
**Kauno technologijos universitetas**

Informatikos fakultetas

# Neapykantos kalbos socialinių tinklų tekstuose atpažinimo metodas

Baigiamasis magistro studijų projektas

**Martynas Čepas**

Projekto autorius

**Prof. Agnius Liutkevičius**

Vadovas

**Kaunas, 2025**

**Kauno technologijos universitetas**

Informatikos fakultetas

# Neapykantos kalbos socialinių tinklų tekstuose atpažinimo metodas

Baigiamasis magistro studijų projektas

Informacijos ir informacinių technologijų sauga (6211BX008)

**Martynas Čepas**

Projekto autorius

**Prof. Agnius Liutkevičius**

Vadovas

**Prof. Jevgenijus Toldinas**

Recenzentas

**Kaunas, 2025**

**Kauno technologijos universitetas**

Informatikos fakultetas

Martynas Čepas

# Neapykantos kalbos socialinių tinklų tekstuose atpažinimo metodas

Akademinio sąžiningumo deklaracija

Patvirtinu, kad:

1. baigiamąjį projektą parengiau savarankiškai ir sąžiningai, nepažeisdama(s) kitų asmenų autoriaus ar kitų teisių, laikydamasi(s) Lietuvos Respublikos autorių teisių ir gretutinių teisių įstatymo nuostatų, Kauno technologijos universiteto (toliau – Universitetas) intelektinės nuosavybės valdymo ir perdavimo nuostatų bei Universiteto akademinės etikos kodekse nustatytų etikos reikalavimų;

2. baigiamajame projekte visi pateikti duomenys ir tyrimų rezultatai yra teisingi ir gauti teisėtai, nei viena šio projekto dalis nėra plagijuota nuo jokių spausdintinių ar elektroninių šaltinių, visos baigiamojo projekto tekste pateiktos citatos ir nuorodos yra nurodytos literatūros sąraše;

3. įstatymų nenumatytų piniginių sumų už baigiamąjį projektą ar jo dalis niekam nesu mokėjęs (-usi);

4. suprantu, kad išaiškėjus nesąžiningumo ar kitų asmenų teisių pažeidimo faktui, man bus taikomos akademinės nuobaudos pagal Universitete galiojančią tvarką ir būsiu pašalinta(s) iš Universiteto, o baigiamasis projektas gali būti pateiktas Akademinės etikos ir procedūrų kontrolieriaus tarnybai nagrinėjant galimą akademinės etikos pažeidimą.

Martynas Čepas

*Patvirtinta elektroniniu būdu*

Martynas Čepas. Neapykantos kalbos socialinių tinklų tekstuose atpažinimo metodas. Magistro studijų baigiamasis projektas / vadovas prof. Agnius Liutkevičius; Kauno technologijos universitetas, Informatikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Informatikos inžinerija (Informatikos mokslai).

Reikšminiai žodžiai: neapykantos kalba, socialiniai tinklai, natūralios kalbos apdorojimas, gilusis mokymasis, LSTM, teksto analizė, lietuvių kalba, komentarų klasifikacija, duomenų apdorojimas.

Kaunas, 2025. 66 p.

## Santrauka

Šiame magistro baigiamajame darbe pristatomas neapykantos kalbos atpažinimo metodas lietuvių kalba, pritaikytas socialinių tinklų tekstams. Atsižvelgiant į tai, kad egzistuojančios neapykantos kalbos atpažinimo sistemos daugiausia orientuotos į anglų kalbą, šio darbo tikslas buvo sukurti lietuvių kalbai pritaikytą sprendimą. Buvo surinktas, anotuotas ir išanalizuotas *Reddit* platformos komentarų rinkinys. Komentarai buvo kruopščiai apdoroti taikant specialiai lietuvių kalbai pritaikytus tekstų valymo žingsnius, įskaitant stop-žodžių pašalinimą, kamienavimą, žargono vertimą ir simbolių normalizavimą.

Tyrimo metu sukurtas LSTM neuroninis tinklas, kuris pasiekė 97 % tikslumą klasifikuodamas komentarus kaip neapykantos kalbą arba ne. Modelio veikimas buvo palygintas su kitais moderniais sprendimais, tokiais kaip BERT ir LitLat BERT. Eksperimentai parodė, kad tinkamai apdorotas tekstas žymiai pagerina klasifikacijos rezultatus, o sukurtas modelis gerokai lenkia alternatyvius metodus lietuvių kalboje.

Darbas prisideda prie saugesnės internetinės erdvės kūrimo, pateikdamas praktinį sprendimą neapykantos kalbos atpažinimui lietuvių kalboje.

Čepas Martynas A Method for Recognizing Hate Speech in Social Media Texts. Master's Final Degree Project / supervisor prof. Agnius Liutkevičius; Faculty of Informatics, Kaunas University of Technology.

Study field and area (study field group): Informatics Engineering (Computing).

Keywords: hate speech, social media, natural language processing, deep learning, LSTM, text analysis, Lithuanian language, comment classification, data preprocessing.

Kaunas, 2025. 66 p.

**Summary**

This Master's thesis presents a hate speech detection method tailored to the Lithuanian language, specifically targeting text data from social media platforms. Given that most existing hate speech detection systems are focused on English, the goal of this work was to design a solution adapted for Lithuanian. A dataset of comments was collected and manually annotated from Reddit. The preprocessing pipeline was carefully optimized for Lithuanian, including stopword removal, stemming, slang normalization, and character transformation.

The developed LSTM neural network achieved an 97% accuracy when classifying comments as hate speech or not. The model’s performance was compared to modern alternatives such as BERT and LitLat BERT. The experiments demonstrated that thorough preprocessing significantly boosts classification performance, and the proposed model outperforms other methods in the Lithuanian context.

This project contributes to building a safer digital space by offering a practical approach for Lithuanian hate speech detection.

**Turinys**

## Lentelių sąrašas

## Paveikslų sąrašas

# Įvadas

Socialiniai tinklai ir žiniasklaida tapo svarbia mūsų kasdienio gyvenimo dalimi, suteikiančia galimybę bendrauti, dalytis informacija ir socialiai sąveikauti. Tačiau šios platformos turi ir tamsiąją pusę, nes jomis gali būti naudojamasi skleidžiant neapykantą kurstančias kalbas, patyčias ir kitas prievartos internete formas.

Neapykantos kalba - tai komunikacijos forma, kuria išreiškiamas išankstinis nusistatymas, diskriminacija ar priešiškumas tam tikrai grupei ar asmeniui dėl jų rasės, etninės kilmės, tautybės, religijos, seksualinės orientacijos, lytinės tapatybės ar kitų savybių. Socialiniuose tinkluose neapykantos kalba kelia problemas tiek naudotojams, tiek pačioms platformoms. Neapykantos kalbos aptikimas įprastai vykdomas automatiniais metodais, yra sukurta aibė tiksliai veikiančių mašininio mokymosi metodais paremtų neapykantos kalbos aptikimo įrankių, tačiau visi jie pritaikyti populiariausioms pasaulio kalboms, o lietuvių kalbą palaikančių metodų tiesiog nėra.

Darbo tikslas - sukurti lietuvių kalbai pritaikytą neapykantos kalbos atpažinimo sistemą, apimančią duomenų rinkinio sudarymą, teksto apdorojimą, giluminio mokymosi modelio kūrimą bei jo palyginamąją analizę su moderniais sprendimais. Taip pat kuriamas modelį naudojantis API bei naršyklės plėtinys, kurie leidžia realiu laiku patikrinti, ar tekstas yra neapykantos kalba.

Projekto uždaviniai:

1. atlikti neapykantos problemos internete analizę.
2. parengti neapykantos kalbos aptikimo įrankio pritaikyto lietuvių kalbai prototipo projektą.
3. sukurti neapykantos kalbos aptikimo sistemos, pritaikytos lietuvių kalbai, prototipą, sukuriant duomenų rinkinį, atliekant lietuvių kalbai pritaikytą pirminį apdorojimą, sudarant mašininio mokymosi modelį bei panaudojant jį kuriant naršyklės plėtinį, kuris galėtų aptikti lietuvišką neapykantos kalbą.
4. atlikti sukurto mašininio mokymosi modelio eksperimentinį vertinimą ir palyginti su kitais sprendimais.

Projektą sudaro 4-ios dalys: analizėje apžvelgiama neapykantos kalbos problema ir teksto analizės metodai, projekto dalyje aprašoma tai, kas bus kuriama ir kokie metodai bus naudojami. Prototipo dalyje aprašomas duomenų rinkinio sudarymas, apdorojimas ir modelio kūrimas, o eksperimente pateikiamas sukurto modelio palyginimas su kitais sprendimais.

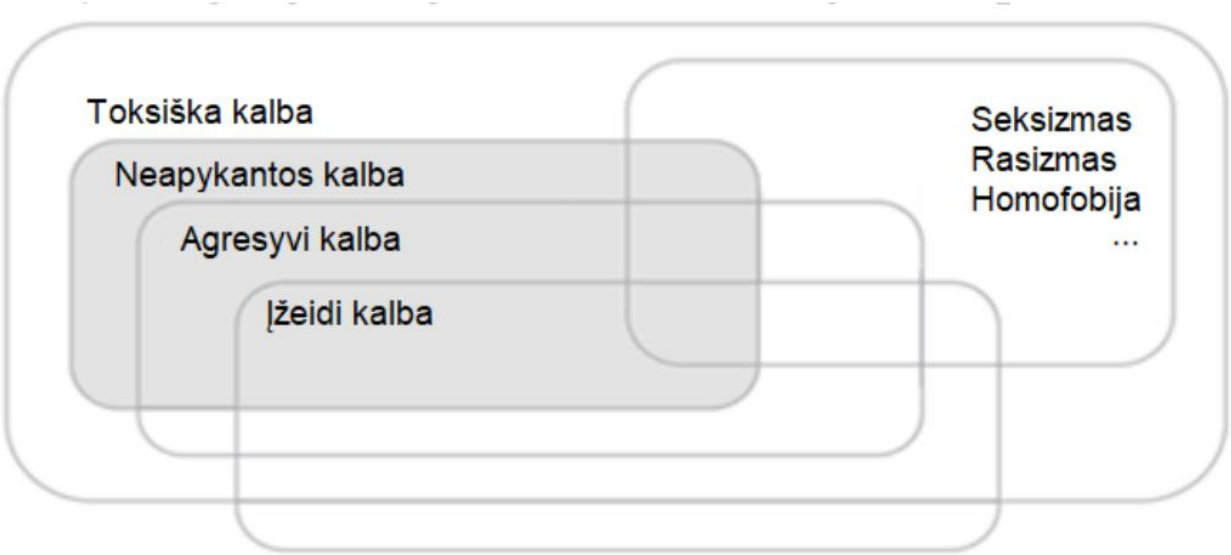
## 1. Neapykantos kalbos socialiniuose tinkluose problemos analizė

Šiame skyriuje analizuojama bei atskleidžiama, kokias problemas tiek socialinių medijų naudotojams, tiek pačioms platformoms kelia neapykantos kalba. Pristatomi tyrimai šioje srityje, mašininio mokymosi metodai naudojami neapykantos kalbos aptikimui.

### 1.1. Neapykantos kalbos apibrėžimas

Neapykantos kalbos atpažinimas – tai sudėtinga ir daugiasluoksnė užduotis, kuriai reikia specifinių sprendimų. Šioje srityje dirbantys tyrėjai kuria įvairius įrankius, formuoja anotuotus duomenų rinkinius, išskiria esminius kalbos požymius ir testuoja klasifikavimo metodus. Tyrimai, orientuoti į neapykantos kalbos analizę skirtingomis kalbomis ir palyginamųjų tekstynų rengimas, skatina tolesnę automatizuoto atpažinimo pažangą. Tokie sprendimai tampa vis svarbesni siekiant mažinti neapykantos ir smurto internete plitimą bei kovoti su dezinformacija.

Vienas iš pagrindinių iššūkių yra tai, kad nėra vienareikšmio neapykantos kalbos apibrėžimo. Į šią sąvoką įeina susiję terminai, tokie kaip įžeidžianti kalba, agresyvūs išsireiškimai ar diskriminacija, įskaitant ir rasizmo formas. Išsamus šių sąvokų atvaizdavimas pateikiamas **1 pav**.




**1 pav**. Neapykantos kalbos ir susijusių sąvokų ryšiai [1]

Nepaisant skirtingų apibrėžimų, tyrėjai pastebi tam tikrą nuoseklumą. Pavyzdžiui, „Fortuna and Nunes“ [2] atliko įvairių apibrėžimų analizę ir išskyrė bendrus bruožus:

- neapykantos kalba visada yra nukreipta į konkretų subjektą;
- ji skatina priešiškumą arba smurtą;
- tokia kalba dažnai pasižymi žeminimu ar puolimu;
- kai kuriais atvejais ji pasireiškia per humorą ar sarkazmą.

Ši įvairovė leidžia manyti, kad neapykantos kalbos suvokimas gali kisti priklausomai nuo konteksto, kultūros ar net vartotojo patirties.

## 1.2. Neapykantos kalbos socialinėse medijose problema

Neapykantos kalbos turi rimtų pasekmių tikslinei grupei ir visai visuomenei, nes skatina susiskaldymą, diskriminaciją ir smurtą.

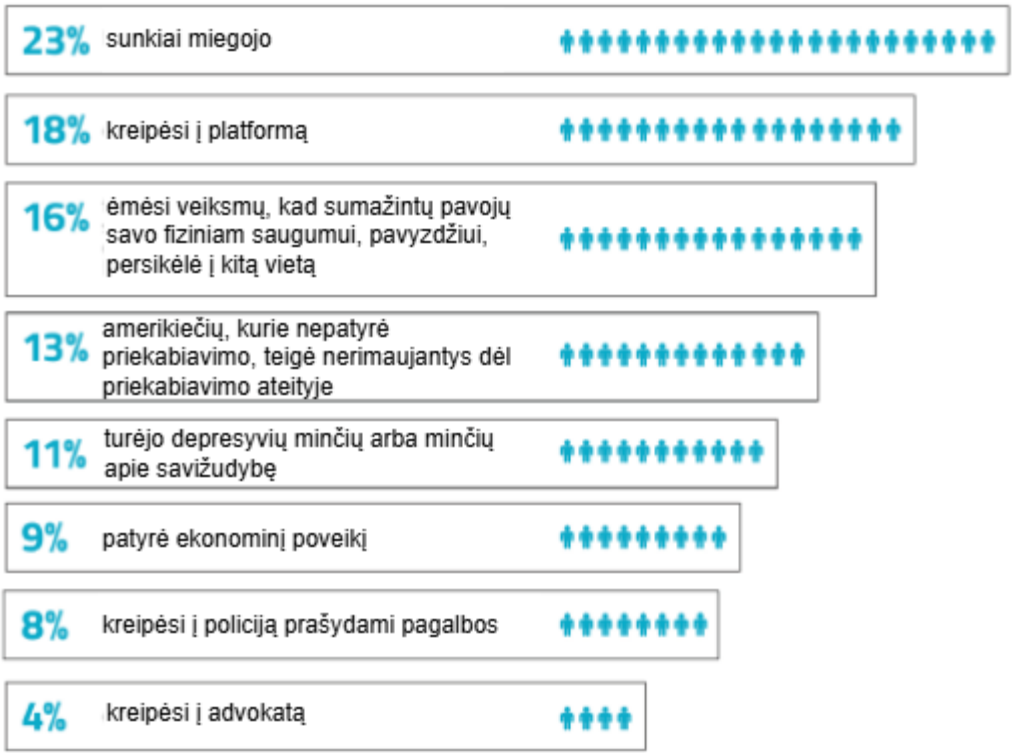
Vienas iš pagrindinių neigiamų neapykantos kalbos socialiniuose tinkluose padarinių yra žala, kurią ji daro asmenims ir grupėms, į kurias ji nukreipta. Neapykantos kalba gali sukelti emocinį sutrikimą, nerimą, depresiją ir kitus psichikos sveikatos sutrikimus. Dėl jos asmenys gali jaustis taikiniais ir nuvertinti, o tai gali turėti neigiamą poveikį jų savigarbai ir gerovei. Neapykantos kalbos taip pat gali sukelti fizinę žalą, nes kai kurie asmenys ar grupės gali jausti grėsmę arba užpulti asmenis ar grupes, prieš kuriuos nukreipta neapykanta. Yra buvę atvejų, kai dėl neapykantos kalbos socialinėje žiniasklaidoje asmenys buvo fiziškai užpulti, jiems buvo grasinama arba jie buvo persekiojami.

Organizacijos „ADL“ (pirmaujanti pasaulyje organizacija, kovojanti su neapykanta) atliktame tyrime [3] apie neapykantą ir priekabiavimą internete buvo nustatyta, kad 41% amerikiečių susidūrė su patyčiomis internete. Apklausos rezultatai vizualiai pateikti **2 pav**.


**2 pav**. Neapykantos kalbos internete paplitimas [3]

Daugelis respondentų, į kuriuos buvo nukreiptas neapykantos ar priekabiavimo turinys arba kurie bijojo, kad į juos bus nukreiptas, nurodė, kad tai paveikė jų ekonominę, emocinę, fizinę ir psichologinę gerovę. 23 proc. respondentų nurodė, kad jiems sunku miegoti, susikaupti ir jaučia nerimą. Nepaisant pranešimų, kad mažiau platformų ėmėsi veiksmų, kai buvo pranešta apie

priekabiavimą, beveik penktadalis priekabiavimo taikinių (18 %) kreipėsi į platformą, kad paprašytų pagalbos arba praneštų apie priekabiavimo turinį. Dar 16 proc. respondentų ėmėsi veiksmų, kad sumažintų pavojų savo fiziniam saugumui, pavyzdžiui, persikėlė į kitą vietą, pakeitė kelionės į darbą ir atgal maršrutą, lankė savigynos kursus, vengė būti vieni arba vengė tam tikrų vietų. Apklausai rezultatai pateikti **3 pav**.




**3 pav**. Neapykantos kalbos poveikis [3]

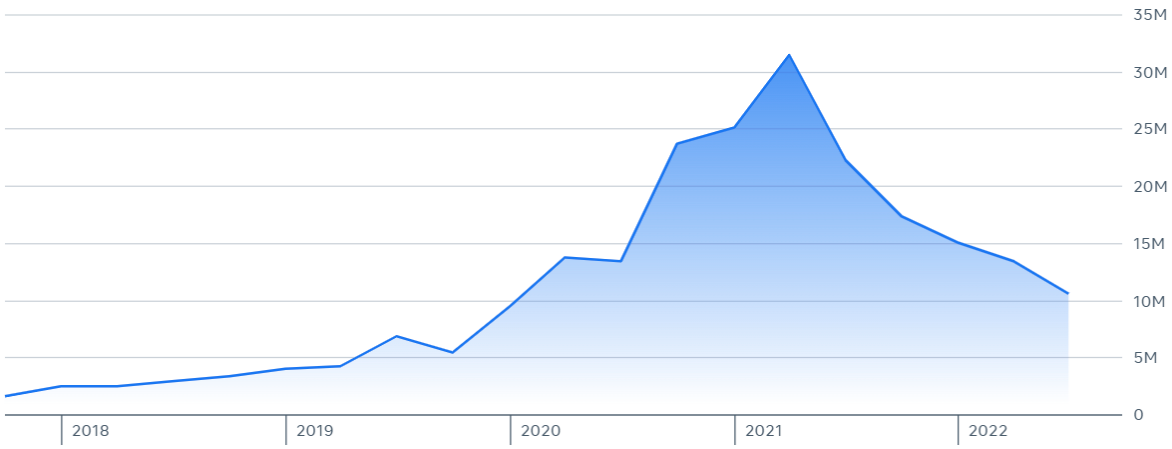
Neapykantos kalba taip pat gali turėti socialinių pasekmių, nes dėl jos gali atsirasti izoliacija, diskriminacija ir išstūmimas iš tam tikrų socialinių sluoksnių ar bendruomenių. Neapykantos kalbos gali sukelti baimės ir nepasitikėjimo jausmą tam tikrose grupėse, dėl to sumažėja socialinis bendravimas ir atsiranda nepritapimo jausmas. Tai gali turėti neigiamą poveikį socialiniam ir emociniam asmenų vystymuisi, taip pat bendrai jų gyvenimo kokybei.

Neapykantos kalba taip pat tai gali prisidėti prie neapykantą kurstančių ideologijų ir pažiūrų normalizavimo ir plitimo, gali kurstyti smurtą ir kitokias prievartos formas. Neapykantos kalbos apžvalgoje interneto socialinių tinklų kontekste [4] teigiama, kad neapykantos kalba ir terorizmas yra labai paplitę ir glaudžiai susiję reiškiniai. Buvo nustatyta, kad vyriausybė, formuodama tinkamą politiką susijusia su neapykanta internete ir bendradarbiaujant su interneto paslaugų teikėjais (IPT) bei socialiniais tinklais, gali padaryti kovą su neapykantos kalba ir terorizmu veiksminga. Todėl būtina parengti politiką ir metodus, kaip užkirsti kelią šiai veiklai internete ir ją kontroliuoti.

Be neigiamo poveikio asmenims ir grupėms, neapykantos kalba socialiniuose tinkluose gali turėti neigiamą poveikį ir pačiai platformai. Dėl jos gali sumažėti naudotojų įsitraukimas ir pasitenkinimas, nes žmonės gali būti mažiau linkę naudotis platforma, jei joje nėra saugios ir pagarbios aplinkos. „Data&Society" tyrimų instituto atliktame tyrime apie priekabiavimą internete, skaitmeninėje erdvėje

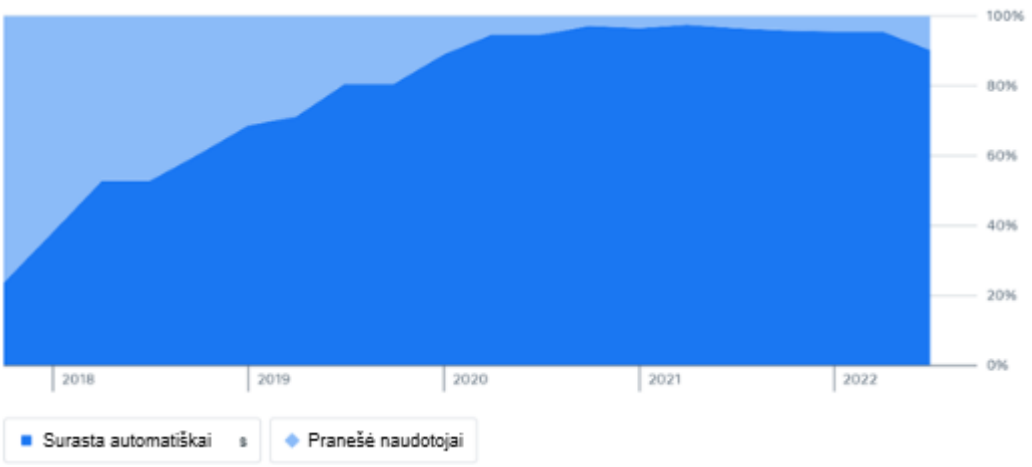
[5] nustatyta, kad 27 proc. visų Amerikos interneto naudotojų cenzūruoja savo įrašus internete baiminantis priekabiavimo. Tai reiškia, kad beveik trečdalis socialinių tinklų naudotojų nesinaudoja šiomis priemonėmis tiek, kiek naudotųsi, jei jaustųsi saugūs. Taip pat gali sumažėti pasitikėjimas ir patikimumas, nes žmonės gali būti mažiau linkę pasitikėti platforma ir jos teikiama informacija, jei joje leidžiama neapykantos kalba ir kitų formų piktnaudžiavimas. Tai taip pat gali sukelti teisinių ir reguliavimo pasekmių, nes kai kuriose šalyse galioja įstatymai, draudžiantys neapykantos kalbą ir kitų formų prievartą internete. Platformoms, kurios leidžia neapykantos kalbą, gali būti skiriamos baudos, iškeliamos bylos ar imamasi kitų teisinių veiksmų, o tai gali turėti finansinių pasekmių ir neigiamų pasekmių reputacijai.

