KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
PROGRAMŲ SISTEMŲ INŽINERIJOS STUDIJŲ PROGRAMA

REMIGIJUS VALYS

ŽINIATINKLIO DOKUMENTŲ AUTOMATINIO ŽYMĖJIMO METODŲ TYRIMAS

Magistro baigiamasis darbas

Darbo vadovas
prof. dr. R. Damaševičius

KAUNAS, 2014

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
PROGRAMŲ SISTEMŲ INŽINERIJOS STUDIJŲ PROGRAMA

REMIGIJUS VALYS

ŽINIATINKLIO DOKUMENTŲ AUTOMATINIO ŽYMĖJIMO METODŲ TYRIMAS

Magistro baigiamasis darbas

Darbo vadovas
prof. dr. R. Damaševičius
2014-05-20

Recenzentas
dr. Martynas Patašius
2014-05-20

Atliko:
IFM-2/2 gr. studentas
Remigijus Valys
2014-05-20

KAUNAS, 2014

SANTRAUKA

Žiniatinklio dokumentų automatinis žymėjimas dar nėra plačiai paplitęs. Tačiau šiandien žymiai gausėjant internetiniams dokumentams tai yra viena iš vis didesnio dėmesio susilaukiančių teksto analizavimo sričių. Automatizuotas žiniatinklio dokumentų žymėjimas galėtų padėti internautams greičiau rasti norimą informaciją, susieti skirtingų autorių parašytą informaciją.

Siekiant nustatyti automatinio žiniatinklio dokumentų žymėjimo galimybes, darbe nagrinėjami teksto analizavimo algoritmai. Aptariami teksto analizavimo sunkumai, bei būtini žingsniai sukurti automatinę teksto žymėjimo sistemą. Apžvelgiami susiję sprendimai, jų teikiamas funkcionalumas.

Darbe pateikiamas įgyvendintos žiniatinklio dokumentų žymėjimo sistemos aprašas, jos esminės savybės. Atliekamas eksperimentinis teksto analizavimo algoritmų metrikų tyrimas. Tyrimo metu tiriamas rastų raktažodžių tikslumas, algoritmo sugebėjimas atkurti realių autorių priskirtus raktažodžius, bei kaip analizuojamas tekstas, jo ilgis, bei dokumento sritis, daro įtaką analizavimo rezultatams.

SUMMARY

Web document automatic tagging isn't very widely used today. But because of significantly increasing number of documents, this area of natural language processing attracts a lot of attention. Automated web document tagging could help users to quickly found information. It also can connect texts that are written by different authors.

In order to determine possibilities of web document tagging system, possible analysis algorithms are discussed. Programming models, common problems of natural language processing and required steps that need to be done to create automatic tagging system, a review of decisions, provided functionality is described.

The paper provides implementation of the web documents automatic tagging system and a description of its essential features. Performed experiment provides information how different algorithms behave, what metrics they have. Experiment also tries to investigate how different text length and context affects precision and recall.

TURINYS

LENTELIŲ SĄRAŠAS

PAVEIKSLŲ SĄRAŠAS

TERMINŲ IR SANTRUMPŲ ŽODYNAS

- XML (angl. „Extensible Markup Language“) – būdas aprašyti perduodamus arba siunčiamus duomenis.
- JSON (angl. „JavaScript Object Notation“) – būdas aprašyti javascript kalbos objektus.
- NLP (angl. „Natural language processing“) – sritis tirianti tekstų analizavimą.
- MIT licencija – viena iš atviro kodo licencijų.
- .NET – kompanijos Microsoft kuriamas ir palaikomas karkasas, skirtas kurti programas.
- LINQ (angl. „Language-Integrated Query“) – duomenų manipuliavimo būdas .NET karkase.
- HTML5 – kalba skirta aprašyti naršyklėje atvaizduojamus interneto puslapius.
- URM (angl. „Universal right management“) – būdas valdyti vartotojo prieigą.

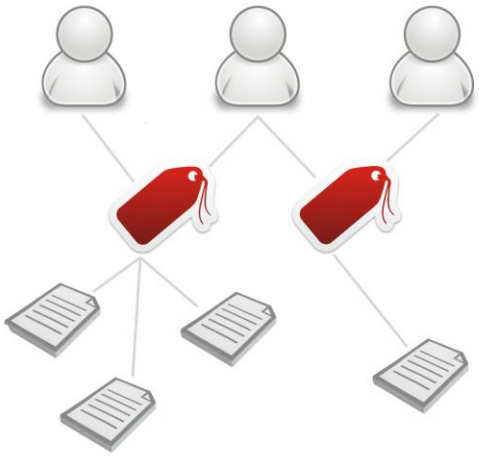
1. ĮVADAS

Pastaraisiais metais internete atsirado nemažai naujų sistemų, kurios įtraukia vartotojus dalyvauti sistemos kūrime. Tokiose sistemose, vartotojai nebėra tik stebėtojai. Kurdami internetinius dokumentus, bei pateikti savo duomenis, bendraudami, dalindamiesi informacija, vartotojai prisideda prie sistemos evoliucionavimo. Dauguma tokio tipo sistemų yra pateikiamos kaip tinklai. Tinklas, apibūdinantis vienos srities dokumentus gali sieti panašių idėjų turinčius bendraminčius. Analizuojant tinklą, galima surasti visus dokumentus apie norimą sritį, kategoriją. Tokios programos susilaukia didelio vartotojų dėmesio, yra populiarios.

Šiandien galime pasidalinti savo mintimis apie viską: įvykius, naujienas, įspūdžius. Kadangi informacijos kiekis yra didelis, internetiniams dokumentams yra suteikiami raktažodžiai apibūdinantys tekstą. Vartotojams yra siūloma prie pateikiamo dokumento pateikti ir kelis raktažodžius, kurie apibūdintų patį dokumentą. Taip kitiems vartotojams lengviau surasti norimą informaciją ar pačiai sistemai teikti rekomendacijas apie turinį, kuris galėtų būti aktualus kitiems vartotojams [1]. Pagrinde egzistuoja kelios problemos:

- toks turinio apibūdinimas nėra privalomas, yra tik pageidautinas. Tai sąlygoja, kad didelė informacijos dalis yra tiesiog tekstas, be papildomų duomenų,
- vartotojams sunku panaudoti jau sistemoje panaudotus raktažodžius, taip sistemoje atsiranda sinonimų,
- egzistuoja rašybos klaidų tikimybė. Vartotojas priskirdamas raktažodį gali suklysti jį rašydamas.

Vienas iš būdų kaip spręsti tokias problemas yra naudoti automatinio dokumentų žymėjimo sistemą. Dokumentų žymėjimo sistema gali sujungti raktažodžius su informacija bei su vartotojais [2].



1.1 pav. Grafas jungiantis žymes su dokumentais bei vartotojais

Tokia sistema gali pasiūlyti raktažodžius pateikiamai vartotojo informacijai. Sistemai pateikus siūlomus raktažodžius vartotojas gali pasirinkti iš pateikto sąrašo. Jei vartotojas mano, kad pateikiami raktažodžiai neatitinka realios prasmės jis gali koreguoti raktažodžius. Taip galima spręsti problemas dėl neaprašytų duomenų, sinonimų naudojimo, gramatinių klaidų.

Sudėtingos sistemos naudoja algoritmus, kurie analizuoja ne tik pateikiamą tekstą, tačiau atsižvelgia ir į patį turinį, ankstesnius vartotojo raktažodžius.

1.1. DARBO TIKSLAS

Darbo tikslas – išanalizuoti automatinį internetinių dokumentų žymėjimo procesą. Suprasti pagrindines proceso kliūtis, pasirenkamus sprendimus problemoms spręsti. Sukūrus galima žiniatinklio dokumentų automatinio žymėjimo procesą ištirti pasirinktų algoritmų panaudojimo galimybes. Ištirti kaip tiriami algoritmai gali analizuojami realius, kitų autorių tekstus. Išanalizuoti skirtingų kategorijų, bei teksto ilgių įtaką analizavimo procesui.

2. ANALITINĖ DALIS

2.1. DUOMENŲ GAVYBA

Bendru atveju duomenų gavyba yra procesas, kurio metu duomenys yra analizuojami įvairių aspektais. Analizės metu rasti duomenys yra apjungiami į naudingą informaciją. Teksto duomenų gavyba automatizuotai atranda naują, prieš tai nežinomą informaciją, analizuojamuose šaltiniuose [3]. Teksto duomenų gavyba prasideda nuo to, kad analizuojamame tekste yra išskiriami faktai ar įvykiai, kurie vėliau yra analizuojami tradiciniais duomenų gavybos metodais. Galima išgauti reikalingą informaciją apie norimą tekstą, susieti žmones, vietas, ar panašius objektus. Identifikuoti, organizuoti ir grupuoti dokumentus pagal jų turinį.

Naudojant analizavimo algoritmus sprendimai remiasi „skaldyk ir valdyk“ principu [4]. Kai yra žinoma konkreti užduotis kurią reikia išspręsti, algoritmai sudalina sudėtingą ir didelę užduotį į mažesnes, bei paprastesnes. Mažesnės užduotys yra išsprendžiamos atskirai ir tuomet gauti rezultatai yra apjungiami. Juose egzistuojantys pagrindiniai žingsniai [5][6]:

- teksto paruošimo algoritmams žingsnis. Šio etapo metu yra formuojamas tekstas, filtruojami nereikalingi, nenaudingi simboliai,
- informacijos išgavimo žingsnis. Iš atskirų tekste esančių žodžių yra ieškoma susijusi informacija. Šiame etape yra pašalinami žodžiai kurie egzistuoja tekste, tačiau nesuteikia informacijos. Šiame etape norint išgauti kuo tikslesnę informaciją be paprastų šablonais grįstų algoritmų yra naudojami protingesni algoritmai, analizuojami kelių žodžių junginiai, ar žodynai, norint išgauti žodžių prasmes.
- normalizavimo etapas. Šiame etape iš atrengtų žodžių yra išskiriamas žodžio kamienas. Bei bandoma įvertinti, kad egzistuoja keli žodžiai kurie sudaro pavadinimą („saulės slėnis“). Tuomet galima atlikti papildomą teksto analizę.
- žodžių pasirinkimas. Šiame žingsnyje pasirenkami žodžiai, kurie galėtų būti raktažodžiai. Šis žingsnis panaikina neinformatyvius žodžius. Neinformatyvūs žodžiai(pavyzdžiui „ir“, „ar“, „arba“) nebus naudojami kaip raktažodžiai, todėl juos galima drąsiai pašalinti. Taip padidinamas būsimų raktažodžių tikslumas, bei sumažinamas būsimų skaičiavimų laikas.
- svorių skaičiavimo žingsnis. Šiame etape stengiamasi priskirti žodžiams svorius, kurie apibūdina kaip naudojamas žodis yra svarbus konkrečiame kontekste. Skirtingi algoritmai skaičiuoja, interpretuoja ir pateikia svorius skirtingai, tačiau bendru atveju yra laikoma, kad didesnį svorį turintis žodis yra svarbesnis tekste, nei tas, kurio svoris yra mažesnis.

