Kauno technologijos universitetas

Informatikos fakultetas

Lietuviškų naujienų grupavimas pasitelkiant dokumentų vektorizavimus

Baigiamasis magistro projektas

Lukas Stankevičius
Projekto autorius

Mantas Lukoševičius
Vadovas

Kaunas, 2019

Kauno technologijos universitetas

Informatikos fakultetas

Lietuviškų naujienų grupavimas pasitelkiant dokumentų vektorizavimus

Baigiamasis magistro projektas

Programų sistemų inžinerija (6211BX011)

Lukas Stankevičius
Projekto autorius

Mantas Lukoševičius
Vadovas

Dominykas Barisas
Recenzentas

Kaunas, 2019

Kauno technologijos universitetas
Informatikos fakultetas
Lukas Stankevičius

Lietuviškų naujienų grupavimas pasitelkiant dokumentų vektorizavimus

Akademinio sąžiningumo deklaracija

Patvirtinu, kad mano, Luko Stankevičiaus, baigiamasis projektas tema „Lietuviškų naujienų grupavimas pasitelkiant dokumentų vektorizavimus“ yra parašytas visiškai savarankiškai ir visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymų nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.

Aš suprantu, kad išaiškėjus nesąžiningumo faktui, man bus taikomos nuobaudos, remiantis Kauno technologijos universitete galiojančia tvarka.

(vardą ir pavardę įrašyti ranka) (parašas)

Turinys

Paveikslų sąrašas

Lentelių sąrašas

Stankevičius, Lukas. Lietuviškų naujienų grupavimas pasitelkiant dokumentų vektorizavimus. Magistro baigiamasis projektas / vadovas Mantas Lukoševičius; Kauno technologijos universitetas, Informatikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Informatikos inžinerija (E100), Programų sistemų inžinerija (E160).

Reikšminiai žodžiai: dokumentų klasterizavimas, lietuviškų naujienų straipsniai, pirminis teksto apdorojimas, dokumentų vektorizavimas, doc2vec..

Kaunas, 2019. 64 p.

Santrauka

Šiame darbe tiriamas lietuviškų naujienų straipsnių grupavimo uždavinys, taikant *doc2vec* naujienų straipsnių vektorizavimą. Darbu metu iš viso surinkti 82793 naujienų straipsniai iš trijų lietuviškų naujienų svetainių. Vykdytų grupavimo eksperimentų metu nustatyti optimalūs *doc2vec* parametrai - epochų skaičius ir dokumentų vektorių ilgis. Tiriami ir lyginami TF-IDF ir *doc2vec* modeliai, apmokyti su pilnu bei daliniu duomenų rinkiniu. Atskirai analizuojama nereikšminių žodelių, lemavimo bei maksimalaus požymių skaičiaus filtravimo įtaka *doc2vec* ir TF-IDF vektorizavimo schemoms. Lemavimui atlikti sudaryta virš 2 mln. žodžių ir 72587 lemų duomenų bazė. Darbe parodyta, kad optimizuotas *doc2vec* yra pranašesnis už įprastą TF-IDF naujienų straipsnių vektorizavimą, bei kad *doc2vec* optimalus pirminis teksto apdorojimas gerokai skiriasi nuo įprastai taikomo TF-IDF.

Stankevičius, Lukas. Clustering of Lithuanian News Articles using Document Embeddings. Master's Final Degree Project / supervisor assoc. Mantas Lukoševičius; Faculty of Informatics, Kaunas University of Technology.

Study field and area (study field group): Informatics Engineering (E100), Software Engineering (E160).

Keywords: document clustering, Lithuanian news articles, text preprocessing, document embeddings, doc2vec.

Kaunas, 2019. 64 pages.

Summary

In this work document embeddings are studied for Lithuanian news clustering. Total of 82793 news articles from three Lithuanian news websites are collected. The best values for *doc2vec* parameters, namely number of epochs and vector size, are estimated. TF-IDF weighted scheme is compared with *doc2vec* models trained on full and partial datasets. Text preprocessing techniques as number of max features, stop words removal and lemmatization are studied separately for both TF-IDF and *doc2vec* models. Database consisting of more than 2 million words forms and 72587 lemmas was composed for lemmatization. This work depicts how optimized *doc2vec* is much better than usual TF-IDF weighting scheme and elaborate different optimal text preprocessing approach for *doc2vec* text representation model.

TERMINŲ IR SANTRUMPŲ ŽODYNAS

AE, autoenkoderis	(angl. *Autoencoder*), dirbtinio neuroninio tinklo rūšis, skirta išmokti efektyvias duomenų reprezentacijas.
BOW	(angl. *Bag of Words*), liet. žodžių maišas, dokumentuose esančių žodžių perteikimo modelis
BP	(angl. *back propagation*), efektyvus neuroninių tinklų apmokymo algoritmas.
CBOW	(angl. Continuous Bag of Words) tam tikra *word2vec* architektūra.
Enkoderis	Autoenkoderio dalis, kurioje vyksta informacijos suspaudimas.
DCNN	(angl. *Dynamic Convolutional Neural Network)*, liet. dinaminis konvoliucinis neuroninis tinklas.
Dekoderis	Autoenkoderio dalis, kurioje vyksta informacijos atkūrimas.
Dirbtiniai neuroniniai tinklai	(angl. *Artificial Neural Networks, ANN*) tarpusavyje sujungtos sistemos, sudarytos pagal biologinius gyvūnų neuroninius tinklus.
doc2vec	Dviejų sluoksnių neuroniniai tinklai, kurie atlieka dokumentų vektorizavimą.
EM	(angl. *Expectation-Maximization*) tikimybinis grupavimo algoritmas.
F1	Grupavimui įvertinti naudojamas įvertis.
Grupavimas	(angl. *Clusteringi*), suskirstymas į panašaus tipo objektų visumas.
HTML	(angl. *Hyper Text Markup Language*), programavimo kalba skirta internetinių puslapių kūrimui.
Java	Programavimo kalba
Nereikšminiai žodžiai	(angl. *stop words*)
Klasteris	Panašių ar panašaus tipo objektų visuma.
LDA	(angl. *Latent Dirichlet allocation*), generatyvus statistinis modelis, kuris leidžia stebimus duomenis išreikšti per nestebimas grupes, kurios paaiškina, kodėl tam tikri duomenys panašūs.
Lema	Žodžio bendrinė forma.
Lemavimas	Žodžių pakeitimas jų bendrinėmis formomis.
LSA	(angl. Latent Semantic Analysis), analizė, kurios metu dokumentams ir jų terminams sugeneruojama šiuos siejančių temų aibė.
MCC	(angl. *Matthews correlation coefficient*) normalizuotas grupavimo įvertis
Morfema	Mažiausias kalbos vienetas, turintis tam tikrą reikšmę.
Morfologija	Gramatikos skyrius, nagrinėjantis žodžių formas.
Naujienų agregatorius	Internetinė paslauga arba paprasčiau svetainė, kuri surenka informaciją š daugybės šaltinių ir parodo ją viename lange.
POS	(angl. *part of speech*), liet. kalbos dalis.

Požymių atranka	(angl. *feature selection*) procesas, kurio metu iš visų objekto požymių išrenkami tik aktualūs
Požymių išgavimas	(angl. *feature extraction*) procesas, kurio metu iš tam tikrų objekto požymių sukonstruojami nauji požymiai
PV-DBOW	Tam tikra *doc2vec* architektūra.
PV-DM	Tam tikra *doc2vec* architektūra.
RNN	(angl. *Recurrent Neural Network*), liet. rekurentinis neuroninis tinklas.
Skip-gram	Tam tikra *word2vec* architektūra.
Šaknies išgryninimas	(angl. *stemming*), žodžio pakeitimas jo šaknimi.
Temų modeliavimas	(angl *topic modeling*) mašininio mokymosi metodas, kuris dokumentų aibėje randa abstrakčias „temas“ – dokumentams bendrus bruožus.
Terminas	Dokumentą išreiškiantys žodžiai ar požymiai.
TF-IDF	Terminų pasvėrimo funkcija, lygi terminų dažnumo ir atvirkščio dokumentų dažnumo sandaugai.
URL	(angl. *Uniform Resource Locator*), žiniatinklio svetainės adresas.
Vektorizavimas	Tam tikro požymio pavaizdavimas skaičių eilute.
VSM	(angl. *Vector Space Model*), skaitinis modelis, kuriame dokumentai ar kiti objektai perteikiami kaip vektoriai identifikatorių (žodžių) erdvėje.
word2vec	Dviejų sluoksnių neuroniniai tinklai kurie apmokyti atlieka žodžių vektorizavimą.
Žodžių maišas	Žr. BOW.

ĮŽANGA

Žinios yra neatskiriama mūsų civilizacijos dalis. Nuo seniausių laikų tiek įrankių gamybos, tiek priešo karių artinimosi žinojimas nulemdavo mūsų protėvių išgyvenimą. Būtent gebėjimas dalintis informacija – kalbėti, mus išskyrė iš visų Žemės gyvūnų rūšių. Nuo seniausių laikų kaupdama ir naudodama įvairias žinias žmonija sau sukūrė labai patogią aplinką, kokią turime šiandien. Taip kaip ir anksčiau, taip ir dabar įvairi informacija, tik jau kitokiu - laikraščių, televizijos bei interneto pavidalu liko neatsiejama mūsų gyvenimo dalis.

Nors prieš šimtmečius pagrindinis informacijos iššūkis buvo jos trūkumas, šiandien turime priešingą situaciją. Dėl didžiulius informacijos kiekius talpinančių ir perduodančių interneto technologijų iškyla informacijos atsirinkimo problema. Pavyzdžiui, didžiausias Lietuvos naujienų tinklalapis per dieną publikuoja virš 80 naujienų straipsnių. Pridėjus kartu ir kitus naujienų portalus bei svetaines iš viso pasaulio turime tokį informacijos kiekį, kurio žmogui tiesiog neįmanoma peržvelgti. Kyla klausimas: kaip rasti svarbiausią informaciją?

Vystantis duomenų mokslui buvo pradėti įvairūs tyrimai tekstinę informaciją kuo labiau supaprastinti ir geriau įsisavinti. [1] išskiriami trys automatiniai žinių išgavimo būdai: informacijos apibendrinimas, jos klasifikavimas bei grupavimas. Šiuo metu geriausi apibendrinimo algoritmai žmonių vertinimu gauna 2 – 4 taškus iš 5 [2], klasifikavimo tikslumas siekia 50 – 94 % [3], o grupavimui gaunami apie 0,4 F1 [4] balo įverčiai. Klasifikavimas priklausomai nuo duomenų rinkinio gali pasiekti ir labai didelius tikslumus, tačiau naujienų straipsnių tekstiniams duomenims būtent grupavimas yra nepamainomas dėl savo universalaus grupių išskyrimo šiame sparčiai kintančiame pasaulyje.

[5] parodžius, kad dirbtiniai neuroniniai tinklai gali būti sėkmingai apmokyti ir išgauti efektyvias duomenų reprezentacijas, buvo sukurta daug naujų sėkmingų duomenų įsisavinimo algoritmų. Šiame darbe pirmą kartą tiriamas vieno tokių algoritmų, *doc2vec*, veikimas lietuviškiems naujienų straipsniams.

Tikslas ir uždaviniai

Šio darbo tikslas – ištirti *doc2vec* dokumentų vektorizavimą lietuviškų naujienų grupavimui. Tikslas bus pasiekiamas įgyvendinant šiuos uždavinius:

1. rasti geriausius *doc2vec* dokumentų vektorizavimo parametrus;
2. ištirti, kokia įtaką *doc2vec* dokumentų vektorizavimui turi pirminis teksto apdorojimas;
3. palyginti *doc2vec* su klasikiniu TF-IDF dokumentų vektorizavimu.

Šio darbo naujumas yra tas, kad literatūroje dar nėra publikuota, kaip *doc2vec* dokumentų vektorizavimas veikia lietuviškų naujienų grupavimą. Šis tyrimas leis palyginti plačiai naudojamą TF-IDF bei *doc2vec* dokumentų vektorizavimo schemas.

Dokumento struktūra

Dokumentas pradedamas naujienų grupavimo srities analize, kurioje apžvelgiamos literatūroje dažnai aptinkamos šiam darbui aktualios praktikos. 2-ame skyriuje aprašomas lietuviškų naujienų grupavimo sistemos projektavimas nuo reikalavimų išskyrimo iki statinio ir dinaminio sistemos vaizdų apibūdinimo. Paskutiniajame skyriuje aprašomi šio darbo uždavinių įgyvendinimo rezultatai bei galiausiai pateikiamos darbo išvados.

1. NAUJIENŲ GRUPAVIMO SRITIES ANALIZĖ

Šiame skyriuje pateikiama teksto grupavimo proceso etapų [1], aktualios literatūros ir naudotinų programinės įrangos technologijų apžvalga. Taip pat apžvelgiamos rinkoje esančios naujienų grupavimu ar klasifikavimu užsiimančios svetainės.

1.1. Pirminis teksto apdorojimas

Pirminis teksto apdorojimas prasideda su pirmuoju prisilietimu prie duomenų. Patys paprasčiausi tekstinių duomenų vienetai – žodžiai, raidės, skaičiai, skyrybos ženklai. Dažniausiai kaip pagrindiniai teksto elementai naudojami žodžiai.

Jei tai yra naujienų straipsniai esantys interneto svetainėse, galima nuspręsti išgauti ne tik tekstinę, bet ir struktūrinę ar kitą informaciją. Kuo daugiau aktualios informacijos sugebama ištraukti, tuo žinių suradimas iš šių duomenų vėliau bus efektyvesnis. Naujienų straipsniai pateikiami HTML formato rėmuose ir čia reikia pašalinti tokį triukšmą kaip reklamos, navigacija puslapyje ir kt. Iš kitos pusės, internetiniai straipsniai gali turėti papildomą raktinių žodžių informaciją, kuri, kaip pažymima [6], yra labai patikima. Šaltinis [7] nurodo, kad svarbi gali būti papildoma struktūrinė informacija, jei nagrinėjamas terminas (žodis) yra:

- pavadinime;
- pirmame sakinyje;
- sakiniuose 2-6;
- sakiniuose pradedant septintuoju.

Kitas šaltinis [6], nagrinėjantis raktažodžių dokumente radimą, pasiūlė gana panašius papildomus požymius, pagal kuriuos galėtų būti įvertintas termino (raktažodžio) reikšmingumas:

- terminas yra pavadinime;
- terminas yra paveikslėlio aprašyme;
- termino šriftas yra kitoks, nei pagrindinio teksto;
- terminas dengia didžiausią teksto dalį;
- termino pasikartojimų skaičius pirmame paragrafe;
- termino pasikartojimų skaičius paskutiniame paragrafe;
- termino kalbos dalis POS (angl. *Part of Speech*).

[6] buvo gauta, kad pagal aukščiau išvardintus papildomus požymius apmokius BP (angl. *back propagating*) neuroninį tinklą ir naudojant LM (Levenberg-Marquardt) optimizavimo algoritmą, buvo pagerintas raktažodžių radimas, palyginus su įprastais metodais. Pridedant papildomą termino kalbos dalies informaciją nustatoma, kokia kalbos dalis yra žodis ir pagal tai įvertinamas jo svoris. Pavyzdžiui, šaltinyje [8], nagrinėjusiame dvikalbių straipsnių klasterizavimą pastebėta, kad matuojant panašumą vien tik tarp daiktavardžių, gaunami geriausi grupavimo rezultatai.

Grupavimui pagerinti taip pat naudojamas terminų reikšmės išsamesnis apibūdinimas. Tam, pavyzdžiui, šaltinis [9] panaudojo visas angliškosios „Wikipedia“ bei „Open Directory Project“ duomenų aibes, kad gautų terminų semantinius ryšius. Tokiu būdu vienam teksto fragmentui galima gauti geriausiai atitinkančius kitus teksto fragmentus, kiekvieną su tam tikru svoriu, kurie gali padėti pagerinti grupavimą. Iš kitos pusės šaltinyje [8] pastebima, kad kai buvo bandyta žodžius straipsniuose pakeisti jų visais sinonimais, grupavimo kokybė sumažėjo ir iškelta prielaida, kad turi būti naudojami tik statistiškai geriausiai atitinkantys sinonimai.

Dažniausiai teksto reprezentavimui naudojamas BOW (angl. *Bag of Words*) modelis ir jame nėra išsaugoma žodžių sekos informacija. Taip pat kartais tam tikri požymiai glūdi keliuose paskui vienas kitą einančiuose terminuose, pavyzdžiui žodžių seka „Kauno technologijos universitetas“ apibūdina konkretų unikalų universitetą. BOW modelyje iš šios sekos liktų tik bendrinė informacija, kad yra požymiai „universitetas“, „Kauno“, bei „technologijos“, kurios neidamos kartu turi visai kitas reikšmes. Šiems atvejams išspręsti išsaugoma pasikartojanti sekos informacija, angl. vadinama *n-gram*. Jei pasirinktume, kad tų *n-gram* bus 3, tai pvz. skaičių seka nuo 1 iki 10 būtų konvertuojama į tokius elementus turintį BOW: „1 2 3, 2 3 4, 3 4 5, 4 5 6, 5 6 7, 6 7 8, 7 8 9, 8 9 10. Galima taip pat vienu metu sudaryti *n-gram* nuo 1 iki 3 elementų, tuomet BOW būtų tokie elementai: „1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 1 2, 2 3, 3 4, 4 5, 5 6, 6 7, 7 8, 8 9, 9 10, 1 2 3, 2 3 4, 3 4 5, 4 5 6, 5 6 7, 6 7 8, 7 8 9, 8 9 10“. Taigi, kaip matyti, tokiu būdu gerokai išauga turimas žodynas, tačiau užtikrinama, kad tam tikros svarbios požymių sekos būtų išsaugotos.

Pirminis teksto apdorojimas lemia, kas bus elementarūs teksto dokumentų požymiai. Esant galimybėms, turėtų būti paimama kuo daugiau informacijos. Tačiau dažniausiai struktūrinė ar kita papildoma dokumento vizualizacija yra sunkiau išgaunama ir nenaudojama, paliekant esminę žodžių sekos informaciją.

1.1.1. Požymių filtravimas

Papildomi teksto duomenys yra naudingi tik tuomet, jei jie praplečia žinias, esančias jau turimame tekste. Kitu atveju teksto grupavimo procesas gali būti tik pablogintas. Šaltinyje [10], analizavusiame lietuviškų naujienų agregavimą gauta, kad net „30,78 procento originalaus teksto sudaro nereikšmingi žodžiai“. Tokiu atveju reikia stengtis sumažinti netinkamus duomenis. Dalį duomenų galima išmesti, nes:

- kai kuriuos duomenis (požymius) pralenkia triukšmo intensyvumas;
- dalis požymių koreliuoja vieni su kitais (pavyzdžiui tiesinė koreliacija).

Duomenų filtravimas dar vadinamas dimensiškumo mažinimu, nes taip mažėja kiekvienam dokumentui tenkančio BOW modelio vektorio ilgis. Šaltiniuose [11,12] išskiriamos dviejų tipų populiariausios dimensiškumo sumažinimo technikos (angl. *Dimension Reduction Techniques, DRT*):

- požymių atranka (angl. *feature selection*);

- požymių transformacija/ išgavimas (angl. *feature transformation/ extraction*);

Požymių atranka nuo požymių transformacijos skiriasi tuo, kad paskutiniuoju atveju nebegalima tiksliai atsekti iš kokių požymių išgauti naujieji požymiai. Transformacijos metu stengiamasi rasti kitą, kompaktiškesnį požiūrį į tuos pačius duomenis. Tokia nauja projekcija nebeturi senosios prasmės (pirminės projekcijos kiekviena dimensija BOW modelyje reiškė atskirą žodį), nes naudojant daugelį požymių vienu metu apimančiomis transformacijomis, atsirado nauji išvestiniai požymiai. Tuo tarpu atrankos atveju dalis požymių išmetama, o kiti paliekami – taip išsaugoma pradinė požymių reikšmė. Paskutinioji, dimensiškumo sumažinimo technika, yra populiariausia, nes tokiu būdu aiškiai matoma, kurie požymiai (pvz. žodžiai) yra išmetami, o kurie – paliekami.

Dažniausiai yra naudojami šie dimensiškumo mažinimo būdai:

- dažniausių ir rečiausių žodžių išmetimas;
- nereikšminių žodių išmetimas (angl. *stop words removal*);
- šaknies išgryninimas (angl. *stemming*);
- lemavimas (žodžio pavertimas bendrine forma);
- skaičių normalizavimas.

Kai kurie žodžiai dokumentuose yra tokie reti, kad dėl jų retumo jie neteikia jokios informacijos vėliau sekančiam žinių išgavimui. Lygiai taip pat ir labai dažni žodžiai, pasitaikantys visur, nesuteikia informacijos, galinčios išskirti kai kurios dokumentus iš kitų. Kaip dažnumo pamatavimas naudojama procentinė išraiška, kiek dokumentų turi šį žodį. Pagal pasirinktą ribinę vertę tiek labai dažni, tiek labai reti žodžiai turėtų būti išmesti [13].

Nereikšminių žodžių pašalinimas pagerina grupavimo kokybę, nes sumažinamas matmenų skaičius ir nebus grupuojama pagal šiuos beveik nieko nepasakančius žodelius. Vis dėlto, kaip teigiama šaltinyje [14], išmesti anglų kalbos žodeliai „it“ and „us“ turės neigiamą efektą, jei tekste būtų tokie žodžiai kaip „IT specialist“ ar „US government“.

Šaknies išgryninimas sprendžia problemą, kai tokios pačios reikšmės žodis yra parašytas kita morfologine forma. Tai yra tą pačią reikšmę turintys žodžiai, tačiau pagrindiniai naudojami grupavimo algoritmai skirtingos morfologinės formos žodžius laikys skirtingais. Šiam žingsniui yra parašyta įvairių algoritmų, dažniausiai angliškiesiems žodžiams šaltiniai, kaip šis [15] naudoja Porter'io algoritmą arba jo modifikacijas [14]. Kitaip, nei anglų, lietuvių kalboje šaknies išgryninimas turi būti atliekamas pašalinant simbolius ne tik iš dešinės, bet kartais ir iš kairės pusės. Pavyzdžiui, žodyje „bepasikiškiakopūsteliaudamas“ turėtų būti pašalinta „bepasi“ bei „eliaudamas“. Iš kitos pusės šaltinyje [14] išskiriamos šios šaknies išgryninimo problemos:

- nesprendžiamas sinonimų sugrupavimas;
- gali atsirasti homonimai (žodis, turintis kitą reikšmę);
- ne visi tos pačios reikšmės žodžiai turi vienodos šaknies morfologines formas.

Vis dėlto, nepaisant išvardintų trūkumų, grupuojant lietuviškas naujienas šaknies išgryninimas pagerina rezultatus [16].

Šiame poskyryje išvardinti įvairūs požymių filtravimo būdai skirtingais aspektais sprendžia tas pačias triukšmo ir tarpusavio koreliacijos problemas. Vis dėlto, dėl savo savybių ir poveikio įvairumo, kiekvieno filtro tinkamumas nagrinėjamam duomenų rinkiniui turėtų būti nustatomas eksperimentiškai.

1.2. Dokumentų reprezentacija

Tam, kad kompiuteris galėtų atlikti skaičiavimus su dokumentais ir jų savybėmis – šie turi būti vektorizuoti. Kiekvienam dokumentui tuomet tenka aprašantysis vektorius. Jei jo ilgį pavadintume dimensijomis, tai kiekvieną dokumentą būtų galima įsivaizduoti kaip tašką minėto dimensiškumo erdvėje. Tokia dokumentų reprezentacija dar vadinama VSM (angl. *Vector Space Model*) modeliu.