Remiantis socialinio tinklo „Facebook“ pateiktais savais duomenimis „Hate Speech“ kategorijoje [6] matoma, kad pastaraisiais metais itin išaugo dėmesys šiai temai – nuo 2.5 milijono įrašų pašalinimo 2018 metais iki 31.5 milijono 2021-aisiais. Šie duomenys vizualizuoti **4 pav**.



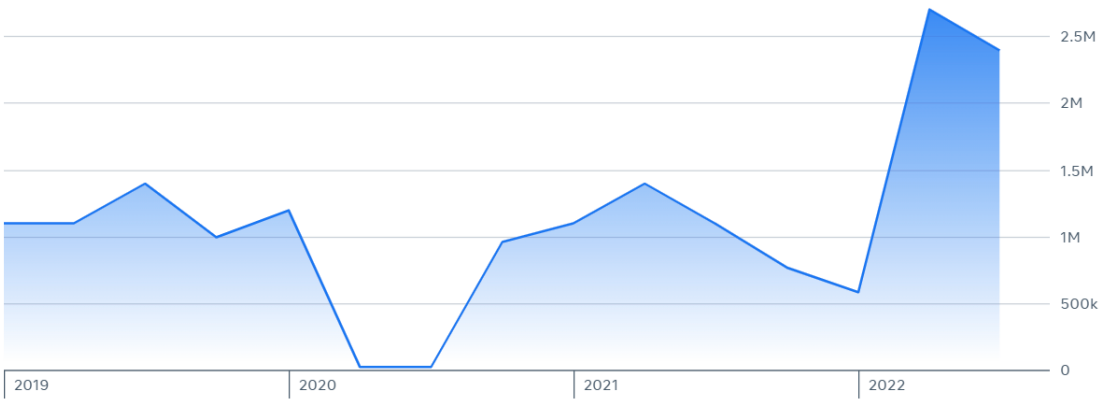

**4 pav**. „Facebook“ turinys, dėl kurio imtasi veiksmų [6]

Taip pat matoma **5 pav**. pateiktoje diagramoje matoma, kad tuo laikotarpiu buvo pradėtos naudoti automatinės priemonės neapykantą skleidžiančiam turiniui aptikti – 2018-aisiais 52.90% turinio buvo aptikta iki naudotojo pranešimo, o 2022-aisiais šis skaičius išaugo iki 95.60% - matome, kad smarkiai išaugo neapykantą skleidžiančio turinio aptikimo galimybės.




**5 pav**. Automatiškai aptikti neapykantos kalbos įrašai „Facebook“ [6]

Tačiau šis aptikimas, nors ir apdoroja didžiulius kiekius informacijos, nėra tikslus – apie 2.7 milijono cenzūruotų įrašų 2022-aisiais buvo apeliuota įrašų kūrėjų, o apie 0.5 milijono iš jų pripažinti neteisingais. Taigi, net patiems pažangiausiems aptikimo metodams tikrai yra vietos tobulėti. Klaidingai aptiktų įrašų kiekvienais metais diagrama pateikta **6 pav**.




**6 pav**. Klaidingai aptikti neapykantos kalbos įrašai "Facebook" [6]

Apibendrinant galima teigti, kad neapykantos kalba socialiniuose tinkluose kelia reikšmingas emocines, fizines, socialines ir ekonomines pasekmes tiek tikslinei grupei, tiek platesnei visuomenei. Nors pastaraisiais metais socialinės medijos platformos ženkliai padidino pastangas automatiškai aptikti ir šalinti neapykantos kupiną turinį, šie metodai vis dar nėra pakankamai tikslūs, todėl kyla poreikis toliau tobulinti aptikimo technologijas bei politikos priemones. Ypač svarbu užtikrinti efektyvų technologijų ir žmogaus moderavimo derinimą, siekiant sukurti saugią ir pagarbią skaitmeninę erdvę.

## 1.3. Neapykantos skleidimo socialiniuose tinkluose sprendimo iššūkiai

### 1.3.1. Kalbos niuansų ir konteksto sudėtingumas

Socialinių tinklų įrašai dažnai pasižymi elementais, tokiais kaip emocijas reiškiantys *emoji* ar *hashtag'ai*, ir būna parašyti laisvu stiliumi su klaidomis, o tai apsunkina automatinį neapykantos kalbos atpažinimą [7]. Ironija ir sarkazmas, kuriais neretai išreiškiama subtili arba užmaskuota neapykanta, dažnai yra semantiškai dviprasmiški, todėl standartiniai modeliai juos identifikuoja netiksliai [8]. Konteksto trūkumas taip pat stipriai komplikuoja klasifikavimo užduotis – tyrimai rodo, kad be papildomos diskurso analizės patikimai atskirti neapykantos kupinus pranešimus nuo neutralių yra itin sudėtinga [9].

### 1.3.2. Įvairovė ir nestandartinė kalba

Socialiniuose tinkluose vartotojai dažnai naudoja nestandartinę rašybą, gramatiką, žargoną, santrumpas ar specifinius rašymo būdus, kuriuos sunku apdoroti tradiciniais natūralios kalbos apdorojimo metodais [10]. Be to, vartotojai dažnai maišo skirtingas kalbas ar tarmes, o modeliai, apmokyti pagal vienakalbius duomenis, tokio mišraus turinio negeba tinkamai interpretuoti. Daugiakalbystė tokiu atveju padidina klaidų tikimybę ir sumažina neapykantos kalbos aptikimo tikslumą.

### 1.3.3. Duomenų žymėjimo ir anotavimo iššūkiai

Prižiūrimi mašininio mokymosi metodai reikalauja didelių kiekių rankiniu būdu pažymėtų duomenų, o šis procesas yra ne tik brangus, bet ir laiko imlus – vieno tokio duomenų rinkinio anotavimas gali kainuoti dešimtis tūkstančių dolerių [11]. Ekspertinių žinių reikalavimas anotavimo procese dar labiau padidina finansines ir laiko sąnaudas [12]. Siekiant sumažinti šias problemas, taikomi silpnai prižiūrimi metodai, kurie naudoja mažesnius pažymėtų duomenų kiekius ir remiasi papildomais išoriniais informacijos šaltiniais, tokiais kaip vartotojų profiliai ar socialinių tinklų struktūra [13].

### 1.3.4. Modelių ir metodų ribotumai

Nors prižiūrimo mokymo metodai paprastai pasižymi aukštu tikslumu, jie dažnai sunkiai apibendrina naujose duomenų srityse ar subtilaus turinio atvejais, nes prielaidos apie skirtingas neatpažinto neapykantos turinio savybes ne visada pasiteisina [14]. NLP metodai, tokie kaip nuotaikų analizė, yra jautrūs žodžių poliariškumo klaidoms dėl sarkazmo ar nestandartinės leksikos, o tai gali iškreipti galutinius rezultatus. Nors diskurso analizė ir pažangūs dėmesio mechanizmo modeliai (pvz., HAN ar BERT) gerina konteksto supratimą, jų efektyvumą riboja teksto įvairovė, kalbinis sudėtingumas ir dideli skaičiavimo resursų poreikiai [15].

### 1.3.5. Tradicinio rankinio moderavimo iššūkiai

Daugelis socialinių tinklų platformų pasitiki žmonių moderatoriais, tačiau tokia praktika dažnai būna neefektyvi – pavyzdžiui, vien „Facebook“ samdo apie 15 000 turinio peržiūros specialistų, tačiau net ir toks kiekis negali tinkamai susidoroti su didžiuliu duomenų srautu ir subtilia neapykantos kalba [16]. Be to, moderavimo procese neišvengiamai pasireiškia žmogiškasis šališkumas - nepakankamos

kalbinės kompetencijos ir nevienodai taikomų turinio gairių pasekmė yra nevienodas politikos taikymas [17].

## 1.4. Egzistuojantys neapykantos kalbos aptikimo įrankiai

Atlikta automatizuotų įrankių ir platformų, skirtų neapykantos kalbai tekste aptikti, analizė. Viešai prieinami įrankiai yra riboti, o išsamią informaciją apie juos rasti sudėtinga.

„Perspective API“ [18] - tai „Jigsaw“ sukurtas įrankis, naudojantis natūralios kalbos apdorojimo ir mašininio mokymosi algoritmus, skirtus potencialiai toksiškam turiniui, tokiam kaip neapykantos kalba ar patyčios, identifikuoti. Platformą lengva integruoti į įvairius sprendimus, ji pasižymi aukštu tikslumu ir geba apdoroti didelius duomenų srautus realiuoju laiku. Tačiau ši priemonė gali nepakankamai tiksliai nustatyti subtilias neapykantos kalbos formas, susidurti su šališkumo problemomis ir nėra atvirojo kodo. Be to, šiuo metu „Perspective API“ oficialiai palaiko tik anglų, prancūzų, ispanų ir dar kelias pagrindines kalbas, o lietuvių kalba tarp jų nepalaikoma Rekomenduojama šį įrankį naudoti kartu su žmogiškuoju moderavimu, siekiant užtikrinti tikslumą ir objektyvumą.

„Expert.ai Hate Speech Detector“ [19] - tai platforma, skirta aptikti ir klasifikuoti tiesioginės neapykantos kalbos atvejus privačiose žinutėse, komentaruose bei socialinės žiniasklaidos įrašuose. Šis įrankis geba išskirti specifinius įžeidžiančios ar smurtinės kalbos atvejus, priskirdamas juos įvairioms neapykantos kalbos kategorijoms. Tačiau tikslumo parametrai, detalesnė analizė ir palaikomų kalbų sąrašas viešai nėra pateikti. Palaikomos kalbos – anglų, prancūzų, italų. Lietuvių kalba nepalaikoma.

„Detoxify“ [20] - tai atvirojo kodo įrankis, sukurtas „UnitaryAI“, skirtas nustatyti toksiškus ir įžeidžiančius tekstus. Įrankis yra populiarus ir vertinamas kaip geriausias savo kategorijoje „GitHub“ platformoje. Jame naudojami pažymėtų toksiškų ir netoksiškų tekstų duomenys bei pažangūs mašininio mokymosi algoritmai, užtikrinantys labai aukštą aptikimo tikslumą. Nepaisant aukšto tikslumo, „Detoxify“ gali klaidingai pažymėti humoristinius arba savęs ironizavimo atvejus kaip toksiškus, ypač jei tekste pasitaiko su įžeidimais ar keiksmažodžiais susijusių žodžių. Deja, šis įrankis taip pat neturi oficialaus lietuvių kalbos palaikymo.

## 1.5. Neapykantos kalbos aptikimo technologijos

Pastaruoju metu socialiniuose tinkluose naudojamų neapykantos kalbos aptikimo metodų įvairovė smarkiai išaugo. Technologiniai sprendimai šioje srityje dažniausiai skirstomi į tris pagrindines grupes: tradicinius mašininio mokymosi metodus, giliojo mokymosi algoritmus bei perkėlimo mokymosi modelius. Kiekviena iš šių metodinių grupių pasižymi specifiniais technologiniais aspektais, lemiančiais jų tinkamumą ir efektyvumą konkrečiuose uždaviniuose.

### 1.5.1. Tradiciniai mašininio mokymosi metodai

Tai metodai, kurie remiasi rankiniu būdu išskirtais požymiais, kurių pagrindu yra mokomi klasifikavimo algoritmai. Tipiški šios grupės algoritmai yra logistinė regresija (angl. *logistic*

*regression*) [21], naivusis *Bayeso* klasifikatorius (angl. *naive bayes*) [22], atsitiktinių miškų algoritmas (angl. *random forests*) [23], atraminių vektorių mašina (angl. *support-vector networks*) [24], artimiausių kaimynų (angl. *nearest neighbor*) [25], sprendimų medis (angl. *decision trees*) [26], *AdaBoost* [27] bei daugiasluoksnis perceptronas (angl. *multilayer perceptron*) [28]. Dažniausiai naudojami *bigraminiai* požymiai (*n*-gramai, kai n = 2), kurie atspindi teksto fragmentų dažnius. Nepaisant aukšto našumo trumpuose ir gerai apibrėžtuose kontekstuose, tradiciniai algoritmai ribotai geba interpretuoti sudėtingesnius semantinius santykius ar ilgalaikę teksto kontekstinę informaciją [29].

### 1.5.2. Giliojo mokymosi metodai

Nuo tradicinių jie skiriasi gebėjimu automatiškai išmokti požymių reprezentaciją iš teksto. Šie algoritmai paprastai derinami su žodžių įterpinių (angl. *word embeddings*) technikomis, tokiomis kaip *Word2Vec* [30], kurios kiekvieną žodį reprezentuoja tankiu, semantiškai prasmingu vektoriumi. Vieni dažniausiai taikomų giliojo mokymosi modelių yra konvoliuciniai neuroniniai tinklai (angl. *Convolutional Neural Networks, CNN*) [31] ir pasikartojantys neuroniniai tinklai (angl. *Recurrent Neural Networks, RNN*) [32]. CNN tinklai, nors ir efektyviai aptinka specifinius lokalizuotus kalbos požymių derinius, susiduria su ribotais ilgalaikio konteksto vertinimo gebėjimais.

Šią ilgalaikės kontekstinės priklausomybės problemą efektyviai sprendžia specialūs RNN variantai, iš kurių populiariausias yra **ilgosios trumpalaikės atminties tinklas** (angl. *Long Short-Term Memory, LSTM*)[33]. Skirtingai nuo klasikinių RNN modelių, kurie patiria išnykstančio gradiento problemą, LSTM naudoja specializuotą atminties ląstelės mechanizmą, kurį sudaro trijų tipų vartai.

1. **Užmaršties vartai** (angl. *forget gate*) nusprendžia, kokia informacija bus pašalinta iš ląstelės būsenos, užkertant kelią nereikalingos informacijos kaupimui.
2. **Įvesties vartai** (angl. *input gate*) kontroliuoja naujos informacijos įtraukimą į atminties ląstelę, leidžiant tinklui išmokti svarbių ir aktualių požymių iš įeinančių duomenų.
3. **Išvesties vartai** (angl. *output gate*) nustato, kuri dalis vidinės ląstelės būsenos bus perduota kitiems tinklo sluoksniams ar išvesties sluoksniui.

Dėl šios specializuotos architektūros *LSTM* efektyviai perduoda klaidos signalus atgal per daugelį sekos žingsnių treniruojant tinklą, todėl gali išmokti prasmingas ilgalaikes priklausomybes tarp žodžių net ir labai ilguose tekstuose. Tai itin svarbu neapykantos kalbos aptikime, nes šis reiškinys dažnai priklauso nuo subtilių kalbos niuansų, tokių kaip sarkazmas, ironija ar netiesioginės užuominos.

### 1.5.3. Perkėlimo mokymosi modeliai

Tai naujausių ir pažangiausių technologijų klasė, kuri naudoja iš anksto su dideliais tekstynais apmokytus modelius, gebančius efektyviai perteikti bendras kalbos žinias naujoms, konkrečioms užduotims. Tarp jų išsiskiria **BERT** (angl. *Bidirectional Encoder Representations from Transformers*) – dvipusis *transformerių* architektūros modelis [34]. *BERT* modelis geba kiekviename savo sluoksnyje „matyti“ žodžių kontekstą iš abiejų pusių – tiek iš kairės, tiek iš dešinės pusės. Tai pasiekiama naudojant *Transformer* architektūrą su dėmesio mechanizmu (angl. *self-attention*), kuri

vienu metu analizuoja visus sekos žodžius, ir specialų mokymosi būdą: modelis iš anksto apmokomas dvipusio konteksto pagrindu, sprendžiant užduotį atspėti atsitiktinai užmaskuotus žodžius tekste. Toks išankstinis mokymas be mokytojo leidžia *BERT* įgyti bendrą kalbos supratimą. Vėliau, pritaikant (angl. *fine-tuning*) *BERT* konkrečiai užduočiai, tereikia pridėti papildomą išvesties sluoksnį – visas likęs modelio svoris jau būna „išmokęs“ kalbos dėsningumų. Šis perkėlimo mokymosi principas lėmė, kad *BERT* pasiekė naujausius pažangiausius rezultatus įvairiose užduotyse. Pavyzdžiui, *BERT* reikšmingai pagerino kelių standartinių testų (*GLUE, SQuAD* ir kt.) rodiklius lyginant su ankstesniais modeliais. Dėl gebėjimo suprasti žodžių prasmę plačiame kontekste *BERT* ypač tinka sudėtingoms kalbos analizės užduotims, kur svarbus konteksto supratimas – ne išimtis ir neapykantos kalbos aptikimas. Iš tiesų, pastaruoju metu *BERT* tapo vienu populiariausių modelių neapykantos kalbos aptikimo srityje [35]. Tyrimai rodo, kad *BERT* pagrindu sukurti sprendimai dažnai pranoksta tradicinius algoritmus: pavyzdžiui, viename konkurse anglų kalbos įžeidžiančiai kalbai atpažinti, *BERT* modelio adaptacija pasiekė **83.9%** F1 ir aplenkė daugumą kitų dalyvių[36].

## 1.6. Mašininio mokymosi metodų vertinimo metrikos

Šiame skyriuje pateikiami mašininio mokymosi metodų vertinimo metrikų apibrėžimai. Šios metrikos taip pat naudojamos šiame projekte.

### 1.6.1. Tikslumas

Tikslumas (angl. *accuracy*) [37] yra klasifikatoriaus **teisingų rezultatų dalis** tarp visų atliktų klasifikacijų. Ši metrika apskaičiuojama kaip teisingai priskirtų tiek teigiamų, tiek neigiamų atvejų skaičiaus santykis su bendru atvejų skaičiumi. Tikslumo reikšmė gali svyruoti nuo 0 iki 1 (arba 0–100 %). Kuo ji didesnė, tuo modelis tiksliau klasifikuoja duomenis (idealiu atveju tikslumas lygus 1, reiškiant, kad modelis nepadarė nė vienos klaidos). Tikslumas yra intuityvus rodiklis, tačiau jis ypač tinkamas tik tada, kai duomenų klasės yra subalansuotos – esant labai netolygiam klasių pasiskirstymui, vien tik tikslumo nepakanka modelio veikimui įvertinti.

### 1.6.2. Preciziškumas

Preciziškumas (angl. *precision*) [37] apibūdina, **kokią dalį modelio prognozuojamų teigiamų atvejų sudaro iš tiesų teisingi teigiami atvejai**. Tai yra, preciziškumas rodo modelio „tikslumą“ identifikuojant teigiamą klasę. Aukštas preciziškumas reiškia, kad tarp modelio pažymėtų teigiamų pavyzdžių yra labai mažai klaidingai teigiamų. Šios metrikos reikšmė taip pat yra intervale [0,1]. **Kuo preciziškumo reikšmė didesnė, tuo geriau** – 1 (arba 100 %) reikštų, kad *visi* modelio priskirti teigiami atvejai buvo teisingi. Preciziškumas yra svarbus, kai klaidingai teigiami atvejai turi didelę kainą (pavyzdžiui, medicininiuose testuose, kur klaidingas „teigiamas“ gali nereikalingai sukelti gydymo procedūras). Žemesnis preciziškumas indikuotų, kad modelis „per dažnai klysta“ pažymėdamas neigiamus atvejus kaip teigiamus, tad šią vertę siekiama kuo aukštesnė.

### 1.6.3. Atkūrimas

Atkūrimas (angl. *recall*) [37], dar vadinamas jautrumu (angl. *recall* arba *true positive rate)*, parodo, **kokią dalį visų teigiamų atvejų modelis sugeba teisingai atpažinti**. Kitaip tariant, tai tikimybė, kad esant teigiamam faktiniam atvejui, modelis grąžins teigiamą prognozę. Šios metrikos maksimali vertė 1 (100 %) reikštų, jog modelis aptinka visus teigiamus atvejus (nėra klaidingai neigiamų). **Kuo atkūrimo reikšmė didesnė, tuo geriau** – didelė reikšmė ypač svarbi tokiais atvejais, kai praleisti teigiamą atvejį (FN) būtų labai žalinga (pavyzdžiui, praleista teigiama diagnozė medicinoje). Pastebėtina, kad atkūrimas ir preciziškumas dažnai yra tarpusavyje susiję kompromiso principu – padidinus jautrumą, paprastai sumažėja preciziškumas, ir atvirkščiai, todėl praktikoje svarbu atsižvelgti į abu rodiklius balansuojant modelio veikimą.

### 1.6.4. F1 įvertis

F1 įvertis [37] yra harmoninis preciziškumo ir atkūrimo vidurkis, apibendrinantis šiuos du klasifikavimo kokybės aspektus vienu skaičiumi. F1 įvertis siekia subalansuoti preciziškumą ir atkūrimą: jis bus aukštas tik tuo atveju, jei abi šios metrikos yra pakankamai didelės (jei viena labai maža, F1 taip pat bus gana žemas). F1 įverčio reikšmė taip pat priklauso [0,1] intervalui, o **didesnė reikšmė reiškia geresnį modelio bendrą tikslumą**. Maksimali F1 = 1 pasiekiama tik tada, kai preciziškumas ir atkūrimas abu lygūs 1. Ši metrika ypač naudinga, kai duomenų klasės yra nevienodos.

### 1.6.5. Netekties funkcija

Netekties funkcija (angl. *validation loss*) [38] yra klaidos įvertis, rodantis, kaip stipriai modelio prognozės nukrypsta nuo tikrųjų reikšmių. Mokymosi metu optimizavimo algoritmas siekia minimizuoti šią funkciją – reguliuodamas modelio parametrus, jis mažina netekties funkcijos reikšmę. Validacijos netekties funkcija apskaičiuojama analogiškai treniravimo netekčiai, tačiau naudojant antrą (validavimo) duomenų rinkinį, kurio modelis nematė treniruodamasis. Taip įvertinamas modelio bendrinimo gebėjimas. Maža reikšmė rodo, kad modelis gerai pritaikomas nematytiems duomenims, o didėjanti – gali signalizuoti persimokymą.

Netekties funkcijos vertę siekiama minimizuoti – kuo ji mažesnė, tuo modelio prognozės tikslesnės.

## 1.7. Skirtingų neapykantos kalbos aptikimo technologijų palyginimas

Siekiant nustatyti efektyviausius metodus neapykantos kalbos aptikimo srityje, buvo atlikta detali sisteminė įvairių giliojo mokymosi modelių apžvalga, pristatyta straipsnyje „A Systematic Review of Hate Speech Automatic Detection Using Natural Language Processing“ [39]. Ši apžvalga leido identifikuoti dažniausiai mokslinėje bendruomenėje naudojamus modelių derinius bei jų paplitimą. Apžvelgiant literatūrą išryškėjo akivaizdus *LSTM* ir *BERT* architektūrų dominavimas, kurį galima paaiškinti šių modelių gebėjimu gerai interpretuoti kontekstinius kalbos niuansus.

