīzes metožu klasifikācija [8]

Arī Liu [2] identificē trīs galvenās pieejas lai klasificētu teksta noskaņojumu – pārraudzīto un nepārraudzīto mašīnapmācību un leksikonā balstītu pieeju. Tobias darbā [3] piedāvāta papildu metožu grupu „Grafos balstīta iezīmju izplatīšanās" un Devika et al. darbā [16] piedāvā papildu metožu grupu „Likumos balstīta pieeja". „Grafos balstītā iezīmju izplatīšanās" metode apraksta veidu, kādā noteikt stopvārdus, kas noteikti jāpatur (dotais piemērs stopvārdam „the" ir: „shit" – negatīva nozīme un „the shit" pozitīva nozīme). Tātad šī metode apraksta nevis kopīgu pieeju noskaņas analīzei, bet gan teksta priekšapstrādes metodi. Savukārt „Likumos balstīta pieeja" izmanto īpašus likumus lai paplašinātu un pielāgotu leksikonu izmantojot doto tekstu, t.i. šī metode pēc autora domām ir iekļaujama pie leksikonā balstītajām pieejām. Klasificēšanas metodes tuvāk apskatītas tiek 4. nodaļā.

1.2. Noskaņojuma analīzes procesa soļi

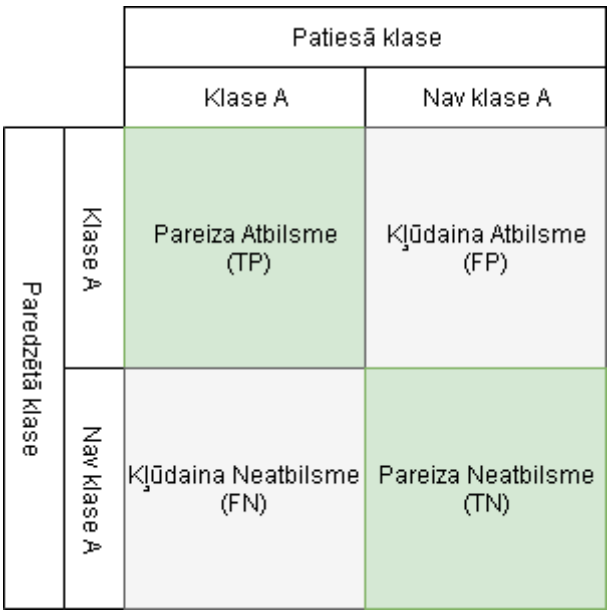
Noskaņojuma analīze ir sarežģīts uzdevums – ir jāveic vairāki apakš uzdevumi (skat. 1.2. att.), lai veiksmīgi noteiktu tekstā pausto noskaņojumu.

1.2. att. Noskaņojuma analīzes apakšuzdevumi

Datu iegūšanas posmā tiek iegūti apmācības dati, turklāt ir jāpārliecinās, ka tie ir derīgi uzstādītajam mērķim (bez atkārtojumiem, izvēlētajā valodā, u.t.t.). Datu priekšapstrādes solī notiek tekstu sākotnējā apstrāde, izmetot nevajadzīgo informāciju un izlabojot acīmredzamas kļūdas tekstā. Faktoru izgūšanas solī no teksta tiek izgūti faktori (visbiežāk vienkārši sadalot tekstu vārdos un pieņemot, ka vārdu secība nav svarīga – t.s. "soma ar vārdiem" (*Bag of words*) pieeja [2]). Mašīnapmācības metožu kontekstā faktors ir kāda individuāla pazīme vai īpašība, ko var izmērīt un kas raksturo pētāmo objektu. Sīkziņu gadījumā faktors var būt kāda vārda vai vārdu n-grammas esamība tekstā, pieturzīmju lietojums, vārdu skaits, emocijzīmju lietojums, vai jebkādi citi dati par sīkziņu, kas to raksturo. Pēc faktoru izgūšanas, tiek izvēlēti faktori, kas tiks lietoti klasificējot teksta noskaņojumu, šī faktoru kopa ir faktoru vektors, kuru izmantojot tiek izvēlēts un apmācīts klasifikators. Klasifikatoru novērtē izmantojot iepriekš atdalīto testa datu kopu, un, ja klasifikatora darbība ir apmierinoša, to var lietot paredzētajam mērķim. Atbilstoši šiem noskaņojuma analīzes procesa soļiem ir strukturētas šā darba tālākās nodaļas, apvienojot apakš uzdevumus, kas tipiski ir cieši saistīti (datu priekšapstrāde un faktoru izgūšana).

1.3. Iegūto rezultātu novērtēšana

Izveidoto noskaņojuma analīzes modeli var novērtēt, padodot tās izveidotajam klasifikatoram datus (testpiemērus), kas netika izmantoti tās izveidošanai, un novērtējot iegūtos rezultātus. Rezultātus ir iespējams apkopot pārpratumu matricā (*confusion matrix, coincidence matrix*). 1.3. attēlā redzams šādas matricas piemērs attiecībā pret klasi A.



1.3. att. Vienkārša pārpratumu matrica. Aizgūts no [17]

Jebkurš no iegūtajiem rezultātiem pieder vienai no četrām kategorijām, kur TP un TN ir pareizi klasificētie piemēri, bet FP un FN ir nepareizi klasificētie piemēri. Izmantojot atrastās TP, TN, FP un FN vērtības var izskaitļot dažādus modeli raksturojošos darbības rādītājus [17] klasei A:

- Precizitāte (*Precision* – P) novērtē, cik no kādas klases prognozētajām vērtībām ir pareizi prognozētas. Tās noteikšanai lieto formulu $P = \frac{TP}{TP+FP}$;
- Pārklājums (arī jutīgums, *Recall* – R) novērtē, cik no patiesi pareizām vērtībām ir klasificētas pareizi. Noteikšanai lieto formulu $R = \frac{TP}{TP+FN}$;
- F1 mērs ir precizitātes un pārklājuma harmoniskais vidējais un to aprēķina šādi: $F1 = 2 * \frac{P*R}{P+R}$;
- Akurātums (*Accuracy* – A) kopējā modeļa precizitāte norāda, cik no klasificētajiem rezultātiem tika pareizi klasificēti. Aprēķina izmantojot formulu $A = \frac{TP+TN}{TP+TN+FP+FN}$.

Gadījumos kad klasificēšanai tiek lietotas vairāk nekā divas klases, šāda pārpratumu matrica var tikt izveidot katrai klasei un attiecīgi aprēķināta Precizitāte, Pārklājums un F1.

Makro vidējo vērtību tad var atrast kā matemātisko vidējo visām klasēm. Veicot šķērsvalidēšanu, P, R, F1 un A vērtības var tikt aprēķinātas katrā šķērsvalidēšanas iterācijā un tad aprēķināts vidējais, vai arī var izmantot mikro-vidējo (*micro-average*) vērtību aprēķināšanu [18]. Papildus minētajiem rādītājiem var tikt lietoti arī citi (pareizas atbilsmes intensitāte – *True Positive Rate,* ROC laukums – *ROC area,* prevalence - *Prevalence*)[17][19], tomēr tie emocionālās ekspresijas noteikšanā tiek lietoti reti un tāpēc šajā darbā netiks apskatīti.

1.4. Secinājumi

Šajā nodaļā tika apskatīta noskaņojuma analīzes problēmas konceptuāls risinājums, un iegūtā risinājuma novērtēšana. Tālākajās nodaļās tiek tuvāk apskatīti dotā procesa soļi un veikti eksperimenti, lai noskaidrotu dažādu metožu lietderīgumu strādājot ar sīkziņām latviešu valodā.

2. DATU IEGŪŠANA

Jebkurai sentimenta analīzes sistēmai ir nepieciešami dati – anotēts tekstu korpuss, izmantojot kuru, mašīnapmācības modelis tiek apmācīts un novērtēts. Metodēm, kas izmanto leksikonu, ir nepieciešams iegūt leksikonu – vārdnīcu ar vārdiem, kur katram vārdam ir piesaistīta kāda sentimenta klase.

Datu iegūšana sākas ar datu avota izvēli, kas var būt jebkādi pieejami teksta dati, tomēr lielākoties latviešu valodā veiktajos pētījumos tikuši lietoti komentāri no ziņu portāliem [13] un Twitter sīkziņas [15],[14]. Sociālie tīkli un Twitter tiek bieži lietots arī citvalodu tekstu korpusu iegūšanai [20]. Twitter sīkziņās lietotāji brīvā formā pauž savus pārdzīvojumus un dalās ar pieredzēto un viedokļiem, tādēļ šādi teksti nereti ir emocionāli un izmantojami noskaņojuma analīzei. Twitter piedāvā saskarni[2], kuru izmantojot, iespējams klausīties sīkziņu straumi un filtrēt to lai iegūtu vēlamās sīkziņas izmantojot atslēgvārdus, ģeolokāciju vai citus parametrus. Java valodā implementēta bibliotēka twitter4j[3] ļauj viegli pieslēgties Twitter un lejupielādēt sīkziņas, kas atbilst izvēlētajiem parametriem. Lejupielādētās sīkziņas ir JSON formātā un satur ne tikai sīkziņas tekstu, bet arī dažādus metadatus, kā avota informācija, lietotāja informācija (vārds, vieta u.c.), valoda, ģeolokācija u.c[4].

Pārraudzītās mašīnapmācības modeļa apmācīšanai nepieciešami dati, kuros katrai datu vienībai ir piesaistīta atbilstošā klase. Twitter sīkziņām iespējams piesaistīt klasi manuāli – cilvēks izlasa sīkziņu un nosaka tās klasi – pozitīva, neitrāla vai negatīva (vai citās izvēlētajās klasēs, piemēram, agresīva-neagresīva, vai pozitīva-negatīva), vai arī automatizēti, izmantojot kādu klasificēšanas metodi, piemēram, emocijzīmju un atslēgvārdu leksikonu [20]. Šāda pieeja ļauj precīzi atrast pozitīvas un negatīvas sīkziņas, tomēr neitrālās sīkziņas ar šādu metodi pēc autora domām nevar droši iegūt, jo leksikonā definēto atslēgvārdu neesamība tekstā neliecina par tā piederību neitrālai klasei. Šādu automatizētu iegūtu anotētu tekstu sauc par „trokšņaini anotētu", jo tas satur arī nepareizi anotētus tekstus. Manuāli anotētās sīkziņas tiek izmantotas kā „zelta standarts", pret kuru salīdzināt ar izvēlēto metodi iegūtos klasificēšanas rezultātus.

[2] https://developer.twitter.com/en/docs

[3] http://twitter4j.org/en/index.html

[4] https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/intro-to-tweet-json

Metodes, kas neizmanto mašīnapmācību, lieto iepriekš definētu leksikonu lai klasificētu tekstu kā piederīgu kādai no klasēm. Leksikons ir vārdnīca ar tekstvienībām vai citiem faktoriem, kam piesaistīta kāda no mērķa klasēm. Teksta klases noskaidrošana vienkāršākajā gadījumā var tikt reducētu uz vienkāršu leksikonā atrodamo tekstvienību skaitīšanu un galīgās teksta klases atrašanu pēc vairākuma principa. Lai šāda metode darbotos veiksmīgi, ir nepieciešams kvalitatīvs un dotajai jomai pielāgots leksikons, kuru iespējams iegūt manuāli izvēloties vārdus, vai arī automatizēti, nelielai vārdu kopai ar zināmu noskaņojuma klasi atrodot sinonīmus, locījumus un radniecīgus vārdus [2]. Twitter sīkziņu noskaņojuma analīzei piemērota leksikona izveidošanu ir pētījis Špats darbā [19], kura laikā iepriekš identificētais leksikons tika papildināts ar vairāk nekā 70000 unigrammām. Tomēr arī izmantojot leksikonā balstītu metodi klasificēšanai, ir nepieciešams anotēts teksta korpuss, lai novērtētu izveidotā modeļa veiktspēju.

Angļu valodā ir pieejami daudzi resursi ar anotētiem tekstiem [7], piemēram, [21],[22],[23], taču latviešu valodā ir maz šādu resursu. Tālākajās nodaļā tiek aplūkotas dažādu autoru lietotās pieejas lai iegūtu šādus korpusus vai leksikonus, kā arī iegūtie datu korpusu apjomi un struktūra.

Ja anotētā apmācības kopa ir neliela, korpusa izmēru var palielināt dublicējot tekstus un tad aizvietojot vārdus ar sinonīmiem lietojot sinonīmu vārdnīcas. Darbā [24] tika lietoti tikai vārdi ar refleksīviem sinonīmiem (A ir sinonīms B, un B ir sinonīms A). Izmantojot šo metodi, apmācības korpuss tika palielināts no 136016 līdz 195372 tekstiem. Cita korpusa papildināšanas metode bija jaunu īpašības vārdu pievienošana lietvārdiem. Izmantojot šo metodi vispirms korpuss tiek analizēts, lai atrastu īpašības vārdus, kas parasti ir sastopami kopā ar kādu lietvārdu, tad atbilstoši atrastajām varbūtībām, apskati tiek papildināti, pievienojot īpašības vārdus pie lietvārdiem, kam vēl nav piesaistīts īpašības vārds. Izmantojot šo metodi, apmācības kopa tika papildināta līdz 196268 tekstiem. Galinsky et al. [24] demonstrēja, ka korpusa papildināšana izmantojot sinonīmus var ievērojami uzlabot modeļa klasificēšanas spēju, kamēr korpusa papildināšana, izmantojot īpašības vārdus, var pasliktināt modeļa klasificēšanas spēju.

2.1. Datu iegūšana latviešu valodā

Garkāje et al. [13] izmantoja korpusu, kas satur vairāk nekā 11 miljonus lietotāju komentāru no populārākajiem Latvijas ziņu portāliem (Apollo, Tvnet un Delfi). 3000 no iegūtajiem dokumentiem tika manuāli klasificēti 3 kategorijās: agresīvi, neagresīvi un neitrāli.

Katrs dokuments tika novērtēts divreiz, un, ja vērtētāju viedokļi nesakrita, trešais anotētājs veica galīgo novērtējumu.

Peisenieks [25] ievāca 1.2 miljonus sīkziņu no Twitter. Lai ievāktu sīkziņas latviešu valodā, kā vaicājuma parametrs tika norādīta aptuvena Latvijas kontūra. Peisenieks un Skadiņš [15] veica sīkziņu anotēšanu izmantojot īpaši šim nolūkam veidotu tīmekļa vietni, kur jebkurš interesents varēja novērtēt sīkziņas kā piederošas vienai no trim klasēm. Galīgais darbā lietotais korpuss sastāv no 383 pozitīvi, 627 neitrāli un 167 negatīvi novērtētām sīkziņām[5]. Peisenieks lietoja Fleisa daudzvērtētāju Kappa koeficientu, lai novērtētu vērtētāju vienprātību. Aprēķinātā k vērtība 0.284 norāda, ka sakritība ir pieņemama.

Nicmanis [26] izmantoja Peisenieka un Skadiņa izveidoto korpusu un papildus izveidoja arī jaunu korpusu izmantojot Twitter. Lai iegūtu sīkziņas latviešu valodā, tika ievāktas sīkziņas no viena lietotāja un tā sekotājiem. Daļu no savāktajām sīkziņām manuāli novērtēja un ieguva anotētu korpusu, kas sastāv no 3131 sīkziņas (1085 no tām pozitīvi novērtētas, 1712 – neitrāli un 334 negatīvi). Arī šis korpuss ir brīvi pieejams Github[6].

Gediņš [27] izmantoja 1 miljonu sīkziņu, ko pētījuma mērķiem nodrošināja SIA „SOON". Lai trenētu modeli, no tiem tika izgūtas sīkziņas ar emocijzīmēm, un sīkziņas tika anotētas izmantojot emocijzīmes. Tādējādi korpuss tika anotēts kā saturošs 13000 negatīvi, 130000 pozitīvi un 750000 neitrāli novērtētas sīkziņas. Lai apmācītu modeli, datu kopai ir jābūt līdzsvarotai, tādēļ apmācībai tika lietotas 13000 negatīvi, 13000 neitrāli un 13000 pozitīvi novērtētas sīkziņas. Sīkziņas, kas nesaturēja emocijzīmes, tika uzskatītas kā piederošas pārsvarā neitrālām, tā kā sīkziņa var saturēt sentimentu arī tad ja tā nesatur emocijzīmes. Papildus tika nolīgts arī vērtētājs, kas manuāli novērtēja sīkziņas. Salīdzinot cilvēka novērtētās sīkziņas ar sīkziņām, kas tika novērtētas izmantojot emocijzīmes, novērtējums sakrita 69.3% sīkziņām, ja tika izmantotas 3 klases. Taču 85.4% sīkziņu, ko cilvēks novērtēja kā pozitīvas un 94.8% sīkziņu, ko cilvēks novērtēja kā negatīvas tika pareizi novērtētas izmantojot emocijzīmes. No 1000 sīkziņām, kas tika klasificētas kā neitrālas jo nesaturēja emocijzīmes, 65.3% par neitrālām atzina arī anotētājs. No tā Gediņš secināja, ka neitrālā korpusa ticamība vērtējama kā zema.

