Kauno technologijos universitetas

Matematikos ir gamtos mokslų fakultetas

Giliuoju mokymusi grįstas diakritinių ženklų atstatymas lietuvių kalbai

Baigiamasis magistro projektas

Lukas Pakalniškis

Projekto autorius

Prof. dr. Evaldas Vaičiukynas

Vadovas

Prof. dr. Robertas Alzbutas

Vadovas

Kaunas, 2022

Kauno technologijos universitetas

Matematikos ir gamtos mokslų fakultetas

Giliuoju mokymusi grįstas diakritinių ženklų atstatymas lietuvių kalbai

Baigiamasis magistro studijų projektas

Didžiųjų verslo duomenų analitika (6213AX001)

Lukas Pakalniškis

Projekto autorius

Prof. dr.
Evaldas Vaičiukynas
Vadovas

Prof. dr.
Robertas Alzbutas
Vadovas

Dr.
Paulius Danėnas
Recenzentas

Doc. dr.
Beata Šeinauskienė
Recenzentė

Kaunas, 2022

Kauno technologijos universitetas

Matematikos ir gamtos mokslų fakultetas

Lukas Pakalniškis

Giliuoju mokymusi grįstas diakritinių ženklų atstatymas lietuvių kalbai

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.

Lukas Pakalniškis

Patvirtinta elektroniniu būdu

Pakalniškis, Lukas. Giliuoju mokymusi grįstas diakritinių ženklų atstatymas lietuvių kalbai. Magistro studijų baigiamasis projektas / vadovas Prof. dr. Evaldas Vaičiukynas; Kauno technologijos universitetas, Matematikos ir gamtos mokslų fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Taikomoji matematika.

Reikšminiai žodžiai: diakritinių ženklų atstatymas, sentimento analizė, gilusis mokymasis.

Kaunas, 2022. 73 p.

Santrauka

Internetinėje erdvėje vis daugėja tekstinių duomenų, tačiau dažnai vartotojai tekstą rašo netvarkingai. Lietuvių kalbos atveju, vienas dažniausiu netaisyklingo teksto pavyzdžių – diakritinių ženklų nenaudojimas. Netvarkingi tekstai dažnai gali įnešti triukšmo ir atliekant įvairias natūralios kalbos apdorojimo užduotis rezultatas nuo to gali nukentėti.

Šiame darbe tiriami giliuoju mokymusi grįsti diakritinių ženklų atstatymo modeliai. Sukurti 2 modeliai naudojant *Sequence to Sequence* (*Seq2Seq*) architektūrą (raidės atstatymo tikslumas – 98,12%) bei sureguliuojant jau prieš tai lietuvių kalbos rašybos taisymui apmokytą transformerio tipo *ByT5* modelį (raidės atstatymo tikslumas 99,65%). Šiais modeliais yra atstatomi diakritiniai ženklai turimam atsiliepimų duomenų rinkiniui ir naudojant standartiškai išvalytą tekstą, diakritinių ženklų atstatymui skirtais modeliais sugeneruotus tekstus, yra atliekamas sentimento klasifikavimas taip pat naudojant skirtingus teksto vektorizavimo dimensionalumus. Taip pat, yra išbandomi skirtingi sentimento klasifikavimo modeliai – logistinė regresija, atsitiktiniai miškai bei tas pats *ByT5* modelis, sureguliuotas sentimento klasifikavimo uždaviniui. Gauti rezultatai vertinami ir palyginami naudojant AUC įvertį. Geriausias rezultatas pasiektas naudojant sentimento klasifikavimui sureguliuotą *ByT5*, naudojant tuo pačiu modeliu sugeneruotą tekstą su diakritiniais ženklais (AUC įvertis – 0,975). Nei viename iš teksto vektorizavimo dimensionalumo ar skirtingų modelių bandymo atvejų, diakritinių ženklų atstatymas reikšmingo rezultatų pagerėjimo nedavė. Palyginus skirtingų sentimento klasifikavimo modelių tipų AUC įverčius tarpusavyje, *ByT5* sentimento klasifikavimo modelis davė reikšmingą rezultatų pagerėjimą (p reikšmė arti 0), lyginant su geriausią rezultatą pasiekusiu mašininio mokymosi modeliu.

Pakalniškis, Lukas. Deep learning based diacritic restoration for Lithuanian language. Master's Final Degree Project / supervisor Prof. dr. Evaldas Vaičiukynas; Faculty of Mathematics and Natural Sciences, Kaunas University of Technology.

Study field and area (study field group): Applied Mathematics.

Keywords: diacritics restoration, sentiment analysis, deep learning.

Vilnius, 2022. 73.

Summary

The amount of text data on the Internet is continuously increasing. However, some online users are making mistakes when writing text. In case of Lithuanian language, the main reason of text being not grammatically correct is not using diacritics. Noisy text can be a problem to achieve satisfying results for most of NLP tasks.

This thesis is focused to research deep learning based diacritics restoration methods for Lithuanian language. 2 models are created using *Sequence to Sequence* (*Seq2Seq*) model (character restoration accuracy – 98,12%) and transformer *ByT5* model (character restoration accuracy – 99,65%). By using these trained models diacritics are restored for customer review text data. Then sentiment classification is made by using clean text, and text with restored diacritics by these deep learning models. Also, different dimensionalities for text vectorization are tested. Logistic regression, random forest and *ByT5* fine-tuned models are used for sentiment classification sentiment classification. Results are compared by using AUC score. *ByT5* model fine-tuned for sentiment classification gave the highest AUC score (0,975). Diacritics restoration did not have any significant increase in sentiment classification using machine learning models and different dimensionalities. Even though, *ByT5* model for sentiment classification showed significant improvement when comparing with machine learning models (p-value almost 0).

Turinys

Lentelių sąrašas

Paveikslų sąrašas

Santrumpų sąrašas

ASCII – „Amerikietiškas informacijos mainų koduotės standartas“ (angl. *American Standard Code for Information Interchange*);

AUC – plotas po kreive (angl. *area under the curve*);

EER – lygių klaidų vertė (angl. *equal error rate*);

GRU – sulaikomo pasikartojančio vieneto tinklas (angl. *Gated recurrent unit*);

LDA – latentinis Dirichlė paskirstymas;

LSI – latentinis semantinis indeksavimas;

LSTM – ilgas laikinosios atminties tinklas (angl. *Long short-term memory*);

NLP – natūralios kalbos apdorojimas;

PR – preciziškumas-atkuriamumas (angl. *precision-recall*);

RNN – rekurentinis neuroninis tinklas (angl. *recurrent neural network*).

Įvadas

Internetas jau kurį laiką daugeliui yra tapęs neatskiriama gyvenimo dalimi. Didėjantis interneto naudojimas bendraujant, ieškant informacijos, apsiperkant, kuriant turinį ar ieškant pramoginio turinio, sukuria vis daugiau struktūrizuotų ar nestruktūrizuotų duomenų. Nemaža dalis tų duomenų yra įvairūs žmonių sugeneruoti tekstai. Tad mokslininkai turi pakankamai duomenų atlikti įvairias natūralios kalbos apdorojimo užduotis. Keletas iš populiariausių – automatinis vertimas, ilgų tekstų ar straipsnių santraukos, pokalbių robotai (angl. *chatbots*), sentimento analizė.

Tiesa, įvairių vartotojų rašomas tekstas dažnu atveju būna rašomas skubotai, todėl gali būti netvarkingas, su praleistomis ar sumaišytomis raidėmis (angl. *mistyping*) ar nenaudojamais diakritiniais ženklais. Diakritiniai ženklai šiuo atveju yra svarbūs, kadangi iš 36 Europos kalbų, tik anglų kalba neturi šių ženklų [1]. Lietuvių kalba šiuo atveju turi 9 diakritinius ženklus: *ą, č, ę, ė, į, š, ų, ū, ž*, kurie interneto vartotojų atitinkamai būna pakeičiami artimiausiomis ne diakritinėmis šių raidžių versijomis: *a, c, e, i, s, u, z*. Dėl įvairių priežasčių vartotojų nevartojami diakritiniai ženklai sukuria tekstą, kuris žmogui gali būti suprantamas, tačiau atliekant įvairias natūralios kalbos apdorojimo (NLP) užduotis, pavyzdžiui verčiant tekstą ar atliekant sentimento analizę, gali būti prarandamas tikslumas. Dėl šios priežasties kuriamas lietuvių kalbos giliuoju mokymusi grįstas diakritinių ženklų atstatymo modelis, kai pateikus originalų ir galimai netvarkingą, be diakritinių ženklų parašytą tekstą, šis modelis grąžintų tikslų tekstą jau su tinkamai atstatytais diakritiniais ženklais. Tokio įrankio panaudojimo atvejis bei veiksmingumas patikrinamas su sentimento analizės uždaviniu, kuomet jis atliekamas su standartiškai išvalytu, tačiau galimai netaisyklingu tekstu bei palyginamas sentimento klasifikavimo tikslumas atlikus šią analizę su jau sutvarkytu tekstu.

Sentimento analizė taip pat yra vis labiau populiarėjanti natūralios kalbos apdorojimo užduotis, kuri apibrėžiama kaip – kompiuterinės lingvistikos šakos dalis, kurioje vartotojų nuomonės, nusistatymai, emocijos išreikštos tekstu gali būti aptiktos, kategorizuojamos pagal norimą skalę ar temą [2]. Dažnu atveju verslo įmonės (ypač užsiimančios elektronine komercija, informacinėmis technologijomis, finansais), nori sužinoti apie savo kaip įmonės ar jos gaminamų bei parduodamų produktų, vykdomų komunikacijos kampanijų reputaciją, visuomenės ar vartotojų nuomonę bei nuotaikas. Taigi, atlikusios sentimento analizę įvairios kompanijos ar organizacijos gali aptikti prieš tai nežinomų verslo įžvalgų, stebėti vartotojų nuotaikas, matuoti vartotojų pasitenkinimo lygį ar prognozuoti pardavimus [2]. Sentimento analizė taip pat plačiai taikoma ir finansų, investicijų, verslo vystymo srityse. Tinkamai sentimento analizę įdiegus į šias sritis ar procesus, įmonės gali įgyti konkurencinį pranašumą, lyginant su tais, kuriuo šios užduoties savo procesuose netaiko.

Taigi, sentimento analizė tampa svarbi bei naudinga, įvertinant didėjančius vartotojų generuojamos tekstinės informacijos kiekius, kuriuos verslai ar organizacijos gali panaudoti savo paslaugų ar produktų vystymui. Tačiau, tokie surinkti tekstiniai duomenys gali būti triukšmingi ir sentimento analizės atveju neduoti tenkinamo rezultato. Ši problema bandoma išspręsti naudojant moderniausius giliuoju mokymusi grįstus metodus, kurių dėka kuriami įvairūs modeliai skirti teksto „triukšmo“ mažinimui. Tokio modelio gerumą galima įvertinti matuojant jo tikslumą ar kitus įverčius, tačiau realų veiksmingumą galima įvertinti atliekant kitą verslui svarbią natūralios kalbos apdorojimo užduotį – sentimento analizę.

Darbo problema – Internete vartotojai dažnai komentuoja, rašo atsiliepimus ar įvertinimus „triukšmingai“ – nesivadovauja bendrinėmis kalbos taisyklėmis, nepastebėdami praleidžia ar sukeičia

žodžio raides. Lietuvių kalboje didelė dalis „triukšmo“ atsiranda dėl nelotyniškų, lietuvių kalboje esančių raidžių (diakritinių simbolių), todėl toks tekstas kaip duomenys natūralios kalbos apdorojimo užduotyse gali pateikti mažiau tikslų rezultatą.

Darbo tikslas – Mašininiu bei giliuoju mokymusi grįstos sentimento analizės lietuvių kalbai pagerinimas, atstatant diakritinius ženklus netvarkingame tekste.

Darbo objektas – natūralios kalbos apdorojimo metodai.

Darbo uždaviniai:

1. Atlikti mokslinės literatūros apžvalgą, kurioje apžvelgiami automatiniai teksto koregavimo, diakritinių ženklų atstatymo modeliai bei natūralios kalbos apdorojimo taikymo pavyzdžiai lietuvių kalboje.
2. Pasirinkti modelius bei metodus diakritinių ženklų atstatymui.
3. Pasirinkti modelius bei metodus sentimento analizės uždaviniui.
4. Pasirinkti tekstynus diakritinių ženklų atstatymo uždaviniui. Perpanaudoti lietuvių kalbos atsiliepimų *evertink.lt* ir „Facebook“ platformose surinktą duomenų rinkinį sentimento detekcijos eksperimentams.
5. Sukurti kiek įmanoma tiksliausią modelį diakritinių ženklų atstatymui.
6. Palyginti sentimento analizės tikslumą naudojant netvarkytus duomenis bei diakritinių ženklų atstatymo dėka sutvarkytus duomenis.
7. Įžvalgos ir rekomendacijos.

1. Literatūros analizė

1.1. Automatinis teksto koregavimas

Automatinis teksto koregavimas yra puikiai žinoma bei plačiai nagrinėjama natūralios kalbos generavimo užduotis bei problema, kuri numato, kokie žodžiai tekstiniame dokumente yra netinkamai parašyti. Tokie žodžiai vartotojui gali būti pabraukiami ar kitaip pažymimi, taip pat pateikiami variantai tinkamiems žodžių pakeitimams [3]. Tai yra svarbu ne tik vartotojui dėl patogumo, teksto taisyklingumo bei suprantamumo, tačiau ir kitoms natūralios kalbos generavimo užduotims, pavyzdžiui, teksto santraukos kūrimui, sentimento analizei ar mašininiam vertimui [4] [5].

Viena populiariausių automatinio teksto koregavimo panaudojimo sričių – elektroninės komercijos platformų paieškos sistemos, kurių užklausose vartotojai dažnai padaro įvairių rašybos klaidų. Hasan‘as, Heger‘is ir Mansour‘as [6], naudodami statistinį mašininį vertimą, sukūrė modelį, atrandantį bei taisantį šias vartotojų klaidingai parašytas užklausas. Šiam modeliui autoriai panaudojo vartotojų paieškos tekstus bei prieš ir po paieškos atliekamus kitus veiksmus elektroninės komercijos platformoje. Daugiausia buvo renkami tokie paieškos tekstai po kurių vartotojas per trumpą laiką grįždavo į paiešką vėl bei nežymiai pataisydavo tekstą, tikėtina dėl netaisyklingai įrašyto teksto. Ištaisant tokias klaidas, pagerinama paieškos rezultatų kokybė ir sparta, kadangi tiksliau nustatoma vartotojo intencija bei pagreitinamas procesas. Taip pat, sumažinama protinė apkrova vartotojui reikiamos informacijos beieškant [7]. Tokiais būdais sukuriama gera patirtis gali skatinti vartotoją grįžti apsipirkti pakartotinai, tokią platformą parekomenduoti kitam ar apskritai palikti teigiamą atsiliepimą internetinėje erdvėje.

Be elektroninės komercijos platformų, klaidų tikrinimas bei taisymas taip pat yra naudojamas ir elektroninio pašto, failų laikymo debesijoje ar kitose paieškos sistemose. Šiuo atveju, bendrinis modelis yra nepakankamas, kadangi kiekvienas vartotojas gali pasižymėti skirtingu rašymo stiliumi ar leksikonu. Gupta ir kt. [8] pasiūlė klaidų tikrinimo ir taisymo procesą, kuris yra vykdomas personalizuotai, prisitaikant prie vartotojo žodyno keliomis skirtingomis kalbomis. Toks sprendimas statistiškai reikšmingai pagerino CTR (angl. *click-through rate*) – santykinę dalis paieškų, kurių rezultatai buvo paspaudžiami, MRR (angl. *mean reciprocal rank*) – metriką įvertinančią kelintas paieškos rezultatas buvo pasirenkamas, kuo aukštesnis įvertis, tuo vidutiniškai pirmesnis paieškos rezultatas yra pasirenkamas. Taipogi pagerinti buvo paieškos su esamu bent vienu rezultatu kiekis ir apskritai paieškos rezultatų kiekis, lyginant su globaliu klaidų taisymo sprendimu. Pagerinus šias metrikas vartotojui kuriama vertė yra greitesnis ir sklandesnis naudojimasis elektroniniu paštu, kas gali sukurti konkurencinį pranašumą, lyginant su kitomis tokią pačią paslaugą siūlančiomis platformomis.

Automatinis teksto koregavimas taip pat dažnai naudojamas kartu su įvairiais išmaniųjų telefonų ar kompiuterių klaviatūrų tobulinimais, kai pagal įvairius kontekstus teikiami tinkamų žodžių pasiūlymai, pradedant vesti žodį yra nustatoma koks tai žodis ir duodamas pasiūlymas jį užbaigti arba tekstas taisomas automatiškai. Tai reikšmingai paspartina teksto rašymą bei užtikrina rašomo teksto taisyklingumą [9] [10].

Didėjantis taikymas pastebimas ir medicinos srityje, kuri pasižymi tik jai būdingais terminais bei tekstynu. Dėl šios priežasties įprasti ir globalūs automatiniai teksto korektoriai gali tinkamai neveikti.

Tinkamam veikimui šioje srityje yra reikalingas išsamus medicinos žodynas [11]. Taip pat akademinėje literatūroje yra tiriama ir apie automatinio teksto koregavimo naudą žmonėms su skaitymo raidos sutrikimu (disleksija). Automatinis teksto koregavimas leidžia žmonėms su šiuo sutrikimu patobulinti teksto rašymą bei supratimą [12].

Automatinis teksto korektorius taip pat naudingas ir kuriant įvairias ranka parašyto teksto atpažinimo bei konvertavimo į skaitmeninį tekstą sistemas. Jis ypač yra aktualus apdorojant prastos kokybės, daug triukšmo turinčius ar kitaip sugadintus paveikslėlius su tekstu. Paveikslėlių atpažinimo sistemos dažnai nuo prastos kokybės paveikslėlių nuskaito tekstą netaisyklingai, todėl papildomai naudojant dirbtinių neuronų tinklų užkoduotojo-dekoduotojo (angl. *encoder-decoder*) tipo modelį iš paveikslėlio atpažintam ir sugeneruotam tekstui papildomai ištaisyti, paveikslėlių konvertuojamų į tekstą tikslumas reikšmingai padidėja [13].

Kalbant apie automatinį teksto koregavimą pagal kalbas, kaip ir daugelyje natūralios kalbos generavimo užduočių, daugiausiai tyrimų atlikta anglų kalboje [14]. Vienas žymesnių ir svarbesnių šiuolaikinių automatinio teksto koregavimo tyrimų atliktas Whitelaw‘o, Hutchinson‘o, Chung‘o ir Ellis‘o [3], kurie sukūrė klaidų bei n-gramų modelį, taisantį žmogaus padarytas rašybos klaidas. Taikant šį modelį, nereikia rankiniu būdų priskirtų taisyklingo teksto šaltinių, o visas lingvistinis žinynas automatiškai surenkamas iš interneto. Taip autoriai modelį galėjo pritaikyti ne tik anglų, bet ir kitoms kalboms (vokiečių, arabų, rusų). Universalų, įvairioms kalboms pritaikomą bei realiu laiku taikomą automatinį teksto koregavimo modelį pasiūlė Gupta [14].

Naudodami Levenšteino atstumo algoritmą bei palyginę su n-gramų algoritmo modeliu, Nejja ir Yousfi [15] arabų kalbai pritaikė automatinį teksto koregavimą su reagavimu į kontekstą. Naudojant gilųjį mokymąsi bei *Sequence to Sequence* (*Seq2Seq*) modelį, Etoori, Chinnakotla ir Mamidi [5] automatinį teksto koregavimą pritaikė indų kalboms (hindi, telugų). Su jų pasiūlytu metodu, nereikia didelių mokymo duomenų kiekio, nei lyginant su kitais giliojo mokymosi modeliais, Todėl, kaip teigia autoriai, jų modelis gali būti taikomas ir kitoms daug tekstynų ar kitų kalbos išteklių neturinčioms kalboms [5]. Be to, naudodamas bekontekstį bei su kontekstu susijusį *Seq2Seq* modelius, Buyuk‘as [16], pasiūlė didelio tikslumo automatinius teksto korektorius turkų kalbai.

Taigi, automatinis teksto koregavimas yra plačiai nagrinėjama užduotis natūralios kalbos generavimo tyrimų srityje. Automatiniai teksto korektoriai yra naudojami elektroninės komercijos platformose, elektroninio pašto platformose, mobiliuose ir stacionariuose įrenginiuose bei tai stipriai prisideda prie tokių platformų ar įrenginių naudojimo pasitenkinimo ir spartos vartotojui. Dažnu atveju čia yra taikomi personalizuoti bei realiu laiku veikiantys teksto tikrinimo ir koregavimo metodai, kas dar labiau pagerina šio uždavinio rezultato tikslumą. Taip pat, automatizuotas teksto koregavimas yra pravartus ir atskiroms sritims, tokioms kaip medicina, kuri turi tik sau specifinį žodyną. Nors daugiausia tyrimų ir sprendimų šioje NLP taikymo srityje yra padaryta anglų kalbai, kuo toliau, tuo daugiau atsiranda taikymų ir kitoms kalboms. Taip pat, laikui bėgant, atsiranda vis naujų ir tiksliau bei greičiau veikiančių automatinio klaidų tikrinimo ir teksto koregavimo modelių bei metodų.

Žemiau lentelėje pateikiama keletas skirtingų metodų taikymo pavyzdžių automatinei teksto koregavimo užduočiai įvairiomis kalboms bei nurodomas jai pasiektas tikslumas:

1 lentelė. Automatinio teksto koregavimo literatūros pagrindinių rezultatų suvestinė

Autorius	**Taikoma kalba**	**Algoritmas/metodas/modelis**	**Tikslumas (angl. *accuracy*) arba kitas pateikiamas įvertis**
Whitelaw, Hutchinson, Chung ir Ellis (2009) [3]	Anglų	n-gramų	3,8% – absoliutus klaidos įvertis (angl. *total error rate*)
Gupta (2020) [14]	Anglų, ispanų, prancūzų, vokiečių, olandų ir kitos 19 kalbų	Trie Data Structure, Burkhard-Keller Tree (BK Tree), Directed Acyclic Word Graphs (DAWGs), Symmetric Delete Algorithm (SDA)	98,5% – anglų kalbai, 98,6% – ispanų kalbai, 98,3% – prancūzų kalbai, 97,4% – vokiečių kalbai, 95,8% – olandų kalbai
Nejja, Yousfi (2017) [15]	Arabų	n-gramų, Levenšteino atstumas	93,0%
Etoori, Chinnakotle ir Mamidi (2018) [5]	Indų kalbos – hindi, telugų	*Seq2Seq*	85,4% – hindi kalbai, 89,3% – telugų kalbai,
Buyuk (2020) [16]	Turkų	*Seq2Seq*	92,1%

1.2. Diakritinių ženklų atstatymas

Prieš tai apžvelgtas automatinis teksto koregavimas ir jo taikymo sritys, dažniausiai sprendžia ortografines ar rašymo (spausdinimo) klaidas ir jų aptikimo bei taisymo uždavinį. Vis dėlto yra ir kita klaidų kategorija, kuri nebūdinga NLP srityje plačiausiai taikomai anglų kalbai – diakritinių ženklų atstatymas [17]. Diakritinių ženklų atstatymas – tai dažniausiai lotynų abėcėlės raidžių su papildomais ženkleliais (brūkšneliais, kabliukais, taškeliais) įdėjimas į tekstą ten, kur jų trūksta, užduotis [18].

Diakritiniai ženklai paplitę daugelyje slavų, skandinavų kalbų. Taip pat jų yra ir arabų ar vokiečių bei ispanų kalbose. Viena iš pagrindinių priežasčių, kodėl su šia problema susiduriama, yra tai, kad daugumai vartotojų, rašant elektroninius laiškus, diskutuojant forumuose, ar paliekant atsiliepimą apie produktą ar įmonę, paprasčiau ir greičiau rašyti be diakritinių ženklų (naudojant standartinę lotyniškų raidžių klaviatūrą (ASCII koduotę), neperjungiant papildomų savo kalbos raidžių klaviatūros). Be to, kartais yra susiduriama su užkodavimo problema, kai tekstiniai failai nenuskaito diakritinių ženklų [17]. Problema nerašant taisyklingai su diakritiniais ženklais arba naudojant turimą tekstą be jų išryškėja kai žodis su diakritiniais ženklais ir be gali įgauti visai kitą reikšmę. Lietuvių kalboje tokie pavyzdžiais galėtų būti šios žodžių poros: „karstas“ ir „karštas“, „sūnelis“ ir „šunelis“, „lupa“ ir „lūpa“. Todėl taisyklingai sudėlioti diakritiniai ženklai yra svarbūs ne tik šnekamojoje ar rašomojoje kalboje, tačiau ir įvairiuose NLP srities algoritmams, kuriems svarbus teksto kontekstas bei žodžių tikroji reikšmė.

Yra išskiriami du pagrindiniai diakritinių ženklų atstatymo būdai: žodžiais grįstas (angl. *word based*) ir žodžio raidėmis grįstas (angl. *character based*) [18]. Žodžiais grįstas būdas reikalauja didelės apimties žodyno bei statistinių kalbos modelių, todėl šiuo būdu procesas dažniausiai kuriamas tik vienai konkrečiai kalbai. Šis būdas yra pranašesnis kalboms, kuriose diakritiniai ženklai keičia gramatinę ar semantinę žodžio reikšmę [19]. Kitas būdas – grįsta raidėmis, kuris naudoja nuo kalbos nepriklausančius algoritmus. Tokie algoritmai dažniausiai naudoja statistinę informaciją iš mokymosi

imties duomenų. Šis būdas yra paprastesnis, greitesnis bei lengviau įgyvendinamas ir nereikalauja konkrečiai kalbai būdingų metodų ar algoritmų bei labai didelio tekstinių duomenų kiekio. Taigi, renkantis tinkamą būdą, reikia įvertinti diakritinių ženklų vaidmenį kalboje, mokymosi imties duomenų prieinamumą, duomenų apdorojimo kiekį, greitį bei kuriamo sprendimo ar sistemos vartotojų lūkesčius bei poreikius [17].

1.2.1. Diakritinių ženklų atstatymas Europos kalboms

Diakritinių ženklų turinčių kalbų yra gausu Europoje, todėl yra atlikta įvairių tyrimų Europoje vartojamoms kalboms. Kroatų kalbai buvo taikytas žodyno bei bigramų modelis [17]. Čekų kalbai, naudojant įvairius statistinius modelius, sukurtas automatinio teksto korektorius kartu su diakritinių ženklų atstatymu [20]. Vengrų kalbai sukurtas automatinis diakritinių ženklų atstatymas naudojant statistiniu mašininiu vertimu grįstą modelį [21]. Slovakų kalbos diakritiniams ženklas atstatyti naudoti rekurentinis dirbtinis neuronų tinklas (angl. *recurrent neural network*) su LSTM sluoksniais [22]. Serbų kalbai naudotas žodynų bei taisyklių (angl. *rule-based*) diakritinių ženklų atstatymas [23]. Rumunų kalbos diakritiniams ženklas atstatyti panaudotas sąsūkų dirbtinių neuronų tinklas [24] bei dviejų sluoksnių BiLSTM dirbtinių neuronų tinklo modelis [25].

Taip pat yra tyrimų, kuriuose kuriamas diakritinių ženklų atstatymo modelis naudojant kelias skirtingas kalbas. Be to, kurdami diakritinių ženklų atstatymo transfomerio metodu grįstą modelį, Laki ir Yang‘as [26] kartu su kitomis kalbomis (bosnių, čekų, estų, vengrų, kroatų, latvių, lenkų, rumunų, slovakų, slovėnų, albanų ir serbų), ištestavo ir lietuvių kalbą bei pasiekė 97,70% žodžio lygmens tikslumą. Verta pažymėti tai, kad autoriai dar išbandė modelį apmokytą su visų kalbų tekstiniais duomenimis. Tai lietuvių kalbos, kaip ir kai kurių kitų kalbų tikslumą pagerino (iki 97,91%). Autoriai kelia klausimą, kad galimai modelis pagerėja dėl to, kad tam tikrą duomenų kiekį ir iš jų apmokytą neuroninio tinklo informaciją bei svorius perima ir iš kitų kalbų [26]. Kiti tyrėjai Naplava ir kt. [27] rekurentinių dirbtinių neuronų tinklų modelį pritaikė latvių, lenkų, airių, prancūzų, turkų, ispanų kalboms. Po to, šie autoriai naudodami transformerio tipo *BERT* modelį diakritinių ženklų atstatymą pritaikė 12 kalbų (čekų, vietnamiečių, rumunų, latvių, lenkų, slovakų, airių, vengrų, prancūzų, turkų, ispanų ir kroatų) ir daugumai kalbų buvo pasiektas aukštas, didesnis nei 99% žodžio lygmens tikslumas [28]. Autoriai teigia, kad transformerio tipo *BERT* modelis yra pranašesnis nei prieš tai taikyti moderniausi metodai ir šiuo metu transformerio tipo modeliai yra geriausią rezultatą diakritinių ženklų atstatymo užduotyse rodantys modeliai.