Paprasčiausias VSM variantas yra žodžių maišo“ BOW (angl. Bag of Words) modelis. Jame yra dvi eilutės – pirmoje nesikartojantys terminai, o antroje – termino pasirodymo dokumente kartų skaičius. Atvaizduojant daugiau dokumentų, atsiranda daugiau eilučių, o stulpelių terminų sąrašas ilgėja. Tokiu būdu kiekvieno dokumento ir žodžio susikirtime įrašoma, kiek kartų tas žodis yra pavartotas tame dokumente. BOW modelis išsaugo terminų ir jų pasikartojimo skaičiaus dokumentuose informaciją, bet neišsaugo jų vietos ir konteksto informacijos. Iš BOW nebegalima atkurti, koks žodis po kokio žodžio ėjo (dėl to ir vadinama žodžių maišu, nes neaišku kuris žodis buvo įmestas pirmas į maišą). Nors tai atrodo kaip akivaizdus trūkumas, šaltinis [17] teigia, kad sekos informacijos ir nereikia, nes yra labai mažai būdų bet kokiems žodžiams išdėstyti ir dėl to nestruktūruoti modeliai praktikoje veikia geriau už tuos, kurie naudoja struktūrinę informaciją.

BOW modelis, kuriame yra paprasti žodžių pasikartojimų skaičiai, paprastai nenaudojamas, nes jame esanti informacija yra nepasverta. Jei vienas žodis dokumente yra pakartotas 100 kartų, o kitas – 20, tai dar nereiškia, kad pirmasis yra 5 kartus svarbesnis, jis galbūt tik 5 kartus dažniau naudojamas. Dėl to kiekvienam žodžiui paprastai priskiriamas išvestinis svoris, kad būtų gaunami tikslesni grupavimo algoritmo rezultatai. Pagal [18,19] egzistuoja šie populiariausi terminų pasvėrimo algoritmai:

- *idf* (angl. *Inverse document frequency*);
- *chi2*: χ^2- statistika grįstas terminų pasvėrimas;
- *ig* (angl. *information-gain*);
- *rf* (angl. *relevance frequency*);
- *or* (angl. *odds-ratio*).

Minėtame šaltinyje [18] buvo pasiūlyti nauji terminų pasvėrimo algoritmai, kurie buvo išmokti įvairiomis dalimis panaudojant aukščiau išvardintus populiariuosius ir juos aplenkę. Pats

populiariausias, daugumoje šaltinių [13,16] rezultatyviai naudojamas yra TF-IDF metodas. Jo metu termino *t* svoris *W* dokumente *d* randamas:

$$W_{td} = tf_d \cdot idf_{td} = tf_{td} \cdot \log \frac{n}{df_{td}} \quad (1.1)$$

čia tf_{td} – termino *t* pasikartojimų dokumente *d* skaičius, df_{td} – dokumentų skaičius, kuriuose yra terminas *t* ir *n* – visų dokumentų skaičius. Nors šis terminų svorio skaičiavimas yra kone klasika, kaip teigiama šaltinyje [20], mažų tekstų grupavimas turi duomenų trūkumo problemą, todėl TF-IDF čia negali gerai veikti.

Galima teigti, kad geriausia klasikinė dokumentų reprezentacija gaunama su TF-IDF terminų pasvėrimu. Šiame projekte paskutinysis modelis bus naudojamas kaip lyginamasis, tiriant sudėtingesnes, neuroninio tinklo apmokymu gaunamas reprezentacijas.

1.2.1. Neuroninių tinklų naudojimas

Pastarąjį dešimtmetį pastebimas dirbtinių neuroninių tinklų kompiuteriniuose skaičiavimuose populiarėjimas. Jie „suklestėjo“, nes sparti kompiuterių pažanga leido pasiekti reikiamus skaičiavimų pajėgumus, o interneto technologijos dėka tapo prieinami dideli duomenų kiekiai, reikalingi šiems algoritmams. Prestižiniame žurnale „*Nature*“ 2015 m. išleistame straipsnyje, analizuojančiame gilųjį mokymąsi [21] prognozuojama, kad jis ateityje bus vis sėkmingesnis, nes jam tobulėti reikia labai nedaug „rankinės inžinerijos“, o tik tinkamai parinktos architektūros, skaičiavimo pajėgumų ir didžiulių kiekių duomenų. Tai rodo, kad dirbtinių neuroninių tinklų pagrindu kuriami skaičiavimo algoritmai yra vieni perspektyviausių.

Vienas efektyviausių gan neseniai pasiūlytų neuroninių tinklų modelių teksto reprezentacijų gavimui – *doc2vec* [22]. Pasiūlytas 2014 metais jis aplenkė visus kitus konkuravusius dokumentų reprezentacijos modelius [22]. *Doc2vec* buvo adaptuotas iš žodžių reprezentacijų tyrimuose išrasto *word2vec* modelio [23], kur žodžių vektoriuose išsaugoma įvairi, iki tol neišgauta informacija. Tokiame modelyje, pavyzdžiui, gaunama, kad žodžiai „Berlynas“ ir „Vokietija“ turi tokį patį ryšį kaip žodžiai „Paryžius“ ir Prancūzija. Pritaikius tą patį mokymosi algoritmą dokumentams, gautas ir jau minėtas *doc2vec*, pasižymintis analogiškomis savybėmis, tik dokumentų vektoriams. Tokiu būdu kiekvienas dokumentas gali būti suvedamas į vos 100 – 200 skaitmenų ilgio reprezentatyvų vektorių, savo saugoma informacija gerokai lenkiantis paprastą BOW vektorių [22].

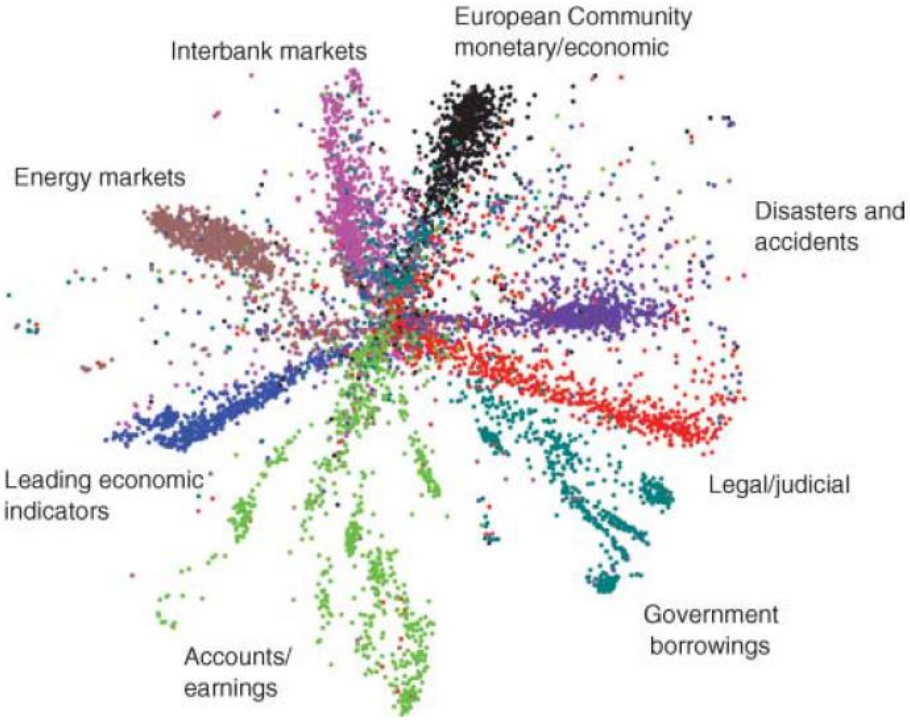
Doc2Vec modelis gaunamas apmokant daugelį dokumentų tam tikra dirbtinio neuroninio tinklo architektūra [22]. Kiekvieno stochastinio gradiento leidimosi iteracijoje iš tam tikro ilgio teksto lango atsitiktinai pasirenkamas žodis. Tuomet suformuojamas klasifikavimo uždavinys, kad neuroninis tinklas išskirtų būtent pasirinktą žodį, žinodamas tik duoto dokumento vektorių. Tokiu būdu po daugelio iteracijų dokumento vektoriaus reikšmės pasikeičia taip, kad geriau atliktų žodžių

klasifikaciją ir tampa reprezentuojančios minėtą dokumentą. Taigi, kaip matyti, ši neuroninio tinklo architektūra yra ne tik efektyvi, bet ir labai paprasta.

Analizuojant literatūrą nepavyko rasti tokių tyrimų, kur *do2vec* modelis būtų pritaikytas lietuvių kalbai. Buvo rasti tik keli darbai, susiję su analogiškais žodžių vektoriais *word2vec*. Šaltinyje [24] buvo išbandyti įvairūs žodžių vektorių modeliai bei mokymosi algoritmai, naudojant labai didelę, apie 234 milijonus žodžių siekiantį duomenų rinkinį. Buvo nustatyta, kad CBOW (angl. *Continuous Bag of Words*) architektūra pralenkė *skip-gram* architektūrą, o vektorių dydis reikšmingos įtakos rezultatams neturėjo. Kitas darbas [25] palygino tradicinius ir giliojo mokymosi metodus, naudojančius žodžių vektorius, tekstų nuotaikos analizei. Čia buvo gauta, kad giliojo mokymosi metodai buvo geresni tik tuomet, kai tekstai buvo labai maži, o kitu atveju geriausiai veikė tradiciniai metodai. Taigi apžvelgus tyrimus, susijusius su žodžių vektoriais, galima pastebėti, kad čia geriausiai veikianti yra CBOW architektūra, o žodžių vektoriai ne visada pagerina atliekamą žinių išgavimo užduotį.

Doc2vec modelio neuroninis tinklas nėra gilus, tačiau esama teksto reprezentaciją išgaunančių modelių, naudojančių ir giliuosius neuroninius tinklus. Vienas pirmųjų tokių tinklų buvo autoenkoderis AE (angl. autoencoder) [5]. Jis sukonstruotas taip, kad išmoktų informaciją suspausti taip, kad iš tankios reprezentacijos vėliau galėtų atkurti visą buvusią informaciją. Informacijos suspaudime ir dalyvauja daug dirbtinio neuroninio tinklo sluoksnių, kurie spaudimą atlieka palaipsniui. Aprašytas metodas leido aplenkti LSA (angl. *Latent Semantic Analysis*) metodą ir aiškiai išskyrė 8 rastas kategorijas net su 2000-500-250-125-2 automatiniu kodavimu, t. y. suprojektavus vos į dvimatę erdvę. Kaip matyti 1.1. pav., tokiu būdu gauta reprezentacija net vizualiai parodo į skirtingas grupes išsidėsčiusius straipsnius.

Nors gilusis mokymasis paprastai taikomas grupuojamo teksto reprezentacijai išgauti, vis dažniau pasirodo tyrimai, kuriuose reprezentacija ir grupavimas yra apjungiami [26]. Tokiu būdu galima ne tik efektyvinti visą procesą, bet ir grupavimui panaudoti giliųjų neuroninių tinklų savybes. 2018 metų šaltinyje [27], nagrinėjančiame giliuosius neuroninius tinklus, siūloma grupavimui naudoti ne suspaustą informaciją, esančią pačiame mažiausiame autoenkoderio sluoksnyje, o daugelio spaudimo sluoksnių svorius. Nurodoma, kad čia naudojamas KCC (angl. *K-means-based Consensus Clustering*) [28] grupavimo algoritmas kartu su daugelio sluoksnių informacija aplenkė kitas tirtas grupavimo technikas. Šaltinis [29] pastebi, kad nors kartu optimizuojant neuroninius tinklus ir grupavimą gaunami geresni rezultatai, yra sunku paaiškinti pagerėjimo priežastį. Taip pat minėtas šaltinis išskiria ir daugiau neuroninių tinklų architektūrų kategorijų, naudojamų grupavimui: VAE (angl. *variational autoencoder*), GAE (*generative autoencoder*) ir CDNN (angl. *clustering deep neural network*). Vis dėlto [29] apgailestaujama, , kad dauguma literatūroje pasiūlytų metodų buvo kuriami vaizdų rinkinių grupavimui ir tik labai mažai modelių sukurti tekstinių dokumentų grupavimui.



1.1. pav. Tekstinių dokumentų projekcija, gauta autoenkoderio dvimačiu sluoksniu [5]

Kitaip nei gilieji dirbtiniai neuroniniai tinklai, rekurentinis neuroninis tinklas labiau atsižvelgia į laiko eilučių duomenis. Tokie, pavyzdžiui, yra natūralios kalbos sakiniai. Jei norime nuspėti žodį, koks bus po dabar pasakyto, reikia atsiminti, kokie žodžiai buvo ir prieš dabar pasakytą. RNN metodo architektūra tai ir įgyvendina. Kol kas geriausiai nuosekliems teksto duomenims ir yra taikomi rekurentiniai neuroniniai tinklai, su įgyvendintais dėmesio (angl *attention*) sluoksniais ir struktūrinės informacijos išsaugojimu [30].

Nors yra įprasta, kad kita neuroninių tinklų architektūros rūšis - konvoliuciniai neuroniniai tinklai naudojami vaizdų klasifikavimui, tačiau juos taip pat galima pritaikyti ir teksto reprezentacijai. Šaltinyje [31] dinaminis konvoliucinis neuroninis tinklas DCNN sėkmingai panaudojamas klausimams klasifikuoti, ir jo tikslumas prilygsta sudėtingoms technikoms, naudojančiomis bent 10 įvairių „rankinių“ teksto požymių. Šaltinyje [32] konvoliucijos ir sujungimo operacijomis gaunami sakinius reprezentuojantys požymiai, o po to, pakartojant konvoliucijos ir sujungimo operacijos nebe sakiniams, o iš jų gautiems požymiams, gaunami visą dokumentą charakterizuojantys požymiai. Naudojantis šiais metodais galima išsaugoti ir panaudoti informaciją apie sakinių bei žodžių vietą, ko negalima padaryti naudojant tradicinę žodžių maišo reprezentaciją. Vis dėlto analizuojant literatūrą pastebėta, kad konvoliucinių neuroninių tinklų tyrimai tekstiniams duomenims yra nepopuliarūs.

Galima apibendrinti, kad neuroniniai tinklai teksto grupavime padeda išgauti vienus geriausių rezultatų. *Doc2Vec* modelis, nors dar netirtas lietuviškiems tekstams, pasaulyje yra jau plačiai naudojamas. Tuo tarpu kitų giliaisiais neuroniniais tinklais paremtų modelių tyrimai yra dar tik tobulinami, bet tikimasi, kad ateityje turėtų pasireikšti dar rezultatyvesniais pasiekimais.

1.3. Grupavimo algoritmas

Literatūroje nagrinėjant algoritmus dažnai minimi klasifikavimo ir grupavimo (angl. *clustering*) terminai. Klasifikavimas atliekamas remiantis patirtimi ir žiniomis, t. y. naudojant jau turimas tam tikras kategorijas. Daugumą klasifikavimą atliekančių algoritmų išmoksta pagal tam tikrus požymius priskirti grupuojamą tekstą atitinkamai kategorijai. Tuo tarpu grupavime, grupės sudaromos tik pagal tai, kokius konkrečiu metu požymius turi teksto dokumentai, t. y. grupes galima apibūdinti tik pagal tai, kas į jas pateko. Pavyzdžiui, klasifikuojant naujienas, jei staiga į Žemę nusileistų ateiviai ir apie tai būtų rašomi naujienų straipsniai, tai klasifikavimas šias naujienas priskirtų jau esamoms kategorijoms kaip „Pasaulio naujienos“, arba „Politika“, o grupavimo algoritmai išskirtų atskirą grupę, kurią būtų galima pavadinti „Ateiviai“. Dėl minėtų priežasčių klasifikavimo uždavinys yra lengvesnis, o grupavimo universalesnis.

Analizuojant literatūrą buvo rasta labai daug įvairių grupavimo algoritmų ir jų modifikacijų, individualių siūlymų. Jų visų išnagrinėjimas ar palyginimas savo sudėtingumu galėtų pralenkti net ir doktorantūros darbą, todėl čia jie bus pristatyti labai glaustai. Analizuotoje literatūroje aptikti grupavimo algoritmai pateikti 1.1 lentelėje. Bene geriausiai algoritmai apibendrinami knygoje [12], kur išskiriami net 4 skyriai atskiroms grupavimo metodų grupėms:

- tikimybiniai grupavimo modeliai:
    - mišinio modeliai (angl. *mixture models*);
    - EM algoritmas (angl. *Expectation-Maximization*);
    - tikimybiniai temų modeliai (angl. *probabalistic topic models*);
- paremti panašumu:
    - dalijantys (angl. *partitional*);
        - kieti (angl. *hard*), pvz. K-means;
        - minkšti (angl. *soft*), pvz. fuzzy, C-means [33];
    - hierarchiniai:
        - aglomeraciniai (angl. *agglomerative clustering* arba *bottom-up*);
            - vienos jungties (angl. *single link*);
            - pilnos jungties (angl. *complete link*);
            - grupių vidurkių (angl. *group average*);
            - centroido panašumo (angl. *centroid similarity*);
            - Ward‘o kriterijaus (angl. *Ward‘s criterion*).
        - nesutapimo (angl. *divisive clustering* arba *top-down*);
- tankiu grįsti algoritmai (angl. *density-based clustering*);
- gardele grįsti algoritmai (angl. *grid-based clustering*);

1.1 lentelė Literatūroje aptiktų grupavimo algoritmų pavyzdžiai

Metodų grupė	Literatūroje aptikti algoritmų pavyzdžiai
Tikimybiniai	LDA [33,34]; PLSA (angl, Probabilistic Latent Semantic Analysis), EM [12].
Dalijantys	K-means, K-medians, K-modes, Fuzzy K-means, X-means, Intelligent K-means, Bisecting K-means, Kernel K-means, Mean Shift, Weighted K-means, Genetic K-means, K-means filtering algorithm, Rough C-means, Possibilistic C-means [12]; K-harmonic Means, HFKHM (angl. *Hybrid Fuzzy K-harmonic means*), fuzzy C-means [35]; Sk-means [36]; FTDL (angl. *Fuzzy K-means Based on Text Dual Language Model*) [37]; K-means++, PAM, CLARA, CLARANS [33]; k-prototype [38]; K-medoids [39]; Affinity Propagation [36]; PKBN, MCSKM [40], KCC [28]
Hierarchiniai	Zahn Minimum Spanning Tree-Based Divisive Clustering, CURE, CHAMELEON, COBWEB, SOM (angl. *Self Organising Maps*), GAAC, [12]; ROCK (angl. *robust hierarchical clustering algorithm*) [38]; Group-Average Hierarchical Clustering (GAHC), Suffix Tree GAHC (ST-GAHC) [36]; Suffix Tree KNearest Neighbors (STKNN) [41]; Enhanced ROCK, BIRCH, modified BIRCH [39]; DHCA (angl. *Dynamic Hierarchical Compact Algorithm*), HCA (angl. *Hierarchical Compact Algorithm*), The Dynamic Hierarchical Star Algorithm, ICA (angl. *Incremental Clustering Algorithm*) [14].
Tankiu grįsti	DBSCAN, DENCLUE, OPTICS [12]; HDBSCAN, F-OPTICS, OPTICS-ξ, HDBSCAN-EOM, DBSCAN-Martingale [34].
Gardele grįsti	[42] analizuojamas STING, kitur [38] nagrinėtas CLIQUE algoritmas. Visi kiti variantai apibendrinti [12].
Kiti	Spektrinio grupavimo [8,20,34,35,43]; giliojo mokymosi [44].

Hierarchiniai ir tankiu grįsti algoritmai gali būti patogūs tuo, kad juose nereikia nurodyti būsimų grupių skaičiaus. Tokiu atveju išvengiama išankstinio spėjimo, kiek aiškių grupių turėtų būti duomenų rinkinyje. Iš kitos pusės, vis tiek reikia priimti tam tikrus sprendimus, kurie nulemia galutinį grupių skaičių. Hierarchiniame algoritme reikia nuspręsti, ties kuriuo hierarchijos išsišakojimu bus „nukerpama“ ir sudaromos grupės. Tuo tarpu tankiu grįstuose algoritmuose reikia eksperimentiškai nusistatyti *Eps* bei *MinPts* parametrus, pagal kuriuos bus grupuojamos tik nurodyto tankio sritys. Papildomi minėtų algoritmų privalumai yra tie, kad hierarchinio algoritmo grupavimu gaunama pilna hierarchija, kuri vėliau gali būti panaudojama, o tankiu grįstu algoritmu galima sugrupuoti nesferinio pasiskirstymo duomenis.

Pagal [12], populiariausi grupavimo algoritmai yra hierarchiniai ir dalijantys, taip pat plačiai paplitęs LDA [37,45]. [36] teigiama, kad hierarchiniai grupavimo algoritmai yra tikslesni, bet dalijantys greitesni. Vis dėl to, kokį algoritmą geriausia taikyti, priklauso ir nuo teksto specifikos. Pavyzdžiui, mažiems tekstams grupuoti naudojami metodai, grįsti papildomomis žiniomis [46], nes mažuose tekstuose neužtenka juose esančios informacijos. Tyrimai, nagrinėjantys lietuviškų dokumentų grupavimą, pataria naudoti K-means [16], sferinį K-means [13] arba EM [47] algoritmus. Pastebima, kad K-means pasižymi didesniu greičiu, geriau veikia didesniems duomenų rinkiniams [47] bei grupavimo rezultatų tikslumu lenkia kitus grupavimo algoritmus [16]. Taigi galima pagrįstai teigti, kad šiuo metu pats populiariausias ir dažniausiai naudojamas yra dalijantis K-means algoritmas.

K-means algoritmo veikimas yra gana paprastas. Pirmasis K-means algoritmo žingsnis – atsitiktinai parenkami taškai, kurie bus pradinės iteracijos grupių centrai (iš viso yra nustatytas *K* grupių skaičius). Tuomet kiekvienos iteracijos metu kartojami šie 2 veiksmai:

1. kiekvienas grupuojamas elementas priskiriamas artimiausiam iš *K* centroidų;
2. perskaičiuojami sudarytų grupių centroidai.

Šie veiksmai kartojami kiekvienoje iteracijoje tol, kol patenkinamas konvergencijos kriterijus. Dėl to, kad yra tam tikra galutinio rezultato priklausomybė nuo pradinių atsitiktinai parinktų grupių centrų, algoritmas kartojamas kelis kartus, kad centroidai būtų suvidurkinti. Šio algoritmo paprastumas ir lemia tai, kad jis yra greitesnis už daugumą kitų grupavimo algoritmų.

Dėl grupavimo algoritmų gausos, tinkamas turi būti pasirinktas toks, kuris labiausiai tenkintų projekto reikalavimus. Šiame darbe pasirenkamas naudoti K-means grupavimo algoritmas dėl šių priežasčių:

1. svarbus veikimo greitis;
2. algoritmas panašiuose tyrimuose apibūdinamas kaip geriausias.

Kiti faktoriai, kuriais pasižymi kiti algoritmai, pvz. hierarchiniame gaunama visa elementų hierarchija, arba tai, kad nereikia nurodyti grupių skaičiaus, šiam projektui nėra svarbūs.

1.4. Grupavimo įvertinimas

Daugybė egzistuojančių grupavimo algoritmų turi būti palyginti. Pirmasis iš galimų palyginimo parametrų yra laikas, per kurį yra sugrupuojama. Antrasis parametras – sugrupavimo kokybė. Jei abu šie parametrai bus didesni už tam tikrą pasirinktą ribą, bus galima sakyti, kad grupavimo algoritmas dirba tinkamai. Grupavimo kokybę yra sunkiau suskaičiuoti nei laiką, nes pagal [48] galimi 4 grupavimo rezultatai grupavimo elementams, parodyti 1.2 lentelėje.