[5] https://github.com/FnTm/latvian-tweet-sentiment-corpus

[6] https://github.com/nicemanis/LV-twitter-sentiment-corpus

Špats un Birzniece [14] lietoja Pumpura izveidoto leksikonu[7], kas satur sarakstu ar vārdiem, kam ir pozitīva vai negatīva noskaņa. Špats izmantoja Peisenieka un Skadiņa izveidoto korpusu, un arī izveidoja jaunu latviešu valodā rakstīto Twitter sīkziņu korpusu. Tika ievākti 90000 sīkziņu un tās novērtētas kā pozitīvas vai negatīvas izmantojot emocijzīmes. Sīkziņas, kas nesatur emocijzīmes un jebkādu sentimentu no Špata un Birznieces izveidotā leksikona[8] tiek uzskatītas par neitrālām. Novērtēšanas rezultātā tika iegūts korpuss, kas sastāv no 5556 trokšņaini novērtētām sīkziņām.

2.2. Secinājumi

Kā redzams, lai veiktu sentimenta analīzi tekstiem latviešu valodā, lielākā daļa no apskatīto darbu autoriem izveidoja datu kopas izmantojot Twitter sīkziņas [25],[15],[26],[27],[14]. Latviešu valodā pieejamie anotētie tekstu korpusi ir nelieli, pat kombinējot tos, tādēļ pētnieki vai nu lieto trokšņaini anotētus korpusus, vai arī izmanto leksikonā balstītās pieejas, kurām nav nepieciešama liela anotēta korpusa esamība.

[7] https://github.com/pumpurs/SentimentWordsLV

[8] https://github.com/gatis/om/tree/master/lexicon

3. DATU PRIEKŠAPSTRĀDE UN FAKTORU IZVĒLE

Noskaņojuma analīze ir klasificēšanas problēma. Klasifikatora lietošanai nepieciešams klasificējamo objektu reprezentēt kā faktoru vektoru. Tā kā tīmeklī publicētie dokumenti ir dažādu lietotāju rakstīti un satur kļūdas, transliterāciju, kā arī ir rakstīti dažādos stilos un valodās, ir nepieciešams lietot priekšapstrādes soļus pirms dokumentu pārveidošanas faktoru vektorā, kas ir izmantojams ar attiecīgo klasifikācijas metodi. Papildus tam, piemēram, Twitter sīkziņas satur tēmturus (apzīmēti ar "#"), lietotāju vārdus (apzīmēti ar "@"), atkārtotas sīkziņas (tās satur apzīmējumu "RT") un tīmekļa saites, t.i. vienumus, kas nesatur informāciju par emociju kas tiek pausta tekstā.

3.1. Teksta priekšapstrāde

Priekšapstrāde tiek lietota lai attīrītu un normalizētu tekstu. Datu sagatavošanas pirmajā solī daži no iegūtajiem tekstiem var tikt atmesti pilnībā:

- Automātiski ģenerētie teksti – sīkziņu gadījumā potenciāli nozīmīgs solis, jo automātiski ģenerētas sīkziņas tiek ģenerētas lielā skaitā, un tām ir aptuveni vienāds saturs. Piemēram, sīkziņa, kas tiek ģenerēta draugiem.lv atzīmējot man patīk: „Atzīmēja Man patīk galerijā:...".
- Teksti, kas nav izvēlētajā mērķa valodā – šis solis svarīgs, ja izvēlēts tekstu avots, kurā var būt teksti dažādās valodās (piemēram, sociālie tīkli). Šis solis tiek veikts tādēļ, ka noskaņojuma analīzei kādā valodā, svešvalodu vārdi lielākoties ir troksnis, vai arī tiek atpazīti nepareizi.

Pēc nederīgo tekstu atmešanas, kā arī stāvoklī, kad noskaņojuma analīzes sistēma ir gatava klasificēt jaunus tekstus, vispārīgā gadījumā, teksta priekšapstrāde var saturēt šādas darbības[28]:

- HTML tagu dzēšana – ja tekstā ir jebkādi HTML tagi, piemēram, „ ", "
", "<sub>" utml., tie tiek izdzēsti;
- Skaitļu dzēšana, vai aizvietošana ar īpašu vietturi – skaitļi, kas atrodami tekstos parasti ir unikāli un tādēļ nav derīgi kā faktori [29];
- Pieturzīmju dzēšana;
- Burtu reģistra vienādošana – vai nu uz visiem mazajiem, vai lielajiem;

- Stopvārdu dzēšana – stopvārdi ir vārdi, kam ir maza nozīme un tādēļ tiek uzskatīts, ka tos var dzēst;
- Saknes/pamatformas atrašana – tiek pielietota, lai samazinātu locīto vai paplašināto vārdu formas uz to pamatformu – celmu vai pamatformu. Šis solis ir sevišķi svarīgs tādās valodās, kā latviešu un krievu, kurās vārdiem ir vairāki locījumi, piedēkļi un priedēkļi, kas multiplikatīvi palielina unikālo vārdu skaitu (tātad arī iespējamo faktoru skaitu) padarot maz aizpildīta faktoru vektora un trokšņaina teksta (sk.piem. [7]) problēmu sevišķi aktuālu;
- Transliterācijas aizvietošana – latviešu alfabētā ir burti, kas nav atrodami angliskajās klaviatūrās, tādēļ lietotāji mēdz rakstīt izmantojot transliterāciju (burta „ā" vietā izmantoti divi „a" – "aa", „č" vietā lietotāji mēdz rakstīt „ch" u.t.t.). Garkāje et al. [13] aizvietoja transliterācijas simbolus ar pareizo latviešu valodas burtu, izmantojot īpašus likumus;
- Lietotājvārdu dzēšana – sīkziņu tekstiem aktuāls solis, jo lietotājvārdi (kas sākas ar simbolu „@") ir unikāli un tādēļ nav lietderīgi tos lietot kā faktorus;
- Atsauces tagu dzēšana/aizvietošana – arī šis ir sīkziņu tekstiem aktuāls solis, jo tēmturi (sākas ar simbolu „#") lielākoties nav lietojami kā faktori. Tie tiek dzēsti, vai arī aizvietoti ar vārdiem, ja tie ir atpazīstami kā vārdi, izmantojot vārdnīcu [26];
- Saišu dzēšana/aizvietošana – saites tekstos parasti ir unikālas un tādēļ nav lietderīgi tās lietot kā faktorus. Tās tiek vai nu dzēstas vai aizvietotas ar vietturi;
- Teikumu sadalīšana tekstvienībās (*token*) – šis solis sadala teikumu vārdu vektorā vai simbolu vektorā, ko tālāk pielieto kā faktorus faktoru vektorā.

3.2. Faktoru vektora izveidošana

Priekšapstrādes rezultāts ir tīra normalizēta tekstvienību virkne, kas tālāk var tikt pārveidota par faktoru vektoru. Lai veiktu pārveidošanu apskatīto darbu autori lietoja vai nu "vārdu somas" pieeju (lielākā autoru daļa), vai arī vārdu iegulšanu (*word embedding*) [26],[29], vai arī zīmju līmeņa iegulšanu (*character-level embedding*) [30]. Papildus teksta faktori tiek apskatīti [31], kā, piemēram, izsaukuma zīmju skaits, neķītras valodas esamība tekstā, vai [32] – vārdu skaits, izsaukuma zīmju skaits, citu lietotāju pieminējumi (@) u.c. Gediņš [27] kā faktoru izmanto īpašu pazīmi, kas aizvieto reti lietotos pozitīvos vai negatīvos

vārdus tekstā. Vairāki autori arī faktoru vektorā iekļauj emocijzīmes, vai nu tiešā veidā [33], vai arī kā atsevišķu faktoru, kas norāda uz emocijzīmju skaitu [26],[27],[32]. Shalunts un Backfried [33] izmantoja izsaukuma zīmes, atkārtotus burtus un burtu ar lielo reģistru esamību kā papildus „pastiprinātājus", kas pastiprina atrasto noskaņu. Gulbinskis [34] meklēja specifiskas frāzes, kas satur informāciju par noskaņu dotajā tekstā. Lai iegūtu šādas frāzes, īpaši frāžu likumi tika izveidoti un padoti uz SemTi-Kamols[9] morfoloģiskās analīzes rīku, ar kura palīdzību tika izgūtas frāzes, kas atbilst izveidotajiem likumiem. Bobichev et al. [30] izmēģināja divas metodes lai iegūtu visinformatīvāko faktoru kopu: 1) „Correlation-based Feature Subset Selection", kas novērtē faktorus pēc to individuālās paredzēšanas spējas kopā ar dublēšanās starp faktoriem novērtējumu; 2) „Information gain evaluation", kas novērtē faktoru atbilstoši iegūtajai informācijai attiecībā pret klasi. Abas metodes uzrādīja līdzīgus rezultātus un bija labākas nekā ja tiktu izmantoti visi vārdi, bez jebkādas faktoru izvēles.

3.3. Secinājumi

Kā redzams, faktorus, ko viens pētnieks uzskata par liekiem, cits pētnieks iekļauj kā daļu no faktoru vektora, un, tā rezultātā, izmantoto faktoru saraksts stipri atšķiras starp dažādiem darbiem un līdz šim veiktajos pētījumos, kas veic noskaņojuma analīzi tekstiem latviešu valodā nav veikta priekšapstrādes soļu lietderīguma vērtēšana. Tas ļauj secināt, ka teksta priekšapstrāde un faktoru izvēle ir joma, kurā vajadzīgi turpmāki pētījumi, tādēļ 5.3. nodaļā tiek veikta dažādu priekšapstrādes metožu salīdzināšana.

[9] http://www.semti-kamols.lv/?sadala=220

4. KLASIFICĒŠANAS METODES

Šajā nodaļā tiek apskatītas klasificēšanas metodes, kuras var izmantot, lai klasificētu teksta emocionālo noskaņu.

Noskaņojuma analīzes uzdevums būtībā ir teksta klasificēšanas uzdevums, kur doto tekstu nepieciešams klasificēt kā piederošu vienai no klasēm (piemēram – pozitīvs, negatīvs, neitrāls). Klasificēšanas metodes var iedalīt mašīnapmācības pieejās un leksikonā balstītās pieejās. Mašīnapmācības pieejas sākotnēji izveido modeli ar apmācības datiem un tad pielieto to mērķa datiem, savukārt leksikonā balstītās pieejas iegūst leksikonu un izstrādā klasificēšanas metodi, kas teksta klasificēšanai ņem vērā tekstā atrasto vārdu ar zināmu noskaņu klātbūtni. Šīs metodes tiek apskatītas tālākajās nodaļās.

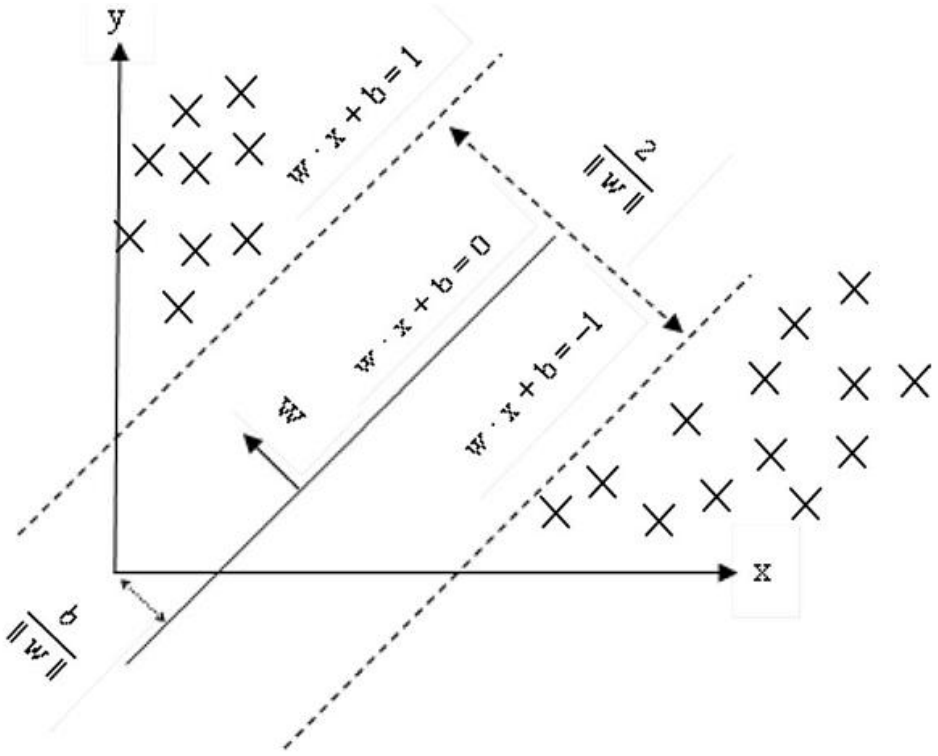
4.1. Pārraudzītā mašīnapmācība

Pārraudzītā mašīnapmācība izmanto apmācības datu kopu, kurā ir norādīts katra dokumenta klasificēšanas rezultāts. Noskaņojuma analīzes uzdevumos šie mašīnapmācības algoritmi tiek plaši lietoti, tomēr tiek uzskatīts, ka šādi apmācīts klasifikators piemērojas jomai, kura visvairāk pārstāvēta apmācības datos un pielietojot iegūto klasifikatoru citās jomās, tiek iegūts vājš rezultāts [35], jo viens un tas pats vārds pielietots dažādās jomās vai kontekstos var izteikt atšķirīgu noskaņu un viedokli (piemēram, runājot par telefonu: „ilgi ielādējas" un „baterija tur ilgi" izsaka atšķirīgu noskaņu). Veicot literatūras apskatu [7][28], tika secināts, ka noskaņojuma analīzē visbiežāk lietotie pārraudzītās mašīnapmācības algoritmi ir Naivā Beijesa (*Naive Bayes*) klasifikators, atbalsta vektoru mašīnas (*SVM – Support Vector Machines*), mākslīgie neironu tīkli (*Artificial Neural Networks*) un maksimālās entropijas klasifikators (*Maximum entropy classifier*) jeb loģistiskās regresijas klasifikators (*logistic regression*).

Naivā Beijesa klasifikators – (*Naive Bayes*) klasifikators aprēķina dokumenta piederību klasei izmantojot „vārdu somas" pieeju, kur tiek pieņemts, ka vārdu secība nav svarīga, pieņemot, ka visi vārdi teikumā ir neatkarīgi viens no otra. Šie pieņēmumi padara klasifikatoru ļoti vienkāršu, bet efektīvu. Tas spēj darboties jomās, kurās ir daudz vienādi svarīgu faktoru, un tas bieži tiek lietots kā pamata mašīnapmācības metode, kam ir labi rezultāti. Naivā Beijesa klasifikators balstās uz Beijesa teorēmu, un tas ļauj izteikt varbūtību, ka dokuments d pieder klasei c kā $P(c|d) = \frac{P(d|c)*P(c)}{P(d)}$. Tad $c_{MAP} = argmax_{c \epsilon C} P(c|d) =$

$argmax_{c\epsilon C}\frac{P(d|c)*P(c)}{P(d)} = argmax_{c\epsilon C}P(d|c) * P(c)$; MAP – „maximum a posteriori" visvairāk iespējamā klase, saucējs $P(d)$ tiek atmests, jo tas visām klasēm ir vienāds. Izsakot $P(d)$ kā faktoru $x_1, x_2, \dots, x_n$ kopu iegūstam $c_{MAP} = argmax_{c\epsilon C}P(x_1, x_2, \dots, x_n|c)P(c)$.

SVM (Support Vector Machine) - Atbalsta vektoru mašīnas ir mašīnapmācības metode, kura cenšas atrast hiperplakni daudzdimensiju telpā, kas vislabāk atdala klasificējamās objektu klases (sentimenta analīzes gadījumā dimensijas ir apskatāmo tekstu faktoru vektora elementi). SVM ir pārraudzītas apmācībās metode, kas izmanto apmācības datus lai atrastu optimālo hiperplakni. 4.1. attēlā attēlota SVM koncepta ilustrācija, kur ar izteiksmi w * x + b = 0 tiek attēlota hiperplakne.