Galiausiai verta panagrinėti, kas diakritinių ženklų atstatymo klausimo buvo nagrinėta bei daryta lietuvių kalbai. Be prieš tai minėto Laki ir Yang‘o [26] lietuvių kalbos įtraukimo į kuriamą modelį, pasitelkę raidžių lygmens grįstą mašininio mokymųsi grįstą, sąlyginio atsitiktinio lauko (angl. *conditional random field (CRF)*) modelį, Kapočiūtė-Dzikienė ir kt. [1] pasiekė 99,5% ženklų lygmens tikslumą bei 98,4% žodžių lygmens tikslumą. Stankevičius ir kt. [29], naudodami transformerio tipo baito lygmens *ByT5* modelį sukūrė tiek diakritinių ženklų atstatymo, tiek rašybos klaidų taisymo modelį lietuvių, kroatų, čekų, prancūzų, vengrų, airių, latvių, lenkų, rumunų, slovakų, ispanų, turkų, vietnamiečių kalboms. Autoriai pasiekė vidutinį 98,30% diakritinių ženklų atstatymo tikslumą žodžio lygmeniui vertinant visas minėtas kalbas bei 98,95% tikslumą konkrečiai lietuvių kalbai. Taip pat autoriai nurodo pasiektą žodžio lygmens tikslumą tuo pačiu metu atstatinėjant diakritinius ženklus bei rašybos klaidas – 94,60% vidutinis visoms kalboms ir 96,73% tikslumas konkrečiai lietuvių

kalbai. Kadangi eksperimentai buvo daromi su keliomis kalbomis, autoriai dar pateikia išvadą, kad kuo daugiau kalba turi žodžių su diakritiniais ženklais, tuo sunkiau yra pasiekti aukštesnį tikslumą.

Vertinant diakritinių ženklų atstatymo užduočiai darytus tyrimus, galima teigti, kad yra labai daug įvairių modelių šiai užduočiai atlikti. Modelių taikymas evoliucionuoja nuo pirminių statistinių, n-gramų modelių iki dirbtinių neuronų tinklų naudojant *Seq2Seq* architektūrą. Pagal Klishinsky, Karpik‘o ir Bodarenko [30] nėra vieno geriausio dirbtinių neuroninių tinklų tipo, kuris diakritinių ženklų atstatymo užduotyje tiktų ir veiktų išskirtinai gerai visoms naudojamoms kalboms. Skirtumai atsiranda dėl skirtingų kiekvienos kalbos teksto išteklių bei turimų duomenų. Duomenų rinkinyje pilną žodyną turinčioms kalboms diakritinių ženklų atstatymo užduotis gali būti formuluojama kaip žodžio paieška iš žodyno ir diakritinių ženklų atstatymas tokiu būdu. Kitu atveju, galimos situacijos kai atliekant šią užduotį yra susiduriama su žodžiais, kurių žodyne apmokant modelį nebuvo ir tokiose situacijose reikalingi kiti metodų ar modelių sprendimai. Dėl šių skirtumų sunku rasti vieną sprendimą, tinkantį visoms kalboms [30]. Vis dėlto, iš apžvelgtų tyrimų galima susidaryti įspūdį, kad šiuo metu vis labiau populiarėjantys yra transformerio tipo modeliai su kuriais pasiekiami rezultatai lenkia prieš tai diakritinių ženklų atstatymo užduotyje pirmavusių rekurentinių dirbtinių neuronų tinklų modelius, *Seq2Seq* architektūros modelius ar kitus mašininio mokymosi algoritmus.

Apibendrinant, daugumai kalbų tiek senesni, tiek tradiciniai metodai leisdavo pasiekti gana aukštą tikslumą atkuriant diakritinius ženklus. Tiesa, besivystant technologijoms bei populiarėjant dirbtinių neuronų tinklų, transformerio giliojo mokymosi metodams, tikslumas daugeliui kalbų yra vis pagerinamas. Žemiau lentelėje pateikiama apžvelgtų straipsnių nagrinėjant diakritinių ženklų atstatymo užduotį pagrinde Europos kalboms:

2 lentelė. Diakritinių ženklų atstatymo literatūros pagrindinių rezultatų suvestinė

Autorius	**Taikoma kalba**	**Algoritmas/metodas/modelis**	**Tikslumas (Accuracy)**
Santic, Snajder, Dalbelo ir Basic (2009) [17]	Kroatų	Žodyno ir bi-gramos	98,4%
Novak ir Sirklosi (2015) [21]	Vengrų	Statistinio mašininio vertimo (SMT)	99,1%
Kapočiūtė-Dzikienė, Davidsonas ir Vidugirienė (2017) [1]	Lietuvių	Tri-gramos CRF	99,5% – ženklų lygmens; 98,4% – žodžio lygmens
Hucko ir Lacko (2018) [22]	Slovakų	Rekurentinių dirbtinių neuronų tinklo (RNN) su LSTM	97,0%
Naplava, Straka, Stranak ir Hajic (2018) [27]	Vietnamiečių, rumunų, latvių, čekų, lenkų, slovakų, airių, vengrų, prancūzų, turkų, ispanų, kroatų	Rekurentinių dirbtinių neuronų tinklo (RNN) su LSTM	97,7% – vietnamiečių kalbai, 98,4% – rumunų kalbai, 97,5% – latvių kalbai, 99,1% – čekų kalbai, 99,6% – lenkų kalbai, 99,1% – slovakų kalbai, 98,7% – airių kalbai, 99,3% – vengrų kalbai, 99,7% – prancūzų kalbai, 99,3% – turkų kalbai,

			99,7% – ispanų kalbai, 99,7% – kroatų kalbai
Iordache, Georgescu, Oneata ir Cucu (2019) [25]	Rumunų	Rekurentinių dirbtinių neuronų tinklo (RNN) su LSTM	99,5%
Nutu, Lorincz ir Stan (2019) [24]	Rumunų	Sąsūkų dirbtinių neuronų tinklo (CNN)	97%
Laki ir Yang (2020) [26]	Estų, slovėnų, vengrų, latvių, serbų, čekų, lenkų, vengrų, slovakų, juodkalniečių, lietuvių, bulgarų, rumunų, albanų	Transformerio	99,8% – estų kalbai, 99,8% – slovėnų kalbai, 99,6% – vengrų kalbai, 99,5% – latvių kalbai, 99,3% – serbų kalbai, 99,2% – čekų kalbai, 99,2% – lenkų kalbai, 99,8% – kroatų kalbai, 98,7% – slovakų kalbai, 99,8% – juodkalniečių kalbai, 97,9% – lietuvių kalbai, 97,8% – bulgarų kalbai, 96,4% – albanų kalbai, 95,0% – rumunų kalbai
Naplava, Straka ir Strakova (2021) [28]	Čekų, vietnamiečių, latvių, lenkų, slovakų, prancūzų, airių, ispanų, kroatų, vengrų, turkų, rumunų	Transformerio – *BERT*	99,2% – čekų kalbai, 98,5% – vietnamiečių kalbai, 98,6% – latvių kalbai, 99,7% – lenkų kalbai, 99,3% – slovakų kalbai, 99,7% – prancūzų kalbai, 98,9% – airių kalbai, 99,6% – ispanų kalbai, 99,7% – kroatų kalbai, 99,4% – vengrų kalbai, 98,9% – turkų kalbai, 98,6% – rumunų kalbai
Stankevičius, Lukoševičius, Kapočiūtė-Dzikienė, Briedienė ir Krilavičius (2022) [29]	Kroatų, čekų, prancūzų, vengrų, airių, latvių, lietuvių, lenkų, rumunų, slovakų, ispanų, turkų, vietnamiečių	Transformerio – *ByT5*	99,4% – kroatų kalbai, 98,4% – čekų kalbai, 99,5% – prancūzų kalbai, 99,3% – vengrų kalbai, 98,4% – airių kalbai, 97,7% – latvių kalbai, 99,0% – lietuvių kalbai, 99,1% – lenkų kalbai, 98,2% – rumunų kalbai, 98,8% – slovakų kalbai, 99,4% – ispanų kalbai, 99,0% – turkų kalbai, 97,5% – vietnamiečių kalbai

1.2.2. Diakritinių ženklų atstatymas kitoms kalboms

Viena iš labiausiai akademinėje literatūroje nagrinėjamų diakritinių ženklų atstatymą naudojančių kalbų – arabų ar atskiriems šios kalbos dialektams, pavyzdžiui – Tuniso [31]. Šios kalbos diakritinių ženklų atstatymas išnagrinėtas taikant įvairiausius metodus: didžiausios entropijos modelį (angl. *maximum entropy model*) [32], n-gramų modelį su dinaminio programavimo algoritmu (angl. *dynamic programming algorithm*) didžiausios tikimybės sekai priskirti [33], paslėptąjį Markovo modelį (angl. *Hidden Markov Model (HMM))* [34], hibridinį modelį apjungus paslėptąjį Markovo modelį ir kitas teksto apdorojimo technikas bei statistinius modelius [35] ar dirbtinių neuronų tinklo su LSTM (angl. *Long short-term memory*) naudojant maksimalios entropijos jungtis (angl. *maximumo entropy (MaxEnt)*) modelį [36].

Tarp Rytų Azijos šalių, šios diakritinių ženklų atstatymo užduoties tyrinėjimai gana intensyviai vykdomi su vietnamiečių kalba. Šios kalbos diakritinių ženklų atstatymas buvo vykdomas naudojant taškinio įverčio metodą (angl. *pointwise approach)* [37], sąlyginių atsitiktinių laukų (angl. *conditional random fields (CFRs)*) bei atraminių vektorių (angl. *support vector machines (SVM)*) metodus [38]. Taipogi buvo naudojami ir sąsūkos dibtinių neuronų tinklus (angl. *convolutional neural network*) [39].

Vykdomi ir mažiau populiarių bei naudojamų ar net nykstančių kalbų diakritinių ženklų atstatymo tyrimai. Pavyzdžiui, maorių kalbai panaudotas algoritmas su naiviuoju Bajeso klasifikatoriumi (angl. *naive Bayes classifier*) [40], igbų kalbai naudotas n-gramų metodas [41], sindhų kalbai naudotas n-gramų bei atmintimi grįsto mokymo (angl. *memory-based learning*) metodas [42], jorubų kalbai naudotas *Seq2Seq* dirbtinių neuronų tinklas [43]. Pastarojoje kalboje, kaip ir prieš tai aptartose arabų ir vietnamiečių kalbose, naudojant dirbtinius neuronų tinklus, buvo pagreitinamas diakritinių ženklų atstatymo procesas, pasiekiant ir didesnį ženklų atstatymo tikslumą [44].

Taigi, diakritinių ženklų atstatymo užduotis yra gana plačiai ištyrinėta tiek Europos kalboms, tiek ir kitoms kalboms visame pasaulyje. Galima įžvelgti tendenciją, kad kuo toliau, tuo labiau yra naudojami giliojo mokymosi metodai, ar tai būtų *Seq2Seq* ar dar modernesni transformerio tipo modeliai, šios NLP užduoties modelių tikslumas yra vis geresnis. Taipogi, nors literatūroje ir nepavyko rasti konkrečių tyrimų dėl diakritinių ženklų atstatymo naudos verslui ar jų siūlomiems technologijų produktams, tačiau galima daryti prielaidą, kad kaip ir automatinio teksto koregavimo atveju, nauda elektroninei komercijai, elektroniniams paštams, failų laikymo ar kitų sistemų paieškos funkcijos pagerinimui gali būti pastebima.

1.3. Sequence to Sequence (*Seq2Seq*) modelio pritaikymo sritys

Jau kurį laiką natūralios kalbos automatinio vertimo (angl. *machine translation*), automatinio teksto koregavimo ar diakritinių ženklų atstatymo užduotyse dominuoja dirbtiniai neuronų tinklai, o ypač rekurentiniai dirbtinių neuronų tinklai (RNN) su užkoduotuoju – dekoduotuoju (angl. *encoder-decoder*). Taip vadinamas Sequence to Sequence (*Seq2Seq*) modelio (liet. „sekos į seką“) prototipas buvo išgrynintas Cho ir kt. [45]. Dar labiau išgrynintą versiją, kuri dažnai ir laikoma *Seq2Seq* modelio pradžia, aprašė ir automatinio vertimo užduočiai anglų – prancūzų kalboms pritaikė Sutskever ir kt. [46]. Tiesa, pradinė *Seq2Seq* modelio versija gerai veikė tik trumpiems ar vidutinio ilgio sakiniams. Kartais kai pateikiamas sakinys yra labai ilgas, paprastam *Seq2Seq* modelis negali įsiminti visos duodamos informacijos sekos ir taip vertimo kokybė suprastėja. Tam Bahdanau ir kt. [47] toliau

gerino modelį bei pasiūlė dėmesio (angl. *attention*) mechanizmą, kuris padeda *Seq2Seq* tipo modeliams susitvarkyti su labai ilgais sakiniais, išsaugant informaciją ir „kreipiant į dėmesį“ į esminius to sakinio žodžius. Giliojo mokymosi ar natūralios kalbos generavimo bendruomenėse tai yra laikoma viena esminiu giliojo mokymosi idėjų, kurios dėka tolimesniame etape buvo sukurtas ir transformerio tipo modelis.

Seq2Seq modelio architektūra plačiai yra naudojama įvairioms natūralios kalbos generavimo užduotims. Dažniausiai, tai automatinis vertimas naudojant dirbtinius neuronų tinklus (angl. *neural machine translation (NMT)*), kuriam Crego ir kt. [48] pritaikė *Seq2Seq* architektūros modelį. Taipogi, Lambda ir Hsu [49] pritaikė *Seq2Seq* ir kitai NLP užduočiai – klausimų generavimui. Sekantis užduoties pavyzdys – pavadinto subjekto atpažinimas (angl. *named entity recognition*). Chen ir Moschitti [50] *Seq2Seq* modelį pritaikė subjektų atpažinimui, pavyzdžiui įmonių, bankų, politinių institucijų ar asmenybių vardų detekcija. Šioje užduotyje *Seq2Seq*, lyginant su kitais metodais, pasižymėjo aukštu tikslumu tiek sugebant išlaikyti informaciją ir atpažinti duomenyse matytų subjektų atvejis, tiek mokymo metu nematytų subjektų atpažinime.

Dar vienas su tekstine informacija susijęs *Seq2Seq* modelio pritaikymo ir panaudojimo atvejis – informacijos ištraukimas iš įvairaus tipo tekstinių dokumentų. Pires ir kt. [51] modelį įvertino kaip sėkmingai naudotiną metodą registracinės ar teisinės informacijos dokumentavime, apdorojime ir struktūrizavime. Autoriai taip pat pabrėžia, kad *Seq2Seq* modelio pranašumas, lyginant su prieš tai naudotais metodais, yra tai, kad užtenka naudoti vieną modelį nereikalaujantį didelių techninių resursų. Prieš tai nusistovėjusi praktika buvo linkusi naudoti atskirus modelius informacijos ištraukime bei normalizavime. Autoriai taip pat mini, kad *Seq2Seq* modelius yra paprasta pritaikyti siauriai tekstinių dokumentų tipo sričiai (šiuo atveju įvairūs registraciniai bei teisiniai dokumentai). Taigi, sureguliuojant (angl. *fine-tune*) *Seq2Seq* modelį reikiamai NLP užduočiai, pateikus atitinkamos srities duomenis, yra gan paprastai pritaikomas informacijos išgavimui, apdorojimui ir struktūrizavimui.

Kitas panaudojimo būdas, kuriame gan stipriai pasižymi *Seq2Seq* modelis – automatinis kalbos atpažinimas. Pradžioje modelis buvo išbandytas audio segmentų vektorizavimui [52]. Tačiau jau dabar šiuo modeliu kalbos atpažinimas gali būti prilyginamas žmogaus sugebėjimų lygiui atpažįstant kalbą ir sakomus žodžius [53]. Dėmesio mechanizmu grįstą *Seq2Seq* modelį, integruojant akustinius, tarimo bei kalbos modelius į vieną neuroninį tinklą pateikė Chiu ir kt. [54]. Vis dėlto, anot autorių esminės atpažinimo užduoties problemos – žodžio nebuvimo apmokymo žodyne *Seq2Seq* modelis iki galo vis dar neišsprendžia, nors tokio modelį naudojant specifinėje srityje iš kurios duomenų modelis buvo apmokytas, rezultatas yra tikslus ir tenkinantis [55]. Jiang ir Peollabauer [56], automatinę kalbos atpažinimo sistemą naudojant *Seq2Seq* modelį pritaikė medicinos sričiai. Anot autorių, automatinis kalbos atpažinimas gali pasitarnauti ligoninėse, pradedant nuo to, kad sistema veiktų bendraujant su pacientu ir gydytojui nereikėtų gaišti laiko rankiniam informacijos suvedinėjimui ar ieškojimui. Taipogi, pasak autorių, tai pasitarnauja netgi chirurginių operacijų metu, kuomet yra svarbu laikytis rekomenduojamų žingsnių, laikytis nustatyto operacijos protokolo ar kitų įprastų operacijos eigos praktikų. Sistema, atpažindama balso komandas, tokius nukrypimus galėtų fiksuoti ir įspėti.

Iš kitos pusės, *Seq2Seq* modelis yra naudojamas ne tik automatiniam balso atpažinimui, tačiau ir sakytinės kalbos generavimui ir jos konvertavimui. Zhang ir kt. [57] modelį pritaikė balso

konvertavimui kai turimas originalus kalbos signalas gali būtų konvertuojamas ir skambėti lyg būtų tariamas tikslinio kalbėtojo. Tokio pritaikymo potencialas gali būti nuo tiesiog pramoginės mobiliosios aplikacijos iki naujų garsinių duomenų generavimo ir šių duomenų panaudojimo gerinti personalizuotam kalbos generatoriui konvertuojant tekstą į sakytinę kalbą.

Seq2Seq modelis ar jo architektūra gali būti naudojama ne tik su balsu ir žodžiais susijusiai audio informacijai atpažinti bei apdoroti, tačiau ir garsinių failų konvertavimui į MIDI formatą ar kitą muzikinę transkripciją [58]. Muzikos srityje *Seq2Seq* modelis dar pasitarnauja ir kaip muzikos rankraščių atpažinimo ir konvertavimo į natas įrankis. Įdomu tai, kad modelis neblogai veikia atpažįstant ir ypatingai senus ir triukšmingus ar net žmogui sunkiai atpažįstamus rankraščius [59]. Naudojant *Seq2Seq* modelio architektūrą ir dar apjungiant sąsūkų dirbtinių neuroninų tinklus yra sėkmingai atliekama rankraščio teksto nuskaitymo iš paveikslėlių bei konvertavimo į tekstą užduotis [60]. *Seq2Seq* šiuo atveju padeda sąsūkų dirbtinių neuroninų tinklams kontekstualizuojant nuskaitomą informaciją, ypatingai tokiais atvejais kai yra sunku nuskaityti tekstą iš paveiksliuko dėl prastos kokybės. Tokiu atveju pasitelkiamas *Seq2Seq* modelio ištrauktas teksto kontekstas.

Verta paminėti ir tai, kad *Seq2Seq* modelis taipogi puikiai veikia ir su video duomenimis. Venugopalan ir kt. [61], naudojant *Seq2Seq* modelio idėją, sėkmingai atliko antraščių generavimo video kadrams užduotį. Tai padaryta asocijuojant video kadrų sekas su žodžiu sekomis. Tuo tarpu Xu ir kt. [62], sugebėjo *Seq2Seq* modelį panaudoti video segmentavimo atlikimui. Video objektų segmentavimas šiuo metu yra svarbus bei reikšmingas aspektas video turinio supratime bei gali būti panaudojamas daugelyje aplikacijų, pavyzdžiui įvairių objektų video atpažinime bei sekime, taipogi interaktyviame video redagavime [62].

Kaip jau minėta prieš tai, *Seq2Seq* modeliai originaliai buvo sukurti natūraliam kalbos apdorojimui (angl. *natural language processing*), tačiau, kaip ir galėtų sufleruoti modelio pavadinimas, šis modelis pasižymi tiksliais rezultatais ir su kitokio tipo sekomis. Biologijos, chemijos ar biochemijos srityse sekos yra randamos molekulėse, virusuose, baltymuose. Liu ir kt. [63], sėkmingai *Seq2Seq* pritaikė retrosintetinių reakcijų analizei bei prognozavime. Tai gali būti sėkmingai naudojama gaunant prieigą prie labai retų molekulių ar net niekada prieš tai neaptiktų molekulių atradimui. Tai taip pat prisideda prie įvairių naujų medžiagų atradimo, farmacijos srityje prie naujų vaistų sukūrimo ar įvairių atradimų aplinkosaugos mokslo srityje, [63]. Tuo tarpu Tang ir kt. [64], *Seq2Seq* modelį sėkmingai panaudojo baltymo sekos identifikavimui. Naudojant šį modelį, tai pat galima nustatyti netvarkingos baltymo sekos vietas. Šios iš esmės netvarkingos vietos sekoje yra svarbūs fiziologiniai procesai, kurie gali nulemti Alzhaimerio ligą, vėžį ar kitas sudėtingai aptinkamas ir diagnozuojamas ligas [64]. Iš kitos pusės, stabilių baltymų sekos inžinerija taip pat yra viena iš sričių, kurioje randama *Seq2Seq* modelio pritaikymo pavyzdžių. Kawano ir kt. [65] savo moksliniame darbe pritaikė *Seq2Seq* modelį baltymo sekos stabilumo pokyčiui prognozuoti. Stabilių baltymų inžinerija yra svarbi įvairiems biosensoriniams ar fermentiniams katalizatoriams. Toks modelio pritaikymas stabilaus baltymo prognozavime leistų lengviau kurti naujus baltymus [65]. Galiausiai, dar vienas svarbus aspektas, ypatingai COVID-19 pandemijos metu, yra viruso mutacijų sekimas bei prognozavimas. Berman ir kt. [66], panaudojo *Seq2Seq* modelį pilnos baltymo sekos generavimui su galimomis virusų mutacijomis. Tokiu atveju patogeno vystymosi prognozavimas ryškiai prisideda prie galimybės kontroliuoti, valdyti ar apskritai panaikinti viruso sukeltą ligą [66]. Taipogi, aptariamas modelis gali būti sėkmingai panaudojamas miego sutrikimų prognozavime. Mousavi ir kt. [67], naudodami giliojo mokymosi sąsūkos neuroninius tinklus apskaičiavo laike nekintamą funkciją bei dažnių informaciją,

o pridėjus *Seq2Seq* modelį, užfiksuojamos kompleksinės priklausomybės tarp miego stadijų bei miego kokybės ar sutrikimų. Taigi, galima vertinti, kad *Seq2Seq* modelis yra gana plačiai naudojamas ypatingai svarbiose mokslo srityse, kaip medicina, farmacija, genetika ar virusologija.

Panaudojamos energijos tvarumas ir ekologija dabar taip pat yra labai svarbios ir mokslinėje srityje plėtojamos temos. Šiuo atveju, užtikrinti tvarumo poreikius padeda efektyvus energetikos valdymas minimizuojant elektros ar kitų energijos šaltinių panaudojimą. Taigi, Marino ir kt. [68], panaudojo *Seq2Seq* modelį energijos apkrovos prognozavime. Energijos valdymas gali būti susijęs ir su tikslia bei efektyvia trumpojo periodo elektros apkrovos prognoze, kuri svarbi energijos sistemos patikimumui bei ekonomiškumui. Taigi, Li ir kt. [69], *Seq2Seq* modelį panaudojo prognozuojant elektros apkrovą laike. Prie ekologijos ir tvarumo gali prisidėti ir efektyvus ūkininkavimas bei dirvožemio kokybės priežiūra. Li ir kt. [70] *Seq2Seq* modelį panaudojo dirvožemio temperatūros bei drėgnumo faktorių prognozavimui. Būtent šie faktoriai turi didžiulę įtaką visai augmenijai bei klimatui. Naudojantis šiomis prognozėmis ūkininkai galėtų planuotis kokius augalus, kada ir kaip auginti, taip užtikrinant ūkininkavimo efektyvumą, ekologiją bei tvarumą.

Vystantis technologijoms yra svarbu užtikrinti jų kokybę bei prieinamumą. Tai yra ypatingai svarbu ir 5G technologijos vystymosi atveju. Taigi, Hu ir Han [71], *Seq2Seq* modelį panaudojo milimetrinių bangų nustatymui ir sekimui užtikrinant aukšto dažnio duomenų belaidei komunikacijai. Automobilių technologijų tobulinimui bei vystymui šis modelis gali pasitarnauti taip pat. Park ir kt. (2018) [72], pasiūlė giliuoju mokymusi grįstą automobilio trajektorijos nuspėjimo techniką, kuri galėtų sugeneruoti būsimos trajektorijos seką važiuojantiems ar aplink esantiems automobiliams. Tokia technologija galėtų prisidėti prie saugumo, išvengiant tragiškų automobilių avarijų, ypatingai esant nesaugioms sąlygoms žiemą ar lietaus metu. Įdomus panaudojimo būdas, kuris taipogi prisideda prie transporto kokybės gerinimo yra keleivių srautų prognozavimas didelėje metro sistemoje, kurį naudojant *Seq2Seq* modelį sugebėjo įgyvendinti Hao ir kt. [73]. *Seq2Seq* modelis šiuo atveju pasitarnavo apdorojant duomenis bei prognozuojant išlipančių keleivių skaičių kiekvienoje stotelėje artimiausiu metu. Modelis apmokytas jam pateikiant įlipančių kelių paskutinių periodų keleivių skaičių kiekvienoje metro stotyje. Šiuo atveju, *Seq2Seq* modelis ir jo rezultatas nukonkuravo laiko eilučių prognozavimo užduočiai tradiciškai naudojamus ARIMA, SVM ar kitus dirbtinių neuronų tinklų modelius. Taigi, modelio panaudojimai atvejai gan stipriai prisideda prie įvairių besivystančių technologijų tobulėjimo.

Kaip ir įvairiausi kiti modeliai bei metodai, taip ir *Seq2Seq* modelis yra bandomas naudoti akcijų ar kriptovaliutų kainų prognozavime. Tai pastaruoju metu viena iš didžiausio populiarumo bei susidomėjimo susilaukiančių sričių. Nenuostabu, kad tam yra bandomi įvairiausi tradiciniai laiko eilučių metodai, kaip ARIMA ar rekurentinių dirbtinių neuronų tinklų modeliai. Visgi, Rebane ir kt. [74], sėkmingai eksperimente panaudojo *Seq2Seq* prognozuojant bitkoino kainą, ypatingai per didesnio kainos stabilumo periodus, nugalint prieš tai įvardintą ARIMA modelį. Be to, Gao [75] naudojo *Seq2Seq* modelį akcijų kainų prognozavimo užduočiai.

Galiausiai, įdomus *Seq2Seq* modelio panaudojimo būdas – automatinis programinio kodo arba programinių klaidų (angl. *bugs*) taisymas, kurį panaudojo Chen ir kt. [76]. Sėkmingu tokio modelio panaudojimo atveju, programuotojai galėtų sparčiau rašyti programinį kodą ar jame aptikti klaidas.

Taigi, *Seq2Seq* modelis, pirmiausiai panaudotas automatinio vertimo tikslams, puikiai pasitarnauja ir gausybėje kitų verslo ar mokslo sričių dėl savo savybės apdoroti sekas. Pradedant nuo įvairiausių

NLP užduočių, tokių, kaip klausimų generavimas, informacijos ištraukimas. Modelis taip pat veikia garsinės kalbos atpažinime ar net garsinės kalbos generavime. Panaudojimo atvejai tęsiasi ne tik garso bet ir pritaikomumu video segmentavime. Ekologijos srityje prisidedama prie elektros tinklų optimizavimo ar net tvaresnio ūkininkavimo. Technologijų srityje tinkamas modelio panaudojimas gali prisidėti prie įvairiausių technologijų ir jų funkcijų tobulinimo. Galiausiai svarbios modelio panaudojimo sritys yra biologijos, chemijos, farmacijos, medicinos, finansų srityse. Iš to galima teigti, kad tai populiarus ir veiksmingas modelis, kuris gali būti pritaikomas aibėje sričių, kuriuose yra sekos tipo duomenų. Tad, duomenų kiekiams didėjant, tokio tipo modeliams ir jų pritaikymui vis dar atrandama naujų būdų.

1.4. Sentimento analizė ir jos taikymo sritys

Sentimento analizė apima nuomonių, sentimentų, emocijų ar nusistatymo išreiškimą tekstu tam tikro subjekto atveju [77]. Taip pat tai atlieka detekcijos, nuomonių išgavimo bei suklasifikavimo uždavinį pagal užduotą tekstinę įvestį. Sentimento analizė padeda stebėti visuomenės nuotaikas, politinius judėjimus, prognozuoti rinkimų rezultatus, sekti įvairių rinkų įžvalgas, matuoti klientų pasitenkinimą, nuspėti klientų atsiliepimus ar įverčius apie siūlomas paslaugas, prognozuoti pardavimus ar sekti ligų protrūkius, kas yra labai aktualu COVID-19 pandemijos kontekste [2] [78].