Žemiau pateikta **1 lentelė** detalizuoja dažniausiai naudojamų modelių architektūrų pasiskirstymą tarp tyrėjų. Iš pateiktų duomenų aiškiai matyti, kad *LSTM* ir *BERT* modeliai yra dažniausiai pasirenkami tyrimuose. Tai rodo šių architektūrų pritaikomumą ir patikimumą įvairiose neapykantos kalbos

aptikimo užduotyse. Be to, išryškėja tendencija derinti įvairius žodžių įterpimo metodus (*Word2Vec, FastText, GloVe*) su specifinėmis neuroninių tinklų architektūromis siekiant pagerinti teksto požymių reprezentacijos kokybę ir tikslumą.

**1 lentelė**. Dažniausiai tyrimuose naudojamos giliojo mokymosi architektūros neapykantos kalbos aptikimui [39]

| Mašininio mokymosi modelio architektūros pavadinimas | Dažnis | Mašininio mokymosi modelio architektūros pavadinimas | Dažnis |
|---|---|---|---|
| *Word2Vec + LSTM* | 6 | *Word2Vec + CNN* | 4 |
| *RandomEmbedding + LSTM* | 4 | *RandomEmbedding + CNN* | 3 |
| *Word2Vec + BiLSTM* | 2 | *Word2Vec + CNN + LSTM* | 4 |
| *FAstText + LSTM* | 4 | *FastText + CNN* | 3 |
| *FAstText + GRU* | 4 | *FastText + GRU* | 1 |
| *FAstText + GRU* | 4 | *FastText + GRU* | 1 |
| *AraVec + LSTM* | 4 | *AraVec + CNN* | 3 |
| *AraVec +CNN+ LSTM* | 1 | *Skip-gram + CNN+LSTM* | 1 |
| *ELMO + CNN* | 1 | *BERT + CNN* | 3 |
| *ELMO + BERT* | 1 | *SKIP-GRAM + CNN* | 2 |
| *BERT Base* | 7 | *BERT Large* | 8 |
| *GloVe+CNN* | 2 | *GloVe+GBDT+CNN* | 1 |
| *CNN+CNN+CNN* | 2 | *CNN+BiGRU* | 1 |

Siekiant įvertinti modelių praktinį veiksmingumą realiose situacijose, buvo išanalizuoti rezultatai iš kelių svarbių tarptautinių konkursų: „SemEval-2019“, „SemEval-2020“ ir „Hasoc-2020“[39]. Šių konkursų rezultatai pateikti **2 lentelėje**. Lentelėje apibendrinti įvairūs socialinių tinklų tekstų klasifikavimo scenarijai skirtingomis kalbomis ir skirtingais neapykantos kalbos aspektais (pvz., rasizmas, seksizmas, religinė diskriminacija).

**2 lentelė**. Tarptautinių konkursų rezultatų analizė naudojant giliojo mokymosi modelius neapykantos kalbos aptikimui [39]

| Duomenų rinkinio turinys | Tikrintas tipas | Žodžių įterpimo metodai | Naudoti mašininio mokymosi algoritmai | F1 - įvertis |
|---|---|---|---|---|
| „Twitter“„ 16k | Neapykantos kalba | *FastText, Random embedding, GloVe* | *CNN, LSTM, GBDT* | .93 |
| - | - | | *CNN, GRU and LSTM* | - |
| „Twitter“„ 24k | Neapykantos kalba | *FastText, Word2Vec, GloVe* | *CNN, LSTM, GRU* | .69 |
| „Twitter“„ 3.8k | Neapykantos kalba | *Word2Vec* | *LSTM, BiLSTM, CNN* | .80 |
| „Twitter“„ | Neapykantos kalba | *FastText* | *LSTM, GRU, BERT* | .78 |

| | | | | |
|---|---|---|---|---|
| „Twitter“,, (arabų kalba) | Neapykantos kalba | *Word2Vec, Aravec* | *CNN+LSTM* | .71 |
| „Twitter“,, 11k (arabų kalba) | Rasizmas, seksizmas | *Keras word embeding* | *LSTM,GURU, CNN+GRU, CNN+LSTM* | .73 |
| Twitter, 9k,2k (arabų kalba) | Neapykanta, piktnaudžiavimas, mizoginija, rasizmas, religinė diskriminacija | *SG, CNN, CBOW* | *CNN, CNN-LSTM, and BiLSTM-CNN* | - |
| Twitter | Neapykanta, Įžeidimas | *ELMO* | *BERT, Multilingual-BERT* | 83 |
| „Twitter“, ‚Reddit‘, laikraščių komentarai (danų kalba) | Neapykanta, piktnaudžiavimas,rasizmas, religinė diskriminacija | - | *AUX-FastBiLSTM* | .67 |
| „Twitter“, | Seksualinė orientacija, religija, negalia, tikslinė grupė | *BOW* | *LR, biLSTM* | 80 |
| „Twitter“, | Neapykanta, Įžeidimas | - | *CNN and BiLSTM-CNN* | - |

Išanalizavus **2 lentelėje** pateiktus rezultatus, galima pastebėti, jog *CNN*, *LSTM* ir *BERT* pagrindu sukurti modeliai buvo tarp dažniausiai naudojamų ir efektyviausių konkursuose. Pavyzdžiui, anglų kalbos „Twitter“ duomenų rinkinyje (16 tūkst. įrašų), *CNN*, *LSTM* ir *Gradient Boosting Decision Tree* (GBDT) kombinacija, naudojant *FastText*, *Random embedding* ir *GloVe* požymius, pasiekė aukštą klasifikavimo tikslumą (F1 = 0,93). Kitas anglų kalbos „Twitter“ tekstų analizės pavyzdys, kuriame naudota CNN, GRU ir LSTM kombinacija, dar geriau atspindi sudėtingų modelių potencialą (P = 0,94).

Panašiai ir kitose situacijose, tokiose kaip mažesnė „Twitter“ duomenų imtis (3,8 tūkst. tekstų), taikant *Word2Vec* kartu su *LSTM* ir *BiLSTM* architektūromis, gauti aukšti rodikliai (F1 = 0,80). Įdomu pastebėti, kad daugiakalbiame kontekste F1 reikšmė taip pat išliko aukšta (0,83), parodydama *BERT* architektūros lankstumą ir gebėjimą apdoroti įvairių kalbų tekstus.

Analizuojant rezultatus arabų kalbos socialinių tinklų tekstuose, matoma, kad *CNN* ir *LSTM* kombinacija pasiekė patenkinamus rezultatus (F1 = 0,71), tačiau pastebimai žemesnius nei analogiški modeliai anglų kalbos tekstuose. Tai rodo, kad modelių našumas gali skirtis priklausomai nuo kalbos specifikos, reikalaujančios kalbai specifiškai pritaikytų žodžių įterpinių ar papildomų metodinių adaptacijų.

Apibendrinant šią praktinių rezultatų analizę, **2 lentelės** duomenys aiškiai patvirtina, kad giliojo mokymosi metodai, ypač *LSTM* ir *BERT* pagrindu sukurti modeliai, efektyviausiai susidoroja su kompleksinėmis socialinių tinklų neapykantos kalbos klasifikavimo užduotimis. Taip pat pastebėtina, kad efektyviausi modeliai naudoja ne vieną, o keletą skirtingų neuroninių architektūrų derinių, kas leidžia geriau įvertinti įvairiapusiškus kalbos aspektus, svarbius neapykantos kalbos atpažinimui.

## 1.8. Duomenų apdorojimas

Teksto pirminis apdorojimas - tai teksto duomenų paruošimas mašininio mokymosi užduotims atlikti. Jis apima keletą etapų, kuriais siekiama išvalyti, normalizuoti ir transformuoti tekstą į tokią formą, kurią galima lengvai analizuoti ir apdoroti mašininio mokymosi algoritmais [40].

Toliau bus apžvelgiami duomenų išvalymo metodai, tai yra populiariausi ir dažniausiai atliekami veiksmai, kurie naudojami ir šiame projekte.

### 1.8.1. Specialiųjų ženklų valymas ir skyrybos ženklų šalinimas

Tai teksto pirminio apdorojimo etapas, kurio metu iš teksto išvalomi ir pašalinami specialieji simboliai ir skyrybos ženklai.

**Specialieji ženklai** - tai simboliai, kurie nėra standartinės abėcėlės ar skaitmenų dalis, taip pat gali būti akcentai, diakritiniai ar kiti ženklai.
**Skyryba** - tai simboliai, naudojami teksto struktūrai ir tvarkai nurodyti, pavyzdžiui, taškai, kableliai ar kabutės.

Specialiųjų ženklų ir skyrybos ženklų valymas ir šalinimas gali būti naudingas dėl kelių priežasčių.

**Normalizavimas**. Specialiosios raidės ir skyrybos ženklai gali skirtis įvairiose kalbose ir rašmenyse, todėl tekstą gali būti sunkiau apdoroti ir analizuoti. Jų pašalinimas gali padėti normalizuoti tekstą, padaryti jį nuoseklesnį ir vienodesnį.

**Triukšmo mažinimas**. Specialiosios raidės ir skyrybos ženklai kartais gali būti įtraukti į tekstą per klaidą arba kaip nepageidaujamų laiškų ar triukšmo dalis ir gali trukdyti analizuoti tekstą. Jų pašalinimas gali padėti sumažinti triukšmą ir pagerinti teksto duomenų kokybę.

### 1.8.2. Didžiųjų ir mažųjų raidžių suvienodinimas

Dažniausiai teksto valymui naudojamos didžiosios arba mažosios raidės, nes didžiosios raidės yra įvairios sakiniui sudaryti. Taikant šį metodą visi teksto ir dokumento žodžiai projektuojami į tą pačią požymių erdvę. Tačiau jis taip pat sukeltų problemų išskirtiniais atvejais, pavyzdžiui, JAV ar Jungtinėje Karalystėje, kurias būtų galima išspręsti pakeičiant rašybos klaidas, slengą, akronimus ar neoficialias santrumpas technika.

### 1.8.3. Triukšmo šalinimas

Triukšmas - tai bet kokie duomenys, kurie nėra svarbūs ar reikšmingi užduočiai atlikti ir gali apimti klaidas, neatitikimus ar nukrypimus, kurie gali trukdyti analizuoti ar apdoroti duomenis.
Teksto duomenyse gali būti įvairių nereikalingų ženklų ar skyrybos ženklų, pavyzdžiui, URL adresų, HTML žymių, ne ASCII ženklų ar kitų specialiųjų ženklų (simbolių, „emoji“ ir kitų grafinių ženklų).

### 1.8.4. Kitos rankinio teksto valymo užduotys

1. Pakeisti rašybą, žargoną, akronimus ar neoficialias santrumpas.
2. Rašybos taisymas - taip pat gali būti laikomas neprivaloma išankstinio apdorojimo užduotimi, nes socialinės žiniasklaidos tekstiniuose duomenyse dažnai pasitaiko rašybos klaidų. Tačiau rašybos taisymo išvestis turėtų būti atidžiai dar kartą patikrinta su pradiniu įvesties tekstu, nes gali būti padaryta klaida.

### 1.8.5. Teksto skaidymo į kalbos vienetus

Tai yra įprastas metodas, kuriuo sakinys suskirstomas į žetonus, o žetonas gali būti simboliai, žodžiai, frazės, simboliai ar kiti reikšmingi elementai. Suskaidžius sakinius į mažesnius gabalėlius, tai padėtų ištirti sakinyje esančius žodžius, taip pat atlikti tolesnius teksto apdorojimo proceso etapus, pavyzdžiui, kamienavimą.

### 1.8.6. Kamienavimas

Tai žodžio šaknies išskyrimo procesas, kurio metu nustatomas bendras kamienas tarp įvairių žodžio formų (pvz., vienaskaitos ir daugiskaitos daiktavardžių formų), pavyzdžiui, žodžiai "sodininkystė", "sodininkas" arba "sodai" turi tą patį kamieną - sodas. Kamienų kirčiavimas iš žodžių iškerta priesagas, kad panašios reikšmės žodžiai būtų sujungti po standartiniu kamienu.

## 1.9. Duomenų rinkiniai

Duomenų rinkiniai yra esminiai komponentai giliojo mokymosi modelių kūrime, nes jie naudojami modelių mokymui, vertinimui ir testavimui. Tipiškai tokie rinkiniai susideda iš įvesties duomenų ir atitinkamų etikečių arba išvesties duomenų, kurie reikalingi modeliui mokytis atlikti konkrečias užduotis ar prognozuoti rezultatus.

Giliojo mokymosi algoritmai efektyviausiai mokosi iš didelių ir įvairių duomenų rinkinių. Tyrimai rodo, kad didesni duomenų rinkiniai leidžia modeliams geriau apibendrinti mokymosi metu įgytas žinias, taip pagerinant jų bendrą našumą ir gebėjimą apibendrinti [41].

Be duomenų kiekio, svarbi ir jų įvairovė. Įvairūs duomenų rinkiniai padeda modeliams išmokti atpažinti skirtingus kalbos požymius ir struktūras, todėl jie tampa atsparesni naujiems ar netikėtiems duomenims [42].

Tačiau duomenų kokybė ir įvairovė yra esminiai veiksniai, galintys stipriai paveikti modelio tikslumą ir užduoties sėkmę. Nepakankama duomenų įvairovė gali lemti modelio šališkumą ir prastą veikimą su neregėtais duomenimis.

**Lietuvių kalbai pritaikytų viešai prieinamų neapykantos kalbos aptikimo duomenų rinkinių šiuo metu nėra.** Egzistuojantys populiariausi duomenų rinkiniai, tokie kaip „Hate Speech and Offensive Language“ [43] iš Kalifornijos universiteto Berklyje arba „Toxic Comment Classification Challenge“ [44] iš Kaggle, yra anglų kalba, todėl netinka tiesioginiam lietuvių kalbos modelių mokymui.

Sukurti įvairų ir reprezentatyvų duomenų rinkinį yra sudėtinga užduotis. Vienas iš pagrindinių iššūkių yra klasės disbalanso problemos sprendimas. Neapykantos kalba yra retas reiškinys, todėl duomenų rinkiniai dažnai būna nesubalansuoti - juose daug tinkamos kalbos pavyzdžių ir mažai neapykantos kalbos pavyzdžių. Dėl to modelis gali būti šališkas. Tyrimai rodo, kad klasės disbalansas gali reikšmingai paveikti modelio veikimą, ypač mažumų klasių atpažinimą [45].

Kitas iššūkis - neapykantos kalbos įvairovė. Neapykantos kalba gali būti įvairių formų ir gali būti nukreipta į įvairias grupes. Todėl duomenų rinkiniai turi būti įvairūs ir atspindėti skirtingas neapykantos kalbos rūšis ir skirtingas grupes.

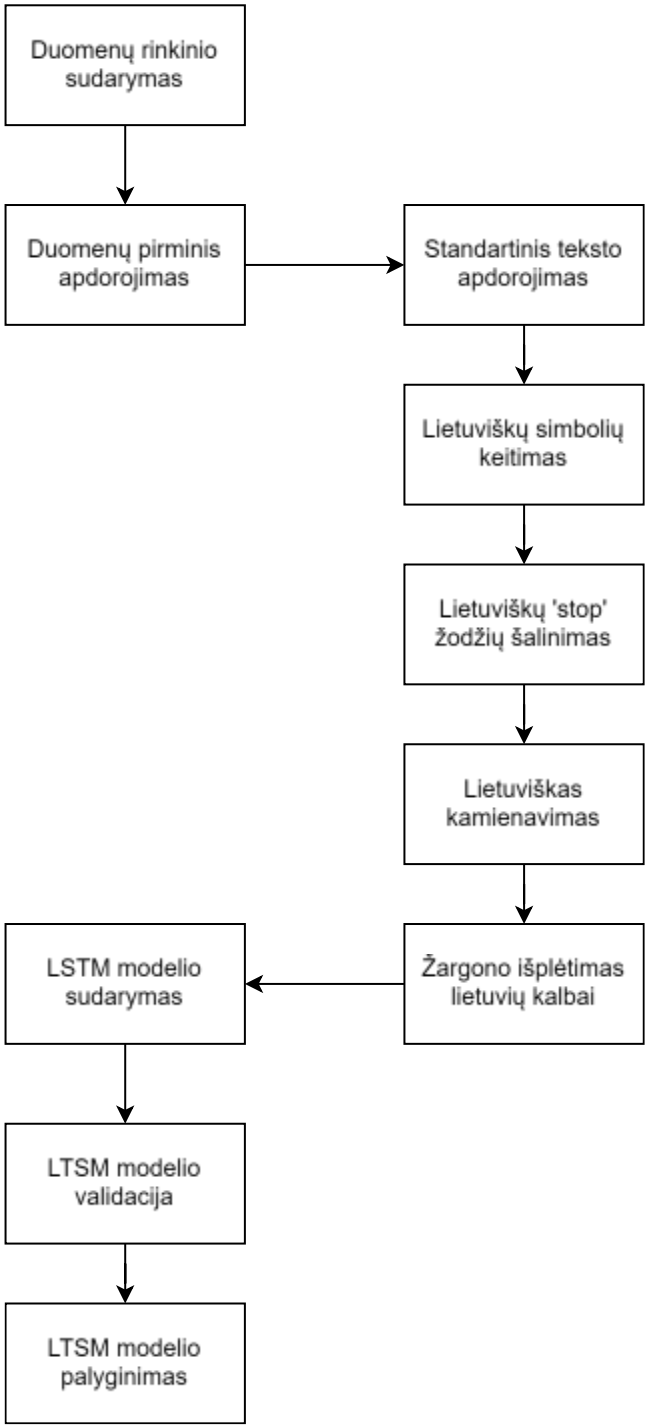
Atsižvelgiant į šiuos iššūkius, būtina sukurti lietuvių kalbai skirtą duomenų rinkinį, kuris būtų pakankamai didelis, įvairus ir reprezentatyvus, kad būtų galima efektyviai mokyti ir vertinti giliojo mokymosi modelius neapykantos kalbos aptikimo užduotyje.

## 1.10. Išvados

1. Didelė dalis interneto naudotojų susiduria su neapykantos kalba socialiniuose tinkluose, patirdami neigiamas emocines pasekmes, o platformos dėl to praranda naudotojų aktyvumą ir pajamas.
2. Nors automatinio aptikimo technologijos, pavyzdžiui, „Facebook“ sistemos, pastaraisiais metais reikšmingai patobulėjo, aptikimo tikslumas vis dar kelia iššūkių.
3. Viešai prieinamų neapykantos kalbos aptikimo įrankių yra nedaug, visi jie pritaikyti tik populiariausioms kalboms – lietuvių kalbą palaikančių komercinių įrankių nėra.
4. Literatūros analizė parodė, kad neapykantos kalbos atpažinimui taikomi įvairūs giluminio mokymosi modeliai, tačiau nė vienas jų neišsiskiria visapusišku pranašumu.
5. Neapykantos kalbos įvairovė reikalauja įvairių ir subalansuotų duomenų rinkinių.
6. Viešai prieinamų neapykantos kalbos rinkinių lietuviu kalba nėra.
7. Šiuo metu daugiausia dėmesio skiriama LSTM ir BERT pagrindu veikiančioms architektūroms, kurios demonstruoja aukštą tikslumą ir gerą kalbos subtilybių interpretavimą.
8. Pastebima, kad nėra prieinamų neapykantos kalbos aptikimo modelių ar įrankių, kurie palaikytų lietuvių kalbą. Tyrimų, kurie analizuotų lietuvišką neapykantos kalbą, taip pat yra labai nedaug.

## 2. Neapykantos kalbos aptikimo sistemos prototipo projektas

Šiame skyriuje pateiktas išsamus prototipo aprašymas. Aprašomas metodas, pradedant pirminio apdorojimo etapais, skirtais neapdorotiems duomenims patobulinti, o vėliau pateikiama LSTM modelio architektūrą. Aprašoma interaktyvi naudotojo sąsaja, sukurtą patogiai prieigai ir funkcionalumui užtikrinti. Apibendrintas modelio kūrimo procesas pateiktas **7 pav**.




**7 pav**. Neapykantos kalbos aptikimo modelio kūrimo eiga

Duomenų rinkinio sudarymas – šio etapo metu sudaromas anotacijomis papildytas tekstinių duomenų rinkinys, surinktas iš lietuviškų socialinių tinklų komentarų, siekiant užtikrinti klasifikavimui tinkamą duomenų bazę.

Duomenų pirminis apdorojimas – atliekamas pradinis neapdorotų tekstinių duomenų transformavimas į analizei tinkamą struktūrą, apimantis triukšmo mažinimą ir teksto normalizavimą.

Standartinis teksto apdorojimas – apima bendruosius natūralios kalbos apdorojimo žingsnius, tokius kaip simbolių, nuorodų ir kitų nereikšmingų elementų šalinimas, bei simbolių suvienodinimas.

Lietuviškų simbolių keitimas – atliekamas specifinių lietuviškų simbolių keitimas į ASCII atitikmenis, siekiant sumažinti žodyno sudėtingumą ir padidinti apdorojimo nuoseklumą.

Lietuviškų *stop* žodžių šalinimas – pašalinami semantiškai mažai informatyvūs dažnio požiūriu dominuojantys funkciniai žodžiai, kurie neturi reikšmingos įtakos klasifikavimui.

Lietuviškas kamienavimas – atliekamas žodžių redukavimas iki jų kamienų, siekiant sumažinti teksto įvairovę ir sustiprinti semantinį žodžių apibendrinimą.

Žargono išplėtimas lietuvių kalbai – neformalios kalbos vienetai, santrumpos ir žargonas transformuojami į standartines kalbos formas, padidinant duomenų interpretavimo aiškumą.

LSTM modelio sudarymas – sukuriamas giliojo mokymosi modelis, pagrįstas ilgosios trumpalaikės atminties neuroniniu tinklu, skirtu sekinių klasifikavimui.

LSTM modelio validacija – vertinamas sukonstruoto modelio veikimas taikant standartines klasifikacijos metrikas, siekiant nustatyti bendrą tikslumą ir gebėjimą atpažinti abi klases.

LSTM modelio palyginimas – atliktas sukurto modelio palyginimas su alternatyviais sprendimais, įvertinant našumą taikant skirtingas duomenų apdorojimo strategijas.

## 2.1. LSTM pasirinkimo pagrindimas

Planuojant neapykantos kalbos aptikimo prototipą, svarbus sprendimas buvo tinkamos neuroninės architektūros parinkimas. Literatūros analizės metu (žr. 1.5 skyrių) įvertintos įvairios klasifikavimo metodikos – nuo klasikinių algoritmų (*SVM, Naive Bayes, Random Forest*) iki šiuolaikinių giluminio mokymosi architektūrų, tokių kaip *CNN, GRU, LSTM* bei *transformer* tipo modeliai (*BERT, RoBERTa* ir kt.).

Remiantis apžvelgtų šaltinių duomenimis, LSTM (angl. *Long Short-Term Memory*) modeliai išlieka vienais iš dažniausiai taikomų sekinių klasifikavimo užduotyse, ypač natūralios kalbos apdorojimo srityje. Kaip rodo sisteminė apžvalga ir tarptautinių konkursų (*SemEval, Hasoc*) analizė, LSTM yra plačiai pritaikoma architektūra, naudojama tiek savarankiškai, tiek kartu su žodžių įterpiniais.