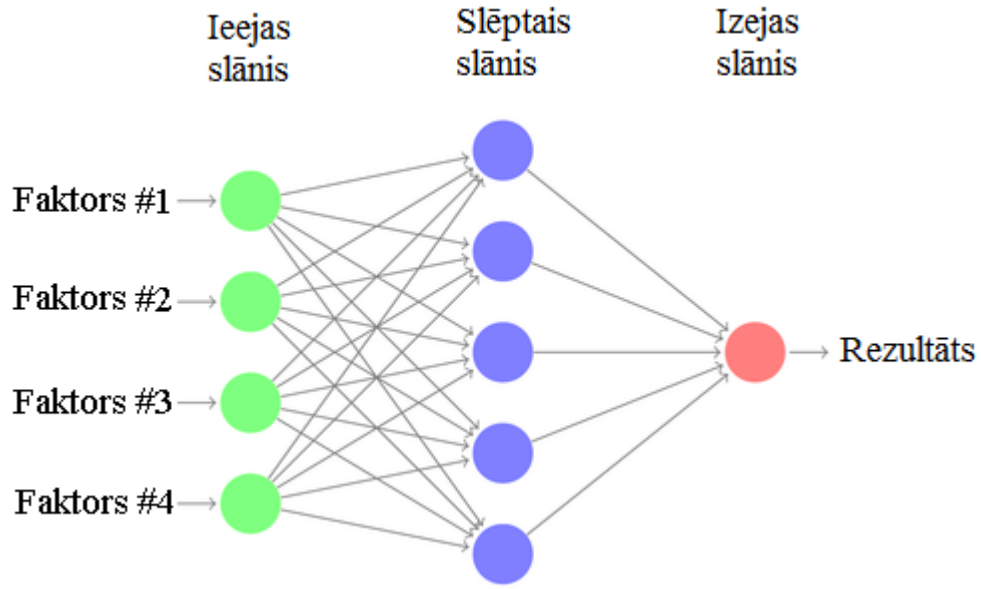


4.1. att. SVM koncepta ilustrācija, kur ar izteiksmi w*x+b=0 tiek attēlota hiperplakne

Apmācības kopu, kas sastāv no n punktiem, pieraksta formā (x_1, y_1), ... , (x_n,y_n), $x \in R^n$,$y \in \{1,-1\}$, kur x ir ieejas faktoru vektors un y ir ieejas punkta piederība klasei ar vērtējumu 1 vai arī -1. Ja apmācības dati ir lineāri sadalāmi, tad ir iespējams atrast divas hiperplaknes, kas atdala klases vienu no otras un attālums starp taisnēm ir maksimālais. Šīs hiperplaknes apraksta ar formulām w*x +b =1 un w*x+b=-1. Attālums starp plaknēm ir $\frac{2}{||w||}$, tādēļ ir jācenšas minimizēt lielumu $||w||$. Turklāt visiem punktiem x jāatrodas ar pareizajā hiperplaknes pusē, tādēļ tiem ir jāizpilda nosacījums w*x+b>= 1 ja y = 1 vai w*x+b<=-1 ja y=-1. Tad var uzrakstīt klasifikatoru $f(x) = sgn((w,x) + b)$. Hiperplakne ar lielāko

attālumu no objektu klasēm tiek pilnībā noteikta ar punktiem x, kas atrodas tai vistuvāk, šie *x* tad arī tiek saukti par atbalsta vektoriem.

Mākslīgais neironu tīkls (*Artificial Neural Network*) sastāv no savā starpā savienotiem elementiem, sauktiem par neironiem, kas izvietoti vienā vai vairākos slāņos. Katrs neirons ir saistīts ar citiem neironiem caur sinapšu attiecībām. Neirons saņem ieejas signālus no citiem neironiem un rada izejas signālu, kas tiek novadīts uz citiem neironiem. Katram ieejas signālam ir piesaistīts svars ar ko tas tiek reizināts, un katra neirona ieejas signāli tiek apkopoti un novērtēti ar kādu funkciju, iegūstot izejas signālu. Ieejas signāli (faktoru vērtības) tiek padoti ieejas slānī esošajiem neironiem, un rezultāts tiek saņemts no izejas slānī esošajiem neironiem.



4.2. att. Neironu tīkla sastāvdaļas.

Katras sinapses (pelēkās līnijas 4.2. attēlā) svara noteikšanai tiek lietos apmācības process, kurš var būt pārraudzīts (4.2. att.) - (ir doti gan ieejas faktori, gan rezultāts) vai nepārraudzīts (tiek doti tikai ieejas faktori). Neironu tīkls [36] ļauj izmantot neskaitliskus datus, kas var būt nelineāri un darbojas arī ar trokšņainiem (nepilnīgiem) datiem un liela faktoru skaita. Neironu tīklu vājās puses ir grūtības noteikt sinapšu svaru sākumvērtības sākot apmācību, apmācībai vajag ilgu laiku, faktori ir uzmanīgi jāizvēlas, kā arī grūti izskaidrot modeļa iegūto rezultātu, jo modelis ir ”melnā kaste” [37].

Neironu tīklus iespējams arī kombinēt, kā to dara Xu un Mao [38] – izmanto trīs dziļos neironu tīklus (bildēm, tekstam un apvienojošo tīklu), kas ļauj apvienot informāciju no teksta un attēla (ja tāds bija pieejams kopā ar tekstu), lai uzlabotu prognozes precizitāti. Tā piemēram, 4.3. attēlā (a) ir pozitīva emocija gan tekstā, gan attēlā, savukārt (b) ir negatīva

emocija gan attēlā gan tekstā, tomēr (c) teksts ir neitrāls, taču pēc attēla [REFUSE] ir iespējams noteikt negatīvu emocionālo noskaņu.

4.3. att. Teksts ar attēliem

Maksimālās entropijas klasifikators (*Maximum entropy classifier*) jeb loģistiskās regresijas klasifikators (*logistic regression*) atšķirībā no Naivā Beijesa klasifikatora nepieņem, ka faktori ir neatkarīgi, bet faktoriem tiek piešķirti papildu parametri (svari), kas norāda uz kādām apmācības datos atklātajām īpatnībām, kam būtu jāparādās arī modelī. Tā piemēram, ja četru klašu teksta klasificēšanas uzdevumā zināms, ka vidēji 40% no dokumentiem, kas satur vārdu „profesors" pieder pie fakultātes dokumentu klases, tad saņemot dokumentu, kas satur vārdu „profesors", var pieņemt, ka pastāv 40% varbūtība, ka tas pieder fakultātes dokumentu klasei un 20% varbūtība, ka tas pieder jebkurai citai no klasēm. Bez šādas papildu informācijas, tiktu pieņemts, ka dokuments var piederēt jebkurai no klasēm ar 25% varbūtību. Kā šādu papildu informāciju var būt tikt izmantots vārdu skaits, jo ja kāds vārds sevišķi bieži ir sastopams kādas klases dokumentos, tad šādam vārda-klases pārim svars būs lielāks nekā šī paša vārda un citas klases pārim [39].

4.1.1. Latviešu valodā rakstīto tekstu noskaņojuma analīzei izmantotās metodes

Garkaje et al. [13] izmantoja Naive Bayes klasifikatoru. Anotējot datus tika novērots, ka anotētāju viedokļi par dotā teksta klasi sakrīt 78% gadījumu, kas arī tika uzskatīts par klasifikatora precizitātes augstāko robežu. Teksti tika klasificēti divās klasēs: agresīvi un neagresīvi. Labākie klasificēšanas rezultāti tika sasniegti, izmantojot tekstu, kas tika normalizēts un transliterācija pārveidota – akurātums 72.2% un F1 vērtība 33.1% agresīvajai klasei un F1 vērtība 82.4% neagresīvajai klasei.

Nicmaņa [26] izmantotais korpuss ar sīkziņām latviešu valodā bija pārāk mazs un rezultāti, kas tika iegūti apmācot neironu tīklu ar šo korpusu bija vāji un Nicmanis izlēma nepublicēt iegūtos rezultātus, bet neironu tīkla apmācībai izmantot angļu valodas korpusu.

Gediņš [27] apmācīja Naivā Beijesa klasifikatoru izmantojot vārdu unigrammas, bigrammas, unigrammas kopā ar bigrammām un unigrammas ar morfoloģisko informāciju. Tika iegūta akurātums attiecīgi 70.1%, 66.1%, 70.8% un 71.4%. Rezultātu pasliktināšanās izmantojot bigrammas tiek skaidrota ar palielināto faktoru skaitu, kam nesatur informāciju par noskaņojumu. Tika atklāts, ka 13000 sīkziņu latviešu valodā sastāvēja no 75454 unikāliem vārdiem, kamēr apmēram tāda paša izmēra korpuss angļu valodā satur no 4227 līdz 9045 unikāliem vārdiem. Tālākais darbs fokusējās uz biežāk sastopamo faktoru ar lielāko izteiksmīgumu atrašanu. Izmantojot Paikena izstrādāto rīku „Morphological analyzer for Latvian language"[10] tika iegūtas vārdu saknes un modelis tika vēlreiz apmācīts izmantojot vārdu saknes kā faktorus, kā rezultātā tika iegūta akurātums 69%. Labākie rezultāti tika sasniegti atmetot vārdus, kas korpusā ir sastopami reti (mazāk nekā n reizes). Tika noskaidrots, ka labākie rezultāti tiek iegūti, ka atmet vārdus, kas korpusā atkārtojas mazāk nekā 7 reizes, samazinot izmantoto faktoru (vārdu) skaitu no 75454 līdz 5260, vienlaikus sasniedzot akurātumu 75.8%.

Peisenieks un Skadiņš [15] tulkoja tekstus latviešu valodā uz angļu valodu izmantojot Google Translate[11], Bing Translator[12] un Tilde Translator[13]. Pēc tulkošanas tekstu noskaņojums tika noteikts izmantojot trīs tiešsaistes noskaņojuma analīzes rīkus: Alchemy API[14], Textalytics[15] un Semantria[16]. Visas deviņas kombinācijas (trīs tulkotāji un trīs noskaņojuma analīzes rīki) tika novērtētas un salīdzinātas. Peisenieks un Skadiņš secina, ka šāda pieeja nodrošina kopējo precizitāti no 45.6% līdz 76%, un labākā kombinācija ir Bing Translator kopā ar Alchemy API – sasniedzot labāko kopējo precizitāti. Tomēr visas izmēģinātās rīku kombinācijas uzrādīja sliktus rezultātus klasificējot neitrālas sīkziņas –

[10] https://github.com/PeterisP/morphology

[11] https://translate.google.lv/

[12] https://www.bing.com/translator

[13] https://translate.tilde.com/en

[14] https://www.alchemyapi.com

[15] https://www.meaningcloud.com/

[16] https://www.lexalytics.com/

kopējā precizitāte bija no 21.3% līdz 35.5%; arī šajā reizē labākos rezultātus uzrādīja Bing Translator un Alchemy API kombinācija.

4.1.2. Citās valodās rakstīto tekstu noskaņojuma analīzei izmantotās metodes

Gunther [3] eksperimentēja ar dažādu priekšapstrādes metožu izmantošanu Twitter sīkziņu analīzei un izmantoja scikit-learn[17] bibliotēkā implementētos Naivā Beijesa un SVM mašīnapmācības algoritmus. Kā galīgais mašīnapmācības modelis tika izvēlēts SVM un faktoru vektors tika izveidots pielietojot īpaši izstrādātu teksta sadalīšanas skriptu tekstvienībās, teksta normalizāciju apvienojot atkārtotus burtus, veicot celmošanu, negācijas marķēšanu, tiek lietoti 4 dažādi leksikoni lai atrastu attiecīgās sīkziņas polaritāti, ko pievieno faktoru vektoram, papildus faktoru vektorā tiek atzīmēta iepriekš definētu vārdu kopumu esamība tekstā un izmantojot uni- un 2-grammas tiek apmācīts galīgais modelis, kas trīs klašu klasificēšanā sasniedz 74.56 F1 mēru un 74.90 akurātumu.

Galinsky et al. [24] izmantoja rakstzīmju līmeņa iegulšanu ar dziļo neironu tīklu. Izmantojot nepapildināto apmācības korpusu, tika sasniegta akurātums 84.6% pirmajai kopai un 71.6% otrajai kopai. Izmantojot apmācības korpusu, kas tika papildināts ar sinonīmiem(skat 2 nodaļā iepriekš), tika sasniegta precizitāte 87.0% pirmajā kopā un 70.2% otrajā testa kopā.

Bobichev et al. [30] eksperimentēja ar vairākiem mašīnapmācības algoritmiem – SVM, Bernulli Naivā Beijesa, Multinomial Naivā Beijesa, Diskriminatīvo Naivā Beijesa. Labākie rezultāti tika sasniegti lietojot Bernulli Naivā Beijesa algoritmu – F1 mērs 84.5% ekonomikas domēnam, 80.2% sociālajam domēnam un 81.7% sporta domēnam.

Loukachevich un Rubtsova [40] apkopoja rezultātus izmantojot dažādas klasificēšanas metodes (Atbalsta vektora mašīnas – SVM, maksimālās entropijas klasifikatoru, likumos balstītu klasifikatoru) trīs klašu klasificēšanas (pozitīvs/neitrāls/negatīvs) problēmai. Labākie sasniegtie rezultāti novērtējo ar F1 rādītāju bija 48.8% telekomunikāciju jomā lietojot SVM, un 36.0% banku jomā lietojot maksimālās entropijas klasifikatoru un vārdu n-grammas, simbolu n-grammas un jomas modelēšanas rezultātus. Neatkarīgu ekspertu novērtēšanā tika iegūta F1 mēra vērtība 70.3% telekomunikāciju jomā, kas tika uzskatīta par maksimālo iespējamo rezultātu šāda tipa uzdevumos. Zemās sasniegtās F1 vērtības tika skaidrotas ar uzdevuma sarežģītību un apmācības kopas ierobežoto izmēru.

[17] http://scikit-learn.org/stable/

Loukachevitch un Chetviorkin [31] ziņo, ka labākie sasniegtie rezultāti atvērtās novērtēšanas uzdevumā tika sasniegti izmantojot SVM un maksimālās entropijas klasifikatorus. Rezultāti trijās jomās klasificējot divās klasēs - akurātums 83.1%-96.1%, trijās klasēs – akurātums 69.4%-75.2%, un klasificējot piecās klasēs: 40.7%-51.3%; F1 divām klasēm 66.9%-71.5%, F1 trim klasēm 48.0%-56.0%, F1 klasificējot piecās klasēs: 33.6%-40.2%.

Sakenovich [29] lietoja Long Short-Term Memory (LSTM) rekurento neironu tīklu. LSTM ņem vērā vārdu atkarību vienam no otra virknē, kā arī tas atrisina gradienta pārliecīgas samazināšanās vai palielināšanās problēmu, kas bieži tiek novērota rekurentajos neironu tīklos. Tika salīdzināti vairāki LSTM veidi un labāko darbību uzrādīja divu līmeņu LSTM ar vidējo precizitāti 84.5% , pārklājumu 86.4% un akurātumu 86.3%.

Tutubalina un Nikolenko [41] izstrādāja metodi aspektam pielāgota leksikona iegūšanai. Aspektam tiek iegūti šādi faktori: zīmju mazajā reģistrā n-grammas, leksikona-balstītas unigrammas, konteksta unigrammas un bigrammas, aspekta balstītas bigrammas, kā arī leksikona-balstīti faktori: max un min noskaņojuma vērtējums, kopējais un vidējais vārdu noskaņojuma vērtējumi. Tika salīdzināts manuāli apkopots vispārīgs leksikons ar maksimālās entropijas klasifikatoru, kas tika veidots izmantojot ģenerēto leksikonu. Modeļi, kas lietoja ģenerēto leksikonu uzrādīja vidēji 1-2% uzlabojumu salīdzinot ar modeli, kas izmantoja manuāli veidoto leksikonu. Tika sasniegta precizitāte 74.8.% , pārklājums 66.3% un F1 mērs 69.1%, turpretī izmantojot vispārīgo leksikonu tika iegūti šādi rezultāti: precizitāte 73.8%, pārklājums 65.7% un F1 67.6%.

4.2. Nepārraudzīta mašīnapmācība

Atšķirībā no pārraudzītās mašīnapmācības, kur nepieciešams liels daudzums iepriekš klasificētu dokumentu, nepārraudzītā mašīnapmācība ļauj izmantot iepriekš neklasificētus dokumentus. Metodes priekšrocība ir iespēja atteikties no iepriekš klasificētu dokumentu korpusa.

Liu [2] aprakstīta viena no šādām metodēm, kas veic klasificēšanu pamatojoties uz fiksētu sintaktisku paraugu atrašanu, kas visticamāk tiek lietoti, lai izteiktu noskaņojumu. Sintaktiskie paraugi (frāzes) tiek veidoti izmantojot vārdšķiru iezīmes. Algoritms sastāv no 3 soļiem:

1. Divi vai vairāki vārdi (frāzes) tiek izgūti, ja to vārdšķiru iezīmes atbilst kādam no iepriekš definētiem paraugiem, piemēram, „īpašības vārds, lietvārds”.