- panašumo įvertinimo žingsnis. Norint gauti tikslesnius raktažodžius, kai kurie algoritmai naudoja būdus, kurie įvertina vieno teksto fragmento panašumą su kitais tekstais. Tokie skaičiavimai yra sudėtingi, bei reikalauja daug laiko, tačiau daug efektyviau leidžia panaudoti jau naudotus raktažodžius [7].
- raktažodžių parinkimo žingsnis. Šiame žingsnyje yra pasirenkami konkretūs raktažodžiai, geriausiai apibūdinantys tekstą.
- testavimas. Žingsnis nenaudojamas, kai algoritmas yra naudojamas realioje sistemoje, tačiau būtinas kiekvienam algoritmui. Žingsnyje yra įvertinamas algoritmo tikslumas ir atkūrimas. Tai parodo kaip gerai algoritmas gali parinkti raktažodžius.

2.2. TEKSTO AUTOMATINIS ŽYMĖJIMAS

Teksto žymėjimas suprantamas kaip procesas, kurio metu automatizuotai analizuojamam tekstui yra priskiriami raktažodžiai, kurie apibūdina patį tekstą. Toks procesas susideda iš dviejų pagrindinių žingsnių, tai: mokinimo ir spėjimo [8]. Pirmasis, mokinimosi žingsnis yra skirtas "apmokyti" analizavimo algoritmus. Šiame žingsnyje analizei yra pateikiami tekstai, bei raktažodžiai išskirti žmogaus. Mašininio mokinimosi algoritmams pateikti duomenys leidžia suprasti dalykinę sritį. Kuo didesnis, bei kuo tikslesnis mokinimo duomenų yra pateikiama algoritmams tuo geresnių rezultatų galima tikėtis sekančiame etape. Priklausomai nuo pačio algoritmo šis žingsnis gali būti ne tik naudojamas pradinėje fazėje, prieš pradedant naudoti sistemą, tačiau ir pačios naudojimo metu, taip tik padidinant vėlesnių duomenų tikslumą.

Kadangi kompiuterizuota sistema negali logiškai suprasti kas tiksliai yra norima pasakyti konkrečiame dokumente ar jo ištraukoje, yra atliekami spėjimai, kurie remiasi ankstesnių teksto analizių rezultatais. Jau sukaupta algoritmo "patirtis" yra panaudoti realių, vartotojų pateiktų duomenų analizei [9].

Priklausomai nuo konkretaus algoritmo, bei norimo tikslumo tekstai gali būti analizuojami įvairiai. Vieni algoritmai naudoja statistinę analizę siekdami nustatyti atskirų žodžių, bei žodžių grupių pasikartojimą tekste, kiti, sudėtingesni, bei daugiau laiko reikalaujantys algoritmai naudoja analizę pagrįsta žodynais, kurie gali interpretuoti tekstą.

2.3. EGZISTUOJANTYS SPRENDIMAI

Bet kokia sistema turi pirma analizuoti tekstą norėdama pateikti informaciją susijusią su tekstu. Skirtingos sistemos naudoja gauta informaciją apie analizuojamą tekstą skirtingai. Automatinė dokumentų analizė yra naudojama įvairiose srityse. Internetinių dokumentų paieškos sistemos nori susieti vartotojo ieškomą frazę su geriausiais atitikmenimis. Automatinio teksto pavadinimo sudarymo sistemos analizuoja tekstą norint išskirti pagrindinį teksto subjektą, kuris bus naudojamas pavadinime. Automatinio žymėjimo sistemos analizuoja tekstą, bei išskiria geriausiai tekstą alegorizuojančius raktažodžius.

Nors visų šių sistemų galutinis tikslas yra skirtingas, teksto analizavimo algoritmai yra panašūs. Toliau yra apžvelgiami keli egzistuojantys sprendimai, kurie naudoja teksto analizavimo algoritmus.

2.3.1. Term Extraction

Kompanijos “Yahoo” teikiama paslauga. Ši paslauga sparčiai tobulinama, atnaujinama. Galima gauti naudotis interneto naršyklės pagalba, nereikia savoje mašinoje įdiegti jokių programų, tačiau pagrindinis autorių tikslas sukurti paslaugą, kurią kaip komponentą būtų galima integruoti į kitas sistemas. Todėl prieinama interneto naršyklės versija yra skirta tik derinimo tikslas ir nėra patogi naudojimui galutiniam vartotojui. Puslapyje yra vieta, į kurią vartotojas gali įklijuoti tekstą, duomenys yra gaunami xml formatu.

Teikiama paslauga yra mokama. Bandomoji versija riboja galimų užklausų kiekį. Norint naudoti paslaugą be apribojimų reikia įsigyti komercinę licenciją.

Autoriai nepateikia naudojamų algoritmų pavadinimų, tačiau iš eksperimentų yra aišku, jog naudojami mašininio mokinimosi algoritmai, bei žodynai. Pavyzdžiui nurodant tekstą kuriame yra minimas žymaus paveikslo „Mona Liza“ pavadinimą, vienas iš pateikiamų rezultatų buvo gautas autoriaus „Leonardo da Vinčio“ vardas, nors pats autorius tekste nebuvo minimas. Norint gauti tikslesnius rezultatus reikia įvesti didelį kiekį duomenų.

2.3.2. TagCrowd

Paslauga pateikiama kaip interneto puslapis. Apsilankęs puslapyje vartotojas turi tris galimybes tekstui analizuoti:

- įklijuoti tekstą,
- nurodyti dokumento internetinį adresą,
- įkelti dokumentą į serverį

Rezultatai pateikiami eilute, kurią nesudėtinga kopijuoti. Be to galima nurodyti papildomų parametrų, analizavimui patikslinti:

- pasirinkti kalbą, sistema palaiko 18 skirtingų kalbų,
- pasirinkti norimų rasti raktažodžių kiekį,
- pasirinkti minimalų dažnumą, tik tie žodžiai, kurių dažnumas yra didesnis nei nurodytas minimalus, bus siūlomi kaip raktažodžiai,
- grupuoti panašius žodžius,
- įvesti nenorimus rasti raktažodžius.

Patogesnis būdas naudoti sistemą galutiniam vartotojui. Egzistuojantys parametrai leidžia įtakoti analizavimą.

Sistema galinti analizuoti tekstą bei išskirti esminius, daugiausia pasikartojančius žodžius duotame tekste. Algoritmas yra grįstas žodžių dažnumo skaičiavimu. Taip pat sistema turi naudoti bent minimalų žodyną norėdama grupuoti panašius žodžius. Mašininio mokinimosi algoritmai nėra naudojami.

Analizavimo paslauga yra nemokama, naudojant ją nekomerciniais tikslais. Už papildomą mokestį autoriai siūlo pasinaudoti rastų raktažodžių vizualizavimo paslauga.

2.3.3. VisualText

Tai integruota kūrimo aplinka, kuria naudojantis vartotojai gali analizuoti tekstus. Kurdami šią sistemą autoriai taip pat sukūrė mažą programavimo kalbą, kuri palengvina darbą su tekstu analizavimu. Turi daug funkcijų:

- automatinis žymėjimas – iš pateikto sistemai teksto yra išgaunami tekstą apibūdinantys raktažodžiai,
- informacijos išgavimo – pateikiant tekstą algoritmai randa tekstuose esančią informaciją. Šią informaciją gali toliau analizuoti, bei lyginti su turimais duomenimis,
- esybių radimo – iš teksto gali išskirti vietas, vardus, asmenis, pavadinimus,
- kuri informacijos indeksus – skirta išanalizavus didelį informacijos kiekį sukurti indeksus, kurie leistų greitai ir efektyviai paskui surasti informaciją pagal kriterijų. Panaudojama didelių duomenų kiekiui, interneto puslapių indeksavimui.
- filtravimo – pateikiant raktažodžius ir tekstą, sistema gali atskirti ar tekstas yra tiesiogiai susijęs su raktažodžiais,
- kategorijų priskyrimo – dokumentus suskaido į kategorijas,
- aprašymo – galima pateikus tekstą gauti trumpą jo aprašymą,
- vykdyti natūraliosios kalbos užklausas – galima pateikti klausimus apie tekstą.

Sistema yra nemokama, visi norintys gali ją parsisiųsti ir įsidiegią savo kompiuteryje naudoti. Nesudėtinga analizuoti tekstą, bei gauti teksto informaciją. Tačiau norint išnaudoti kitas sistemos funkcijas reikia išmokti autorių sukurtą kalbą. Tokia sistema yra labai sudėtinga naudoti.

Autoriai naudoja NLP biblioteką tekstų analizavimui, bibliotekas, mašininį mokinimąsi, bei neuroninių tinklų algoritmus.

2.3.4. Topicalizer

Internetinis teksto analizės įrankis. Panašiai kaip ir „Term extraction“ pagrinde yra skirtas integruoti į kitas sistemas, kaip komponentas. Sistema gali analizuoti pateikiamą tekstą arba interneto šaltinį. Turi realizuotus kelis skirtingus algoritmus pagrįstus žodžių dažnio skaičiavimu, bei mašininiu mokinimusi. Įrankis iš duoto teksto gali:

- išskirti raktažodžius,
- sukurti trumpą aprašymą iš ganėtinai didelio kiekio duomenų,
- aptikti žodžių junginius,
- aptikti naudojamą kalbą.

Įrankis yra nemokamas, jis yra paviešintas naudojant MIT licenciją.