Taip pat sentimento analizė praverčia verslo srityje, pavyzdžiui tobulinant verslo strategijos įgyvendinimą, verslo teikiamų paslaugų ar produktų kokybę [78]. Be to, kaip vienas iš sentimento analizės pavyzdžių, gali būti pateikiamas, Saura [79] tyrimas, kur naudojant socialinio tinklo „Twitter“ žinutes buvo atliekama sentimento analizė identifikuojant esminius faktorius sėkmingam startuolio sukūrimui. Autorius taipogi identifikavo temas, kurios sukuria teigiamą, neigiamą bei neutralų sentimentą. Dar vienas pavyzdys, kur sentimento analizė gali būti panaudojama versle – Jain [80], tyrimas kur panaudojus socialinio tinklo „Twitter“ žinutes, sėkmingai prognozavimo filmų populiarumą.

Apžvelgiama dar viena sritis, kurioje sentimento analizė yra sparčiai naudojama – finansai. Šioje srityje dažniausiai sentimento analizė yra pasitelkiama prognozuoti kaip finansų rinka reaguos į su finansai susijusią tekstine informacija medijoje. Li ir kt. [81] savo tyrime atskleidė, kad, naudojant finansinių straipsnių tekstus ir juose esančią informaciją bei sentimento analizės modeliais ištraukus iš jų reikiamą informaciją, tai padeda pagerinti akcijų prognozavimo tikslumą. Kearney ir Liu [82] savo tyrime taip pat atskleidė, kad teksto sentimentas gali turėti stiprią įtaką akcijų grąžoms bei akcijų prekybos apimtims finansų rinkoje.

Taigi, įmonių ar pavienių asmenų atliekama sentimento analizė gali sukurti pridėtinės vertės verslo srityje, pavyzdžiui analizuojant klientų poreikius ir pasitenkinimą teikiamomis paslaugomis, identifikuojant sėkmingo verslo ir jo strategijos faktorius ar prognozuojant teikiamų paslaugų ar prekių populiarumą, kas galimai prisidėtų prie tikslesnės jų pardavimo prognozės. Finansų srityje sentimento analizė taip pat gali būti naudinga prognozuojant akcijų grąžas ar leisti įvertinti akcijų prekybos apimtis rinkoje.

1.4.1. Sentimento analizė naudojant transformerio tipo modelius

2017 metais pirmą kartą pristatyta transformerio tipo modelio architektūra, kuri yra pastatyta ant jau prieš tai apžvelgtos *Seq2Seq* principo naudoti užkoduotoją bei dekoduotoją bei *Seq2Seq* naudojamo

dėmesio mechanizmo [83]. Transformerio modelis yra plačiai naudojamas automatinio vertimo, automatinio teksto generavimo, diakritinių ženklų atstatymo bei kituose NLP uždaviniuose. Kaip jau buvo galima pastebėti, diakritinių ženklų atstatymo literatūros apžvalgoje, daug kur transformeriai lenkia prieš tai dominavusią *Seq2Seq* modelio architektūrą. Vis dėlto, kaip ir pastarojo modelio atveju, taip ir transformerio pritaikymo sritys yra gana gausios, o viena iš tokių sričių, kurioje transformerio tipo modeliai yra dominuojantys savo tikslumu – sentimento analizės užduotys, ištraukiant reikiamą informaciją ar atliekant sentimento klasifikaciją.

Delbrouck ir kt. [84] savo tyrime naudodami transformerio modelio architektūrą, atliko emocijų atpažinimo bei sentimento analizės užduotį, naudojant tiek tekstinę, tiek garsinę informaciją. Jiang ir kt. [85] naudojant transformeriu grįstą atminties tinklą (angl. *memory network*), sukūrė modelį, kuris iš internetinių komentarų socialinės medijos platformoje gali ištraukti su emocijomis susijusią informaciją. Myagmar, Li ir Kimura [86], naudodami *BERT* ir *XLNet* transformerio modelius sentimento klasifikavimo skirtingose temose užduočiai aptiko, kad šie modeliai savo tikslumu stipriai lenkia prieš tai naudotus ne transformerio tipo modelius. Tiesa, tarpusavy lyginant abu transformerio modelius, šiame tyrime tikslesnį rezultatą parodė *XLNet* modelis. Šis modelis pranašesnis už ne transformerio tipo modelius yra ir tuo, kad jam apmokyti užteko žymiai mažesnio kiekio duomenų. Taigi, transformerio tipo modeliai apžvelgtoje literatūroje yra dominuojantys sentimento analizės bei sentimento klasifikavimo užduotyse.

Viena iš sričių, kurioje pastebimas transformerio modelių gausus panaudojimas sentimento analizei – finansų sritis. Araci [87] naudojant *BERT* modelį, pristatė finansų sričiai pritaikytą jo versiją – *FinBERT*, kuris gali būti naudojamas finansų srities NLP uždaviniams atlikti. Modelis sukurtas sureguliuojant pirminį *BERT* modelį su finansų srities tekstynų, finansų sentimento klasifikavimo užduotyje aplenkė taip pat testuotus kitus mašininio ir giliojo mokymosi metodus. Heidari ir Rafatirad [88], naudojant *BERT* transformerio modelį, prognozavo pelningą nekilnojamą turtą remiantis įvairiais tekstinės informacijos šaltiniais. Toks modelis pasitarnauja ir sukuria lygias galimybes žmonėms, kurie neturi aukštų pajamų ar apskritai yra praradę darbą dėl COVID-19 pandemijos ar kitų priežasčių bei neturi investavimo patirties nekilnojamo turto srityje. Modelis veikia su *BERT* atliekant sentimento klasifikavimo uždavinį naudojant atsiliepimų apie nuomojamą nekilnojamąjį turtą tekstus. Stevenson ir kt. [89], taipogi naudodamas transformerio *BERT* modelį jį pritaikė sentimento išgavimo užduočiai nustatant kreditingumo riziką įvairių dydžių įmonėms bankinei paskolai gauti.

Transformerio tipo modeliai yra plačiai panaudojami sentimento analizės ar jo klasifikavimo uždaviniams, aptinkant emocijos sentimentą socialinių medijų platformose talpinamuose įrašuose. Taip pat šiame poskiryje apžvelgiami keletas pavyzdžių, kuomet šio tipo modeliai yra panaudojami finansų srityje ir šios srities sentimento analizėje. Iš apžvelgtų pavyzdžių, galima apibendrinti, kad naudojant tokius modelius pavieniai asmenys ar įmonės gali gauti papildomą naudą prognozuojant nekilnojamoj turto rinką ar nustatinėjant kreditingumo riziką bankinei paskolai gauti.

1.5. Natūralios kalbos generavimo taikymai lietuvių kalboje

Prieš tai aptartuose pavyzdžiuose galima įžvelgti, kad anglų ar kitose kalbose sentimento analizės taikymo srityje yra padaryta labai daug, šiame darbe yra aktualus ir sentimento klasifikavimo progresas lietuvių kalboje.

Vienas pirmųjų sentimento analizės bandymų buvo atliktas Kapočiūtės-Dzikienės ir kt. [90]. Šiame darbe autoriai naudojo įvairius tiek žiniomis grįstus, tiek mašininio mokymosi metodus. Vis dėl to geriausią rezultatą parodė mašininio mokymosi metodai: atraminių vektorių klasifikatorius (angl. *support vector machine*) bei Naivusis Bajeso daugianario metodas (angl. *Naive Bayes Multinomial*). Geriausias rezultatas ir pasiektas su pastaruoju, tikslumas – 67,9%. Dzikienė, Damaševičius, Wozniak (2018) ir toliau tobulino sentimento analizės lietuvių kalboje uždavinį pritaikant giliojo mokymosi metodikas: dirbtinių neuronų tinklą su LSTM (angl. *Long short-term memory*) sluoksniais, bei sąsūkos dirbtinių neuronų tinklu (CNN). Nors ir tradiciniai mašininio mokymosi metodai, tokie kaip atraminių vektorių klasifikatorius (SVM) ar Naivusis Bajeso daugianaris pasiekė aukštesnį tikslumą (73,5%), tačiau giliojo mokymosi metodai įgavo pranašumą, kuomet apmokymui buvo naudojamas mažesnis duomenų rinkinys. Sentimento detekcijos uždavinyje lietuvių kalboje yra bandymų pritaikyti analizę ir konkrečiai temai ar sričiai. Štrimaitis ir kt. [91], naudojant prižiūrimojo mašininio mokymosi modelius, atliko sentimento analizę finansų naujienoms iš lietuviškų interneto puslapių. Geriausiai šiai užduočiai pasitarnavo ir aukščiausią tikslumą parodė Naivusis Bajeso algoritmas (71,1%), nors ne ką prasčiau pasirodė ir dirbtinių neuronų tinklas su LSTM sluoksniais (71%) bei atraminių vektorių mašininis modelis (70,4%).

Taipogi, sentimento analizę lietuvių kalbai savo magistriniuose darbuose nagrinėjo Daugėla [92] ir Morkūnaitė [93]. Naudojant internetinius atsiliepimus iš evertink.lt puslapio bei jų įvertinimus Daugėla [92] sentimento klasifikavimo užduotį atliko naudojant SVM bei giliojo mokymosi modelius, kurie iš esmės pasiekė panašius rezultatus. Tuo tarpu, Morkūnaitė [93] šį darbą papildė panaudojant ir atsiliepimus iš socialinės medijos „Facebook“. Naudojant SVM ir XGBoost kombinacinį modelį, pagerino detektoriaus tikslumą iki 91,36 proc. bei 0,972 AUC įverčio. Autorė taipogi išbandė logistinės regresijos, atsitiktinių miškų metodus bei skirtingus teksto vektorizavimo dimensionalumus. Įdomu tai, kad lemavimo ar žodžio dalies afikso pašalinimas didelės įtakos neturėjo bei reikšmingai tikslumo nepagerino. Šiame tyrime yra kuriamas šių magistrinių darbų tęstinumas ir atsižvelgiant į pateiktass rekomendacijas yra kuriamas modelis, kurio prielaida, kad diakritinių ženklų atstatymas lietuvių kalbai bei netaisyklingo teksto sutvarkymas turėtų pagerinti šiuose darbuose naudotų duomenų sentimento analizės tikslumą. Taipogi šiame darbe yra naudojami autorių surinktas atsiliepimų teksto duomenų rinkinys su priskirtu sentimento požymiu.

Taip pat verta paminėti ir kitus natūralios kalbos generavimo taikymus lietuvių kalboje ir be sentimento analizės. Alumae ir Tilk [94] aprašė automatinės kalbos atpažinimo įrankio kūrimą, skirtą garso transliacijų transkripcijų generavimui. Autoriai naudodami 84 valandų rankiniu būdu transkribuotos audio informacijos bei virš 400 mln. žodžių tekstinės informacijos, pasiekė 14,7% klaidos įvertį žodžiui. Dzikienė [95], naudojant skirtingus *Seq2Seq* modelius, kūrė lietuvių bei kitoms morfologiškai kompleksiškai kalboms pritaikytą pokalbių robotą (angl. *chatbot*). Nors ir nepasiektas aukštas tikslumas (lyginant su anglų kalba), tačiau anot autorės tai yra pirmasis toks bandymas pritaikyti pokalbių robotą morfologiškai kompleksiškai kalbai. Pipiras [96] savo magistriniame darbe taipogi naudojo įvairias *Seq2Seq* modelio variacijas su tikslu sukurti automatinio lietuvių šnekos atpažinimo sistemą, kuri atpažintų ištartus žodžius. Galiausiai Stankevičius ir Lukoševičius [97] panaudojo multilingvistinius *BERT, XLM-R* transformerio metodika grįstus modelius ir pritaikė juos lietuviškų naujienų klasterizavimui. Anot autorių, tokių multilingvistinių transformerio modelių atsiradimas labai padeda natūralios kalbos generavimo užduotims ne anglų kalba, kadangi prieš tai dauguma apmokytų modelių bei metodų buvo kuriami bei pritaikomi anglų kalbai. Tie patys autoriai,

naudodami *ByT5* bei *T5* transformerius kūrė lietuvių kalbai skirtą klaidų taisymo modelį [98]. Lyginant abu transformerio modelius, *ByT5* pasiekė geresnį rezultatą nei *T5*. Autorių *ByT5* modelis buvo apmokomas 100 valandų, naudojant lietuvių kalbos tekstą bei sintetiškai sugeneruotas klaidas. Apmokytas modelis yra atvirai prieinamas ir jis šiame darbe naudojamas sureguliuoti konkrečiai diakritinių ženklų atstatymo užduočiai.

Utka ir Danilevičius [99] atliko lietuviškų socialinės medijos tekstų normalizavimo tyrimą. Nagrinėdami populiarių lietuviškų naujienų puslapių komentarus, autoriais nustatė, kad 17,85% žodžių yra nerandami žodyne (angl. *out-of-vocabulary* (*OOV*)). Net 74,6% tokių žodžių yra nerandami dėl diakritinių ženklų trūkumo (interneto vartotojai dažnai nenaudoja lietuviškos klaviatūros simbolių), kitos priežastys tai įvairūs pavadinimai (angl. *named entities*), neteisingos žodžių ribos ar tiesiog neteisingai parašytas žodis (praleistos, sumaišytos raidės). Taipogi, autoriai parodė, kad standartinis lietuvių kalbos rašybos tikrintojas, gali gan reikšmingai pagerinti morfologinės analizės rezultatus. Vadovaujantis šia prielaida, šiame darbe taipogi tikimasi, kad diakritinių ženklų atstatymas pagerins sentimento analizės tikslumą.

1.6. Apibendrinimas

Taigi, literatūros apžvalgoje aptarti automatinio teksto koregavimo taikymai, metodai bei modeliai, taipogi diakritinių ženklų atstatymo taikymai įvairioms kalboms, taikant skirtingus metodus. Išsiaiškinta, kad automatinio teksto koregavimo užduotis gali turėti ir tiesioginę naudą verslui. Nagrinėti pavyzdžiai atskleidė, kad toks teksto sutvarkymas gali pagerinti elektroninėje komercijoje, elektroniniame pašte ar kitose sistemose esančias paieškos funkcijas. Užtikrinant tikslius paieškos rezultatus.

Literatūro analizės pagalba sudaryta nuomonė, kad dabar dominuoti pradėjo giliojo mokymosi metodai, kurie tikslumu pralenkia tradicinius mašininio mokymosi metodus. Ypatingai gerais rezultatais pasižymi *Seq2Seq* modelis, kur taikomas multilingvistinis modelis gan aukštu tikslumu suveikė ir lietuvių kalbai [26]. Plačiau taikymo sritys buvo apžvelgtos *Seq2Seq* modeliui, kurio panaudojimo sritys varijuoja nuo tikslinių taikymų natūraliame kalbos generavime, mašininiame vertime ar kitose su tekstu ir kalba susijusiose užduotyse iki video segmentavimo, pritaikymų įvairiame energetikos prognozavimo, akcijų kainų prognozavime, rankraščių nuskaityme, muzikos atpažinime ar biologijos bei chemijos srityse. Tačiau net ir *Seq2Seq* dabar jau yra lenkiamas dar naujesnės, transformerio architektūros modelio.

Taip pat, šioje literatūros apžvalgoje skyriuje tyrinėtas ir sentimento analizės bei kitų natūralios kalbos generavime vykdomų užduočių tyrimai lietuvių kalbai. Galima įžvelgti, kad nors tyrimų kiekis nėra gausus, tačiau pastaruoju metu jų vis daugėja ir yra pasiekiami vis tikslesni rezultatai. Tai galimai susiję ir su transformerio modelių populiarėjimu ir tobulėjimu, kadangi šie dažnu atveju yra apmokyti ant daugybės duomenų bei skirtingų kalbų. Tada naudojant tokį jau apmokytą modelį, jį galimai lengvai susireguliuoti ir prisitaikyti sau reikalingai kalbai ar užduočiai ir tai nereikalauja didelių techninių resursų.

Kalbant apie sentimento analizę, galima susidaryti įspūdį, kad tai šiuolaikiniam verslui svarbus ir taikytinas uždavinys, su kuriuos galima sekti įmonės ar jos produktų reputaciją, susidaryti bendrą vaizdą apie klientus ir jų poreikius. Taipogi, literatūroje gausu tyrimų bei taikymų finansų ar verslo

vystymo srityse. Atskirai apžvelgus transformerių taikymo pavyzdžius sentimento analizei, rasta nemažai sėkmingų taikymo pavyzdžių iš kurių daugiausiai naudojant *BERT* modelį.

Nors daugiausiai NLP tyrimų yra atlikta anglų kalboje, jų taikymų kitoms kalboms, įskaitant ir morfologiškai kompleksinę lietuvių kalbą daugėja. Apžvelgta ir keletas pavyzdžių, kuriems buvo taikomi *Seq2Seq* modeliai pokalbių roboto kūrimo ar kalbos atpažinimo tyrimuose. Transformerio tipo modeliai naudoti lietuviškų naujienų klasterizavimui bei rašybos klaidų taisymui.

Mokslinėje literatūroje apžvelgus gausybę įvairiausių modelių bei jų pritaikymo pavyzdžių, daroma išvada, kad giliuoju mokymusi grįsti *Seq2Seq* bei transformerio tipo modeliai yra veiksmingiausi, tiksliausi bei sėkmingo pritaikymo atveju nešantys pridėtinę naudą. Taigi, kadangi atsiliepimai internete yra netvarkingi ir juos gali būti sunku pritaikyti įvairioms NLP užduotims, pavyzdžiui, sentimento analizei, kurios rezultatai tokiu atveju gali netenkinti. Lietuvių kalboje viena didžiausių netvarkingo teksto priežasčių yra diakritinių ženklų nenaudojimas tekste. Vadovaujantis šia prielaida, yra išsikeliamas tikslas sukurti diakritinių ženklų atstatymo įrankį, naudojant literatūros apžvalgoje tyrinėtus – *Seq2Seq* bei transformerio tipo modelius, kurie galėtų sutvarkyti netvarkingą internete parašytą tekstą, atstatytų praleistus ar netinkamai naudojamus diakritinius ženklus bei su sutvarkytu tekstu pakartotinai atliktų sentimento analizę. Naudojant atsiliepimus su atstatytais diakritiniais ženklais, sugeneruotų sentimento klasifikavimo modelį bei palyginus rezultatus išsiaiškinti ar tokio tipo teksto sutvarkymas gali prisidėti prie sentimento klasifikavimo užduoties rezultato pagerinimo.

2. Metodologija

2.1. *Sequence to Sequence* (*Seq2Seq*) modelis

Seq2Seq, kaip giliojo mokymosi metodas yra daugiausiai naudojamas automatiniam teksto vertimui, tačiau, kaip buvo apžvelgta pirmoje darbo dalyje, ši architektūra pasitelkiama ir kitoms užduotims, kuriose modelis kuriamas naudojant sekos tipo duomenis.

Šiame metode yra naudojamas rekurentinių neuroninių tinklų (RNN) (angl. recurrent neural network) tipas. Tai yra tokie neuroniniai tinklai, kuriuose praėjusio žingsnio išvestis yra išsaugoma ir panaudojama, kaip įvestis esamajam žingsniui. Juose yra naudojamas paslėptasis sluoksnis (angl. *hidden layer*), kurio paslėptojoje būsenoje (angl. *hidden state*) yra išsaugoma dalis informacijos. Tokiu principu neuroninis tinklas gali nuspėti sekantį žodį ar kitą sekos elementą, panaudodamas šią sluoksnio būsenoje išsaugotą informaciją.

RNN užkoduotojo-dekoduotojo (angl. *encoder-decoder*) tipo architektūra dažniausiai yra naudojama pasitelkiant LSTM (angl. *Long shor-term memory*) ląsteles, nors gali būti naudojami ir GRU (angl. *Gated recurrent unit*) ląstelės. Toliau modelio metodologija aprašoma naudojant LSTM tipo ląsteles.

Seq2Seq modelio, pagrindas yra sudarytas iš dviejų dalių: užkoduotojo ir dekoduotojo. Užkoduotojo paskirtis – nuskaityti paduodamus duomenis ir sukurti įvesties duomenų reprezentaciją. Ši reprezentacija sukuriama užkoduotojo LSTM ląstelei paduodant seką ($x = x_1, x_2, \dots, x_T$) pagal kurią yra atnaujinama paslėptoji būsena esamajame žingsnyje (h_t), pagal esamojo žingsnio įvestį (x_t) ir praėjusio žingsnio paslėptąją būseną (h_{t-1}):

$$h_t = f(x_t, h_{t-1});$$

čia f – neuroninio tinkle ląstelė (LSTM).

Kuomet paskutinis sekos narys (x_T) yra apdorotas, toliau naudojama galutinė paslėptosios būsenos versija (h_T). Taip pat užkoduotojo reprezentacijai yra naudojamas kontekstinis vektorius sudarytas iš sekos paslėptųjų būsenų:

$$c = q(\{h_1, h_2, \dots, h_T\};$$

čia q – tiesinė funkcija.

Abu šie dydžiai – h_T ir c, dar yra vadinami vidine būsena (angl. *internal state*).

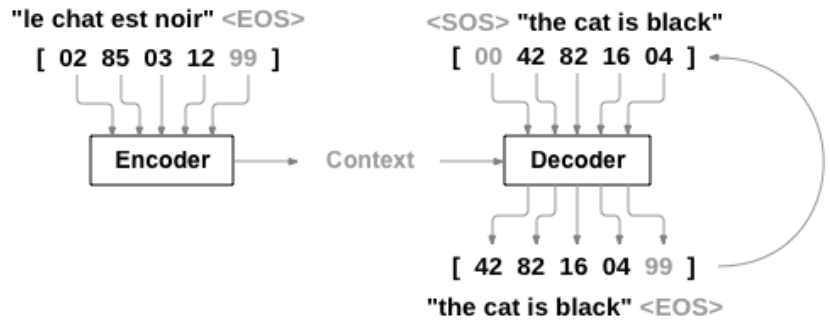
Sekančios architektūros dalies – dekoduotojo paskirtis yra panaudoti užkoduotojo reprezentaciją (kontekstinį vektorių c) kaip įvestį ir per dekoduotojo LSTM sugeneruoti išvesties seką ($y = (y_1, y_2, \dots, x_{T'})$. T bei T' gali būti skirtingi, kadangi ir išvesties elementų skaičius tarp x ir y gali būti skirtingas.

Paprasčiausias *Seq2Seq* modelis gali turėti po vieną neuroninio tinklo sluoksnį, tačiau dažniausiai yra naudojami keli ar net tūkstančiai sluoksnių tiek užkodavime, tiek dekodavime. Vienas pirmųjų tokių *Seq2Seq* modelio bandymų naudojant keletą sluoksnių buvo atlikta Sutskever ir kt. [46]. Tokiu atveju, paskutinis tankaus neuroninio tinklo sluoksnis naudoja minkšto maksimumo (angl. *softmax*) galutinei išvesčiai sugeneruoti:

$y_t = softmax(W^S h_t)$;

čia W^S – sujungtų neuronų sluoksnių svorių matrica.

Žemiau pateiktoje *Seq2Seq* architektūros schemoje galima pamatyti, kad tiek įvesties, tiek išvesties tekstas yra sugeneruojamas pasitelkiant tokenus. Tai skaitinė žodžio, simbolio ar kito lygmens reprezentacija, kuri yra sugeneruojama tekstą vektorizuojant. Vektorizacija gali būti daroma įvairiais būdais. Vienas iš naudojamų – raidės lygmens. *Seq2Seq* modeliuose tai veikia kuomet vienai raidei yra priskiriamas skaitmuo, taip pat kiekvienai sekai (sakiniui, pastraipai, tekstiniam dokumentui) yra pridedamas pradžios tokenas (schemoje jis žymimas, kaip <SOS> (angl. *start of sequence*) bei sekos pabaigos žymeklis <EOS> (angl. *end of sequence*). Tokiu būdu modelis mokydamasis turi žymeklius, kurie nurodo šiuos atributus.



1 pav. *Seq2Seq* modelio schema [100]

Seq2Seq modeliai, paduodant prieš tai einančio tokenus, yra apmokomi prognozuoti sekantį tokeną, kaip nurodyta schemoje. Taip yra sukuriamas pranašumas naudojant vieną neuroninį tinklą, kur kiekvienas įvesties vienetas atitinka išvesties vienetą.

Taigi, aptariamas modelis yra apibrėžiamas taip, kad kiekvienu žingsniu yra maksimizuojama sekančio tokeno teisingo priskyrimo tikimybė. Šiam procesui dažnu atveju yra naudojama kryžminės entropijos nuostolio funkcija:

$Loss = -\sum_{c=1}^{M} y_{o,c} \log(p_{o,c})$;

čia M– klasių skaičius; y – binarinis kintamasis (0 arba 1); p – tikimybė, kad stebėjimas o yra priklausantis klasei c.

Vis dėlto, kadangi prieš tai aptartoje metodologijoje visos paduodamos sekos (sakinio) reikšmė yra sutraukiama į vieną vektorių (c). Apdorojant ilgus sakinius šis vektorius gali prarasti savo prasmę, kadangi jame nebus išsaugota visa svarbi ir prasminga informacija. Pagerinti architektūros tikslumą apdorojant ilgesnes sekas, Bahdanau ir kt. [47] pasiūlė dėmesio mechanizmą. Šiuo mechanizmu dekoduotojui yra paduodama ne tik paskutinė paslėptoji būsena (h_T), tačiau visos sekoje sugeneruotos paslėptos būsenos. Tai padaroma paskaičiuojant dėmesio svorių lygiavimo vektorių (α_{ts}), lyginant paskaičiuotą užkoduotojo paslėptąją būseną ($\bar{h}_s$) su kiekviena gaunama dekoduotojo paslėpta būsena (h_t):

$$\alpha_{ts} = \frac{\exp\left(score(h_t, \bar{h}_s)\right)}{\sum_{s'=1}^{S} \exp\left(score(h_t, \bar{h}_{s'})\right)};$$

čia s – užkoduotojo laiko momentas; t – dekoduotojo laiko momentas; h_s – užkoduotojo paslėptoji būsena; h_t – dekoduotojo paslėptoji funkcija.

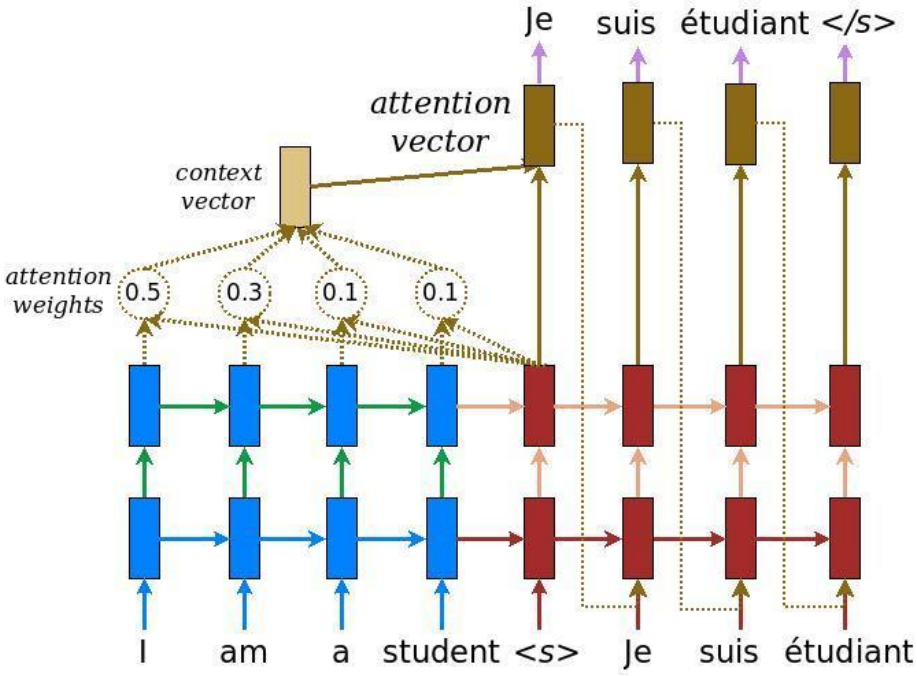
$score$ – palyginimo funkcija tarp užkoduotojo ir dekoduotojo paslėptųjų būsenų. Be jau minėto Bahdanau pasiūlymo, kitą palyginimo funkciją pasiūlė Luong ir kt. [101]. Nurodomi abu dažniausiai dėmesio mechanizme naudojami palyginimo funkcijos variantai:

$$score\left(h_t, \bar{h}_s\right) = \begin{cases} h_t^\top W \bar{h}_s \ (Luong\ skaliarinės\ sandaugos\ funkcija) \\ v_a^\top \tanh\left(W_1 h_t + W_2 \bar{h}_s\right) \ (Bahdanau\ suminė\ funkcija) \end{cases}$$

Dėmesio mechanizmo kontekstini vektorius (c) gaunamas paskaičiuojant užkoduotojo paslėptųjų būsenų svertinę sumą:

$$c = \sum_s \alpha_{ts} \bar{h}_s$$

Pavaizduojama tokio *Seq2Seq* modelio su dėmesio mechanizmu pavyzdinė schema (2 pav.), kurioje mėlyna spalva pavaizduotas užkoduotojas, o raudona – dekoduotojas bei kontekstinio vektorio sluoksnis sukurtas dėmesio mechanizmu.