2. Tiek novērtēta izgūto frāžu noskaņojuma orientācija lietojot PMI (pointwise mutual Information) mēru: $PMI(term_1, term_2) = log_2(\frac{Pr(term_1 \wedge term_2)}{Pr(term_1)\,\Pr(term_2)})$. PMI novērtē cik lielā mērā divi vārdi ir statistiski atkarīgi. Šeit $Pr(term_1 \wedge term_2)$ ir patiesā varbūtība, ka divi vārdi būs atrodami kopā, bet $Pr(term_1)\,\Pr(term_2)$ - varbūtība, ka divi vārdi ir statistiski neatkarīgi. Noskaņojuma orientācija novērtē frāzes asociāciju ar kādu pozitīvu atskaites vārdu, piemēram, "lielisks", vai negatīvu vārdu, piemēram, „slikts". Tad $\text{noskaņojuma orientācija(frāze)} = \text{PMI(frāze,"lielisks")} - \text{PMI(frāze,„slikts")}$. PMI aprēķinam vajadzīgās varbūtības var tikt iegūtas izmantojot tīmekļa meklētāju un attiecīgu vaicājumu, kā tas izdarīts [34].
3. Tiek atrastas definētajiem likumiem atbilstošās frāzes un noskaņojuma orientācijas vērtības. Noskaņojuma orientācijas vērtībām tiek atrasta vidējā un, ja tā ir pozitīva, teksts tiek klasificēts kā pozitīvs un, ja tā ir negatīva, teksts tiek klasificēts kā negatīvs.

Gulbinskis [34] izmantoja Pointwise Mutual Information and Information Retrieval (PMI-IR) algoritmu [42] lai analizētu dažādu tīmekļa vietņu rakstu un komentāru noskaņojumu. PMI-IR algoritmam nav vajadzīgi apmācības dati, tā vietā tiek lietoti tīmekļa meklētāja atgrieztie rezultāti vaicājumam, kas satur doto frāzi un kādu no noskaņojumu izsakošajiem vārdiem (piemēram, "labi" vai "slikti"). Precizitāte divām klasēm „pozitīvs" un „negatīvs" tika sasniegta 75%.

4.3. Leksikonā balstīta pieeja

Leksikonā balstītās pieejas pamatojas uz pieņēmumu, ka dokumenta vai teikuma emocionālās noskaņas klasi var definēt kā atsevišķo tekstā esošo vārdu klašu apkopojumu. Metodes darbībai nav vajadzīgi nekādi apmācības dati, jo katrs analizējamā teksta vārds tiek salīdzināts ar vārdnīcu un apstrādāts atsevišķi, ignorējot saites starp vārdiem. Vārdnīca tiek veidota neatkarīgi no apmācības datiem, tādēļ nevar iestāties pārmērīga pielāgošanās. Gadījumos, ja vārds nav atrasts vārdnīcā, to var meklēt izmantojot tīmekļa meklētāju, un no iegūtajiem rezultātiem noteikt tā noskaņojuma klasi. Papildus noskaņojuma klasei, vārdnīcā var glabāt arī pozitīvos vai negatīvos svarus, kas norāda uz vārda svarīgumu [43].

Špats un Birzniece [14] salīdzināja Naivā Beijesa metodi ar leksikona balstītu pieeju. Izmantojot Naivā Beijesa metodi, tika sasniegta 62% kopējā precizitāte trenējot ar cilvēka anotētiem tekstiem un 55% vidējā precizitāte apmācot ar trokšņaini anotētiem tekstiem. Izmantojot leksikona pieeju tika sasniegta 73% kopējā precizitāte.

Shalunts un Backfried [33] izstrādāja leksikona balstītu klasificēšanas metodi SentiSAIL. Metode implementē trīs algoritmus noskaņojuma vērtējumu aprēķināšanai: maksimizācija – pozitīvie un negatīvie vērtējumi tiek ņemti vērtējumi no vispozitīvākajiem un visnegatīvākajiem vārdiem; vidējošana - pozitīvie un negatīvie vērtējumi tiek atrasti kā vidējais no tekstā esošo vārdu vērtējumiem; agregācija – pozitīvie un negatīvie vērtējumi tiek atrasti agregējot tekstā esošo terminu vērtējumus, ierobežojot ar 5 pozitīvām vērtībām un ar -5 negatīvajām vērtībām. Teksta vērtējums tiek aprēķināts kā vidējais visu teikumu vērtējums dokumentā. Galīgā noskaņojuma klase tiek piešķirta izmantojot slieksni, t.i. ja vērtējuma vērtība pārsniedz kādu vērtību, dokumentam tiek piešķirta attiecīgā klase. SentiSAIL sasniedza vidējo testa kopas precizitāti 90%.

4.4. Secinājumi

4.1. tabula. Latviešu un citās valodās izmantoto metožu apkopojums

Klasificēšanas metode, ar kuru iegūti labākie rezultāti	Valoda	Datu avoti un joma	Klases	Iegūtais rezultāts	Piezīmes	Atsauce
Naivā Beijesa klasifikators	Latviešu	Ziņas no jaunumu portāliem	Agresīvs, Neagresīvs	Akurātums 72.2%, agresīvajai klasei F1 32.9%, neagresīvajai klasei F1 82.4%	Starpvērtētāju viedokļu sakritība 78%	Garkāje et al. [13]
Mašīntulkošana un tīmekļa rīku izmantošana angļu valodai	Latviešu valodas tekstu tulkojumi angļu valodā	Twitter	Pozitīvs, (Neitrāls), Negatīvs	Akurātums 76% bez neitrālās klases un 35.5% iekļaujot neitrālo klasi	Labākais rezultāts sasniegts lietojot Bing Translator + AlchemyAPI	Peisenieks and Skadiņš [15]
PMI-IR algoritms	Latviešu	Emuāri, ziņas, Twitter	Pozitīvs, Negatīvs	Akurātums 75%		Gulbinskis [34]
Naivā Beijesa klasifikators	Latviešu	Twitter	Pozitīvs, Negatīvs	Akurātums 75.8%		Gediņš [27]
Leksikona balstīta un Naivā Beijesa klasifikators	Latviešu	Twitter	Pozitīvs, Neitrāls, Negatīvs	Akurātums 73% izmantojot leksikona balstīto metodi, 62% Naivais Beijess	Naivā Beijesa precizitāte 55% izmantojot trokšņaini anotētos datus	Špats un Birzniece [14]
SVM, maksimālās entropijas klasifikators, Likumos balstīts klasifikators	Krievu	Twitter (telekomunikāciju un banku jomas)	Pozitīvs, Neitrāls, Negatīvs	F1 48.8% telekomunikāciju domēnam; F1 36.0% banku domēnam	Ekspertu novērtējuma F1 70.3%. Bāzlīnija: 18% telekomunikāciu, 13% banku jomai	Loukachevich un Rubtsova [40]
SVM, maksimālās entropijas klasifikators	Krievu	Filmu, grāmatu, fotoaparātu atsauksmes	2, 3 and 5 noskaņojuma klases	Akurātums: 2 klases 83%–96%, 3 klases 69%–75%, 5 klases 41%–51%	F1: 2 klasēm 67%–72%, 3 klasēm 48%–56%, 5 klasēm 34%–40%	Loukachevich un Chetviorkin [31]
LSTM rekurents	Krievu	Ziņas no ziņu	Pozitīvs,	Precizitāte 84.5%,		Sakenovich [29]

neironu tīkls		portāliem	Neitrāls, Negatīvs	pārklājums 86.4%, akurātums 86.3%		
Leksikona balstīta pieeja ar maksimālās entropijas klasifikatoru	Krievu	Restorānu atsauksmes	Pozitīvs, Neitrāls, Negatīvs	Precizitāte 74.8%, pārklājums 66.3%, F1 69.1%	Bāzlīnija: precizitāte 73.8%, pārklājums 65.7%, F1 67.6%	Tutubalina and Nikolenko [41]
Leksikona balstīta (SentiSAIL)	Krievu	Ziņas no ziņu portāliem	Pozitīvs, Neitrāls, Negatīvs	Akurātums 90%	Starpvērtētāju viedokļu sakritība 92.7%	Shalunts and Backfried [33]
Dziļais neironu tīkls	Krievu	Restorānu un produktu atsauksmes	Pozitīvs, Neitrāls, Negatīvs	Precizitāte 87.0% pirmajā testa kopā un 71.6% otrajā testa kopā		Galinsky et al. [24]
SVM and trīs veidu Naivais Beijess	Krievu	Ziņas no ziņu portāliem	Pozitīvs, Neitrāls, Negatīvs	F1 80.2%–84.5%, atkarībā no domēna		Bobichev et al. [30]
SVM	Angļu	Twitter	Pozitīvs, Neitrāls, Negatīvs	Akurātums 74.9% F1 74.56%	Bāzlīnija Akurātums 68.87%, F1 67.47%	Gunther T. [3]

Tabulā 4.1. redzams apskatītajos darbos izmantoto metožu apkopojums (publicēts arī [28]) – izmantotās klasificēšanas metodes, klases, kā arī iegūtie rezultāti. Kā redzams, visi autori izmantoja mašīnapmācības algoritmus izņemto autorus [14],[41],[33] kas izmantoja arī leksikonā balstītu pieeju. Angļu valodā līdzīgs pētījums veikts [7], kur apkopotas populārākās izmantotās klasificēšanas metodes angļu valodā – SVM, leksikonā balstītas, Naivais Beijess, mākslīgie neironu tīkli, lēmumu koki un loģistiskā regresija u.c.

Latviešu valodā pieejamais nelielais anotēto tekstu daudzums neļauj pētniekiem izmantot neironu tīklus vai citas metodes, kas prasa lielu apmācības korpusu, tādēļ latviešu valodā rakstīto tekstu analīzei visbiežāk ticis lietots Naivā Beijesa klasifikators vai arī citas metodes, kas neprasa lielu teksta korpusu lietošanu. Izmantojot aprakstītās metodes tika sasniegti līdzīgi rezultāti – precizitāte līdz 73% (3 klases) un līdz 76% (2 klases). Apskatītajos darbos lietoto klasificēšanas metožu un iegūto rezultātu kopsavilkums redzams tabulā 4.1. tabula. Citās valodās rakstīto tekstu noskaņojuma analīzei tiek lietots plašāks paņēmienu loks: leksikonā balstītas metodes, SVM, dziļie un LSTM neironu tīkli, likumos balstītas metodes, maksimālās entropijas metode, kā arī dažādi Naivā Beijesa algoritma varianti. Labākie rezultāti tika sasniegti izmantojot neironu tīklus – Sakenovitch [29] (apmācot LSTM rekurento neironu tīklu ar 30000 dokumentiem) un Galinsky [24] (apmācot dziļo neironu tīklu ar 195372 dokumentiem). Lai arī Shalunts un Backfried [33] izmantotā leksikona pieeja sasniedz lielisku 90% precizitāti, šāds rezultāts vismaz daļēji skaidrojams ar stipri šaurāku domēnu nekā citu autoru darbos.

Šajā nodaļā veikts literatūras apskats, izpētītas populārākās noskaņojuma klasificēšanas metodes, kuras ir izmantojuši dažādi autori, kā arī veikts to apkopojums un iegūto rezultātu salīdzinājums. Pēc darbu analīzes var secināt, ka klasificēšanas rezultāti, kurus publicē dažādi autori nav tieši salīdzināmi, jo tie ir iegūti izmantojot dažādus korpusus un lieto nesaderīgus veiktspējas novērtējuma mērus. Tālākajā darbā tiks veikta papildus anotēta korpusa izveidošana un izmantojot izveidoto korpusu tiks veikta dažādu faktoru vektora izveidošanas metožu un klasificēšanas metožu salīdzināšana un aprakstīto metožu veiktspējas novērtējums un salīdzinājums izmantojot vienu un to pašu korpusu un veiktspējas mērus.

5. METODOLOĢIJA

Šajā nodaļā tiek pētīta dažādu priekšapstrādes metožu un faktoru izvēles ietekme uz izveidoto modeļu veiktspēju, kā arī salīdzinātas vairākas klasificēšanas metodes.

5.1. Sīkziņu korpusa iegūšana

Tā kā pētījuma gaitā netika identificēts neviens piemērots anotēts Twitter sīkziņu korpuss latviešu valodā, šāds korpuss tika izveidots izmantojot Java valodā rakstītu programmu, kas, izmantojot Twitter streaming API, lejupielādēja sīkziņas vairākos periodos. Sīkziņas tika lejupielādētas vairākos laika posmos, lai iegūtu pēc iespējas dažādākas sīkziņas, par dažādiem aktuāliem notikumiem, kopā 73419 sīkziņas 8 laika posmos (sk. 5.1. tabula).

5.1. tabula. Lejupielādēto sīkziņu skaits dažādos laika posmos

Periods	Sīkziņas
2018.04.03-2018.04.04	4075
2018.03.05	10236
2018.01.18	4551
2017.11.16	4521
2017.09.10-2017.09.07	40726
2017.08.11	5176
2017.07.09-2017.07.05	2720
2017.05.22	1414

Lai iegūtu sīkziņas latviešu valodā, vaicājumā Twitter API tika norādīts filtrs, kas filtrēja sīkziņas pēc valodas „lv", kā arī papildus filtrēšana pēc atslēgvārdiem, kas ir bieži sastopami latviešu valodas vārdi. Atslēgvārdu sarakstā tika iekļauti:

- saikļi - "un", "arī", "bet", "tomēr", "taču", "turpretī", "turpretim", "nevis", "vai", "jeb", "ka", "lai", "ja", "jo", "kamēr", "līdz", "kopš", "iekams", "pirms",
- prievārdi - "aiz", "apakš", "bez", "iz", "kopš", "no", "pēc", "pie", "pirms", "priekš", "uz", "virs", "zem", "dēļ", "labad", "līdz", "ap", "caur", "gar", "pa", "pār", "par", "pret", "starp", "uz", "ar",
- vietniekvārdi - "es", "tu", "viņš", "viņa", "mēs", "jūs", "viņi", "viņas", "sevis", "mans", "mana", "tavs", "tava", "savs", "sava", "jūsu", "mūsu", "viņu", "šis", "šī", "tas", "tā", "šāds", "šāda", "tāds", "tāda", "viņš", "viņa", "kas", "kurš", "kura", "kāds", "kāda", "dažs", "daža", "cits", "cita", "jebkas", "jebkurš", "jebkura", "jebkāds", "jebkāda", "abi",

"abas", " visi", "visas", "pats", "pati", "katrs", "katra", "ikkatrs", "ikkatra", "ikkurš", "ikkura", "ikviens", "ikviena", "nekas", "nekāds", "nekāda", "neviens", "neviena",

- partikulas - "arī", "diez", "diezin", "diemžēl", "gan", "ik", "it", "itin", "jau", "jā", "jel", "kā", "kaut", "laikam", "ne", "nez", "nezin", "nē", "nu", "tātad", "tik", "tikai", "vai", "varbūt", "vēl", "vis", "ar", " diemžēl", "gan", "ir", "jau", "jel", "jo", "pat", "tad", "taču", "tak", " vien", "vienīgi", "vis".

Šāds garš atslēgvārdu saraksts vajadzīgs, jo atgriezti tiek tikai rezultāti, kas satur minētos atslēgvārdus, tātad jānorāda pēc iespējas lielāks atslēgvārdu skaits, lai iegūtu vairāk rezultātu.

5.2. Anotēta sīkziņu korpusa iegūšana

Kā jau tika secināts nodaļā 2.2., latviešu valodā pagaidām nav pieejams apjomīgs un kvalitatīvs anotētu sīkziņu korpuss. Tomēr, literatūras izpētes laikā tika identificēti šādi publiski pieejami anotēti sīkziņu korpusi latviešu valodā:

Peisenieka [25] izveidotais korpuss. Sākotnēji tika ievāktas 1.2 miljoni Twitter sīkziņas, kas pēc atfiltrēšanas saruka līdz 260 tūkstoš sīkziņām, daļa no tām tika anotētas izmantojot īpaši šim nolūkam veidotu tīmekļa vietni, kas lietotājam piedāvā novērtēt pseido-nejauši izvēlētu sīkziņu. Šāds sīkziņas izvēles veids tika izvēlēts, lai nodrošinātu derīgas datu kopas izveidi pie iepriekš nezināma vērtējumu skaita. Pseido-nejaušās izvēles algoritms darbojas šādi [25]: *Atlasīt 10 sīkziņas, kurām ir mazāk par 11 vērtējumiem. Atlasīt 10 sīkziņas, kurām nav neviena vērtējuma. Apvienot abas kopas. No apvienotās kopas, pēc nejaušības izvēlēties vienu sīkziņu.* Šāds algoritms nodrošināja ātru un resursu taupīgu sīkziņu izvēli. Sīkziņas tika klasificētas 4 klasēs: pozitīvās, neitrālās, negatīvās, un papildus tika ieviesta klase „ne-latviski", kurā tika iekļautas sīkziņas, kas piemēram, satur tikai atsauces birkas, saīsinājumus, abreviatūras, Twitter komandas u.c. Par pieņemamām tika uzskatītas sīkziņas, kurām kādai klasei piederošo vērtējumu proporcija pārsniedza 51%. Kopumā tika savākti 19613 vērtējumi, kas deva 1722 pietiekami novērtētas sīkziņas. Pēc atlases tika iegūtas 1177 pietiekamā kvalitātē anotētas sīkziņas, no tām kā pozitīvai klasei piederošas novērtētas 383 sīkziņas, neitrālai klasei piederošas 627 sīkziņas un negatīvai klasei piederošas 167 sīkziņas. Dotajam korpusam tika aprēķināts Fleiss Kappa [44] koeficients 0.284, kas ir pieņemams sakritības rādītājs. Lai šo rādītāju palielinātu līdz viduvējai sakritībai, būtu jāatstāj

tikai aptuveni 700 sīkziņas. Dotais korpuss ir brīvi izmantojams (MIT licence) un pieejams adresē[18].