1.2 lentelė Galimi sumaišymo matricos (angl. *Confusion Matrix*) variantai

	Tinkama	**Netinkama**
Atrinkta	*tp* – tinkamai atrinkti elementai (*angl. true positives*)	*fp* – netinkamai atrinkti elementai (*angl. false positives*)
Neatrinkta	*fn* – neatrinkti tinkami elementai (*angl. false negatives*)	*tn* – neatrinkti netinkami elementai (angl. *true negatives*)

Atliekant grupavimo įvertinimą, reikia apsibrėžti, kas bus pagal sąlygą vertinami tinkami ar netinkami elementai. Jei būtų sprendžiamas paprastas klasifikavimo uždavinys, tuomet užtektų vienintelio mato – procento, kiek tekstinių dokumentų pateko ten, kur turėjo patekti. Deja, nagrinėjamame grupavimo (klasterizavimo) procese, negalime iš anksto žinoti, į kokią klasę turi patekti grupuojamas elementas. Grupavimo algoritmas, pvz. K-means grupavimo rezultatus grąžina tik kaip grupės numerį, kuriai priklauso kiekvienas nagrinėjamas elementas, o klasifikavimo algoritmai grąžintų aiškių kategorijų (pvz. „Sportas“) nuorodas. Taigi grupavimo atveju belieka pasikliauti papildoma informacija, kuri dažniausiai yra turima dokumentų klasifikacija, jų kategorijos. Tik šiuo atveju yra nagrinėjamas ne tam tikro straipsnio patekimas į tam tikrą kategoriją, o visų įmanomų straipsnių porų išskyrimas arba neiškyrimas. Tuomet minėti grupavimo atvejai naujienų straipsnių grupavimui gali būti apibrėžti:

- *tp* – naujienų straipsnių poros, priklausančios tai pačiai naujienų kategorijai ir sugrupuotos į tą pačią grupę;
- *tn* - naujienų straipsnių poros, priklausančios skirtingoms naujienų kategorijoms ir sugrupuotos į skirtingas grupes;
- *fp* - naujienų straipsnių poros, priklausančios skirtingoms naujienų kategorijoms, bet sugrupuotos į tą pačią grupę
- *fn* - naujienų straipsnių poros, priklausančios tai pačiai naujienų kategorijai, bet sugrupuotos į skirtingas grupes.

Literatūroje [6,8,14,41,49,50,51,52,53,54] galima rasti šiuos grupavimo algoritmų kokybę įvertinančius parametrus:

- Grynumas (angl. *purity*);
- Entropija (angl. *entropy*);
- Tikslumas (angl. *precision*);
- Atkūrimas (angl. *recall*);
- Tikslumas 2 (angl. *accuracy*);
- *F* – balas;
- MCC (angl. Mathews correlation coefficiant);
- NMI (angl. *normalized mutual information metric*);
- *c* (angl. *cophenetic correlation coefficient*);
- S (angl. *silhouette value*).

Apžvelgus daugelį analizuotų straipsnių sužinota, kad *F*-balo parametro variantas, F_1- balas, yra pats populiariausias. Toliau dar dažnai sutinkamas parametras yra tikslumas, entropija, o tam tikriems, specializuotiems atvejams, kiti. Pavyzdžiui *c* (angl. *cophenetic correlation coefficient*) taikomas palyginti rezultatus, gautus naudojant skirtingus atstumų tarp taškų VSM modelyje skaičiavimus. [53] išskiriama, kad turėtų būti naudojamas MCC parametras, nes tai yra balansuota metrika, apimanti visus sumaišymo lentelės kvadrantus.

Žemiau pateiktas metrikų tikslumo *P*, atkūrimo *R*, F_1 ir MCC skaičiavimas:

$$P = \frac{tp}{tp + fp} \tag{1.2}$$

$$R\,\frac{tp}{tp + fn} \tag{1.3}$$

$$F_1 = \frac{P \cdot R}{P + R} \tag{1.4}$$

$$MCC = \frac{tp \cdot tn - fp \cdot fn}{\sqrt{(tp + fp)(fp + fn)(tn + fp)(tn + fn)}} \tag{1.5}$$

Galima apibendrinti, kad norint įvertinti grupavimo tikslumą, tereikia rasti sumaišymo matricos vertes ir iš jų išskaičiuoti F_1 ir MCC metrikas. Taip pat būtina atkreipti dėmesį, kad grupavimui sumaišymo matricos vertės skaičiuojamos specifiniu būdu, vertinant elementų porų išskyrimą ar neišskyrimą.

1.5. Egzistuojantys naujienų agregatoriai

Šiame darbe tiriamas naujienų grupavimo uždavinys pritaikomas daugelyje internete randamų naujienų agregatorių. Šie skenuoja pasirinktas naujienų svetaines ir jas visas apibendrina, išskiriant naujienų kategorijas su tam tikru metu esančiomis populiariausiomis ar aktualiausiomis naujienomis. Šiame poskyryje apžvelgiami pasaulyje ir Lietuvoje rasti naujienų agregatoriai.

1.5.1. Pasaulyje

Pasaulyje yra daugybė įvairių naujienų grupavimo realizacijų, pateiktų svetainių pavidalu. Daugumos jų funkcionalumas grupavimo prasme išsiskiria. Apibendrintas atitikimas įvairiems funkcionalumo aspektams bei svetainių populiarumas, pagal *SimilarWeb* pateikiamą apsilankymų skaičiaus statistiką pateikiamas 1.3 lentelėje.

1.3 lentelė Įvairių naujienų agregatorių savybės

Naujienų agregatorius	**Populiariausių naujienų pateikimas**	**Naujausių naujienų pateikimas**	**Kelių naujienos šaltinių radimas**	**Naujienų priskyrimas kategorijoms**	**Apsilankymų skaičius, tūkst. per 2017 m. spalio mėnesį**
Google News [55]	++	++	++	++	511 500
News Now [56]	++	++	-	+++	43 000
Digg [57]	+	+	-	++	19 000
Slashdot [58]	+	+	-	++	14 300
Techmeme [59]	+	++	+++	+	2 300
Newstral [60]	+	+	+	++	600
Newsmap [61]	+	+	-	+	220
News Sola [62]	+	+	-	+	55
Bing News [63]	++	++	-	++	
Wikipedia: Current Events [64]	++	++	-	++	
Reddit/r/news [65]	++	+	-	+	
The States Report [66]	+	+	-	++	
Virwire [67]	++	++	-	-	
Freshnews.org [68]	+	++	-	-	
Corax [69]	+	+	-	-	

Kaip matyti 1.3 lentelėje, sunkiausia naujienų grupavimo svetainėms rasti tos pačios naujienos šaltinius – tokiu funkcionalumu pasižymi tik kai kurios populiariausios svetainės. Lengviausiai realizuojamas populiariausių ir naujausių naujienų pateikimas. Dera paminėti, kad naujienų priskyrimas kategorijoms taip pat yra daugiau realizuojamas, bet dauguma tai yra iš anksto apibrėžtos kategorijos, o ne sąlygojamos to meto esančių įvykių ir generuojamos automatiškai. Analizuotos svetainės neteikia duomenų, kokiu būdu jos automatiškai sugrupuoja naujienas. Dauguma produktų vis tiek neužtenka automatizuotų sistemų ir 1.3 lentelėje minėtus funkcionalumus padeda įgyvendinti samdomi redaktoriai, pvz. svetainėse Techmeme, digg, Corax .

Pati populiariausia naujienų grupavimo svetainė neabejotinai yra *Google News*. Čia ne tik pateikiamos naujienų kategorijos, populiariausios, naujausios naujienos, bet ir kiekvienai naujienai parodomi alternatyvūs šaltiniai. Tai neabejotinai yra geriausio funkcionalumo grupavimo svetainės sprendimas.

Kita, Jungtinėje Karalystėje pati populiariausia naujienų grupavimo svetainė yra *News Now*. Čia, kitaip nei *Google News*, nėra pateikiami konkrečios naujienos keli šaltiniai, bet vietoj to pateikiami apibendrintos temos šaltiniai, pavyzdžiui, *Donald Trump*, ar *North Korea*. Tai rodo puikų naujienų skirstymą pagal temas.

Gan išskirtinesni yra *Newsmap* ir *News sola* sprendimai. Šie skiriasi tik tuo, kad *Newsmap* vizualizavimui naudoja *Flash*, o *News sola – HTML5*. Čia visos naujienos pateikiamos blokais, kurių skiriasi dydis, spalva ir ryškumas. Didesni blokai reprezentuoja svarbesnius straipsnius, ryškesni – naujesnius, o skirtinga spalva – skirtingą naujienų kategoriją. Tai yra gan originalus sprendimas. Vis dėlto, kad ir kokie originalūs šie tinklalapiai atrodo, jie naujienas gauna iš *Google News* ir atlieka tik jų pavaizdavimą, panaudodami *treemap* algoritmą. Be to, kai kurie naujienų blokai yra tokie maži, kad nieko juose negalima įskaityti (žr. 1.2. pav.).

1.2. pav. *News sola* svetainė

Taigi pasaulyje egzistuoja daugybė naujienų grupavimo problemos sprendimų. Pagrindiniai iššūkiai su kuriais susiduriama – tai kelių naujienos šaltinių radimo funkcionalumo įgyvendinimas, pilnas grupavimo proceso automatizavimas, reprezentatyvus grupavimo rezultatų pateikimas.

1.5.2. Lietuvoje

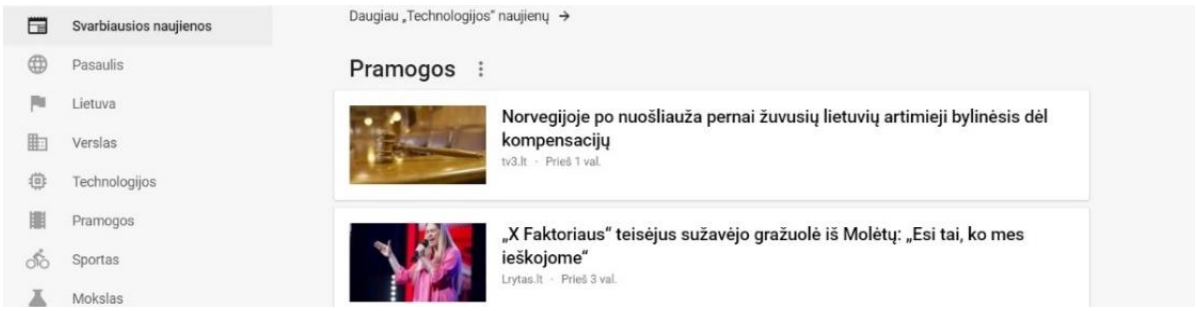
Situacija Lietuvos rinkoje yra skurdi, palyginus su pasaulio. Čia vyrauja vienintelis naujienų agregatorius *Google Naujienos*. Tai yra angliškosios versijos analogas lietuvių kalba, naudojantis Lietuvos internetines žiniasklaidos priemones. Buvusi antroji tokio tipo svetainė, pavadinimu *visosnaujienos.lt*, užsidarė 2018 m. Ši gerokai atsiliko nuo *Google Naujienos*: nebuvo tos pačios naujienos pateikimo skirtinguose šaltiniuose funkcijos, negalėjo sugrupuoti naujienų srauto pagal temas (žr. 1.4 lentelė). Tai galėjo ir būti priežastys, lėmusios *Google Naujienos* konkurencinį pranašumą.

1.4 lentelė Lietuviškų naujienų agregatorių palyginimas

	Google Naujienos	**Visosnaujienos.lt**
Populiariausių naujienų išskyrimas	++	++
Greitas naujienų pateikimas	++	++
Kelių naujienos šaltinių radimas	+	-
Naujienų priskyrimas kategorijoms	+	+

Google Naujienos puslapis pirmiausia buvo kurtas angliškajai auditorijai ir tik vėliau pritaikytas kitų kalbų vartotojams. Kadangi lietuvių kalba yra sudėtinga ir skiriasi nuo anglų kalbos, gali būti, kad grupavimas būtų pagerintas, jei į tai būtų labiau atsižvelgta.

Lietuviškojoje *Google Naujienos* versijoje galima kartais pastebėti klaidingų sugrupavimų. Pavyzdžiui, 2017 m. spalio 7 dienos naujiena „*Norvegijoje po nuošliauža pernai žuvusių lietuvių artimieji bylinėsis dėl kompensacijų*“ priskirta prie „*Pramogos*“ skilties (žr. 1.3 pav.). Iš vienos pusės, straipsnis apie žūtį tikrai neteisingai priskirtas prie „*Pramogos*“ skilties, iš kitos pusės – žūtis juk įvyko pramogaujant. Aišku yra tai, kad esama naujienos straipsnio reprezentacija buvo nepakankama teisingai priskirti naujienos kategoriją.



1.3 pav. *Google Naujienos* svetainėje matomas neteisingas straipsnio priskyrimas pramogų skilčiai

Taigi Lietuvoje naujienų grupavimo sprendimų rinka yra skurdi, dominuoja vienintelė sistema Google Naujienos. Jos funkcionalumas yra bene geriausias iš pasaulyje esančių naujienų grupavimo

sistemų. Nepaisant to, net ir ši sistema neišvengia grupavimo klaidų. Kadangi ji buvo kurta pagal anglų kalbai skirtos programos versiją, tikėtina, kad lietuviškoji versija galėtų būti patobulinta, labiau atsižvelgiant į lietuvių kalbos specifiką.

1.6. Programinės priemonės

Panašiuose projektuose naudojamos įvairios programavimo kalbos ir priemonės. Dažniausiai pastebima *Java* [24,25], rečiau – *Scala* [4]. Šiuo metu vis labiau populiarėja *Python* [8,53] programavimo kalba. Paskutinioji pasižymi paprastu, įvairių palaikomų bibliotekų kiekiu bei gausybe pavyzdžių, kuriuos galima rasti internete. Taip pat ši kalba yra brandi, skaičiuojanti keletą gyvavimo dešimtmečių. Dėl šių priežasčių šiame projekte ir pasirinkta naudoti *Python* programavimo kalbą.

Naujienų straipsnių išgavimui iš svetainių bus naudojamas specialus *Python* kalboje veikiantis *Scrapy 1.5* karkasas. Duomenims saugoti bus naudojama *PostreSQL* duomenų bazė su *Psycopg2* komunikavimo sistema. Šių ir kitų įrankių pasirinkimo argumentai gan plačiai išdėstyti 2017 m. tapusio magistru Nathan S. Guerin baigiamojo darbo apie naujienų grupavimą tezėje [8]. Toliau šiame poskyryje bus aprašytos bibliotekos, įgyvendinančios algoritmus, reikalingus teksto grupavimo procesui.

1.6.1. *Scikit-learn* biblioteka

Šioje bibliotekoje randami įvairūs populiariausi mašininio mokymosi algoritmai. Taip pat įrankyje galima rasti įvairių klasių, įgyvendinančių teksto apdorojimo funkcijas. Čia galima rasti pirminio teksto apdorojimo, BOW modelio, TF-IDF pasvėrimo, grupavimo algoritmų įrankius.

Bibliotekos klasė *CountVectorizer* atlieka pirminį teksto apdorojimą ir skirta gauti žodžių maišo modelį iš dar neapdoroto teksto. Čia galima pasirinkti šiuos aktualius parametrus:

- maksimalus žodyno ilgis;
- maksimalus procentas, keliuose dokumentuose gali būti kiekvienas žodis;
- minimalus procentas, keliuose dokumentuose gali būti kiekvienas žodis;
- *n-gram* skaičius, t. y. ar požymiai turi būti sudaryti iš vieno žodžio, ar po du, ar po tris ir t. t.;
- ar didžiosios raidės turi būti paverstos mažosiomis;
- ar iš teksto išskiriami žodžiai, ar tik simboliai.

Algoritmui taip pat galima paduoti sąrašą nereikšminių žodelių, kurie bus išfiltruoti. Deja, bibliotekoje nėra lietuviškų nereikšminių žodelių sąrašo, taip pat čia nėra galimybės atlikti šaknies išgryninimo, lemavimo veiksmus. Iš kitos pusės, *CountVectorizer* galima naudoti savo paties pasirašytą teksto analizatorių, kuris gali atlikti šiuos veiksmus.

TF-IDF terminų reprezentaciją galima išgauti su klase *TfidfVectorizer*, kuri tuo pačiu metu atlieka ir visus *CountVectorizer* klasės pirminio teksto apdorojimo veiksmus. Jei naudojami terminai tik iš nurodyto žodyno, kad būtų išvengta dalybos iš nulio, TF-IDF skaičiavimo schemoje yra pridedamas

papildomas dokumentas, turintis visus žodyno terminus. *TfidfVectorizer* taip pat galima pasirinkti naudojamą normalizaciją bei nurodyti naudoti tik *tf* (angl. *term frequency*) svorius.

Scikit-learn bibliotekoje taip pat gausu įvairių rūšių grupavimo algoritmų:

- K-means;
- affinity propagation;
- mean-shift;
- spektriniai;
- hierarchiniai;
- DBSCAN;
- Gauso mišinių;
- Birch;

K-means algoritmą įgyvendina *KMeans* klasė. Čia galima pasirinkti, ar pradiniai grupių centrai būtų pasirinkti atsitiktinai, ar algoritmo *k-means++* būdu, kai juos parenkama atsitiktinai taip, kad tarpai tarp jų būtų kuo didesni. Taip pat galima užduoti pradinius centroidų taškus rankiniu būdu. Kitas svarbus parametras – maksimalus iteracijų skaičius. Svarbu atkreipti dėmesį, kad ši klasė leidžia atlikti lygiagrečius skaičiavimus, nurodant grupavimo darbų skaičių *n_jobs*. Taigi, kaip matyti, *K-means* algoritmas *Scikit-learn* bibliotekoje įgyvendintas naudojant visus geriausius žinomus patobulinimus.

1.6.2. *Gensim* biblioteka

Ši *Python* biblioteka skirta dokumentų temų modelių sudarymui. Jos pavadinimas kilęs nuo angliškų žodžių *generate similar*, kas turėjo reikšti „*rasti panašiausią dokumentą*“. *Gensim* pritaikyta atlikti veiksmus su dideliais tekstinių duomenų kiekiams, naudojant vieną ar daugiau skaičiavimo mašinų. Kitaip, nei *Scikit-learn*, čia naudojami duomenų generatoriai, kurie duomenis į atmintį užkrauna ne vienu metu, o mažomis porcijomis. Ši savybė leidžia apmokyti mašininio mokymosi modelius naudojant didelius duomenų rinkinius.

Gensim bibliotekos klasė *doc2vec* įgyvendina dokumentų reprezentacijos *doc2vec* [22] modelį. Dėka minėtų generatorių įgyvendinimo, dokumentų vektorius galima gauti modelį apmokant su dideliais dokumentų kiekiais. Taip pat galima gauti vektorinę dokumento, nenaudoto apmokyme, reprezentaciją, naudojant su kitais duomenimis apmokytą modelį. Šios savybės *doc2vec* klasę paverčia gana universalia.

doc2vec klasei galima parinkti tokius svarbiausius parametrus:

- mokymosi algoritmas; PV-DM, arba PV-DBOW (atitinka CBOW žodžiams);
- pradinė mokymosi sparta;
- maksimalus atstumas tarp dabartinio ir spėjamo žodžio sakinyje;
- apmokomų dokumentų vektorių dydžiai;
- minimalus žodžių dažnis, kad šie nebūtų išfiltruoti;

- maksimalus žodyno ilgis;
- skaičiavimo gijų skaičius;
- mokymosi iteracijų (epochų) skaičius;
- veiksmai su konteksto žodžiais – skaičiuoti vidurkį, sumą, ar neagreguoti.

1.6.3. Neuroninių tinklų skaičiavimų biblioteka

Remiantis [70], 2018 m. trys pačios populiariausios neuroninių tinklų bibliotekos buvo šios:

1. *TensorFlow*;
2. *Keras*;
3. *PyTorch*.

Nors padeda atlikti tuos pačius uždavinius, šios bibliotekos išsiskiria savo savybėmis. Pati populiariausia, *TensorFlow* biblioteka, sukurta *Google*, pasižymi tiek plačiausiu panaudojimu, tiek didelėmis galimybėmis. Ji plačiai naudojama tiek tyrimuose, tiek produkcijoje. Antra pagal populiarumą *Keras* biblioteka yra orientuota į tuos programuotojus, kurie mažiau nusimano apie dirbtinių neuroninių tinklų taikymą ir sukurta taip, kad ja būtų kuo paprasčiau naudotis. Dėl tokio paprastumo, *Keras* neturi tiek daug architektūrinių kūrimo galimybių, kiek jų turi *TensorFlow* biblioteka. Tuo tarpu trečioje vietoje pagal populiarumą esanti *Facebook* sukurta *PyTorch* biblioteka apima pirmųjų abiejų savybes – yra paprasta naudoti (nors ne taip lengva kaip *Keras*), bet galimybėmis lyginasi su *TensorFlow*. Šiame projekte yra svarbu lengvas naudojimas ir kuo įvairesnė neuroninių tinklų architektūrų kūrimo galimybė, todėl bus naudojama minėtomis tarpinėmis savybėmis pasižyminti *PyTorch* biblioteka.

Viena esminių *PyTorch* bibliotekos ypatybių yra ta, kad ji veikia naudojant dinaminį skaičiavimo grafą. Kitos bibliotekos, pavyzdžiui *Keras* ar *TensorFlow* naudoja statinį grafą. Paskutiniuoju atveju sudarius architektūrą, skaičiavimai yra paleidžiami iškart visai programai ir tikrinamas jos įvykdytas rezultatas. Tuo tarpu *PyTorch* bibliotekoje paleidus programos vykdymą galima stebėti skaičiavimus ar juos keisti jų vykdymo metu. Dėl to yra ne tik daug lengviau suprasti programos veikimą, jį testuoti, bet ir pats kodo kūrimas vyksta daug greičiau. Šios savybės yra ypač svarbios įvairioje eksperimentinėje veikloje, kokia ir yra šis projektas.

Visos šiame poskyryje nagrinėtos bibliotekos turės būti integruotos į vieną sistemą. *Scikit-learn* bibliotekos *CountVectorizer* ir *Gensim doc2vec* klasių iškvietimo parametrai yra skirtingi (išskyrus maksimalų žodyno ilgį), todėl tai sudaro papildomų abiejų bibliotekų integracijos iššūkių. Vis dėlto, jos turi būti integruojamos, nes *Scikit-learn* neturi įgyvendintos *doc2vec* teksto reprezentacijos, o *Gensim* apsiriboja tik temų modeliais. Tuo tarpu *Pytorch* biblioteka sudaryto autoenkoderio atliekamą BOW vektoriaus dimensiškumo mažinimą būtų galima tiesiog įterpti kaip papildomą galima žingsnį prieš VSM modelyje esančių duomenų grupavimą.

2. PROJEKTINĖ DALIS

2.1. Sistemos paskirtis

Sistemos paskirtis – atlikti naujienų grupavimo eksperimentus parenkant įvairius dokumentų vektorizavimo parametrus. Tokiu būdu bus išbandyti įvairūs vektorizavimo algoritmai ir rasti tinkamiausi filtrai pirminiam teksto apdorojimui. Tikimasi, kad bus patogu parinkti įvairius eksperimentų parametrus, o gautus rezultatus vaizdžiai pavaizduoti. Tokiu būdu bus galima ne tik rasti geriausius parametrus, bet ir matyti jų poveikį grupavimo rezultatams. Taigi sistema automatizuos įvairių grupavimo eksperimentų atlikimą ir palengvins optimalių vektorizavimo parametrų radimą.