2 pav. *Seq2Seq* su dėmesio mechanizmu pavyzdinė schema [101]

Galiausiai, apmokius modelį, su juo yra generuojamas vertimas, atliekami įvairūs klasifikavimai ar kiti su sekos tipo duomenimis galimi atlikti uždaviniai. Vienas paprasčiausių metodų yra „gobšus“ dekodavimo (angl. *greedy decoding*). Naudojant jį kiekviename žingsnyje, naudojant *arg max* funkciją yra pasirenkamas tokenas su didžiausia tikimybe. Kitas panašus metodas yra „spindulio paieškos“ (angl. *beam search*) metodas su kuriuo yra bandomas nurodytas skaičius sekančių tokenų

skaičius su aukščiausiomis tikimybėmis ir einant per seką renkamasis vis geriausias pasirinkimas kol galiausiai aukščiausia tikimybė atsiranda sekos pabaigos tokenui ir vertimas yra baigtas. Tokia seka šis metodas primena sprendimų medžio metodą.

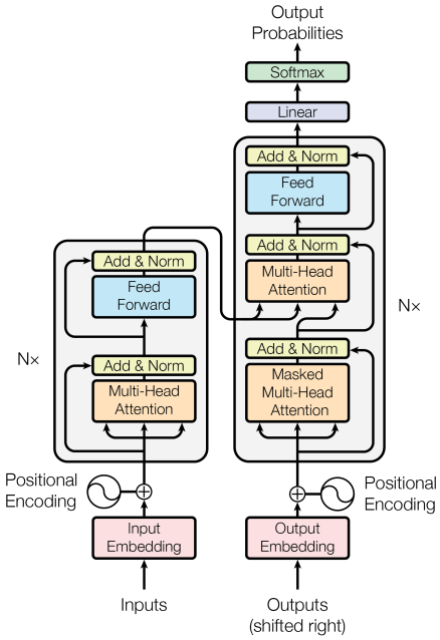
2.2. Transformerio modelis

Transformerio modelio tipo pradžia yra laikomi 2017 metai, kuomet „Google“ mokslininkai Vaswani ir kt. [83] paviešino savo publikaciją. Nuo to laiko savo rezultatais ir tikslumu transformeriai pranoko iki to laiko naudotus ir prieš tai aptartus *Seq2Seq* modelius ir iki šiol yra moderniausias (angl. *state of the art*) ir populiariausias natūralios kalbos apdorojime naudojamas metodas.

Šio modelio architektūros pagrindas – *Seq2Seq* modeliuose naudojamas dėmesio mechanizmas. Vienas esminių šių modelių dėmesio skirtumų, tai kad dėmesio mechanizmas transformerio atveju yra atliekamas kiekviename iš architektūros komponentų atskirai (priešingai nei *Seq2Seq* modelio atveju, kuomet dėmesio mechanizmas yra tarp užkoduotojo ir dekoduotojo), todėl dar yra vadinamas „dėmesio į save“ (angl. *self-attention*) mechanizmu. Taip pat transformeryje, skirtingai nei prieš tai aptartos architektūros atvejis, nėra naudojamas sekos principas ir visas įvesties elementas yra apdorojamas iš karto. Vaswani ir kt. [83] teigia, kad tai sukuria platesnes paralelinio skaičiavimo (angl. *parallel computing*) galimybes ir sumažina apmokymo laiką.

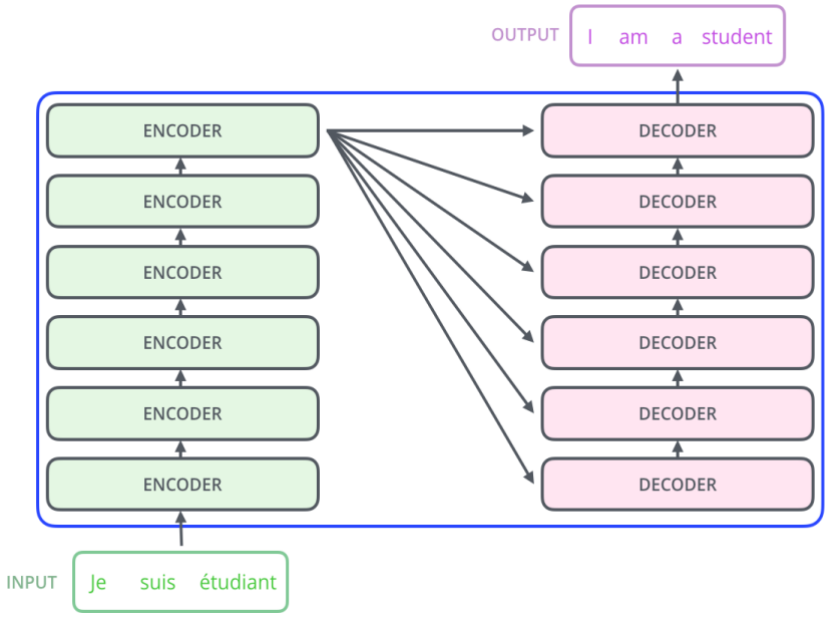
Kaip ir *Seq2Seq* modelio atveju, taip ir transformerio modelio atveju yra naudojama užkoduotojo-dekoduotojo struktūra. Šiuo atveju užkoduotojas įvesties seką, kuri sudaroma iš tekstinio vieneto skaitinės reprezentacijos ($x = (x_1, \dots, x_n)$), kuri paverčiama į tolydžią reprezentaciją ($z = (z_1, \dots, z_n)$). Ši z reprezentacija dekoduotojo yra išskaidoma po vieną elementą ir gaunama išvesties seka $y = (y_1, \dots, y_m)$. Kiekvienas modelio žingsnis yra autoregresinis – prieš tai sugeneruoti tekstiniai vienetai yra naudojami kaip papildoma įvestis.

Transformerio architektūra sudaro n skaičius užkoduotojo sluoksnių ir toks pat skaičius dekoduotojo sluoksnių, kaip tai nurodoma žemiau pateiktoje schemoje:



3 pav. Transformerio modelio architektūra [83]

Vaswani ir kt. [83] savo straipsnyje nurodo 6 sluoksnių architektūrą. Supaprastinta tokio modelio schema pateikiama žemiau:



4 pav. Supaprastinta transformerio modelio architektūra naudojant 6 sluoksnius [102]

Kaip matoma 3 pav., užkoduotojas, sudarytas iš n skaičiaus sluoksnių, kiekvienas sluoksnis taip pat yra sudaromas iš dviejų posluoksnių (angl. *sub-layer*). Pirmasis posluoksnis turi savyje daugelio galvų dėmesio mechanizmą (angl. *multi-head self-attention*). Antrasis posluoksnis – persiuntimo (angl. *feed-forward*) neuroninis tinklas. Aplink abu posluoksnius yra naudojama liekanų jungtis (angl. *residual connection*) po kurio seka sluoksnio normalizavimas: $LayerNorm(x + Sublayer(x))$, kur $Sublayer(x)$ – pačio posluoksnio implementuojama funkcija. Autoriai nurodo, kad modelyje visų modelio posluoksnių sugeneruojama išvestis atitinka d_{model} (jų pavyzdžio atveju $d_{model} = 512$) dimensijų skaičių.

Dekoduotojas, sudarytas iš n skaičiaus sluoksnių, iš kurių kiekvienas sudaromas iš dar 3 posluoksnių. Dekoduotojui yra paduodama užkoduotojo išvestis, kuri yra paslenkama per vieną poziciją į dešinę, tam, kad modelio prognozė i pozicijai būtų priklausoma tik jau nuo prieš tai žinomų užkoduotojo išvesties elementų ir tai būtų mažiau nei i pozicijų skaičius. Taip pat, lyginant su užkoduotoju, atsiradęs papildomas sluoksnis yra skirtas užkoduotojo sugeneruotos išvesties praleidimui per papildomą daugelio galvų dėmesio mechanizmą. Galiausiai, liekanų jungtis bei sluoksnio normalizavimas dekodavime aplink visus sluoksnius yra naudojamas tuo pačiu principu, kaip ir užkodavime,.

Kaip ir minėta, transformerio vienas iš pranašumų, lyginant su *Seq2Seq* modeliu yra tai, kad vektorizuotas teksto vienetas (angl. *embedding*) nėra apdorojamas sekos principu, o visas įvesties elementas yra apdorojamas visas iš karto. Taip pat, kad būtų išvengta informacijos praradimo susijusio su teksto vienetų eiliškumu, yra naudojamas pozicinis užkodavimas (angl. *positional encoding*). Vaswani ir kt. [83] savo straipsnyje naudoja sinuso ir kosinuso funkcijas tokiam užkodavimui atlikti. Lyginėms dimensijoms yra taikoma sinuso funkcija, o nelyginėms – kosinuso:

$$PE_{(pos,2_i)} = \sin\left(pos/10000^{2i/d_{model}}\right)$$

$PE_{(pos,2_{i+1})} = \cos(pos/10000^{2i/d_{model}})$;

čia pos – esama pozicija; i – esama dimensija.

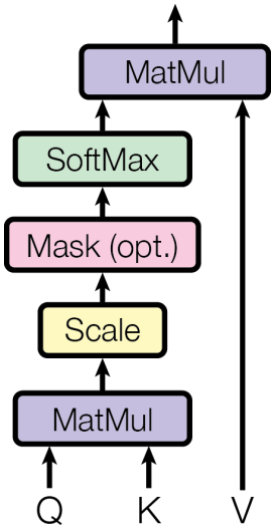
Transformerio modelyje naudojamas „dėmesio į save“ mechanizmas, kurio tikslas – nustatyti kontekstinius ryšius tarp teksto vienetų viename įvesties elemente (pavyzdžiui, tarp žodžių sakinyje) ir sukurti dėmesio mechanizmo paskaičiavimu grįsta vektorių kiekvienam įvesties vienetui. Taip yra įvertinamas kiek kiekvienas elementas yra reikšmingas, lyginant su kitais elementais įvesties vienete.

Pirmasis „dėmesio į save“ mechanizmo metodas – skaliarinės sandaugos dėmesio mechanizmas (angl. *scaled dot-product attention*). Šiuo metodu paskaičiuojama funkcija, kuriai paduodami užklausų (angl. *query*) – Q vektorių matrica ir raktų (angl. *keys*) – K vektorių matrica, kurių ilgis yra d_k dimensija bei reikšmių (angl. *values*) – V vektorių matrica su d_v dimensija. Pirmuoju žingsniu suskaičiuojamos skaliarinės sandaugos tarp Q ir K, tuomet padalijama iš $\sqrt{d_k}$. Šiam paskaičiavimui pritaikoma minkštojo maksimumo (angl. *softmax*) funkcija gaunami dėmesio svoriai, kurie po to padauginami iš V. Praktikoje taikomą funkciją galima aprašyti taip:

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V;$$

čia Q – užklausų vektorių matrica; K – raktų vektorių matrica; V – reikšmių vektorių matrica; d_k – reikšmių ir raktų vektorių matricų dimensionalumas.

Taip pat pavaizduojama funkcijos skaičiavimo proceso schema:



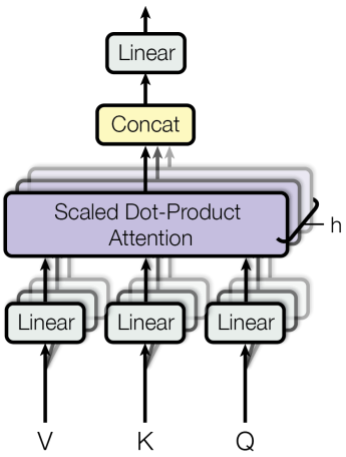
5 pav. Skaliarinės sandaugos dėmesio mechanizmo schema [83]

Kitas „dėmesio į save“ mechanizmo metodas – daugelio galvų dėmesio metodas. Jis padeda tam pačiam teksto vienetui sugeneruoti keletą dėmesiu grįstų reprezentacijų viename įvesties elemente, naudojant h kiekį skirtingų užklausų. Šių užklausų rezultatai į vieną matricą yra apjungiami (angl. *concatenate*) juos perleidžiant per tiesinės funkcijos sluoksnį:

$$MultiHead(Q,K,V) = Concat(head_1,\dots,head_h)W^O$$

$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$;

čia W_i^Q, W_i^K, W_i^V – neuronų sluoksnių svorių matricos.



6 pav. Daugelio galvų dėmesio mechanizmo schema [83]

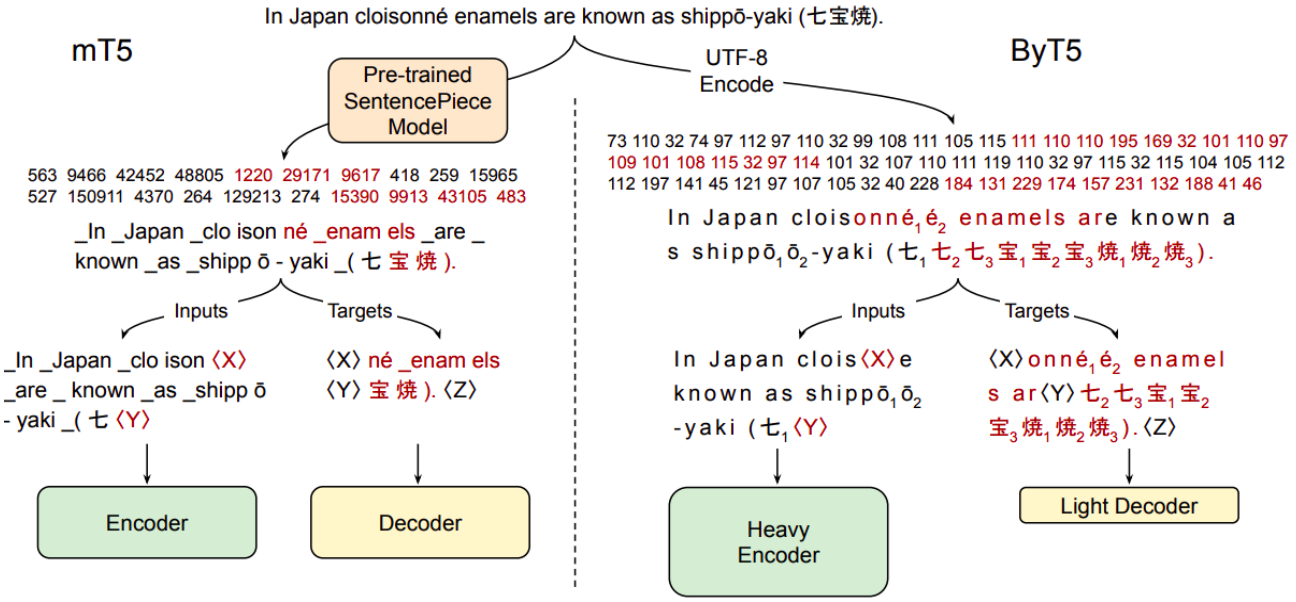
2.2.1. *ByT5* modelis

ByT5 yra Xue ir kt. [103] pristatytas transformerio tipo modelis, kuris teksto tokenizavimą atlieka naudojant simbolio baito lygmenį. Šiuo metodu kiekvienas simbolis teksto duomenų rinkinyje yra konvertuojamas pagal UTF-8 simbolių koduote, kur ASCII koduotės simboliai yra konvertuojami į vieno baito atitikmenį, o nepriklausantys šiai koduotei į 2 ar daugiau baitų atitikmenį.

Šio modelio pirmtakai – multilingvistinis, naudojant 101 skirtingą kalbą apmokytas *mT5* modelis [104] bei pastarojo pirmtakas *T5* modelis [105] naudoja požodinį tokenizavimo lygmenį, kuris dažnai pasikartojantiems žodžiams priskiria vieną tokeną, o rečiau pasikartojančius žodžius skaido į dalis. Lyginant šiuos modelius su *ByT5*, pastarasis yra pranašesnis apdorojant netvarkingą ar klaidingą tekstą. *mT5/T5* modeliai gali netvarkingam tekstui priskirti netinkamos reikšmės tokeną, o naudojant *ByT5* to yra išvengiama, kadangi pilni žodžiai ar jų dalys nėra aktualūs. Taipogi, tinkamai *ByT5* modeliui tinkamai apmokyti nereikia teksto su plačiu žodynu. Pavyzdžiui, *mT5* bazinio modelio atveju 66% svorio parametrų buvo priskirti su žodynu susijusiems parametrams [103]. Vadinasi, didelė dalis žodžių ar jų dalių svorių lieka labai mažai apmokyti. Tai yra išsprendžiama naudojant *ByT5* modelį, kadangi jo žodynas sudaro 256 skirtingus tokenus.

Vis dėlto, lyginant *ByT5* su *mT5/T5* modeliais, baitų lygmens modelius apmokyti gali trukti žymiai ilgiau, kadangi tekstą tokenizuojant šiuo lygmeniu, sekos yra žymiai ilgesnės nei pats tekstas pažodžiui. Transformerio modelio „dėmesio į save“ mechanizmas pasižymi kvadratiniu algoritmo sudėtingumu (angl. *quadratic time complexity*) – padidinus įvesties dydį, jo įvykdymo greitis padidėja kvadratu.

7 pav. pateiktoje schemoje galima įžvelgti skirtumus *mT5* ir *ByT5* modeliu tokenizuotiems sakiniams bei iš jų sugeneruotų sekų ilgiams:



7 pav. *mT5* ir *ByT5* modelių teksto tokenizavimo pavyzdžiai [103]

2.3. Latentinis Dirichlė pasiskirstymas

Pagal Hoffman, Bach ir Blei [106], latentinio Dirichlė pasiskirstymo metodas yra Bajeso tikimybinis metodas, naudojamas tekstiniams dokumentams. Šiuo metodu sugeneruojamas nurodytas K skaičius temų. Duotam temų skaičiui yra sugeneruojamas žodynas, kuris apibrėžiamas tikimybių vektoriumi – $\beta_K \sim Dirichlet(\eta)$. Generavimo procesas vykdomas kiekvienam dokumentui (d), kurio metu jame esančiam pavieniam žodžiui priskiriamas temos numeris – z_{di}. Įvertinus kiekvieno žodžio tematikų svorius $z_{di} \sim \theta_d$, pateikiamas nurodytas skaičių žodžių, kurie stipriausiai prezentuoja atitinkamą temą.

Šis mokymosi be mokytojo metodas klasifikavimo metodas pasitarnauja norint atskleisti turimo tekstinio duomenų rinkinio pagrindines temas arba sentimento analizės uždavinio atveju – nustatyti temų pasiskirstymą per kintamojo klases.

2.4. Latentinė semantinė analizė

Latentinė semantinė analizė arba latentinis semantinis indeksavimas (LSI) yra NLP metodas skirtas iš tekstinių duomenų ištraukti kontekstinę informaciją bei statistiškai ją apdirbti. Taipogi šiuo metodu sumažinama vektorinė erdvė ir taip išvengiama tekste esamo triukšmo. Galutinis šio algoritmo rezultatas – matrica, kurioje viena eilutė reprezentuoja teksto vienetą (sakinį, paragrafą), ir nurodyto dimensialumo vektorių. Tokia matrica po to gali būti naudojama įvairioms užduotims: teksto paieškai, klasifikavimui, išgavimui. Šis metodas taip pat populiarus vykdant sentimento klasifikavimą.

Dažnu atveju latentinės semantinės indeksavimo metodui naudojama termino dažnio – atvirkštinio dokumento dažnio (TF-IDF) metodą. Šiuo metodu sugeneruojama pirminė matrica reprezentuojanti esamų terminų dažni tekstiniuose dokumentuose. Taip pamatuojama kiek svarbus yra žodis atitinkamam teksto dokumentui viso turimo tekstyno kontekste. Metodas kiekvienam terminui

paskaičiuoja santykio logaritminę reikšmė tarp viso dokumentų kiekio bei dokumentų kiekio, kurie turi šį konkretų terminą:

$$idf(terminas) = \ln\left(\frac{n_{dokumentai}}{n_{dokumentai\ su\ terminu}}\right)$$

Šis metodas yra vertingas tuo, kad jis sukuria atitinkamus svarbumo svorius dažniau pasikartojantiems žodžiams, tačiau tuo pačiu ir neiškelia bendrinių, jungiamųjų beveik kiekviename dokumente po vieną ar kelis kartus pasikartojančius žodžių.

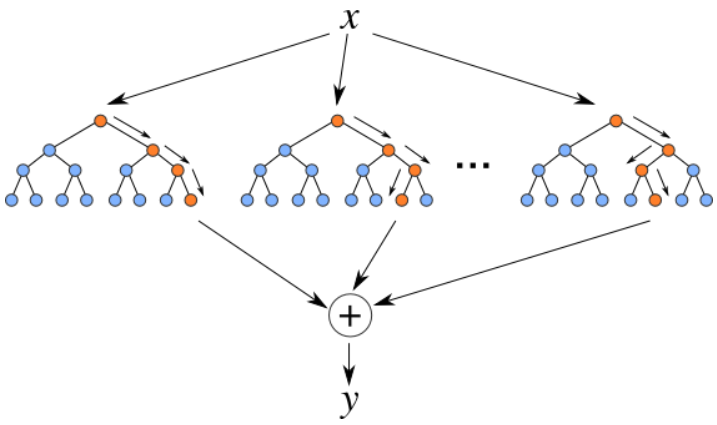
Pirmuoju LSI metodo žingsniu yra sukuriama teksto matrica (C), kur kiekvienoje eilėje yra unikalus žodis, o stulpeliuose nurodoma TF-IDF metodu apskaičiuotas skaičius. Naudojant singuliarių reikšmių dekompozicijos (SVD) algoritmą iš matricos C yra sukuriamos atskiros matricos: terminų dokumentų matrica (T), singuliarių reikšmių matrica (Σ) ir konteksto matricą (D), kurioje reprezentuojama matricos elementų struktūra. Pirminę matricą galima vėl sugeneruoti matricinė daugybos būdu: $C = T\Sigma D^T$. Svarbu paminėti, kad latentinės semantinės analizės metodu matricos T, Σ ir D yra sumažinamos iki k dimensionalumo. Tad matricinės daugybos būdu sudauginus šias mažesnės dimensijos matricas, gaunama pradinė matrica (C), tačiau jau mažesniu dimensionalumu. Tokiu būdu sumažinamas latentinėje erdvėje esantis triukšmas bei sukuriami stipresni ryšiai tarp skirtingų teksto dokumentų ar skirtingų klasių vykdant klasifikavimo uždavinį.

2.5. Sentimento analizės modeliai

2.5.1. Atsitiktiniai miškai

Atsitiktinių miškų metodas yra mokymosi su mokytoju (angl. *supervised learning*) mašininio mokymosi algoritmas, naudojamas klasifikavimo ir regresijos uždaviniuose. Šis metodas buvo išgrynintas mokslininko Breiman‘o [107] ir jis yra priskiriamas ansamblių metodams (angl. *ensemble methods*), kurių tikslas iš daugelio bandomų modelių pasirinkti ir sugeneruoti vieną geriausią modelį.

Metodas veikia atsitiktinai kurdamas sprendimo medžius ir jų ansamblius. Jų sudarymui yra naudojama atsitiktinė imties dalis, tad vienas iš šio metodo privalumų yra tai, kad jie nėra linkę į persimokymą [107]. Klasifikavimo uždavinyje kiekvienas sprendimų medis balsuoja ir iš jų balsavimo populiariausia klasė yra pasirenkama kaip atitinkamo stebėjimo klasifikacijos rezultatas. Supaprastinta atsitiktinio miško schema pateikiama 8 pav.



8 pav. Atsitiktinio miško klasifikavimo schema

Atsitiktinių miškų metodas turi tris pagrindinius parametrus, kurie nustatomi prieš pradedant mokymą: mazgų skaičius (angl. *node size*), kuriamų medžių skaičius bei kintamųjų skaičius naudojamas konstruojant sprendimų medį. Pirmuoju žingsniu, pagal nurodomą kuriamų medžių skaičių, yra kuriamas atsitiktinis miškas su atsitiktinai priskirtais duomenų atributais. Tada iš šių medžių yra paliekami tik geriausiai klasifikuojantys medžiai. Šis klasifikavimo gerumas testuojamas su mokymosi procese nedalyvavusia testavimo duomenų imtimi (angl. *out-of-bag*). Iš geriausiai pasirodžiusių medžių balsavimo, atitinkamam stebėjimui yra priskiriama daugumą balsų surinkusi klasė.

2.5.2. Logistinė regresija

Logistinė regresija yra vienas elementariausių statistinių modelių naudojamų mašininio mokymosi klasifikavimo uždaviniuose. Logistinė regresija iš dalies primena tiesinę regresiją, tačiau naudojant šį metodą apskaičiuojama konkretaus rezultato tikimybė. Dėl to šis metodas puikiai tinka apskaičiuoti binominiam rezultatai, pavyzdžiui tiesa/netiesa, taip/ne, teigiamas/neigiamas ir pan.

Vienas iš pagrindinių skirtumų nuo tiesinės regresijos, tai kad logistinės regresijos išvedamas skaičius yra apribotas tarp 1 ir 0. Taip yra dėl šioje logistinėje funkcijos formulės:

$$Logistinė\ funkcija = \frac{1}{1 + e^{-x}}$$

Logistinės regresijos modelio funkcija aprašoma tokia lygtimi:

$$logit(P) = \ln\left(\frac{P}{1-P}\right) = \alpha + \beta X;$$

$$P = \frac{e^{\alpha+\beta X}}{1+e^{\alpha+\beta X}};$$

čia P – įvykio tikimybė; α – laisvasis narys; β – regresijos koeficientas; X – nepriklausomas kintamasis.

Logistinėje regresijoje, lyginant su kai kuriais kitais modeliais, nėra reikalaujama, kad duomenys būtų homoskedastiniai, kad paklaidos atitiktų normalųjį skirstinį. Taipogi, kintamieji šioje regresijoje galimi ne tik intervaliniai, tačiau ir kategoriniai.

Galiausiai, dažniausiai naudojama daug nepriklausomųjų kintamųjų (n), tad pateikiamas universalus logistinės regresijos modelio apibrėžimas prognozuojant priklausomąjį kintamąjį Y:

$$logit(Y) = \ln\left(\frac{P}{1-P}\right) = \alpha + \beta_1 X_1 + \cdots + \beta_n X_n$$

2.6. Modelių tikslumo vertinimo metrikos

2.6.1. Raidės ir diakritinio ženklo atstatymo tikslumas

Raidės ir diakritinio ženklo atstatymo tikslumas paskaičiuojamas pagal Naplava ir kt. [27] atvirame kode pateiktą funkciją, kuri įvertina modelio ženklais atstatyto teksto bei tikslo (angl. *target*) teksto simbolių skaičių santykį:

$$raidės\ tikslumas = \frac{sutampančių\ raidžių\ skaičius\ atstatytame\ tekste}{raidžių\ skaičius\ tikslo\ tekste}$$

Taip pat, įvertinti konkrečiai diakritinių ženklų atstatymo tikslumą, tokiu pačiu principu yra paskaičiuojamas diakritinių ženklų skaičius tikslo tekste bei paskaičiuojamas santykis su sutampančiais diakritiniais simboliais modelio atstatytame tekste:

$$diakritinio\ ženklo\ tikslumas = \frac{sutampančių\ diakritikų\ skaičius\ atstatytame\ tekste}{diakritinių\ ženklų\ skaičius\ tikslo\ tekste}$$

2.6.2. Sumaišymo matrica

Sumaišymo matrica (angl. *confusion matrix*) yra išdėstomos turimų duomenų tikrosios klasės bei modelio prognozuotos klasės. Iš jos galima nustatyti kuri klasė yra prognozuojama tiksliau.

Žemiau pateiktoje matricoje yra nurodomi stebėjimų tipai: TP – teisingai suklasifikuoti teigiamos klasės stebėjimai (angl. *true positive*), TN – teisingai suklasifikuoti neigiamos klasės stebėjimai (angl. *true negative*), FN – neteisingai suklasifikuoti neigiamos klasės stebėjimai (trūkstami rezultatai) (angl. false negative), FP – neteisingai suklasifikuoti teigiamos klasės stebėjimai (netikėti rezultatai) (angl. false positive).

3 lentelė. Sumaišymo matrica

		Nustatyta klasė	
		T	**N**
Tikroji klasė	**T**	TP	FN
	N	FP	TN

Naudojant sumaišymo matricoje gaunamus skaičius yra paskaičiuojamos sekančiuose skyreliuose aprašomos modelio tikslumo vertinimo metrikos.