Nicmanis [26] ievāca sīkziņas no viena lietotāja un tā sekotājiem. Daļu no savāktajām sīkziņām Nicmanis manuāli novērtēja un ieguva anotētu korpusu, kas sastāv no 3131 sīkziņas (1085 no tām pozitīvi novērtētas, 1712 – neitrāli un 334 negatīvi). Arī šis korpuss ir brīvi izmantojams (GNU General Public License v3.0) un pieejams Github adresē[19]. Tomēr uzmanīgi apskatot šo sīkziņu korpusu github, atklājās, ka anotētas ir tikai pirmās 1705 sīkziņas (894 neitrāli novērtētas, 670 pozitīvi un 141 negatīvi), bet pārējās ir pievienotas bez anotācijām.

Špats [19] lejupielādēja Twitter sīkziņas izmantojot dažādus atslēgvārdus ('es', 'man', 'ka', 'ir', 'nav', 'kas', 'tas', 'kad', 'ko', 'to', 'vai', 'un', 'tik', 'ar', 'esmu', 'tu', 'uz') kā vaicājuma parametrus un norādot valodu ‘lv’, lai iegūtu sīkziņas lielākoties latviešu valodā. Pēc priekšapstrādes tika secināts, ka iegūta 90171 sīkziņa. No iegūtā korpusa tika izdalītas 3104 sīkziņas, kas satur pozitīvu noskaņas vērtējumu >1 un satur vārdu paldies, vai kādu no izvēlētajām emocijzīmēm :) ;) smiley thumbs-up; 506 sīkziņas ar negatīvā sentimenta vērtējumu >1 un satur vārdus „slikt” vai „stulb” vai arī emocijzīmes :(sad-emoji vai arī manuāli novērtētas kā negatīvas; 2617 sīkziņas ar manuālu novērtējumu neitrāls vai ar negatīvo sentimenta vērtējumu 0 un pozitīvā sentimenta vērtējumu 0. Kopā šajā trokšņaini anotētajā korpusā iekļautas 6227 sīkziņas, kas brīvi pieejamas Github[20], kur tam ir pievienotas arī sīkziņas no Peisenieka korpusa, kas pirms tālākas korpusa izmantošanas ir jāatdala, lai nerastos dublikāti. Šo korpusu izlases kārtā novērtēja un tika secināts, ka korpusa un divu vērtētāju korpusa viedokļu sakritība bija 88% pozitīvajai klasei, 84% negatīvajai un 81% neitrālajai sīkziņu klasei.

5.2. tabula. Pieejamās datu kopas.

Autors	Pozitīvo sīkziņu skaits	Neitrālo sīkziņu skaits	Negatīvo sīkziņu skaits	Kopējais korpusa apjoms	Novērtējums
Peisenieks	383	627	167	1177	Fleiss kappa 0.284
Nicmanis	1085(670)	1712(894)	334(141)	3131(1705)	1 vērtētāja

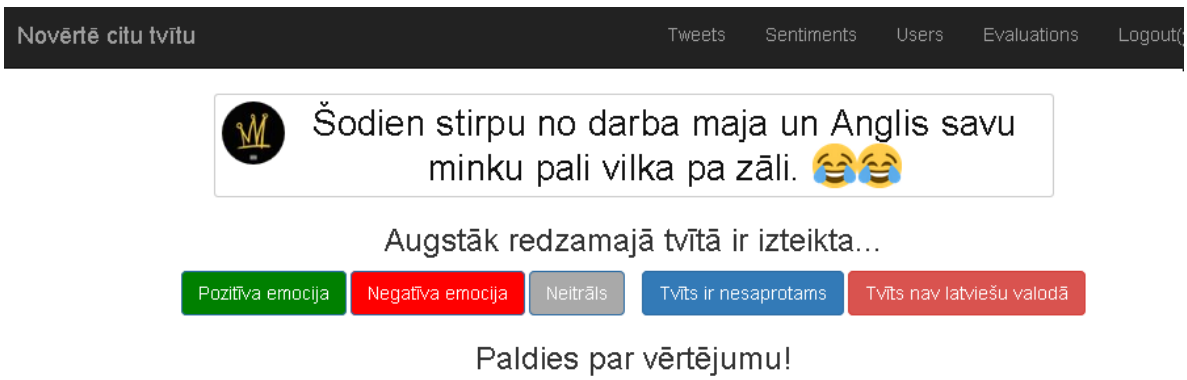
[18] https://github.com/FnTm/latvian-tweet-sentiment-corpus

[19] https://github.com/nicemanis/LV-twitter-sentiment-corpus

[20] https://github.com/gatis/om/blob/master/data/psgs.arff

					viedoklis, korpuss pieejams daļēji
Špats	3104	2617	506	6227	Anotētas izmantojot emocijzīmes. Salīdzinot ar cilvēka viedokli 81-88% sakritība
Autors	897	2202	519	3618	1-4 vērtētāji

Ņemot vērā nelielos esošo korpusu apjomus (5.2. tabula), tika izlemts anotēt papildus sīkziņu korpusu. Šim nolūkam tika izveidota tīmekļa vietne http://www.calotava.lv (skat. 5.1. att.), kurā jebkurš interesents var novērtēt piedāvātās sīkziņas. Tīmekļa vietne tika izveidota izmantojot Yii2 PHP programmēšanas ietvaru un MySQL datubāzi. Tīmekļa vietne piedāvā uzreiz novērtēt sīkziņas, kā arī īsu paskaidrojumu, kā pareizi jāvērtē un iespēju piereģistrēties. Reģistrētajiem lietotājiem tiek piedāvāts novērtēt tikai sīkziņas ko šis lietotājs vēl nav novērtējis, taču lietotājiem, kas nav reģistrējušies tiek piedāvāts novērtēt sīkziņu pēc gadījuma izvēles. Reģistrācija ir veidota vienkārša – jānorāda tikai vēlamais lietotāja vārds un parole. Šāda reģistrēšanās metode tika izvēlēta, lai ļautu piedāvāt lietotājam sīkziņas, ko tas vēl nav vērtējies, kā arī radītu aizsardzību pret vandāļiem un robotiem. Sistēmā tika ielādētas 10000 sīkziņas, kas tika demonstrētas lietotājam. Darba laikā tika secināts, ka sīkziņas, kas ir atbilde uz kāda cita lietotāja teikto (sīkziņas sākumā ir simbols @) bieži ir nesaprotamas bez konteksta, tādēļ tās netika piedāvātas tālākai vērtēšanai.



5.1. att. Tīmekļa vietnes vērtēšanas sadaļa

Twitter sīkziņu vērtēšana tika veikta piecās klasēs – „Pozitīva emocija", „Negatīva emocija", „Neitrāls", „Tvīts ir nesaprotams" un „Tvīts nav latviešu valodā". Sīkziņas, kam

tika piešķirtas klases „Tvīts ir nesaprotams” vai „Tvīts nav latviešu valodā” tālākajā darbā netiek lietotas. Izgūstot sīkziņas un vērtējumus no datubāzes tika secināts, ka kopumā ir veikti 11000 novērtējumi. Dati tika izgūti JSON formātā (5.2. att.), kas satur sīkziņas ID, tekstu, un skaitu, cik reizes sīkziņa novērtēta kā pozitīva, neitrāla, negatīva, nesaprotama vai svešvalodā.

{"tweet id":866567610661576704,"text":"@Kasparinhs B\u016btu labi!","POS":1,"NEG":0,"NEU":0,"IMP":0,"NOT LV":0},

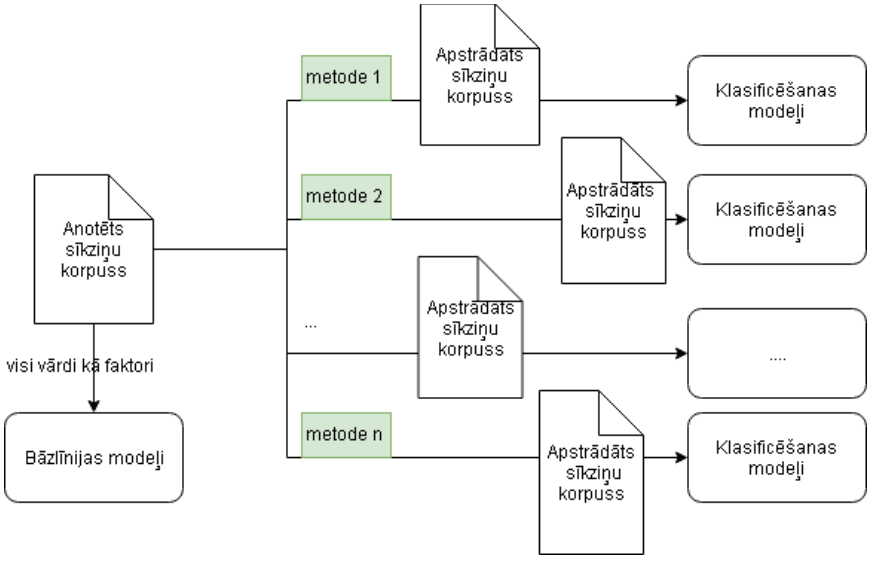
5.2. att. Sīkziņas piemērs JSON datu formātā

Anotētās sīkziņas no JSON formāta tiek pārveidotas tsv (ar tabulatoru atdalīti dati) formātā izmantojot Python skriptu, kas nolasa JSON datus un saglabā sīkziņas id, tekstu un emocionālās noskaņas klasi, ja sīkziņai ir iespējams noteikt piederību kādai no trīs klasēm („POS”, „NEG” vai „NEU”). Lai lemtu par klases piederību sīkziņām, kuras bija novērtējis vairāk nekā viens vērtētājs, tika izmantots vairākuma princips, t.i. tiek uzskatīts, ka sīkziņa pieder tai klasei, kura šai sīkziņai ir izvēlēta visvairāk. Ja vērtējumu skaits vairākām klasēm ir vienāds un >0, tad sīkziņa tiek atmesta, jo tai nevar noteikt klasi.

Autora izveidotajam 3618 sīkziņu korpusam tika pievienots Nicmaņa [26] izveidotais korpuss (1705 sīkziņas), iegūstot rezultātā 5323 sīkziņu lielu korpusu, kas tiek lietots tālākajā darbā.

5.3. Faktoru vektora izveidošana un testi

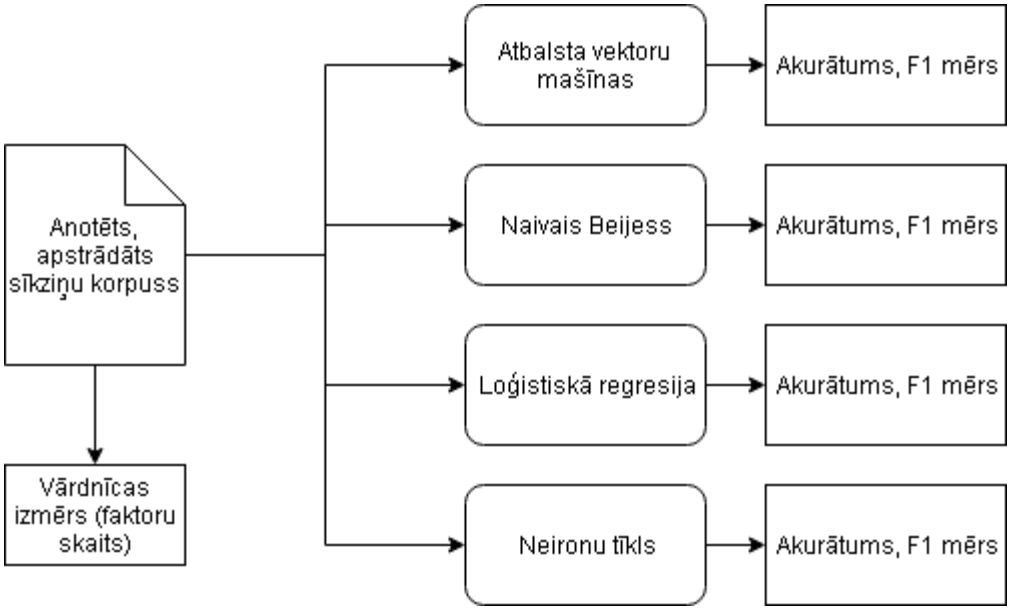
Izmantojot izveidoto sīkziņu korpusu, tika izveidoti modeļi, kas tālāk kalpo kā bāzlīnija, ar kuru novērtēts dažādu faktoru vektora izveidošanas metožu iespaids uz modeļu veiktspēju. 5.3. attēlā attēlota darba plūsma dažādu priekšapstrādes metožu novērtēšanai.



5.3. att. Faktoru vektora izveidošanas metožu salīdzināšanas darbu plūsma

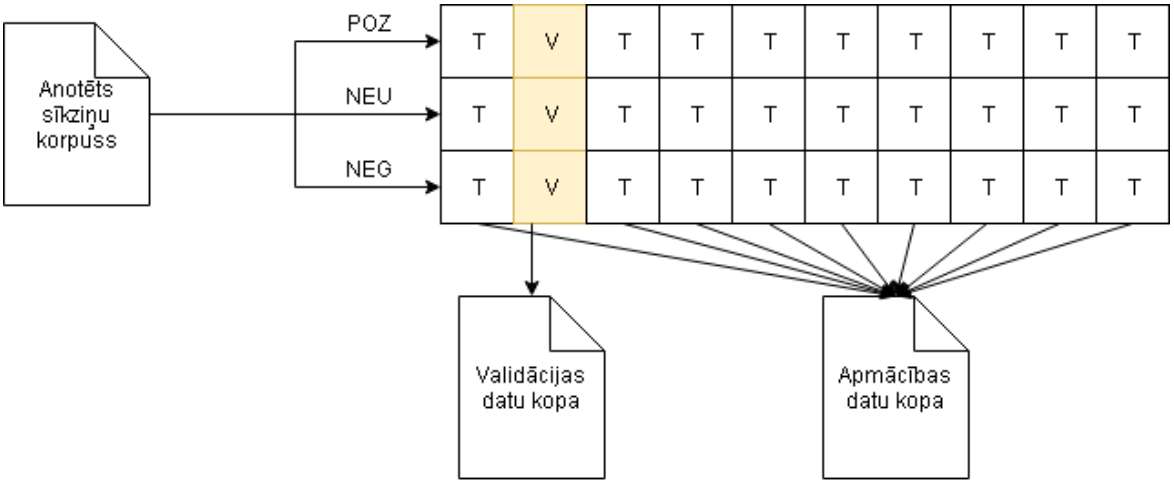
Izmantojot 3. nodaļā identificētās faktoru vektora izveidošanas metodes, tika veikta sīkziņu tekstu priekšapstrāde (transliterācijas aizvietošana, celma iegūšana, skaitļu

aizvietošana u.c.) un iegūtais apstrādātais sīkziņu korpuss tika izmantots, lai apmācītu četrus modeļus: atbalsta vektoru mašīnas, naivā beijesa klasifikatoru, loģistisko regresiju un neironu tīklu (5.4. att.).