LSTM tinklai yra sukurti spręsti sekomis grįstas užduotis, todėl jie ypač tinkami analizuoti sakinio ar komentaro struktūrą. Dėl atminties mechanizmo jie geba išlaikyti svarbią kontekstinę informaciją iš ankstesnių žodžių, o tai yra itin aktualu neapykantos kalbos atpažinimui, kur dažnai svarbus ne atskiras žodis, o jo reikšmė visame sakinyje ar tekste.

Kitas svarbus pasirinkimo argumentas – LSTM architektūros palankumas ribotam skaičiui anotuotų duomenų. Nors pažangūs modeliai, tokie kaip BERT, pasižymi gilesne kontekstine analize, jie reikalauja didelių skaičiavimo resursų ir gausaus treniravimo. Tuo tarpu LSTM modeliai gali būti efektyviai apmokomi ir mažesniuose duomenų rinkiniuose, todėl yra tinkamas pasirinkimas prototipo vystymo stadijoje.

Atsižvelgiant į šiuos metodologinius aspektus bei esamą praktinį taikymą mokslinėje literatūroje, LSTM buvo pasirinktas kaip tinkamiausias giliojo mokymosi modelis šiam projektui.

## 2.2. Lietuvių kalbai pritaikytas pirminis apdorojimas

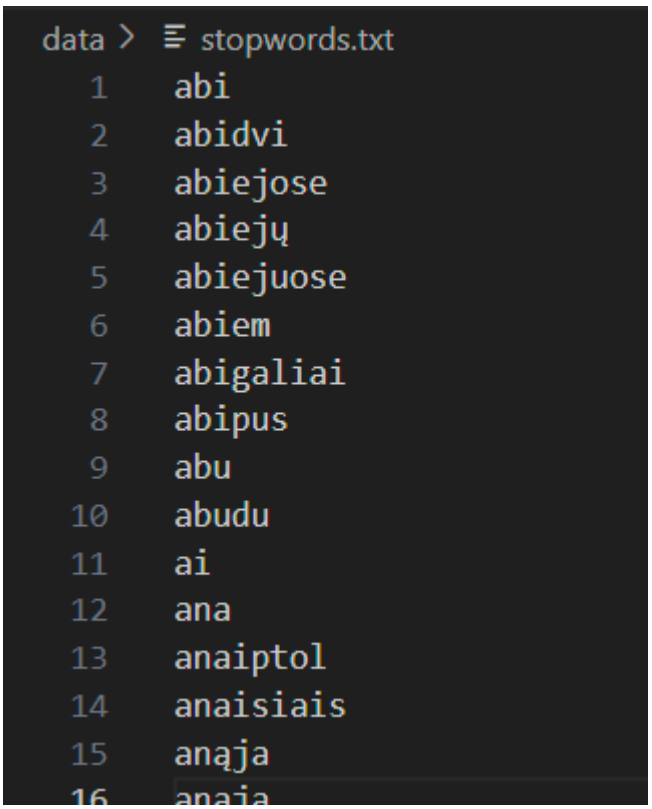
Lietuvių kalba pasižymi savita morfologine struktūra, turtingu linksnių bei galūnių sistemų rinkiniu, taip pat plačiai vartojamais specifiniais diakritiniais simboliais. Dėl šių ypatybių bendrieji natūralios kalbos apdorojimo metodai nėra tiesiogiai pritaikomi. Todėl šiame projekte taikomi teksto apdorojimo žingsniai yra specifiniai lietuvių kalbai: jie kuriami individualiai projekto autoriaus arba perimami iš lietuvių mokslininkų darbų, nes standartiniuose įrankiuose tokie sprendimai nėra prieinami. Šiame poskyryje aprašomi taikomi metodai, kurių tikslas – padidinti duomenų kokybę ir modelio tikslumą.

### 2.2.1. Lietuviškų simbolių konvertavimas

Lietuvių kalboje naudojami simboliai su diakritiniais ženklais sukelia problemų teksto analizės įrankiams, kurie orientuojasi į ASCII simbolių rinkinį. Siekiant sumažinti simbolių įvairovę ir užtikrinti didesnį suderinamumą tarp duomenų apdorojimo etapų, atliekamas simbolių konvertavimas, paremtas specialiu žemėlapiu, kuriame kiekvienam lietuviškam simboliui priskiriamas atitinkamas ASCII simbolis. Šis procesas supaprastina vėlesnius žingsnius, ypač kamienavimą.

### 2.2.2. Lietuviškų nereikšmingų žodžių šalinimas

Lietuvių kalbai būdingas didelis funkcinio žodyno dažnumas, ypač kalbant apie jungtukus, įvardžius ir kitus dažnus, tačiau semantiškai mažai reikšmingus žodžius. Šie žodžiai neturi esminės įtakos klasifikacinei vertei, todėl šalinami iš teksto, siekiant sumažinti triukšmo kiekį ir pagerinti modelio gebėjimą identifikuoti neapykantos kalbos požymius.




**8 pav**. Lietuviškų nereikšmingų žodžių failo pavyzdys

Šiame projekte taikomas lietuviškų nereikšmingų žodžių sąrašas (pavyzdys **8 pav**.), paremtas lietuvių kalbos apdorojimo tyrimais [46]. Į šį sąrašą įtraukiami žodžiai, kurių prognostinė reikšmė neapykantos kalbos atpažinimui yra minimali. Šis išteklius nėra prieinamas natūralios kalbos apdorojimo bibliotekose pagal nutylėjimą ir yra specifinis lietuvių kalbai.

### 2.2.3. Lietuviškas kamienavimo algoritmas

Lietuvių kalbos morfologija pasižymi didele gramatinių formų įvairove – tas pats žodis gali turėti įvairias linksnių, laikų, nuosakų ar skaičių formas. Siekiant sumažinti šią įvairovę ir pagerinti žodyno bendrumą, atliekamas kamienavimas – žodžių trumpinimas iki jų šaknies.

Šiame projekte taikomas algoritmas, pritaikytas specialiai lietuvių kalbai. Naudojamas lietuviškas *stemmeris* [47], ir kuris įgyvendinamas *Snowball* kalba (parodomas **9 pav**.). Algoritmas orientuotas į daiktavardžių kamienavimą, kuris laikomas svarbiausiu informacijos paieškai ir teksto analizės tikslams. Jis eliminuoja dažniausiai pasitaikančias galūnes ir grąžina žodžius į apibendrintas formas.

Šis metodas leidžia sumažinti modelio įvesties erdvės sudėtingumą, pagerina semantinį nuoseklumą ir padidina modelio mokymosi efektyvumą.

# Lithuanian stemming algorithm

## Links to resources

- Javascript demo
- The stemmer in Snowball
- Sample Lithuanian vocabulary
- Its stemmed equivalent

This algorithm was contributed by Dainius Jocas.

Its intended domain of use is information retrieval, and so handling of nouns is considered more important than that of verbs, adjectives, etc.

## The full algorithm in Snowball

```
externals ( stem )

// escape symbols for substituting lithuanian characters
stringescapes { }

/* Special characters in Unicode Latin Extended-A */
// ' nosine
stringdef ak   '{U+0105}'  // ą a + ogonek
stringdef ek   '{U+0119}'  // ę e + ogonek
stringdef ik   '{U+012F}'  // į i + ogonek
stringdef uk   '{U+0173}'  // ų u + ogonek

// . taskas
stringdef e.   '{U+0117}'  // ė e + dot

// - ilgoji
stringdef u-   '{U+016B}'  // ū u + macron

// v varnele
```

**9 pav**. Lietuviško kamienavimo algoritmo pavyzdys

1. **Pradinis žymėjimas** - pažymi žodžio pradžios ribą ir nustato pirminę žodžio struktūrą, identifikuodamas balsių ir priebalsių sekas.
2. **Konfliktų taisymas** - išsprendžia specifinius konfliktus, atsirandančius dėl darybos panašumų, ir atkuria originalius žodžių kamienus tais atvejais, kai standartinės galūnių taisyklės gali klaidingai juos sutrumpinti.
3. **Pirmasis kamienavimo žingsnis** - pašalina dažniausiai pasitaikančias lietuvių kalbos galūnes iš daiktavardžių, būdvardžių ir veiksmažodžių pagal iš anksto nustatytą taisyklių rinkinį.
4. **Raidžių taisymas** - sutvarko specifines raidžių kombinacijas, pakeisdamas sudėtingesnes priebalsių jungtis paprastesnėmis.
5. **Antrasis kamienavimo žingsnis** - papildomai pašalina įvairias lietuvių kalbos priesagas ir papildomas žodžių galūnes, taip dar labiau apibendrindamas žodžių kamienus.
6. **Papildomas raidžių taisymas** - dar kartą atlieka raidžių kombinacijų taisymą po antrojo kamienavimo etapo.
7. **Priebalsių jungčių taisymas** - ištaiso specifines priebalsių jungtis, palikdamas tik paprasčiausias jų formas.

Galutinis rezultatas - trumpesni, apibendrinti žodžių kamienai, skirti tolimesnei teksto analizei ar mašininio mokymosi užduotims.

### 2.2.4. Žargono išplėtimas

Socialinių tinklų tekstuose dažnai pasitaiko neoficiali, supaprastinta ar trumpinta kalba, kuri formuojama naudojant žargoną, santrumpas ir netaisyklingas kalbos struktūras. Dėl šios priežasties tradiciniai teksto analizės metodai nėra pajėgūs tiksliai interpretuoti tokių vienetų reikšmės, o jų prasmė tampa prarandama modeliui, kuris negeba jų susieti su semantiškai reikšmingu turiniu.

Šiame projekte taikomas individualiai autoriaus sudarytas žargono žodynas, atspindintis dažniausiai vartojamas trumpintas frazes anglų ir lietuvių kalbomis, kurios transformuojamos į išplėstines kalbines formas. Šis išteklius nėra randamas standartiniuose natūralios kalbos apdorojimo bibliotekose ir yra sukurtas remiantis praktine socialinių tinklų kalbos analize. Juo užtikrinamas vieningos kalbos struktūros atkūrimas ir sumažinamas reikšmių praradimo pavojus. Failo dalis pateikta **10 pav**.

data > {} slang.json > ...

```
{
    "pls": "prašau",
    "thx": "ačiū",
    "brb": "grįšiu greitai",
    "imo": "mano nuomone",
    "btw": "beje",
    "omg": "o mano dieve",
    "fml": "mano gyvenimas yra prakeiktas",
    "lol": "juokinga",
    "idk": "nežinau",
    "jk": "juokauju",
    "bff": "geriausias draugas amžiams",
    "np": "nėra už ką",
    "tgif": "dėkui dievui, penktadienis",
    "asap": "kaip įmanoma greičiau",
```

**10 pav**. Žargono išplėtimo failo pavyzdys

Sukurto žodyno struktūra JSON formatu, kuriame pateikiamos žargono santrumpos kaip raktai ir jų atitikmenys pilnais sakiniais - kaip reikšmės. Galutiniame faile apibrėžtos 45 žargoninės frazės, dažniausiai sutinkamos lietuviškuose ir tarptautiniuose socialinių tinklų komentaruose.

## 2.3. Neapykantos kalbos aptikimo naršyklės plėtinio projektas

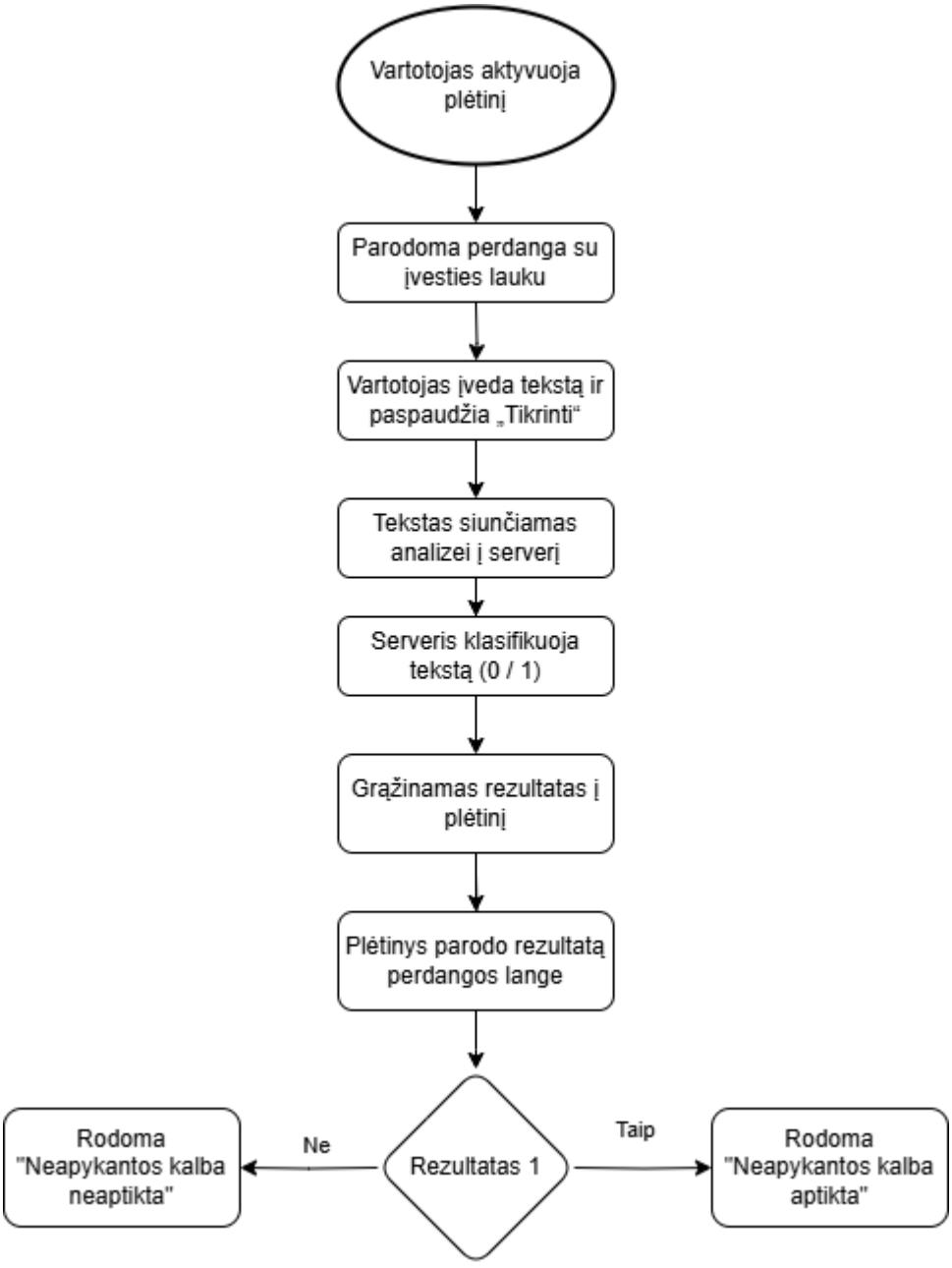
Planuojama sukurti naršyklės plėtinį, skirtą vartotojui patogiu būdu aptikti neapykantos kalbos požymius internete. Tikslas – užtikrinti, kad naudotojai galėtų savarankiškai analizuoti tekstą bet

kuriame tinklalapyje, be būtinybės naudotis išorinėmis sistemomis. Funkcionalumas grindžiamas interaktyvia vartotojo sąsaja bei automatiniu teksto klasifikavimu naudojant serverinę API ir neuroninį tinklą.

Plėtinys turi būti aktyvuojamas per naršyklės įrankių juostos mygtuką. Aktyvavus, tinklalapyje atsidaro perdangos langas su tekstiniu įvedimo lauku ir analizės mygtuku. Įrankis leidžia vartotojui įklijuoti, įvesti arba pažymėti bet kokį tekstą, kuris vėliau siunčiamas į serverio pusę analizės tikslais.

### 2.3.1. Neapykantos kalbos aptikimo sistemos kūrimo procesas

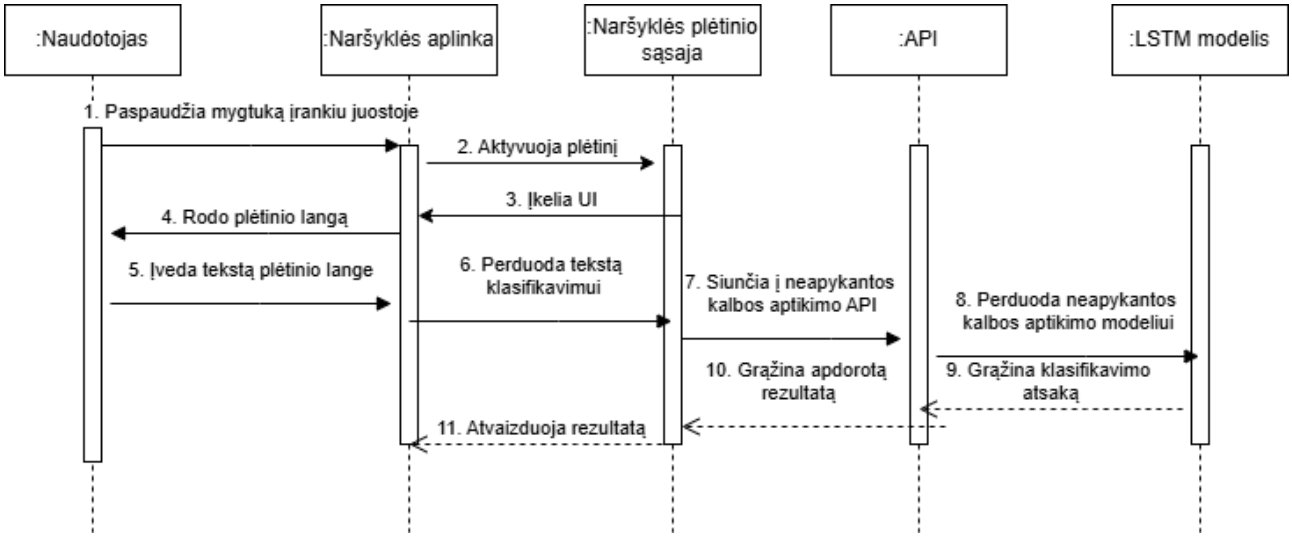
Planuojamos sistemos funkcionalumo logika papildomai perteikiama **11-tame paveiksle**, kuriame pateikta supaprastinta proceso diagrama. Ji parodo pagrindinius naudotojo ir sistemos sąveikos žingsnius - nuo plėtinio aktyvavimo iki klasifikacijos rezultato pateikimo.




**11 pav**. Neapykantos kalbos aptikimo naršyklės plėtinio proceso diagrama

Ši schema išskiria tik esminius proceso etapus, leidžiančius suprasti bendrąją veikimo eigą: naudotojas įveda tekstą, inicijuoja analizę, o serveris, atlikęs klasifikaciją, grąžina rezultatą į naršyklės perdangos langą. Rezultatas interpretuojamas dvejetainiu principu – jei aptinkama neapykantos kalba, vartotojui tai aiškiai nurodoma.

**12-tame paveiksle** pavaizduota planuojamo plėtinio veikimo sekos diagrama, atskleidžianti komponentų tarpusavio sąveiką, kai vykdoma neapykantos kalbos analizė. Schema atvaizduoja žingsnių eigą nuo naudotojo veiksmo naršyklėje iki atsakymo gavimo iš klasifikavimo modelio.

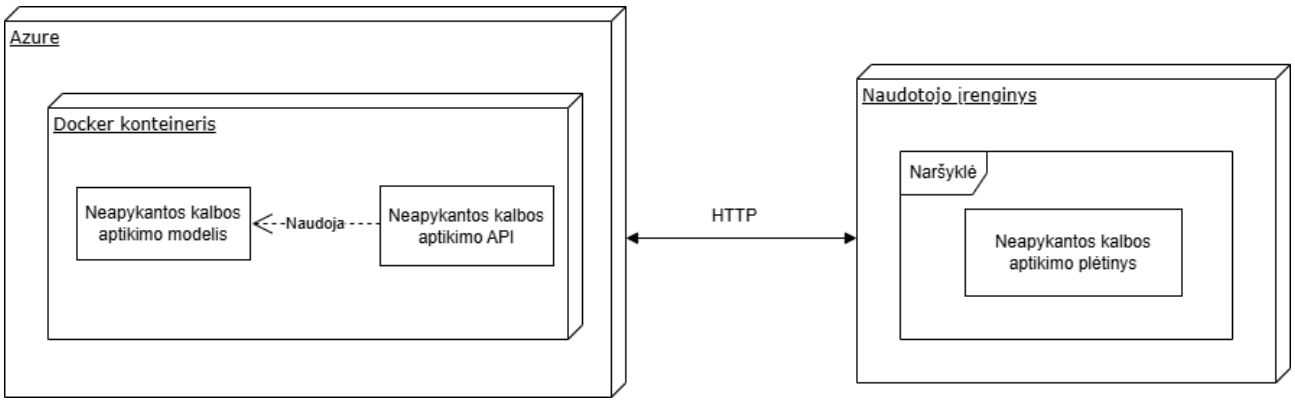



**12 pav**. Neapykantos kalbos aptikimo naršyklės plėtinio sekų diagrama

Sekos diagramoje išskiriami penki aktoriai.
**Naudotojas** – sąveikauja su naršykle ir plėtiniu, inicijuoja veiksmus ir gauna klasifikavimo rezultatą.
**Naršyklės aplinka** – reaguoja į naudotojo veiksmus, aktyvuoja plėtinį ir leidžia įkelti reikiamus komponentus.
**Naršyklės plėtinio sąsaja** – apima *content.js*, kuris realizuoja perdangos sąsają, apdoroja įvestį ir komunikuoja su API.
**API** – priima užklausas iš plėtinio, atlieka teksto apdorojimą ir valdo klasifikavimo proceso eigą.
**LSTM modelis** – gauna apdorotą tekstą ir grąžina klasifikavimo rezultatą (0 – neapykantos kalba neaptikta, 1 – aptikta).

### 2.3.2. Neapykantos kalbos aptikimo sistemos diegimas

Šio projekto įgyvendinimo metu sukurta neapykantos kalbos aptikimo sistema diegiama naudojant „Docker“ konteinerių technologiją, debesų kompiuterijos platformoje „Azure“. Sistemos architektūrinis sprendimas vaizduojamas **13 pav**.




**13 pav**. Neapykantos kalbos aptikimo sistemos diegimo diagrama

„Azure“ platformoje paleistas „Docker“ konteineris, kuriame integruotas neapykantos kalbos aptikimo modelis ir jį naudojanti API. API komponentas realizuoja komunikacijos sąsają su naudotojo įrenginiu, naudodamas HTTP protokolą. Sistemos veikimo principas paremtas tuo, kad naršyklėje veikiantis neapykantos kalbos aptikimo plėtinys siunčia naudotojo pateiktą tekstą į „Azure“ debesijoje veikiančią API. Gavęs užklausą, API kreipiasi į „Docker“ konteineryje esantį modelį, kuris analizuoja pateiktą tekstą ir grąžina klasifikavimo rezultatą (aptikta ar neaptikta neapykantos kalba).