2.6.3. ROC kreivė

ROC kreivė (angl. *receiver operating characteristic curve*) dažnai naudojama detekcijos uždavinių tikslumui nustatyti bei kelių modelių rezultatams palyginti. ROC kreivė įvertina ryšį tarp jautrumo (angl. *sensitivity*) ir specifiškumo (angl. *specificity*). Šie santykiniai dydžiai yra paskaičiuojami taip:

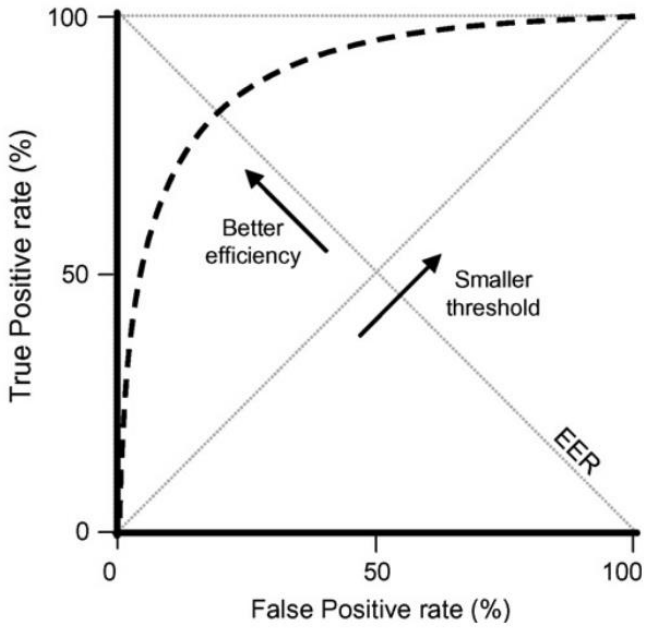
$$jautrumas = TPR = \frac{TP}{(TP + FN)}$$

$$specifiškumas = TNR = \frac{TN}{(TN + FP)}$$

$$FNR = \frac{FN}{(TP + FN)}$$

$$FPR = \frac{FN}{(TP + FN)} = 1 - specifiškumas$$

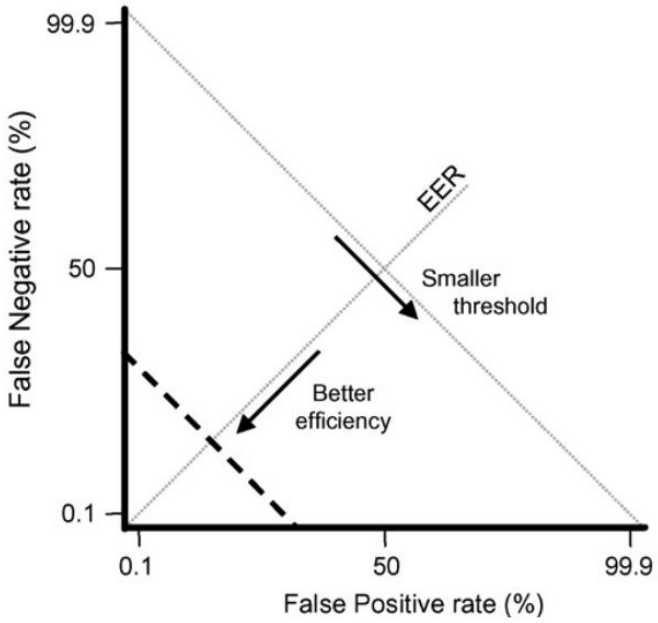
ROC kreivė yra nubrėžiama nustačius skirtingus modelio gautų rezultatų (tikimybių) klasifikavimo slenksčius. Šiuose slenksčiuose gaunami TPR ir FPR taškai yra sudedami į grafiką bei apjungiami į liniją. Tiesi įstriža linija simbolizuoja atsitiktinį spėliojimą, tad kuo kreivė yra toliau nuo šios linijos, tuo konkretus modelis yra vertinamas kaip geriausias. 9 pav. parodytame pavyzdyje iš kairės į dešinę kylanti įstrižainė simbolizuoja šią liniją. Taip pat tikslesniam geresnio modelio nustatymui, kreivėms esant šalia viena kitos arba joms persidengiant, pravartu paskaičiuoti AUC (angl. *area under curve*), kuris nurodo plotą po ROC kreive. Šis įvertis gali būti nuo 0 iki 1, o kai AUC < 0,5, tuomet modelį galima vertinti kaip prastesnį nei atsitiktinis spėjimas.



9 pav. ROC kreivės pavyzdys [108]

2.6.4. DET grafikas

DET grafikas (angl. *detection error tradeoff graph*) nurodo neteisingai suklasifikuotų abiejų klasių įverčių (FNR, FPR) ryšį. Naudojant šį grafiką galima įvertinti, kurią klasę prognozuojant yra klystama labiau. 10 pav. parodytame DET grafiko pavyzdyje, įstrižainė besileidžiančioje iš kairės į dešinę simbolizuoja atsitiktinį spėliojimą, tad šiuo atveju kuo kreivė yra žemiau šios ribos, tuo modelis vertinamas geriau. Ašys šio grafiko atveju dažniausiai yra logaritmuojamos.



10 pav. DET grafiko pavyzdys [108]

Taip pat prie šio grafiko taip pat dažnu atveju būna nurodomas EER (angl. *equal error rate*), nurodantis tašką, kuriame FNR ir FPR dydžiai yra vienodi. Kuo šis įvertis mažesnis, tuo modelio klasifikavimas yra tikslesnis.

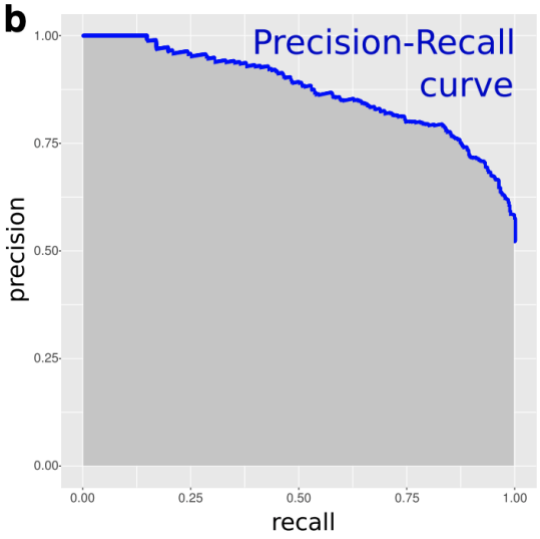
2.6.5. PR kreivė

Preciziškumo-atkuriamumo (angl. *precision-recall (PR)*) kreivė padeda įvertinti klasifikavimo modelio rezultatą tuomet kai duomenyse yra klasių disbalansas. Šis grafikas įvertina preciziškumo (angl. *accuracy*) bei atkuriamumo (angl. *recall*) įverčius:

$$preciziškumas = \frac{TP}{(TP + FP)};$$

$$atkuriamumas = \frac{TP}{(TP + FN)}$$

Kaip ir ROC kreivės atveju, tai ir po PR kreive esantis didelis ploto parodo geresnį modelio rezultatą. Aukštas preciziškumas reiškia, kad modelis rečiau klysta neteisingai prognozuodamas teigiamą klasę, o aukštas atkuriamumas signalizuoja, apie retą klydimą prognozuojant neigiamą klasę.



11 pav. PR kreivės pavyzdys [109]

Iš šioje kreivėje naudojamų įverčių, juos apjungus yra paskaičiuojamas F-įvertis (angl. *F-score*), kuris yra harmonizuotas preciziškumo ir atkuriamumo vidurkis:

$$F\ score = 2\frac{preciziškumas \times atkuriamumas}{preciziškumas + atkuriamumas}$$

Didžiausia galima F-įverčio reikšmė – 1, o mažiausia – 0, kas simbolizuotų, kad kažkuris iš formulėje naudojamų įverčių yra lygus 0.

2.7. Diakritinių ženklų atstatymo modeliams naudojami tekstynai

Diakritiniai ženklai bus atstatomi sentimento analizės uždaviniui, kuriam naudojami internete vartotojų paliekami atsiliepimai. Juose rašoma natūralia, šnekamąja kalba, todėl ir diakritikų atstatymo modeliams apmokyti yra reikalingi tokios kalbos gausūs tekstynai. Pirmasis – *OpenSubtitles* tekstynas, kuris yra sudarytas iš įvairių filmų subtitrų. Antrasis – lietuvių kalbos *ASTRA* tekstynas sudarytas iš parlamentarų pasisakymų Seime 1990 m. kovo mėn. – 2013 gruodžio mėn. laikotarpiu.

2.7.1. *OpenSubtitles* lietuvių kalbos tekstynas

OpenSubtitles [110] originalų 58,9 MB .txt formato failą sudaro 2,13 mln. eilučių bei 8,46 mln. žodžių. 2,69 mln. iš šių žodžių turi bent vieną lietuvių kalbos diakritinį ženklą (ą, č, ę, ė, į, š, ų, ū, ž) ir tai sudaro 31,8% nuo visų žodžių skaičiaus. Kadangi kiekviena eilutė atitinka vieną kino subtitrą, eilutės (sakiniai) yra gana trumpi ir vidutiniškai viena eilutė sudaro 4 žodžius.

4 lentelė. *OpenSubtitles* tekstyno statistikos santrauka

OpenSubtitles	**Eilučių kiekis**	**Žodžių kiekis**	**Žodžių kiekis su diakritikais**	**Žodžių su diakritikais procentas**	**Vidutinis žodžių kiekis eilutėje**
	2125805	8460899	2691095	31,8%	3,98

Taip pat žemiau lentelėje apžvelgiami populiariausi pasikartojantys žodžiai. Galima įžvelgti, kad tai daugiausia jungiamieji žodžiai arba įvardžiai, kurie dažniausiai ir yra naudojami šnekamojoje kalboje:

5 lentelė. *OpenSubtitles* tekstyne dažniausiai pasikartojantys žodžiai

Žodis	ir	aš	kad	tai	tu	jis	į	su	ar	kaip	ką
Kiekis	138249	120918	106087	103214	88811	69592	63309	60710	59835	58280	55272

2.7.2. *ASTRA* tekstynas (parlamentarų pasisakymai Seime)

ASTRA tekstynas [111] pateiktas atskiruose 110 tūkst. failų, kur kiekvienas yra atskira parlamentaro kalbos tekstas konkrečią dieną bei papildoma informacija tokia kaip vidutinis sakinio ilgis tekste, vidutinis žodžio ilgis tekste ir t.t. Iš viso failai užima 9,1 GB. Panaudojus „Bash“ komandų kalbą iš kiekvienos failo ištraukiamas tik reikalingas tekstas bei sutraukiama į vieną 185 MB .txt failą. Atskyrus kiekvieną sakinį į atskirą eilutę, šis failas sudaro 2,06 mln. eilučių (sakinių). Visumoje tekstyne yra 23,3 mln. žodžių, kas vidutiniškai sudaro 11,33 žodžių per sakinį. 8,59 mln. žodžių yra turintys bent vieną lietuvišką diakritinį ženklą ir tai sudaro 36,7% visų žodžių. Galima pastebėti, kad šio tekstyno sakiniai vidutiniškai yra beveik trigubai ilgesni nei kino filmų subtitrų tekstyno.

6 lentelė. *ASTRA* tekstyno parlamentarų pasisakymų Seime tekstyno statistikos santrauka

ASTRA	**Eilučių kiekis**	**Žodžių kiekis**	**Žodžių kiekis su diakritikais**	**Žodžių su diakritikais procentas**	**Vidutinis žodžių kiekis eilutėje**
	2064250	23397429	8587879	36,7%	11,33

Taip pat pateikiami 10 labiausiai tekstyne pasikartojančių žodžių:

7 lentelė. *ASTRA* tekstyno parlamentarų pasisakymų Seime dažniausiai pasikartojantys žodžiai

Žodis	ir	kad	yra	tai	aš	mes	ar	dėl	į	seimo
Kiekis	774170	431200	336904	229881	206040	191519	163621	163396	152753	138706

2.7.3. Tekstiniai duomenys diakritinių ženklų atstatymo modeliams apmokyti

Diakritinių ženklų modeliams apmokyti dėl reikalingo didelio kiekio tekstinių duomenų, šie du tekstynai yra apjungiami. Tekstiniai duomenys yra pakoreguojami, kad jie neturėtų didžiųjų raidžių (pakeičiamos visos raidės į mažąsias), pašalinami visi ne raidžių simboliai (įskaitant ir skaitmenis). Kadangi buvo pastebėta, kad subtitrų tekstynas turi sakinių tarp kurių žodžių nėra tarpų bei įvertinus, kad lietuvių kalboje nėra gausu žodžių ilgesnių nei 20 simbolių, išimami žodžiai ilgesni nei 20 raidžių.

Iš viso tekstiniai duomenys sudaro 4,16 mln. eilučių bei 31,54 mln. žodžių iš kurių 493 tūkst. yra unikalūs. Vidutiniškai vieną eilutę sudaro 7,59 žodžių. Žodžių turinčių bent vieną diakritinį ženklą yra 11,35 mln. ir tai sudaro 35,9% visų žodžių.

8 lentelė. Bendro tekstyno statistikos santrauka

Tekstiniai duomenys	Eilučių kiekis	Žodžių kiekis	Žodžių kiekis su diakritikais	Žodžių su diakritikais procentas	Skirtingų žodžių kiekis	Vidutinis žodžių kiekis eilutėje
	4157482	31544155	11352384	35,9%	492513	7,59

Teksto duomenis iš viso sudaro apie 185 mln. raidžių. Žemiau lentelėje pateikiama kiekvienos raidės procentinė išraiška nuo visų raidžių. Daugiausiai pasikartojanti raidė – i, kuri sudaro 13,7% po jos sekanti raidė – a, sudaranti 12,4%. Dažniausiai pasikartojanti priebalsė – s, sudaranti 7,9%, o populiariausi diakritiniai ženklai – ė ir š, atitinkamai sudarantys 1,5% ir 1,3%.

9 lentelė. Procentinis raidžių pasiskirstymas tekstiniuose duomenyse

Raidė	%	Raidė	%	Raidė	%	Raidė	%
a	12,40	f	0,17	n	4,58	ū	0,61
ą	0,79	g	1,71	o	5,30	v	2,00
b	1,71	h	0,05	p	3,07	y	1,57
c	0,33	i	13,65	r	5,62	z	0,16
č	0,46	į	0,64	s	7,90	ž	0,75
d	2,54	j	1,87	š	1,30	q	0,0004
e	5,80	k	4,81	t	6,99	w	0,004
ę	0,20	l	2,79	u	4,75	x	0,02
ė	1,51	m	3,91	ų	1,13		

2.7.4. Diakritinių ženklų atstatymui reikalingų duomenų paruošimas

Mokymui yra reikalingos dvi imtys – įvesties tekstiniai duomenys be diakritinių ženklų, pateikiami modeliui, bei tvarkingas ir taisyklingas, su diakritiniais ženklais esantis išvesties tekstas, kurį modelis grąžina. Įvesties tekstiniams duomenims sugeneruoti naudojamas Naplava ir kt. [27] straipsnyje naudotas „Bash“ komandų kalbos atviras kodas, kuris diakritinį simbolį paverčia į ASCII koduotės simbolį – anglų kalbos abėcėlės raidę. Konvertuojami lietuvių kalbos diakritiniai ženklai ir jų atitikmenys nurodomi žemiau lentelėje. Taip pat, jei tekste pasitaiko, naudojamas atviras kodas taip pat paverčia ir kirčių simbolius ar kitų kalbų raides (à, í, ü ir pan.) į atitinkamus jų anglų abėcėlės atitikmenis.

10 lentelė. Diakritinių ženklų konvertavimo atitikmenys

Diakritinis ženklas	Atitikmuo	Diakritinis ženklas	Atitikmuo
ą	a	š	s
č	c	ų	u
ę	e	ū	u
ė	e	ž	z
į	i		

Po raidžių konvertavimo gaunami du tekstinių duomenų rinkiniai, kurie naudojami modelių apmokymui. Pateikiama keletas įvesties ir išvesties teksto pavyzdžių:

11 lentelė. Įvesties ir išvesties teksto pavyzdžiai

Įvesties tekstas	**Išvesties tekstas**
kitaip nors vieno is pateiktu konkreciu ir matyt logisku reikalavimu neivykdzius yra logiska kad skrydziai yra uzdraudziami be kazkokio atskiro ir politinio sprendimo nes kalbama jau apie techninius reikalavimus kurie turi buti ivykdyti	kitaip nors vieno iš pateiktų konkrečių ir matyt logiškų reikalavimų neįvykdžius yra logiška kad skrydžiai yra uždraudžiami be kažkokio atskiro ir politinio sprendimo nes kalbama jau apie techninius reikalavimus kurie turi būti įvykdyti
ka	ką
makaronai	makaronai
greiciau	greičiau
turi ji paleisti	turi jį paleisti

3. Rezultatai

Šioje darbo dalyje aprašomi žingsniai diakritinių ženklų atstatymo ir sentimento analizės modeliams sukurti, juos apmokyti ir įvertinti jų gerumą. Taip pat išanalizuojami ir aptariami gauti rezultatai.

3.1. Naudojamos programinės įrangos aprašymas

Seq2Seq diakritinių ženklų atstatymo modelis kuriamas naudojant *Python* programinę kalbą (versija – 3.7.12), *Tensorflow* atviro kodo biblioteką (versija – 2.6.3) bei papildomą pagalbinę *Tensorflow* biblioteką – *TensorFlow SIG Addons* (versija – 0.14.0). Stankevičiaus ir Lukoševičiaus [98] *ByT5* apmokytą lietuvių kalbos gramatikos taisymo modelį sureguliuoti diakritinių ženklų atstatymo užduočiai naudojama *PyTorch* atviro kodo bibliotekos implementacija *Hugging Face transformers* bibliotekai (versija – 4.18.0).

Modeliams apmokyti naudojama *kaggle.com* siūloma nemokama prieiga prie *NVidia* K80 grafinio procesoriaus vieneto (GPU) su 16GB operatyviąja atmintimi bei 13GB operatyviosios atminties centrinio procesoriaus (CPU). Vienos sesijos laikas ribojamas iki 12 valandų, todėl testuojant ir treniruojant modelius buvo saugojamos jų tarpinės versijos.

3.2. Diakritinių ženklų atstatymo modeliai

Diakritinių ženklų atstatymo modeliui sukurti naudojamos dvi natūralios kalbos apdorojimo ir vertimo giliojo mokymosi architektūros – *Seq2Seq* bei transformerio tipo modelis – *ByT5*.

3.2.1. *Seq2Seq* modelis

Seq2Seq diakritinių ženklų atstatymo modelis kuriamas ženklo lygmeniu. Sakiniai yra skaidomi juos sudarančius ženklus, kiekvienam ženklui priskiriant atitinkamą skaitmenį (tokeną) ir fiksuojant atitinkamo ženklo skaitmenį atskirame žodyne. Dėl operatyvios atminties ribojimų, modeliui apmokyti naudojami sakiniai ne ilgesni nei 75 ženklų ilgis. Šis skaičius pasirodė optimalus eksperimentuojant įvairiais sakinių ilgiais, kol pasiektas optimalus mokymosi laiko trukmė ir rezultatas.

Atmetus ilgesnius sakinius, duomenų rinkinį sudaro 3,26 mln. eilučių sakinių. Taip pat atsitiktiniu būdu 10 proc. duomenų imties buvo atskirta raidės tikslumo ir diakritinio ženklo tikslumo nustatymui skirta testavimo imtis. Galiausiai, modelis apmokomas ant 2,93 mln. sakinių.

Kuriant modelį buvo testuojamos *Seq2Seq* architektūros tiek su GRU sluoksniais kartu su Bahdanau dėmesio sluoksniu, tiek su LSTM sluoksniais. Taipogi su LSTM sluoksnių modeliu buvo išbandyti ir Bahdanau bei Luong dėmesio sluoksniai. Įvertinus tai, kad GRU sluoksnių *Seq2Seq* modelį užkrauti apmokymui pavyko tik su mažesniu vienos partijos (angl. *batch*) dydžiu, automatiškai prailgėjo tokio tipo modelio apmokymo laikas, lyginant su LSTM. Galiausiai, optimalų rezultatą, vertinant tiek apmokymo trukmę, tiek nuostolio (angl. *loss*) metriką, parodė LSTM sluoksnių *Seq2Seq* modelis su Luong dėmesio sluoksniu.

Galutiniame modelyje buvo nustatytas 256 vienetų partijos dydis (angl. *batch size*), vektorizavimo dimensijų kiekis (angl. *embedding dimensions*) – 128, neuronų kiekis užkuoduotojo ir dekoduotojo atvejais – 1028. Mokymosi optimizavimui naudojamas standartinis „Adam“ optimizatorius su 0,001 mokymosi norma (angl. *learning rate*).

12 lentelė. *Seq2Seq* modelio pagrindiniai apmokymo parametrai

Parametras	**Parametro dydis/nustatymas**
Batch size	256
Embedding dimensions	128
Encoder/decoder units	1028
Optimizer	Adam
Learning rate	0,001

Galutinis *Seq2Seq* modelis su Luong dėmesio sluoksniu buvo apmokomas 7 epochas. Iš viso mokymas užtruko ~12 val. Nuostolio metrika siekė 0,0065.

Raidės ir diakritinio ženklo tikslumo metrikos, dėl gan lėto diakritinių ženklų atstatymo trukmės, buvo testuojama ne su pilna testavimo imtimi, tačiau su 20000 testavimo imties sakinių. Raidės tikslumo metrika siekė 98,12%, o konkrečiai diakritinio ženklo atstatymo tikslumas – 93,71%. Galima daryti išvadą, kad modelis gan tiksliai atstato ne diakritinius ženklus, tačiau konkrečiai tose vietose, kuriose yra diakritiniai ženklai tikslumo gali ir pritrūkti.

13 lentelė. *Seq2Seq* modelio raidės ir diakritinio ženklo atstatymo tikslumas

Tikslumo metrika	**Tikslumas**
Raidės atstatymo tikslumas	98,12%
Diakritinio ženklo atstatymo tikslumas	93,71%

Tai galima įžvelgti ir žemiau lentelėje pateiktuose testavimo imties įvesties ir po diakritinių ženklų atstatymo esančiuose išvesties sakiniuose. Kaip galima matyti viename iš pavyzdžių, modelis sukūrė netgi papildomo triukšmo žodyje *šūvius* (pavertė į *šūviuš*) arba žodis *pozicija* buvo paverstas į *požicija*.

Įsitikitinti ar modelis įvertina aplinkinių žodžių kontekstą galima pasitelkiant tokius žodžius, kurie be diakritinių ženklų ir su diakritiniais ženklais absoliučiai keičia savo konotaciją. Pavyzdžiui: *sunelis – šunelis* arba *sūnelis*, *karstas – karštas*. Lentelėje paskutiniai du sakiniai būtent tai ir testuoja. Galima pastebėti, kad sakinyje *sūnelis* yra atskiriamas nuo žodžio *šunelis* teisingai, tačiau sakinyje žodis *karstas* yra sugeneruojamas kaip *karštas* abejais kontekstų atvejais. Tai parodo, kad modelis aplinkinių žodžių konteksto iki galo neįvertina ir tai galimai įvyksta, kadangi yra naudojamas raidžių lygmens, o ne žodžių lygmens diakritinių ženklų atstatymo modelis.

14 lentelė. Diakritinių ženklų atstatymo pavyzdžiai naudojant *Seq2Seq* modelį

Įvedamas tekstas	**Išvedamas tekstas**
as siulyciau minuciu pertrauka	aš siūlyčiau minučių pertrauką
mamyte dar turi apziureti daug kitu namu	mamyte dar turi apžiūrėti daug kitų namų
beje jie vyksta siauliuose lietuvos mieste	beje jie vyksta šiauliuose lietuvos mieste
kada jau bijosite iseiti is namu tada pradesite spresti as manau	kada jau bijosite išeiti iš namų tada pradėsite spręsti aš manau
kiekvienas turite po penkis suvius	kiekvienas turite penkis šūviuš

turbut man uztektu nueiti paziureti vaidinimu	turbūt man užtektų nueiti pažiūrėti vaidinimų
tie asmenys perreze gerkles tukstanciams	tie asmenys perrezė gerklės tūkstančiams
kitaip sakant likes neuzbrauktas zodelis reiskia seimo nario pozicija	kitaip sakant likęs neužbrauktas žodelis reiškia seimo nario požicija
bet saknys yra visai ne ten kur jus galvojate ne monopolyje o mokesciuose	bet šaknys yra visai ne ten kur jūs galvojate ne monopolyje o mokesčiuose
as esu sunelis jonas ir mano sunelis brisius garsiai loja	aš esu sūnelis jonas ir mano šunelis brisius garsiai loja
karstas maistas garuoja ant stalo kol medinis karstas guli kapinese	karštas maistas garuoja ant stalo kol medinis karštas guli kapinėse

Taigi, įvertinus *Seq2Seq* modelį su LSTM sluoksniais ir Luong dėmesio sluoksniu, galima teigti, kad jis gan aukštu tikslumu sugeba atstatyti diakritinius ženklus. Tačiau išskirtiniais atvejais galima pastebėti, kad modelis kai kuriose vietose įveda netgi papildomo triukšmo, dėl kurio žodžiai apskritai prarandą savo tikrąją reikšmę.

3.2.2. *ByT5* modelis

ByT5 modelis kuriamas naudojant Stankevičiaus ir Lukoševičiaus [98] lietuviško teksto gramatinių klaidų taisymui skirtą modelį bei prie jo pridėtą atvirą kodą. Modelis sureguliuojamas (angl. *fine-tune*) naudojant turimą duomenų rinkinį diakritinių ženklų atstatymui. Autoriai savąjį *ByT5* modelį apmokė 100 valandų.

Pagal Stankevičiaus ir Lukoševičiaus [98] straipsnyje pateiktas rekomendacijas, apmokymo duomenų rinkinys buvo atfiltruotas, kad jame nebūtų sakinių, ilgesnių nei 700 simbolių. Taip apmokymui panaudojami 4,16 mln. sakinių, iš kurių 0,5% atskiriami validavimo bei 0,5% galutiniam testavimui – raidės ir diakritinio ženklo atstatymo tikslumui paskaičiuoti.

Duomenys tokenizuojami baito lygmeniu pagal autorių apmokytą modelį. Taip pat mokymui naudojamas toks pats, kaip ir autorių straipsnyje bei atvirame kode nurodytas partijos dydis lygus 1 ir *Adafactor* optimizatorius su 0,001 mokymosi norma.

15 lentelė. *ByT5* modelio pagrindiniai apmokymo parametrai

Parametras	**Parametro dydis/nustatymas**
Batch size	1
Optimizer	Adafactor
Learning rate	0,001

Modelis buvo mokomas 12 val. Per šį laiką buvo įvykdyta 0,12 epochos, tačiau ir tai jau davė gan gerą 0,009 nuostolio metrikos įvertį bei 0,005 nuostolio metrikos įvertį validavimo imčiai. Validavimo imties geresnis rezultatas pasiektas dėl, lyginant su mokymo imtimi, maža validavimo imtimi (20787 vienetai). Vis dėlto, matuojant raidės tikslumą su 20000 testavimo imties sakiniais, pasiektas aukštas 99,65% tikslumas bei 98,52% diakritinio ženklo atstatymo tikslumas. Lyginant su *Seq2Seq* modeliu, galima įžvelgti, kad diakritinio ženklo atstatymo tikslumo skirtumas yra gana reikšmingas. *ByT5* modelis pranašesnis dar ir tuo, kad gali atstatyti žymiai ilgesnius sakinius.

16 lentelė. *ByT5* modelio raidės ir diakritinio ženklo atstatymo tikslumas

Tikslumo metrika	**Tikslumas**
Raidės atstatymo tikslumas	99,65%
Diakritinio ženklo atstatymo tikslumas	98,52%

Nagrinėjant žemiau lentelėje pateiktus, atsitiktinai iš testavimo imties atrinktus, sakinių bei jų diakritinių ženklų atstatymo pavyzdžius, galima įžvelgti, kad modelis tiksliai sugeba atstatyti ženklus net ir labai ilguose sakiniuose. Testuojant tokius žodžius, kurie be diakritinių ženklų ir su diakritiniais ženklais pakeičia prasmę, paskutiniuose dviejuose lentelės pavyzdžiuose žodis *karstas* į *karstas* arba *karštas* buvo pagal kontekstą atstatytas tinkamai, tačiau *sunelis* atstatytas į *sūnelis* arba *šunelis* nesėkmingai. Lyginant su *Seq2Seq* modeliu, sėkmingai išvestas sakinio pavyzdys buvo priešingas.

17 lentelė. Diakritinių ženklų atstatymo pavyzdžiai naudojant *ByT5* modelį

Įvedamas tekstas	**Išvedamas tekstas**
uzsirasydamas pries as noriu pritarti prezidento dekretui nes prezidentas atkreipe demesi kad seimas dar karta daro broka ir bent jau pasiule iki birzelio d nukelti sio istatymo nuostatos isigaliojima tam kad seimas dar karta pasiziuretu ar europos sajungos direktyvos is tikruju yra taip suprastos ir ar tikrai reikia dvigubo patikrinimo tiek prie iejimo i losimo automatu salonus tiek keiciant pinigus	užsirašydamas prieš aš noriu pritarti prezidento dekretui nes prezidentas atkreipė dėmesį kad seimas dar kartą daro broką ir bent jau pasiūlė iki birželio d nukelti šio įstatymo nuostatos įsigaliojimą tam kad seimas dar kartą pasižiūrėtų ar europos sąjungos direktyvos iš tikrųjų yra taip suprastos ir ar tikrai reikia dvigubo patikrinimo tiek prie įėjimo į lošimo automatų salonus tiek keičiant pinigus
deja siandien mes buvom vienokiu ar kitokiu statuto straipsniu nuostatu priversti tiksliau jus issireikalavote jus tai padarete balsuoti atviru balsavimu nors tas pats seimo kancleris gerbiamas j	deja šiandien mes buvom vienokių ar kitokių statuto straipsnių nuostatų priversti tiksliau jūs išsireikalavote jūs tai padarėte balsuoti atviru balsavimu nors tas pats seimo kancleris gerbiamas j
ji mate ir dare tai ko kiti nesuprato todel jie eme jos bijoti	ji matė ir darė tai ko kiti nesuprato todėl jie ėmė jos bijoti
tik turedami toki palanku pasaulini fona ir musu tvirtas nuostatas vesti derybas kaip lygus partneriai galime lemti teigiamas derybu isvadas	tik turėdami tokį palankų pasaulinį foną ir mūsų tvirtas nuostatas vesti derybas kaip lygūs partneriai galime lemti teigiamas derybų išvadas
nebeturiu kur grizti	nebeturiu kur grįžti
dievuleliau tau butina viska zinoti	dievulėliau tau būtina viską žinoti
na taip as suprantu kad yra ka tenkina ir sitaip toliau elgsis tie kurie sitaip sako	na taip aš suprantu kad yra ką tenkina ir šitaip toliau elgsis tie kurie šitaip sako
ar yra priestaravimu	ar yra prieštaravimų
dar vienas zingsnis ir as toliau nuo namu nei kada nors buvau	dar vienas žingsnis ir aš toliau nuo namų nei kada nors buvau
isgerti ir atsvesti	išgerti ir atšvęsti
as esu sunelis jonas ir mano sunelis brisius garsiai loja	aš esu šunelis jonas ir mano šunelis brisius garsiai loja
karstas maistas garuoja ant stalo kol medinis karstas guli kapinese	karštas maistas garuoja ant stalo kol medinis karstas guli kapinėse

Lyginant *Seq2Seq* ir *ByT5* diakritinių ženklų atstatymo modelius, *ByT5* yra pranašesnis, kadangi turi aukštesnį raidės atstatymo tikslumą ir dar geresnį diakritinio ženklo atstatymo tikslumą. Taipogi, *ByT5* leidžia sėkmingai atstatyti diakritinius ženklus žymiai ilgesniam įvesties tekstui. Tiesa, pats atstatymo procesas yra beveik dvigubai spartesnis *Seq2Seq* modelio atveju. Pavyzdžiui, atstatyti diakritinius ženklus sentimento analizei naudojamiems atsiliepimams *Seq2Seq* modelis užtruko 2 val.