Sistema nėra skirta atlikti naujienų agregatoriaus funkcijas. Jos vykdomų eksperimentų metu turėtų būti randamos tokios parametrų vertės, kurios jau vėliau galėtų būti naudojamos galimiems naujienų agregatorių kūrimams. Taigi šio projekto sistema skirta tik grupavimo tyrimui, o ne jo panaudojimui.

2.2. Reikalavimų analizė

Sistemos projektavimo pradžioje buvo atlikta reikalavimų analizė. Nustatyti šie funkciniai reikalavimai:

- produktas turi parsiųsti žodžių lemas;
- produktas turi parsiųsti naujienų straipsnius;
- produktu turi būti galima pasirinkti kiek ir kurių metų straipsnių norima grupuoti;
- produktu turi būti galima sukonfigūruoti dokumentų vektorizavimą;
- produktu turi būti galima paleisti ir atlikti eksperimentą;
- produktas turi išsaugoti eksperimento rezultatus;
- produktu turi būti galima pavaizduoti eksperimento rezultatus.

Buvo išsiaiškinta, kad pagrindiniai projekto sistemos vartotojai yra dviejų tipų – studentai ir mokslo darbuotojai, apibendrintai vadinami tyrėjais. Kadangi tyrėjų patirtis informacinėse technologijose yra tolygi pažengusių ar dar didesnė, šie atliks tiek paprasto vartotojo, tiek administratoriaus funkcijas. Dėl to keliami reikalavimai, kad produktą būtų galima lengvai modifikuoti ir prižiūrėti, juo būtų lengva išmokti naudotis ir būtų patrauklus, taupytų brangų tyrėjų laiką. Detali būsimų vartotojų charakteristika pateikta 2.1 lentelėje.

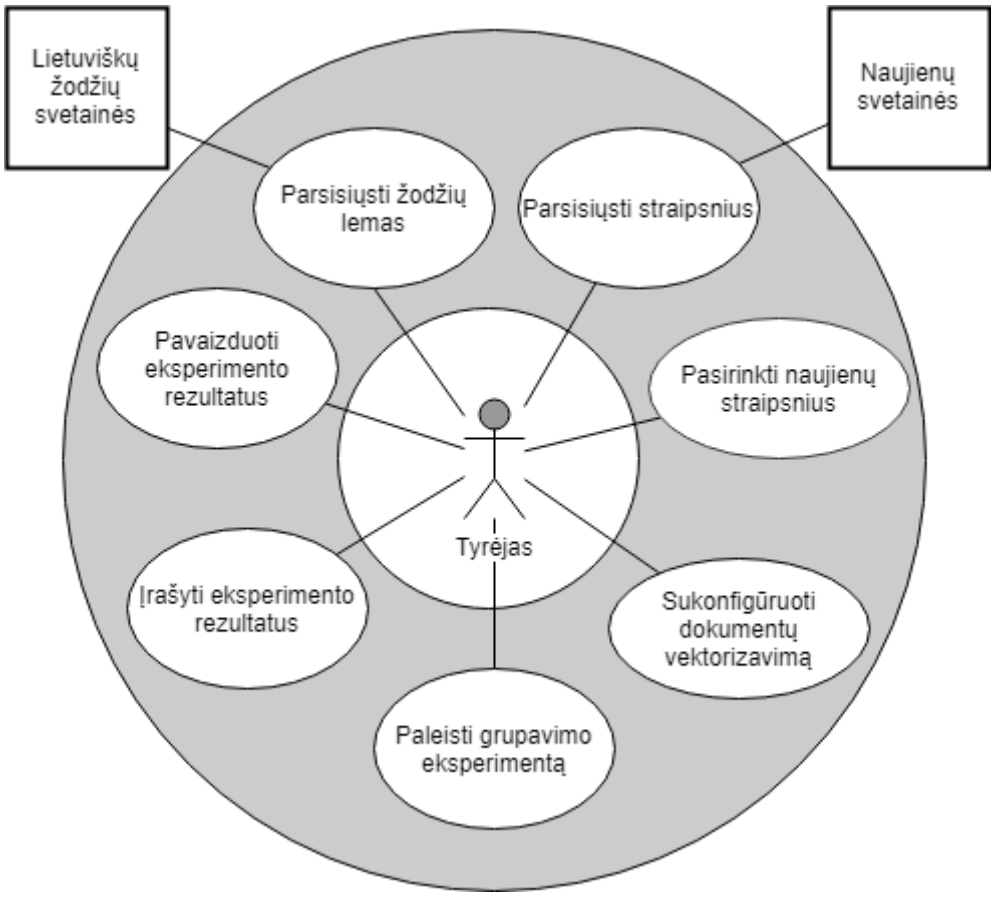
Kiti išskirti nefunkciniai reikalavimai:

- produktas turi neapsunkinti darbo detalėmis apie savo vidinę struktūrą;
- produktas turi naudoti suprantamus žodžius ir simbolius;
- produktas turi atsiminti svarbiausius rankiniu būdu įvedamus duomenis ir gebėti iškart pateikti kaip numatytuosius;
- bet koks programinės įrangos atsakas į vartotojo veiksmus turi įvykti greičiau nei per 3 s;

- visi sudėtingi ilgiau nei 2 s trunkantys programinės įrangos skaičiavimai turi interaktyviai rodyti, kiek jau skaičiavimų atlikta, ir kiek dar liko;
- produktas turi veikti be klaidų nepertraukiamai mažiausiai 100 val;
- produktas turi sklandžiai veikti, apdorodamas ir naudodamas iki 7000 straipsnių;
- produktas turi sklandžiai veikti mažiausiai 2 metus.

2.1 lentelė. Vartotojai

Vartotojo kategorija	**Studentas**	**Mokslo darbuotojas**
Vartotojo sprendžiami uždaviniai	Atlikti dėstytojų paskirtus darbus	Vykdyti projektus, vykdyti mokslinį tiriamąjį darbą, atlikti tyrimus
Patirtis dalykinėje srityje	Pažengęs	Įgudęs
Patirtis informacinėse technologijose	Pažengęs	Įgudęs
Papildomos charakteristikos	19–25 metų jaunuoliai, daugiausia vaikinai, studijuojantys informatikos mokslo studijų programas	> 25 metų žmonės, turintys daktaro laipsnį, lietuviai arba užsieniečiai
Vartotojų prioritetai	Antraeiliai vartotojai	Svarbiausi vartotojai



2.1 pav. Panaudojimo atvejų diagrama

Reikalavimų surinkimo metu sudaryta panaudos atvejų diagrama parodyta 2.1 pav. Kaip matyti, ji apima visas eksperimento atlikimo fazes – nuo duomenų surinkimo (žodžių lemų ir naujienų straipsnių parsiuntimo), jų apdorojimo, reprezentavimo (vektorizavimo), žinių išgavimo (grupavimo)

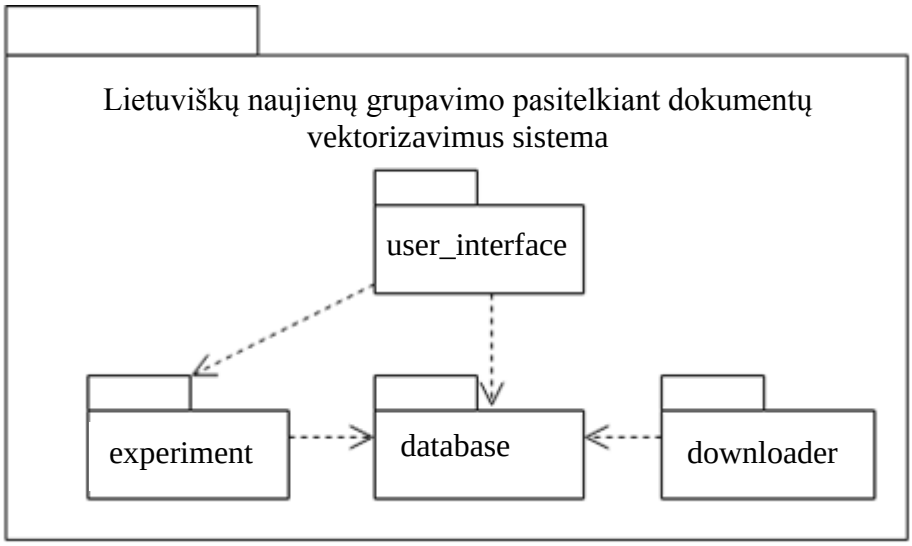
iki rezultatų įrašymo ir pavaizdavimo. Visi šie grupavimo proceso žingsniai jau buvo detaliai išnagrinėti 1 skyriuje.

2.3. Sistemos projektavimas

Šiame poskyryje pateikiama sukurtos sistemos specifikacija. Naudojami sistemos išdėstymo, statinis, dinaminis bei duomenų vaizdai.

2.3.1. Sistemos statinis vaizdas

Projekto sistema aukščiausiame lygmenyje suskaidyta į 4 pagrindinius paketus pagal tai, kokius uždavinius atlieka. Šie paketai ir jų ryšiai pavaizduoti 2.2 pav.



2.2 pav. Projekto sistemos pagrindiniai paketai

Paketas „user_interface“ skirtas įgyvendinti visas sistemos užduotis, susijusias su vartotojo sąsaja. Čia yra metodai skirti tiek kiekvieno eksperimento proceso, tiek jo rezultato, tiek keleto eksperimentų rezultatams pateikti grafiniu pavidalu. Jame yra įgyvendintas automatinis daugelio eksperimentų vykdymas, kreipiantis į kitą paketą „experiment“. Taip pat sukurti vaizdavimo metodai, pateikiantys tiek vieno, tiek dviejų, trijų ar keturių kintamųjų eksperimentų rezultatų pasiskirstymus. Paketas prisideda beveik prie visų panaudojimo atvejų realizavimo, pavaizduodamas aktualiausius juose vykstančius procesus.

„experiment“ paketas atsakingas už visų eksperimentų su įvairiais parametrais įvykdymą. Čia yra patalpinti ir kiti sudėtiniai paketai:

- „clustering_*alg*“ realizuotos visos funkcijos grupavimo įvykdymui;
- „defaults“ pakete saugomos visos numatytosios parametrų reikšmės;
- „My_Metrics“ pakete įgyvendinti įvairių grupavimo įverčių skaičiavimai;
- „preprocessing“ patalpintos visos klasės įgyvendinančios įvairių teksto reprezentacijų gavimą (žr. klasių diagramą 1 priede);

- „run_experiment“ pakete įgyvendintos funkcijos reikalingos vieno eksperimento paleidimui ir rezultatų gavmui;
- „Simple_BOW_autoencoder“ paketas įgyvendina neuroninio tinklo autoenkoderio modelį.

„experiment“ paketas gali veikti ir be „user_interface“ paketo pagalbos, tačiau šiuo atveju vienu metu bus įvykdomas tik vienas eksperimentas, o šio parametrų vertes reikės nurodyti rankiniu būdu.

„database“ paketas atlieka pagrindinius veiksmus su duomenimis. Tai apima eksperimentų, žodžių formų, naujienų straipsnių duomenų įrašymą ir nuskaitymą į duomenų bazę. Taip pat čia yra įgyvendinti metodai skirti teksto suskaidymui į terminus, jų išfiltravimui, straipsnių kategorijų normalizavimui. Taigi šis paketas padeda „experiment“ paketui pateikdamas duomenis, paruoštus teksto reprezentacijos skaičiavimui.

Paketas „downloader“ atlieka duomenų iš interneto parsiuntimo užduotis. Tai apima žodžių formų ir lemų bei naujienų straipsnių parsiuntimą. Jo pagalba realizuojami panaudojimo atvejai „Parsisiųsti žodžių lemas“ bei „Parsisiųsti straipsnius“. Paketo pagrindas yra *Scrapy 1.5* biblioteka, pritaikyta pasirinktoms naujienų straipsnių ir žodžių formų svetainėms. Kadangi šis paketas yra naudojamas retai, jai specialios grafinės vartotojo sąsajos nėra. Paketas bus naudojamas darbo sistema pradžioje, o vėliau tik tuomet, jei bus nuspręsta atnaujinti turimų naujienų straipsnių duomenų bazę.

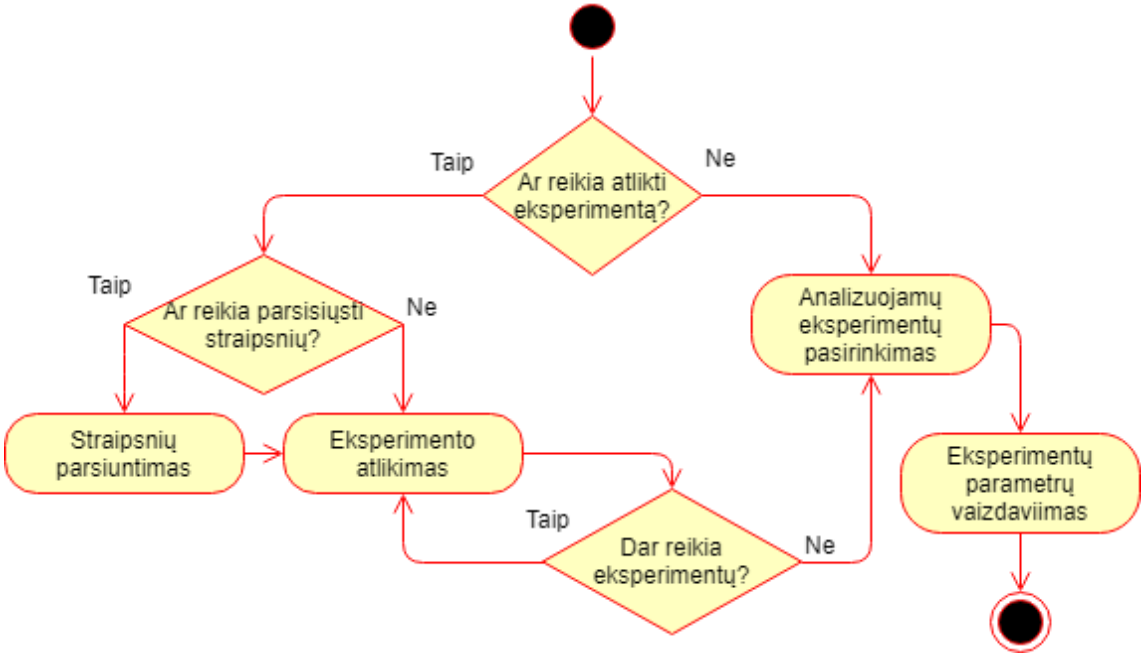
2.3.2. Sistemos dinaminis vaizdas

Pagrindinės projekto užduotys, kurioms bus naudojama sistema – eksperimentų atlikimai ir jų analizės. Norint rasti geriausio grupavimo parametrus, reikia keisti kiekvieno iš jų vertes ir sekti, ar grupavimo įvertis didėja, ar mažėja. Radus vieno iš parametrų maksimumą, šis fiksuojamas ir tuomet kaitaliojamos jau kito parametro vertės. Atlikus visų parametrų keitimo tyrimą, šis kartojamas iš naujo, nes kai kurie pakeitimai gali sąlygoti naują pagerėjimo potencialą jau keistiems parametrams. Kiekvienas toks keitimas baigiamas grupavimo įvertinimo priklausomybės nuo keistų parametrų verčių vizualizavimu. Taigi eksperimentų atlikimas ir jų analizė užima didžiausią sistemos darbo laiką.

Veiklos diagrama eksperimentų analizei parodyta 2.3 pav. Norėdamas atlikti grupavimo parametrų analizę, tyrėjas privalo būti įvykdęs bent kelis eksperimentus, kuriuose naudojami jį dominantys parametrai. Jei to nėra padaryta, tuomet yra įvykdomi reikalingi grupavimo eksperimentai. Jiems pasibaigus, grafinėje vartotojo sąsajoje pažymimi dominantys parametrai ir paleidžiama jų vizualizacija.

Pirmasis klausimas, kurį savęs turėtų paklausti tyrėjas prieš paleisdamas eksperimentą – ar yra pakankamai duomenų? Paprastai naujienų straipsniai turėtų būti parsiunčiami kartu su žodžių formomis projekto sistemos įdiegimo metu, tačiau gali pasitaikyti atvejų, kuomet norima analizuoti kitus straipsnius, nei kad tie, kurie yra duomenų bazėje. Pavyzdžiui, tai gali būti straipsniai, pasirodę ankstesniais, ar vėlesniais metais. Tuomet turėtų būti atliekamas šių straipsnių parsiuntimo žingsnis.

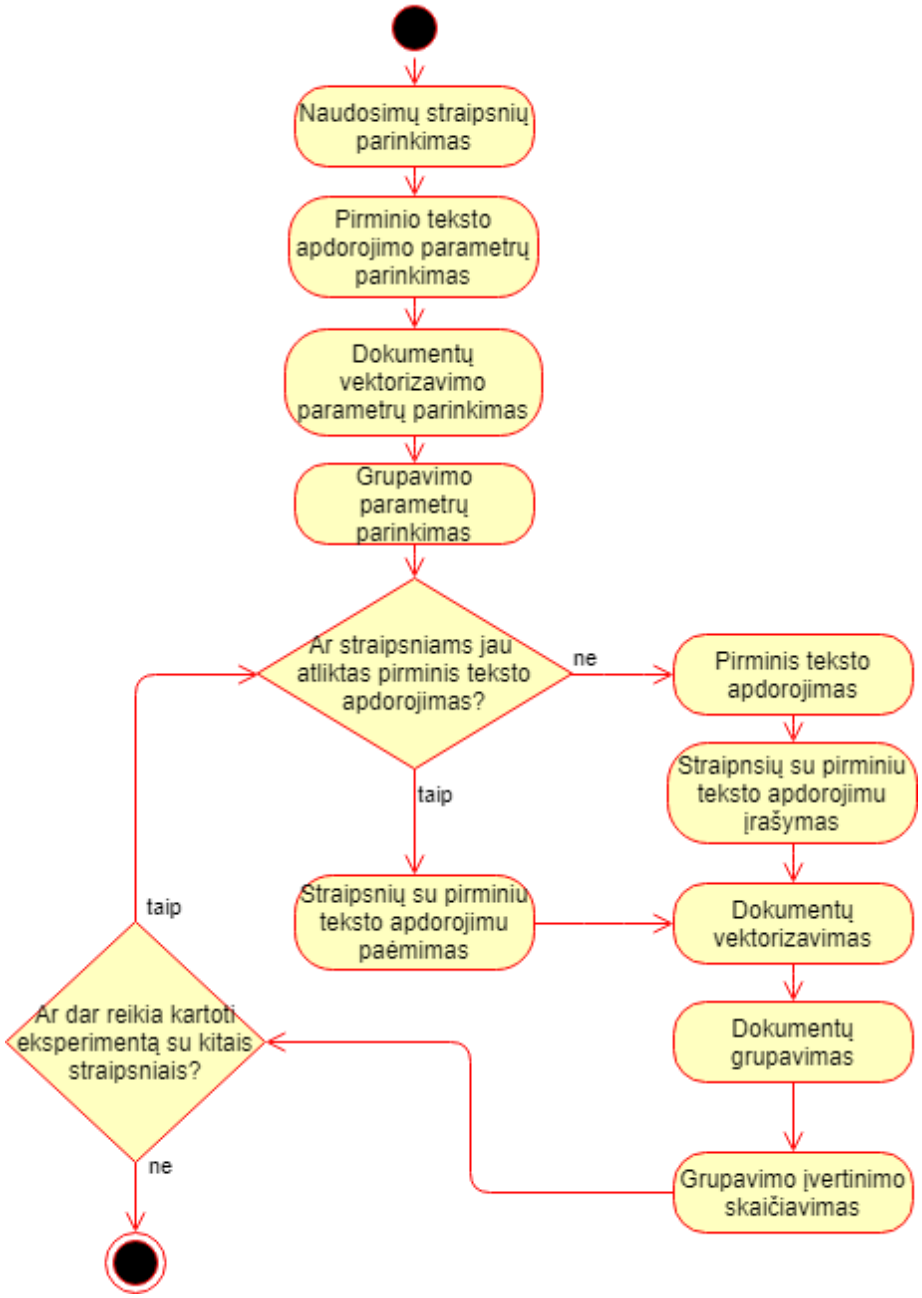
Rekomenduojama, kad turimų straipsnių skaičius bent 10 kartų viršytų kiekvieno grupavimo eksperimento metu grupuojamų straipsnių skaičių, tokiu būdu užtikrinant kuo didesnę duomenų imtį ir reprezentatyvesnius rezultatus.



2.3 pav. Veiklos diagrama eksperimentų analizei

Veiklos diagrama eksperimento atlikimui parodyta 2.4 pav. Ją galima suskirstyti į eksperimento paruošimo ir vykdymo fazes. Pirmosios metu pasirenkami visi eksperimento parametrai:

- vienu metu grupuojamų naujienų straipsnių skaičius;
- kiek kartų kartojamas grupavimas su vis skirtingais naujienų straipsniais;
- maksimalus termino aptikimo įvairiuose dokumentuose santykis (BOW modeliui)
- minimalus termino aptikimo įvairiuose dokumentuose santykis (BOW modeliui)
- maksimalus leistinas terminų skaičius;
- minimalus vieną terminą sudarančių simbolių skaičius;
- minimalus žodžių *n-gram* skaičius;
- maksimalus žodžių *n-gram* skaičius;
- trumpų ir kitų pagalbinių žodelių išmetimo sprendimas;
- teksto reprezentavimo būdas;
- dokumento vektoriaus dydis (*doc2vec* modeliui);
- epochų skaičius (*doc2vec* modeliui);
- lango ilgis (*doc2vec* modeliui);
- minimalus termino pasikartojimo skaičius (*doc2vec* modeliui).
- enkoderio ir dekoderio neuronų skaičiaus mažėjimų santykiai;
- enkoderio ir dekoderio aktivacijų funkcijų tipai.