Pasirinktas diegimo būdas užtikrina didelį našumą ir patikimumą bei leidžia lengvai atnaujinti ar plėsti sistemą ateityje. Debesijos paslaugos taip pat leidžia užtikrinti aukštą sistemos prieinamumo lygį ir optimalų resursų naudojimą.

# 3. Neapykantos kalbos aptikimo prototipo realizacija

Šiame skyriuje pateikiama praktinė sukurto prototipo realizacijos dalis. Aprašomi visi etapai nuo duomenų rinkinio sudarymo ir jo anotavimo procedūrų iki teksto apdorojimo, mašininio mokymosi modelio konstravimo, klasifikavimo API kūrimo bei naršyklės plėtinio įgyvendinimo.

## 3.1. Projekto aplinka

Neapykantos kalbos aptikimo mašininio mokymosi modeliui bei su juo susietam naršyklės plėtiniui sukurti pasitelkiama šiuolaikinė programinė įranga bei atvirojo kodo bibliotekos, užtikrinančios sklandų duomenų analizės, apdorojimo ir naudotojo sąsajos veikimą.

**Mašininio mokymosi dalis**

- **Python 3.10** [48] – universali, plačiai naudojama programavimo kalba, palaikanti daugelį mokslinių ir dirbtinio intelekto bibliotekų;
- **Pandas 2.0.3** [49] – duomenų analizės biblioteka, skirta darbui su struktūriniais duomenimis;
- **NumPy 1.26.4** [50] – matematinė biblioteka, leidžianti efektyviai dirbti su daugiamačiais masyvais ir vykdyti skaitinius skaičiavimus;
- **Scikit-learn 1.2.2** [51] – mašininio mokymosi algoritmams įgyvendinti ir modeliui vertinti;
- **TensorFlow 2.15.0** [52] – giliojo mokymosi karkasas neuroninių tinklų kūrimui ir treniravimui (įskaitant LSTM modelį);
- **Flask 2.2.5** [53] – lengvas **Python** karkasas, naudojamas API kūrimui ir integracijai su plėtiniu.

**Naršyklės plėtinio dalis**

- **JavaScript ES6** [54] – naudojamas sąsajai realizuoti, apdoroti įvesties duomenims bei siųsti užklausas į API;
- **HTML5 + CSS3** [55] – vartotojo sąsajos struktūrai ir stiliui apibrėžti;
- **Chrome Manifest V3** [56] – naujausia „Google Chrome“ plėtinių kūrimo specifikacija, apibrėžianti saugumo ir funkcionalumo reikalavimus;
- **Chrome Extensions API** [57] – naudojama sąsajai su naršyklės funkcijomis (aktyviu langu, *skriptų* įkėlimu, įrankių juostos elementais).

### 3.2. Duomenų rinkinio kūrimas

Duomenų rinkinys šiam projektui buvo sudarytas naudojant socialinės platformos ***Reddit*** komentarus. Duomenų šaltiniu pasirinktas *subreddit'as* **/r/lietuva**, kur aktyviai vyksta lietuviškai kalbančių vartotojų diskusijos įvairiomis temomis.

Komentarams atrinkti buvo pritaikyti šie kriterijai:

1. Naudota *subreddit'o /r/lietuva* komentarų duomenų bazė.
2. Komentarai išrinkti iš diskusijų, surikiuotų pagal *Reddit* platformos „controversial“ rodiklį, per visą *subreddit'o* egzistavimo laikotarpį. Šis rūšiavimo būdas išryškina diskusijas, kurios sukėlė didžiausią nesutarimą tarp vartotojų – gavusios panašų kiekį teigiamų ir neigiamų balsų, taip pat aktyvią diskusiją komentaruose.
3. Komentarų kiekis – 5000.

Duomenų surinkimui sukurtas *Python* algoritmas, kuris vykdė Reddit API užklausas ir surinko reikiamus komentarus.

Neapykantos kalbos anotavimas atliktas rankiniu būdu.

#### 3.2.1. Duomenų rinkinio turinio analizė


**14 pav**. Neapykantos žodžių debesis

**14 pav**. pateiktame žodžių debesyje vizualizuojamas dažniausiai duomenų rinkinyje pasitaikančių žodžių pasiskirstymas. Matomi žodžiai, tokie kaip „vyrai“, „visuomenė“, „moterys“ rodo, jog šie terminai buvo dažniausiai vartojami analizuotame socialinių tinklų komentarų rinkinyje. Žodžių dažnis leidžia identifikuoti pagrindines temas, kurios aptariamos kontekste, kur gali būti vartojama neapykantos kalba. Pastebima, kad dažnai minima „Lietuva“, „Europa“, „lietuvių“, taip pat tokie žodžiai kaip „naciai“, „žmonės“, „vaikai“ rodo, jog dažnos diskusijų temos yra susijusios su tautine, socialine ar ideologine skirtimi. Taip pat pastebimas dažnas keiksmažodžių naudojimas.

### 3.2.2. Duomenų rinkinio struktūra

Žemiau **3 lentelėje** pateikiama surinkto komentarų rinkinio struktūra. Kiekvienas stulpelis apibrėžia skirtingą informaciją, reikalingą tolimesniam teksto apdorojimui ir klasifikavimui.

**3 lentelė**. Duomenų rinkinio struktūra

| | Stulpelis | Aprašymas |
|---|---|---|
| 1 | *comment_id* | Unikalus komentaro identifikatorius. |
| 2 | *submission_id* | Unikalus įrašo identifikatorius, kuriame yra komentaras. |
| 3 | *author* | Komentaro autoriaus slapyvardis „*Reddit“* platformoje. |
| 4 | *body* | Komentaro tekstas. |
| 5 | *created_utc* | Komentaro sukūrimo laikas UNIX laiko formatu. |
| 6 | *created_dt* | Komentaro sukūrimo laikas suprantama data ir laiku (YYYY-MM-DD HH:MM:SS). |
| 7 | *score* | Komentaro įvertinimas (balsų skaičius, kurį gavo komentaras). |
| 8 | *permalink* | Nuoroda į komentarą Reddit platformoje. |
| 9 | *is_hate_speech* | Žyma, nurodanti, ar komentaras laikomas neapykantos kalba (1 - taip, 0 - ne). |

### 3.2.3. Anotavimo procedūros

Prieš rašant anotaciją, perskaitoma kiekvieną komentarą ar pranešimą iki galo.

Žymima komentarą ar pranešimą kaip neapykantos kalbą, net jei neapykantos kalba yra tik dalyje pranešimo.

Tais atvejais, kai norint nustatyti, ar komentaras ar pranešimas yra neapykantos kalba, ar ne, reikia atsižvelgti į kontekstą.

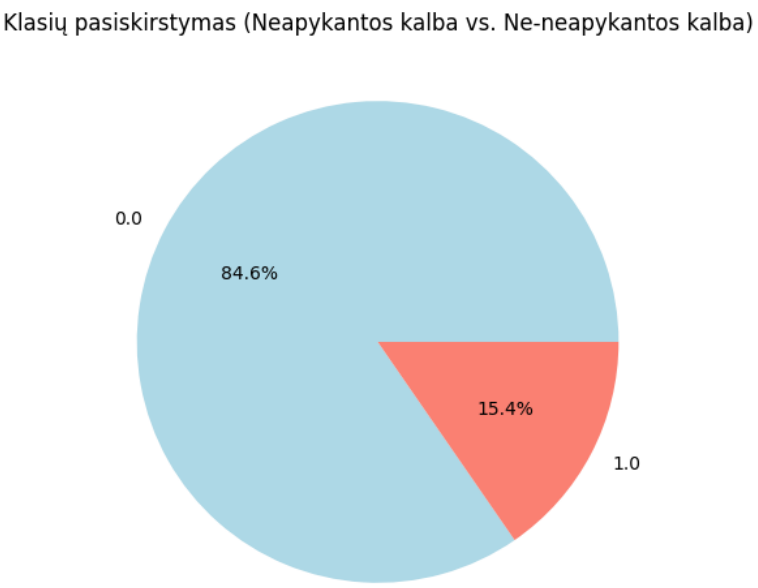
Atsižvelgiant į tai, kad socialinės žiniasklaidos pranešimuose dažnai vartojamas slengas, santrumpos ar kūrybiška rašyba, sunku priimti teisingą sprendimą, todėl bus pasiremta asmeninėmis kalbos žiniomis.

### 3.2.4. Duomenų rinkinio anotavimas

Duomenys buvo anotuojami rankiniu būdų vienu parametru.

**Ne neapykantos kalba (0)** - Komentarai arba pranešimai, kuriuose nėra jokios neapykantos kalbos.

**Neapykantos kalba (1)**- Komentarai arba pranešimai, kuriuose yra bet kokios formos neapykantos kalbos, kaip apibrėžta pirmiau.




**15 pav**. Klasių pasiskirstymas

**15 pav**. pateikiamas duomenų rinkinio klasių pasiskirstymas, vaizduojantis, kiek procentų komentarų buvo priskirti neapykantos kalbai ir kiek neutraliai kalbai. Akivaizdu, kad duomenų rinkinys nėra lygiai subalansuotas, nes dauguma komentarų nepriskirti neapykantos kalbai. Tačiau atsižvelgiant į duomenų rinkinio turinio pobūdį, neapykantos kalbos įrašų yra pakankamai daug.

### 3.2.5. Duomenų rinkinio skaidymas

**Mokomasis rinkinys:** Didelė duomenų dalis (70 %), naudojama mašininio mokymosi modeliams mokyti.

**Testavimo rinkinys:** Likusi duomenų dalis (30 %), naudojama galutiniam modelio veikimui įvertinti, siekiant nešališkai įvertinti jo gebėjimą apibendrinti nepastebėtus duomenis.

### 3.3. Duomenų apdorojimas

Šiame skyriuje aprašomi visi atlikti duomenų apdorojimo veiksmai. Lentelėse pateikiami kiekvieno žingsnio pavyzdžiai, naudojant realius duomenis iš sudaryto duomenų rinkinio.

#### 3.3.1. Simbolių keitimas

Išvalomi lietuviški simboliai iš teksto (pagal aprašymą 2.2.1). Šio apdorojimo pavyzdys pateikiamas **4 lentelėje**.

**4 lentelė**. Pirminio apdorojimo pavyzdys

| Prieš apdorojimą | Po apdorojimo |
|---|---|
| viską ką išvardinote tinka ir žmonių medžioklei. ar turėtume gyventi pagal prigimtį ir medžioti žmones? ar turėtume nenaudoti dirbtinės šviesos? nenešioti drabužių ir t.t.? | viska ka isvardinote tinka ir zmoniu medzioklei. ar turetume gyventi pagal prigimti ir medzioti zmones? ar turetume nenaudoti dirbtines sviesos? nenesioti drabuziu ir t.t.? |
| tai jūs manote, kad tėvai turi teisę suvalgyti savo vaikus? | tai jus manote, kad tevai turi teise suvalgyti savo vaikus? |

Matoma, kad lietuviški simboliai buvo sėkmingai pakeisti standartiniais atitikmenimis.

#### 3.3.2. Lietuviškų stopžodžių sąrašas

Stop-žodžių šalinimas yra vienas iš bazinių žingsnių teksto analizės metu, siekiant pašalinti perteklinę informaciją, kuri neturi reikšmingos įtakos klasifikacijai. Projektui naudotas specialiai lietuvių kalbai parengtas stop-žodžių sąrašas, išsamiai aprašytas 2.1.2 poskyryje.

**5 lentelė**. Stop-žodžių šalinimo pavyzdys

| Prieš apdorojimą | Po apdorojimo |
|---|---|
| vegetarizmas yra tiesiog dieta o veganizmas yra principas pries gyvunu isnaudojima kadangi sveiki gyvunai nori gyventi ir nenori buti skriaudziami tai logiska gerbti tokius kitu norus kaip ir mes nenoretume kad nebutu gerbiami musu norai i gyvybe ar gyvenima be isnaudojimo pieno ir kiausiniu pramonese daznai bereikalingai mirsta dalis gyvunu dalis yra specialiai nuzudoma o like beveik visi iki vieno parduodami mesos industrijai | vegetarizmas yra dieta veganizmas yra principas pries gyvunu isnaudojima sveiki gyvunai nori gyventi nenori buti skriaudziami logiska gerbti tokius kitu norus nenoretume nebutu gerbiami musu norai i gyvybe gyvenima isnaudojimo pieno kiausiniu pramonese daznai bereikalingai mirsta dalis gyvunu dalis yra specialiai nuzudoma like visi vieno parduodami mesos industrijai |
| o tai kam ta retorika naudoji atsakyk uz bazara | kam retorika naudoji atsakyk uz bazara |

5 lentelėje pateikiamas pavyzdys, kaip pasikeičia sakinio struktūra prieš ir po stop-žodžių šalinimo – aiškiai matomas semantinių žodžių išryškėjimas bei informacinio triukšmo sumažėjimas.

### 3.3.3. Kamienavimas

Šis etapas leidžia žodžius sumažinti iki jų kamienų, taip supaprastinant analizės procesą ir sumažinant žodyno įvairovę. Šio metodo taikymas išsamiai aptariamas 2.1.3 poskyryje, kuriame nurodoma, jog lietuvių kalbai naudotas specialiai pritaikytas algoritmas.

**6 lentelė**. Kamienavimo pavyzdys

| Prieš apdorojimą | Po apdorojimo |
|---|---|
| baltaodziui hetero europieciui vyrui nereiketu stresuoti liberalams ateina galas | baltaodz heter europiec vyr nereiket stres liberal atein gal |
| zodzio laisve istatymu yra aiskiai apibrezta nera absoliuti neapykantos kalba skatinimas yra nelegalu cia as turejau omenyje | zodz laisv istatym yr aisk apibrezt ner absoliu neapyk kalb skatinim yr nelegal cia as turej om |

**6 lentelėje** pateikiamas kamienavimo poveikio pavyzdys. Joje matyti, kaip žodžių formos redukuojamos iki bazinių vienetų, išlaikant semantinę prasmę ir užtikrinant efektyvesnį teksto reprezentavimą modelyje.

### 3.3.4. Žargono ir santrumpų išplėtimas

Žargono transformavimas į standartines kalbines formas yra svarbus žingsnis apdorojant socialinių tinklų tekstus, kuriuose gausu santrumpų, slengo ir neformalaus žodyno. Kaip aprašyta 2.1.4 poskyryje, šiame projekte naudojamas individualiai sudarytas žargono žodynas, leidžiantis automatiškai išplėsti dažniausiai vartojamas santrumpas į jų reikšminius atitikmenis.

**7 lentelė**. Žargono taisymo pavyzdys

| Prieš apdorojimą | Po apdorojimo |
|---|---|
| senam žmogui nukvakimą atleidi ir nepyksti, nes mums tai irgi pareina.<br><br>bet kaip tą požiūrio nukvakimą toleruot idk | senam zmogui nukvakima atleidi ir nepyksti, nes mums tai irgi pareina. bet kaip ta poziurio nukvakima toleruot nežinau |
| kuo labiau pulsim juos, tuo jie labiau užknis irl. | kuo labiau pulsim juos, tuo jie labiau užknis realybėje. |

**7 lentelėje** pateikiami transformacijos pavyzdžiai atspindi, kaip žargonas („idk“, „irl“) pakeičiamas į prasmines frazes, o tai padeda užtikrinti semantinį aiškumą ir pagerina modelio klasifikavimo galimybes.

### 3.3.5. Bendrasis teksto pirminis apdorojimas

**Mažųjų raidžių rašymas**.

Šiame etape visas tekstas paverčiamas mažosiomis raidėmis, kad algoritmas tų pačių žodžių su skirtingomis didžiosiomis raidėmis nelaikytų unikaliais

### 3.3.6. Specialiųjų ženklų pašalinimas.

Naudojant reguliarias išraiškas, iš pranešimų buvo pašalinti įvairūs ne raidiniai / skaitiniai simboliai, įskaitant naudotojų paminėjimus, *hipersaitus* ir tipografinius simbolius, kurie nėra svarbūs neapykantos kalbos turinio analizei.

**8 lentelė**. Teksto išvalymo pavyzdys

| Prieš apdorojimą | Po apdorojimo |
|---|---|
| Kerta per smegenis veganizmas stipriai. 😅 | kerta per smegenis veganizmas stipriai |
| taip, laikas: [https://youtu.be/ib-7rpt5dk8](https://youtu.be/ib-7rpt5dk8) | taip laikas |

**8 lentelėje** pateikti pavyzdžiai rodo, kaip bendrieji valymo žingsniai sumažina nereikšmingų simbolių triukšmą tekste, išlaikant esminę semantinę informaciją. Toks pirminis apdorojimas yra būtina sąlyga siekiant užtikrinti tolygią tolimesnių analizės etapų eigą.

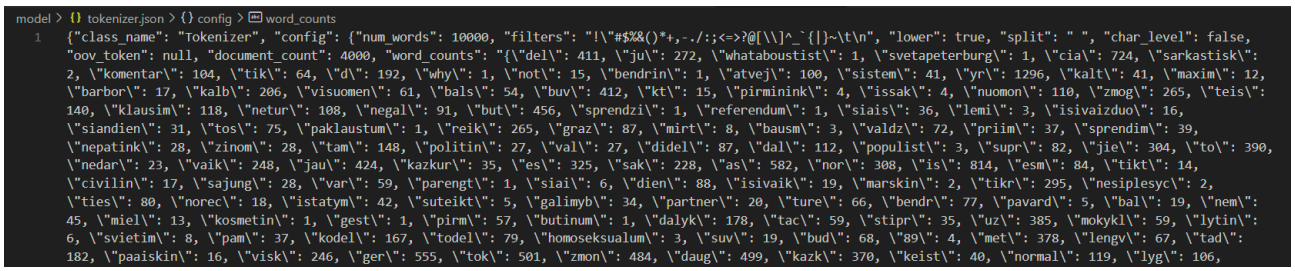
### 3.3.7. Teksto skaidymas į kalbos vienetus

Teksto skaidymas į atskirus elementus, arba kalbos vienetus (angl. *tokens*), leidžia analizuoti tekstą smulkesniu lygmeniu ir efektyviau jį paversti tinkama įvestimi neuroniniam tinklui.

**9 lentelė**. Teksto skaidymo į kalbos vienetus pavyzdys

| Prieš apdorojimą | Po apdorojimo |
|---|---|
| galimyb | 454 |
| amerik | 21 |

**9 lentelėje** pateiktas pavyzdys iliustruoja, kaip žodžiai transformuojami į skaitines reikšmes, atitinkančias jų pozicijas mokymo rinkinio žodyne. Kiekvienas žodis priskiriamas atitinkamam *indeksui*, kuris vėliau naudojamas modelio įėjime.




**16 pav**. Teksto skaidymo į kalbos vienetus failo pavyzdys

Šiam tikslui naudojama „Keras“ programos *Tokenizer* funkcija. Ši funkcija sukuria žodžių atvaizdavimą į sveikuosius skaičius ir užtikrina, kad būtų atsižvelgiama tik į dažniausius mokymo rinkinio žodžius, o tai padeda sumažinti modelio sudėtingumą ir išteklių poreikį. Rezultatas pateikiamas **16 pav**.

### 3.4. Mašininio mokymosi modelio kūrimas

Analizės dalyje nustatyta, kad teksto apdorojimo užduočiai atlikti itin tinkami yra pakartotiniai neuroniniai tinklai (RNN), tiksliau, jų ilgosios trumpalaikės atminties (angl. *Long Short-Term Memory*, LSTM) variantas, dėl jiems būdingų sekos apdorojimo galimybių. LSTM tinklai geba suprasti ilgesnes sekas.

#### 3.4.1. LSTM modelio parametrų aprašymas

**MAX_NB_WORDS** – tai parametras, nustatantis didžiausią žodyno dydį, t. y. kiek daugiausiai skirtingų žodžių bus naudojama teksto duomenims reprezentuoti. Žodyno dydžio keitimas lemia informacijos ir triukšmo balansą. Per mažas žodynas gali lemti, kad dalis svarbios informacijos bus prarasta – reti, bet reikšmingi žodžiai nebus įtraukti į modelį, todėl gali mažėti tikslumas dėl neišmokytos informacijos. Tuo tarpu per didelis žodynas gali apimti labai retus žodžius, kurie duomenų aibėje pasitaiko vos kelis kartus. Į modelį įtraukus tokius retus žodžius, didėja persimokymo rizika – modelis gali išmokti tuos retus žodžius (pritaikyti svorius jiems), tačiau tai negerins apibendrinimo gebėjimo naujiems duomenims

**EMBEDDING_DIM** - žodžių vektorinės reprezentacijos (angl. *embedding*) dimensija. Pavyzdžiui, EMBEDDING_DIM = 100 reiškia, kad kiekvienas žodis bus pavaizduotas 100-ilypiu vektoriumi. reikšmės keitimas veikia modelio išraiškingumą ir persimokymo riziką. Maža dimensija (pvz., 50) gali lemti, kad modelis nepajėgs atskirti tam tikrų subtilių žodžių reikšmių – tokioje mažoje erdvėje skirtingos sąvokos gali susilieti, prarasdamos reikšmingus skirtumus. Dėl to per maža *EMBEDDING_DIM* gali sukelti nepakankamą išmokimą – modelis neturės galimybės įgyti visos reikalingos informacijos apie žodžių santykius. Kita vertus, labai didelė dimensija (pvz., 300 ar daugiau) ne visuomet duoda naudos. Nors teoriškai ji gali užkoduoti daugiau niuansų, praktikoje per didelė dimensija gali į modelį įnešti triukšmo arba perteklinių, pasikartojančių požymių, ypač jei mokymosi duomenų kiekis ribotas.

**MAX_SEQUENCE_LENGTH** – maksimalus sekos (žodžių eilutės) ilgis. Šis parametras nurodo, iki kelių žodžių yra apribojama kiekviena teksto seka. Jei tekstas ilgesnis nei nustatyta riba, pertekliniai žodžiai nukerpami (dažniausiai nuo galo), o jei trumpesnis – seka papildoma specialiais ženklais (pvz., <PAD>) iki nustatyto ilgio. Jei MAX_SEQUENCE_LENGTH per mažas, modelis gali gauti nepakankamai informacijos: dalis teksto (pvz., sakinio ar dokumento pabaiga) bus nukirpta ir modelis neteks galbūt svarbių požymių, dėl ko kris klasifikavimo ar prognozavimo tikslumas. Per didelė sekos riba irgi nėra optimali – į modelį pateks visos teksto dalys, įskaitant galbūt nebūtinai reikšmingą ar pasikartojančią informaciją.

**LSTM_UNITS** - LSTM sluoksnio paslėptųjų vienetų (atminties ląstelių) skaičius. Tai nusako LSTM paslėptos būsenos vektoriaus dimensiją. Pavyzdžiui, LSTM_UNITS = 128 reiškia, kad LSTM sluoksnis turi 128 neuronų vienetus, kiekviename iš jų kaupiama informacija per laiką. Pasirinkimas reikalauja balanso tarp modelio galios ir persimokymo rizikos. Mažas vienetų skaičius (pvz., 16 ar 32) gali būti nepakankamas sudėtingiems raštams išmokti – modelis gali turėti per mažai „atminties“ ir parametrų, kad aprėptų visus duomenų dėsningumus, dėl to didės mokymo ir validacijos paklaidų

skirtumas dėl nepakankamo išmokimo. Didelis vienetų skaičius (pvz., 128 ar 256) suteikia modeliui daugiau laisvės pritaikyti duomenis, tačiau kartu labai padidina persimokymo tikimybę, ypač jei mokymo duomenų nedaug.