2.3.5. OpenCalais

Sistema tekstų analizavimui pagrįsta interneto paslaugomis. Kaip ir „Term extraction“ autoriai sukūrė karkasą, kuris gali būti integruojamas kaip komponentas į kitas sistemas. Pagrindinis tikslas yra automatizuoti žiniatinklio dienoraščių tekstų raktažodžių priskyrimą. Tam tikslui autoriai siūlo naudoti jau esamus papildinius įvairiems interneto dienoraščių karkasams, tokiems kaip „Wordpress“ ar „Drupal“. Sistemoje taip pat realizuota peržiūra naudojant naršyklę. Tai ne pagrindinis sistemos naudojimas, skirta derinimo tikslais.

Norint naudoti sistemą pakanka užsiregistruoti, bei gauti vartotojo raktą. Nemokama sistemos versija turi apribojimus, ribojamas užklausų skaičius, per valandą, bei per mėnesį. Norint naudotis sistemą be apribojimų reikia įsigyti licenciją.

Sistemoje analizuojant tekstus yra naudojami mašininio mokinimosi algoritmai, kurie išskiria:

- esybes – žmones, įmones, organizacijas, knygas, vietas,
- faktus – išsilavinimą, darbo vietą,
- įvykius – sporto, valdymo, darbo pasikeitimą.

Ši išskirta informacija yra gražinama vartotojui, kuris turi pasirinkti, kuriuos raktažodžius naudoti.

2.3.6. Sistemų palyginimas

2.1 Dokumentų žymėjimo sistemų palyginimas

Sistema	**Nemokama**	**Ar būtina įdiegti**	**Ar galima integruoti į kitas sistemas**	**Naudojamų algoritmų tipas**	**Sąlyginis naudojamų algoritmų sudėtingumas**	**Sąlyginis randamų rezultatų tikslumas**	**Patogu naudoti**
Term extractor	Yra apribojimų	Ne	Taip, per interneto paslaugas	Mašininio mokinimosi	3	3	Vidutiniškai
TagCrowd	Taip	Ne	Ne	Dažnumo	1	1-2	Taip
VisualText	Taip	Taip	Ne	Mašininio mokinimosi, neuroniniai tinklai	5	4-5	Ne
Topicalizer	Taip	Ne	Taip, kaip komponentas	Mašininio mokinimosi	2	1-2	Taip
OpenCalais	Yra apribojimų	Ne	Taip, per bibliotekas	Mašininio mokinimosi	4	4-5	Automatizuotas procesas

Sąlyginis naudojamų algoritmų sudėtingumas skirtas įvertinti sistemas tarpusavyje. Didesnį balą turinti sistema naudoja sudėtingesnius algoritmus, randa daugiau informacijos.

Dokumentų sistemų palyginimas yra pateiktas 2.1 lentelėje. Egzistuoja skirtingų sistemų, tačiau geriausi rezultatai yra sistemų, kurios naudoja sudėtingesniu, mašininio mokinimosi algoritmus. Patogiausia naudoti interneto sistemas, kurių nereikia instaliuoti, bei nereikia mokintis naujos programavimo kalbos, tam kad būtų galima atlikti analizę.

2.4. SPRENDIMO BŪDAI

2.4.1. Į užduotį orientuoti sprendimai

Į užduotį orientuoti būdai pasižymi algoritmas, kurie turi aiškią, formalią struktūrą. Analizuojant tekstą stengiamasi išskirti keturis pagrindinius aspektus:

- esybes – tai pagrindiniai objektai analizuojami tekste. Pavyzdžius žmonės, įmonės ar vietos,
- atributai – tai ypatybės apibūdinančios esybes. Pavyzdžius asmens vardas, pavardė, amžius, užimamos pareigos,
- faktai – tai yra ryšiai tarp esybių. Pavyzdžius faktas, kad žmogus dirba įmonėje,
- įvykiai – tai veiksmai kuriuose dalyvauja esybės. Pavyzdžiui asmuo švenčia gimtadienį, dalyvauja renginyje.

Analizuoto teksto dalys yra susiejamos su rastomis esybėmis. Pagrindinė problema yra, jog tokiose sistemose analizuojant tekstą yra iš anksto nustatytos taisyklės, kuriomis remiantis bus analizuojamas tekstas. Šios taisyklės, priklausomai nuo nagrinėjamos srities gali būti sudėtingos ir dažniausiai sudarinėjamos srities eksperto, bei skirtos konkrečiai užduočiai atlikti. Jei dėl kokių nors priežasčių reikės pakeisti taisykles, tai padaryti gali būti sudėtinga. Be to, tokie algoritmai yra „kieti“ jie negali prisitaikyti prie pasikeitusio stiliaus. Nors į sprendimą orientuoti būdai gali duoti tikslius rezultatus, tačiau tokios sistemos yra skirtos spręsti konkretų uždavinį ir tokius sprendimus sunku panaudojamos kitose srityje.

2.4.2. Mašininio mokinimosi sprendimai

Mašininio mokinimosi būdas remiasi prielaida, kad naudojami algoritmai bandys „suprasti“ analizuojamus tekstus ir duos tuo tikslesnius rezultatus, kuo daugiau dokumentų jie bus analizavę prieš tai[10]. Naudojami algoritmai, gali „išmokti“ rasti, bei apjungti informaciją analizuodami susijusius duomenis.

Dažniausiai analizuodami tekstą algoritmai išskirdami raktažodį tekstui jam priskiria ir požymį apibūdinantį rastą duomenų tikslumą. Naudojant mašininio mokinimosi algoritmus bandoma išspręsti problemas susietas su:

- besikeičiančiais duomenimis – sistema gali interpretuoti duomenis ir reaguoti lanksčiai. Pavyzdžiui jei daug žmonių staiga pradeda rašyti apie katastrofinę nelaimę, algoritmas į tai atsižvelgti skaičiuodamas raktažodžių svorius.
- daugkartinio panaudojimo – algoritmai, nėra skirti analizuoti konkrečią sritį ir gali būti panaudojami kitoje.
- algoritmai gali prisitaikyti prie konkretaus asmens ir jo rašymo stiliaus, gali būti personalizuojami.

Mašininio mokinimosi algoritmai taip pat turi ir trūkumų:

- greitis – norint neprisirišti prie konkrečios srities, bei pateikti tikslesnius raktažodžius reikia atlikti papildomą analizę.
- sudėtingumas – algoritmai yra sudėtingesni, nei į problemą orientuoti, bei jų rezultatai negali būti matomi iš karto.

2.5. ĮGYVENDINIMO PROBLEMOS

Kompiuterinės sistemos negali tiesiogiai suprasti konkretaus dokumento prasmės. Analizuojant dokumentus, išskiriant esybes, vietas, įvykius, faktus, bei juos siejant su kitais dokumentais, sistemos aptinka panašumą. Aptiktas panašumas apibūdina dokumentus, grupuoja juos į kategorijas. Analizavimo procesas yra sudėtingas, todėl susiduriama su problemomis. Pasitaikančios analizavimo problemos yra aprašytos toliau.

2.5.1. Greitis

Idealiomis sąlygomis kiekvienam duotam tekstui norima rasti visą susijusią informaciją. Didesnis susijusios informacijos kiekis leistų tiksliau apibūdinti dokumentą. Greitis glaudžiai susijęs su analizuojamu ir jau analizuotų duomenų kiekiu. Didesniam duomenų kiekiui apdoroti reikalinga daugiau resursų.

Dauguma sistemų šią problemą sprendžia tiesiog nebandydamos surasti visą įmanomą informaciją. Prieš analizuojant yra nustatomas dydis, kuris nustato ribą rekursiniam gyliui. Algoritmai analizuodami dokumentą bei radę informaciją, bandys ieškoti papildomos informacijos tik tada, jei tai leis nustatytas dydis.

Norint pasiekti greitesnių rezultatų taip pat yra ieškoma tik ta informacija, kurį galėtų būti naudinga. Yra naudojama euristika, manoma, kad tik daugiausiai kartų tekste pasikartoje žodžiai tekste yra svarbūs, bei apie juos rasta informacija bus reikšminga tekstui.

2.5.2. Tikslumas

Naudojami algoritmai yra skirtingi. Mašininio mokinimosi algoritmams gauti tikslesniu rezultatus reikia gerų duomenų „apsimokyti“. Apmokymas vykdomas algoritmams pateikiant duomenis, bei leidžiant juos analizuoti. Išanalizavus didelį duomenų kiekį, yra sukaupiama informacijos bazė, kuri vėliau naudojama realioje aplinkoje. Neįmanoma išanalizuoti visų įmanomų dokumentų, todėl pasirenkama dalis. Pasirinkti dokumentai, bei jų kiekis labai įtakoje tolimesnį sistemos veikimą.

Realioje aplinkoje tikslumas priklauso nuo galimybės aptikti susietus duomenis. Kuo tikslesni pradiniai duomenys, kuriais yra apmokami algoritmai, tuo tikslesnis gaunamas rezultatas. Realioje aplinkoje dirbantys algoritmai gali ir toliau tobulėti. Vartotojams analizuojant dokumentus rasti rezultatai gali būti pridedami prie egzistuojančios informacijos bazės.

Tikslumas yra glaudžiai susijęs su greičiu. Jei algoritmai naudoja euristiką arba paieškos gylio ribas, gali nukentėti randamų duomenų tikslumas. Reikalingas kompromisas.

2.5.3. Prisitaikymas

Algoritmas gali puikiai veikti vienoje aplinkoje, tačiau blogai kitoje. Pavyzdžiui jei naudojamas algoritmas buvo „apmokytas“ analizuojant su muzika susijusius įrašus jam bus sunku analizuoti medicininius dokumentus.

Kitas svarbus aspektas yra tai, kad informacija nuolat keičiasi. Svarbūs įvykiai tokie prasidėjusi olimpiada ar išsiveržęs ugnikalnis atsitinka santykinai retai. Rečiau naudojami įvykiai

yra prastesni kandidatai į raktažodžius. Analizavimo algoritmai turi reaguoti į staigų šuolį vienos srities tematika.