2.4 pav. Veiklos diagrama eksperimento atlikimui

Antroji eksperimento vykdymo fazė prasideda eksperimento paleidimu. Tuomet programa patikrina, ar reikalingiems straipsniams atliktas pirminis apdorojimas. Teigiamu atveju jau apdoroti straipsniai yra paimami iš duomenų bazės. Jei pirminis teksto apdorojimas nėra atliktas, kiekvienam naujienų straipsniui jis atliekamas tokia veiksmų seka:

1. visos raidės paverčiamos mažosiomis;
2. pagal žodžių ribų požymius randami terminai ir jų seka išsaugoma;
3. jei reikia, visi skaičiai normalizuojami į vieną terminą *#NUMBER*;
4. išfiltruojami visi terminai, kurių ilgis mažesnis nei du simboliai;
5. jei taikomas, atliekamas vienas iš pasirinktų lemavimų:
    a. paliekami tik tie terminai, kuriuos pavyko lemuoti;

b. paliekami ir tie terminai, kurių nepavyko lemuoti;

6. išfiltruojami nereikšminiai žodeliai;
7. sudaromos pasirinkto ilgio (paprastai vieno žodžio) terminų sekos (dar žinomos kaip *n-gram*);

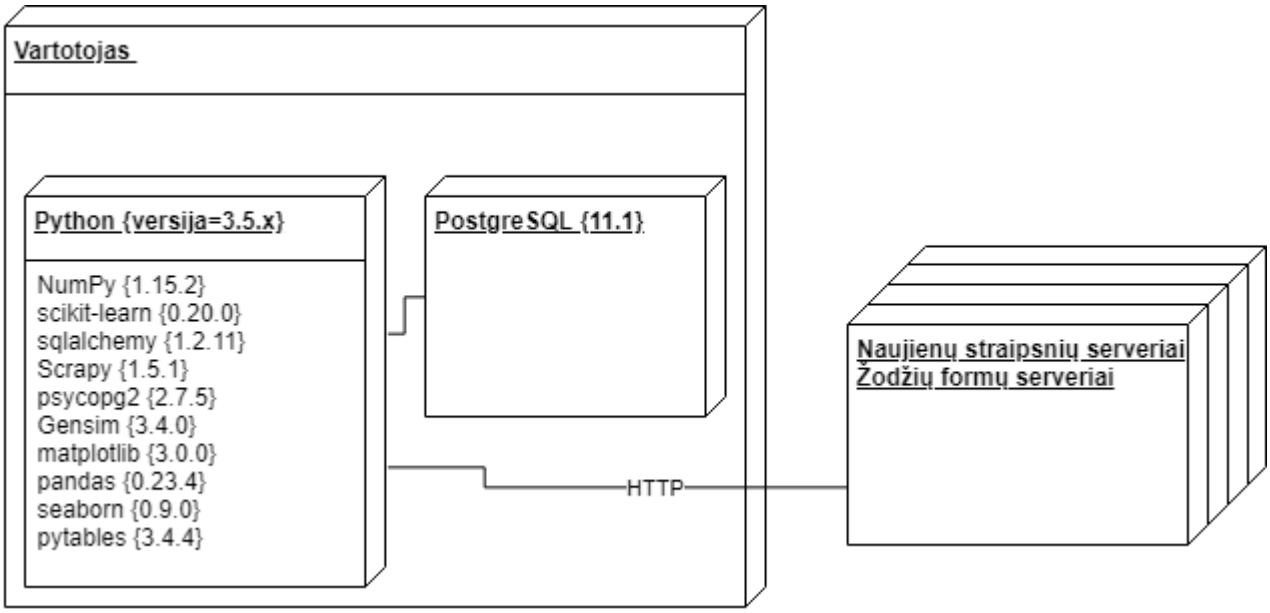
Aukščiau išvardintas pirminis teksto apdorojimas yra išsaugomas duomenų bazėje, kad ateityje jo nereikėtų kartoti. Likęs teksto apdorojimas yra vykdomas jau pačiose teksto reprezentavimo bibliotekose *gensim* bei *scikit-learn*:

1. paliekami tie terminai, kurie pasikartoja daugiau nei nurodyta kartų (*doc2vec* modeliui);
2. pagal tai, kiek procentų dokumentų aptinkamas terminas, išmetami dažniausi ir rečiausi terminai (TF-IDF modeliui);
3. išfiltruojami terminai paliekant tik nurodytą maksimalų leistiną jų kiekį.

Gauta teksto reprezentacija toliau VSM modeliu pateikiama grupavimo algoritmui, kuris atlieka grupavimą ir grąžina kiekvienam naujienų straipsniui priskirtą grupės numerį. Ši informacija kartu su straipsniams priskirtomis tikromis kategorijomis toliau perduodama įvertinimo skaičiavimo algoritmams, kur gaunami F1 ir MCC įverčiai. Galiausiai eksperimento rezultatai su suskaičiuotais įverčiais išsaugomi duomenų bazėje. Tada eksperimentas yra baigiamas vykdyti.

2.3.3. Išdėstymo vaizdas

Projektuojamos sistemos išdėstymo vaizdas parodytas 2.5 pav. Kaip matyti, pagrindinės šio projekto dedamosios techninėje įrangoje yra *Python 3.5* programavimo kalba, įvairios joje veikiančios bibliotekos, *PostgreSQL 11.1* duomenų bazė bei užtikrinta interneto prieiga pirminiam naujienų straipsnių ir žodžių formų parsiuntimui.



2.5 pav. Projekto sistemos išdėstymo diagrama

Dėl plataus *Python* programavimo kalbos paplitimo, palaikomos visos trys didžiosios operacinės sistemos – *Windows*, *Linux* ir *Mac OS X*. Vartotojas privalo turėti šiuolaikinį, mažiausiai 2 GB darbinės atminties personalinį kompiuterį. Tai gali būti ne tik fizinis, bet ir virtualus kompiuteris – tuomet sąsajos įrenginiui keliami tik kokybiškos sąsajos užtikrinimo reikalavimai, jei reikalavimai išpildomi virtualiai skaičiavimo mašinai.

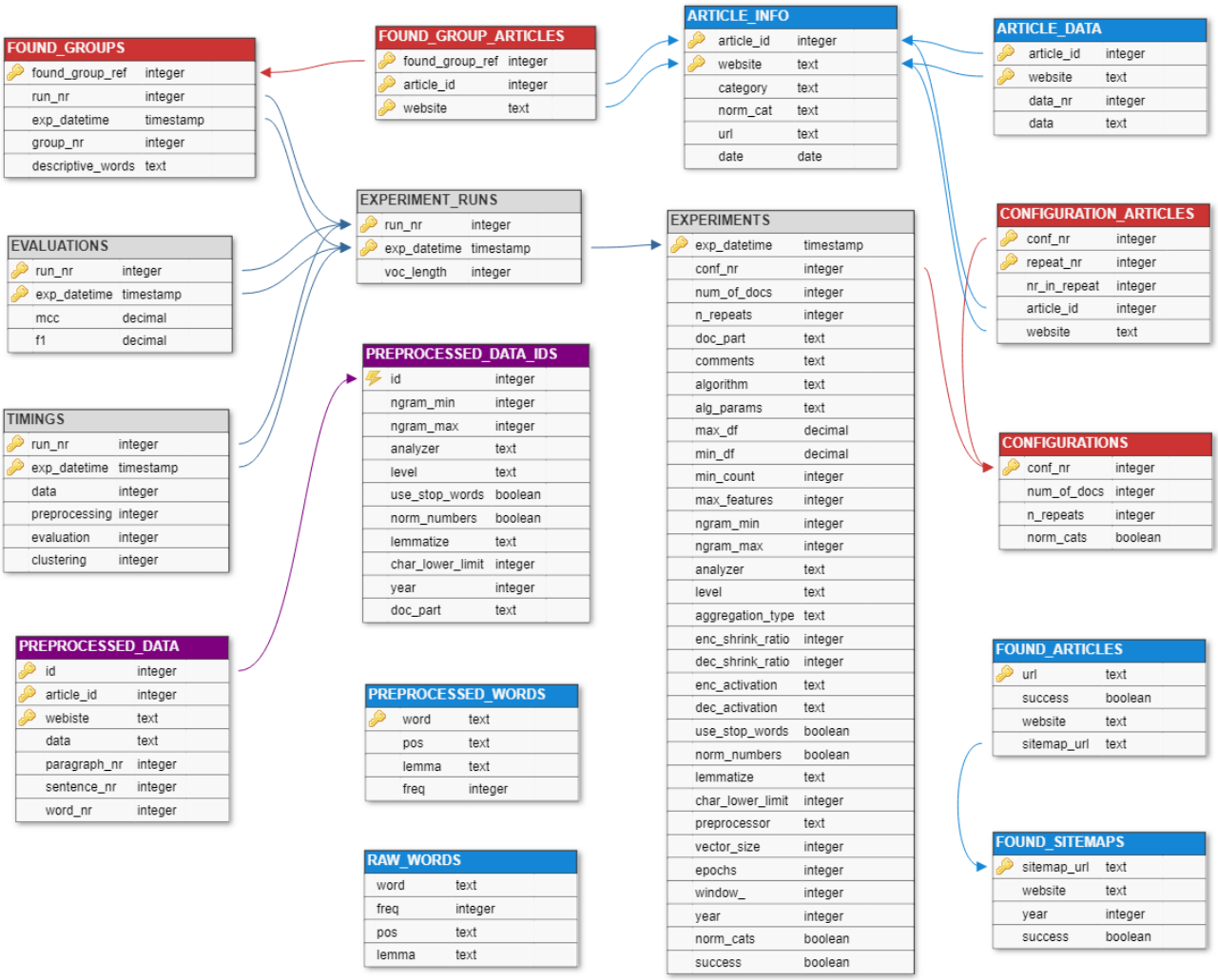
Dėl to, kad sistema bus naudojamasi individualiai, numatoma, kad duomenų bazė turi būti įdiegta toje pačioje vartotojo aplinkoje. Kintant reikalavimams ateityje, tai gali būti pakeista ir modifikuota į keleto vartotojų pasiekiamą duomenų bazę naudojantis interneto prieiga.

Visa projektui reikalinga programinė įranga yra atviro kodo ir nemokama. Tai pasakytina ne tik apie *Python* kalbą, *PostgreSQL* duomenų bazę, tačiau ir apie *Linux* operacinę sistema, kurioje dabar dažnai jau būna įdiegta *Python* programavimo kalba.

2.3.4. Duomenų vaizdas

Duomenų bazės modelis parodytas 2.6 pav. Ką tik gauti iš interneto duomenys yra išsaugomi lentelėse „FOUND_ARTICLES“, ‚FOUND_SITEMAPS“ bei „RAW_WORDS“. Atlikus šių duomenų apdorojimą jie patalpinami lentelėse „ARTICLE_INFO“, „ARTICLE_DATA“ bei „PREPROCESSED_WORDS“, iš kur bus naudojami grupavimo eksperimentuose. Keletas grupavimo eksperimentų naudoja tuos pačius straipsnių rinkinius, kurių informacija (iš kokių naujienų straipsnių sudaryti) išsaugoma lentelėse „CONFIGURATION_ARTICLES“ bei „CONFIGURATIONS“. Kiekvieno eksperimento konfigūracija išsaugoma lentelėje „EXPERIMENTS“, o jo kartojimo rezultatai lentelėse „EXPERIMENT_RUNS“, „EVALUATIONS“ ir „TIMINGS“. Taip pat kiekvieno eksperimento kartojimo metu rastų straipsnių grupių informacija įrašoma lentelėse „FOUND_GROUPS“ ir „FOUND_GROUP_ARTICLES“. Tam, kad nereikėtų atlikti to paties pirminio teksto apdorojimo vykdant panašius eksperimentus, apdoroti straipsniai ir jų informacija yra išsaugoma lentelėse „PREPROCESSED_DATA“ ir „PREPROCESSED_DATA_IDS“. Dirbtinio neuroninio tinklo apmokymo pagrindu gaunami dokumentų vektorių modeliai išsaugomi ne duomenų bazėje, o darbiniame aplanke.

Sudarytas duomenų bazės modelis užtikrina pilną eksperimentų išsaugojimą. Norint peržiūrėti anksčiau atliktą eksperimentą, tereikia rasti jo pirminį raktą *exp_datetime* lentelėje „EXPERIMENTS“, jį atitinkančius vykdytus grupavimus lentelėje „EXPERIMENT_RUNS“ bei kiekvieną grupavimą atitinkančias rastas grupes lentelėje „FOUND_GROUPS“. Taip pat visada galima patikrinti, kokie straipsnių rinkiniai buvo naudoti eksperimento metu, pagal lentelėje „CONFIGURATIONS“ išsaugotą informaciją. Visi likę eksperimento parametrai randami lentelėje „EXPERIMENTS“ prie atitinkamo eksperimento pirminio rakto. Taigi duomenų bazės modelis pilnai įgyvendina vieną iš funkcinių reikalavimų – „ Produktas turi išsaugoti eksperimento rezultatus“



2.6 pav. Duomenų bazės modelis

Duomenų bazės modulis taip pat padeda efektyvinti atliekamus eksperimentus. Kiekvienąkart prieš atliekant straipsnių pirminį apdorojimą, yra patikrinama lentelėje „PREPROCESSED_DATA_IDS“, ar nėra tikrinamiems naujienų straipsnis šis apdorojimas jau atliktas. Jei yra – tuomet iškart naudojami atitinkami duomenys lentelėje „PREPROCESSED_DATA“. Tokie atvejai pasitaiko, pavyzdžiui, atliekant grupavimo priklausomybės nuo grupuojamų naujienų straipsnių skaičiaus eksperimentus. Taigi tokiu būdu duomenų bazės modulis užtikrina kaip įmanoma greitesnį eksperimentų atlikimą.

3. EKSPERIMENTINĖ IR TYRIMINĖ DALIS

Atliekant eksperimentus siekiama nustatyti, kokiomis sąlygomis gaunami geriausi grupavimo rezultatai. Jų metu keičiami įvairūs pirminio teksto apdorojimo ir vektorizavimo parametrai, siekiant rasti tuos, kurie atitiks geriausias grupavimo įvertinimo vertes. Eksperimentų metu siekiama išsiaiškinti žemiau išvardintus konkrečius klausimus.

- Kokie yra optimalūs *doc2vec* metodo parametrai?
- Kaip skiriasi pirminio teksto apdorojimo įtaka TF-IDF ir *doc2vec* dokumentų vektorizavimams?
- Kuris vektorizavimas yra geriausias?
- Kokie grupavimo rezultatai gaunami grupuojant nenormalizuotus savaitinius duomenis?

Toliau esančiuose poskyriuose aprašyti kokie eksperimentuose buvo naudojami numatytieji parametrai, duomenys. Patikrinama sistemos realizacija ir pateikiami gauti kiekvieno eksperimento rezultatai. Taip pat pateikiama trumpa diskusija, kas dar galėtų būti padaryta panašaus pobūdžio darbuose ateityje.

3.1. Numatytieji parametrai

Daugumoje eksperimentų kiekvienas grupavimas atliktas imant 1500 straipsnių, kurie yra po lygiai pasiskirstę po 10 kategorijų. Toks gan mažas straipsnių kiekis parinktas tam, kad eksperimentai būtų vykdomi greičiau. Iš kitos pusės, siekiama turėti vienodai straipsnių, tenkančių vienai kategorijai, o tokių sunormalizuotų kategorijų lieka vos 10. Dėl šių priežasčių ir pasirinktas 1500 straipsnių skaičius.

Žinoma, kad *doc2vec* algoritmų apmokymo rezultatai taip pat priklauso nuo duomenų kiekio. Dėl to bus išbandytos dvi apmokymo galimybės:

1. apmokymui naudojami tik tie naujienų straipsniai, kurie tuo metu yra grupuojami (toks modelis toliau bus vadinamas *doc2vec_dalis*);
2. apmokymui naudojami visi duomenų bazėje turimi straipsniai (toks modelis toliau bus žymimas kaip *doc2vec_pilnas*).

Pirmu atveju *doc2vec* naudoja tuos pačius duomenis, kaip kad ir TF-IDF vektorizavimo atveju ir apmokomas gan greitai. Antruoju atveju apmokymas užtrunka daug ilgiau, dėl to sunkiau išgauti daugiau parametrų verčių, tačiau, kaip toliau bus parodyta, gaunami geresni rezultatai.

Kokie eksperimentuose naudoti numatytieji parametrai, parodyta 3.1 lentelėje. Kai kurie jų buvo optimizuoti eksperimentų metu, kiti parinkti pagal atliktą literatūros analizę ar tiesiog praktiškumą (žr. stulpelis „Optimizuota?“, 3.1 lentelė). Minėtoje lentelėje parodytos parametrų vertės bus panaudotos šiame skyriuje aprašomuose eksperimentuose, nebent bus nurodyta kitaip. Šių ir kitų parametrų parinkimą galima matyti sistemos vartotojo sąsajos lange (žr. 2 priedas).

3.1 lentelė. Eksperimentuose naudoti numatytieji parametrai

	doc2vec_pilnas	***doc2vec_dalis***	**TF-IDF**	**Optimizuota?**
Grupuotų straipsnių skaičius	1500			Ne
Grupavimo algoritmas	K-means			Ne
Architektūra	DBOW		-	Ne
Apmokymo lango ilgis	12 požymių (žodžių)		-	Taip
Minimalus žodžių pasikartojimo skaičius	4		-	Ne
Epochų skaičius	10	20	-	Taip
Dokumento vektoriaus ilgis	100	52	-	Taip
Lemavimas	Neatliekama	Nelemuoti žodžiai palikti	Nelemuoti žodžiai išmesti	Taip
Nereikšminių žodžių išmetimas	ne	taip	taip	Taip
Visų skaičių normalizavimas į vieną požymį #NUMBER	ne	taip	taip	Taip
Maksimalus žodyno ilgis	ne	40000	1000	Taip
Maksimalus terminų pasikartojimas dokumentuose	-	-	95 %	Taip

Kiekvieną kartą kartojant tą patį teksto grupavimą, galima gauti skirtingus rezultatus. Taip atsitinka dėl to, kad kai kurie algoritmai, pvz. *K-means* turi atsitiktinai parenkamas vertes, kurios kiekvieną kartą gali būti parenkamos vis kitos. Vienas būdas to išvengti – naudoti tą pačią atsitiktinio skaičių generatoriaus būseną. Tai užtikrintų grupavimo eksperimento atkartojamumą, tačiau neužtikrintų to paties rezultato atkartojamumo, naudojant kitus tekstinius duomenis. Dėl šių priežasčių kiekvienas eksperimentas yra vykdomas atliekant 50 grupavimų su skirtingais straipsnių duomenų rinkiniais ir gaunamas grupavimo rezultatų vidurkis. Be to, kiekviename kitame eksperimente vėl 50 kartų imami straipsnių rinkiniai tarp skirtingų eksperimentų yra vienodi. Tokiu būdu galima užtikrinti tiek reprezentatyvų vieno eksperimento rezultatą, tiek jo palyginimą su kitų eksperimentų rezultatais.

Minėtų 50 grupavimo rezultatų pasiskirstymas kiekviename šio skyriaus eksperimentų grafike bus vaizduojamas standartiniu nuokrypiu. Kiekvienas vidutinės MCC vertės taškas turės lydinčius du taškus, atidėtus aukščiau ir žemiau, kurie yra gaunami prie vidutinio rezultato pridėjus ir atėmus standartinį nuokrypį. Tokiu būdu bus galima atpažinti, ar stebimas kiekvieno parametro verčių dėsningumas nepatenka į paklaidos ribas, apibrėžtas minėtuoju standartiniu nuokrypiu.

Svarbu pažymėti, kad eksperimentų grafikuose nenaudojamos pritaikytos funkcijos, o tolygios linijos brėžiamos tik tam, kad būtų galima lengviau atskirti skirtingą duomenų rūšį. Tokiu būdu aiškiai išsiskiria, kurie taškai kuriai vaizduojamų taškų aibei priklauso.

3.2. Duomenys

3.2.1. Žodžiai

Siekiant atlikti žodžių lemavimo eksperimentą, buvo sudaryta įvairių žodžių formų ir juos atitinkančių lemų duomenų bazė. Tokiu būdu pirminio teksto apdorojimo metu kiekvienam duomenų bazėje esančiai žodžio formai galima rasti atitinkamą pagrindinę to žodžio formą.

Žodžių duomenys gauti iš internete laisvai prieinamų lietuvių kalbos morfologijos [71] ir morfemikos [72] duomenų bazių. Paskutinioji sudaryta „*iš skirtingų stilių, kuo įvairesnės tematikos, kiek įmanoma panašesnės apimties tekstų atkarpų iš mokslinio, publicistinio, grožinio stiliaus darbų, ir šiek tiek mažesnės apimties administracinės kalbos pavyzdžių*“ [72]. Čia iš viso yra išsaugotos 72264 žodžių formos, kartu su semantine informacija ir pavartojimų minėtuose tekstuose skaičiumi. Tuo tarpu [71] galima rasti virš 2,5 milijono žodžių formų ir virš 60 tūkst. lemų, tačiau čia nėra aišku, iš kokių tekstų žodžiai yra paimti. Dėl to abiejų semantinių duomenų bazių žodžių formos buvo apjungtos, teikiant pirmumą morfemikos duomenų bazės žodžiams. Galutinę projekto žodžių duomenų bazę sudaro 2212726 žodžių formos, tarp kurių yra 72587 lemos.

3.2.2. Straipsniai

Lietuviškų naujienų straipsniams gauti pasirinkti 3 naujienų portalai: nacionalinis *lrt.lt* bei komerciniai *delfi.lt* ir *15min.lt*. Straipsnių URL adresai buvo rasti pasinaudojant visų trijų svetainių *robot.txt* failuose buvusiomis svetainių schemomis. Iš viso buvo parsisiųsti 82793 naujienų straipsniai, apimantys atsitiktines 2017 metų datas. Jų pasiskirstymas pagal svetaines yra toks:

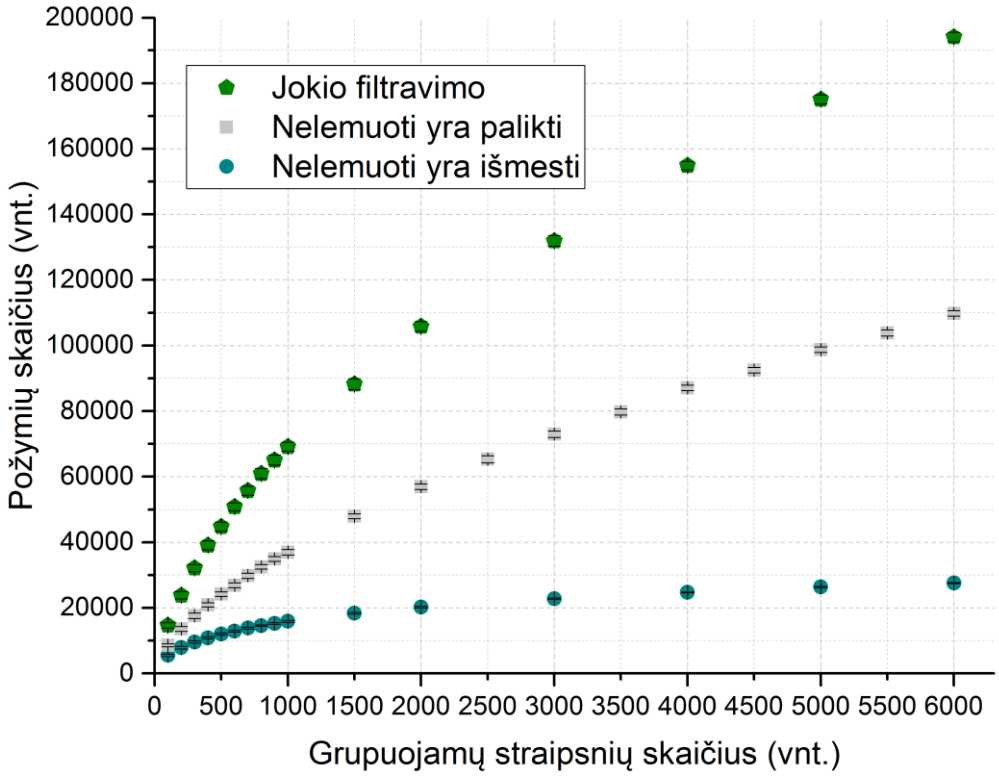
- *lrt.lt* – 26336;
- *15min.lt* – 31397;
- *delfi.lt* – 25060.

Gautas straipsnių rinkinys gali būti suskaidytas net į 30338937 terminus iš kurių 641697 yra unikalūs. Šie gali būti sumažinti dar labiau, naudojant įvarius požymių filtravimo būdus, iki:

1. 641254 (-443, -0,07 %), išmetus nereikšminius žodžius;
2. 635257 (-6440, -10,04 %), normalizavus skaičius, t. y. pavertus visus skaičius į vieną terminą;
3. 441178 (-200519, -31,25 %), žinomus žodžius lemavus ir nežinomus palikus;
4. 41933 (-599764, -93,47 %), žinomus žodžius lemavus ir nežinomus išmetus;
5. 434472 (-207225, -32,29 %), apjungiant 1, 2 ir 3 atvejus.