5.4. att. Faktoru vektora novērtēšana

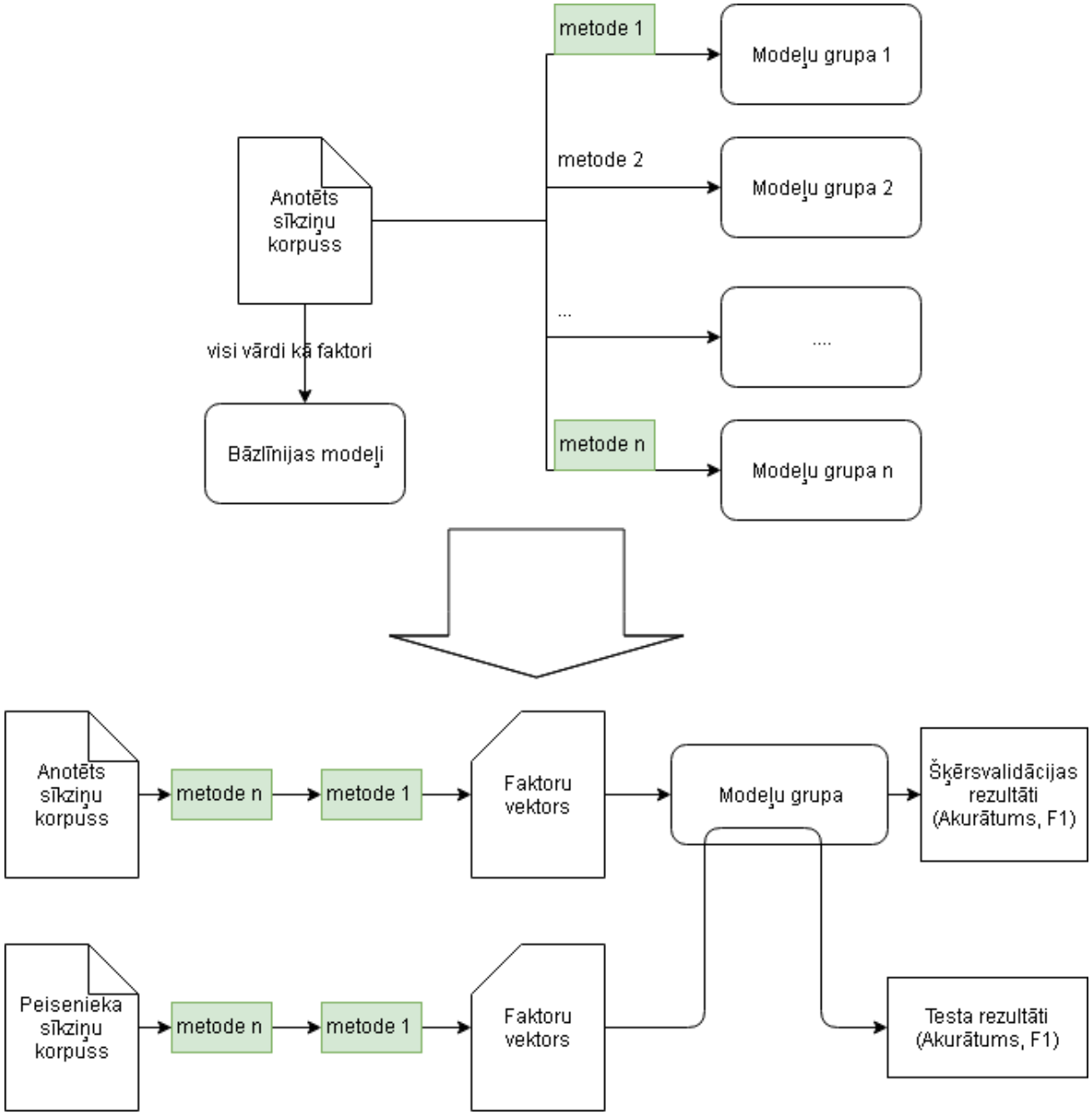
Katrs no modeļiem tika apmācīts, izmantojot vienādus datus, un arī novērtēts, izmantojot identiskus mērus (akurātums un F1), kas nodrošina iespēju korekti salīdzināt dažādu datu priekšapstrādes metožu ietekmi uz izveidoto modeļu veiktspēju. Tā kā izveidotā datu kopa nav liela (5323 ziņas) un tajā dominē neitrālas ziņas, tad šim nolūkam tika izmantota 10-kāršas šķērsvalidēšanas stratificētā (*stratified*) versija. Tā nodrošina, ka katrā iterācijā apmācības un validācijas datu kopās tiks saglabāta sākotnējā klašu proporcija, kas ir svarīgi, jo iegūtajā datu kopā dominē neitrālas un pozitīvas sīkziņas. Tas tiek realizēts, sadalot sākotnējo datu korpusu klasēs un katrā iterācijai nepieciešamo datu daļu ņemot atsevišķi no katras klases un kombinējot (5.5. att.).



5.5. att. Viena no 10-kāršas šķērsvalidēšanas iterācijām

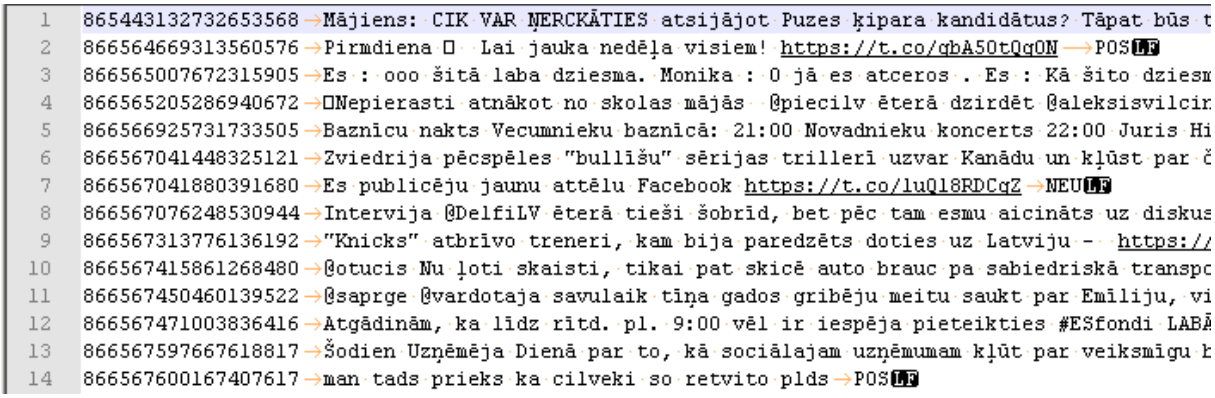
Rezultātā katrs modelis ir izveidots 10 reizes uz mazliet atšķirīgiem apmācības datiem un novērtēts uz datu kopu, kas apmācības datos nav iekļauta, un attiecīgi atšķirās arī iegūtie rezultāti – gan faktoru vektora garums, gan akurātums un F1. Lai novērtētu faktoru vektora garumu tika atzīmēts lielākais faktoru vektora garums, bet akurātums un F1 tika atrasti makro –vidējie rādītāji [18]. Lai tos atrastu tika izveidota visām iterācijām kopīga pārpratumu matrica un akurātums un F1 tika aprēķināti izmantojot to.

Pēc darbā izmantoto priekšapstrādes un faktoru vektora izveidošanas metožu novērtēšanas tiek veikta to lietderīguma analīze un izstrādāta metožu secība, kas ļauj iegūt galīgo faktoru vektoru. Izmantojot izstrādāto metožu secību tiek izveidoti klasificēšanas modeļi, ko novērtēti, izmantojot 10-kāršu šķērsvalidēšanu, kā arī veikti papildus testi, izmantojot Peisenieka [25] izveidoto sīkziņu korpusu, lai salīdzinātu iegūto modeli ar iepriekš iegūtajiem rezultātiem.



5.6. att. Galīgās datu priekšapstrādes metožu plūsmas izveidošanas shēma

Sīkziņas (gan autora anotētās, gan citu pētnieku anotētās) tika saglabātas teksta failos utf-8 kodējumā, sīkziņas id, teksta saturu un novērtēto klasi atdalot ar tabulatora zīmi. Emocijzīmes, ko tādi teksta redaktori, kā notepad++ neprot attēlot tiek parādītas attēlā 5.7. kā taisnstūri □, tomēr kodi tiek saglabāti un var tikt lietoti tālākā darbā.



```
1  865443132732653568→Mājiens: CIK VAR ĶERCKĀTIES atsijājot Puzes ķipara kandidātus? Tāpat būs t
2  866564669313560576→Pirmdiena □ Lai jauka nedēļa visiem! https://t.co/qbA50tQqON→POSLF
3  866565007672315905→Es : ooo šitā laba dziesma. Monika : O jā es atceros . Es : Kā šito dziesm
4  866565205286940672→□Nepierasti atnākot no skolas mājās @piecilv ēterā dzirdēt @aleksisvilcir
5  866566925731733505→Baznīcu nakts Vecumnieku baznīcā: 21:00 Novadnieku koncerts 22:00 Juris Hi
6  866567041448325121→Zviedrija pēcspēles "bullīšu" sērijas trillerī uzvar Kanādu un kļūst par č
7  866567041880391680→Es publicēju jaunu attēlu Facebook https://t.co/luQl8RDCqZ→NEULF
8  866567076248530944→Intervija @DelfiLV ēterā tieši šobrīd, bet pēc tam esmu aicināts uz diskus
9  866567313776136192→"Knicks" atbrīvo treneri, kam bija paredzēts doties uz Latviju - https://
10 866567415861268480→@otucis Nu ļoti skaisti, tikai pat skicē auto brauc pa sabiedriskā transpo
11 866567450460139522→@saprge @vardotaja savulaik tīņa gados gribēju meitu saukt par Emīliju, vi
12 866567471003836416→Atgādinām, ka līdz rītd. pl. 9:00 vēl ir iespēja pieteikties #ESfondi LABĀ
13 866567597667618817→Šodien Uzņēmēja Dienā par to, kā sociālajam uzņēmumam kļūt par veiksmīgu b
14 866567600167407617→man tads prieks ka cilveki so retvito plds→POSLF
```

5.7. att. tsv faila saturs

Lai iegūtu bāzlīniju, ar ko salīdzināt turpmāk iegūtos rezultātus, klasificēšanas modeļu apmācība tiks veikta, izmantojot datus, kuriem nav veikta nekāda priekšapstrāde un kā faktori tiek izmantoti visi tekstā pieejamie vienumi. Faktoru vektors ir pārāk liels, lai to glabātu kā skaitļu vektoru, tādēļ vektorizācijai tiek lietota sklearn bibliotēkas klase sklearn.feature_extraction.text.CountVectorizer ar noklusētajiem uzstādījumiem:

- input='content',
- encoding='utf-8',
- decode_error='strict',
- strip_accents=None, # nemaina latviešu burtus ar garumzīmēm
- lowercase=True, # vārdi tiek pārveidoti par mazajiem burtiem
- preprocessor=None,
- tokenizer=None,
- stop_words=None,
- token_pattern='(?u)\b\w\w+\b',
- ngram_range=(1, 1), # unigrammas
- analyzer='word', # vārdu n-grammas
- max_df=1.0,
- min_df=1,
- max_features=None,
- vocabulary=None,
- binary=False,
- dtype=<class 'numpy.int64'>

CountVectorizer pēc noklusējuma pārveido visus burtus par mazajiem, aizvieto visas pieturzīmes, izmanto vārdu unigrammas un utf8 kodējumu.

Veicot 10-kāršu šķērsvalidēšanu, lielākais novērotais faktoru vektora izmērs bija 19138 faktori. Izmantojot šādus datus iegūto modeļu veiktspēja redzama 5.3. tabulā.

5.3. tabula. Izmantojot neapstrādātus datus iegūtie rezultāti

Modelis	Akurātums	F1
SVM	0.615	0.503
Naivais Beijess	0.653	0.529
Loģistiskā regresija	0.642	0.508
Neironu tīkls	0.617	0.512
vidēji	0.632	0.513

Lai noskaidrotu ieguvumus izmantojot katru no faktoru vektora apstrādes metodēm, tās tika pielietotas sākotnējiem datiem un iegūto datu kopu apmācīti modeļi un iegūtie rezultāti saglabāti tabulās.

5.3.1. Vārdu pamatformu iegūšana

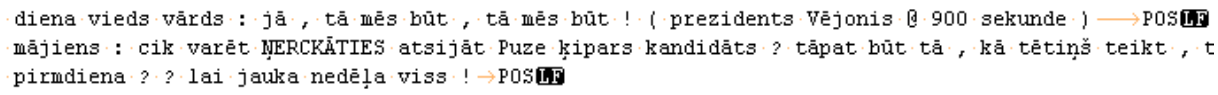
Latviešu valodā vārdus iespējams locīt, tādējādi no viena pamatvārda radot veselu saimi ar radniecīgiem vārdiem. Viens un tas pats vārds dažādos locījumos nereti pauž vienu un to pašu noskaņojumu, tādēļ tika veikta vārdu pamatformu (lemmas) iegūšana izmantojot rīku LVTagger[21], kas sīkāk aprakstīts publikācijā [45]. Tā kā šis rīks apstrādā arī tīmekļa saites, tās sadalot komponentēs un mēģinot to daļām noteikt pamatformu, tad tīmekļa saites tika izdzēstas pirms teksta apstrādes ar rīku. Ar šo rīku tika apstrādāti dati un tika apmācīti klasifikatori un novērtēta to veiktspēja. Rezultāti ir redzami tabulā 5.4. Veicot 10-kāršu šķērsvalidēšanu lielākais novērotais faktoru vektora izmērs bija 12818 faktori. Izmantojot šādus datus iegūto modeļu veiktspēja redzama tabulā 5.4.

5.4. tabula. Izmantojot vārdu pamatformas iegūtie rezultāti

Modelis	Akurātums	F1
SVM	0.605	0.501
Naivais Beijess	0.648	0.531
Loģistiskā regresija	0.637	0.508
Neironu tīkls	0.606	0.505
vidēji	0.624	0.511
Uzlabojums pret bāzlīniju, %	-1.27	-0.39

[21] https://github.com/PeterisP/LVTagger

Vārdu pamatformu iegūšanas cenšas vārdus sadalīt sastāvdaļās, kas dažos gadījumos netiek veikts pareizi, piemēram, īpašvārds „900sekundes” (raidījuma nosaukums) tiek sadalīts – „900” un „sekundes” (5.8. att.).

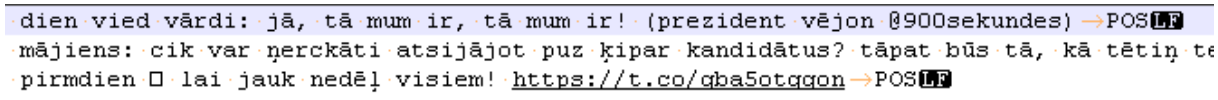


5.8. att. Teksta piemērs pēc pamatformu iegūšanas

Tāpat rīkam nav pilnīgs emocijzīmju atbalsts un tās tiek sadalītas pieturzīmēs, tātad emocijzīmes ir atsevišķi jāapstrādā pirms šī rīka lietošanas, rīku izmantojot tikai vārdu analīzei. Kā redzams, faktoru skaits ievērojami samazinājās (no 19138 uz 12818), kas atvieglo mašīnapmācības modeļu apmācību. Tomēr izveidoto modeļu veiktspēja lielākoties samazinājās (izņēmums – Naivais Baijess, kam uzlabojās F1), kas nozīmē, ka tika zaudēti nozīmīgi faktori.

5.3.2. Vārdu celma iegūšana

Alternatīvs variants ir izmantot nevis vārdu pamatformu, bet gan celmu, kas visiem radniecīgajiem vārdiem ir vienāds, bet var nesakrist ar morfoloģisko vārda sakni un var nebūt jēgpilns vārds. Šāda celma iegūšana ir mazāk sarežģīta par pamatformas iegūšanu, jo vārds nav jāapskata kontekstā, taču iespējama situācija, kad līdzīgi, tomēr ar pilnīgi dažādu jēgu vārdi tiek pārveidoti par vienu celmu (piemēram, roka un roku, kur pirmais vārds ir lietvārds, bet otrais darbības vārds). Lai veiktu celmošanu, tika izmantots rīks, kas ir Kārļa Krēsliņa [46] izstrādātā algoritma modificēta versija Python valodā, ko izstrādājis Rihards Krišlauks[22].



5.9. att. Teksta piemērs pēc celmošanas

Izmantojot celmotos sīkziņu datus (5.9. att.) tika apmācīti klasifikatori un iegūtie veiktspējas rādītāji attēloti tabulā 5.5. Veicot 10-kāršu šķērsvalidēšanu lielākais novērotais faktoru vektora izmērs bija 16514 faktori.

[22] https://pypi.org/project/LatvianStemmer/1.0.1/

5.5. tabula. Izmantojot vārdu celmus iegūtie rezultāti

Modelis	Akurātums	F1
SVM	0.620	0.506
Naivais Beijess	0.654	0.536
Loģistiskā regresija	0.644	0.520
Neironu tīkls	0.612	0.507
vidēji	0.633	0.517
Uzlabojums pret bāzlīniju, %	0.16	0.78

Kā redzams, attiecībā pret bāzlīniju vidējie rezultāti nav pārliecinoši, akurātums dažiem modeļiem nedaudz samazinās un citiem nedaudz palielinās, bet F1 palielinās (izņēmums neironu tīkls, kam samazinās F1). Samazinājās faktoru vektora izmērs, tomēr ne tik jūtami, kā veicot pamatformu iegūšanu, kas izskaidrojams ar to, ka izmantotajā celmošanas skriptā netiek celmoti visi vārdi, bet tikai daļa, ko spēj celmot izmantotais algoritms.