2.5.4. Personalizavimas

Susijusios informacijos gavimas priklausomai nuo to koks vartotojas pateikė užklausą yra nauja tyrimo kryptis. Internetinių dokumentų srityje yra norima, kad algoritmas sugebės susidoroti su plačiu vartotojų ratu, bei su dokumentais, kurių turinys gali būti labai platus. Viena iš anksčiau minėtų galimybių yra nuolatinis tobulėjimas, kai algoritmai nuolat pildo savo informacinę bazę vartotojo analizuotais dokumentais. Šios idėjos minusas yra, kad labai platus vartotojų spektras nesukonkretina konkretaus vartotojo analizuojamos srities. Geresnis sprendimas yra po vartotojo teksto analizės gautus duomenis panaudoti prisitaikant prie konkretaus vartotojo. Taip sistema turės vieną bendrą tašką nuo kurio visi sistemos vartotojai atskirai analizuodami pritaikys sistemą sau.

2.5.5. Nereikšmingi žodžiai

Kalboje yra naudojami žodžiai, kurie netinkami būti naudojami kaip raktažodžiai, tačiau egzistuoja tekste. Pavyzdžiui „aš“, „ir“, „arba“ ir t.t. Kad tokie žodžiai nebūtų naudojami kaip raktažodžiai yra sudaromas specialus sąrašas (angl. terminas „stop words list“)[11]. Šie žodžiai priklausomai nuo algoritmo analizuojant yra pašalinami arba neįtraukiami į atitinkamus sąrašus. Šis pašalinimas teigiamai įtakoje greitį.

Realioje aplinkoje kiekvienas vartotojas turėtų turėti galimybę keisti šį sąrašą. Tačiau konkretaus vartotojo pakeitimai neturi įtakoti kitų vartotojų.

2.5.6. Linksniai, bei sinonimai

Išskiriant raktažodžius yra tikslinga siūlyti vartotojui pasirinkti iš jau esamų ir naudojamų raktažodžių. Taip sistemoje nesudubliuoja esami raktažodžiai, lengviau rasti norimą informaciją. Taip pat analizuojant tekstus, žodžius su panašia reikšme verta. Grupuotų žodžių, bei dažniau pasikartojančių žodžių svoris yra didesnis.

Norint išvengti linksnių žodžiai yra normalizuojami. Iš duoto žodžio yra gaunama jo šaknis ir tik tuomet žodis yra naudojamas analizavimo procese. Taip tokie žodžiai kaip „greitaveika“ ar „greitumas“ yra normalizuojami ir toliau analizuojamas bendras žodis. Žodžio normalizavimo algoritmai, tokie kaip Porter-Stemmer[12], nebūtinai gražins žodį, kuris yra teisingas. Todėl normalizuoti žodžiai nėra naudojami konkrečių raktažodžių pasirinkimui.

Norint pašalinti sinonimus naudojami žodynai. Yra gaunami kiekvieno žodžio sinonimai, jie normalizuojami. Toliau visi žodžiai su sinonimais yra lyginami, bei randami panašūs. Pašalinant dažniausia pasirenkamas tas sinonimas, kuris yra dažniau naudojamas šnekamojoje kalboje arba dažniausiai naudojamas analizuojamame tekste. Vienas iš tokių sinonimų žodynų yra „WordNet“[13]. Procesas yra sudėtingesnis, tačiau reikalingas norint sumažinti informacijos dubliavimą. Siekiant didesnės spartos galima naudoti euristikas, pasirinkti tik tuos sinonimus, kurie yra dažnai naudojami kalboje, pasirinkti tik iš anksto nustatytą, maksimalų sinonimų skaičių.

Dar viena galimybė išvengti sinonimų naudojimo raktažodžiuose yra sudaryti galimų raktažodžių sąrašą. Tik žodžiai esantys sąraše bus galimi naudoti kaip raktažodžiai.

2.6. ALGORITMAI

Panašių sistemų autoriai nepateikia konkrečių naudojamų algoritmų pavadinimų. Toliau aprašomi algoritmai, rasti moksliniuose straipsniuose, kurie galėtų būti naudojami žiniatinklio dokumentų žymėjimui.

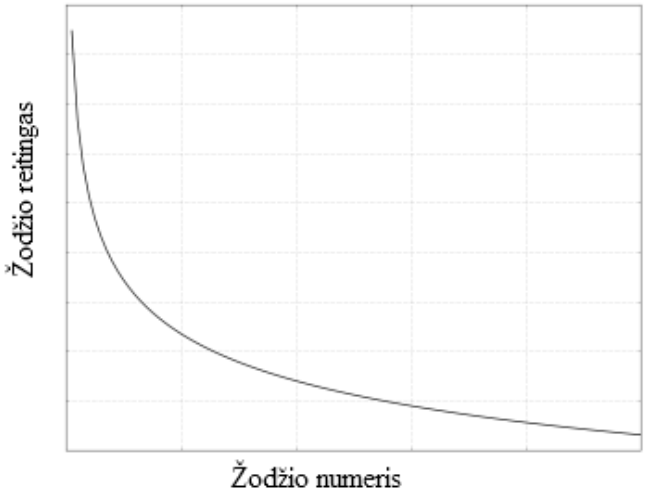
2.6.1. Dažniu grįstas algoritmas

Dažniu grįstas algoritmas yra klasikinis [22]. Algoritmas remiasi euristika, jog dažnai tekste pasikartojantys žodžiai yra geri kandidatai į raktažodžius. Tai sąlyginai daug skaičiavimų nereikalaujantis būdas raktažodžiams rasti.

Algoritmo žingsniai:

1. Suskaidyti tekstą į žodžius,
2. Išskirti kiekvieno žodžio kamieną naudojant Porter [12] algoritmą,
3. Suskaičiuoti žodžių svorius pagal dažnumą,
4. Pasirinkti dažniausiai tekste pasikartojančius raktažodžius.

Paprasčiausiu atveju toks algoritmas gali veikti be papildomos informacijos. Žodžių dažnumas gali būti skaičiuojamas tik pagal jų pasikartojimą tekste. Tačiau tokiu atveju algoritmas gerai galėtų analizuoti tik didelės apimties tekstus. Šį algoritmą nesudėtingai galima patobulinti. Prieš pradedami dirbti realioje aplinkoje turi būti apmokytas. Yra pateikiami tekstai, kurie yra analizuojami automatiškai. Analizuojant žodis yra normalizuojamas, bei yra skaičiuojama, kiek kartų žodis pasikartojo tekste. Išanalizavus didelį duomenų kiekį, algoritmas gali daryti prielaidą, kaip dažnai žodžiai yra naudojami kalboje. Apmokymo metu siekiama gauti žodžio dažnumą kalboje. Nors anglų kalboje kasmet atsiranda apytiksliai po porą tūkstančių naujų žodžių, tačiau žmonių įpratimai nesikeičia ir egzistuoja žodžiai, kurie yra naudojami dažniau nei kiti. Dažniau naudojami žodžiai yra geresni kandidatai į raktažodžius.



2.2 pav. Zipf taisyklė

Surinkus visus kalboje naudojamus žodžius, bei kiekvienam žodžiui priskyrus jo numerį, bei reitingą. kuris parodys kaip dažnai šis žodis yra naudojamas kalboje. Galima nupiešti žodžių dažnumo grafiką. Nupieštas žodžių dažnumo grafikas būtų panašus į „Zipf“ dėsnio grafiką, pateiktą 2.2 paveikslėlyje. „Zipf“ dėsnis teigia, kad žodžio reitingas yra atvirkščiai proporcingas žodžio numeriui. Tai parodo, kad tik nedidelė žodžių dalis yra naudojama dažnai. Dažniu grįsti algoritmai remdamiesi šia taisykle gali generuoti pakankamai tikslius rezultatus.

2.6.2. Žodžio dažnumo – atvirkštinio dokumento dažnumo algoritmas

Žodžio dažnumo – atvirkštinio dokumento dažnumo algoritmas (angl. „term frequency–inverse document frequency“) skaičiuoja konkretaus žodžio svarbumą konkrečiame tekste [23]. Bendram žodžio svoriui apskaičiuoti yra naudojamos dvi statistikos. Skaičiuojant, kiekvieną kartą rastas pasikartojantis žodis tekste didina svorį. Tačiau kiekvieno žodžio svoris yra sumažinamas pagal tai, kaip dažnai šis žodis yra naudojamas kalboje. Tai reiškia, kad kalboje dažniau naudojami žodžiai, turės mažesnį svorį tekste. Kuo svoris yra didesnis, tuo konkretus žodis yra laikomas svarbesnis tekstui ir jį yra prasminga naudoti kaip raktažodį. Toks skaičiavimo būdas padeda valdyti žodžių reikšmingumo skaičiavimą kalboje, nes tam tikri žodžiai tiesiog yra naudojami dažniau kalboje nei kiti [14]. Šis algoritmas taip pat remiasi „Zipf“ taisykle, tačiau skirtingai nei vien tik dažniu grįstas algoritmas, stengiamasi parinkti tuos raktažodžius, kurie kuris yra aktualūs tekstui.

Algoritmo žingsniai:

1. Suskaidyti tekstą į žodžius,
2. Išskirti kiekvieno žodžio kamieną naudojant Porter [12] algoritmą,
3. Suskaičiuoti žodžių svorius pagal dažnumą,
4. Išskirti teksto kontekstą,
5. Sumažinti dažnai pasikartojančių žodžių kontekste svorius.
6. Pasirinkti geriausius raktažodžius.

Skaičiuoti žodžių dažnumą tf(w, d) tekste galima įvairiai. Paprasčiausiu atveju skaičiuojant žodžių svorį skaičiuojamas vien tik konkretus dažnumas f(w, d), t.y. kiek kartų jis pasikartojo tekste. Egzistuoja kelios šio algoritmo atmainos:

- loginė – kai įvertinamas tik faktas jog žodis pasikartoja tekste:

$$tf(w,d) = \begin{cases} 1, \text{ jei žodis w yra dokumente d} \\ 0, \text{ priešingu atveju} \end{cases} \quad (1)$$

- logaritminė – kai skaičiuojant dažnumą, yra skaičiuojama logaritmo reikšmė:

$$tf(w,d) = \log(f(w,d)+1), \quad (2)$$

- normalizuotas – kai svoris priklauso nuo to, ne tik kiek kartų žodis panaudojamas tekste, bet ir nuo to, kiek iš vis tekste yra žodžių. Toks skaičiavimo būdas yra naudojamas, kai yra analizuojami didelės apimties tekstai:

$$tf(w,d) = 0{,}5 + \frac{0{,}5 * f(w,d)}{\max\{f(t,d) : t \in d\}} \quad (3)$$

Bendru atveju mažinant žodžių svorį yra naudojamas bendras kalboje naudojamų žodžių dažnumas. Norint jį gauti algoritmą galima apmokyti tokiu pačiu principu kaip ir dažnu grįstą algoritmą. Norint gauti tikslesnius rezultatus algoritmą galima papildyti. Vietoj bendro visos kalbos dažnumo, naudoti atskirų kontekstų dažnumą. Tuomet žodžio svoriai bus mažinami, nebe pagal žodžio pasikartojimą kalboje, bet pagal žodžių pasikartojimą kontekste.