48 min., o *ByT5* modelio atveju – 4 val. 42 min. Vis dėlto, reikėtų įvertinti ir tai, kad *ByT5* modelis buvo apmokytas tik 0,12 epochos, tad galima numanyti, kad pratęsus šio modelio apmokymo laiką, raidės ir diakritinio ženklo atstatymo tikslumas galėtų dar pakilti.

3.3. Duomenų rinkinio sentimento analizei žvalgomoji analizė

Lietuviškų atsiliepimų rinkinį, kuris naudojamas sentimento klasifikavimo uždaviniui, sudaro atsiliepimai apie įvairias įmones bei įstaigas iš socialinio tinklo „Facebook“ bei atsiliepimai apie internetines parduotuves iš puslapio *evertink.lt*. Šie duomenys buvo surinkti ir analizuojami magistro baigiamuosiuose projektuose [92] [93]. Iš viso rinkinyje sukaupti 18539 vartotojų atsiliepimai, tarp kurių 66% yra surinkti iš *evertink.lt* ir 44% – „Facebook“. Kiekvienas atsiliepimo tekstas taip pat turi nurodytą atributą „bad“, kuris nusako ar šis atsiliepimas turi teigiamą ar neigiamą/neutralią konotaciją. Neigiamą/neutralų poliariškumą turinčių atsiliepimų yra 4693 (25,3%), o teigiamą poliariškumą – 13846 (74,7%). Žemiau pateikiama po vieną blogo/neutralaus ir teigiamo atsiliepimo pavyzdį iš turimo duomenų rinkinio:

18 lentelė. Atsiliepimų duomenų rinkinyje pavyzdžiai

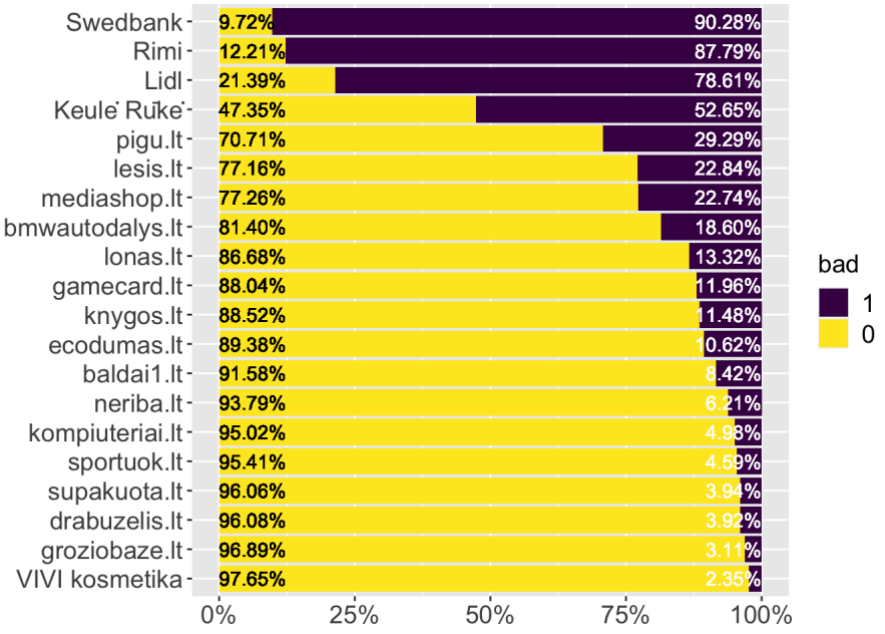
Atsiliepimas	**Atsiliepimo klasė (1 – blogas/neutralus, 0 – geras)**
Pamirštat senas kainas nulupt, kai akcijas darot :D	1
Linkėjimai virtuvei! Labai skanu!	0

Duomenų rinkinyje daugiausia atsiliepimų turi *pigu.lt, knygos.lt* bei *neriba.lt* internetinės parduotuvės. Iš „Facebook“ socialinio tinklo daugiausia atsiliepimų turi „Lidl“.

19 lentelė. Daugiausia duomenų rinkinyje atsiliepimų turinčios įmonės

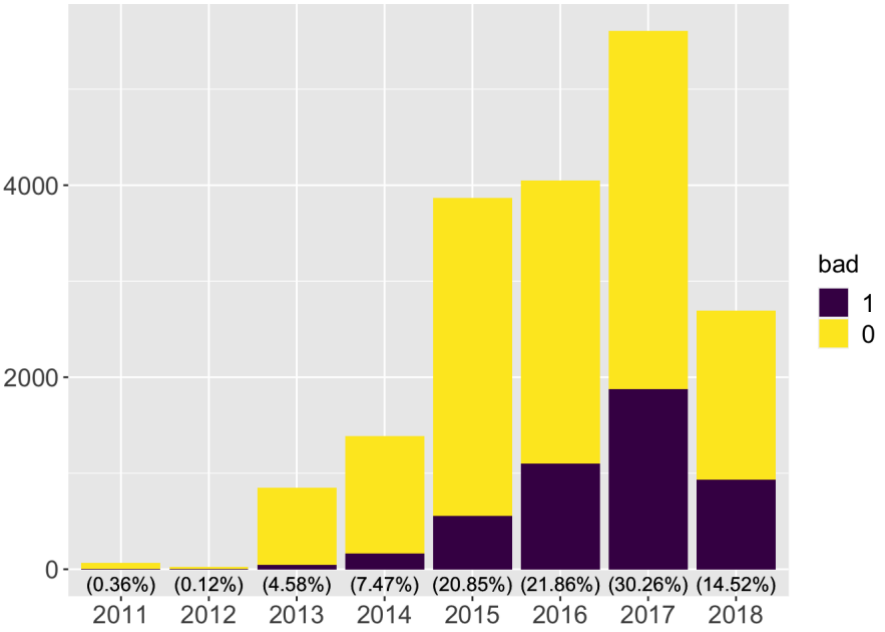
Šaltinis	**Puslapis/įmonė**	**Atsiliepimų kiekis**
Evertink	pigu.lt	3431
Evertink	knygos.lt	1698
Evertink	neriba.lt	1610
Evertink	Sportuok.lt	1090
Evertink	bmwautodalys.lt	784
Evertink	lonas.lt	548
Evertink	mediashop.lt	392
Facebook	Lidl	360
Evertink	gamecard.lt	326
Evertink	ecodumas.lt	320

Žemiau pateiktame paveikslėlyje parodoma, kurios įmonės ir internetinės parduotuvės susilaukė procentiškai daugiausiai neigiamų bei teigiamų atsiliepimų (1 – nurodo neigiamos/neutralios konotacijos atsiliepimą, 0 – teigiamos). Galima įžvelgti, kad procentiškai, daugiausiai neigiamų atsiliepimų rinkinyje turi „Swedbank“ (90,3%), po to seka prekybos tinklai „Rimi“ bei „Lidl“. Daugiausia teigiamų įvertinimų turi „VIVI kosmetika“ (96,7%) bei internetinės parduotuvės *drabuzelis.lt* (96,9%) ir *groziobaze.lt* (96,1%).



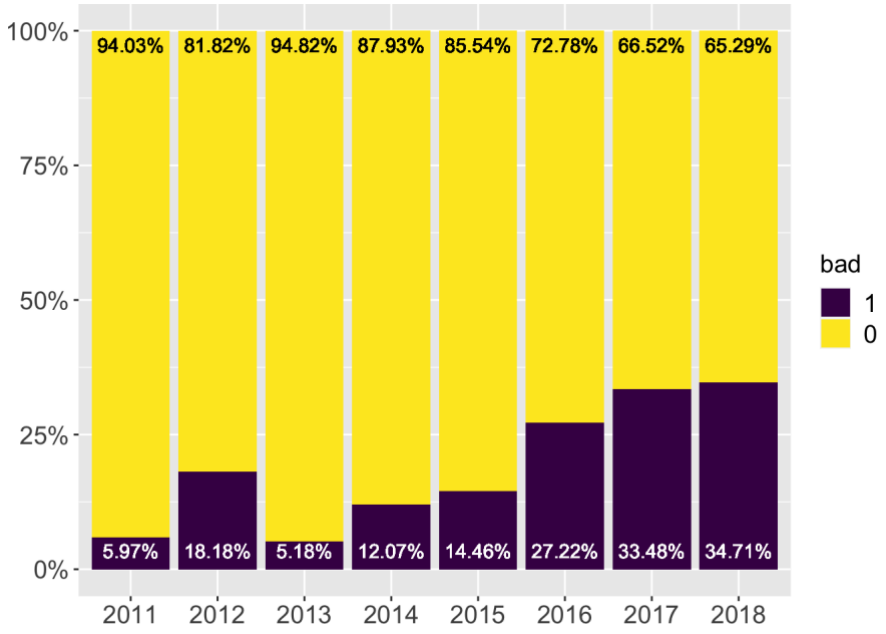
12 pav. Daugiausiai ir mažiausiai teigiamų atsiliepimų turinčių įmonių pasiskirstymas

Nagrinėjant atsiliepimus pagal metus, galima įžvelgti, kad daugiausiai atsiliepimų surinkta iš 2017 metų (30,3%). Šiek tiek mažiau atsiliepimų sudaro 2016 (21,9%) bei 2015 (20,9%) metais parašyti atsiliepimai bei įvertinimai.



13 pav. Atsiliepimai pagal jų pateikimo metus

Iš paveikslėlio žemiau galima įžvelgti, kad visais metais daugiausia atsiliepimų yra turintys teigiamą poliariškumą. Lyginant su ankstesniais metais, paskutinius 3 metus neigiamų/neutralių atsiliepimų procentas yra išaugęs iki ~30%.



14 pav. Teigiamų ir neigiamų/neutralių atsiliepimų balansas pagal metus

Nors ir teigiamų atsiliepimų yra daugiau, disbalansas nėra toks jau ir didelis ir duomenys yra tinkami naudoti sentimento analizei. Taip pat galima įvertinti, kad duomenys yra pateikti iš įvairių šaltinių bei apie įvairių sričių įmones, kurios jau tarpusavy gali turėti tam tikrą nusistovėjusių teigiamų/neigiamų atsiliepimų balansą ir duomenų rinkinys atitinka realybę.

3.3.1. Tekstinių duomenų paruošimas ir diakritinių ženklų atstatymas

Kadangi turimi atsiliepimų duomenys yra gana triukšmingi (turi daug ne raidinių simbolių, šypsenėlių, jaustukų, internetinių nuorodų). Išvalant tekstą yra pašalinami simboliai, kurie yra ne raidės, pašalinami papildomi tarpai, pakeičiamos šypsenėlės ar kiti jaustukai į tą emociją išreiškiančio atitikmens anglų kalbos žodinį atitikmenį, nuimami diakritiniai ženklai, pakeičiamos piniginės išraiškos, datos, elektroniniai paštai, nuorodos, grotažymės į atitinkamus juos apibūdinančius žodžius.

Išvalytas tekstas yra naudojamas diakritinių ženklų atstatymo modeliuose. *Seq2Seq* modeliui atsiliepimas pagal pilnus žodžius yra skaidomas į dalis ne ilgesnes nei 75 simboliai, tuo tarpu *ByT5* modeliui skaidoma į dalis ne ilgesnes nei 700 simbolių. Žemiau lentelėje pateikiami keli standartiškai išvalyto, po diakritinių ženklų atstatymo sugeneruoto naudojant *Seq2Seq* ir *ByT5* modelius atsiliepimų pavyzdžiai. Juose galima įžvelgti, kad *ByT5* diakritinių ženklų atstatymas yra tikslesnis, kadangi nesukuriama papildomo triukšmo, kaip tai matoma *Seq2Seq* modelio pavyzdyje (*wc* pakeičiama į *wč, circle* į *čirčle, teve atleisk* tampa *teveisk*).

20 lentelė. Atsiliepimų teksto sutvarkymo pavyzdžiai

Originalus atsiliepimo tekstas	**Išvalytas atsiliepimo tekstas**	***Seq2Seq* modelio diakritikų ženklų atstatymo tekstas**	***ByT5* modelio diakritinių ženklų atstatymo tekstas**
Pamiršt senas kainas nulupt, kai akcijas darot :D	pamirstat senas kainas nulupt kai akcijas darot laughing	pamiršt senas kainas nulupt kai akcijas darot laughing	pamiršt senas kainas nulupt kai akcijas darot laughing
Na mums teko apžiūrėti viską kelis kartus,viskas labai patiko,ir kainos prieinamos. Žadame pirkti sofą ☺	na mums teko apziureti viska kelis kartus viskas labai patiko ir kainos prieinamos zadame pirkti sofa smiley	na mums teko apžiūrėti viską kelis kartus viskas labai patiko ir kainos prieinamos žadame pirkti sofą smiley	na mums teko apžiūrėti viską kelis kartus viskas labai patiko ir kainos prieinamos žadame pirkti sofą smiley
Teko šiandien apsilankyti Plungės Circle K. Deja nauja sistema prie WC nenudžiugino.. Ji apsunkino pirkėjų, sustojusių kuro ar skaniosios kavos ar užkandžių, nuėjimą į wc.. eilės, neveikianti ‚touch and pay' sistema.. Tai nepasirodė panašu į Jūsų nuolatinį orientavimasi į klientą... Teks dabar tą degalinę iškeist į kitą – tikiuos nebus visur tokių naujovių ;)	teko siandien apsilankyti plunges circle k deja nauja sistema prie wc nenudziugino ji apsunkino pirkeju sustojusiu kuro ar skaniosios kavos ar uzkandziu nuejima i wc eiles neveikianti touch and pay sistema tai nepasirode panasu i jusu nuolatini orientavimasi i klienta teks dabar ta degaline iskeist i kita tikiuos nebus visur tokiu naujoviu wink	teko šiandien apsįlankyti plungės čirčle k deja nauja sistema prie wč nenudžiugino ji apsunkino pirkėjų sustojusių kuro ar skaniosios kavos ar užkandžių nuėjimą į wc eilės neveikianti touch and pay sistemą tai nepasirodė panašu į jūsų nuolatavimasi į klientavimasi į klientavimas ta degalinė iškeist į kitą tikiuos nebus visur tokių naujovių wink	teko šiandien apsilankyti plungės circle k deja nauja sistema prie wc nenudžiugino ji apsunkino pirkėjų sustojusių kuro ar skaniosios kavos ar užkandžių nuėjimą į wc eilės neveikiantį touch and pay sistemą tai nepasirodė panašu į jūsų nuolatinį orientavimąsi į klientą teks dabar tą degalinę iškeist į kitą tikiuos nebus visur tokių naujovių wink
„Tėve, atleisk jiems, nes jie nežino, ką darą"	teve atleisk jiems nes jie nezino ka dara	tėveisk jiems nes jie nežino ką dara	tėve atleisk jiems nes jie nežino ką dara
Nu kiek galima tyčiotis iš pirkėjų Lazdynuose, Architektų Rimi?Kiek dar laiko egzistuos šitas supuvusių daržovių -vaisių skyrius?? Kiek gali tai tęstis?	Nu kiek galima tyciotis is pirkeju lazdynuose architektu rimi kiek dar laiko egzistuos sitas supuvusiu darzoviu vaisiu skyrius kiek gali tai testis	nu kiek galima tyčiotis iš pirkėjų lazdynuose architektų rimi kiek dar laiko egzistuos šitas supuvusių daržovių vaisių skyrius kiek gali tai tęstis	nu kiek galima tyčiotis iš pirkėjų lazdynuose architektų rimi kiek dar laiko egzistuos šitas supuvusių daržovių vaisių skyrius kiek gali tai tęstis

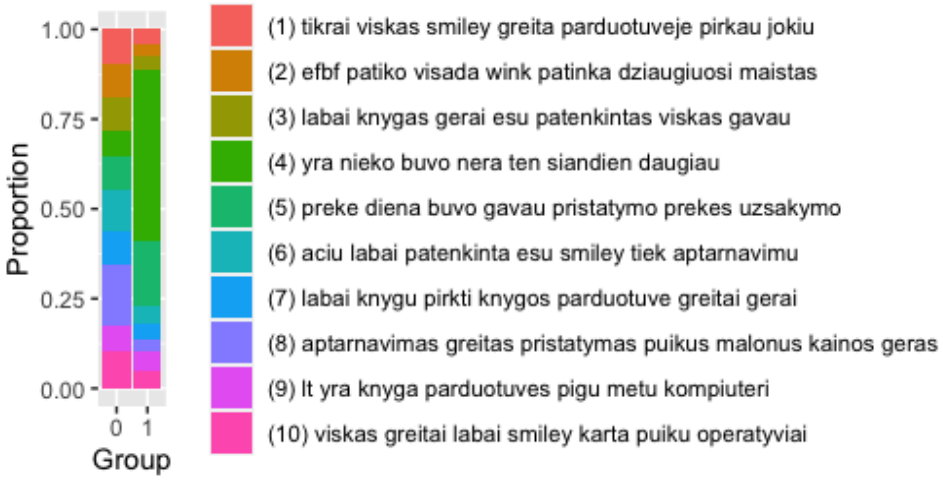
3.3.2. Temų modeliavimas naudojant LDA

Temų modeliavimas daromas norint įžvelgti tekstyną jungiančias temas. Taip pat, šio darbo tikslu apžvelgiami skirtumai tarp temų susiformavimo naudojant švarų ir abejais diakritinių ženklų atstatymo modeliais. Naudojamas LDA (Latentinio Dirichlė pasiskirstymo) metodas. Apibrėžiami atitinkami parametrai, kurie nurodo, kad nebūtų naudojami reti (rečiau nei 11 kartų pasikartojantys) žodžiai, nurodomas minimalus žodžio ilgis – 2 simboliai, išmetami bendriniai, konteksto neapibrėžiantys žodžiai (angl. *stop-words*).

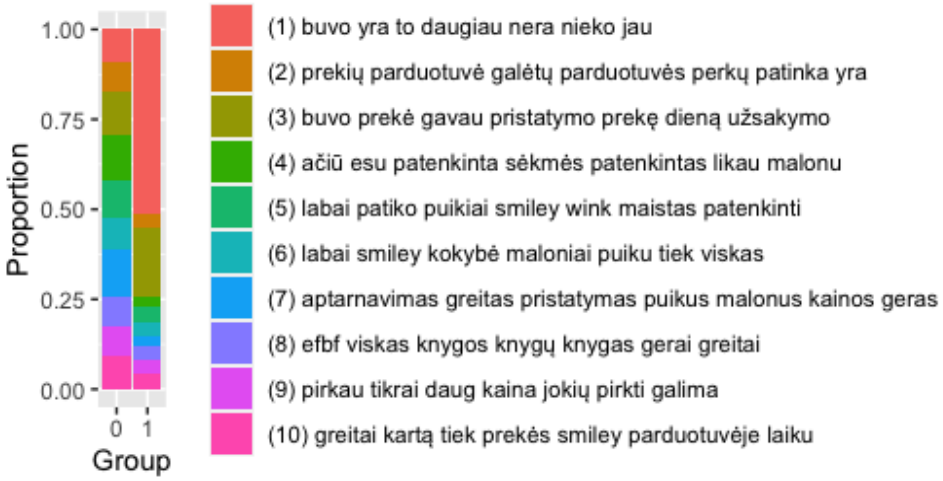
Žemiau pavaizduotuose grafikuose atvaizduojama 10 temų bei pagrindiniai 7 žodžiai kiekvienoje temoje. Juose galima įžvelgti, kad pagal žodžių prasmę aiškiai atskiriamų temų nėra. Taipogi, galima teigti, kad pateiktos pagrindinės temos yra pastebimos abejose sentimento klasėse. Tiesa, standartiškai išvalyto teksto atveju didelę dalį neigiamos sentimento klasės sudaro atsiliepimai turintys žodžius (4 tema): *yra, nieko, nera, ten siandien, daugiau*. *Seq2Seq* modeliu atstatytų

diakritinių ženklų teksto duomenų rinkinyje taipogi neigiamo sentimento klasėje didelė dalis sudaro tema su panašiais žodžiais (1 tema): *buvo, yra, nėra, daugiau*. *ByT5* modeliu sutvarkytame tekste, neigiamoje sentimento klasėje gan lygiomis proporcijomis atsiskiria 2 temos (3 ir 8), kurios tiek tarpusavyje, tiek lyginant su kitais tekstyno variantais yra gana panašios.

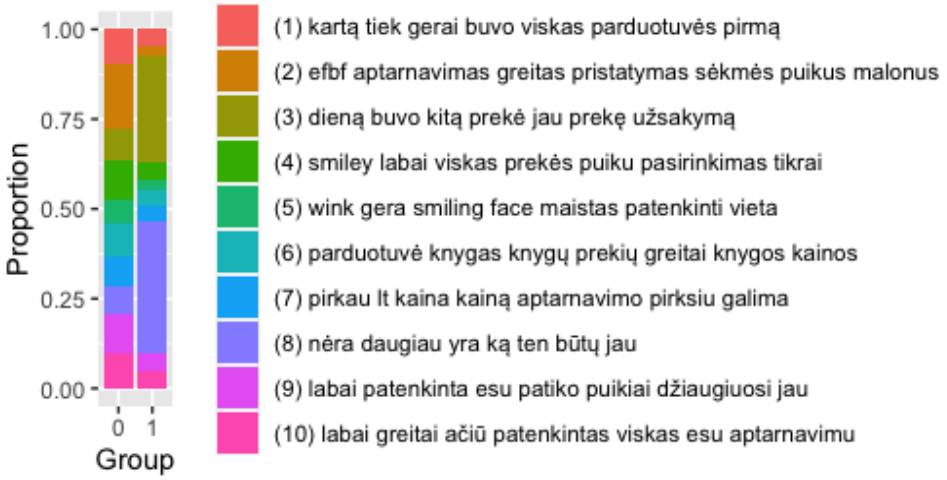
Iš temų modeliavimo metodo rezultatų, galima įžvelgti, kad turimas duomenų tekstynas ir juose esantys atsiliepimai turi daug panašių ir dažnai pasikartojančių žodžių. Dėl to aiškiai neatsiskiria konkrečios ar aiškiai interpretuojamos temos. Taipogi, tokių pačių temų yra tiek tarp neigiamos/neutralios klasės atsiliepimų, tiek tarp teigiamos klasės atsiliepimų. Galima daryti išvadą, kad toks interpretacinis metodas nėra pakankamas aiškiai ir tiksliai nustatyti sentimento klasę, todėl toliau sentimento klasifikavimo uždaviniui išbandomi mašininio bei giliojo mokymosi modeliai.



15 pav. Standartiškai išvalyto teksto temos pagal sentimento klasę



16 pav. *Seq2Seq* atstatytų diakritikų teksto temos pagal sentimento klasę



17 pav. *ByT5* atstatytų diakritinių ženklų teksto temos pagal sentimento klasę

3.4. Sentimento klasifikavimo modeliai

Sentimento klasifikavimas daromas naudojant *R* programinę kalbą ir jos bibliotekas su atsitiktinių miškų metodu (*tuneRanger* bibliotekas), logistinę regresiją. Tuo pačiu atliekamas eksperimentinis variantas sureguliuojant *ByT5* modelį sentimento klasifikavimo užduočiai. Visi modeliai apmokomi naudojant skirtingas duomenų imtis: standartiškai išvalytas tekstas su nuimtais diakritiniais ženklais, tekstas su atstatytais diakritiniais ženklais naudojant *Seq2Seq* modelį, tekstas su atstatytais diakritiniais ženklais naudojant *ByT5* modelį.

3.4.1. Duomenų imties paruošimas modelių apmokymui

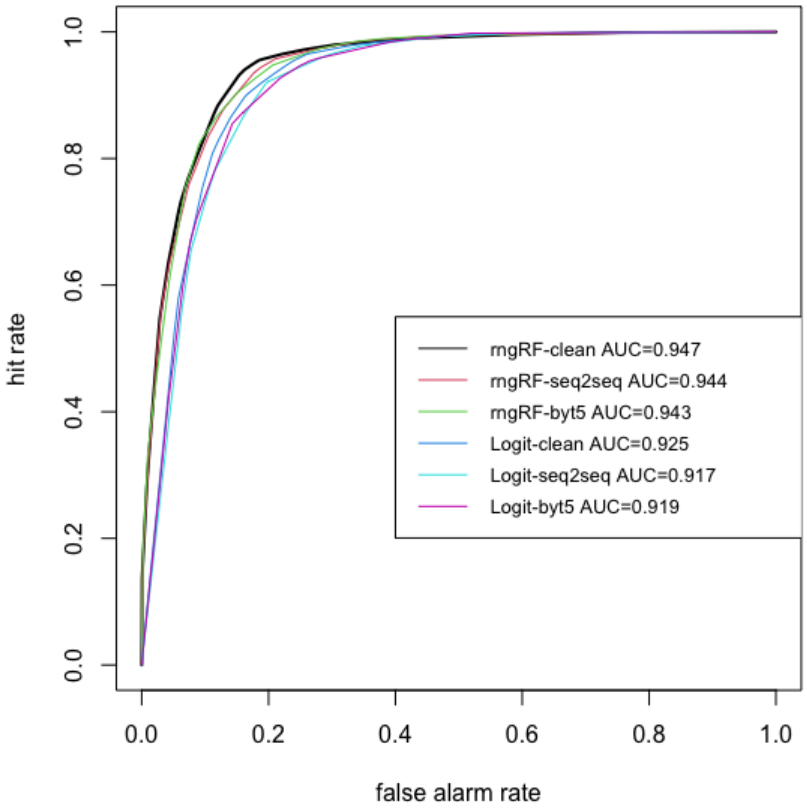
Testinei imčiai naudojami visi 2018 metų atsiliepimai (2691), likusių kitų metų atsiliepimai (15848) naudojami mokymo imčiai.

Logistinės regresijos bei atsitiktinių miškų modeliams duomenys vektorizuojami naudojant latentinį semantinį indeksavimą (LSI). Gaunama matrica kur viena eilutė reprezentuoja vieną turimų tekstinių duomenų atsiliepimą su stulpeliuose esančiais atitinkamo dimensialumo vektoriais ir sentimento klasės atributu, kurį modeliai prognozuos (1 – blogas/neutralus atsiliepimas, 0 – teigiamas atsiliepimas). Turint omeny, kad atstačius diakritinius ženklus, skirtingų žodžių ir jų reikšmių gali padaugėti, šiuose eksperimentuose išbandomi ir aprašomi 32, 64 ir 128 dimensionalumo variantai visiems sugeneruotiems tekstinių duomenų rinkiniams.