5.3.3. Lietotājvārdu, atsauces tagu un saišu aizvietošana

Lietotājvārdi, atsauces tagi un saites ļauj identificēt kādu personu, notikumu, vai objektu, tātad nosaukto vienumu. Nosauktie vienumi lielākoties ir unikāli, vai reti sastopami un paši par sevi nenes informāciju par teksta noskaņojumu (lai arī var gadīties, ka kāds lietotājs vai notikums tiek minēts tikai negatīvās sīkziņās). Izmantojot apstrādātos datus tika izveidoti modeļi un to veiktspējas rādītāji fiksēti tabulā 5.6. Veicot 10-kāršu šķērsvalidēšanu lielākais novērotais faktoru vektora izmērs bija 18092 faktori, kas nozīmē, ka tika izdzēsti vairāk kā 1000 teksta vienību. Tā kā modeļu darbības rādītāji pasliktinājās, var secināt, ka šajā korpusā lietotājvārdi vai tēmturi tomēr ļāva paredzēt sīkziņu emocionālā noskaņojuma klasi mazliet precīzāk. Tas varētu būt izskaidrojams ar to, ka tiek izmantota šķērsvalidēšana, un gan apmācības, gan testa dati ir saistīti un satur vienus un tos pašus lietotājvārdus un populārus tēmturus.

5.6. tabula. Izmantojot datus ar aizvietotiem lietotājvārdiem, saitēm un tagiem iegūtie rezultāti

Modelis	Akurātums	F1
SVM	0.621	0.506
Naivais Beijess	0.649	0.528
Loģistiskā regresija	0.639	0.505
Neironu tīkls	0.610	0.502
vidēji	0.630	0.511
Uzlabojums pret bāzlīniju, %	-0.32	-0.39

5.3.4. Transliterācijas aizvietošana

Transliterācija ir vārdu rakstīšana, izmantojot lietotājam pieejamos simbolus, kuru starpā nereti nav sastopami latviešu valodas burti ar garumzīmēm vai mīkstinājuma zīmēm. Tāpat pie transliterācijas var pieskaitīt gadījumus, kad rakstot vārdu kāds burts ir izlaists, vai arī kļūdas pēc rakstīts izmantojot burtu bez garumzīmes. Transliterācijas problemātikas risināšanai [13] tika izveidots rīks „Ruukjiishi"[23]. Šo rīku izmantot nebija iespējams, tomēr balstoties uz tajā izmantotajiem transliterācijas aizvietošanas likumiem tika izstrādāts vienkāršots transliterācias aizvietošanas skripts, kas tiek izmantots šajā darbā. Ar šo skriptu tika apstrādāti dati un tika apmācīti klasifikatori un novērtēta to veiktspēja. Rezultāti ir redzami tabulā 5.7. Veicot 10-kāršu šķērsvalidēšanu lielākais novērotais faktoru vektora izmērs bija 19102 faktori, kas ir samazinājums par 36 faktoriem. Arī šāds uzlabojums ir noderīgs, un izveidoto modeļu darbība lielākoties uzlabojās.

5.7. tabula. Izmantojot datus, kuros aizvietota transliterācija iegūtie rezultāti

Modelis	Akurātums	F1
SVM	0.619	0.505
Naivais Beijess	0.654	0.527
Loģistiskā regresija	0.644	0.514
Neironu tīkls	0.617	0.513
vidēji	0.634	0.514
Uzlabojums pret bāzlīniju, %	0.32	0.19

[23] https://bitbucket.org/Ginta/ruukjiishi

5.3.5. Skaitļu aizvietošana un pieturzīmju aizvietošana

Skaitļi un pieturzīmes – tādi kā punkts, komats u.c. parasti nenes informāciju par teksta emocionālo nokrāsu, tādēļ atsevišķi esošie skaitļi tika aizvietoti ar vietturi „_num", bet pieturzīmes (,.!?) tika dzēstas, un izmantojot apstrādātos datus tika apmācīts klasifikators un rezultāti apkopoti tabulā 5.8. Veicot 10-kāršu šķērsvalidēšanu lielākais novērotais faktoru vektora izmērs bija 18915 faktori, tas ir tika dzēsti 223 unikāli skaitļi, kas arī bija sagaidāms, jo korpuss nav liels. Skaitļu izdzēšanas rezultātā modeļu uzrādītie rezultāti uzlabojās pavisam nedaudz.

5.8. tabula. Izmantojot datus, kuros aizvietoti skaitļi iegūtie rezultāti

Modelis	Akurātums	F1
SVM	0.615	0.501
Naivais Beijess	0.654	0.531
Loģistiskā regresija	0.646	0.513
Neironu tīkls	0.619	0.514
vidēji	0.633	0.514
Uzlabojums pret bāzlīniju, %	0.16	0.19

5.3.6. Stopvārdu dzēšana

Stopvārdi ir bieži sastopami vārdi, kam nepiemīt emocionāla nokrāsa, un tika pieņemts, ka to dzēšana samazina faktoru skaitu, tādējādi atvieglojot noskaņojuma analīzi. Latviešu valodā sastopamo 342 stopvārdu saraksts ir izveidots darbā [13] un ir brīvi pieejams[24], kas tika lietots arī šajā darbā. Stopvārdi tika dzēsti, un izmantojot apstrādātos datus tika apmācīts klasifikators un rezultāti apkopoti tabulā 5.9. Veicot 10-kāršu šķērsvalidēšanu lielākais novērotais faktoru vektora izmērs bija 19037 faktori, kas ir par 101 faktoru mazāk nekā neapstrādātajos datos. Daudzi no stopvārdiem lietotajā sarakstā ir burti no latviešu un krievu alfabēta, kas izskaidro, šādu faktoru skaita samazinājumu. Izveidoto modeļu akurātums un F1 mērs uzrāda nelielu uzlabojumu.

[24] http://barometrs.korpuss.lv/?from=2017-04-29&to=2018-04-29&site=lv&section=stopwords

5.9. tabula. Izmantojot stopvārdu dzēšanu iegūtie rezultāti

Modelis	Akurātums	F1
SVM	0.627	0.503
Naivais Beijess	0.647	0.532
Loģistiskā regresija	0.652	0.534
Neironu tīkls	0.613	0.506
vidēji	0.635	0.518
Uzlabojums pret bāzlīniju, %	+0.47	+0.97

5.3.7. Bigrammas un 3-grammas

Izmantojot Python bibliotēkas CountVectorizer klasi, tika veikta sīkziņu teksta pārveidošana faktoru vektorā. Lai izgūtu n-grammas, jānorāda parametrs ngram_range(min,max), kuram noklusētā vērtība ir (1,1) – izgūst unigrammas. Lai papildus unigrammām iegūtu faktoru vektoru, kas satur bigrammas un 3-grammas, tiek norādīts ngram_range(1,2) un ngram_range(1,3). Ar šādām parametra vērtībām veicot modeļu veidošanu iegūtie rezultāti redzami tabulās 5.10 un 5.11. Veicot 10-kāršu šķērsvalidēšanu lielākais novērotais faktoru vektora izmērs bija uni- un 2-grammām - 64153 faktori un izmantojot uni- 2- un 3-grammas – 110417 faktori.

5.10. tabula. Izmantojot uni- un 2-grammas iegūtie rezultāti

Modelis	Akurātums	F1
SVM	0.639	0.509
Naivais Beijess	0.651	0.530
Loģistiskā regresija	0.645	0.515
Neironu tīkls	0.640	0.515
Vidēji	0.644	0.519
Uzlabojums pret bāzlīniju, %	1.90	1.17

Atšķirība starp unigrammām, uni- un 2-grammām un uni-, 2- un 3-grammām ir neliela, 2 grammu pievienošana dod vidēji 1.9% akurātuma uzlabojumu un 1.17% F1 mēra uzlabojumu un 3 grammu pievienošana dod vidēji 2.22% akurātuma uzlabojumu un 1.36% F1 mēra uzlabojumu. Tomēr kā klasifikatoru izmantojot Naivo Beijesu (kas uzrāda labākos rezultātus), 2- un 3- grammu izmantošana pasliktina akurātumu vairāk, un F1 uzlabojums ir tikai 0.38%, kas ļauj secināt, ka izmantojot kā klasifikatoru Naivo Beijesu, nav lietderīgi lietot

2- un 3-grammas, jo izmantoto faktoru skaits pieaug 5 reizes, kas prasa lielākus atmiņas, dator- un laika resursus, kamēr veiktspēja dažos gadījumos pat samazinās.

5.11. tabula. Izmantojot uni-, 2- un 3-grammas iegūtie rezultāti

Modelis	Akurātums	F1
SVM	0.641	0.504
Naivais Beijess	0.650	0.531
Loģistiskā regresija	0.645	0.513
Neironu tīkls	0.648	0.530
vidēji	0.648	0.520
Uzlabojums pret bāzlīniju, %	2.22	1.36

5.3.8. Emocijzīmju apstrāde

ASCII emocijzīmes (tādas kā :), :(, :P :/ u.c.) un arī unicode emocijzīmes (emoji - ❄,✈,❤ u.c) nepieciešams apstrādāt atsevišķi, jo Python bibliotēka CountVectorizer neveic to apstrādi un tās tiek dzēstas no faktoru vektora, taču emocijzīmes ir būtisks faktors, kas ļauj noteikt teksta emocionālo noskaņu. Emocijzīmes tiks aizvietotas ar vietturiem "emoji_pos", "emoji_neu" un "emoji_neg". Emocijzīmju pārveidošanai tiks izmantots Novak et al. [47] veiktā pētījuma rezultātā iegūtais emocijzīmju saraksts[25], kā arī ASCII latīņu simbolu emocijzīmju saraksts no Wikipedia[26], kuru emocionālā nokrāsa ir saprotama pēc to paskaidrojuma. Apvienojot abas kopas tika iegūts saraksts ar 867 emocijzīmēm, kurām ir zināma emocionālā noskaņa. Izmantojot iegūto faktoru vektoru tika izveidoti modeļi, kuru veiktspēja novērtēta tabulā 5.12. tabula. Pēc datu priekšapstrādes maksimālais faktoru skaits palielinājās līdz 19141, kas izskaidrojams ar 3 jaunām tekstvienībām: „emoji_pos", „emoji_neu" un „emoji_neg". Neapstrādātas emocijzīmes CountVectorizer klase neņem vērā, kas arī izskaidro ievērojamo klasificēšanas mēru uzlabojumu.

[25] http://kt.ijs.si/data/Emoji_sentiment_ranking/

[26] https://en.wikipedia.org/wiki/List_of_emoticons

5.12. tabula. Izmantojot emocijzīmes iegūtie rezultāti

Modelis	Akurātums	F1
SVM	0.700	0.596
Naivais Beijess	0.704	0.590
Loģistiskā regresija	0.726	0.608
Neironu tīkls	0.673	0.572
Vidēji	0.701	0.591
Uzlabojums pret bāzlīniju, %	10.92	15.20

5.4. Priekšapstrādes un faktoru izvēles rezultāti

Pēc dažādu priekšapstrādes metožu izmantošanas ir iespējams salīdzināt katras no tām ietekmi uz rezultātiem. Izmantojot katru metodi tika izveidoti modeļi, tie novērtēti ar šķērsvalidēšanas metodi un vidējie rezultāti apkopoti tabulās 5.13 un 5.14. Ar dzeltenu iezīmēti testi, kuros labākos rezultātus uzrādīja Naivā Beijesa modelis un ar sarkanu – loģistiskās regresijas modelis. Atbalsta vektora mašīnas un neironu tīkli neuzrādīja labākos rezultātus nevienā no testiem, kas izskaidrojams ar to, ka modeļiem ir vairāki konfigurējami parametri, tomēr tie tika lietoti ar noklusētajām parametru vērtībām, kas varētu nebūt piemērotas.

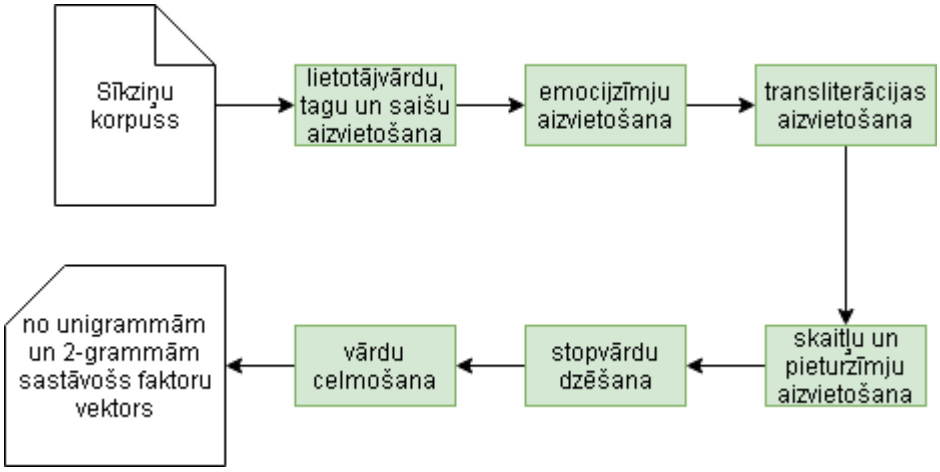
5.13 tabula. Priekšapstrādes un faktoru izvēles metožu salīdzinājums (vidējie rādītāji)

Metode	Akurātums	F1	Vārdnīcas izmērs	Akurātuma uzlabojums %	F1 uzlabojums %
Bāzlīnija	0.632	0.513	19138	0.00	0.00
Vārdu pamatformas	0.624	0.511	12818	-1.27	-0.39
Vārdu celmi	0.633	0.517	16514	0.16	0.78
Lietotājvārdu, atsauces tagu un saišu aizvietošana	0.630	0.511	18092	-0.32	-0.39
Transliterācijas aizvietošana	0.634	0.514	19102	0.32	0.19
Skaitļu un pietruzīmju aizvietošana	0.633	0.514	18915	0.16	0.19
Stopvārdu dzēšana	0.635	0.518	19037	0.47	0.97
Uni un 2-grammas	0.644	0.519	64153	1.90	1.17
Uni, 2- un 3-grammas	0.646	0.520	110417	2.22	1.36
Emocijzīmju aizvietošana	0.701	0.591	19141	10.92	15.20

5.14 tabula. Priekšapstrādes un faktoru izvēles metožu salīdzinājums (labākie rādītāji)

Metode	Akurātums	F1	Vārdnīcas izmērs	Akurātuma uzlabojums %	F1 uzlabojums %
Bāzlīnija	0.653	0.529	19138	0.00	0.00
Vārdu pamatformas	0.648	0.531	12818	-0.77	0.38
Vārdu celmi	0.654	0.536	16514	0.15	1.32
Lietotājvārdu, atsauces tagu un saišu aizvietošana	0.649	0.528	18092	-0.61	-0.19
Transliterācijas aizvietošana	0.654	0.526	19102	0.15	-0.57
Skaitļu un pietruzīmju aizvietošana	0.654	0.531	18915	0.15	0.38
Stopvārdu dzēšana	0.652	0.534	19037	-0.15	0.95
Uni un 2-grammas	0.651	0.530	64153	-0.31	0.19
Uni, 2- un 3-grammas	0.650	0.531	110417	-0.46	0.38
Emocijzīmju aizvietošana	0.726	0.608	19141	11.18	14.93

Redzams, ka vienīgo nopietno rezultātu uzlabojumu nodrošina emocijzīmju aizvietošana ar vietturiem, ko Python sklearn bibliotēkā implementētie modeļi spēj apstrādāt. Emocijzīmju izmantošana uzlabo modeļu akurātumu vidēji par 10% un F1 par vidēji 13%. Iespējams salīdzināt arī kā faktorus izmantojot vārdu celmus un saknes iegūtos rezultātus. Kopumā, izmantojot lemmas, gan akurātums, gan F1 samazinās, tomēr, F1 uzlabojas, ja tiek izmantots Naivais Beijess (kas ir labākais), kas norāda, ka labāk tiek veikta neneitrālo klašu klasificēšana. Veicot vārdu pamatformu iegūšanu, izmantotajā rīkā tika konstatētas dažas nepilnības, kuras iespējams apiet, rīku pielietojot pēc īpašvārdu un emocijzīmju aizvietošanas, kas, iespējams, uzlabotu rezultātus, tomēr, tā kā vārdu celmošanas rīks nodrošina līdzīgu funkcionalitāti un labākus rezultātus, tad tika lietots tas. Lietotājvārdu un tēmturu aizvietošana samazina faktoru skaitu par 1000 un, lai arī šajā korpusā mazliet samazina modeļu veiktspēju, tomēr tiek paturēta, galīgajā procesā, jo, modeli izmantojot uz citiem datiem, tajos sagaidāms, ka lietotājvārdi nesakritīs ar šo korpusu. Transliterācijas aizvietošana, skaitļu aizvietošana, pieturzīmju dzēšana, stopvārdu dzēšana un papildus 2-grammu lietošan nodrošina gan akurātuma, gan F1 uzlabojumu, tādēļ šie soļi tiek iekļauti galīgajā priekšapstrādes procesā (5.10. att.).