**DROPOUT_RATE** - tai atsitiktinio neuronų atjungimo (angl. *dropout*) norma, taikoma neuroninio tinklo sluoksnyje. LSTM kontekste šis parametras paprastai reiškia, kokia dalis įėjimo jungčių į LSTM neuronus bus atsitiktinai nulinės per kiekvieną mokymo žingsnį. Pavyzdžiui, *DROPOUT_RATE* = 0,5 reiškia, kad per mokymą kiekviename žingsnyje atsitiktinai išjungiama 50% LSTM įėjimo vienetų (skirtingi kiekvieno *batch* metu). Taip neuronai „nepasiekia“ dalies įvesties duomenų, tarsi tie duomenys nebūtų buvę perduoti. Atsitiktinai išjungiant dalį neuronų, modelis mokomas labiau apibendrinti požymius, o ne pasikliauti konkrečių neuronų išmoktais ryšiais. Teoriškai tai priverčia tinklą kurti atsparius požymių derinius – tarsi mokytume daugybę mažesnių tinklų. DROPOUT_RATE didinimas paprastai mažina persimokymą

**RECURRENT_DROPOUT** - tai *dropout* norma, taikoma pasikartojantiems LSTM ryšiams, t. y. tarp LSTM ląstelių skirtinguose laiko žingsniuose. Skirtingai nuo DROPOUT_RATE, kuri veikia įėjimo signalus į LSTM, RECURRENT_DROPOUT lemia, kokia dalis LSTM vidinės būsenos (paslėptų neuronų išvesties) yra atsitiktinai nunulinama kaskart perduodant būseną kitam laiko žingsniui. *RECURRENT_DROPOUT* tiesiogiai skirtas persimokymui mažinti sekos modeliuose, tačiau jo poveikis kiek kitoks nei paprasto *dropout*. Kadangi jis atjungia atsitiktinę dalį laikinų ryšių, tai **trikdo LSTM gebėjimą ilgai išlaikyti informaciją**. Tyrimai parodė, kad per didelis *recurrent dropout* gali apsunkinti ilgojo laikotarpio priklausomybių išmokimą – LSTM tiesiog „užmiršta“ informaciją per anksti, jei tarp žingsnių ryšiai dažnai nutraukiami. Dėl šios priežasties dažnai rekomenduojama naudoti saikingą *RECURRENT_DROPOUT*

**BATCH_SIZE** - mokymo partijų dydis, nurodantis, kiek treniravimo pavyzdžių (teksto sekų) modelis apdoroja prieš atnaujindamas svorius. Mokant modelį, duomenų rinkinys padalijamas į partijas po BATCH_SIZE pavyzdžių; kiekviena partija perėjus tinklą suformuoja svorių gradiento vidurkį, ir tada atliekamas vienas mokymosi žingsnis. Nustatyta, kad 32–512 dydžio partijos lemia geresnį apibendrinimą nei labai didelės.

**EPOCHS** - mokymo epochų skaičius, nusakantis, kiek kartų modelis praeis per visą mokymo duomenų rinkinį. EPOCHS skaičius iš esmės susijęs su persimokymu ir nepakankamu išmokimu. Jei epochų per mažai, modelis gali nespėti išmokti duomenų dėsningumų – treniravimo klaida išliks didelė, o tikslumas – mažas tiek treniravimo, tiek testavimo aibėse (modelis bus nepakankamai išmokytas). Dažnai praktikoje tikslinga naudoti ankstyvo stabdymo metodą vietoje fiksuoto epochų skaičiaus – nustatomas pakankamai didelis maksimalus EPOCHS, tačiau realų mokymo trukmę reguliuoja **PATIENCE** kriterijus. Taip užtikrinama, kad modelis treniruojamas tol, kol gerėja jo veikimas su validacijos duomenimis, ir sustabdomas, kai pagerėjimas nustoja.

**PATIENCE** - ankstyvojo stabdymo (angl. *early stopping*) parametras, nusakantis epochų skaičių, kiek laukti nesulaukus modelio pagerėjimo prieš nutraukiant mokymą. Pvz., PATIENCE = 5 reiškia, kad jei modelio validacijos nuostolis negerėja 5 epochas iš eilės, treniravimas bus nutrauktas šeštos stagnacijos epochos pradžioje.

L2_REGULATION - tai papildoma bauda, pridedama prie nuostolių funkcijos tam, kad būtų ribojamas svorių reikšmių dydis ir sumažinama persimokymo rizika. Konkretus reguliacijos intensyvumas nustatomas koeficientu $\lambda$, kuriuo kiekvieno sluoksnio svoriams pridedama $\lambda \cdot \|w\|^2$.

Mažas λ (pvz., 0,001) suteikia lengvą reguliaciją, padedančią išvengti pernelyg didelio svorių augimo, tačiau leidžia modeliui išmokti reikšmingas savybes. Didesnė λ (pvz., 0,1) žymiai slopina svorių didėjimą, priversdama tinklą pasirinkti paprastesnes funkcijas ir taip dar labiau sumažinti persimokymą.

### 3.4.2. Pradinis modelis.

**Pradinis (angl.** *baseline***) modelis.** Eksperimentų pradžioje buvo sukurtas bazinis LSTM modelis, siekiant įvertinti pirminį klasifikavimo efektyvumą. Modelyje naudoti parametrai pateikiami **10-oje lentelėje**.

**10 lentelė**. Neapykantos kalbos aptikimo pradinio LSTM modelio parametrai

| **Parametras** | **Reikšmė** |
|---|---|
| Žodyno dydis | 10000 |
| Įterpimo dimensija | 200 |
| Maksimalus sekos ilgis | 100 |
| LSTM neuronų kiekis | 64 |
| Partijos dydis (batch size) | 32 |
| Epochų skaičius | 10 |
| Ankstyvojo stabdymo kantrybė | 2 |

Modelio struktūrą sudarė: *Embedding* sluoksnis → LSTM sluoksnis → *Dense* sluoksnis su *sigmoid* aktyvacijos funkcija. Gautų rezultatų statistika pateikiama **11-oje lentelėje**.

**11 lentelė**. Neapykantos kalbos aptikimo pradinio LSTM modelio įverčiai

| Klasė | Tikslumas | Atkūrimas | F1-įvertis |
|---|---|---|---|
| 0 | 95% | 98% | 97% |
| 1 | 88% | 74% | 80% |

**Bendras F1: 83%.**

Rezultatai atskleidė modelio silpnybę – žemą atkūrimą mažesnėje klasėje (74%), kas rodo sunkumus atpažįstant neapykantos kalbos atvejus.

### 3.4.3. Parametrų optimizavimo aprašymas

Siekiant sistemingai optimizuoti modelio parametrus, buvo atliekami atskiri eksperimentai, keičiant po vieną parametrą ir fiksuojant validacijos $F_1$ reikšmę.

***EMBEDDING_DIM***

Ištirtos įterpimo dimensijos: 100, 400, 500 ir 1000. Gautos validacijos $F_1$ reikšmės: 0,3081 (100), 0,4600 (400), 0,4134 (500) ir 0,2874 (1000). Aukščiausias validacijos $F_1$ (0,4600) užfiksuotas naudojant EMBEDDING_DIM = 400.

***MAX_NB_WORDS***

Išbandyti žodynai su 10 000, 50 000, 100 000 ir 200 000 įrašų. Validacijos $F_1$ reikšmės: 0,4120 (10 000), 0,3678 (50 000), 0,3954 (100 000), 0,3569 (200 000). Optimalus balansas pasiektas su MAX_NB_WORDS = 10 000; vėlesniame teste padidinus iki 20 000, klasės 1 atkūrimas padidėjo nuo 0,84 iki 0,85, o $F_1$ – nuo 0,88 iki 0,89, todėl galutinėje konfigūracijoje pasirinkta MAX_NB_WORDS = 20 000.

***LSTM_UNITS***

Testuoti 32, 64 ir 128 vienetai. Validacijos $F_1$ reikšmės: 0,4540 (32), 0,3884 (64), 0,3338 (128). Dėl aukščiausio validacijos $F_1$ (~ 0,4540) pasirinkta 32.

***DROPOUT***

Išanalizuotos *dropout* reikšmės 0,2; 0,3; 0,6. Gautos validacijos $F_1$ reikšmės: 0,3606 (0,2), 0,3911 (0,3), 0,3604 (0,6). Optimalus parametras: 0,3.

***RECURRENT_DROPOUT***

Analizuoti lygiai: 0,0; 0,3. Gautos validacijos $F_1$ reikšmės: 0,3914 (0,0) ir 0,3604 (recurrent_dropout 0,3). Optimalus parametras: RECURRENT_DROPOUT = 0,0.

***BATCH_SIZE***

Išbandytos reikšmės: 16 ($val\_F_1$ = 0,3596), 32 (0,3604), 64 (0,3965). Dėl aukščiausio validacijos $F_1$ pasirinkta 64.

***L2_REG***

Išanalizuoti $L_2$ reguliacijos koeficientai: 0,0; 0,001; 0,01; 0,1. Gautas $val\_F_1$: 0,3604 (0,0), 0,4389 (0,001), 0,3656 (0,01) ir 0,4496 (0,1). Optimalus parametras: 0,1.

### 3.4.4. Patobulintas (galutinis) modelis.

Patobulinto modelio parametrai apibendrinti **12-oje lentelėje**.

**12 lentelė**. Neapykantos kalbos aptikimo LSTM modelio parametrai

| Parametras | Reikšmė |
|---|---|
| ***MAX_NB_WORDS*** (Žodyno dydis) | 20000 |
| ***EMBEDDING_DIM*** (Įterpimo dimensija) | 400 |
| ***MAX_SEQUENCE_LENGHT*** (Maksimalus sekos ilgis) | 200 |
| ***LSTM_UNITS*** (LSTM neuronų kieki)s | 32 |
| ***DROPOUT*** | 0,3 |
| ***RECURRENT_DROPOUT*** | 0,0 |
| ***DENSE_UNITS*** (*Dense* neuronų kiekis) | 32 |
| ***L2_REG*** (L2 reguliavimas) | 0,1 |
| ***BATCH_SIZE*** (Partijos dydis) | 64 |
| ***EPOCHS*** (Epochų skaičius) | 10 |
| ***PATIENCE*** (Ankstyvojo stabdymo kantrybė) | 2 |

Modelio kompiliavimo metu kaip nuostolių funkcija buvo pasirinkta dvejetainė sankirtos entropija, kuri tinka užduočiai su dviem klasėmis. Optimizavimui naudotas adaptacinis *Adam* algoritmas, automatiškai pritaikantis mokymosi žingsnį kiekvienam parametrui ir skatindamas greitesnį bei stabilesnį konvergavimą.
Treniravimo procese naudotas ankstyvasis stabdymas *EarlyStopping*, stebint validacijos $F_1$: treniravimas nutraukiamas, kai po iš anksto nustatyto epochų skaičiaus neįvyksta $F_1$ pagerėjimo, o galutinės svorio reikšmės atkuriamos iš epochos, kurioje pasiektas geriausias validacijos rezultatas. Tokiu būdu užtikrinama, kad modelis nevykdytų perteklinio mokymo ir būtų išsaugotos optimalūs rezultatai.

**13 lentelėje** pateikti klasifikavimo rezultatai dviem klasėms.

Neutralios klasės identifikavimas: labai aukštas tikslumas (97 %) ir jautrumas (99 %), rodo, kad neutralūs pavyzdžiai beveik neklasifikuojami klaidingai.
Neapykantos kalbos atpažinimas: tikslumas siekia 91 %, o jautrumas — 87 %, o tai reiškia, kad modelis didžiąją dalį neapykantos atvejų atpažįsta teisingai, nors tam tikra dalis gali likti nepastebėta. $F_1$ balas padidintas iki 98 % ne-neapykantos kalbai ir 89 % neapykantos kalbai, kas rodo subalansuotą modelio gebėjimą valdyti tiek klaidingų teigiamų, tiek klaidingų neigiamų prognozių riziką praktiniam naudojimui.

**13 lentelė**. Neapykantos kalbos aptikimo LSTM modelio įverčiai

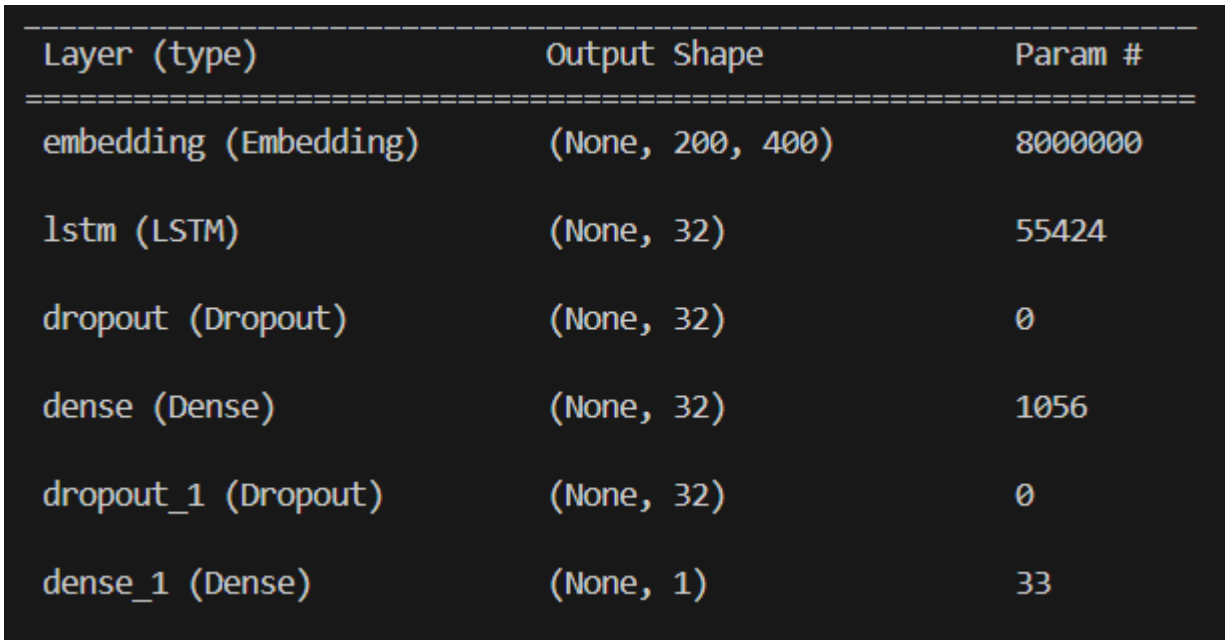
| Klasė | Tikslumas | Atkūrimas | F1-įvertis |
|---|---|---|---|
| 0 | 97% | 99% | 98% |
| 1 | 91% | 87% | 89% |

**Bendras F1: 97%.**

Galutinis modelis parodė ženkliai geresnį mažumos klasės atpažinimą ir bendrą rezultatų balansą, tinkantį praktiniam neapykantos kalbos klasifikavimo uždaviniui spręsti.

### 3.4.5. Neapykantos kalbos aptikimo LSTM modelio struktūra

**17 pav**. iliustruoja galutinio LSTM tinklo sluoksnių eiliškumą ir parametrų srautus.




**17 pav**. Neapykantos kalbos aptikimo LSTM modelio struktūra

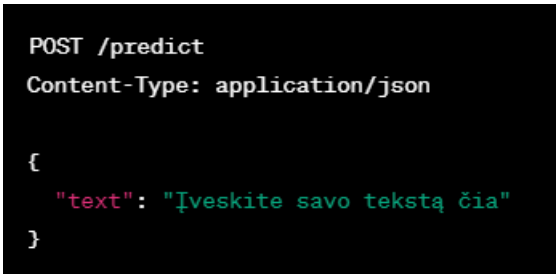
Pirmame etape įvesties tekstas paverčiamas žodžių vektoriais, kurių ilgis atitinka numatytą žodyno dydį ir įterpimo dimensiją. Vektoriai perduodami į LSTM sluoksnį, kuris fiksuoja ilgalaikes sekos priklausomybes. Po LSTM sluoksnio taikomas atsitiktinis neuronų išjungimas (angl. *dropout*), siekiant sumažinti persimokymo riziką. Tada seka tankus sluoksnis su aktyvacijos funkcija, papildytas $L_2$ reguliacija, vėl pritaikant neuronų išjungimą, ir galiausiai - vienas neuronas su *sigmoid* aktyvacija, generuojantis klasės tikimybę.

## 3.5. Neapykantos kalbos aptikimo API

Sistemos serverinė dalis veikia naudojant "Flask" API, kuri pasirinkta dėl paprastumo ir efektyvumo diegiant "Python" programas. API yra tarpininkas tarp LSTM modelio ir naudotojo sąsajos. Jis gauna

teksto įvestį iš naudotojo, apdoroja ją per modelį ir grąžina prognozę, kurioje nurodoma, ar tekstas priskiriamas neapykantos kalbos kategorijai, ar ne.

Pradedant veikti API, yra įkeliamas iš anksto apmokytas modelis, saugomas faile.




**18 pav**. Neapykantos kalbos aptikimo API struktūra

API turi vieną pagrindinį įvesties tašką */predict* („**POST“** metodas, pavaizduotas **18 pav**.).

1. Priima JSON objektą, turintį rakto „text“ reikšmę, kurią vartotojas pageidauja klasifikuoti.
2. Gautas tekstas yra apdorojamas specialiai sukurta funkcija, kuri atlieka šiuos veiksmus:
3. Apdorotas tekstas perduodamas į modelį prognozavimui atlikti.
4. Modelis grąžina skaičių tarp 0 ir 1, kuris nurodo neapykantos kalbos tikimybę.
5. Rezultatas pateikiamas JSON formatu.

### 3.6. Neapykantos kalbos aptikimo naršyklės plėtinys

Projekto metu sukurtas naršyklės plėtinys „Neapykantos Kalbos Detektorius“ suteikia vartotojams galimybę aptikti neapykantos kalbą internete naršomame turinyje. Plėtinys realizuotas remiantis Chrome *Manifest* V3 standartu, kuris numato naujus funkcinius ir saugumo reikalavimus naršyklės plėtiniams.

#### 3.6.1. Plėtinio veikimo principas

Plėtinys aktyvuojamas naršyklės įrankių juostos mygtuku, kuris paleidžia specialią perdangą (angl. *overlay*). Šioje perdangoje vartotojai gali įvesti arba įklijuoti tikrinamą tekstą. Perdangos funkcionalumas sukurtas naudojant JavaScript ir CSS, kurie užtikrina interaktyvią vartotojo sąsają.

#### 3.6.2. Funkciniai komponentai

***background.js*** - atsakingas už plėtinio paleidimą bei perdangos aktyvavimą aktyviame naršyklės lange, naudojant *chrome.scripting* API, *JavaScript* ir *CSS*.

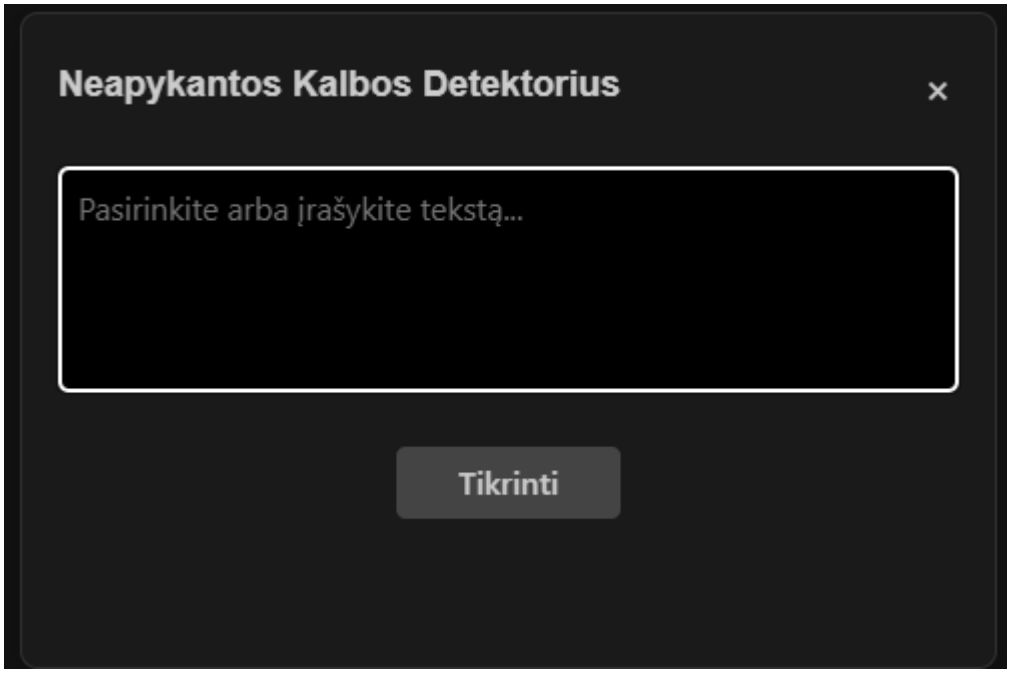
***content.js*** - įterpia vartotojo sąsajos komponentus į aktyvų puslapį, apdoroja vartotojo įvestą tekstą bei siunčia užklausas į serverį. Kodas taip pat leidžia vartotojui patogiai perkelti perdangos langą ekrane.

***manifest.json*** - plėtinio funkcionalumas ir leidimai apibrėžti šiame faile. Pagrindiniai parametrai yra šie:

- Leidimas dirbti su aktyviu naršyklės langu.
- Svetainės leidimai, kurie leidžia plėtiniui veikti bet kokiame URL.
- Naršyklės įrankių juostos veikimo aprašas, skirtas patogiam vartotojų naudojimui.

### 3.6.3. Naudotojo sąsaja

Perdangą sudaro 4 elementai. Vizualiai pavaizduota **19 pav**.




**19 pav**. Neapykantos kalbos naršyklės plėtinio vaizdas

1. Teksto įvesties laukas vartotojų įvedamam arba įklijuojamam tekstui.
2. Mygtukas, inicijuojantis teksto analizę.
3. Rezultatų rodymo sritis, vizualiai pateikianti klasifikavimo išvadą apie neapykantos kalbą.
4. Galimybė vartotojui interaktyviai perkelti perdangos langą ekrane.

## 4. Eksperimentinis modelio vertinimas

Šiame skyriuje atliekamas sukurto modelio vertinimas įvairiais parametrais, taip pat palyginama su kitais sprendimais.

### 4.1. Modelio vertinimas

Vertinimo rodikliai išsamiai aprašyti 1.6 skyriuje.