3.4.2. 32 dimensijų vektorizavimo modeliai

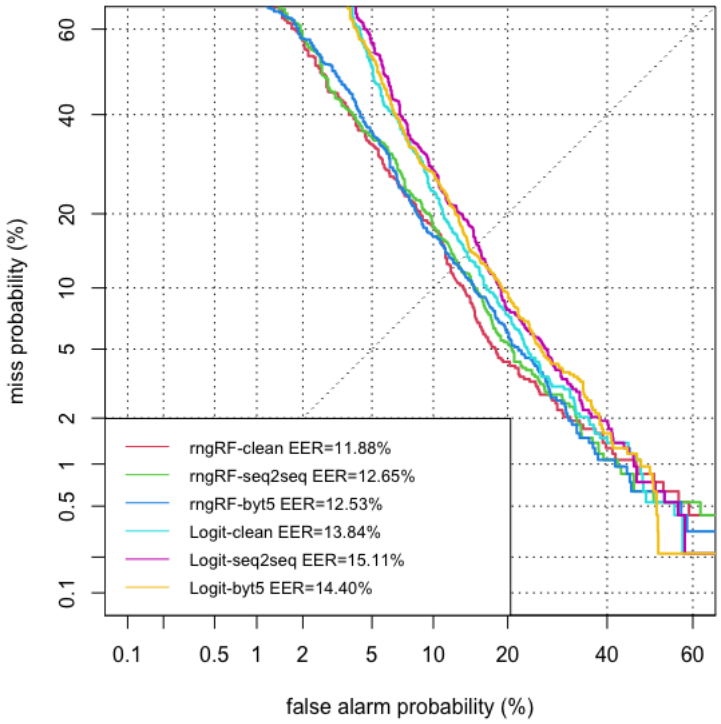
32 dimensijų logistinės regresijos ir atsitiktinio miško modelius užtruko apmokyti 13 minučių. Beveik visas šis laikas buvo skirtas apmokyti atsitiktinio miško modelius. Geriausią rezultatą, vertinant AUC metriką pasiekė atsitiktinio miško modelis su standartiškai išvalytu tekstu (0,947). Atsitiktinio miško modelių atveju nei viena skirtingai sutvarkytų tekstų modelių kombinacija nėra tarpusavyje statistiškai reikšmingai besiskirianti pagal AUC įvertį (p reikšmė > 0,05). Logistinės regresijos modeliai parodė prastesnį rezultatą ir statistiškai reikšmingai skiriasi nuo visų atsitiktinio miško modelių rezultato. Geriausiai logistinės regresijos atveju pasirodęs standartiškai išvalyto modelio

rezultatas taip pat yra statistiškai reikšmingai besiskiriantis nuo *Seq2Seq* sugeneruotu tekstu, kuris šiuo atveju pasirodė prasčiausiai.



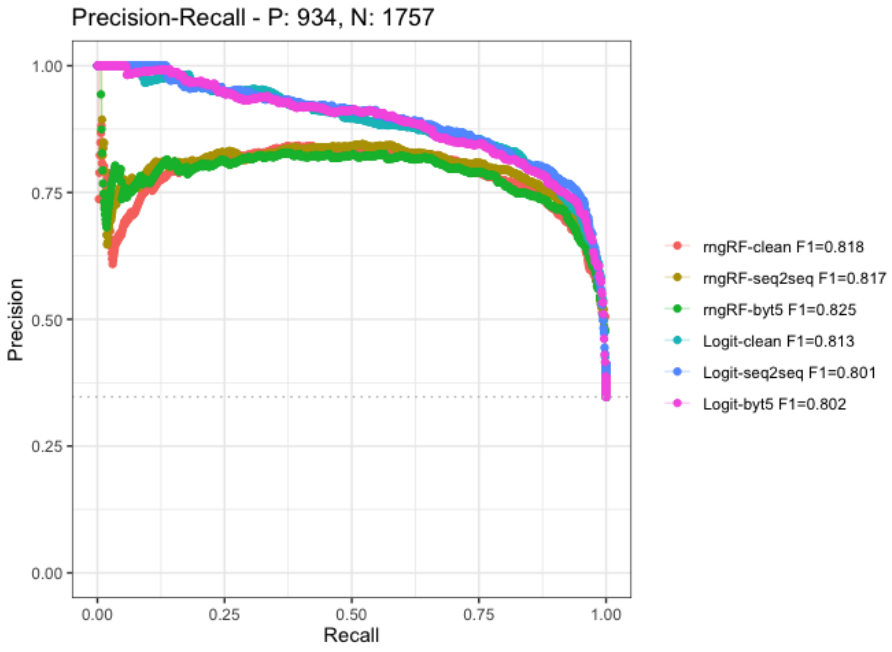
18 pav. ROC kreivės naudojant 32 dimensijų vektorizavimą

19 pav. pateiktas DET grafikas, kurio legendoje nurodomas EER įvertis prognozuojant teigiamą bei neigiamą klases. Pagal mažiausią EER procentą, pirmauja tas pats modelis ir tendencijos yra tokios pačios kaip ir AUC įverčio atveju. Iš grafiko dar galima įžvelgti ir tai, kad logistinės regresijos modelių atveju jie gan stipriai atsiskiria neteisingai prognozuojant neigiamą klasę.



19 pav. DET grafikas naudojant 32 dimensijų vektorizavimą

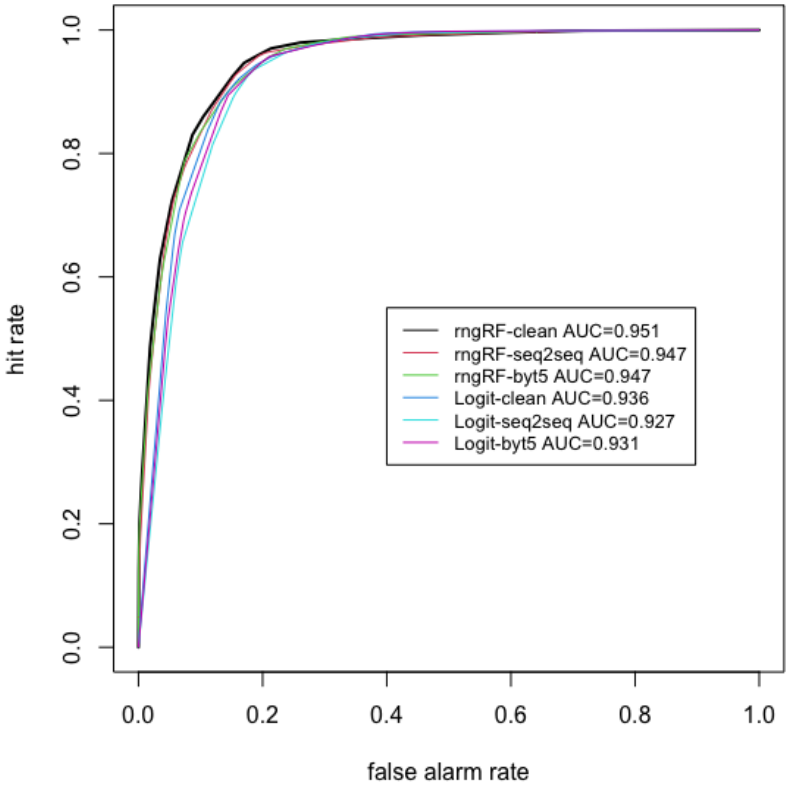
Galiausiai, 20 pav. pateiktas PR – tikslumą bei atkuriamumą vaizduojantis grafikas su šių dviejų metrikų kombinacijos metrika – F-įvertis. Šiuo atveju geriausią įverčio rezultatą pasiekė atsitiktinio miško modelis su *ByT5* atstatytais diakritiniais ženklais, kas rodo, kad šis modelis yra geriausias prognozuojant teigiamą klasę (šiuo atveju – neigiamus/neutralius atsiliepimus).



20 pav. PR kreivės naudojant 32 dimensijų vektorizavimą

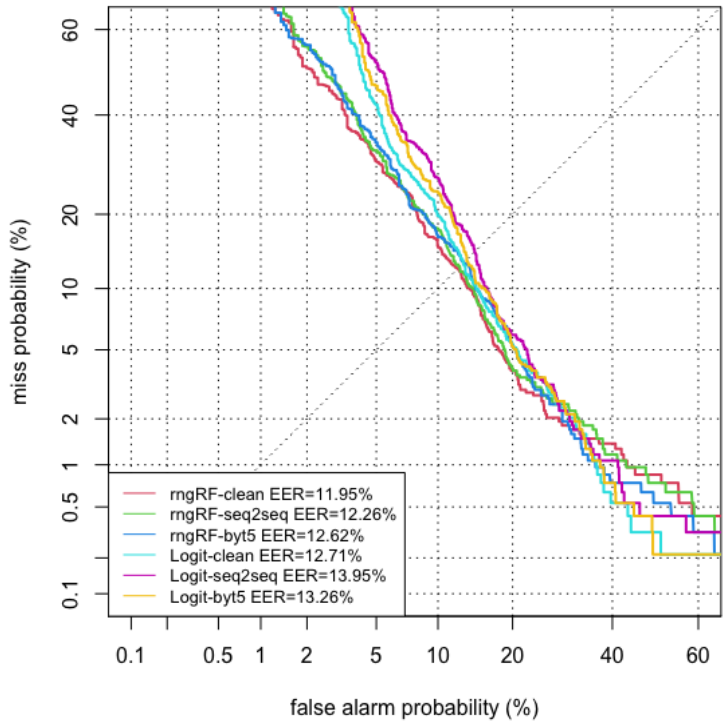
3.4.3. 64 dimensijų vektorizavimo modeliai

64 dimensijų logistinės regresijos ir atsitiktinio miško modelius apmokyti užtruko 27 minutes. Šio dimensialumo modeliai visais atvejais pasiekė geresnį rezultatą nei su 32 dimensijų vektorizavimo. Geriausią rezultatą pagal AUC įvertį pasiekė atsitiktinio miško modelis su standartiškai išvalyto teksto duomenimis (0,951). Antras geriausias rezultatas pasiektas taip pat šio tipo modeliu su teksto duomenų rinkiniu, kuriame diakritiniai ženklai atstatyti naudojant *ByT5* modelį (0,947). Galiausiai, identišką AUC įverčio rezultatą parodė modelis su *Seq2Seq* tekstu (0,947). Įdomu tai, kad pastarieji du įverčiai su geriausią rezultatą parodžiusiu modeliu naudojant 32 dimensijų vektorizavimą. Vis dėlto šiuo atveju, lyginant standartiškai išvalyto teksto modelio AUC įvertį su atstatytų diakritinių ženklų tekstu, šie atvejai skiriasi statistiškai reikšmingai. *rngRF-clean*, lyginant su *rngRF-seq2seq* AUC įverčio skirtumo p reikšmė yra 0,03, o *rngRF-clean* ir *rngRF-byt5* p reikšmė lygi 0,02. Vertinant logistinės regresijos modelius, visų tipų tekstynų modeliai vėlgi pasiekė žemesnio tikslumo rezultatus, o žemiausias rezultatas pasiektas taip pat su *Seq2Seq* sugeneruotu tekstu.



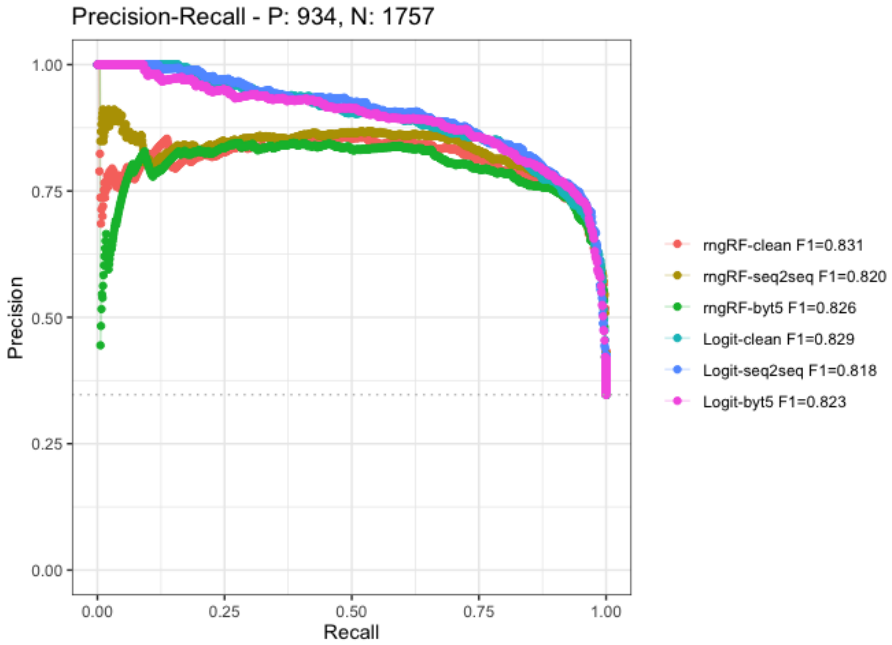
21 pav. ROC kreivės naudojant 64 dimensijų vektorizavimą

Pagal mažiausią EER procentą pirmauja tas pats modelis, kaip ir AUC atveju. Tiesa, atsitiktinio miško modelis su *Seq2Seq* diakritinių ženklų atstatymu čia pasiekia geresnį rezultatą nei *ByT5*, nors logistinės regresijos modelio atveju *Seq2Seq* modeliu tvarkytas tekstas turi didžiausia EER procentą. Vėlgi, logistinės regresijos modelių gan ryškus atsiskyrimas neteisingai prognozuojant neigiamą klasę yra įžvelgiamas ir šio dimensionalumo atveju.



22 pav. DET grafikas naudojant 64 dimensijų vektorizavimą

Geriausias F-įvertis pasiektas atsitiktinio miško standartinio teksto išvalymo atveju. Taipogi gerą rezultatą parodė ir logistinė regresija naudojant šiuos duomenis.



23 pav. PR kreivės naudojant 64 dimensijų vektorizavimą

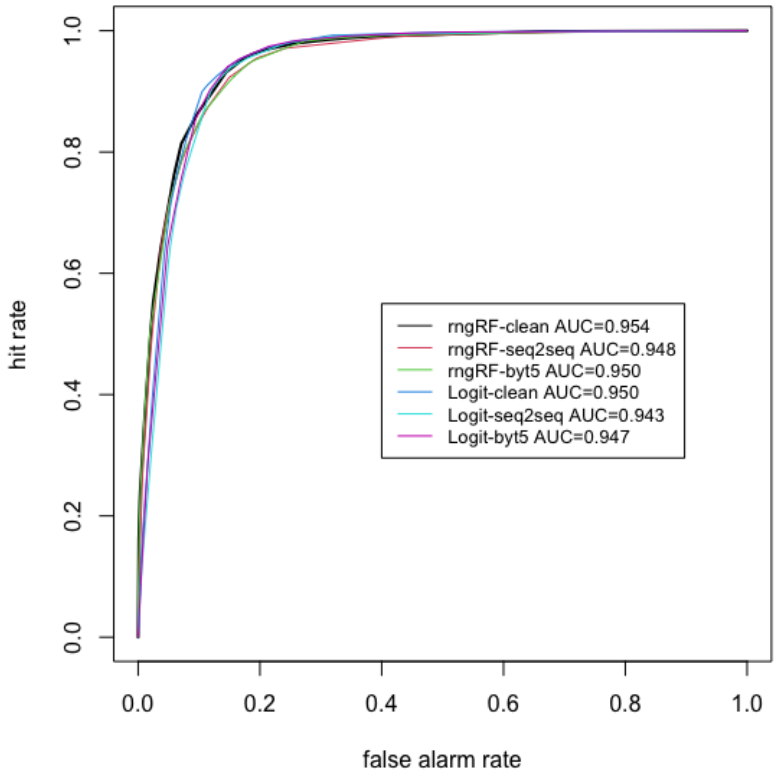
Lyginant geriausius abiejų dimensionalumo modelius, jų AUC įverčio rezultatas skyriasi statistiškai reikšmingai (p reikšmė – 0,004). Tiek 32, tiek 64 dimensijų vektorizavimo atveju diakritinių ženklų

atstatymas nepasitvirtino, kadangi geriausią rezultatą vertinant visus pateiktus grafikus bei įverčius pasiekė modeliai naudoję standartiškai išvalytą tekstą, o pasiektas rezultatas tik susilygino su mažesnio dimensionalumo geriausiu modeliu. Tarpusavyje lyginant diakritinių ženklų atstatymo modelius, rezultatai statistiškai reikšmingai nesiskiria.

3.4.4. 128 dimensijų vektorizavimo modeliai

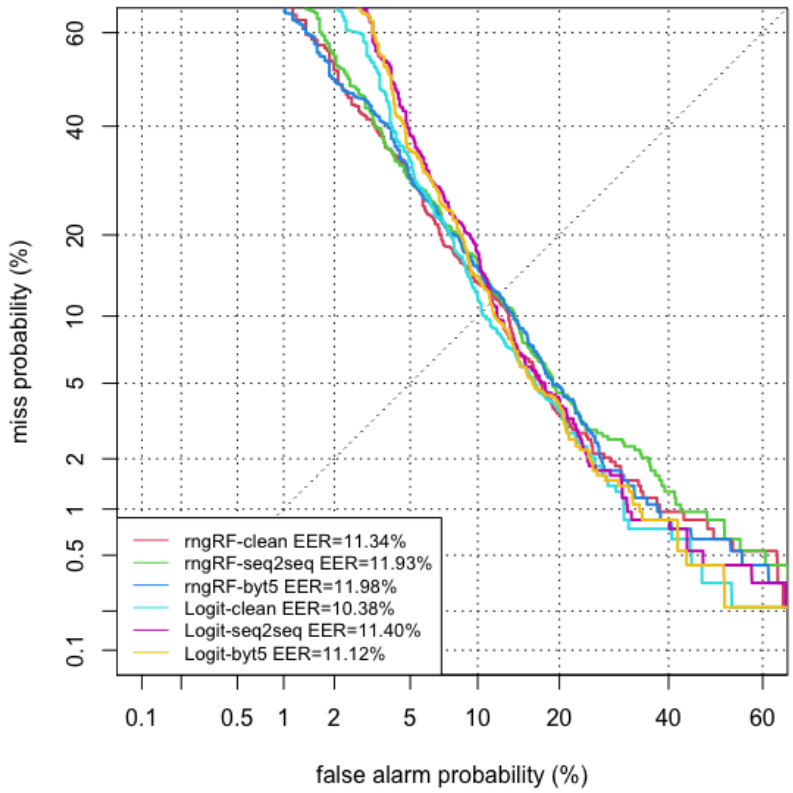
Dar padvigubinus duomenų vektorizavimo dimensionalumą atsitiktinio miško ir logistinės regresijos modelių apmokymo laikas padidėjo daugiau nei dvigubai ir užtruko 1,17 val.

Vertinant 24 pav. esančia ROC kreivę, geriausiu modeliu išliko tas pats atsitiktinio miško modelis su standartiškai išvalytu tekstu. Analizuojant AUC įvertį, galima pamatyti statistiškai reikšmingą pagerėjimą geriausių modelių atveju, lyginant su mažesniu dimensionalumu (p reikšmė – 0,007). Tiesa, aukštesnis dimensionalumas sumažino atotrūkį tarp *rngRF-clean* ir *rngRF-byt5* modelių ir jų AUC įvertis statistiškai reikšmingai nebesiskiria (p reikšmė – 0,056).



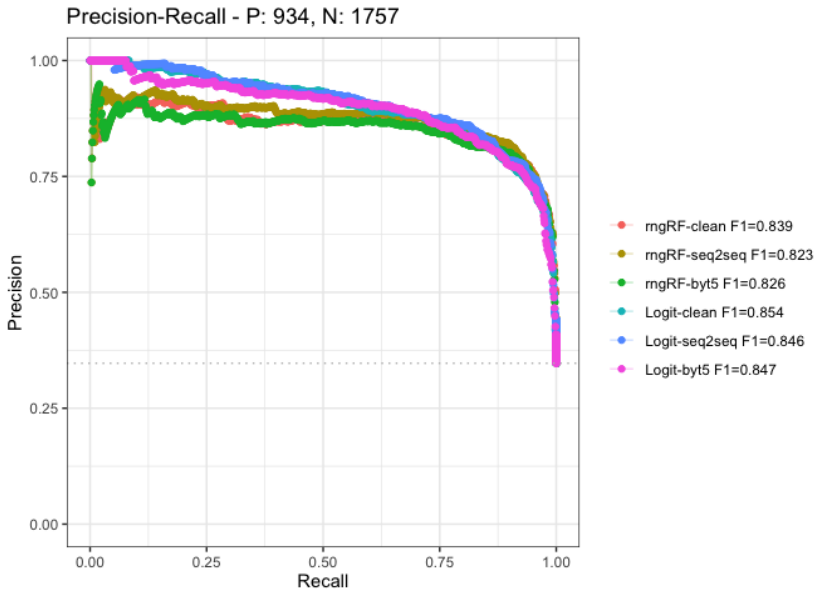
24 pav. ROC kreivės naudojant 128 dimensijų vektorizavimą

DET grafikas 25 pav. vaizduoja, kad dimensionalumo padidinimas taip pat pagerino šį įvertį. Geriausią rezultatą čia rodo jau logistinės regresijos modelis su standartiškai išvalytu tekstu. Vertinant atsitiktinio miško modelį su *Seq2Seq* sugeneruotu tekstu ir jo DET kreivę, galima įžvelgti, kad jis šiek tiek daugiau klysta prognozuodamas teigiamas reikšmes, lyginant su kitais modeliais.



25 pav. DET grafikas naudojant 128 dimensijų vektorizavimą

Padidinus dimensionalumą labiausiai pagerėjo F-įvertis. Šiuo atveju logistinės regresijos modeliai parodė netgi geresnį rezultatą. Apskritai vertinant PR grafikus galima įžvelgti, kad modeliai tiksliau prognozuoja teigiamas (neigiamo/neutralaus sentimento) reikšmes bei pasižymi aukštesniu atkuriamumo įverčiu.



26 pav. PR kreivės naudojant 128 dimensijų vektorizavimą

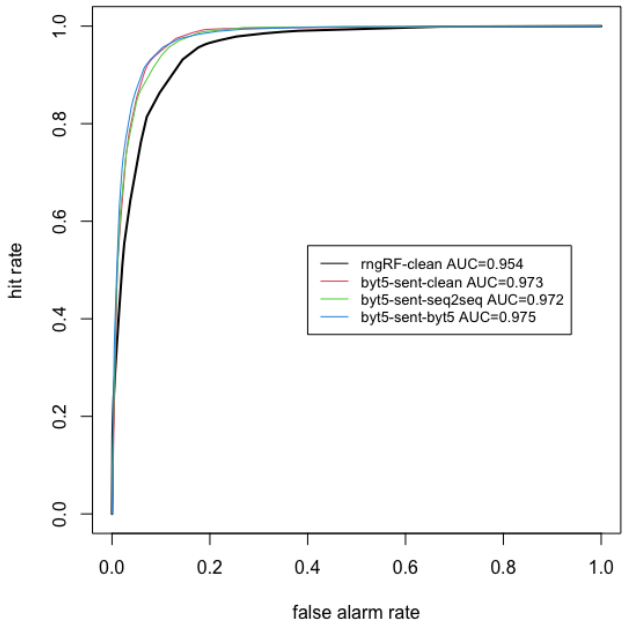
Apibendrinant, galima teigti, kad dimensionalumo padidinimas gerina visas aukščiau aptartas metrikas. Nors visuose dimensionalumo atvejuose geriausią rezultatą rodė standartinis teksto išvalymas, didžiausio išbandyto dimensionalumo atveju *ByT5* modeliu atstačius diakritinius ženklus, AUC įvertis statistiškai reikšmingai nebesiskyrė. Lyginant *Seq2Seq* ir *ByT5* teksto rezultatus, nors jie tarpusavy reikšmingai nesiskyrė, tačiau tik mažiausio dimensionalumo atveju *Seq2Seq* neryškiai aplenkė *ByT5*.

3.4.5. *ByT5* modelis sureguliuotas sentimento klasifikavimui

Nors *ByT5* yra „teksto į tekstą“ dekodavimo modelis, tačiau eksperimentiniu būdu jis išbandomas ir sentimento klasifikavimui. Modelis sureguliuojamas taip, kad padavus tekstą jis išvestų „y“ arba „n“ simbolius, kurie atitinkamai reiškia neigiamą/neutralią ir teigiamą sentimento konotaciją. Tam, kad tinkamai būtų apskaičiuojamos AUC įverčiai ir atvaizduojami grafikai, yra reikalingas klasifikavimo tikimybės įvertis kiekvienam stebėjimo prognozavimui. Tam buvo ištraukiamas atitinkamos prognozuotos raidės baito prognozės įvertis ir naudojant *softmax* funkciją nuo 0 iki 1 įvertinama ar *ByT5* modelio bus prognozuojamas „y“ raidės baitas, lyginant su „n“ baitu.

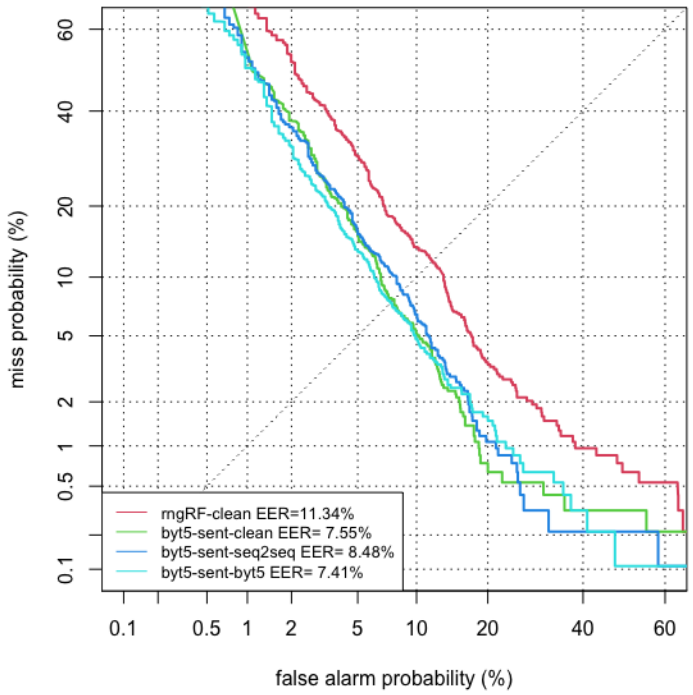
Tokiu būdu naudojant atsilepimų duomenų rinkinius su standartiškai išvalytu tekstu, *Seq2Seq* ir *ByT5* modeliais atstatytų diakritinių ženklų tekstu, 10-čiai epochų sureguliuojami 3 atskiri *ByT5* modeliai. Reguliavimas vykdytas tam pačiam Stankevičiaus ir Lukoševičiaus [98] prieš tai apmokytam modeliui. Kiekvieno sentimento klasifikavimui pritaikymo reguliavimas truko šiek tiek daugiau nei 3 val.

Šiuo atveju 27 pav. atvaizduota ROC kreivė parodo, kad AUC metrikos rezultatas yra pagerinamas statistiškai reikšmingai (p reikšmė arti 0), lyginant AUC įverčius net su geriausiu atsitiktinio miško modeliu (128 dimensionalumo *rngRF-clean*). Taip pat šio modelio atveju pasiteisino ir diakritinių ženklų atstatymas, kadangi AUC įvertis yra šiek tiek didesnis, nors ir ne statistiškai reikšmingai (p reikšmė 0,18), lyginant sureguliavimą su *ByT5* tekstu ir standartiškai išvalytu tekstu. Nors *Seq2Seq* teksto modelis vėlgi pasirodė prasčiausiai, rezultatai supanašėjo ir AUC įverčio nėra reikšmingai skirtingi (p reikšmė 0,07, lyginant su *byt5-sent-byt5* ir 0,74 su *byt5-sent-clean*).



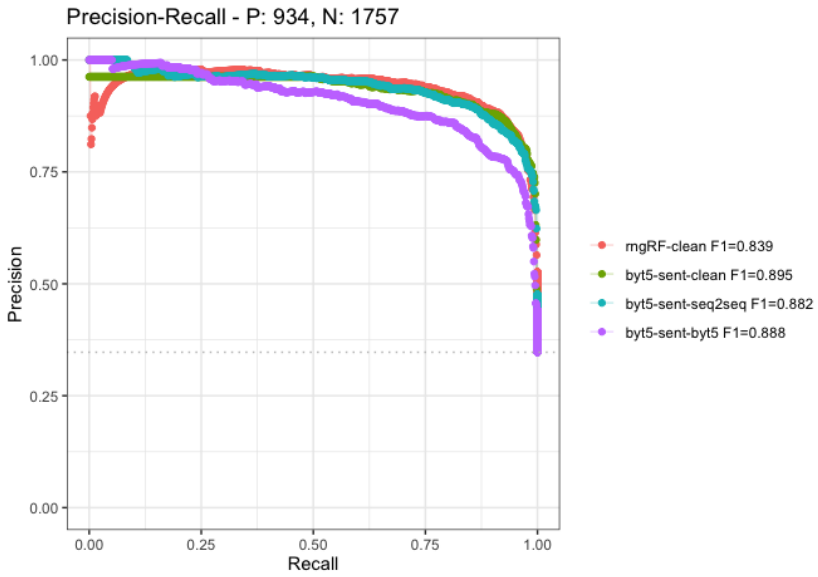
27 pav. ROC kreivės naudojant sureguliuotą *ByT5* sentimento klasifikavimui

EER metrika taip pat žemiausia *ByT5* diakritinių ženklų atstatymo atveju, tuo tarpu, kaip ir dauguma atveju, *Seq2Seq* tekstas pasirodo prasčiausiai to paties tipo modelio klasėje. Akivaizdus skirtumas pastebimas visus *ByT5* sureguliuotus modelius lyginant su geriausiu atsitiktinio miško modelio variantu



28 pav. DET grafikas naudojant sureguliuotą *ByT5* sentimento klasifikavimui

Vis dėlto, F-įvertis geresnis yra standartiškai išvalyto teksto atveju. Šis modelis tiksliau prognozuoja teigiamą klasę, ką galima pastebėti žemiau pateiktose modelių sumaišymo matricose. Aukštas atkuriamumo įvertis, kuris vertina, ar tiksliai prognozuojama teigiama klasė, yra žymiai aukštesnis standartiškai išvalyto teksto atveju (92,83). Tuo tarpu *ByT5* modeliu aptvarkyto teksto modelis šį įvertį turi – 87,9. Taigi, esant poreikiui tiksliai atskirti teigiamą klasę (neigiamus/neutralius atsiliepimus), verčiau būtų naudoti *byt5-sent-clean* modelį. Bendriniam sentimento klasifikavimui tikslesnis modelis – *byt5*-sent-*byt5*.