Dalis filtravimo būdų išfiltruoja tik nedidelę dalį požymių. Nors skaičių normalizavimas atrodo filtruojantis apie 10 % terminų, atliekant mažesnės imties straipsnių grupavimą ši dalis sumažėja (300 straipsnių normalizavimas vidutiniškai išfiltruoja mažiau nei 2 % terminų). Tuo tarpu nereikšminių žodžių sąrašas yra sudarytas rankiniu būdu, todėl jis yra natūraliai mažas. Taigi tiek skaičių normalizavimas, tiek nereikšminių žodžių filtravimas turi gan menką įtaką.

Didžiausią įtaką požymių skaičiui turi lemavimo tipo parinkimas. Ši aiškiai pavaizduota 3.1 pav. Matyti, kad grupuojant 1500 naujienų straipsnių, atlikus lemavimą požymių vidutiniškai sumažėja nuo 88 261 iki 47 963. Jei nežinomos žodžių formos išmetamos, tuomet lieka tik 18 381 žodžių. Taigi naudojant lemavimą galima beveik 5 kartus sumažinti požymių skaičių.



3.1 pav. Požymių skaičiaus priklausomybė nuo grupuojamų straipsnių skaičiaus, naudojant įvairius lemavimo tipus

Vidutiniškai kiekvienas naujienų straipsnis yra sudarytas iš 366 terminų arba 247 nepasikartojančių požymių. Vidutinis kiekvieno termino ilgis yra 6,51 simboliai, su 3 simbolių standartiniu nuokrypiu.

Lietuviškų naujienų straipsnių kategorijų išskyrimui buvo panaudota informacija, esanti kiekvieno straipsnio URL adrese, tarp domeno pavadinimo ir straipsnio identifikacinio numerio. Iš viso buvo išskirtos 166 kategorijos, kurios pagal [4] naudotą metodiką sutrauktos į 12 kategorijų. Gautas toks straipsnių pasiskirstymas pagal kategorijas:

- Lietuva – 20162;
- Pasaulis – 21052;
- Kriminalai – 7502;
- Verslas – 7280;
- Automobiliai – 1557;
- Sportas – 5913;
- Mokslas – 1919;
- Nuomonės – 2553;
- Pramogos – 769;
- Gyvenimas – 944;
- Kultūra – 3478;

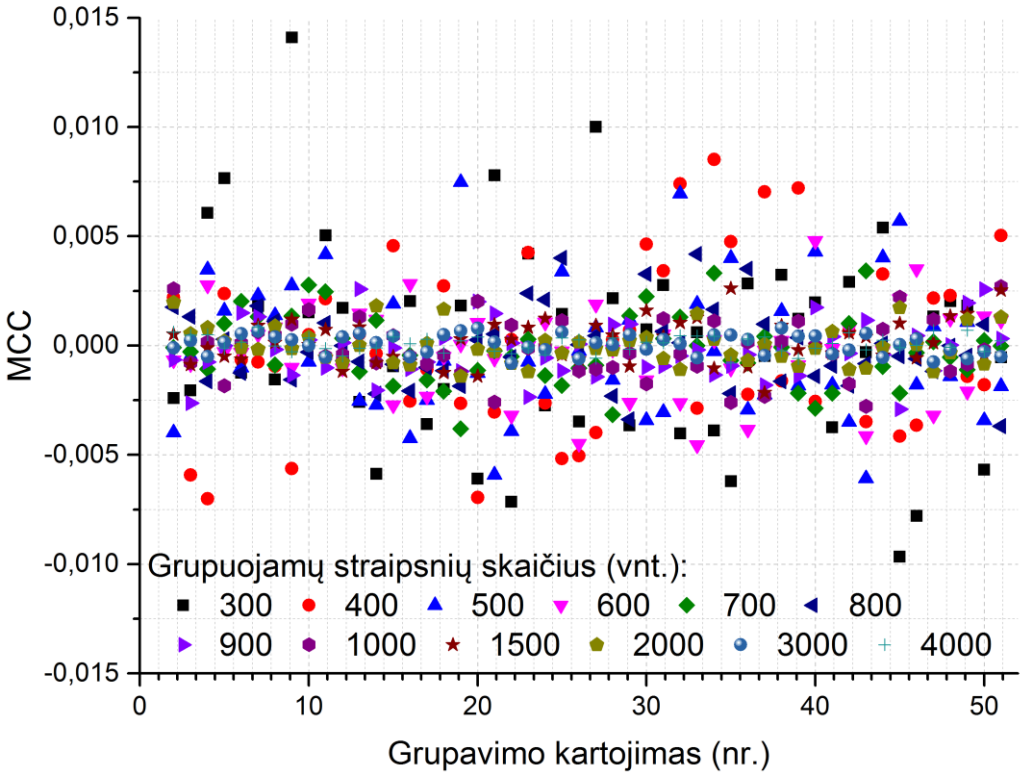
- Kita – 9664.

Kaip matyti, straipsnių pasiskirstymas pagal kategorijas yra ne vienodas. Aiškiai išsiskiria kategorijos *Lietuva* bei *Pasaulis*, kurios kartu sudaro net apie 49 % visų turimų straipsnių. Dėl šios priežasties nutarta eksperimentuose naudoti ne natūralų, o dirbtinį kategorijų pasiskirstymą, kiekvienai kategorijai skiriant po tiek pat straipsnių. Mažiausios kategorijos, t. y. *Pramogos* ir *Gyvenimas* buvo išmestos ir nenaudojamos, nes jose yra mažiau nei 1500 straipsnių. Taigi eksperimentuose naudotas normalizuotas 10 kategorijų straipsnių pasiskirstymas.

3.3. Realizacijos išbandymas

Šiame poskyryje pateikiamas sukurtos sistemos darbo įvertinimas jos našumo ir įverčio tikslumo atžvilgiu.

Siekiant patikrinti, ar pagrindiniai grupavimo bei įverčių skaičiavimo algoritmai veikia tinkamai, buvo sukurtas atsitiktinio skirstymo grupėmis metodas, pakeičiantis naudojamą *k-means*. Tokiu būdu buvo atlikti grupavimo eksperimentai grupuojant įvairius naujienų straipsnių kiekius. Kaip matyti (žr. 3.2 pav.), rezultatų pasiskirstymas, kaip ir tikėtasi, gautas apie 0. Tai rodo, kad sistemoje nėra pašalinių veiksnių, kurie veiktų grupavimo rezultatus.



3.2 pav. MCC įverčio vertės atliekant grupavimą atsitiktiniu būdu

Našumo eksperimentų rezultatai atskleidė skirtingą įvairių vektorizavimo metodų darbo laiką (žr. 3.2 lentelė). Trumpiausiai vektorizavimas yra vykdomas TF-IDF metodu – vidutiniškai 0,3884 s. Tai rodo šiuo metodo paprastumą. Iš kitos pusės, TF-IDF gaunami vektoriai yra dideli ir juos reikia ilgai grupuoti – vidutiniškai net virš 30 s. Tuo tarpu kitų metodų gautos reprezentacijos sugrupuojamos

greičiau nei per 1 s. Ilgiausias naujienų straipsnių pirminio apdorojimo ir vektorizavimo laikas gaunamas autoenkoderiui – net 2818 s. Kartojant grupavimą 50 kartų tai reikštų, kad vienas autoenkoderio eksperimentas užtruktų virš 39 val. Šiame darbe kiekvienam vektorizavimo metodui atlikta daug eksperimentų, todėl nuspręsta turimos autoenkoderio realizacijos dėl didelių laiko sąnaudų atsisakyti ir jo netirti.

3.2 lentelė. Įvairių vektorizavimo metodų darbo laikas, naudojant Intel® Core™2 Duo P8600 2,40GHz procesoriaus bei 4 GB darbinės atminties skaičiavimo mašiną

	Vidutinis laikas (s)		
	Apdorojimo ir vektorizavimo	**Įverčių skaičiavimo**	**Grupavimo**
TF-IDF	0,3884	0,0025	30,4482
doc2vec_pilnas	170,2939	0,0024	0,6694
doc2vec_dalis	15,5481	0,0026	0,5094
autoenkoderis	2818,3548	0,0025	0,5501

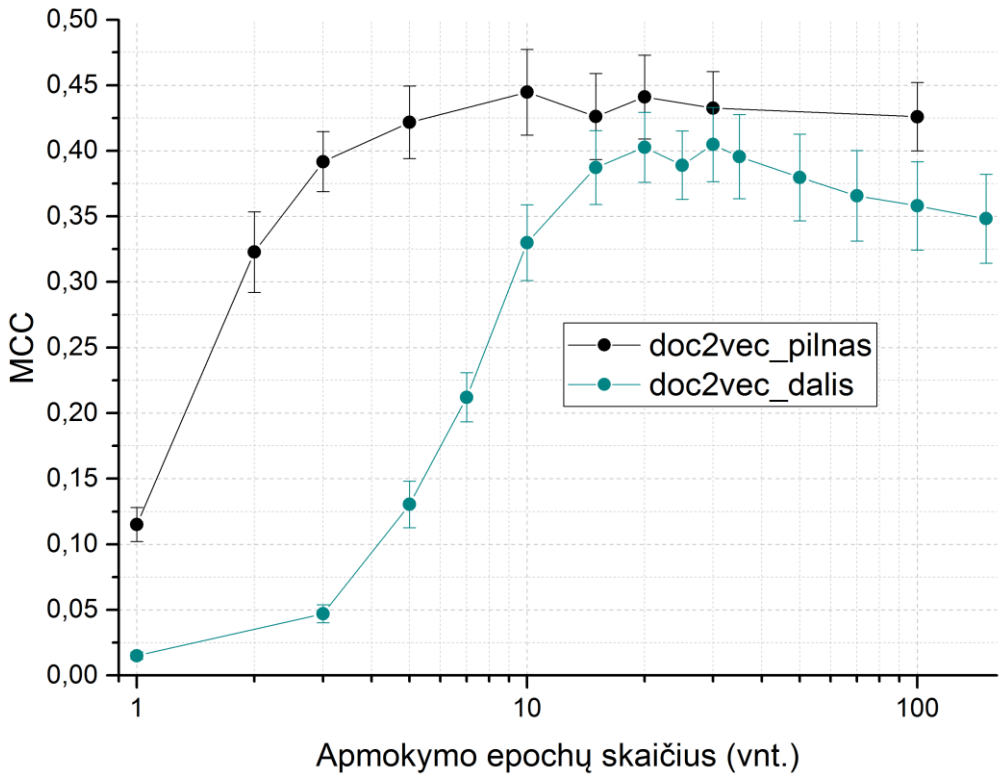
Taigi atlikus realizacijos bandymą buvo patvirtintas korektiškas sistemos veikimas. Dėl labai lėto veikimo nuspręsta atsisakyti eksperimentų su autoenkoderio vektorizavimo modeliu.

3.4. *Doc2vec* vektorizavimo metodų optimizavimas

Kaip jau minėta 1.6.2 skyrelyje, *doc2vec* vektorizavimas turi daug parametrų, kurie turėtų būti eksperimentiškai parinkti. Siekiant rasti optimalias vertes, šiame poskyryje bus aprašytas dviejų pagrindinių parametrų keitimas: epochų skaičiaus ir vektoriaus ilgio.

3.4.1. Apmokymo epochų skaičius

Apmokymo epochų skaičiaus eksperimento rezultatai pateikti 3.3 pav.



3.3 pav. MCC įverčio priklausomybė nuo apmokymo epochų skaičiaus ir *doc2vec* metodo tipo

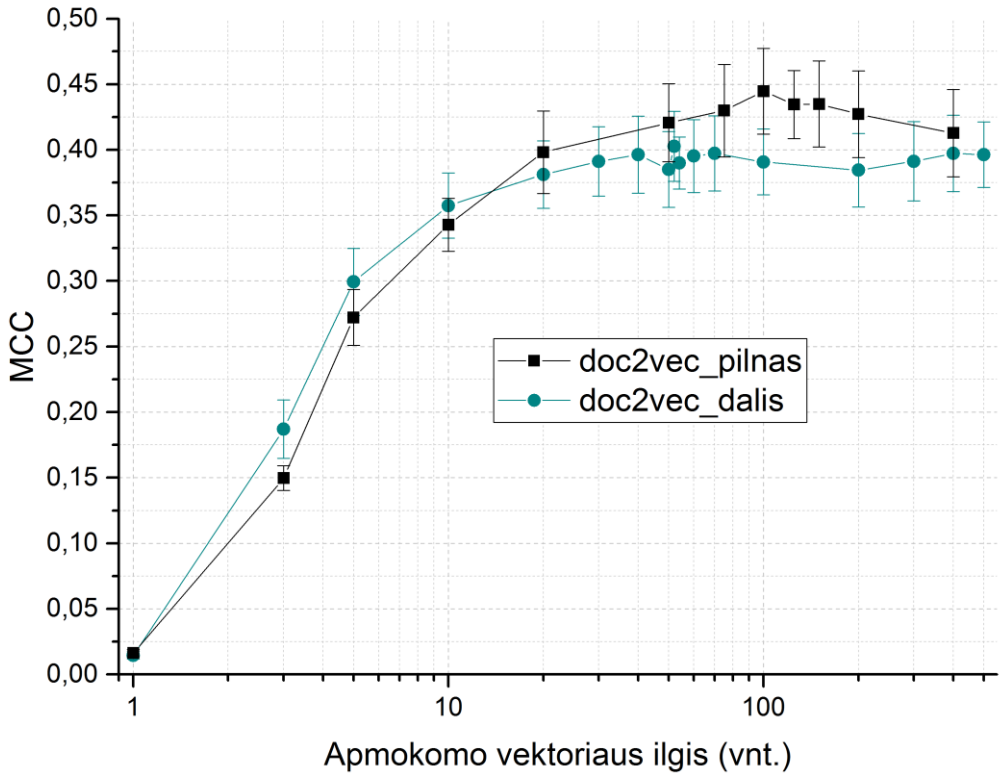
Kaip matyti, abiem *doc2vec* algoritmo apmokymo atvejams egzistuoja tam tikri epochų skaičiaus regionai, atitinkantys didžiausius grupavimo rezultatus. Apmokant modelį su pilnais duomenimis, maksimalus grupavimo rezultatas gaunamas esant 10 apmokymo epochų – vidutinė MCC vertė čia siekia net 0,448. Kitu, *doc2vec_dalis* atveju, pikai pasiekiami ties 20 ir ties 30 epochų, kur vidutinės MCC vertės yra atitinkamai 0,403 ir 0,405. Šios skiriasi labai nežymiai, tačiau pirmajai reikia tik dviejų trečdalių antrosios skaičiavimo laiko, todėl gali būti pasirinkta kaip optimali. Palyginus abiejų apmokymo tipų optimaliausių epochų skaičiaus algoritmų rezultatus (0,448 ir 0,403) gauta, kad geresnis algoritmas yra tas, kuris naudoja pilną duomenų rinkinį – *doc2vec_pilnas*.

Gauta grupavimo rezultatų ir epochų skaičiaus abiem apmokymo atvejams priklausomybė atskleidė, kad *doc2vec_pilnas* algoritmas optimaliai apmokomas per dvigubai mažesnį epochų skaičių. Nors su pilnu straipsnių rinkiniu gaunami tik nedaug geresni optimalūs rezultatai (0,448 > 0,403), esant mažesniam epochų skaičiui stebimas dar didesnis skirtumas. Pilnu straipsnių rinkiniu apmokytas *doc2vec* yra daugiau nei 3,23 karto rezultatyvesnis už mažesnio duomenų rinkinio modelį, apmokymą atlikus per 5 epochas. Tai rodo, kad didesnio duomenų rinkinio modelis turi didesnį potencialą, kurį pasiekia per mažesnį skaičių epochų.

Apmokymo epochų skaičiaus eksperimentu gauta, kad apmokant su pilnu duomenų rinkiniu gaunami geresni rezultatai. Taip pat galima daryti išvadą, kad optimalus epochų skaičius priklauso nuo apmokymui naudojamo duomenų rinkinio dydžio. Pasirinkti optimalūs epochų skaičiai apmokant su pilnu duomenų rinkiniu yra 10, o apmokant tik su grupuojamais straipsniais – 20.

3.4.2. Apmokomo vektoriaus ilgis

Doc2vec metodams vektorio ilgio eksperimento rezultatai parodyti 3.4 pav.



3.4 pav. MCC įverčio priklausomybė nuo apmokomo vektoriaus ilgio ir *doc2vec* metodo tipo

Geriausi rezultatai gaunami tuomet, kada naudojamos rastos optimalios apmokomo vektoriaus ilgio reikšmės. *doc2vec_pilnas* maksimalų vidutinį MCC įvertį (0,445) pasiekia esant 100 vienetų ilgio vektoriui, o *doc2vec_dalis* – 52 (MCC atitinkamai 0,403). Tai, kad paskutiniojo metodo optimalus vektoriaus ilgis yra beveik dukart mažesnis už pirmojo, gali būti paaiškinama tuo, kad apmokant su mažiau duomenų yra reikalingas trumpesnis vektorius šiai informacijai išsaugoti. Dėl to daugiau informacijos turintis didesnio optimalaus vektoriaus ilgio *doc2vec_pilnas* ir pasižymi geresniais grupavimo rezultatais, nei optimizuotas *doc2vec_dalis.*

3.5. Teksto apdorojimo eksperimentai

Šiame poskyryje aprašyti *doc2vec* ir TF-IDF modeliams išbandyti teksto apdorojimo žingsniai.

3.5.1. Nereikšminių žodžių išmetimas ir skaičių normalizavimas.

Šių, mažą dalį požymių išfiltruojančių eksperimentų rezultatai parodyti 3.3 lentelėje.

3.3 lentelė. MCC įverčio priklausomybė nuo įvairaus požymių filtravimo

	MCC			
	Nereikšminiai žodžiai		**Skaičiai**	
	Nefiltruota	**Išfiltruota**	**Nenormalizuoti**	**Normalizuoti**
TF-IDF	0,208±0,024	↑0,236±0,023	0,231±0,020	↑0,236±0,023
doc2vec_pilnas	0,445±0,033	↓0,425±0,028	0,445±0,033	↓0,439±0,029
doc2vec_dalis	0,385±0,030	↑0,403±0,027	0,383±0,024	↑0,403±0,027

Išskyrus geriausią *doc2vec_pilnas* naujienų straipsnių reprezentaciją, nereikšminių žodelių išmetimas pagerina grupavimo rezultatus. TF-IDF metodui jie pagerėja nuo vidutinės MCC vertės 0,208 iki 0,236, o *doc2vec_dalis* – nuo 0,385 iki 0,403. Taigi, kaip matyti, nereikšminių žodžių išmetimas labiausiai pagerina TF-IDF dokumentų vektorizavimo metodą.

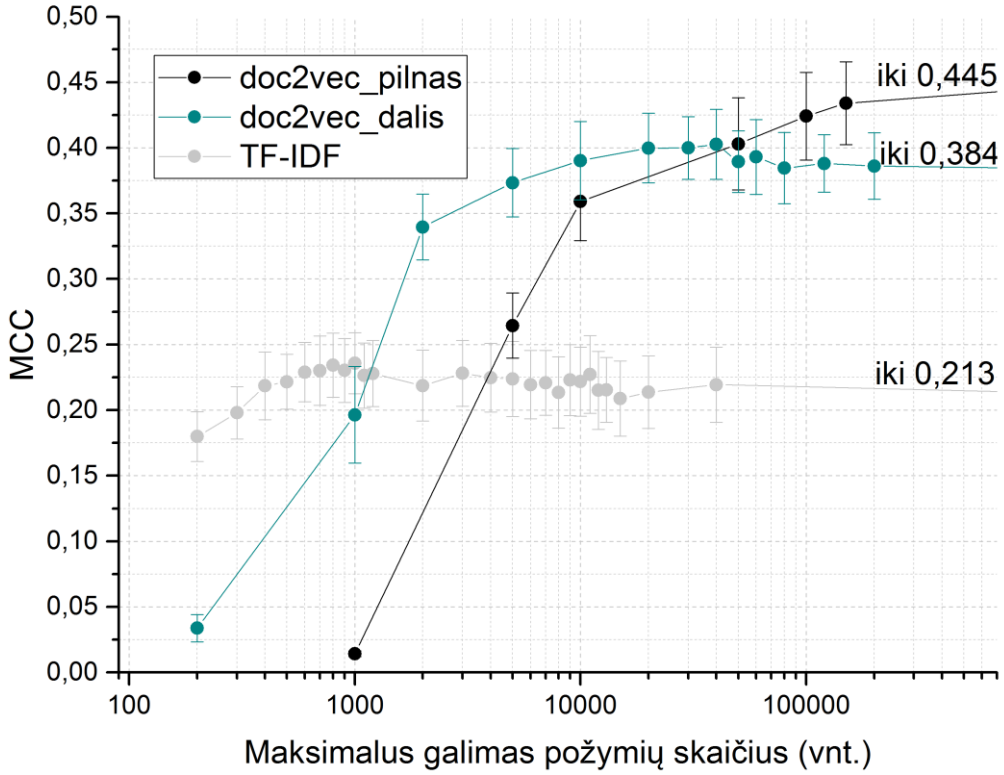
doc2vec_pilnas metodui nereikšminių žodelių išmetimas yra žalingas. Šiuo atveju grupavimo vidutinis MCC įvertis yra pabloginamas nuo 0,445 iki 0,452. Tai gali būti dėlto, kad šis dirbtinių neuroninių tinklų modelis sugebėjo išmokti minėtų nereikšminių žodelių įtaką ir ją statistiškai įvertinti. Tuo tarpu filtravimas yra atliekamas pagal žmogaus atrinktą nereikšmingų žodelių sąrašą, kuriame gali vis dėlto likti modeliui reikalingos informacijos. Taigi galima teigti, kad pakankamai stipriam modeliui yra naudingiau pačiam išmokti, kas turi būti išfiltruojama.

Skaičių normalizavimo rezultatai rodo analogiškus rezultatus. Tiek TF-IDF, tiek *doc2vec_dalis* grupavimo rezultatai yra nežymiai pagerinami. Tuo tarpu *doc2vec_pilnas* skaičių normalizavimas rezultatus šiek tiek pablogina.

Atliktų nereikšminių žodžių filtravimo ir skaičių normalizavimo eksperimentų rezultatams gaunamos per didelės standartinio nuokrypio reikšmės, todėl iškeltą hipotezę apie gerai apmokytam *doc2vec_pilnas* nereikalingą filtravimą turėtų patvirtinti kiti eksperimentai.

3.5.2. Maksimalus požymių skaičius

Maksimalaus žodyno ilgio eksperimento rezultatai įvairiems teksto vektorizavimo metodams pavaizduoti 3.5 pav. Vertėtų atkreipti dėmesį, kad minėtame grafike ant besitęsiančių linijų į požymių skaičiaus begalybę nurodytas grupavimo MCC rezultatas, atitinkantis požymių neribojimą, pvz. „iki 0,445" (žr. 3.5 pav.).