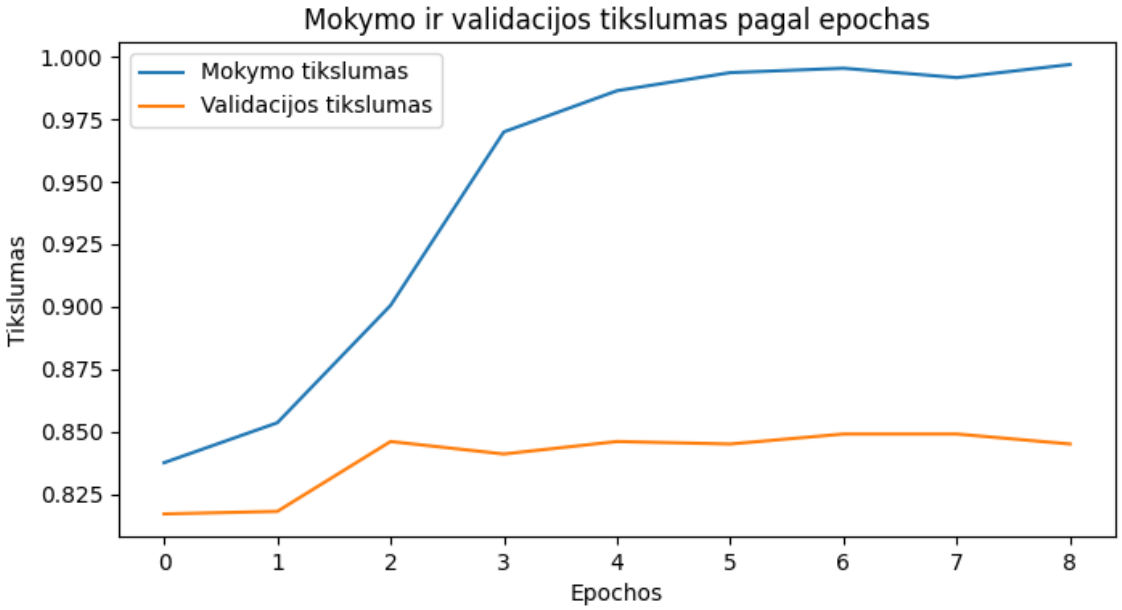
**14 lentelė**. Neapykantos kalbos aptikimo modelio metrikos

| **Metrikos pavadinimas** | **Reikšmė** |
|---|---|
| Tikslumas | 97% |
| Preciziškumas | 95% |
| Atkūrimas | 93% |
| F1 įvertis (F1-Score) | 94% |

Pateiktoje lentelėje (**14 lentelė**) apžvelgiami bendri modelio metrikos rodikliai.

- **Tikslumas**: 97 % – nurodo, kad bendras teisingų prognozių santykis siekia 97 iš 100 atvejų.
- **Preciziškumas**: 95 % – reiškia, kad iš visų teigiamų klasifikacijų 95 % buvo tiksliai identifikuotos.
- **Atkūrimas**: 93 % – parodo, kad modelis aptinka 93 % visų tikrųjų teigiamų atvejų.
- **$F_1$ balas**: 94 % – harmoninis preciziškumo ir atkūrimo vidurkis, suapvalintas iki 94 %, atspindi bendrą modelio gebėjimą balansuotai nustatyti tiek tikrus teigiamus, tiek klaidingai neigiamus atvejus.

### 4.1.1. Mokymo ir validacijos tikslumo pokyčiai per epochas

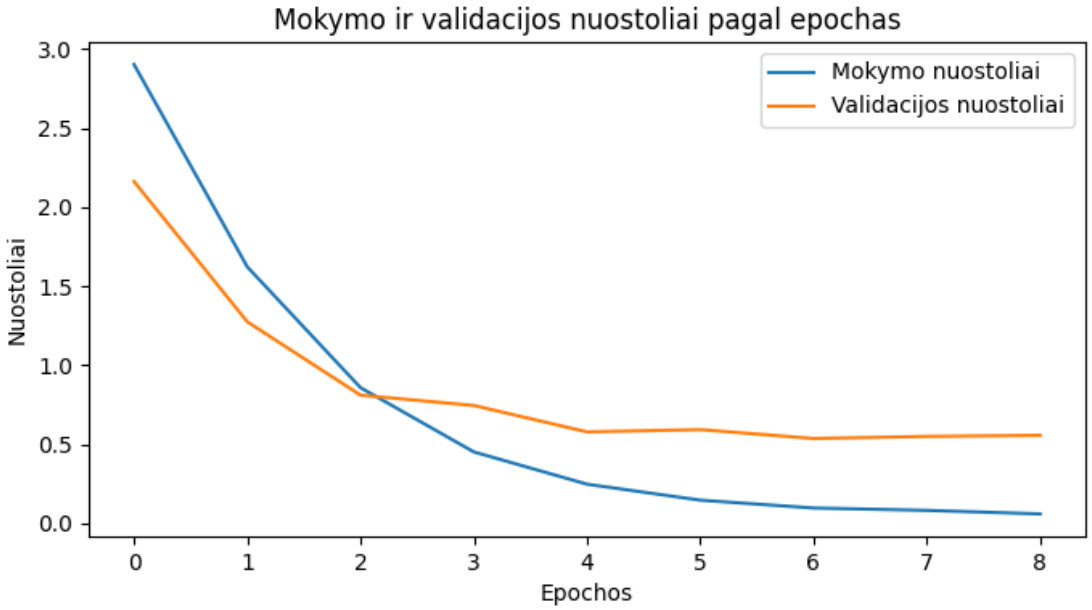



**20 pav**. Mokymosi ir validacijos tikslumo grafikas

Pateiktame grafike (**20 pav**.) matomas stabilus mokymo tikslumo augimas, kuris pasiekia 96% po 6 epochų. Tai rodo, kad modelis mokymo metu sėkmingai išmoksta atpažinti neapykantos kalbos požymius pateiktuose komentaruose. Validacijos tikslumas, kuris matuoja modelio gebėjimą apibendrinti ir atpažinti neapykantos kalbą naujuose, nematytuose komentaruose, taip pat didėja, tačiau išlieka kiek žemesniame lygmenyje (apie 85%–86%) ir stabilizuojasi paskutinėse epochose.

Stebimas skirtumas tarp mokymo ir validacijos tikslumo vadinamas persimokymu (angl. *overfitting*). Šis reiškinys pasireiškia, kai modelis labai gerai išmoksta mokymo duomenis, tačiau ne taip efektyviai apdoroja naujus, nematytus duomenis. Skirtumas, matomas grafike, rodo tik lengvą persimokymą, kuris yra gana dažnas reiškinys, ypač esant mažam ir specifiniam duomenų rinkiniui (šiuo atveju 5000 komentarų). Tokiu atveju nedidelis persimokymas yra laikomas priimtinu, nes modelis išlieka efektyvus praktinėse taikymo situacijose.

### 4.1.2. Mokymo ir validacijos nuostolių pokyčiai per epochas



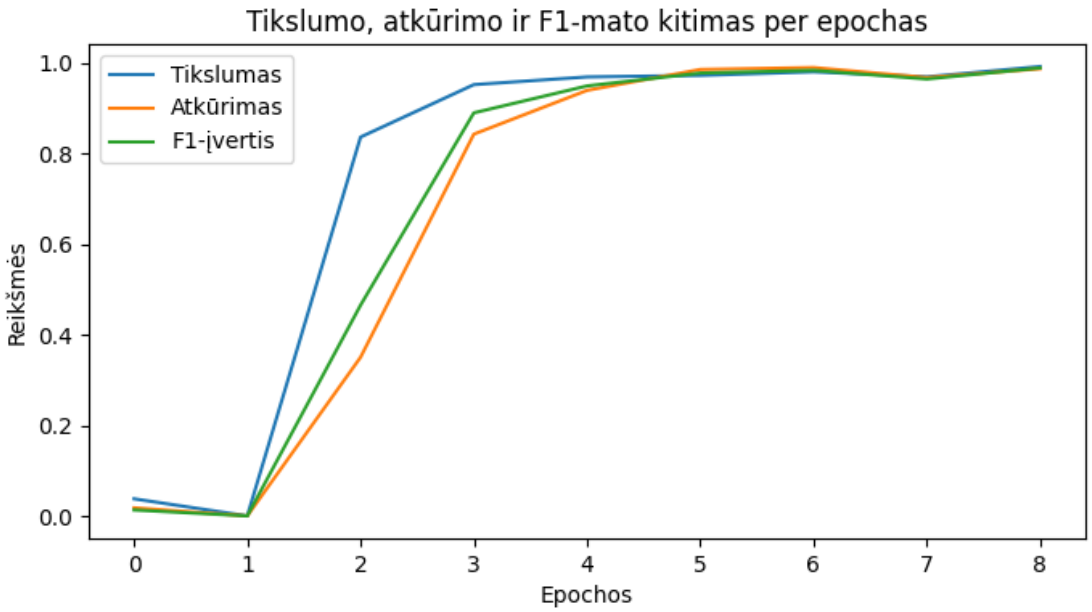

**21 pav**. Validacijos nuostoliai pagal treniravimo epochas

Grafike (**21 pav**.) pavaizduotas modelio nuostolių (angl. *loss*) pokytis per mokymo epochas. Nuostolis yra metrika, naudojama įvertinti modelio prognozavimo paklaidą – kuo mažesnis nuostolis, tuo tikslesnės modelio prognozės.

Pastebimas ryškus mokymo nuostolių mažėjimas, kuris rodo, jog modelis efektyviai mokosi iš pateiktų duomenų. Validacijos nuostoliai po pirminio sumažėjimo išlieka stabilūs, tačiau vėlesnėse epochose matomas jų nežymus didėjimas. Šis reiškinys atspindi jau anksčiau pastebėtą lengvą persimokymą.

Modelis išlieka pakankamai tikslus ir stabilus, todėl gali būti sėkmingai taikomas praktinėje neapykantos kalbos aptikimo užduotyje.

### 4.1.3. Tikslumo, atkūrimo ir F1-mato analizė



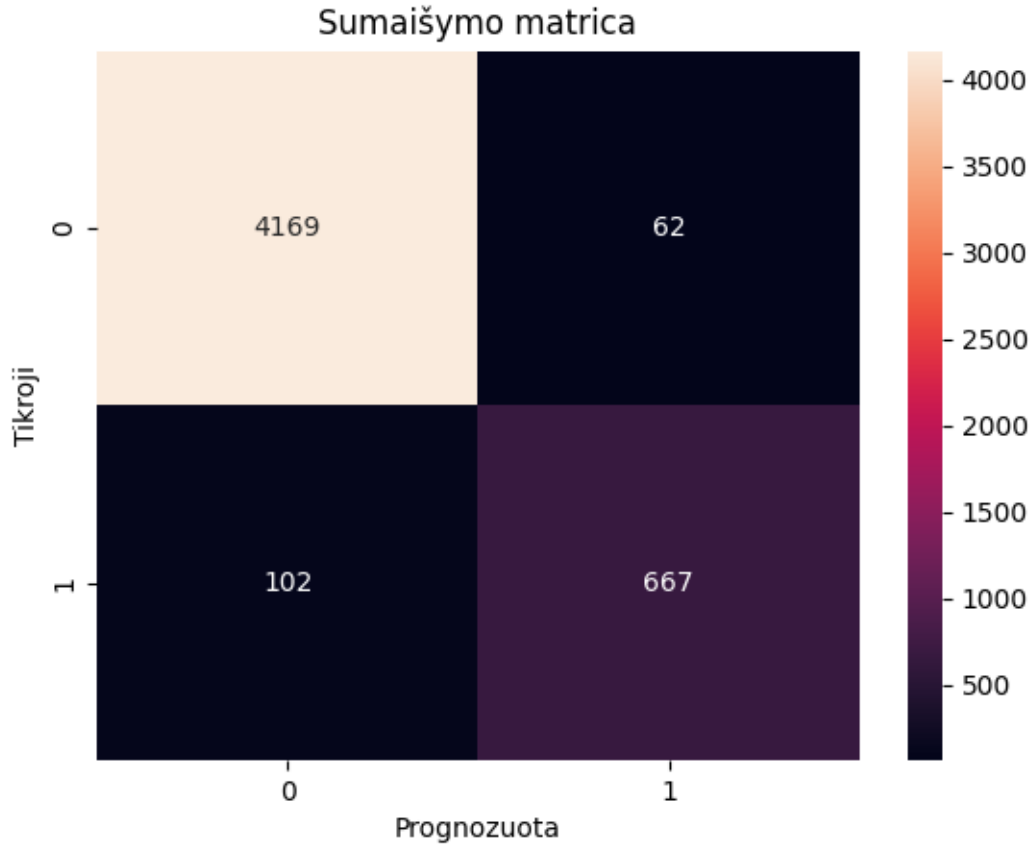

**22 pav**. Tikslumo, atkūrimo bei F1 grafikas pagal treniravimo epochas

**22 pav**. grafike pavaizduoti trys pagrindiniai klasifikavimo vertinimo rodikliai: tikslumas, atkūrimas ir F1 įvertis. Šie rodikliai yra esminiai vertinant modelio veikimą, ypač sprendžiant tokias subtilias užduotis kaip neapykantos kalbos aptikimas.

Tikslumas parodo, kokia dalis modelio prognozuotų neapykantos kalbos atvejų iš tiesų buvo teisingi. Atkūrimas nusako, kiek iš visų realių neapykantos kalbos atvejų modelis sugebėjo teisingai identifikuoti. F1 įvertis yra šių dviejų metrikų harmoninis vidurkis, kuris ypač tinkamas, kai duomenų klasės yra nesubalansuotos.

Pateiktame grafike matomas greitas visų trijų rodiklių augimas per pirmąsias tris mokymo epochas. Po to reikšmės stabilizuojasi ties gana aukštu lygiu. Šių kreivių artumas rodo, jog modelis subalansuotai veikia su abejomis klasėmis – tiek neapykantos, tiek ne neapykantos kalbos atvejais. Toks subalansuotas veikimas yra itin svarbus, kad būtų sumažinta tiek klaidingų teigiamų, tiek klaidingų neigiamų prognozių rizika, kurių padariniai neapykantos kalbos analizėje gali būti reikšmingi.

### 4.1.4. Sumaišymo matrica



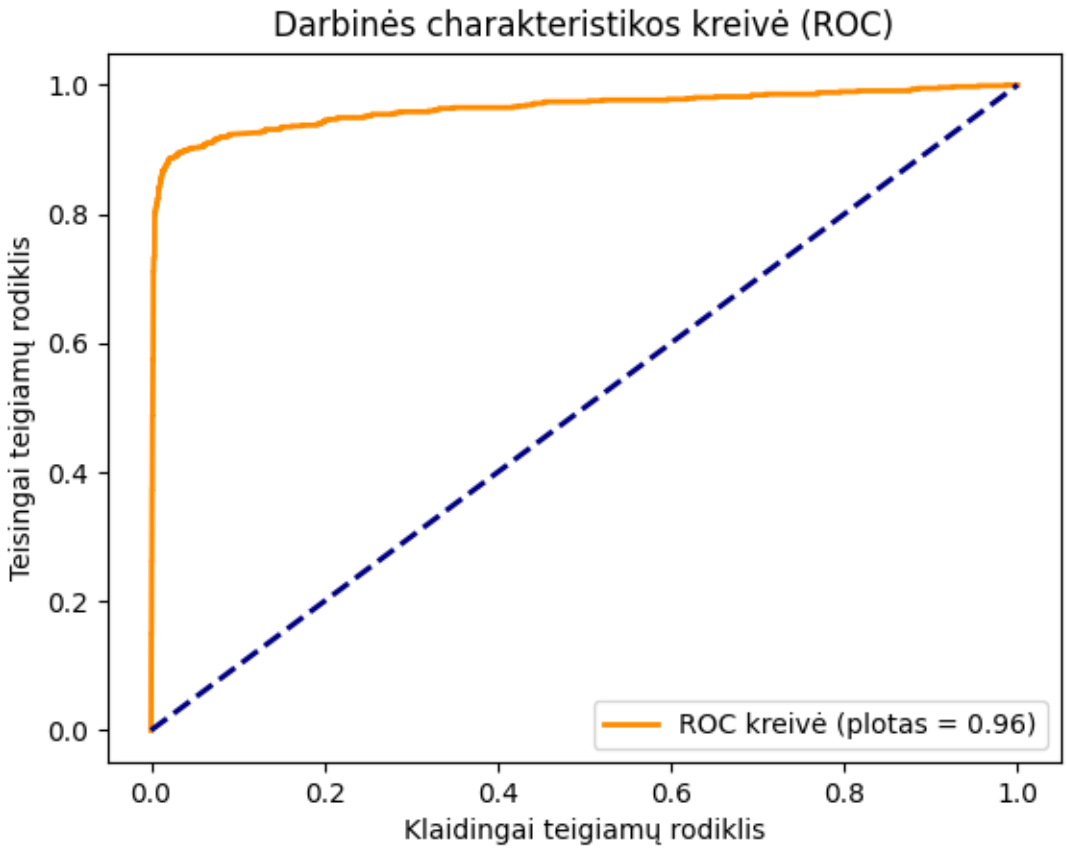

**23 pav**. Sumaišymo matrica

Matricoje (**23 pav**.) pateikiama detali modelio prognozių analizė, leidžianti įvertinti klasifikavimo tikslumą pagal kiekvieną galimą rezultatą. Įstrižainėje esantys skaičiai (4169 ir 667) atitinka teisingai atpažintus atvejus – atitinkamai neapykantos kalbos nebuvimą (0) ir buvimą (1). Šie skaičiai žymi tikruosius neigiamus ir tikruosius teigiamus rezultatus.

Kituose langeliuose pateikiami klaidingi sprendimai: 62 atvejai, kai modelis neteisingai identifikavo neapykantos kalbą (klaidingi teigiami), ir 102 atvejai, kai neapykantos kalba liko neaptikta (klaidingi neigiami).

Pastebimas ryškus skirtumas tarp teisingų ir klaidingų klasifikacijų – dauguma prognozių yra teisingos, o klaidingų atvejų kiekis yra palyginti mažas. Tai rodo, kad modelis sugeba patikimai atskirti tekstus su neapykantos kalba nuo neutralių ar saugių tekstų, o jo sprendimai išlieka stabilūs ir pakankamai tikslūs. Šis pasiskirstymas patvirtina modelio tinkamumą taikyti praktiniuose scenarijuose, kur ypač svarbus yra balansas tarp jautrumo ir specifiškumo.

### 4.1.5. ROC kreivė



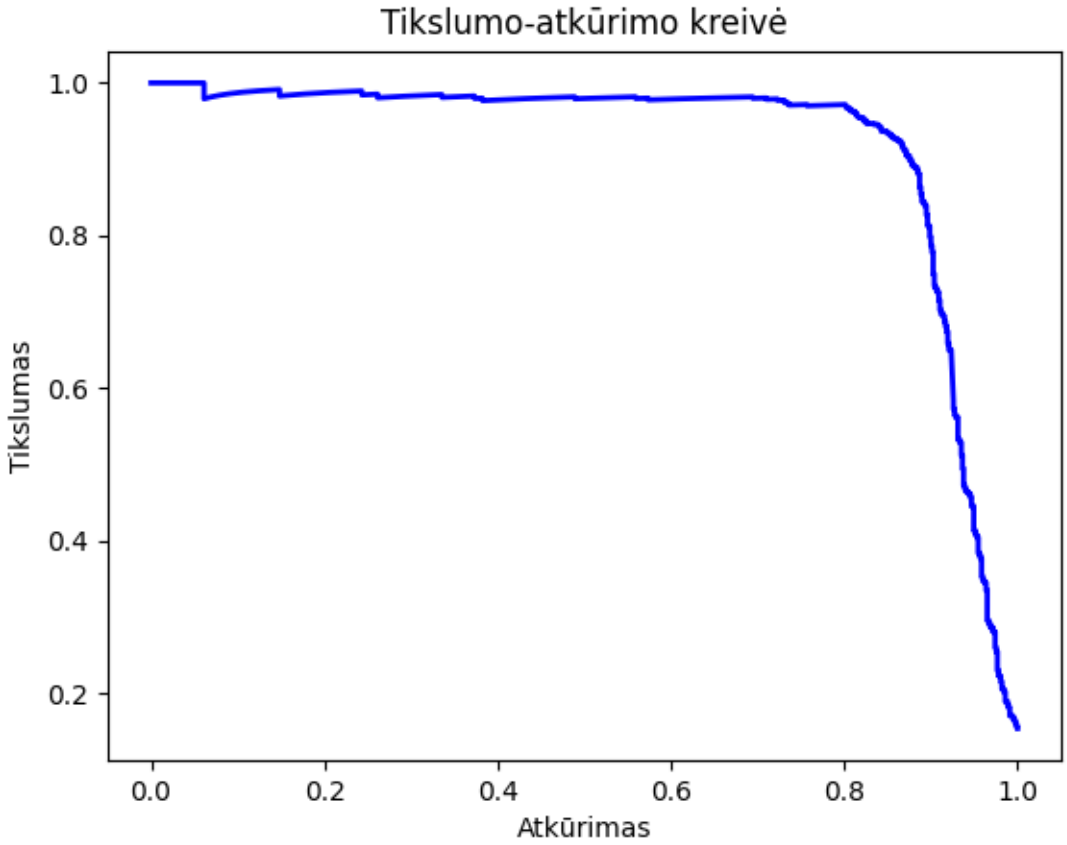

**24 pav**. Darbinės charakteristikos kreivė (ROC)

ROC (angl. „Receiver Operating Characteristic“) kreivė, pateikta **24 pav**., parodo modelio gebėjimą atskirti dvi klases, keičiant sprendimo slenkstį. Horizontalioje ašyje pavaizduotas klaidingai teigiamų prognozių (angl. *false positive rate*) rodiklis, o vertikalioje – teisingai teigiamų prognozių (angl. *true positive rate*) rodiklis.

Modelis, kurio ROC kreivė yra arčiau viršutinio kairiojo kampo, laikomas tikslesniu. Pateiktame grafike ROC kreivės plotas po kreive (AUC – „Area Under the Curve“) siekia 0,96, o tai rodo itin aukštą modelio tikslumą.

Aukštas AUC rezultatas reiškia, kad modelis pasižymi tiek aukštu jautrumu (gebėjimu atpažinti neapykantos kalbą), tiek aukštu specifiškumu (gebėjimu atmesti neutralius komentarus), nepriklausomai nuo pasirinkto klasifikavimo slenksčio. Toks rezultatas rodo modelio tinkamumą taikymui realiose situacijose, kur būtina išlaikyti pusiausvyrą tarp neteisingai atmestų ir neteisingai priimtų sprendimų.

### 4.1.6. Tikslumo-atšaukimo kreivė




**25 pav**. Tikslumo-atkūrimo kreivė

Tikslumo–atkūrimo kreivė, pateikta **25 pav**., atspindi modelio gebėjimą išlaikyti aukštą tikslumą net ir didėjant atkūrimui. Ši kreivė ypač svarbi tais atvejais, kai duomenų klasės yra nesubalansuotos – tai dažnas reiškinys neapykantos kalbos aptikimo užduotyje, kur neapykantos kalbos atvejų dažnis yra žymiai mažesnis nei neutralių komentarų.

Pateiktame grafike matyti, kad modelis išlaiko itin aukštą tikslumą (artimą 1) net ir tada, kai atkūrimo rodiklis siekia 80–85 %. Tai reiškia, jog modelis sugeba identifikuoti didžiąją dalį neapykantos kalbos atvejų neaukodamas klasifikavimo tikslumo.

Toks elgesys yra labai svarbus praktinėse situacijose, kuriose svarbu ne tik aptikti kuo daugiau neapykantos kalbos atvejų (aukštas atkūrimas), bet ir sumažinti klaidingai pažymėtų teigiamų atvejų skaičių (aukštas tikslumas). Todėl ši kreivė patvirtina, kad modelis efektyviai subalansuoja abu aspektus.