Kad algoritmas galėtų sumažinti dažnai kontekste pasitaikančių žodžių svorį, pirmiausia reikia nustatyti teksto kontekstą. Kontekstas pasirenkamas pagal dažniausiai naudojamus tekste žodžius. Konteksto pasirinkimas atliekamas taip:

1. Pasirenkama dešimt daugiausiai naudojamų žodžiai,
2. Ieškomas kontekstas kuriame dominuotų pasirinkti žodžiai,
3. Jei kontekstas nerastas, sumažinamas žodžių skaičius, bei grįžtama į pirmą punktą,
4. Jei nepavyko rasti konteksto yra naudojamas bendras, visos kalbos kontekstas.

Žinant kontekstą galima suskaičiuoti atvirkštinį dokumentų dažnumą naudojant formulę:

$$idf(w,D) = \log \frac{N}{1+|\{d \in D : w \in d\}|} \quad (4)$$

čia:

- N – dokumentų skaičius analizuojamoje srityje,
- $|\{d \in D : w \in d\}|$ – skaičius dokumentų kuriuose žodis d yra rastas.

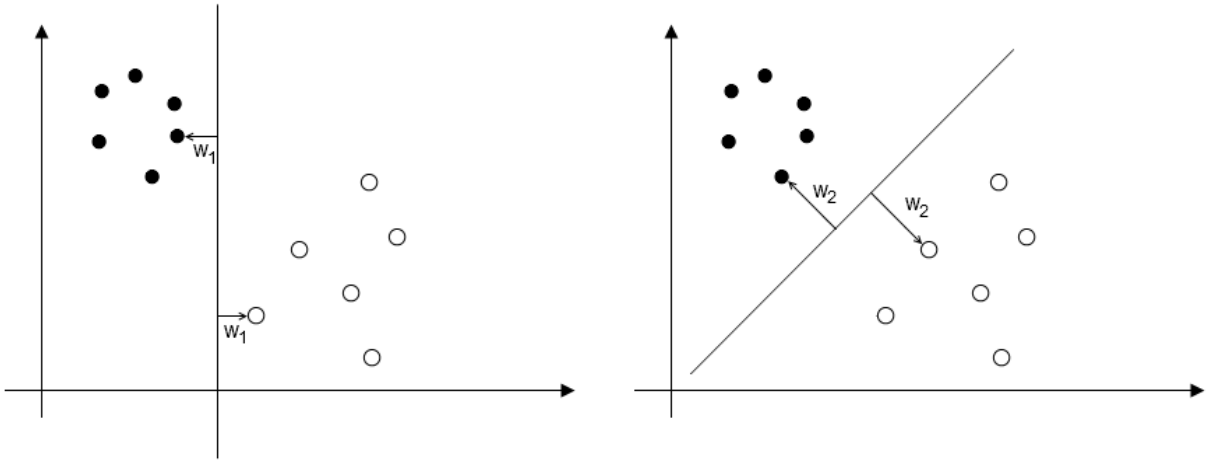
Žinant abu dydžius, žodžių dažnumą, bei atvirkštinį dokumentų dažnumą, galima apskaičiuoti bendrą dydį:

$$tfidf(w,d,D) = tf(w,d) * idf(w,D) \quad (5)$$

2.6.3. Atraminių vektorių klasifikatorius

Atraminių vektorių klasifikatoriaus (angl. „support vector machines“) [24] algoritmai priklauso mašininio mokinimosi algoritmų grupei [15]. Atraminių vektorių klasifikatoriaus algoritmai išskiria pasikartojančias sekas, šablonus, kurios vėliau yra naudojamos klasifikuojant tekstą.

Atraminių vektorių klasifikatoriaus algoritmui būtina atlikti apmokymą. Apmokymui turi būti paruošti duomenys, kurie yra priskirti vienai iš dviejų kategorijų. Mokinimosi algoritmo tikslas atpažinti kategoriją alegorizuojančius faktus. Vektoriais grįstas analizavimo algoritmas gali būti atvaizduotas n-matėje plokštumoje, kur kiekvienas taškas atitinka tam tikrą grupę. Kiekvienas žodis arba rasta seka yra priskiriama vienai iš dviejų konkrečių grupių. Taip atskiros teksto dalys yra padalinamos, bei priskiriama labiausiai tikėtina reikšmė. Vėliau tarp šių grupių stengiamasi nubrėžti aiškią tiesę, kuri atskirtų grupes su didžiausiu atstumu 2.3 paveikslėlis. Paveikslėlyje pavaizduoti du galimi variantai atskirti grupę. Kadangi $w_2 > w_1$, bus pasirinkta antra linija. Baigus pirmąją mokinimosi fazę, bei pateikiant sistemą naudoti vartotojams rastas žodis bus priskirtas jau esamai plokštumai, o tai į kurią linijos pusę, dalinančią plokštumą jis papuls nuspręs koks raktažodis yra tinkamesnis.



2.3pav. Atraminių vektorių klasifikatoriaus ribos pasirinkimas

Atraminių vektorių klasifikatoriaus algoritmas yra pagrįstas neuroninių tinklų idėjomis. Šis algoritmas tiksliai gali tiksliai analizuoti tekstus, tačiau iš anksto turi žinoti kokius įmanoma rasti rezultatus. Tai reiškia, kad prieš pradedant analizuoti tekstus reikia žinoti visus įmanomus rasti raktažodžius, visuose tekstuose. Dėl šio trūkumo vektoriais grįstų mašinų algoritmas nėra tinkamas analizuoti nestandartizuotus, įvairaus turinio, bei internetinius dokumentus.

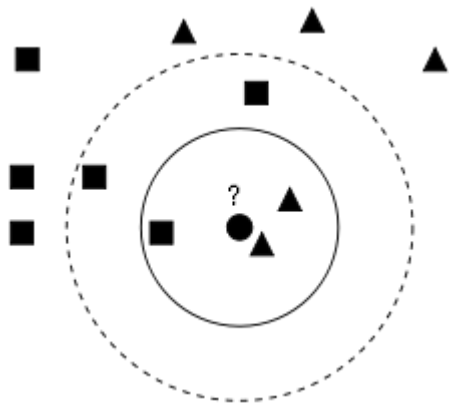
2.6.4. k arčiausio kaimyno algoritmas

Vienas iš paprasčiausių būdų kaip analizuoti tekstą yra tiesiog įsiminti visus duomenis pirmojo apmokymo etapo metu, tuomet vykdant analizę išskirti tik tuos raktažodžius, kurie tiksliai atitinka visus apmokymo duomenis. Toks būdas turi akivaizdžių trūkumų. Tokiu būdu analizuojant tekstą, paprasčiausiai, nebus pasiektas norimas raktažodžių radimo tikslumas. Nes neįmanoma apmokyti algoritmo visais įmanomais tekstais. Daug išmanesnis yra k arčiausio kaimyno (angl. „k nearest neighbor“) algoritmas [25], kuris išskiria iš analizuojamo teksto k elementų grupę, kuriai priskiria dominuojantį raktažodį [16]. Prieš pradėdamas dirbti algoritmas turi būti apmokytas. Algoritmui paduodamos žodžių grupės su jau priskirtais dominuojančiais raktažodžiais.

Realioje aplinkoje, atliekant pasirinkimą algoritmo pagrindą sudaro trys pagrindinės dalys:

- k reikšmė – tai laisvai pasirenkamas dydis, kuris maksimalų leistiną panašumą tarp analizuojamų grupių, bei galimų raktažodžių,
- panašumas tarp grupių ir galimų raktažodžių. Egzistuoja skirtingi būdai skaičiuoti, bei išreikšti panašumui,
- analizuojama grupė, kuriai norima priskirti raktažodį.

Grafinis galimas algoritmo pasirinkimas pavaizduotas 2.4 paveikslėlyje. Jei norima priskirti reikšmę pavaizduotam apskritimui centre, reikšmė priklausys nuo pasirinkto dydžio k. Jei k yra 1, pavaizduotas ištisinės linijos apskritimas, tuomet bus pasirinktas raktažodis žymimas trikampiu, nes pagal pasirinktą atstumą apskritime yra 2 trikampiai ir tik vienas keturkampis. Jei k reikšmė yra 2, pavaizduotas brūkšninės linijos apskritimas. Tuomet bus pasirinktas raktažodis apibūdinamas kvadratu, nes naudojant šį atstumą yra 3 kvadratai, bei 2 apskritimai.



2.4 pav. k-arčiausio kaimyno galimas pasirinkimas

2.6.5. Entropija grįstas algoritmas

Entropija grįstas algoritmas naudoja principą, jog visi arba dalis faktų priskiriamų vienai kategorijai yra susiję su kita kategorija. Algoritmas yra paremtas maksimalios entropijos teorija. Iš visų modelių, kurie buvo pateikti algoritmui apmokyti, yra pasirenkamas tas, kurio entropija yra didžiausia.

Entropijos metodas yra naudojamas, kai nėra žinoma visa informacija apie analizuojamus duomenis, arba kai nėra saugu, tikslu priimti kokias nors prielaidas. Taip pat kai negalima daryti prielaidos, jog duomenys yra nesusiję. Tai ypač tinka teksto analizavimui.

Algoritmas pirmiausia sudalina tekstą į atskirus sakinius (laikoma, kad sakiniai yra skiriami ženklas: („.“, „?”, „!”). Tuomet atskiruose sakiniuose yra grupuojami žodžiai. Eksperimentai parodė, jog geriausia grupę sudaryti nuo vieno iki keturių žodžių. Taip sugrupuojami visi žodžiai w. Iš visų junginių g, yra atrenkama trisdešimt procentų dažniausiai pasikartojančių žodžių junginių. Atrinkti junginiai laikomi geriausiais kandidatais G. Skaičiavimo laikui sumažinti toliau yra naudojamos tik atrinkti junginiai.