29 pav. PR kreivės naudojant sureguliuotą *ByT5* sentimento klasifikavimui

21 lentelė. *byt5-sent-clean* modelio sumaišymo matrica

		Tikroji klasė	
		0	**1**
Prognozuojama klasė	**0**	1620	67
	1	137	867

22 lentelė. *byt5*-sent-*byt5* modelio sumaišymo matrica

		Tikroji klasė	
		0	**1**
Prognozuojama klasė	**0**	1663	113
	1	94	821

3.5. Apibendrinimas ir rekomendacijos

Apibendrinant rezultatus, galima teigti, kad diakritinių ženklų atstatymas akivaizdaus aptartų metrikų rezultatų nepagerino. Vis dėlto, sentimento klasifikavimui sureguliuotas *ByT5* modelis statistiškai reikšmingai pranoko standartinius mašininio mokymosi modelius (atsitiktinių miškų, logistinės regresijos) naudojamus sentimento klasifikavimui. Jo atveju ir diakritinių ženklų atstatymas esminiame AUC įvertyje parodė minimalų, nors ir ne statistiškai reikšmingą skirtumą, lyginant su

standartiškai išvalyto tekstu. Giliojo mokymosi metodai pasižymi didelio kiekio ir įvairovės duomenų reiklumu tam, kad būtų galima juos sėkmingai apmokyti. Šiuo atveju tekstas su diakritiniais ženklais šia įvairove pasižymi labiau nei tekstas be papildomų simbolių. Kai kurie žodžiai su diakritiniais ženklais įgauną kitą reikšmę ar suteikia papildomo konteksto todėl tokio tipo modelius yra verta bandyti apmokyti ir su realią rašomąją kalbą atitinkančiais duomenimis.

Vis dėlto galima įvertinti ir tai, kad *ByT5* ar kitų giliojo mokymosi modelių atveju apmokymo procesas trunka žymiai ilgiau ir yra reikalingi galingesni techniniai resursai, grafiniai procesoriai, o dauguma mašininių modelių galima gan greitai apmokyti ir su paprastu kompiuteriu.

Šio tyrimo tęstinumui pirmiausia būtų rekomenduojama transformerio tipo modelius tiek diakritinių ženklų atstatymui, tiek sentimento klasifikavimo uždaviniui apmokyti žymiai ilgesnį laiką. Tyrimo metu diakritikų atstatymo modelis buvo apmokytas 12 val. ir per tokį laiką sugebėta pasiekti dešimtadalį epochos. Tikėtinas ženklus tikslumo pagerėjimas būtų tikėtinas tokį modelį apmokant bent keletą epochų ar naudojant dar didesnę ir įvairesnės tekstinių duomenų imties.

Taipogi, *ByT5* modelį dar būtų galima tobulinti apjungiant tiek šiame tyrime naudotą duomenų rinkinį diakritinių ženklų atstatymui, tiek ir Stankevičiaus ir Lukoševičiaus [98] modelio idėją. Autorių modelis taiso praleistas raides, supainiotas raides ir kitas smulkias rašybos klaidas. Su tokiu modeliu sutvarkius turimą atsiliepimų duomenų rinkinį vertėtų pakartoti eksperimentus ir patikrinti ar toks teksto sutvarkymas neduoda reikšmingo rezultatų pagerėjimo

Kalbant apie sentimento klasifikavimo uždavinį, vertėtų pasigilinti ir į kitus transformerio tipo modelius, kurie yra labiau pritaikyti konkrečiai klasifikavimo tipo uždaviniui. Šiuo atveju, nors *ByT5* parodė gerą rezultatą, tačiau šis modelis yra skirtas generuoti tekstui, o klasifikavimui jis pritaikytas eksperimentiniu būdu grąžinant vieną simbolį. Literatūros apžvalgoje aptarti pavyzdžiai rodo, kad sentimento analizės uždaviniui yra gausu sėkmingų taikymų, naudojant *BERT* ar *XLNet* transfomerio tipo modelius. Vis dėlto, vertėtų tęsti sureguliavimą sentimento klasifikavimui ir su *ByT5* modeliu, kadangi, nors ir prasukus 10 epochų, ties kiekviena epochą buvo pastebimas nuostolio metrikos mažėjimas.

Galiausiai, ištobulintus abiejų užduočių modelius būtų galima padaryti viešai ir patogiai prieinamus bendrajam naudojimui tiek verslo įmonėms, tiek asmeniniam naudojimui. Su tokiais įrankiais bet kas norintis galėtų sutvarkyti triukšmingą tekstą, o sentimento klasifikavimo modelis gali būti sėkmingai integruojamas į reputacijos, nuomonių apie įmonės teikiamas paslaugas ar produktus, verslo vystymo ar investicinius procesus.

Išvados

1. Atlikus literatūros analizę buvo atrinkti populiariausi ir geriausiai veikiantys klaidų taisymo bei diakritinių ženklų atstatymo metodai ir modeliai. Nustatyta, kad transformerio modeliai yra dominuojantys savo rezultatais ir gan ryškiai lenkia senesnės architektūros *Seq2Seq* modelius. Taip pat, apžvelgti sentimento analizės panaudojimo pavyzdžiai, kurie parodė, kad jos taikymas turi apčiuopiamą naudą finansų ar verslo kūrimo bei vystymo srityse. Galiausiai, literatūros apžvalgoje nustatyta, kad NLP panaudojimo metodai lietuvių kalbai nėra labai gausūs, tačiau ši sritis vystosi ir pritaikymo pavyzdžių paskutiniu metu vis daugėja. Viena to priežasčių yra tai, kad naudojant didelius duomenų masyvus yra apmokomi ir viešai publikuojami transformerio modeliai, kurie yra lengvai ir nereikalaujant didelių resursų pritaikomi įvairioms kalboms.

2. Diakritinių ženklų atstatymą nuspręsta daryti naudojant senesnio tipo architektūros *Seq2Seq* modelį bei šiai užduočiai sureguliuotą Stankevičiaus ir Lukoševičiaus [98] prieš tai lietuvių kalbos rašybos taisymui apmokytą *ByT5* transformerio modelį.

3. Sentimento klasifikavimui ir rezultatų palyginimui pasirinkti keli skirtingi modeliai. Kaip bazinis ir paprasčiausias modelis – logistinė regresija, mašininio mokymosi – atsitiktinio miško modelis bei kaip naujausias metodas – eksperimentiniu būdu sentimento klasifikavimui sureguliuotas *ByT5* transformerio modelis.

4. Surinktas didelis lietuvių kalbos tekstinių duomenų rinkinys (4,16 mln. sakinių) apjungiant kino filmų subtitrų bei politikų pasisakymų Seime tekstynus. Šie tekstynai pasirinkti, kadangi juose gausu bendrinės, šnekamosios kalbos pavyzdžių. Tokio stiliaus kalba yra reikalinga, nes sentimento klasifikavimui naudojami vartotojų parašyti atsiliepimai taip pat dažniausiai yra rašomi šnekamąja kalba.

5. Sukurtas *Seq2Seq* modelis siekia 98,12% atstatymo tikslumą raidės lygmeniu ir 93,12% diakritinio ženklo lygmens tikslumą. Tuo tarpu *ByT5* modelis pranoko pastarąjį pasiekęs aukštą tikslumą tiek raidės (99,65%), tiek diakritinio ženklo lygmeniu (98,52%). Pastarasis modelis buvo apmokytas tik 12 val. ir prasuko šiek tiek daugiau nei 10% epochos, todėl tikėtina, kad tolimesnis tokio modelio apmokymas padėtų pasiekti dar geresnių rezultatų.

6. Panaudojus skirtingus sentimento klasifikavimo modelius su skirtingais metodais ir modeliais aptvarkytu tekstu, diakritinių ženklų atstatymas sentimento analizės rezultatų reikšmingai nepagerino (vertinant geriausių modelių AUC įverčio skirtumą, p reikšmė – 0,18). Vis dėlto, sentimento klasifikavimui sureguliuotas *ByT5*, naudojant to paties *ByT5* atstatytų diakritinių ženklų tekstą, pasiekė geriausią AUC įvertį (0,975), kuris yra statistiškai reikšmingai (p reikšmė arti 0) didesnis, lyginant su geriausiu atsitiktinio miško modelio variantu.

7. Sukurti diakritinių ženklų atstatymo modeliai davė tenkinamą rezultatą, ypatingai *ByT5* modelio atveju ir gali būti naudojami esant triukšmingo teksto sutvarkymo poreikiui. Nors su šiais modeliais atstačius diakritinius ženklus, sentimento klasifikavimas reikšmingo pagerėjimo nepasiekė, tyrimo metu išsiaiškinta, kad transformerio tipo modeliai sentimento klasifikavimo užduočiai veikia labai sėkmingai. Šiame tyrime pasiekto sentimento klasifikavimo tikslumo

modelis gali sėkmingai pasitarnauti įvairiems finansų, investicijų, verslo vystymo, klientų analizės ar reputacijos sekimo procesams tobulinti.

Literatūros sąrašas

1. Kapočiūtė-Dzikienė, J., Davidsonas, A., ir Vidugirienė, A. (2017). Character-Based Machine Learning vs. Language Modeling for Diacritics Restoration. *Journal of Information Technology and Control, 46*(4), 508-520.
2. Ravi, K., ir Ravi, V. (2015). A survey on opinion mining and sentiment analysis: Tasks, approaches and applications. *Knowledge-Based Systems, 89*, 14-46.
3. Whitelaw, C., Hutchinson, B., Chung, G., ir Ellis, G. (2009). Using the Web for Language Independent Spellchecking and Autocorrection. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, 890-899.
4. Belinkov, Y., ir Bisk, Y. (2018). Synthetic and natural noise both break neutral machine translation. *ICLR*.
5. Etoori, P., Chonnakotla, M., ir Mamidi, R. (2018). Automatic Spelling Correction for Resource-Scarce Languages using Deep Learning. *Proceedings of ACL 2018, Student Research Workshop*, 146-152.
6. Hasan, S., Heger, C., ir Mansour, S. (2015). Spelling Correction of User Search Queries Through Statistical Machine Translation. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 451-460.
7. Zhou, Y., Porwal, U., ir Konow, R. (2019). Spelling correction as a foreign language. *Proceedings of the SIGIR 2019 eCom workshop.*
8. Gupta, P. (2020). A context-sensitive real-time Spell Checker with language adaptability. *Conference: 2020 IEEE 14th International Conference on Semantic Computing (ICSC).*
9. Dunlop, M., ir Levine, J. (2012). Multidimensional Pareto Optimization od Touchscreen Keyboards for Speed, Familarity and Improved Spell Checking. *CHI 2012: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2669-2678.
10. Bi, X., Ouyang, T., ir Zhai, S. (2014). Both Complete and Correct? Multi-Objective Optimization of Touchscreen Keyboard. *CHI 2014: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, p. 2297-2306.
11. Lopez-Hernandez, J., Almela, A., ir Valencia-Garcia, R. (2019). Automatic Spelling Detection and Correction in the Medical Domain: A Systematic Literature Review. *Technologies and Innovation*, 95-109.
12. Hiscox, L., Leonavičiūtė, E., ir Humby, T. (2014). The Effects of Automatic Spelling Correction Software on Understanding and Comprehension in Compensated Dyslexia: Improved Recall Following Dictation. *Dyslexia, 20*, 208-224.
13. Neto, A., Leite Dantas Bezerra, B., ir Toselli, A. (2020). Towards the Natural Language Processing as Spelling Correction for Offline Handwritten Text Recognition Systems. *Recent Advances in Handwritten Text Recognition, 10*(21), 7711.
14. Gupta, J., Qin, Z., Bendersky, M., ir Metzler, D. (2019). Personalized Online Spell Correction for Personal Search. *Proceedings of the 2019 World Wide Web Conference (WWW'19).* San Francisco.
15. Neija, M., ir Yousfi, A. (2017). Context's Impact on the Automatix Spelling Correction. *Int. J. Artificial Intelligence and Soft Computing, 6*(1).

16. Büyük, O. (2020). Context-Dependent Sequence-to-Sequence Turkish Spelling Correction. *ACM Trans. Asian Low-Resour. Lang. Inf. Process, 19*(4).
17. Šantić, N., Šnajder, J., ir Bašić, B. (2009). Automatic Diacritics Restoration in Croatian Texts. *INFuture2009: "Digital Resources and Knowledge Sharing"*, 309-319.
18. Mihalcea, R. (2002). Diacritics Restoration: Learning from Letters versus Learning from Words. *CICLing 2002: Computational Linguistics and Intelligent Text Processing*, 339-348.
19. Tufiş, D., ir Ceauşu, A. (2007). DIAC+: A professional diacritics recovering system. *6th language resources and evaluation conference*, 167-174.
20. Richter, M., Stranak, P., ir Rosen, A. (2012). Korektor – A System for Contextual Spell-checking and Diacritics Completion. *Proceedings of COLING*, 1019-1028.
21. Novak, A., ir Siklósi, B. (2015). Automatic Diacritics Restoration for Hungarian. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2286-2291.
22. Hucko, A., ir Lacko, P. (2018). Diacritics Restoration using Deep Neural Networks. *2018 World Symposium on Digital Intelligence for Systems and Machines (DISA)*, 195-200.
23. Krstev, C., Stanković, R., ir Vitas, D. (2018). Knowledge and Rule-Based Diacrtitics Restoration in Serbian. *Proceedings of the Third International Conference Computational Linguistics in Bulgaria (CLIB 2018)*, 41-51.
24. Nutu, M., Lorincz, B., ir Stan, A. (2019). Deep Learning for Automatic Diacritics Restoration in Romanian. *Conference: IEEE 15th International Conference on Intelligent Computer Communication and Processing*, 235-240.
25. Iordache, F., Georgescu, L., Oneata, D., ir Cuhu, H. (2019). Romanian Automatic Diacritics Restoration Challenge. *Proceedings of the 14th international conference "linguistic resources and tools for natural language processing"*, 64-74.
26. Laki, L., ir Yang, Z. G. (2020). Automatic Diacritic Restoration With Transformer Model Based Neural Machine Translation for East-Central European Languages. *Proceedings of the 11th International Conference on Applied Informatics*, 190-202.
27. Náplava, J., Straka, M., Straňák, P., ir Hajič, J. (2018). Diacritics Restoration Using Neural Networks. *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 1566-1573.
28. Náplava, J., Straka, M., ir Straková, J. (2021). Diacritics Restoration using BERT with Analysis on Czech language . *The Prague Bulletin of Mathematical Linguistics*(116), 27-42.
29. Stankevičius, L., Lukoševičius, M., Kapočiūtė-Dzikienė, J., Briedienė, M., ir Krilavičius, T. (2022). Correcting Diacritics and Typos with a ByT5 Transformer Model. *Appl. Sci., 12*(5), 2636.
30. Klyshinsky, E., Karpik, O., ir Bondarenko, A. (2021). A Comparison of Neutral Networks Architectures for Diacritics Restoration. *AIST 2020: Recent Trends in Analysis of Images, Social Networks and Texts*, 242-253.
31. Masmoudi, A., Mdhaffar, S., Sellami, R., ir Hadrich Belguith, L. (2019). Automatic Diacritics Restoration for Tunisian Dialect. *ACM Trans. Asian Low-Resour. Lang. Inf. Process, 18*(3), 18.
32. Zitouni, I., ir Sarikaya, R. (2009). Arabic diacritic restoration approach based on maximum entropy models. *Computer Speech and Language, 23*, 257-276.

33. Hifny, Y. (2013). Restoration of Arabic Diacritics using Dynamic Programming. *2013 8th International Conference on Computer Engineering ir Systems (ICCES)*, 3-8.
34. El-Harby, A., El-Shehawey, M., ir Elbarougy, R. (2015). A Statistical Approach for Qur'an Vowel Restoration. *ICGST-AIML Journal, 8*(3).
35. Chennoufi, A., ir Mazroui, A. (2016). Morphological, syntactic and diacritics rules for automatic diacritization of Arabic sentences. *Journal of King Saud University – Computer and Information Sciencies, 29*(2), 156-163.
36. Hifny, Y. (2018). Hybrid LSTM/MaxEnt Networks for Arabic Syntactic Diacritics Restoration. *IEEE signal processing letters, 25*(10), 1515-1519.
37. Luu, T., ir Yamamoto, K. (2012). A Pointwise Approach for Vietnamese Diacritics Restoration. *Proceedings of the International Conference on Asian Language Processing*.
38. Trung, N., Nhan, N., ir Phuong, N. (2012). Vietnamese Diacritics Restoration as Sequential Tagging. *2012 IEEE RIVF International Conference on Computing ir Communication Technologies, Research, Innovation, and Vision for the Future*.
39. Nga, C., Thing, N., Chang, P., ir Wang, J. (2019). Deep Learning Based Vietnamese Diacritics Restoration. *IEEE International Symposium on Multimedia*, 331-334.
40. Cocks, J., ir Keegan, T. (2011). A word-based approach for diacritic restoration in Māori. *Proceedings of Australasian Language Technology Association Workshop*, 126-130.
41. Ezeani, I., Hepple, M., ir Onyenwe, I. (2016). Automatic Restoration of Diacritics for Igbo Language. *Text, Speech, and Dialogue (TSD 2016)*, 198-205.
42. Shaikh, H., Mahar, J., ir Mahar, M. (2017). Instant Diacritics Restoration System for Sindhi Accent Prediction using N-Gram and Memory-Based Learning Approaches. *International Journal of Advanced Computer Science and Applications, 8*(4), 149-157.
43. Onome Orife, I. (2018). Attentive Sequence-to-Sequence Learning for Diacritic Restoration of Yorub` a Language Text. *Conference Interspeech 2018*.
44. Alqahtan, S., Mishra, A., ir Diab, M. (2019). Efficient Convolutional Neural Networks for Diacritic Restoration. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, 1442-1448.
45. Cho, K., Marrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., ir Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation.
46. Sutskever, I., Vinyals, O., ir Le, Q. (2014). Sequence to Sequence Learning with Neural Networks.
47. Bahdanau, D., Cho, K., ir Bengio, Y. (2015). Neural Machine Translation by Jointly Learning to Align and Translate. *ICLR*.
48. Crego, J., Kim, J., Klein, G., ir Rebollo, A. i. (2016). SYSTRAN's Pure Neural Machine Translation Systems.
49. Lamba, D., ir Hsu, W. H. (2021). Constraint-Based Neural Question Generation Using Sequence-to-Sequence and Transformer Models for Privacy Policy Documents. *International Journal of Knowledge Engineering, 7*(2), 14-20.
50. Chen, L., ir Moschitti, A. (2018). Learning to Progressively Recognize New Named Entities with Sequence to Sequence Models.

51. Pires, R., Suoza, F., Rosa, G., Lotufo, R., ir Nogueira, R. (2022). Sequence-to-Sequence Models of Extracting Information from Registration and Legal Documents.
52. Chung, Y., Wu, C., Shen, C., Lee, H., ir Lee, L. (2016). Audio Word2Vec: Unsupervised Learning of Audio Segment Representations using Sequence-to-sequence Autoencoder.
53. Mimura, M., Ueno, S., Inaguma, H., Sakai, S., ir Kawahara, T. (2018). Leveraging Sequence-to-Sequence Speech Synthesis for Enhancing Acoustic-to-word Speech Recognition.
54. Chiu, C., Sainath, T., Wu, Y., Prabhavalkar, R., Nguyen, P., Chen, Z., . . . Weiss, R. (2018). State-of-the-Art Speech Recognition with Sequence-to-Sequence Models.
55. Mimura, M., Ueno, S., Inaguma, H., Sakai, S., ir Kawahara, T. (2018). Leveraging Sequence-to-Sequence Speech Synthesis for Enhancing Acoustic-to-word Speech Recognition.
56. Jiang, Y., ir Poellabauer, C. (2021). A Sequence-to-sequence Based Error Correction Model for Medical Automatic Speech Recognition. *IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 3029-3035.
57. Zhang, J., Ling, Z., Liu, L., Jiang, Y., ir Dai, L. (2019). Sequence-to Sequence Acoustic Modeling for Voice Conversion. *IEEE/ACM Transaction on Audio, Speech, and Language Processing, 27*(3).
58. Hawthorne, C., Simon, I., Swavely, R., Manilow, E., ir Engel, J. (2021). Sequence-To-Sequene Piano Transcription with Transformers.
59. Torras, P., Baro, A., Kanga, L., ir Fornes, A. (2021). On the integration of language models into Sequence to Sequence architectures for handwritten music recognition.
60. Sueiras, J., Ruiz, V., Sanchez, A., ir Velez, J. (2018). Offline continuous handwriting recognition using Sequence to Sequence neural networks. *Neurocomputing*(289), 119-128.
61. Venugopalan, S., Rohrbach, M., Donahue, J., Mooney, R., Darrell, T., ir Saenko, K. (2015). Sequence to Sequence - Video to Text.
62. Xu, N., Yang, L., Fan, Y., Yang, J., Yue, D., Liang, Y., . . . Huang, T. (2018). YouTube-VOS: Sequence-to-Sequence Video Object Segmentation.
63. Liu, B., Ramsundar, B., Kawthekar, P., ir Shi, J. (2017). Retrosynthetic Reaction Prediction Using Neura; Sequence-to-Sequence Models. *ACS Cent. Sci.*(3), 1103-1113.
64. Tang, Y., Pang, Y., ir Liu, B. (2020). IDP-Seq2Seq: identification of intrinsically disordered regions based on Sequence to Sequence learning. *Bioinformatics, 36*(21), 5177-5186.
65. Kawano, K., Koide, S., ir Imamura, C. (2020). Seq2Seq Fingerprint with Byte-Pair encoding for Predicting Changes in Protein Stability upon SIngle Point Mutation. *IEEE/ACM Transaction on Computational Biology and Bioinformatics, 17*(5).
66. Berman, D., Howser, C., Mehoke, T., ir Evans, J. (2020). MutaGAN: A Seq2Seq GAN Framework to Predict Mutations of Evolving Protein Populations.
67. Mousavi, S., Afghah, F., ir Acharya, U. (2019). SLeepEEGNet: Automated sleep stage scoring with Sequence to Sequence deep learning approach. *PLoS ONE, 14*(5).
68. Marino, D., Amarasinghe, K., ir Manic, M. (2016). Building Energy Load Forecasting using Deep Neural Networks.
69. Li, D., Su, G., Miao, S., Gu, Y., Zhang, Y., ir He, S. (2022). A short-term electric load forecast method based on improved sequence-to-sequence GRU with adaptive temporal dependence. *International Journal of Electrical Power and Energy Systems*.

70. Li, X., Tang, J., ir Yin, C. (2021). Sequence to sequence learning for prediction of soil temperature and moisture.
71. Hu, Z., ir Han, C. (2022). Image and Index Fused Sequence-to-sequence Algorithm for Vision-aided Millimeter-wave Beam Tracking.
72. Park, S., Kim, B., Kang, C., Chung, C., ir Choi, J. (2018). Sequence-to-Sequence Prediction of Vehicle Trajectory via LSTM Encoder-Decoder Architeture. *Intelligent Vehicles Symposium*(3).
73. Hai, S., Lee, D., ir Zhao, D. (2019). Sequence to Sequence learning with attention mechanism for short-term passenger flow prediction in large-scale metro system. *Transportation Research Part C 107*, 287-300.
74. Rebane, J., Karlsson, S., Denic, S., ir Papapetrou, P. (2018). Seq2Seq RNNs and ARIMA models for cryptocurrency prediction: A comparative study.
75. Gao, Z. (2021). Stock Price Prediction with ARIMA and Deep Learning Models. *IEEE the 6th International Conference on Big Data Analytics*.
76. Chen, Z., Kommrusch, S., Tufano, M., ir Pouchet, L. (2019). Sequencer: Sequence-toSequence Learning for End-to-End Program Repair. *IEEE Transactions on Software Engineering*.
77. Medhat, W., Hassan, A., ir Korashy, H. (2014). Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal*(5), 1093-1113.
78. Drus, Z., ir Khalid, H. (2019). Sentiment Analysis in Social Media and Its Application: Systematic Literature Review. *Procedia Computer Science, 161*, 707-714.
79. Saura, J. (2019). Detecting Indicators for Startup Business Success: Sentiment Analysis Using Text Data Mining. *Sustainability, 11*(3), 917.
80. Jain, V. (2013). Prediction of Movie Success using Sentiment Analysis of Tweets. *The International Journal of Soft Computing and Software Engineering [JSCSE], 3*(3).
81. Li, X., Xie, H., Chen, L., Wang, J., ir Deng, X. (2014). News impact on stock price return via sentiment analysis. *Knowledge-Based Systems*, 14-23.
82. Kearney, C., ir Liu, S. (2014). Textual sentiment in finance: A survey of methods and models. *International Review of Financial Analysis*(33), 171-185.
83. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., . . . Polosukhin, I. (2017). Attention Is All You Need .
84. Delbrouck, J., Tits, N., Brousmiche, M., ir Dupont, S. (2020). A Transformer-based joint-encoding for Emotion Recognition and Sentiment Analysis.
85. Jiang, M., Wu, J., Shi, X., ir Zhang, M. (2019). Transformer Based Memory Network for Sentiment Analysis of Web Comments.
86. Miagmar, B., Li, J., ir Kimura, S. (2019). Cross-Domain Sentiment Classification With Bidirectional Contextualized Transformer Language Models.
87. Araci, D. (2019). Finbert: Financial sentiment analysis with pre-trained language models.
88. Heidari, M., ir Rafatirad, S. (2020). Semantic Convolutional Neural Network model for Safe Business Investment by Using BERT. *2020 Seventh International Conference on Social Networks Analysis*.

89. Stevenson, M., Mues, C., ir Bravo, C. (2021). The value of text for small business default prediction: A deep learning approach.
90. Kapočiūtė-Dzikienė, J., Krupavičius, A., ir Krilavičius, T. (2013). A Comparison of Approaches for Sentiment Classification on Lithuanian Internet Comments. *Proceedings of the 4th Biennial International Workshop on Balto-Slavic Natural Language Processing*, 2-11.
91. Štrimaitis, R., Stefanovič, P., Ramanauskaitė, S., ir Slotkienė, A. (2021). Financial Context News Sentiment Analysis for the Lithuanian language. *Appl. Sci*(11), 4443.
92. Daugėla, K. (2018). Customer review classification in Lithuanian language for e-commerce business.
93. Morkūnaitė, L. (2019). Sentimento pliariškumo tyrimas Lietuvos įmonių klientų atsiliepimuose veidaknygėje ir evertink.lt.
94. Alumae, T., ir Tilk, O. (2016). Automatic Speech Recognition System for Lithuanian Broadcast Audio. *Human Language Technologies – The Baltic Perspective*.
95. Kapočiūtė, D. (2020). A Domain-Specific Generative Chatbot Trained from Little Data. *Appl. Sci*(10), 2221.
96. Pipiras, L. (2019). Lietuvių kalbos atpažinimas naudojant gilųjį mokymąsi.
97. Stankevičius, L., ir Lukoševičius, M. (2019). Lithuanian news clustering using document embeddings.
98. Stankevičius, L., ir Lukoševičius, M. (2022). Towards Lithuanian grammatical error correction.
99. Utka, A., ir Amilevičius, D. (2016). Normalisation of Lithuanian Social Media Texts: Towards Morphological Analysis of User-Generated Comments.
100. PyTorch. (n.d.). *PyTorch*. Nuskaityta iš https://pytorch.org/tutorials/intermediate/seq2seq_translation_tutorial.html
101. Luong, M., Pham, H., ir Manning, C. (2015). Effective Approaches to Attention-based Neural Machine Translation .
102. Zhang, A. (n.d.). *A trail to use Transformer to build a time-series prediction model*. Nuskaityta iš medium.com: https://medium.com/analytics-vidhya/a-trail-to-use-transformer-to-build-a-time-series-prediction-model-fa32ce493dc
103. Xue, L., Barua, A., ir Constant, N. (2022). ByT5: Towards a Token-Free Future with Pre-trained Byte-to-Byte Models.
104. Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., . . . Raffel, C. (2020). mT5: A massively multilingual pre-trained text-to-text transformer .
105. Raffel, C., ir Shazeer, N. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer.
106. Hoffman, M., Blei, D., ir Bach, F. (2010). Online Learning for Latent Dirichlet Allocation.
107. Breiman, L. (2001). Random Forests.
108. Saenz-Lechon, N., Llorente, J., Osma-Ruiz, V., ir Gomez-Vilda, P. (2006).
109. Chicco, D. (2017). Ten quick tips for machine learning in computational biology.
110. Lison, P., ir Tiedeman, J. (2016). OpenSubtitles2016: Extracting Large Parallel Corpora from Movie and TV Subtitles. *In Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*.

111. ASTRA. (n.d.). *ASTRA Tekstynas*. Nuskaityta iš http://tekstynas.vdu.lt/~andrius_u/ASTRA-Tekstynai/