5.10. att. Faktoru vektora veidošanas process

5.5. Modeļa novērtējums

Galīgais modelis tiek pārbaudīts tāpat kā iepriekš izveidotie modeļi, un tiek izveidota 5.15. tabula, kurā reģistrēta tā darbību raksturojošie mēri.

5.15. tabula. Izmantojot galīgos klasificēšanas modeļus iegūtie rezultāti

Modelis	Akurātums	F1
SVM	0.733	0.625
Naivais Beijess	0.713	0.601
Loģistiskā regresija	0.739	0.635
Neironu tīkls	0.708	0.598
Vidēji	0.723	0.615
Vidēji attiecībā pret bāzlīniju	14.40	19.88

Modelis tiek testēts arī izmantojot nesaistītu sīkziņu korpusu, ko izveidojis Peisenieks [15] un rezultāti reģistrēti tabulā 5.16. Redzams, ka uzrādītie rezultāti ir ievērojami sliktāki nekā, lietojot šķērsvalidēšanu. Tomēr, lai arī loģistiskās regresijas akurātums samazinās no 0.739 līdz 0.625, F1 mērs samazinās no 0.635 līdz 0.578, kas ir salīdzinoši neliels samazinājums.

5.16. tabula. Izmantojot galīgo klasificēšanas modeli iegūtie rezultāti

Modelis	Akurātums	F1
SVM	0.625	0.574
Naivais Beijess	0.606	0.553
Loģistiskā regresija	0.625	0.578
Neironu tīkls	0.612	0.555

Kā redzams mašīnapmācības modelis, kas uzrādīja labākos rezultātus, ir loģistiskā regresija. Iegūtos datus ir iespējams salīdzināt ar Špata [19] iegūtajiem rezultātiem (testi 3.4, 4.2, 5.2 un 5.3), kuros iegūtie rezultāti apkopoti tabulā 5.17.

5.17. tabula. G.Špata izveidoto modeļu iegūtie rezultāti testējot uz Peisenieka datu kopas

Modelis	Akurātums	F1
SVM	0.64	0.6
Naivais Beijess	0.63	0.588
K-tuvākos kaimiņus (K=1)	0.601	
K-tuvākos kaimiņus (K=3)	0.597	

Redzams, ka autora iegūtie rezultāti ir nedaudz sliktāki, tomēr atšķirība ir mazāka nekā 2%. Šādi iegūtie rezultāti izskaidrojami ar nelielo izmantoto apmācības datu kopu, kura turklāt satur pārsvarā neitrālas sīkziņas, kas atspoguļojas izveidotā loģistiskās regresijas modeļa tendencē lielākoties kļūdīties starp neitrālo un kādu no sentimenta klasēm, turklāt ļoti vāji tiek veikta negatīvi noskaņoto sīkziņu klasificēšana (5.18. tabula un 5.19 tabula).

5.18. tabula Pārpratumu matrica šķērsvalidēšanā

	POS	NEI	NEG
POS	990	567	10
NEI	205	2839	52
NEG	47	507	106

5.19. tabula Pārpratumu matrica testa datiem

	POS	NEI	NEG
POS	258	124	1
NEI	167	452	8
NEG	21	120	26

SECINĀJUMI

Maģistra darba mērķis bija izpētīt emocionālā noskaņojuma noteikšanas metodes, kā arī sistematizēt un salīdzināt emocionālā noskaņojuma analīzes metodes latviešu valodā rakstītām sīkziņām. Darba gaitā tika secināts, ka latviešu valodā pagaidām nav pieejams pietiekami daudz kvalitatīvu, anotētu datu, ko varētu izmantot, lai izveidotu visus vēlamos klasificēšanas modeļus. Tā piemēram, kvalitatīva neironu tīkla apmācībai nepieciešama salīdzinoši liela apmācības datu kopa – 15000-20000 sīkziņu [26], kas latviešu valodā nav pieejama. Jaunas datu kopas izveidošanas nolūkam tika veikta sīkziņu lejupielāde no sociālā tīkla Twitter un tika izveidota tīmekļa vietne www.calotava.lv, kuras lietotāji varēja novērtēt sīkziņas. Darba gaitā, izmantojot šo tīmekļa vietni tika izveidots jauns anotētu sīkziņu korpuss, kas sastāv no 3618 sīkziņām, ko novērtējuši 1-4 vērtētāji. Lielāka sīkziņu korpusa izveidošana izrādījās problemātiska, jo cilvēki nevēlas veltīt ilgu laiku sīkziņu anotēšanai. Izveidotais korpuss, kā arī darbā lietotais Python kods ir izlikts Github[27], kur to var izmantot jebkurš interesents. Izveidotais korpuss tika apvienots ar vienu no jau esošajiem korpusiem un tika izmantots, lai novērtētu dažādu teksta priekšapstrādes metožu lietderību latviešu valodā rakstītu tekstu apstrādei. Metožu lietderība tika salīdzināta, ar katru metodi apstrādājot tekstu un apmācot modeļus, kuru uzrādītie rezultāti tika salīdzināti ar modeļu, kas apmācīti izmantojot neapstrādātos datus, rezultātiem. Emocijzīmju apstrāde, kas gan nav latviešu valodai specifiska priekšapstrādes metode, nodrošināja lielāko noskaņojuma klasificēšanas modeļu veiktspējas pieaugumu. Modeļu veiktspēju uzlabo arī transliterācijas labošana, stopvārdu dzēšana, skaitļu aizvietošana un vārdu celmošana.

Darba gaitā tika izmēģinātas divas metodes vārdu vienkāršošanai – lemmatizēšana un celmošana. Lemmatizēšana ir efektīva faktoru vektora garuma samazināšanai, taču vienlaikus tika novērots, ka izmantotajam rīkam nav unicode atbalsta, turklāt tam ir sava metode teikuma sadalīšanai tekstvienībās, kas padarīja šī rīka izmantošanu nelietderīgu. Tā vietā tika izmantots celmošanas rīks, kas samazināja faktoru vektora garumu par gandrīz 15% un uzlaboja izveidoto modeļu darbību.

Papildus 2-grammu izmantošana un 3-grammu izmantošana uzlabo klasificēšanas rezultātus salīdzinot ar unigrammu izmantošanu, taču, tā kā klasificēšanas rezultātu atšķirība starp modeļiem, kas lietoja uni- un 2-grammas un modeļiem kas lietoja uni- 2- un 3-grammas,

[27] https://github.com/Rinalds Viksna/sikzinu_analize

bija neliela, taču faktoru skaits pieauga gandrīz divkārt, tad tika pielietotas uni- un 2-grammas, jo faktoru skaita pieaugums pasliktina modeļa ātrdarbību un prasa lielu daudzumu datora resursu klasificēšanas modeļu apmācībai.

Izmantojot identificētās priekšapstrādes metodes, tika izveidota metožu secība, kuru izmantojot, tika veikta galīgo noskaņojuma klasificēšanas modeļu apmācīšana. Apmācītie modeļi tika novērtēti gan, izmantojot šķērsvalidāciju, gan arī testēti izmantojot Peisenieka sīkziņu korpusu. Labākos noskaņojuma klasificēšanas rezultātus deva loģistiskās regresijas modelis. Tas šķērsvalidēšanā uzrādīja 74% akurātumu un 64% F1, bet, testējot, izmantojot ārēju korpusu, 63% akurātumu un 58% F1. Salīdzinot ar labākajiem Špata [19] iegūtajiem rezultātiem (testā 4.2 akurātums 64% un F1 60% izmantojot SVM), izveidotais modelis ir nedaudz sliktāks, kas izskaidrojams ar mazāku izmantoto apmācības datu korpusu.

Iespējamie tālākie pētījumu virzieni iekļauj vārdu iegulšanas metožu izmantošanas izvērtēšana, kā arī vēl lielāka anotētu datu korpusa izveidošanu un esošā korpusa kvalitātes uzlabošanu, katru sīkziņu novērtējot vairākas reizes, uzlabojot datu ticamību. Noskaņojuma analīze kopumā vēl ir plašs darba lauks, jo pastāv tādas problēmas kā sarkasma atklāšana un negāciju noliegumi, ko ar vienkāršiem modeļiem atklāt ir ievērojami grūtāk.

Darba rezultātu praktiskais pielietojums varētu būt tādu sistēmu izstrāde, kas kopā ar nosaukto vienumu meklēšanu veiktu tīmekļa tekstu analīzi ar mērķi noskaidrot sabiedrības viedokli par tiem.

LITERATŪRA

1. Akadēmiskā terminu datubāze - Sentiment analysis. http://termini.lza.lv/term.php?term=Sentiment analysis&lang=EN. Accessed 31 Jan 2018
2. Liu B (2012) Sentiment Analysis and Opinion Mining.
3. Gunther T (2013) Sentiment Analysis of Microblogs. Master's thesis, Univ. Gothenbg.
4. Thomson Reuters Adds Unique Twitter and News Sentiment Analysis to Thomson Reuters Eikon | Thomson Reuters. https://www.thomsonreuters.com/en/press-releases/2014/thomson-reuters-adds-unique-twitter-and-news-sentiment-analysis-to-thomson-reuters-eikon.html. Accessed 8 Mar 2018
5. Chen L, Chen G, Wang F (2015) Recommender Systems Based on User Reviews : The State of the Art. Syst. Rev. 2015 4:
6. Ceron A (2013) Enlightening the voters : The effectiveness of alternative electoral strategies in the 2013 Italian election monitored through (sentiment) analysis of Twitter posts. Ecpr 1–25.
7. Ravi K, Ravi V (2015) A survey on opinion mining and sentiment analysis: Tasks, approaches and applications. Knowledge-Based Syst 89:14–46. doi: 10.1016/j.knosys.2015.06.015
8. Medhat W, Hassan A, Korashy H (2014) Sentiment analysis algorithms and applications: A survey. Ain Shams Eng J 5:1093–1113. doi: 10.1016/j.asej.2014.04.011
9. Vohra S, Teraiya J (2013) Applications and Challenges for Sentiment Analysis : A Survey. Int J Eng Res Technol 2:1–6.
10. Kharde VA, Sonawane SS (2016) Sentiment Analysis of Twitter Data: A Survey of Techniques. Int J Comput Appl 139:975–8887. doi: 10.5120/ijca2016908625
11. Dashtipour K, Poria S, Hussain A, et al (2016) Multilingual Sentiment Analysis: State of the Art and Independent Comparison of Techniques. Cognit Comput 8:757–771. doi: 10.1007/s12559-016-9415-7
12. Korayem M, Aljadda K, Crandall D (2016) Sentiment/subjectivity analysis survey for languages other than English. Soc Netw Anal Min 6:1–28. doi: 10.1007/s13278-016-0381-6
13. Garkaje G, Zilgalve E, Dargis R (2014) Normalization and Automatized Sentiment Analysis of Contemporary Online Latvian Language. Front Artif Intell Appl 268:83–

86. doi: 10.3233/978-1-61499-442-8-83

14. Špats G, Birzniece I (2016) Opinion Mining in Latvian Text Using Semantic Polarity Analysis and Machine Learning Approach. 51–59. doi: http://dx.doi.org/10.7250/csimq.2016-7.03
15. Peisenieks J, Skadiņš R (2014) Uses of Machine Translation in the Sentiment Analysis of Tweets. Front Artif Intell Appl 268:126–131. doi: 10.3233/978-1-61499-442-8-126
16. Devika MD, Sunitha C, Ganesh A (2016) Sentiment Analysis: A Comparative Study on Different Approaches. Procedia Comput Sci 87:44–49. doi: 10.1016/j.procs.2016.05.124
17. L.Olson D, Dursun D (2008) Advanced Data Mining Techniques. doi: 10.1007/978-3-540-76917-0
18. Asch V Van (2013) Macro-and micro-averaged evaluation measures [[BASIC DRAFT]]. 1–27. doi: 10.3386/w19544
19. Špats G (2015) Application of Opinion Mining for written content classification in Latvian text. Master's thesis
20. Patodkar VN, I.R S (2016) Twitter as a Corpus for Sentiment Analysis and Opinion Mining. Ijarcce 5:320–322. doi: 10.17148/IJARCCE.2016.51274
21. Datasets - Linked Data Models for Emotion and Sentiment Analysis Community Group. https://www.w3.org/community/sentiment/wiki/Datasets. Accessed 8 Mar 2018
22. Esuli A, Sebastiani F (2006) SENTIWORDNET: A Publicly Available Lexical Resource for Opinion Mining. Proc 5th Conf Lang Resour Eval 417–422. doi: 10.1.1.61.7217
23. MPQA Resources. http://mpqa.cs.pitt.edu/. Accessed 8 Mar 2018
24. Galinsky R, Alekseev A, Nikolenko S (2016) Improving Neural Network Models for Natural Language Processign in Russian with Synonyms. PROCEEDING AINL Fruct 2016 Conf. 3:
25. Peisenieks J (2014) Mašīntulkošanas iespējas Twitter sīkziņu sentimenta analīzē. Bachelor thesis
26. Nicmanis D (2017) Sabiedrības attieksmes modelēšana, izmantojot sentimenta analīzi. Bachelor thesis
27. Gediņš K (2013) Automātiskā teksta emocionālās noskaņas noteikšana latviešu valodā. Bachelor thesis
28. Vīksna R, Jēkabsons G (2018) Sentiment Analysis in Latvian and Russian: A Survey.

Applied Computer Systems, Vol. 23, 2018, 7 p. (in press) ISSN 2255-8683, e-ISSN 2255-8691

29. Sakenovich NS (2017) On One Approach of Solving Sentiment Analysis Task for Kazakh and Russian Languages Using Deep Learning. 10449:537–545. doi: 10.1007/978-3-319-67077-5
30. Bobichev V, Kanishcheva O, Cherednichenko O (2017) Sentiment analysis in the Ukrainian and Russian news. 2017 IEEE 1st Ukr Conf Electr Comput Eng UKRCON 2017 - Proc 1050–1055. doi: 10.1109/UKRCON.2017.8100410
31. Loukachevitch N V., Chetviorkin II (2014) Open evaluation of sentiment-analysis systems based on the material of the Russian language. Sci Tech Inf Process 41:370–376. doi: 10.3103/S0147688214060057
32. Sosa PM, Sadigh S (2016) Twitter Sentiment Analysis with Neural Networks. 1–12.
33. Shalunts G, Backfried G (2015) SentiSAIL: Sentiment Analysis in English, German and Russian. doi: 10.1007/978-3-319-21024-7
34. Gulbinskis I (2010) Digitālo tekstu sentimenta analīze. Bachelor thesis
35. Webb GISC (2017) Encyclopedia of Machine Learning and Data Mining. doi: 10.1007/978-1-4899-7687-1
36. Genriha I, Voronova I (2011) Methods for Evaluating the Creditworthiness of Borrowers. Econ Bus 22:42–50.
37. Hastie T, Tibshirani R, Friedman J (2009) The Elements of Statistical Learning. Bayesian Forecast Dyn Model 1:1–694. doi: 10.1007/b94608
38. Xu N, Mao W (2017) A residual merged neutral network for multimodal sentiment analysis. 2017 IEEE 2nd Int Conf Big Data Anal ICBDA 2017 6–10. doi: 10.1109/ICBDA.2017.8078794
39. Nigam K, Lafferty J, Mccallum A (1999) Using Maximum Entropy for Text Classification. IJCAI-99 Work Mach Learn Inf Filter 61–67. doi: 10.1.1.63.2111
40. Loukachevitch N, Rubtsova Y (2015) Entity-Oriented Sentiment Analysis of Tweets: Results and Problems. 9302:551–559. doi: 10.1007/978-3-319-24033-6
41. Tutubalina E, Nikolenko S (2018) Constructing Aspect-Based Sentiment Lexicons with Topic Modeling. 10716:208–220. doi: 10.1007/978-3-319-73013-4
42. Turney PD (2002) Thumbs up or thumbs down? Semantic Orientation applied to Unsupervised Classification of Reviews. Proc 40th Annu Meet Assoc Comput Linguist 417–424. doi: 10.3115/1073083.1073153

43. Khan MT, Durrani M, Ali A, et al (2016) Sentiment analysis and the complex natural language. Complex Adapt Syst Model 4:2. doi: 10.1186/s40294-016-0016-9
44. Fleiss JL (1971) Measuring nominal scale agreement among many raters. Psychol Bull 76:378–382.
45. Paikens P, Rituma L, Pretkalniņa L (2013) Morphological analysis with limited resources : Latvian example. 267–277.
46. Krēsliņš K (1996) A stemming algorithm for Latvian.
47. Novak PK, Smailović J, Sluban B, Mozetič I (2015) Sentiment of emojis. PLoS One 10:1–22. doi: 10.1371/journal.pone.0144296