Tuomet kiekvienam žodžiui yra skaičiuojamas dydis X[17]:

$$X(w)^2 = \sum_{g \in G} \frac{(freq(w,g) - n_w p_g)^2}{n_w p_g} \qquad (6)$$

$freq(w,g)$ - pakartotinio pasikartojimo dažnumo funkcija,

n_w - kaip dažnai žodis w pasikartojo su geriausiu raktažodžiu g,

p_g - kaip dažnai geriausias raktažodis g pasikartoja tekste.

Apskaičiavus dydį X, tikslumui pagerinti junginiai yra apjungiami:

- pagal panašumą: jeigu junginiai w_1 ir w_2 kartu tekste kartojasi panašiai kaip ir kiti junginiai, jie yra laikomi vienoje grupėje,
- pagal pakartotinį pasikartojimą, jeigu junginiai w_1 ir w_2 tekste pakartotinai kartojasi panašiai, jie yra laikomi toje pačioje grupėje.

Atrinkus raktažodžius ir suskirsčius juos į grupes yra atliekamas rezultatų pasirinkimo žingsnis. Pasirenkami tie junginiai, kurie turi didesnį dydį X, bei dažnai kartojasi tekste. Taip pat pasirenkant jei yra žodžių su panašiais svoriais, bus pasirinkti tie žodžiais, kuriems didesnį reitingą suteikė vartotojas.

Algoritmo žingsniai:

1. Suskaidyti tekstą į sakinius,
2. Suskaidyti tekstą į žodžių junginius,
3. Išskirti kiekvieno žodžio kamieną naudojant Porter [12] algoritmą,
4. Atrinkti iki trisdešimties procentų dažniausiai pasikartojančių junginių,
5. Apskaičiuoti dydį X pasirinktiems junginiams,
6. Apjungti junginius,
7. Pasirinkti raktažodžius.

2.6.6. Sąrašo algoritmas

Anksčiau nagrinėti algoritmai turi teisę laisvai pasirinkti raktažodžius vartotojo analizuojamiems dokumentams. Sąrašu grįstas algoritmas analizuojamam tekstui parinks raktažodžius tik iš anksto nustatyto sąrašo. Algoritmui iš anksto yra pateikiamas sąrašas žodžių, kurie gali būti naudojami kaip raktažodžiai. Tuomet iš visų pateiktų žodžių yra pasirenkami tie, kurie geriausiai tinka tekstui. Šis sprendimas leidžia sistemoje turėti tik norimus raktažodžius. Akivaizdus galimas minusas, jog iš anksto reikia žinoti visus galimus raktažodžius. Norint pridėti arba pašalinti raktažodį, gali tekti iš naujo perindeksuoti visą sistemą.

Algoritmo žingsniai:

1. Suskaidyti tekstą į žodžius,
2. Išskirti kiekvieno žodžio kamieną naudojant Porter [12] algoritmą,
3. Apskaičiuoti žodžio miglotąją funkciją dažnumui tekste,
4. Apskaičiuoti žodžio miglotąją funkciją dažnumui ir pakartojantį pasikartojimą naudojant pateiktą žodžių sąrašą,
5. Apskaičiuoti dydį X,
6. Pasirinkti raktažodžius.

Kiekvienam žodžiui yra skaičiuojama „Takagi-Sugeno (Type-3)"[18] taisyklė rasti dydį X:

$$X(w) = \lambda(w) * (1 - \mu(w)) * \eta(w) \qquad (7)$$

λ – miglotoji funkcija nusakanti žodžio dažnumą tekste,
μ – miglotoji funkcija nusakanti pasikartojimų dažnumą kalboje,
η – miglotoji funkcija nusakanti pakartotinį pasikartojimą tekste.

Algoritmas parinks raktažodžius tik iš anksto paruošto sąrašo, todėl didelis algoritmo tikslumas priklauso nuo gerai paruošto sąrašo, bei teisingų duomenų parinktų algoritmo apmokymui.

Rezultatų pasirinkimą įtakoja dydis X, bei kaip dažnai žodis pasikartojo tekste. Didesnė X reikšmė parodo sąrašo žodžio tikslumą tekstui, o didesnis pasikartojimas, jog žodis yra svarbus tekste.

2.7. IŠVADOS

Pateikiant internetinius dokumentus, dažnai galima pateikti ir duomenų metaduomenis. Metaduomenys apibūdina pateikiamus duomenis, tai gali būti raktažodžiai, kategorijos, faktai. Pateikti metaduomenys gali būti naudojami sistemos atliekant paiešką, siūlant susijusį turinį kitiems sistemos vartotojams. Norint, kad pateikiami metaduomenys būtų tikslūs galima automatizuoti šį procesą, tam naudojant automatinių dokumentų analizavimą. Žiniatinklio dokumentai yra pateikiami naudojant interneto naršyklę. Panašių sistemų analizė rodo, jog patogiausia žiniatinklio dokumentus analizuoti naudojant tą patį įrankį, interneto naršyklę, kuriuo ir yra pateikiami patys dokumentai. Taip vartotojui nereikia įdiegti papildomų programų, analizavimo sistema prieinama naudojant skirtingus įrenginius, bei turi vartotojo analizavimo istorija.

Vienas iš paprasčiausių, dažniu grįstas, algoritmas yra naudojamas greitai, nesudėtingai pateikti galimą dokumento dažnumo analizę. Algoritmas gali būti nesudėtingai papildytas tikslesniems rezultatams gauti. Analizuoti sudėtingesni algoritmai, remiantys mašininiu mokinimusi reikalauja daugiau resursų analizuojant tekstus. Prieš naudojant realioje aplinkoje, juos reikia papildomai apmokyti. Taip pat yra naudojami žodynai, atskirti, bei analizuoti sinonimus.

Norint paspartinti sudėtingą teksto analizavimo procesą yra naudojamos euristikos, nustatomi minimalios analizavimo parametrų ribos, iš teksto yra pašalinami nereikšmingi žodžiai. Sistema gali analizuoti įvairius dokumentus, tačiau dažniausiai vartotojai kuria dokumentus tik vienai ar kelioms specifinėms sritims. Todėl sistemą apmokant konkretaus vartotojo analizuojamais dokumentais galima pritaikyti individualiems vartotojams.

3. PROJEKTINĖ DALIS

Projekto metu realizuota žiniatinklio dokumentų automatinio žymėjimo sistema. Sukurta sistema leidžia vartotojams naudojant interneto naršyklę pasirinkti norimą analizuoti tekstą, bei pateikti jį analizavimui. Vartotojas gali peržiūrėti analizavimo algoritmus. Sistema kaupia duomenis apie individualaus vartotojo analizuojamus dokumentus, rastus raktažodžius, blokuojamus raktažodžius.

Kiekvienas vartotojas peržiūrėdamas sistemoje rastus jo analizuotų dokumentų raktažodžius, gali priskirti raktažodžiams reitingą. Priskirtas reitingas įtakoja rezultatų pasirinkimą.

Blokuojami raktažodžiai yra skirti individualiems vartotojams nurodyti sistemai, jog konkretūs žodžiai yra neinformatyvūs, netinkami būti raktažodžiais. Šie žodžiai nebus pateikiami kaip raktažodžiai.

Pirmoji sistemos versija yra skirta anglų kalbai bei „Google Chrome“ interneto naršyklei. Tačiau sistema projektuota taip, kad būtų galimas pakartotinis elementų panaudojimas ir plėtimas. Todėl surinkus didelį informacijos kiekį apmokyti algoritmams, bei panaudojant kitų kalbų žodynus, sistemą galima pritaikyti kitų kalbų tekstams analizuoti.

3.1. ARCHITEKTŪROS TIKSLAI IR APRIBOJIMAI

Kuriamos sistemos tikslas sukurti bendrą platformą kurią būtų galima panaudoti skirtinguose projektuose. Pirmojoje realizuojamoje versijoje kuriamas tik interneto naršyklės papildinys leidžiantis vartotojams analizuoti tekstus. Bendravimas yra vykdomas naudojant kliento-serverio architektūrą. Tačiau kuriant sistemos architektūrą didelis dėmesys yra skiriamas jos plėtimui ateityje. Keliamas tikslas sukurti nesunkiai plečiamą sistemą, kuri esant pasisekimui galėtų apdoroti didelį klientų skaičių. Sistema realizuojama komponentais, kurie bendrauja tik naudodami sąsajas, bei nieko nežino apie vienas kito įgyvendinimą. Didelis dėmesys skiriamas išeities kodo kokybei, bei būsimam sistemos palaikymui. Kuriamai sistemai iš anksto nustatomi keliami reikalavimai architektūrai, išvaizdai, saugumui. Jie yra aptariami toliau.