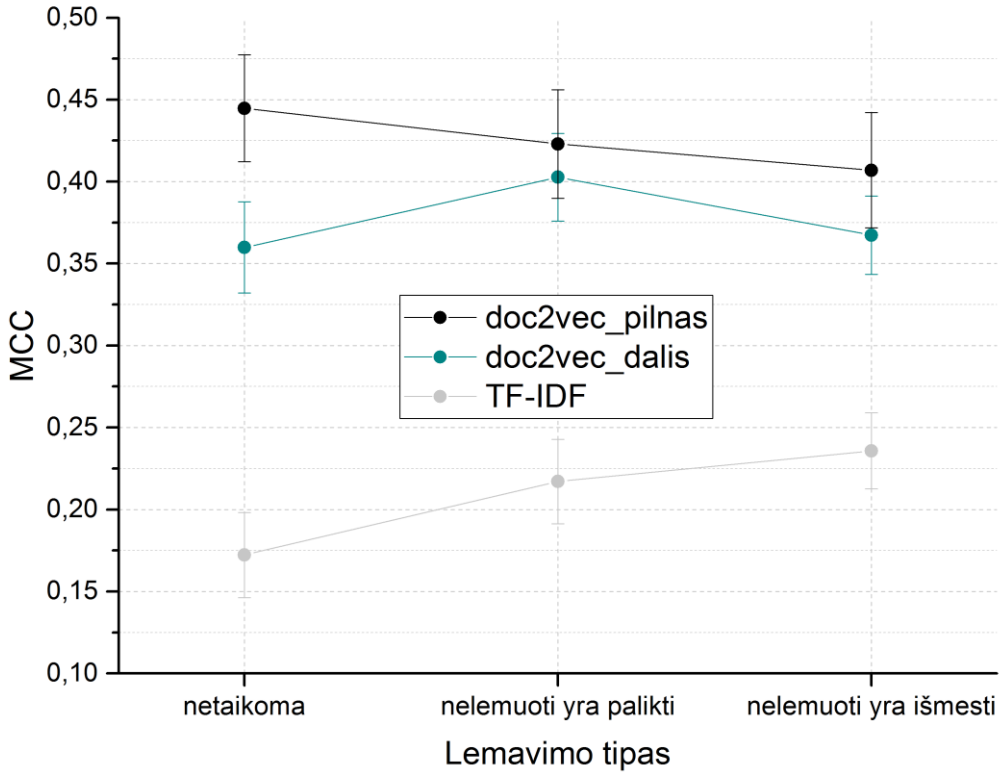


3.5 pav. MCC įverčio priklausomybė nuo maksimalaus galimo požymių skaičiaus ir vektorizavimo metodo

Kiekvienam vektorizavimo metodui atliktas eksperimentas atskleidė skirtingas optimalias maksimalaus požymių skaičiaus vertes. Žemiausius grupavimo rezultatus gaunančiam TF-IDF metodui tai yra 1000 požymių, o *doc2vec_dalis* – 40000. Tuo tarpu *doc2vec_pilnas* gauta, kad optimaliausia yra visai neriboti maksimalaus požymių skaičiaus. Šis rezultatas dera su jau ankstesniame eksperimente išsakytais samprotavimais, kad gerai apmokytas *doc2vec_pilnas* sugeba pats įvertinti filtruotinus požymius ir tokiu būdu nepraranda apmokymo duomenų.

3.5.3. Lemavimas

Visiems vektorizavimo metodams eksperimento metu nustatytas lemavimo poveikis parodytas 3.6 pav.



3.6 pav. MCC įverčio priklausomybė nuo lemavimo tipo ir vektorizavimo metodo

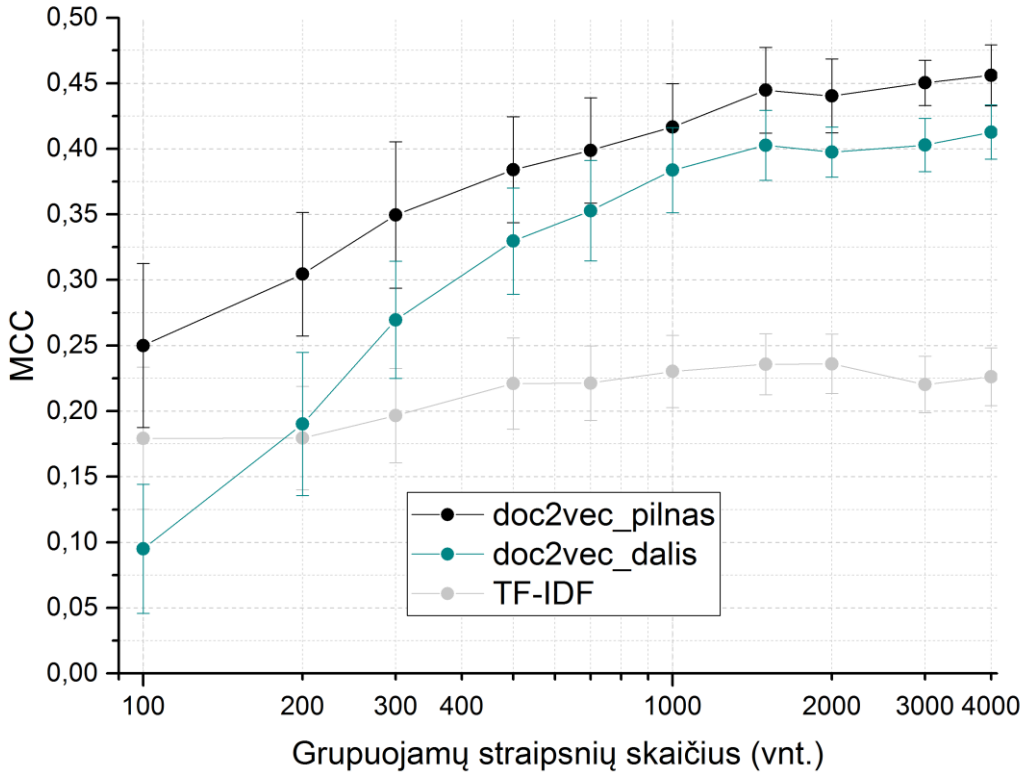
Šis eksperimentas parodė, kad didžiausią lemavimo naudą patiria TF-IDF vektorizavimo metodas. Lemavus turimus duomenų bazėje ir palikus nelemuotus žodžius, grupavimo įverčių MCC vidurkis pagerėja nuo 0,172 iki 0,217. Jei nežinomi žodžiai yra išfiltruojami, tuomet TF-IDF metodui grupavimo efektyvumas pagerėja dar iki 0,236. Taigi iš viso TF-IDF MCC įvertis padidėja net per 0,064.

Kitokia situacija stebima abiem *doc2vec* metodams. *doc2vec_dalis* lemavimas gali pagerinti MCC įvertį nuo 0,360 iki 0,403, jei nelemuoti požymiai yra paliekami. Tuo tarpu šių išmetimas duoda mažesnius rezultatus – 0,367. Taigi, nors lemavimas turi neabejotiną naudą, tačiau didesnė žala MCC įverčiui, net – 0,036, gali būti padaryta atliekant papildomą filtravimą pagal duomenų bazėje turimą žodyną. Tuo tarpu *doc2vec_pilnas*, kaip ir pastebėta ankstesniuose skyreliuose, geriausiai veikia neatlikus iš vis jokio požymių filtravimo.

Apibendrinant šį poskyrį galima teigti, kad papildomas teksto apdorojimas yra naudingas tik prastesniems dokumentų vektorizavimo modeliams. TF-IDF metodui stebimas didelis grupavimo įverčio pagerėjimas dėl nereikšminių žodelių išmetimo ir lemavimo, išmetant nelemuotus požymius. *doc2vec_dalis* modelis turi mažesnį minėtų metodų sąlygojamą MCC pagerėjimą. Tuo tarpu apmokytas su pilnu duomenų rinkiniu *doc2vec_pilnas* geriausiai veikia tuomet, jei tiek lemavimas, tiek maksimalaus žodyno ilgio ir nereikšminių žodžių filtravimas yra visai neatliekamas.

3.6. Straipsnių kiekio eksperimentas

Geriausiai tirtus vektorizavimo metodus galima palyginti straipsnių kiekio eksperimento rezultatuose (žr. 3.7 pav.). Grupuojamų straipsnių skaičiui kintant nuo 200 iki 4000, geriausius grupavimo rezultatų įverčius rodo *doc2vec_pilnas* dokumentų vektorizavimo metodas. Ties 1500 grupuojamų straipsnių MCC įvertis siekia net 0,445. Mažesnės, tačiau vis tiek aukštos MCC vertės gaunamos ir *doc2vec_dalis*. 1500 naujienų straipsnių čia MCC vidutinė vertė siekia 0,403. Tuo tarpu patys blogiausi rezultatai gaunami klasikinei TF-IDF dokumentų reprezentacijai. Grupuojant 1500 dokumentų gaunama vos 0,24 MCC vertė.



3.7 pav. MCC įverčio priklausomybė nuo grupuojamų straipsnių skaičiaus ir vektorizavimo metodo

Grupuojant mažiau nei 1500 naujienų straipsnių, visiems dokumentų vektorizavimo modeliams stebimas grupavimo rezultatų prastėjimas. Labiausiai prastėja abu *doc2vec* modeliai. 1000 straipsnių *doc2vec_pilnas* vidutinė MCC vertė jau nukritusi iki 0,417, o ties 300 – iki 0,350. *doc2vec_dalis* ties 100 grupuojamų straipsnių skaičiumi vidutinė MCC vertė (0,095) nukrenta net žemiau nei TF-IDF (0,179). Jei šį staigų kritimą galima paaiškinti tuo, kad smarkiai sumažėja duomenų, naudojamų apmokyti algoritmą, kiekis, tai *doc2vec_pilnas* MCC vertės (0,250) kritimas tuo pačiu negali būti paaiškinamas - stebima bendra grupavimo rezultatų mažėjimo tendencija. Taigi eksperimentas parodė, kad grupuojant mažai dokumentų, tik *doc2vec_pilnas* naujienų straipsnių reprezentacijos grupavimo rezultatai išlieka geresnis už TF-IDF.

3.7. Dviejų savaitės duomenų kiekio eksperimentas

Šiame eksperimente išbandomi ankstesniuose poskyriuose optimizuoti *doc2vec* bei TF-IDF naujienų straipsnių vektorizavimo algoritmai, nenormalizuojant naujienų straipsnių pagal kategorijas pasiskirstymo. Bus naudojami visi duomenų bazėje turimi naujienų straipsniai, išleisti pradedant 2017 metų balandžio 24 ir baigiant gegužės 7 diena. Šia imtį iš viso sudaro 2020 naujienų straipsnių.

Kiekvienas algoritmas buvo išbandytas 200 kartų ir atrinkti geriausi grupavimo rezultatai pagal MCC įvertį parodyti 3.4 lentelėje, o gautos grupės išsamiai apibūdintos 3 – 5 prieduose.

3.4 lentelė. Geriausi grupavimo įverčiai įvairiems vektorizavimo metodams

Vektorizavimo metodas	**Geriausio grupavimo įverčiai**	
	MCC	**F1**
doc2vec_pilnas	0,324;	0,407
doc2vec_dalis	0,303;	0,392
TF-IDF	0,203;	0,333

Eksperimentai parodė, kad visi vektorizavimo metodai padeda išskirti šias aiškias straipsnių grupes:

- naujienos apie orus;
- sportas;
- Šiaurės Korėjos branduolinis ginklavimasis;
- pasaulyje gyvybių nusinešę išpuoliai, konfliktai.

Doc2vec metodai, kitaip nei TF-IDF, išskyrė dar ir Europos rinkimų grupę (Prancūzijos rinkimai, *Brexit*), o TF-IDF, kitaip nei pirmieji, atpažino mokslo ir švietimo reformų, bei Lietuvos politikų apkaltos temas. Taip pat visi metodai išskyrė ir kultūros temą, tačiau TF-IDF ši buvo stipriau užteršta kitų kategorijų straipsniais.

Stebėtina, kad visi metodai padėjo išskirti orų kategoriją. Ši yra labai maža ir sudaro mažiau nei 2 % visų straipsnių. Be to, suskirstyme į kategorijas ši kategorija nėra išskirta, o įeina į stambesnę „Lietuvos naujienos“ kategoriją. Tai, kad visi metodai išskyrė šią kategoriją, rodo, kad jie vektorizavo gan gerą naujienų straipsnių reprezentaciją.

TF-IDF, kitaip nei *doc2vec*, sudarė dvi labai dideles, siekiančias 419 ir 595 straipsnių grupes. Pirmąją galima iš dalies priskirti kultūrai, o antrąją – politikų kuriamiems įstatymams, projektams. Vis dėl to, jos sudarytos iš daugelio kategorijų ir dėl tokio savo dydžio yra gana plačios. Tuo tarpu *doc2vec_pilnas* didžiausia grupė sudaryta iš 302 naujienų straipsnių, tačiau čia net 85 % jų priklauso „Pasaulio naujienos“ kategorijai. Panašiai ir *doc2vec_dalis* – didžiausia 299 straipsnių grupė yra sudaryta net 79 % iš „Pasaulio naujienos“ kategorijai priklausančių naujienų straipsnių. Taigi TF-IDF išsiskiria ne tik konkrečiomis mokslo ir švietimo reformų, bei Lietuvos politikų apkaltos temomis, tačiau tuo pačiu metu turi ir kelias gana dideles abstrakčias grupės.

Apibendrinant dviejų savaičių eksperimentą galima teigti, kad *doc2vec_pilnas* MCC įvertis buvo pats geriausias. Žvelgiant į konkrečius gautų grupių bruožus pastebėta, kad visi metodai sėkmingai išskiria karo, pasaulinių konfliktų, sporto bei labai mažą orų temas. Likusių grupių išskyrimas yra įvairus ir pagal jų pagrindinius požymius sunku išskirti, kuris metodas yra pranašesnis.

3.8. Įžvalgos ateičiai

Šiame darbe atliekant eksperimentus kilo įvairių idėjų, ką dar būtų galima išbandyti, tačiau ne viską buvo galima įgyvendinti dėl ribotų projekto resursų. Jei panašus darbas būtų atliekamas ir ateityje, siūlyčiau išbandyti šias galimybes:

- vietoje žodžių požymių naudoti simbolių požymius;
- apmokant *doc2vec* kaip įvestį paduoti ne tik dokumento identifikatorių, bet ir kitą informaciją, pvz. struktūrinę (tokį funkcionalumą turi *gensim* biblioteka);
- kaip elementinį požymį naudoti įvairias požymių (žodžių, simbolių) kombinacijas (angl. *n-grams*);
- apjungti skirtingais vektorizavimo metodais gautus vektorius;
- apmokyti *doc2vec* su daug didesniu duomenų kiekiu;
- naudoti autoenkoderį, prieš tai optimizavus ir pagreitinus jo veikimą.

Šiame darbe naudotas nereikšminių žodžių sąrašas, kaip minėta 3.2.2 skyrelyje, sudaro vos 0,07 % visų požymių. Tai yra labai mažas skaičius išfiltruojamų požymių ir toks filtravimas turi menką reikšmę galutiniams rezultatams. Daug naudingiau būtų naudoti statistiškai sudarytą nereikšminių žodžių sąrašą. Jį būtų galima gauti naudojant TF-IDF svorius. Tuomet nereikšminiai žodžiai apimtų ir nereikšminius skaičius bei kitus požymius. Tam tikra dalis mažiausio TF-IDF svorio požymių daug geriau atspindėtų nereikšminius žodžius.

Autorius tikisi, kad šiame darbe atlikti eksperimentai pasitarnaus ir kituose lietuviškų naujienų grupavimo darbuose ateityje ir tokiu būdu prisidės prie reprezentatyvesnių žinių išgavimo iš lietuviškų tekstų.

IŠVADOS

1. Nustatyta, kad geriausias *doc2vec* vektorizavimas yra gaunamas tuomet, jei šis modelis yra apmokomas naudojant pilno straipsnių rinkinio duomenis (šiame darbe vadinamas *doc2vec_pilnas*). Rastas optimalus 10 apmokymo epochų skaičius bei optimalūs 100 vienetų ilgio apmokomų dokumentų vektorių ilgiai. Taip pat gauta, kad jei *doc2vec* apmokomas su mažesniu kiekiu duomenų, reikalingas ilgesnis epochų skaičius ir užtenka mažesnio ilgio vektoriaus.
2. Palyginus TF-IDF ir *doc2vec* metodams taikomą pirminį teksto apdorojimą nustatyta, kad jis reikalingiausias prasčiausius rezultatus atitinkančiam TF-IDF bei su mažiau duomenų apmokytam *doc2vec_dalis* modeliui. Tuo tarpu *doc2vec_pilnas* geresnius rezultatus duoda tuomet, kai nei lemavimas, nei nereikšminių žodžių išmetimas, nei skaičių normalizavimas ar maksimalaus žodyno filtrai nėra taikomi.
3. TF-IDF ir *doc2vec* dokumentų vektorizavimų palyginimas atskleidė, kad *doc2vec* gaunami rezultatai pagal MCC yra daug geresni nei su TF-IDF. Dviejų savaičių naujienų straipsnių grupavimo rezultatų geriausi įverčiai gauti atitinkamai 0,324 ir 0,203. Pastebėta, kad mažėjant grupuojamų straipsnių skaičiui, TF-IDF duoda stabilesnius, taip greitai nemažėjančius rezultatus, nei *doc2vec* metodai. Iš paskutiniųjų tik su daugiau duomenų apmokytas *doc2vec_pilnas* pagal MCC įvertį išlieka geresnis grupuojant mažiau nei 200 straipsnių. Nepaisant MCC įverčio skirtumų, tiek TF-IDF, tiek *doc2vec* dokumentų reprezentacijomis gauti grupavimai sugebėjo aiškiai išskirti orų, sporto, Šiaurės Korėjos branduolinio ginklavimosi, bei pasaulyje vykusių išpuolių ir konfliktų temas.

LITERATŪROS SĄRAŠAS

1. AGGARWAL, Charu C. ir Cheng Xiang ZHAI. *Mining text data*. 2013. 1–522 p. ISBN 9781461432234.
2. LIU, Linqing ir kt. Generative adversarial network for abstractive text summarization. *Thirty-Second AAAI Conference on Artificial Intelligence* .2018. .
3. LIU, Gang ir Jiabao GUO. Bidirectional LSTM with attention mechanism and convolutional layer for text classification. *Neurocomputing* .2019. ISSN 18728286.
4. PRANCKAITIS, Vilius. *Lietuviškų naujienų grupavimo algoritmų tyrimas*. 2017. .
5. HINTON, G.E. ir R.R. SALAKHUTDINOV. Reducing the dimensionality of data with neural networks. *Science* [interaktyvus]. 2006. Vol. 313, no. 5786, p. 504–507. [žiūrėta 2017-12-29]. . ISSN 00368075.
6. LIU, Xiaohui ir kt. Keyword extraction for web news documents based on LM-BP neural network. *Proceedings of the 2015 27th Chinese Control and Decision Conference, CCDC 2015* [interaktyvus]. 2015. p. 2525–2531. [žiūrėta 2017-12-27]. Prieiga per: http://ieeexplore.ieee.org/document/7162346/.
7. AKER, Ahmet ir kt. Automatic label generation for news comment clusters. *Proceedings of the 9th International Natural Language Generation conference* [interaktyvus]. Stroudsburg, PA, USA: Association for Computational Linguistics, 2016. p. 61–69. [žiūrėta 2017-12-01]. Prieiga per: http://aclweb.org/anthology/W16-6610.
8. GUERIN, NS *Cross-Language News Article Clustering* [interaktyvus]. Harvard University, 2017. [žiūrėta 2017-12-01]. Prieiga per: http://www.plaintexttransmissions.org/static/pdf/thesis.pdf.
9. GABRILOVICH, Evgeniy ir Shaul MARKOVITCH. Computing semantic relatedness using wikipedia-based explicit semantic analysis. *IJCAI International Joint Conference on Artificial Intelligence* [interaktyvus]. Morgan Kaufmann Publishers Inc., 2007. p. 1606–1611. [žiūrėta 2017-12-01]. Prieiga per: https://dl.acm.org/citation.cfm?id=1625535.
10. KEBELYTĖ, Ernesta *Lietuviško automatinio naujienų agregatoriaus prototipas: magistro darbas*. Kaunas: Kauno technologijos universitetas. Prieiga per eLABa – nacionalinė Lietuvos akademinė elektroninė biblioteka, 2015. .
11. TANG, Bin ir kt. Comparing Dimension Reduction Techniques for Document Clustering. *Conference of the Canadian Society for Computational Studies of Intelligence* [interaktyvus]. Berlin, Heidelberg: Springer, 2010. p. 292–296. [žiūrėta 2017-12-28]. Prieiga per: http://link.springer.com/10.1007/11424918_30.
12. AGGARWAL, Charu ir Chandan REDDY. *Data Clustering: Algorithms and Applications* [interaktyvus]. Chapman & Hall/CRC, 2013. 652 p. ISBN 1466558210 9781466558212.
13. MACKUTE-VARONECKIENE, Aušra ir Tomas KRILAVIČIUS. Empirical Study on

Unsupervised Feature Selection for Document Clustering. *Frontiers in Artificial Intelligence and Applications*. 2014. Prieiga per: doi: http://dx.doi.org/10.3233/978-1-61499-442-8-107.

14. MARCUS LÖNNBERG ir LOVE YREGÅRD. *Large scale news article clustering* [interaktyvus]. Chalmers University of Technology, 2013. [žiūrėta 2017-12-01]. Prieiga per: http://publications.lib.chalmers.se/records/fulltext/179841/179841.pdf.
15. KIM, Minyoung Simultaneous Learning of Sentence Clustering and Class Prediction for Improved Document Classification. *The International Journal of Fuzzy Logic and Intelligent Systems* [interaktyvus]. 2017. Vol. 17, no. 1, p. 35–42. [žiūrėta 2017-12-01]. . ISSN 1598-2645.
16. PRANCKAITIS, Vilius ir Mantas LUKOŠEVIČIUS. Clustering of Lithuanian news articles. *CEUR Workshop Proceedings* . 2017.
17. WHITE, Lyndon ir kt. Sentence Representations and Beyond. *Neural Representations of Natural Language* [interaktyvus].Singapore: Springer Singapore, 2019. p. 93–114. ISBN 978-981-13-0062-2 Prieiga per: https://doi.org/10.1007/978-981-13-0062-2_5.
18. KIM, Minyoung. Robust Algorithms for Combining Multiple Term Weighting Vectors for Document Classification. *The International Journal of Fuzzy Logic and Intelligent Systems* [interaktyvus]. 2016. Vol. 16, no. 2, p. 81–86. [žiūrėta 2017-12-01]. ISSN 1598-2645.
19. LAN, Man ir kt. Supervised and traditional term weighting methods for automatic text categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence* [interaktyvus]. 2009. Vol. 31, no. 4, p. 721–735. [žiūrėta 2017-12-01]. . ISSN 01628828.
20. XU, Jiaming ir kt. Self-Taught Convolutional Neural Networks for Short Text Clustering. *Elsevier* [interaktyvus]. 2016. Vol. 88, p. 22–31. [žiūrėta 2017-12-01]. ISSN 18792782.
21. LECUN, Yann ir kt. Deep learning. *Nature* [interaktyvus]. 2015. Vol. 521, no. 7553, p. 436–444. [žiūrėta 2017-12-01]. ISSN 0028-0836.
22. LE, Quoc ir Tomas MIKOLOV. Distributed Representations of Sentences and Documents. *Proceedings of the 31st International Conference on Machine Learning* [interaktyvus]. Bejing, China: PMLR, 2014. p. 1188–1196. Prieiga per: http://proceedings.mlr.press/v32/le14.html.
23. MIKOLOV, Tomas ir kt. Efficient Estimation of Word Representations in Vector Space. [interaktyvus]. 2013. .
24. KAPOČIŪTĖ-DZIKIENĖ, Jurgita ir Robertas DAMAŠEVIČIUS. Intrinsic evaluation of lithuanian word embeddings using wordnet. *Advances in Intelligent Systems and Computing* .2019. Prieiga per: doi: http://dx.doi.org/10.1007/978-3-319-91189-2_39.
25. KAPOČIŪTĖ-DZIKIENĖ, Jurgita ir kt. Sentiment Analysis of Lithuanian Texts Using Traditional and Deep Learning Approaches. *Computers*. 2019.
26. TIAN, Kai ir kt. DeepCluster: A General Clustering Framework Based on Deep Learning BT - Machine Learning and Knowledge Discovery in Databases. CECI, Michelangelo ir kt.Sud. .Cham: Springer International Publishing, 2017. p. 809–825. .