### 4.1.7. Bendras vertinimas

Atsižvelgiant į duomenų kiekį (5000 įrašų) ir jų specifiką („Reddit“ komentarai apie neapibrėžtus ir kontekstui jautrius dalykus), pasiekti rezultatai yra geri. Minimalus persimokymas priimtinas praktiniam modeliui, o aukšti ROC ir F1 rodikliai patvirtina modelio tinkamumą praktiniam pritaikymui, nes modelis tiksliai veikia ir su naujais, treniravimo duomenų rinkinyje nematytais duomenimis.

### 4.1.8. Modelių palyginimas

Tikslas – palyginti sukurtąjį pritaikytą lietuvių kalbai modelį su kitais šiuolaikiniais modeliais („BERT“, „LitLat BERT“) neapykantos kalbos aptikimo uždavinyje lietuvių kalba.

**Naudojami modeliai**

1. **Darbe siūlomas LSTM modelis.**
2. **BERT**: Populiarus iš anksto apmokytas *transformer* tipo modelis, gebantis gerai suprasti teksto kontekstą.
3. **LitLat BERT**: trikalbis modelis, naudojantis *XLM-RoBERTa-base* tvirtai optimizuotą (angl. *robustly optimized*) architektūrą [16]) ir apmokytas su lietuvių, latvių ir anglų kalbų duomenimis.

**Pirminio apdorojimo lygiai**

Eksperimente bus lyginami trys pirminio duomenų apdorojimo lygiai:

1. **Be apdorojimo** (originalus tekstas).
2. **Minimalus apdorojimas**: skyrybos ženklų ir specialiųjų simbolių pašalinimas.
3. **Pilnas pritaikytas apdorojimas:**
    a. Lietuviškų simbolių keitimas į ASCII atitikmenis.
    b. Lietuviškų stop-žodžių pašalinimas.
    c. Žargono ir santrumpų išskleidimas į pilną formą.
    d. Lietuvių kalbai pritaikytas kamienavimas.

**Vertinimo metrikos:**

1. Tikslumas.
2. Preciziškumas.
3. Atkūrimas.
4. F1 įvertis.

### 4.1.9. Modelių palyginimas – be apdorojimo

Naudojant neapdorotus tekstus, darbe siūlomo modelio veikimas reikšmingai suprastėjo – tik 21 % atkūrimas rodo, jog modelis gebėjo aptikti tik nedidelę dalį realios neapykantos kalbos, nepaisant to, kad F1 liko aukštas (77 %), kas reiškia, kad klasifikacija buvo labai atsargi. Tuo tarpu BERT išlaikė gerą stabilumą – 78,3 % F1, nes yra treniruotas apdoroti įvairaus pobūdžio tekstus. Palyginimas pateiktas **15-oje lentelėje**.

**15 lentelė**. Eksperimento rezultatai be apdorojimo

| Modelis | Tikslumas | Preciziškumas | Atkūrimas | F1 |
|---|---|---|---|---|
| BERT | 82,9% | 73,7% | 56,7% | 78,3% |
| LitLat BERT | 82,0% | 40,9% | 50,0% | 73,5% |
| Darbe siūlomas LSTM modelis | 80,0% | 52,0% | 21,0% | 77,0% |

### 4.1.10. Modelių palyginimas – minimalus apdorojimas

Pašalinus tik teksto triukšmą, LSTM modelis toliau išliko silpnas – F1 įvertis išliko toks pat (77 %), tačiau preciziškumas nukrito iki vos 15 %. Tai patvirtina, kad lietuvių kalbai būtinas gilus kalbinis apdorojimas, o bendros technikos (minimalus valymas) nėra pakankama. Palyginimas pateiktas **16-oje lentelėje**.

**16 lentelė**. Eksperimento rezultatai pritaikius minimalų duomenų apdorojimą

| Modelis | Tikslumas | Preciziškumas | Atkūrimas | F1 |
|---|---|---|---|---|
| BERT | 81,7% | 40,8% | 50,0% | 73,5% |
| LitLat BERT | 81,5% | 38,8% | 47,0% | 72,5% |
| Darbe siūlomas LSTM modelis | 80% | 15,0% | 6,0% | 77,0% |

### 4.1.11. Modelių palyginimas – pilnas apdorojimas

Panaudojus pilną lietuvių kalbai pritaikytą duomenų apdorojimą modelio rodikliai išaugo drastiškai. Darbe siūlomas modelis vėl pasiekė 96 % F1, o atkūrimas šoktelėjo iki 83 %, parodydamas, kad apdorojimo kokybė yra lemiamas veiksnys. Palyginimas su pilnu apdorojimu pateikiamas **17-oje lentelėje**.

**17 lentelė**. Eksperimento rezultatai pritaikius pilną duomenų apdorojimą

| Modelis | Tikslumas | Preciziškumas | Atkūrimas | F1 |
|---|---|---|---|---|
| BERT | 83,1% | 69,4% | 14,6% | 78,1% |
| LitLat BERT | 82,7% | 66,8% | 81,7% | 73,4% |
| Darbe siūlomas LSTM modelis | 97,0% | 95,0% | 93,0% | 94,0% |

### 4.1.12. Apibendrinimas

Geriausius rezultatus parodė darbe siūlomas LSTM modelis – pasiekęs net 96,0 % tikslumą ir 96,0 % F1 įvertį, kas rodo ne tik puikų bendrą veikimą, bet ir subalansuotą tikslumo ir atkūrimo santykį. Tuo tarpu BERT ir LitLat BERT modeliai, nors ir gerai žinomi bei stiprūs, pasiekė žymiai žemesnius rodiklius – atitinkamai 78,3 % ir 73,5 % F1. Palyginimas pateiktas **18-oje lentelėje**.

**18 lentelė**. Eksperimento modelių parametrų palyginimas naudojant geriausius rezultatus

| Modelis | Tikslumas | Preciziškumas | Atkūrimas | F1 |
|---|---|---|---|---|
| BERT | 82,9% | 73,7% | 56,7% | 78,3% |
| LitLat BERT | 82,7% | 66,8% | 81,7% | 73,4% |
| Darbe siūlomas LSTM modelis | 97,0% | 95,0% | 93,0% | 94,0% |

BERT modelis, taikant pilną pritaikytą apdorojimą, nors ir išlaikė beveik identišką bendrą tikslumą – nuo 82,9 % iki 83,1 %, patyrė ryškių vidinių pokyčių: preciziškumas sumažėjo nuo 73,7 % iki 69,4 %, o atkūrimas smarkiai nukrito nuo 56,7 % iki vos 14,6 %. Tai reiškia, kad nors BERT modelis ir padidino bendrą tikslumą, jo gebėjimas teisingai identifikuoti neapykantos kalbą tapo žymiai silpnesnis, kas rodo, jog pilnas teksto apdorojimas gali iš dalies pašalinti kontekstą, kuris yra svarbus BERT modeliui atpažįstant neapykantos kalbos požymius.

LitLat BERT modelis parodė priešingą tendenciją – bendras tikslumas padidėjo nuo 82,0 % (be apdorojimo) iki 82,7 % (pilnas apdorojimas), tačiau įdomiausia, jog preciziškumas ženkliai pagerėjo nuo 40,9 % iki 66,8 %, o atkūrimas – nuo 50,0 % iki 81,7 %. Tai rodo, jog LitLat BERT modelis, kuris yra specialiai treniruotas lietuvių kalbai, gerokai efektyviau panaudoja pilnai pritaikytą tekstą. Tokie pokyčiai leidžia daryti išvadą, kad LitLat BERT yra jautresnis ir naudingiau išnaudoja mano projekto pasiūlytą pilną kalbinį apdorojimą nei bendras BERT modelis.

Papildomai reikėtų pažymėti, kad eksperimentas buvo atliktas naudojant palyginti nedidelį – 5000 įrašų – duomenų rinkinį. Ypač ryškus pagerėjimas mano LSTM modelyje naudojant pilnai pritaikytą

apdorojimą gali būti susijęs su tuo, jog modelis maksimaliai efektyviai išnaudoja ribotą duomenų kiekį, kuris po išsamaus apdorojimo tampa lengviau interpretuojamas.

## Išvados

1. Analizuojant neapykantos kalbos reiškinį socialinėse medijose nustatyta, kad ji sukelia reikšmingą emocinę ir psichologinę žalą, mažina vartotojų aktyvumą ir pasitikėjimą platformomis, todėl efektyvus jos aptikimas ir šalinimas yra kritiškai svarbus tiek individualiu, tiek visuomeniniu mastu.
2. Literatūros apžvalga parodė, kad giliojo mokymosi architektūros, ypač LSTM ir BERT, yra dominuojančios neapykantos kalbos atpažinimo srityje, tačiau jų efektyvumas priklauso nuo teksto apdorojimo kokybės ir kalbinių ypatumų. Analizuojant esamus neapykantos kalbos aptikimo įrankius, nustatyta, kad lietuvių kalba iki šiol nėra atstovaujama ir komerciniai įrankiai orientuoti tik į populiariausias pasaulio kalbas.
3. Tyrimo metu sukurtas lietuvių kalbai pritaikytas LSTM modelis efektyviai aptiko neapykantos kalbą socialinių tinklų tekstuose ir pasiekė aukštą klasifikavimo tikslumą - 97 %. Šis rezultatas įrodo, kad tinkamai adaptuotos teksto apdorojimo technikos ir specializuotas neuroninis tinklas gali sėkmingai spręsti neapykantos kalbos atpažinimo problemą specifinei kalbai.
4. Eksperimentiniai rezultatai parodė, kad išsamus ir kalbai specifiškai pritaikytas teksto apdorojimas, apimantis lietuviškų simbolių konvertavimą, kamienavimą, stop-žodžių pašalinimą bei žargono išplėtimą, ženkliai pagerina klasifikacijos rezultatus. Pritaikius šiuos žingsnius, F1 rodiklis pakilo nuo 77% iki 94%.
5. Atliekant palyginamąją analizę su kitais šiuolaikiniais sprendimais, BERT ir LitLat BERT modeliais, LSTM pagrįstas modelis parodė aukštesnį tikslumą. Matuojant F1 rodiklį BERT pasiekė 78,3%, LitLat BERT – 73,4%, o LSTM modelis – net 94%. Tai leidžia teigti, kad LSTM architektūra yra efektyvi ir pagrįsta alternatyva sudėtingesniems ir resursams reiklesniems modeliams, ypač tada, kai anotuotų duomenų apimtis yra ribota.
6. Atlikta ROC, PR ir sumaišymo matricos analizė parodė, kad darbe sukurtas modelis pasižymi dideliu jautrumu ir tikslumu, o tai patvirtina jo gebėjimą patikimai atskirti neapykantos kalbą nuo neutralaus turinio. ROC kreivės plotas siekė 0,96, o sumaišymo matrica parodė, kad buvo tik 62 atvejai, kai modelis neteisingai identifikavo neapykantos kalbą (klaidingi teigiami), ir 102 atvejai, kai neapykantos kalba liko neaptikta (klaidingi neigiami), o 4836 įrašai buvo identifikuoti teisingai.
7. Tyrimo metu, remiantis darbe siūlomu metodu, sukurtas neapykantos kalbos atpažinimo sistemos prototipas, susidedantis iš specializuoto teksto apdorojimo programos, LSTM neuroninio tinklo, REST API ir naršyklės plėtinio. Eksperimentinis vertinimas parodė, kad siūloma metodo realizacija turi didelį praktinį potencialą ir leidžia realiame laike aptikti neapykantos kalbą įvairiose interneto svetainėse.

**Literatūros sąrašas**


1. AHN, H. ir kt. SharedCon: Implicit Hate Speech Detection using Shared Semantics. *Findings of the Association for Computational Linguistics ACL*. 2024. Prieiga per: doi: https://doi.org/10.18653/v1/2024.findings-acl.622
2. ALJAWAZERI, J.A. - JASIM, M.N. Addressing Challenges in Hate Speech Detection Using BERT-Based Models: A Review. *Iraqi Journal for Computer Science and Mathematic*s. 2024, vol._5, no. 2, pp. 11-26. ISSN 2788-7421.
3. ALRUWAILI, A. - ALSALIM, M. Data diversity and its impact on machine learning fairness. *International Journal of Cloud Computing and Database Management*. 2024, vol._5, no. 1, pp. 38–41. ISSN 27075907.
4. BAILLY, A. ir kt. Effects of dataset size and interactions on the prediction performance of logistic regression and deep learning models. *Computer Methods and Programs in Biomedicine* . 2022, vol. _213. ISSN 0169-2607.
5. BERRAR, D. Bayes' Theorem and Naive Bayes Classifier. In: *Encyclopedia of Bioinformatics and Computational Biology (Second Edition)*. Oxford: Elsevier, 2025. p. 483–494. ISBN 978-0-323-95503.
6. BREIMAN, L. Random Forests. *Machine Learning*. 2001, vol._45, no. 1, pp. 5–32.
7. CAHYANA, N.H. ir kt. Semi-supervised Text Annotation for Hate Speech Detection using K-Nearest Neighbors and Term Frequency-Inverse Document Frequency. *International Journal of Advanced Computer Science and Applications*. 2022, vol._13, no. 10, pp. 15-32.
8. CORTES, C. Support-vector networks. *Machine Learning* . 1995, vol._20, no. 3, pp. 273–297.
9. COVER, T. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory* . 1967, vol._13, no. 1, pp. 21–27.
10. *DATAQUEST Tools for Text Analysis: Machine Learning and Natural Language Processing* [interaktyvus]. 2022 [žiūrėta 2022-11-28]. Prieiga per: https://www.dataquest.io/blog/using-machine-learning-and-natural-language-processing-tools-for-text-analysis
11. DEVLIN, Jacob. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Informatikos mokslai*. 2019, 1-13. Prieiga per: doi: https://doi.org/10.48550/arXiv.1810.04805
12. FARIAS, D.I.H. - ROSSO, P. Chapter 7 - Irony, Sarcasm, and Sentiment Analysis. In: *Sud. Sentiment Analysis in Social Networks*. Boston: Morgan Kaufmann, 2017, pp. 113–128. ISBN 978-0-12-804412.
13. FITZSIMMONS, L. ir kt. Diversity in Machine Learning: A Systematic Review of Text-Based Diagnostic Applications. *Applied Clinical Informatics* . 2022, vol_.13, no. 3, pp. 569–582.
14. FORTUNA, P. - NUNES, S. A Survey on Automatic Detection of Hate Speech in Text. *Informatikos mokslai*. 2018, vol_.51, no. 4, pp. 1-30.
15. FREUND, Y. - SCHAPIRE, R.E. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Informatikos mokslai*. 1997, vol_.55, no. 1, pp. 119–139.
16. *Detoxify* [interaktyvus]. 2020 [žiūrėta 2025-05-10]. Prieiga per: https://github.com/unitaryai/detoxify
17. HOCHREITER, S. - SCHMIDHUBER, J. Long Short-Term Memory. In Neural Computation. 1997, vol_.9, no. 8, pp. 1735–1780.

18. HÜSÜNBEYI, Z.M. Identifying Hate Speech Using Neural Networks and Discourse Analysis Techniques [interaktyvus]. 2022 [žiūrėta 2022-11-28]. Prieiga per: https://aclanthology.org/2022.lateraisse-1.5/
19. JAHAN, M.S. – OUSSALAH. Towards Weakly-Supervised Hate Speech Classification Across Datasets. *Informatikos mokslai*. 2024, 2305, 1-13. Prieiga per: doi: https://doi.org/10.48550/arXiv.2305.02637
20. JIN, Yiping. Towards Weakly-Supervised Hate Speech Classification Across Datasets. *Informatikos mokslai*. 2023, 2305, 1-10. Prieiga per: doi: https://doi.org/10.48550/arXiv.2305.02637
21. KOVÁCS, G. Challenges of Hate Speech Detection in Social Media. *Informatikos mokslai*. 2021, vol_.2, no. 2, pp. 95.
22. KOWSARI, K. Text Classification Algorithms: A Survey. *Informatikos mokslai*. 2019, vol._10, no. 4, pp. 150.
23. LUDWIG, F. Improving Generalization of Hate Speech Detection Systems to Novel Target Groups via Domain Adaptation. *Informatikos mokslai*. 2012, 29-39. Prieiga per: doi: https://doi.org/10.18653/v1/2022.woah-1.4
24. MACAVANEY, S. Hate speech detection: Challenges and solutions. *Informatikos mokslai*. 2019, vol_.14, no. 8, pp. 7-16.
25. MIENYE, I.D. Recurrent Neural Networks: A Comprehensive Review of Architectures, Variants, and Applications. *Informatikos mokslai*. 2024, vol_.15, no. 9, pp. 517.
26. MIKOLOV, Tomas. Efficient Estimation of Word Representations in Vector Space. *Informatikos mokslai*. 2013, 1-8. Prieiga per: doi: https://doi.org/10.48550/arXiv.1301.3781
27. NIXON, Jeremy. Measuring Calibration in Deep Learning. *Informatikos mokslai*. 2022, 1907, 1-10. Prieiga per: doi: https://doi.org/10.48550/arXiv.1904.01685
28. *Meta struggles with moderation in Hebrew, according to ex-employee and internal documents* [interaktyvus]. 2024 [žiūrėta 2025-05-10]. Prieiga per: https://www.theguardian.com/technology/article/2024/aug/15/meta-content-moderation-hebrew
29. PÉREZ, J.M. Assessing the impact of contextual information in hate speech detection. *Informatikos mokslai*. 2022, 1-10. Prieiga per: doi: https://doi.org/10.48550/arXiv.2210.00465
30. POLETTO, F. Resources and benchmark corpora for hate speech detection: a systematic review. In Language Resources and Evaluation. *Informatikos mokslai*. 2021. vol._55, no. 2, p. 477–523.
31. POWERS, D.M.W. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *Machine Learning*. 2010, 1-10. Prieiga per: doi: https://doi.org/10.48550/arXiv.2010.16061
32. QUINLAN, J.R. Induction of decision trees. *Informatikos mokslai*. 1986, BF00116251, 1-10. Prieiga per: doi: https://doi.org/10.1007/BF00116251
33. SALEH, Hind. Detection of Hate Speech using BERT and Hate Speech Word Embedding with Deep Model. *Informatikos mokslai*. 2021, 2111, 1-6. Prieiga per: doi: https://doi.org/10.48550/arXiv.2111.01515
34. *SIMON, S. Propaganda, Hate Speech, Violence: The Working Lives Of Facebook's Content Moderators* [interaktyvus]. 2019 [žiūrėta 2025-05-10]. Prieiga per: https://www.npr.org/2019/03/02/699663284/the-working-lives-of-facebooks-content-moderators
35. ZHANG, Xi. Character-level Convolutional Networks for Text Classification. *Informatikos mokslai*. 2016, 1501, 1-5. Prieiga per: doi: https://doi.org/10.48550/arXiv.1509.01626
36. *About the API* [interaktyvus]. 2025 [žiūrėta 2025-05-10]. Prieiga per: https://developers.perspectiveapi.com/s/about-the-api?language=en_US

37. *API reference* [interaktyvus]. 2025 [žiūrėta 2025-05-10]. Prieiga per: https://developer.chrome.com/docs/extensions/reference/api
38. *Community Standards Enforcement | Transparency Center* [interaktyvus]. 2021 [žiūrėta 2022-11-28]. Prieiga per: https://transparency.fb.com/data/community-standards-enforcement/hate-speech/facebook
39. *Extensions / Manifest V3* [interaktyvus]. 2025 [žiūrėta 2025-05-10]. Prieiga per: https://developer.chrome.com/docs/extensions/develop/migrate/what-is-mv3
40. *Hate Speech - expert.ai Platform NL Flow user manual* [interaktyvus]. 2025 [žiūrėta 2025-05-10]. Prieiga per: https://docs.expert.ai/platform/latest/nl-flow/reference/knowledge-models/hate-speech/#overview
41. *Hate Speech and Offensive Language Dataset* [interaktyvus]. 2020 [žiūrėta 2025-05-10]. Prieiga per: https://www.kaggle.com/datasets/mrmorj/hate-speech-and-offensive-language-dataset
42. *Hate speech review in the context of online social networks | Elsevier Enhanced Reader* [interaktyvus]. 2022 [žiūrėta 2022-11-28]. Prieiga per: https://reader.elsevier.com/reader/sd/pii/S1359178917301064?token=9CD4802933A3093528A0E9146267178C37E4D393D40E1C0E304CD4CE146E44520D8DB7E3BD15E36EB227A0EB393F504C&originRegion=eu-west-1&originCreation=20221128182556
43. *HTML5 - MDN Web Docs Glossary: Definitions of Web-related terms | MDN* [interaktyvus]. 2025 [žiūrėta 2025-05-10]. Prieiga per: https://developer.mozilla.org/en-US/docs/Glossary/HTML5
44. *JavaScript | MDN* [interaktyvus]. 2025. [žiūrėta 2025-05-10]. Prieiga per: https://developer.mozilla.org/en-US/docs/Web/JavaScript
45. *jigsaw-toxic-comment-classification-challenge* [interaktyvus]. 2021 [žiūrėta 2025-05-10]. Prieiga per: https://www.kaggle.com/datasets/julian3833/jigsaw-toxic-comment-classification-challenge
46. *Lithuanian stemming algorithm - Snowball* [interaktyvus]. 2018 [žiūrėta 2023-06-11]. Prieiga per: https://snowballstem.org/algorithms/lithuanian/stemmer.html
47. *Lithuanian Stop Words* [interaktyvus]. 2022 [žiūrėta 2025-04-13]. Prieiga per: https://docs.druidai.com/1385954/Content/Conversational%20AI/StopWords/LithuanianStopWords.htm
48. *Logistic Regression - an overview* [interaktyvus]. 2024 [žiūrėta 2025-05-10]. Prieiga per: https://www.sciencedirect.com/topics/computer-science/logistic-regression
49. *Multilayer Perceptron - an overview* [interaktyvus]. 2025 [žiūrėta 2025-05-10]. Prieiga per: https://www.sciencedirect.com/topics/computer-science/multilayer-perceptron
50. *NumPy* [interaktyvus]. 2025 [žiūrėta 2025-05-10]. Prieiga per: https://numpy.org
51. *Online Hate and Harassment: The American Experience 2021* [interaktyvus]. 2021 [žiūrėta 2022-11-28]. Prieiga per: https://www.adl.org/online-hate-2021
52. *Online Harassment* [Interaktyvus]. 2016 [žiūrėta 2022-11-28]. Prieiga per: https://www.datasociety.net/pubs/oh/Online_Harassment_2016.pdf
53. *pandas - Python Data Analysis Library* [interaktyvus]. 2025 [žiūrėta 2025-05-10]. Prieiga per: https://pandas.pydata.org/
54. *scikit-learn: machine learning in Python — scikit-learn 1.6.1 documentation* [interaktyvus]. 2025 [žiūrėta 2025-05-10]. Prieiga per: https://scikit-learn.org/stable/
55. *TensorFlow* [interaktyvus]. 2025 [žiūrėta 2025-05-10]. Prieiga per: https://www.tensorflow.org.
56. *Welcome to Flask - Flask Documentation (3.1.x)* [interaktyvus]. 2025 [žiūrėta 2025-05-10]. Prieiga per: https://flask.palletsprojects.com/en/stable/
57. *Welcome to Python.org* [interaktyvus]. 2025 [žiūrėta 2025-05-10]. Prieiga per: https://www.python.org