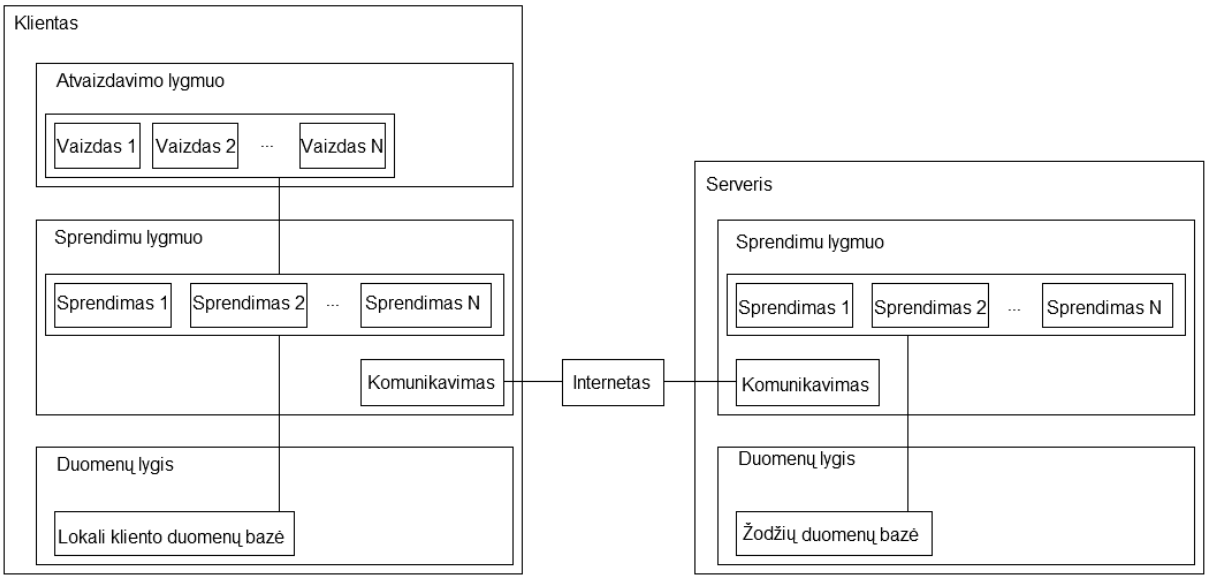
3.1.1. Sistemos architektūriniai apribojimai

- Sistema remiasi trijų lygmenų architektūra. Pasirinkta kurti kelių lygių sistemą, dėl paprastesnio galimo sistemos plėtimo. Sistemą sudaro:
    - atvaizdavimo lygmuo (kol kas tik kliento dalyje) – atsakingas už informacijos atvaizdavimą vartotojui, bei komunikavimą su nutolusiu serveriu.
    - sprendimų lygmuo – atsakingas už vartotojo užklausų apdorojimą, verslo taisyklių įgyvendinimą, teksto analizavimą.
    - duomenų lygmuo – atsakingas už duomenų saugojimą, bei paėmimą iš duomenų bazės.
- Sistema remiasi sprendimų izoliavimo, bei paskirstymo metodiką. Sistemos yra suskaidyta į atskirus elementai, kurie tarpusavyje yra mažai priklausomi. Tokia sistemos struktūra pasiekiama tokiu būdu:
    - atskiroms problemos spręsti yra sukuriami skirtingi komponentai.
    - komponentų bendravimui naudojamos tik sąsajos. Skirtingi komponentai nieko nežino apie vienas kito įgyvendinimą.
    - atskiri sprendimai sujungiami tik aplikacijos lygyje, nebent sprendimui gauti reikalingas kitas bendrinio pobūdžio sprendimas.
- Realizuojami algoritmai įgyvendinami kaip sistemos komponentai, paveldėdami bazinę abstrakčią klasę.
- Naudojami trečios šalies sprendimai, turi būti integruojami per papildomą abstrakcijos lygmenį. Turi būti paprasta pakeisti komponentus pasikeitus reikalavimams arba atnaujinti komponentų versijas.
- Sistema turi laikytis pagrindinių žiniatinklio aplikacijų principų.
- Sistema turi turėti galimybę būti išversta į kitas kalbas.
- Norėdama prisitaikyti prie vartotojo, sistema saugo informaciją apie jos vartotojos, todėl:
    - Slaptažodžiai saugomi šifruoti, naudojant vienpusę kriptografiją.
    - Būtina riboti netinkamų prisijungimų skaičių.
    - Saugoti įrašus apie vartotojo prisijungimus.
    - Sistema turi būti saugi nuo klasikinių interneto svetainių pažeidžiamumų.

- Turi būti įgyvendinta duomenų prieigos kontrolė.

Sistemos architektūrą pavaizduotą 3.1 paveikslėlyje, ą sudaro vaizdai, sprendimai, baziniai elementai.

Sprendimų išskaidymas atskirais komponentais leidžia juos keisti, pildyti ir pernaudoti atskirai nuo visos sistemos. Komponentų bendravimas naudojant sąsajas leidžia neprisirišti prie konkretaus įgyvendinimo ir pasikeitus reikalavimams pakeisti vieną komponentą kitu. Bendras bazinių klasių buvimas nurodo minimalius reikalavimus komponentams.

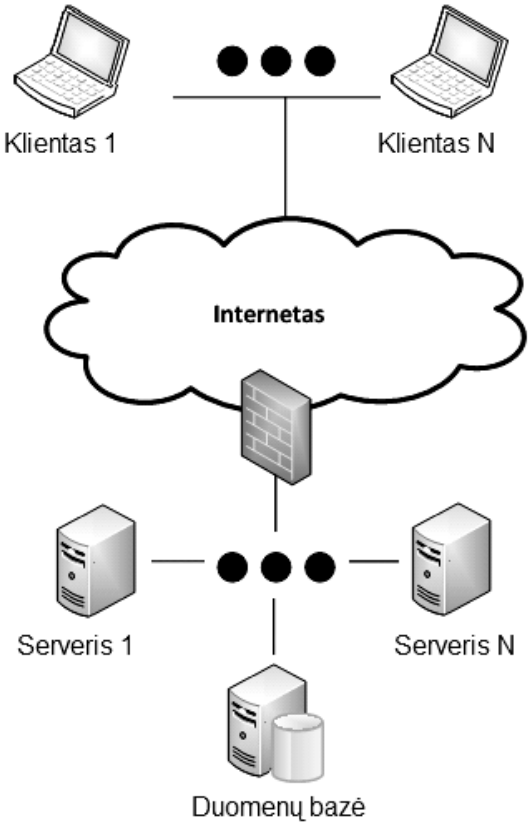


3.1 pav. Abstrakti sistemos architektūra.

3.1.2. Išvaizdos apribojimai

- Turėtų įprastą duomenų įvedimo formą.
- Turėtų patogią navigaciją.
- Daugumai intuityvi, lengvai suprantama vartotojo sąsają,
- Išnaudoti spalvinio kodavimo galimybes.
- Sukurta vartotojo sąsaja turi neklaidinti vartotojo. Kuriamas interneto naršyklės papildinys turi laikytis bendrų naršyklės papildinio reikalavimų.
- Funkcionalumas turi būti kiek galima aiškesnis ir intuityvesnis. Naudojamos grafinio tipo priemonės turi neklaidinti vartotojo.
- Paprasta naudojimas, vartotojas turi aiškiai suprasti sistemos funkcijas.

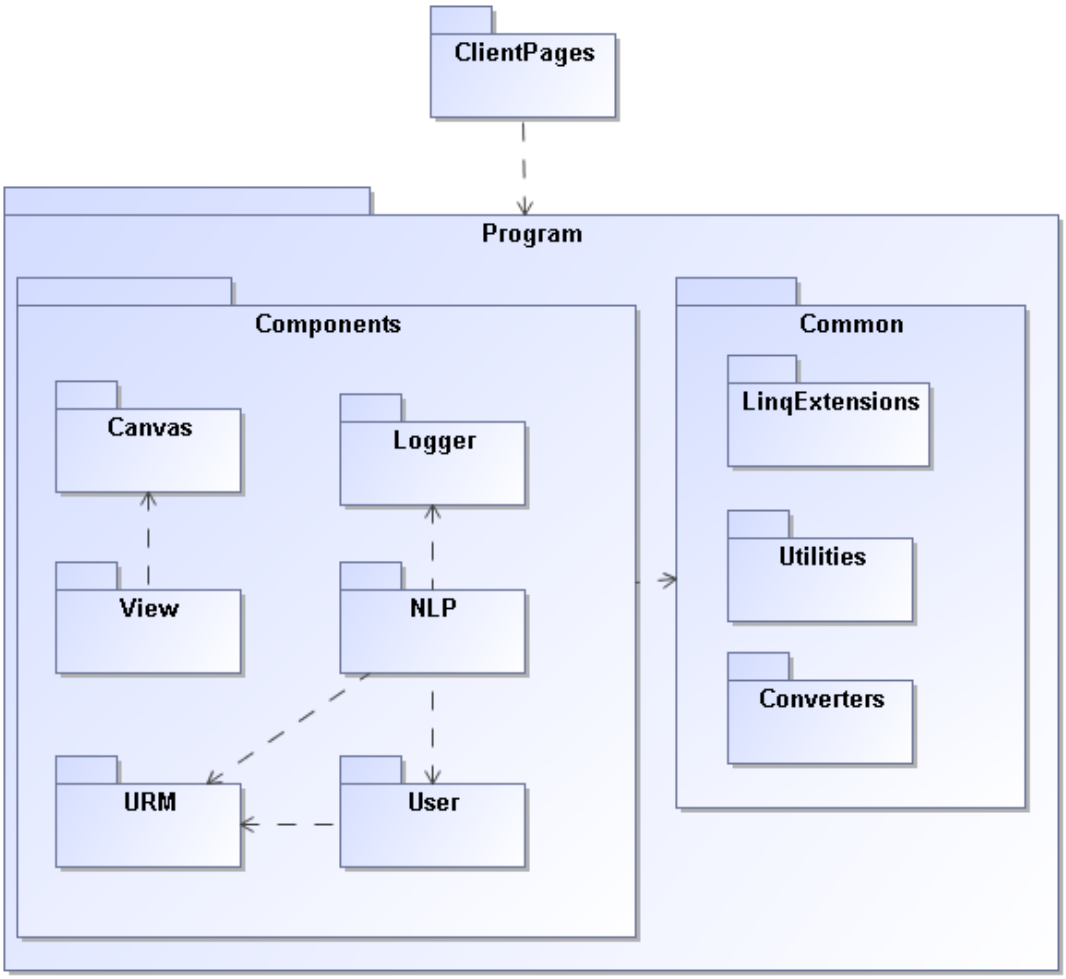
3.2. SISTEMOS IŠDĖSTYMO VAIZDAS



3.2 pav. Aukšto lygio sistemos išdėstymo vaizdas

Paveikslėlyje 3.2 pavaizduota aukšto lygio sistemos išdėstymo vaizdas. Išdėstymas leidžia plėsti sistemą esant didelėms apkrovoms. Sistemos plėtimui naudojama klasikinė išdėstymo schema. Plečiant naudojimą, pridedant naujus serverius, kodas replikuojamas kiekviename serveryje. Kiekvienas serveris turi informaciją kaip prisijungti prie duomenų bazės, bei turi pilną kompiliuotą išeities kodą. Nors sistemos pagrindinis tikslas aptarnauti žiniatinklio klientus, priklausomai nuo poreikio sistema gali veikti ir intranete. Sistemos apsaugai naudojama ugniasienė.