27. WU, M. ir kt. Ensemble Clustering via Learning Representations from Auto-Encoder. *2018 15th International Conference on Service Systems and Service Management (ICSSSM)*. 2018. p. 1–6. Prieiga per: doi: http://dx.doi.org/10.1109/ICSSSM.2018.8464996.
28. WU, J. ir kt. K-Means-Based Consensus Clustering: A Unified View. *IEEE Transactions on Knowledge and Data Engineering*. 2015. Vol. 27, no. 1, p. 155–169. ISSN 1041-4347 VO - 27.
29. MIN, E. ir kt. A Survey of Clustering With Deep Learning: From the Perspective of Network Architecture. *IEEE Access* .2018. Vol. 6, p. 39501–39514. ISSN 2169-3536 VO - 6.
30. LI, Jiwei ir kt. A Hierarchical Neural Autoencoder for Paragraphs and Documents. 2015.
31. KALCHBRENNER, Nal ir kt. A Convolutional Neural Network for Modelling Sentences. 2014.
32. DENIL, Misha ir kt. Modelling, Visualising and Summarising Documents with a Single Convolutional Neural Network. 2014.
33. ONAN, Aytug ir kt. An improved ant algorithm with LDA-based representation for text document clustering. *Journal of Information Science*. 2017. Vol. 43, no. 2, p. 275–292. ISSN 17416485.
34. GIALAMPOUKIDIS, Ilias ir kt. A hybrid framework for news clustering based on the DBSCAN-Martingale and LDA. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* [interaktyvus]. Springer, Cham, 2016. p. 170–184. [žiūrėta 2017-12-27]. Prieiga per: http://link.springer.com/10.1007/978-3-319-41920-6_13.
35. WANG, Heng ir kt. An Improved Clustering Method for Detection System of Public Security Events Based on Genetic Algorithm and Semisupervised Learning. *Complexit*. 2017. Vol. 2017, p. 1–10.. ISSN 1076-2787.
36. AL-JADIR, Ibraheem ir kt. Text Document Clustering Using Memetic Feature Selection. *Proceedings of the 9th International Conference on Machine Learning and Computing* [interaktyvus]. ACM, 2017. p. 415–420. [žiūrėta 2017-12-01]. Prieiga per: http://dl.acm.org/citation.cfm?id=3056603.
37. FAN, Zhaoxin ir kt. A Text Clustering Approach of Chinese News Based on Neural Network Language Model. *International Journal of Parallel Programming*. 2016. Vol. 44, no. 1, p. 198–206. ISSN 08857458.
38. RANI, Usha ir Shashank SAHU Comparison of clustering techniques for measuring similarity in articles. *3rd IEEE International Conference on* [interaktyvus]. IEEE, 2017. p. 1–7. [žiūrėta 2017-12-01]. Prieiga per: http://ieeexplore.ieee.org/document/7977377/.
39. GARG, N. ir kt. Clustering Techniques for Text Mining: A Review. *International Journal of Engineering Research*. 2016. Vol. 5, no. 4, p. 241–243.
40. YI, Junkai ir kt. A Novel Text Clustering Approach Using Deep-Learning Vocabulary Network. *Mathematical Problems in Engineering*. 2017. Vol. 2017, p. 1–13. ISSN 1024-123X.
41. HUNG CHIM ir XIAOTIE DENG. Efficient Phrase-Based Document Similarity for Clustering.

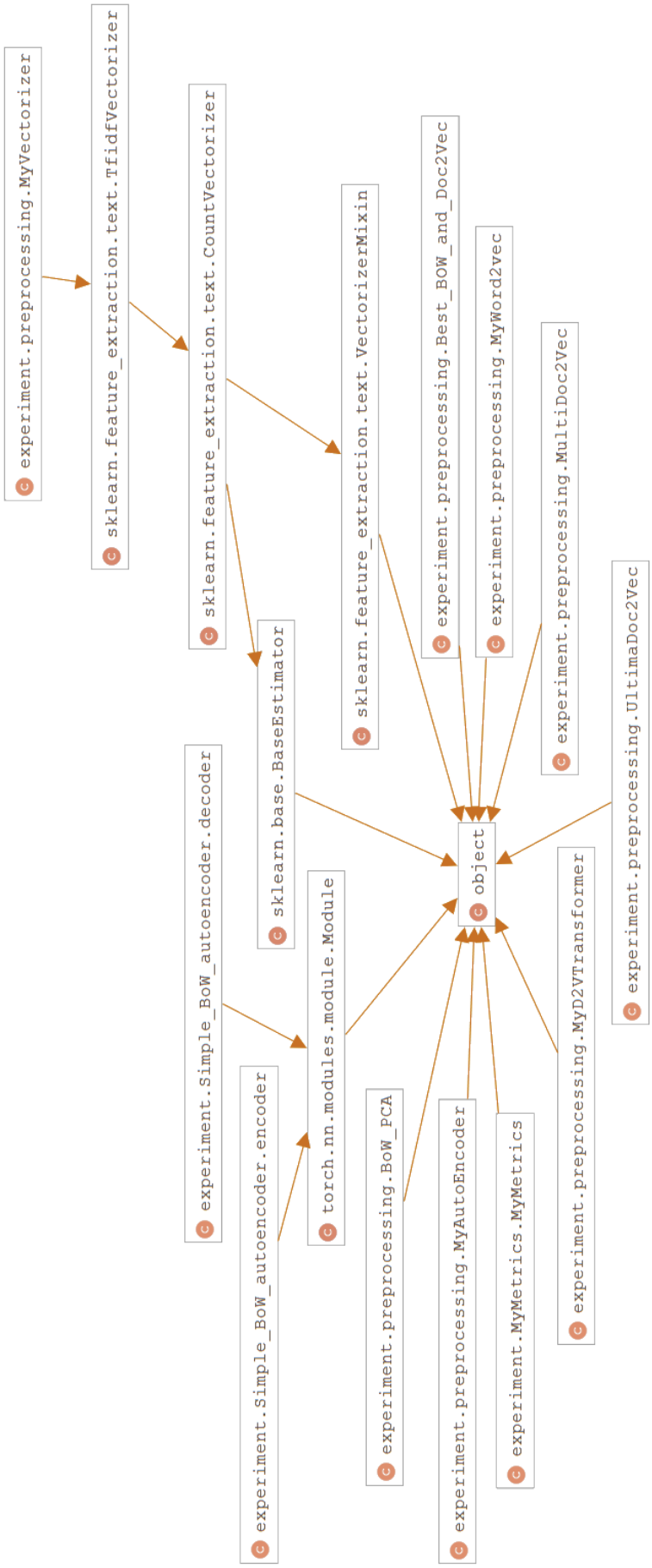
IEEE Transactions on Knowledge and Data Engineering. 2008. Vol. 20, no. 9, p. 1217–1229.

42. JOSHI, Aastha ir Rajneet KAUR. A Review: Comparative Study of Various Clustering Techniques in Data Mining. *International Journal of Advanced Research in Computer Science and Software Engineering*. 2013. Vol. 3, no. 3, p. 2277–128.
43. HONG, Xudong ir kt. Cross-lingual event-centered news clustering based on elements semantic correlations of different news. *Multimedia Tools and Applications*. 2017. Vol. 76, no. 23, p. 25129–25143. ISSN 15737721.
44. E. CHAZAN, Shlomo ir kt. *Deep Clustering Based on a Mixture of Autoencoders*. 2018. .
45. MEI, M. ir kt. Using Semantic Clustering And Autoencoders For Detecting Novelty In Corpora Of Short Texts. *2018 International Joint Conference on Neural Networks (IJCNN)* .2018. p. 1–8. Prieiga per: doi: http://dx.doi.org/10.1109/IJCNN.2018.8489431.
46. LIU, Xin ir kt. Feature Word Vector Based on Short Text Clustering. *Computer Science and Technology* [interaktyvus]. WORLD SCIENTIFIC, 2016. p. 533–545. [žiūrėta 2017-12-01]. Prieiga per: http://www.worldscientific.com/doi/abs/10.1142/9789813146426_0061.
47. CIGANAITĖ, Greta ir kt. Text documents clustering. *Informacinės technologijos. XIX tarpuniversitetinė magistrantų ir doktorantų konferencija" Informacinė visuomenė ir universitetinės studijos"(IVUS 2014): konferencijos pranešimų medžiaga, 2014, p. 90-93* .2014. .
48. RASHMI, Ms Introduction to Information Retrieval Systems. *International Journal on Recent and Innovation Trends in Computing and Communication*. 2015. Vol. 3, no. 4, p. 2051–2054.
49. PRATAMA, M.F.E. ir kt. Digital News Graph Clustering using Chinese Whispers Algorithm. *Journal of Physics: Conference Series* [interaktyvus]. IOP Publishing, 2017. p. 012062. [žiūrėta 2017-12-01]. Prieiga per: http://stacks.iop.org/1742-6596/801/i=1/a=012062?key=crossref.fa600955aa1e527bac62829cbde82350.
50. SAKLANI, NS ir S. SHARMA. Extracting News From The Web Pages By Using Concept Of Clustering With Neural Genetic Approach. 2016.
51. WEN-YEN CHEN ir kt. Parallel Spectral Clustering in Distributed Systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2010. Vol. 33, no. 3, p. 568–586. ISSN 0162-8828.
52. KUMARSHRIVASTAVA, Shailendra ir kt. Text Document Clustering based on Phrase Similarity using Affinity Propagation. *International Journal of Computer Applications*. 2013. Vol. 61, no. 18, p. 38–44. ISSN 09758887.
53. NAVICKAS, Šarūnas. Metodika ir prototipas automatiniam kino filmų žiūrovų įverčio skaičiavimui. Disertacija. Kaunas: Vytautas Magnus University, 2016. .
54. WANG, Haohan ir Bhiksha RAJ On the Origin of Deep Learning.2017.
55. Google News. [interaktyvus]. [žiūrėta 2019-01-22]. Prieiga per: https://news.google.com/?hl=en-US&gl=US&ceid=US:en.

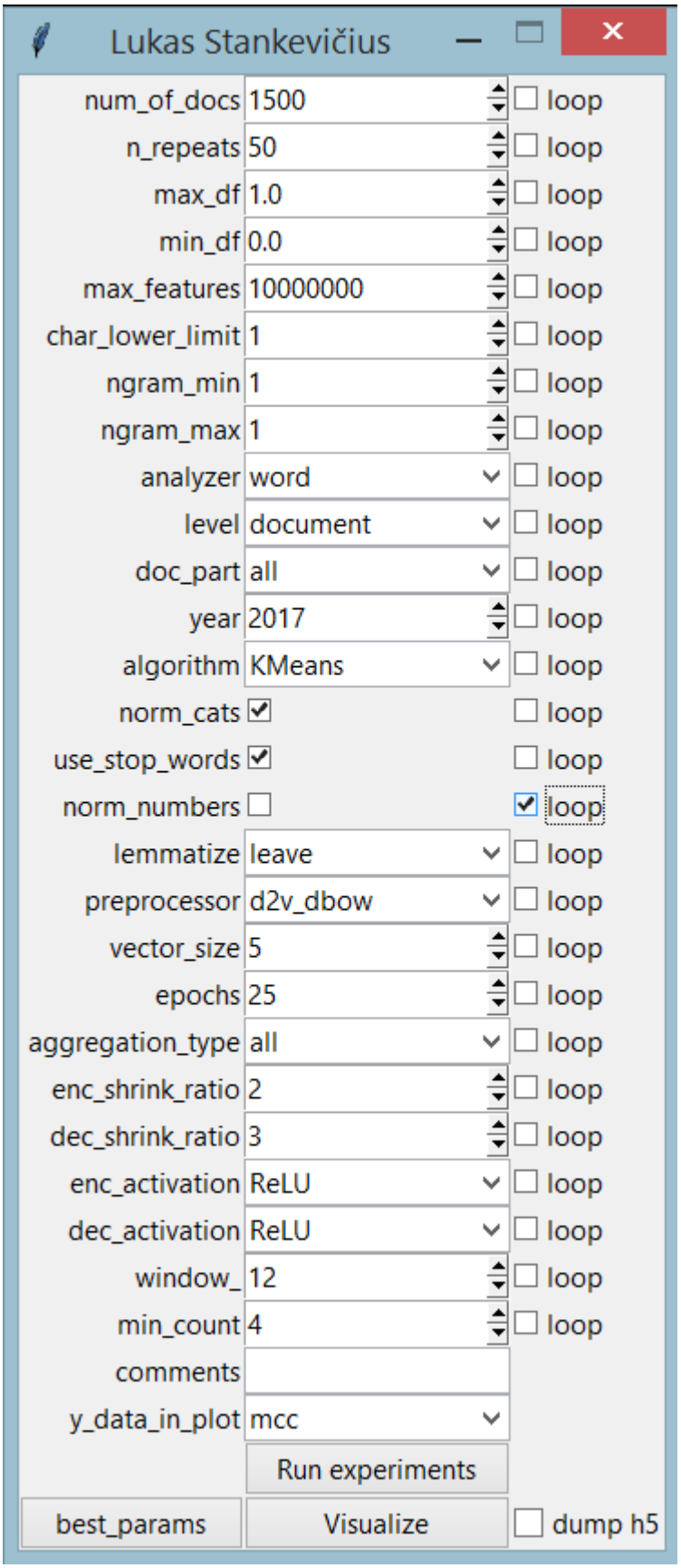
56. NewsNow. [interaktyvus]. [žiūrėta 2019-01-22]. Prieiga per: https://www.newsnow.co.uk/h/.
57. Digg - What the Internet is talking about right now. [interaktyvus]. [žiūrėta 2019-01-22]. Prieiga per: http://digg.com/.
58. Slashdot: News for nerds, stuff that matters. [interaktyvus]. [žiūrėta 2019-01-22]. Prieiga per: https://slashdot.org/.
59. Techmeme. [interaktyvus]. [žiūrėta 2019-01-22]. Prieiga per: https://techmeme.com/.
60. Newstral - News review. [interaktyvus]. [žiūrėta 2019-01-22]. Prieiga per: https://newstral.com/en.
61. Newsmap. [interaktyvus]. [žiūrėta 2019-01-22]. Prieiga per: http://www.newsmap.jp/.
62. newsola - top news at a glance. [interaktyvus]. [žiūrėta 2019-01-22]. Prieiga per: http://www.newsola.com/#/us/.
63. Bing News. [interaktyvus]. [žiūrėta 2019-01-22]. Prieiga per: https://www.bing.com/news.
64. Portal:Current events - Wikipedia. [interaktyvus]. [žiūrėta 2019-01-22]. Prieiga per: https://en.wikipedia.org/wiki/Portal:Current_events.
65. All news, US and international. [interaktyvus]. [žiūrėta 2019-01-22]. Prieiga per: https://www.reddit.com/r/news/.
66. States report. [interaktyvus]. [žiūrėta 2019-01-22]. Prieiga per: https://statesreport.com/.
67. Virwire | Live Viral News. [interaktyvus]. [žiūrėta 2019-01-22]. Prieiga per: https://virwire.com/.
68. freshnews - fresh tech news from around the web. [interaktyvus]. [žiūrėta 2019-01-22]. Prieiga per: http://www.freshnews.org/.
69. Corax. [interaktyvus]. [žiūrėta 2019-01-22]. Prieiga per: https://cor.ax/.
70. HALE, Jeff [interaktyvus].2018. [žiūrėta 2019-05-06]. Prieiga per: https://towardsdatascience.com/deep-learning-framework-power-scores-2018-23607ddf297a.
71. Lietuvių kalbos žodžių formų žodynas. [interaktyvus]. [žiūrėta 2019-05-06]. Prieiga per: http://www.morfologija.lt/.
72. RIMKUTĖ, Erika ir kt. Lietuvių kalbos morfemikos duomenų bazė. [interaktyvus]. [žiūrėta 2019-05-06]. Prieiga per: http://tekstynas.vdu.lt/~irena/apie_morfema_2.php.

PRIEDAI

1 priedas. Modulio „experiment“ klasių diagrama



2 priedas. Sistemos pagrindinis vartotojo sąsajos langas



3 priedas. *doc2vec_pilnas* geriausio dviejų savaičių grupavimo savybės

		Tikroji kategorija										
Grupės Nr.	Straipsnių skaičius	Kita	Kriminalai	Kultūra	Lietuvos naujienos	Mokslas	Nuomonės	Pasaulio naujienos	Pramogos	Sportas	Verslas	Geriausiai aprašantys požymiai, arba 10 terminų su didžiausiu TF-IDF svoriu, gauti išfiltruojant grupes su minimaliu 0,1 ir maksimaliu 0,8 terminų pasikartojimo grupėse dažniu.
1.	231	**72**	0	**72**	39	6	1	4	29	5	3	muzikos, koncertas, džiazo, paroda, režisierius, kultūros, teatro, rež, kino, fusedmarc
2.	112	0	0	0	0	0	0	0	0	**112**	0	taškų, rungtynėse, raptors, rungtynes, pelnė, čempionato, cavaliers, taškus, lkl, tšk
3.	273	10	0	2	**225**	0	17	2	0	6	11	seimo, partijos, socialdemokratų, paluckas, valstybės, apkaltos, seime, mg, seimas, skvernelis
4.	212	13	**159**	1	29	2	1	6	0	1	0	policijos, vyras, gim, policija, kauno, teismas, kokaino, prom, įvykio, vairuotojas
5.	243	22	3	4	**115**	5	4	6	1	1	82	eur, šilumos, savivaldybės, valstybės, mokslo, kauno, įmonės, gyventojų, savivaldybė, turto
6.	302	19	2	3	14	1	0	**258**	2	0	3	pranešė, žuvo, policija, sirijos, policijos, naujienų, islamo, opozicijos, ukrainos, maduro
7.	245	4	0	0	23	7	2	**204**	0	0	5	korėjos, trumpas, nato, trumpo, korėja, sirijos, gynybos, prezidento, pajėgų, branduolinį
8.	37	1	0	0	32	0	0	2	0	0	2	šilumos, temperatūra, laipsnių, laipsnius, debesys, įdienojus, vėjas, debesuotumas, pūs, šils
9.	211	45	1	3	**90**	17	39	6	0	1	9	mokslo, alkoholio, gyventojų, ntv, valstybės, vėžio, visuomenės, švietimo, svarbu, žmogaus
10.	154	10	0	0	8	0	5	**130**	0	0	1	pen, le, macronas, rinkimų, es, prancūzijos, macrono, prezidento, may, brexit
Suma	2020	196	165	85	575	38	69	618	32	126	116	

4 priedas. *doc2vec_dalis* geriausio dviejų savaičių grupavimo savybės

		Tikroji kategorija										
Grupės Nr.	Straipsnių skaičius	Kita	Kriminalai	Kultūra	Lietuvos naujienos	Mokslas	Nuomonės	Pasaulio naujienos	Pramogos	Sportas	Verslas	Geriausiai aprašantys požymiai, arba 10 terminų su didžiausiu TF-IDF svoriu, gauti išfiltruojant grupes su minimaliu 0,1 ir maksimaliu 0,8 terminų pasikartojimo grupėse dažniu.
1.	227	3	0	0	22	4	2	**192**	0	0	4	korėtis, raketa, nato, karinis, branduolinis, pajėgos, korėja, jungtinis, karas, sirijos
2.	299	9	1	1	45	2	0	**236**	3	1	1	policija, įtariamas, žūti, pajėgos, pratybos, sulaikyti, agentūra, lėktuvas, teismas, išpuolis
3.	155	15	0	0	**107**	0	30	0	0	0	3	socialdemokratas, vyriausybė, valstietis, premjeras, mokslas, partija, paluckas, pirmininkas, koalicija, frakcija
4.	194	13	0	0	27	0	5	**145**	0	0	4	le, rinkimas, pen, macronas, es, prancūzijos, brexit, macrono, partija, turas
5.	280	**85**	1	**81**	46	9	8	12	28	5	5	muzika, koncertas, teatras, biblioteka, režisierius, paroda, muziejus, festivalis, filmas, konkursas
6.	235	19	**161**	0	33	1	0	17	0	1	3	policija, vairuotojas, teismas, įtariamas, gim, sulaikyti, eismas, asmuo, ikiteisminis, prom
7.	270	44	1	1	**96**	22	17	14	1	0	74	įmonė, pajamos, eur, mokestis, rinka, asmuo, paslauga, liga, visuomenė, kokainas
8.	208	7	1	2	**167**	0	7	0	0	4	20	partija, pūkas, savivaldybė, teismas, komisija, liberalas, mg, įmonė, apkalti, baltic
9.	37	1	0	0	**32**	0	0	2	0	0	2	šiluma, temperatūra, debesis, įdienojus, debesuotumas, lietus, pūs, šils, termometrai, vėjas
10.	115	0	0	0	0	0	0	0	0	**115**	0	rungtynės, taškas, ekipa, čempionatas, raptors, pelnyti, rinktinė, kėlinys, varžybos, varžovas
Suma	2020	196	165	85	575	38	69	618	32	126	116	

5 priedas. TF-IDF geriausio dviejų savaičių grupavimo savybės

		Tikroji kategorija										
Grupės Nr.	Straipsnių skaičius	Kita	Kriminalai	Kultūra	Lietuvos naujienos	Mokslas	Nuomonės	Pasaulio naujienos	Pramogos	Sportas	Verslas	Geriausiai aprašantys požymiai, arba 10 terminų su didžiausiu TF-IDF svoriu, gauti išfiltruojant grupes su minimaliu 0,1 ir maksimaliu 0,8 terminų pasikartojimo grupėse dažniu.
1.	63	15	0	0	**34**	4	8	1	0	0	1	universitetas, mokslas, švietimas, studija, reforma, rektorius, pertvarka, akademija, mokytojas, studentas
2.	105	0	0	0	0	0	0	0	0	**105**	0	rungtynės, taškas, ekipa, čempionatas, komanda, pergalė, pelnyti, rinktinė, kėlinys, žaidėjas
3.	419	**117**	7	74	94	20	24	32	29	14	8	vaikas, muzika, kultūra, muziejus, biblioteka, liga, koncertas, gyvenimas, knyga, teatras
4.	279	9	0	0	84	0	13	**172**	0	0	1	partija, socialdemokratas, pirmininkas, politinis, turas, premjeras, koalicija, rinkėjas, lyderis, derybos
5.	34	0	0	0	**32**	0	0	2	0	0	0	laipsnis, šiluma, temperatūra, naktis, debesis, lietus, debesuotumas, vėjas, dangus, palyti
6.	595	38	22	10	**227**	11	24	162	1	7	93	valstybė, teismas, įstatymas, įmonė, seimas, vyriausybė, asmuo, projektas, taryba, savivaldybė
7.	171	6	2	1	16	1	0	**141**	1	0	3	lėktuvas, žūti, pajėgos, karinis, uostas, policija, skrydis, valstybė, ataka, agentūra
8.	110	2	0	0	25	2	0	**80**	1	0	0	korėtis, pratybos, branduolinis, raketa, karinis, karys, pajėgos, pietūs, laivas, gynyba
9.	202	9	**134**	0	22	0	0	28	0	0	9	policija, vairuotojas, įtariamas, įvykis, sulaikyti, eismas, asmuo, gatvė, apskritis, komisariatas
10.	42	0	0	0	**41**	0	0	0	0	0	1	seimas, pūkas, apkalti, komisija, parlamentaras, apkaltas, priesaika, saugumas, posėdis, frakcija
Suma	2020	196	165	85	575	38	69	618	32	126	116	