3.3. SISTEMOS DINAMINIS VAIZDAS



3.3 pav. Bendrą sistemą sudarantys paketai

Sistema sudaryta iš paketų, pavaizduotų 3.3 paveikslėlyje. Bendrą sistemą sudaro šie paketai:

- Components – bendri komponentai:
    - Canvas – grafikų atvaizdavimo komponentas realizuotas pritaikant HTML5 „Canvas“ elementui.
    - User – komponentas apibendriną reikalingas funkcijas atliekant vartotojų valdymo funkcionalumą.
    - NLP – teksto analizavimo komponentas.
    - View – komponentas skirtas atvaizduoti duomenims.
    - URM – komponentas yra atsakingas už duomenų priėjimo teisių priskyrimą, bei valdymą

    - Logger – komponentas skirtas registruoti sistemoje vykstančius veiksmus, svarbius įvykius
- Common – baziniai sistemos komponentai:
    - LinqExtensions – įvairių “Linq” funkcijų praplėtimai paprastesniam naudojimui.
    - Utilities – dažnai pasitaikančių funkcijų ir klasių paketas.
    - Converters – įvairių klasių konvertavimo pagalbinės funkcijos.
- ClientPages – tai komponento dalis, kuri naudoja sukurtą sistemą

3.4. TREČIŲ ŠALIŲ SPECIALIZUOTI PROGRAMŲ PAKETAI

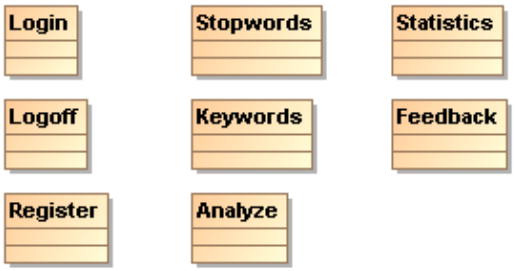
Sistemos realizacijai naudojama trečiųjų šalių komponentai:

- jQuery
- jQuery UI
- Bootstrap

Šie komponentai panaudoti greitesniam realizavimui ir sistemos standartizavimui, neperkuriant savo elementų, o panaudojant esamus ištestuotus ir vystomus.

3.5. SVARBESNIŲ PAKETŲ DETALIZAVIMAS

3.5.1. ClientPages paketo detalizavimas



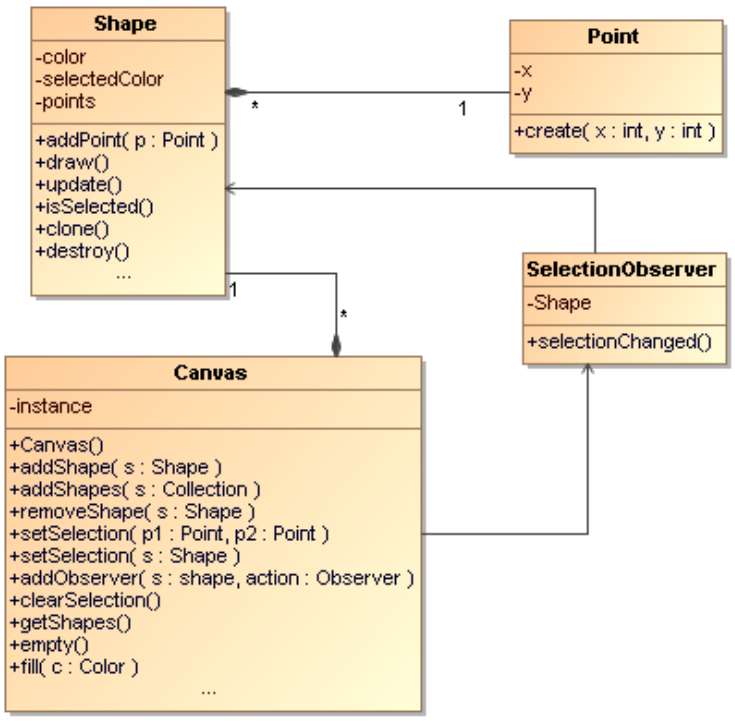
3.4 pav. ClientPages paketo diagrama

Skirtas apvaizduoti sistemos puslapius. Puslapiai atsakingi už:

- Login – vartotojo prisijungimas.
- Logoff – vartotojo atsijungimas.
- Register – vartotojo registracija.
- Analyze – teksto pateikimas analizavimo, bei gautų rezultatų peržiūra.

- Keywords – individualaus vartotojo raktažodžių peržiūra, reitingavimas, pridėjimas, bei pašalinimas
- StopWords – individualaus vartotojo nepageidaujamų žodžių peržiūra, pridėjimas, bei pašalinimas
- Statistics – individualaus vartotojo analizavimo statistikos peržiūra.
- Feedback – galimybė parašyti apie sistemos darbą, palikti atsiliepimą.

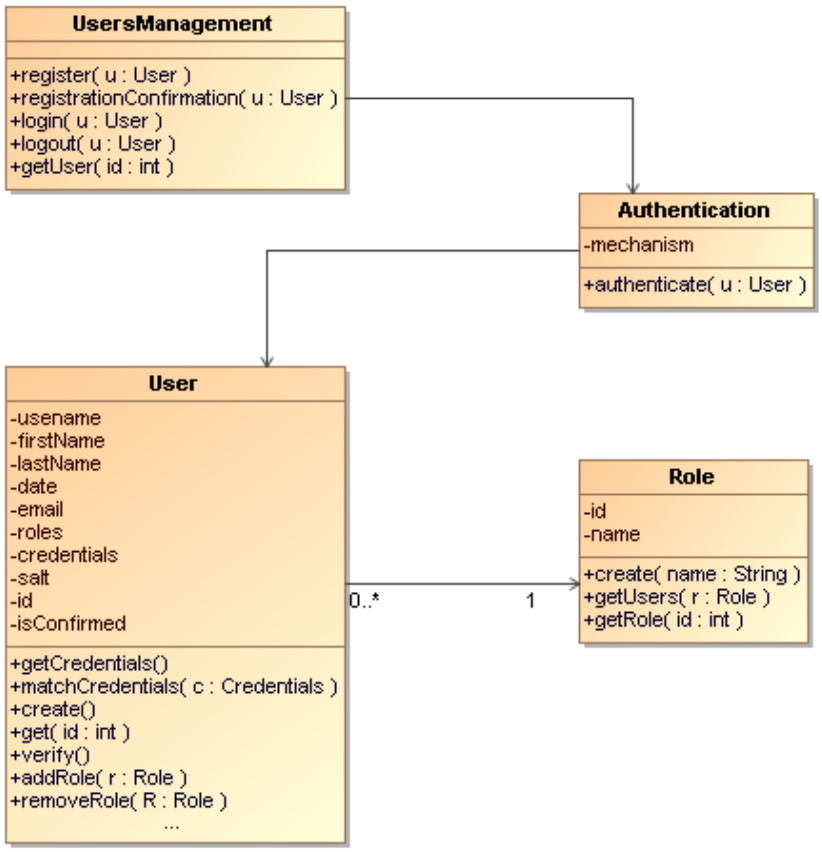
3.5.2. Components.Canvas



3.5 pav. „Components.Canvas“ paketo detalizuota klasių diagrama

Komponentas skirtas pateikti rezultatus grafiniu būdu, piešti interaktyvias diagramas. Egzistuoja pagrindinis langas, į kurį yra talpinamos figūros. Figūros yra apibūdinamos erdvėje išsidėsčiusiais taškais. Plečiant figūras galima išplėsti komponentą.

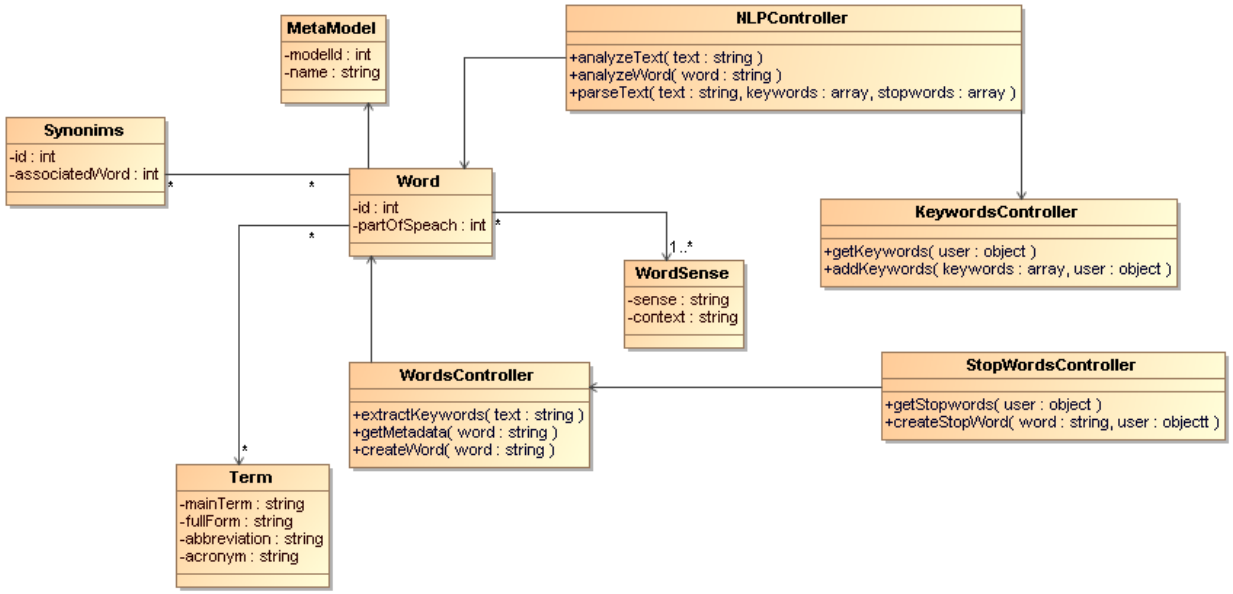
3.5.3. Components.User



3.6 pav. „Components.User“ paketo detalizuota klasių diagrama

Šis komponentas apima apibendrintą vartotojų valdymo funkcionalumą. Komponentas atsakingas už vartotojo sukūrimą, registraciją, bei prisijungimą, autentifikavimą. Komponentas gali būti integruotas į kitas sistemas kurį galima pakartotinai panaudoti keliose sistemose.

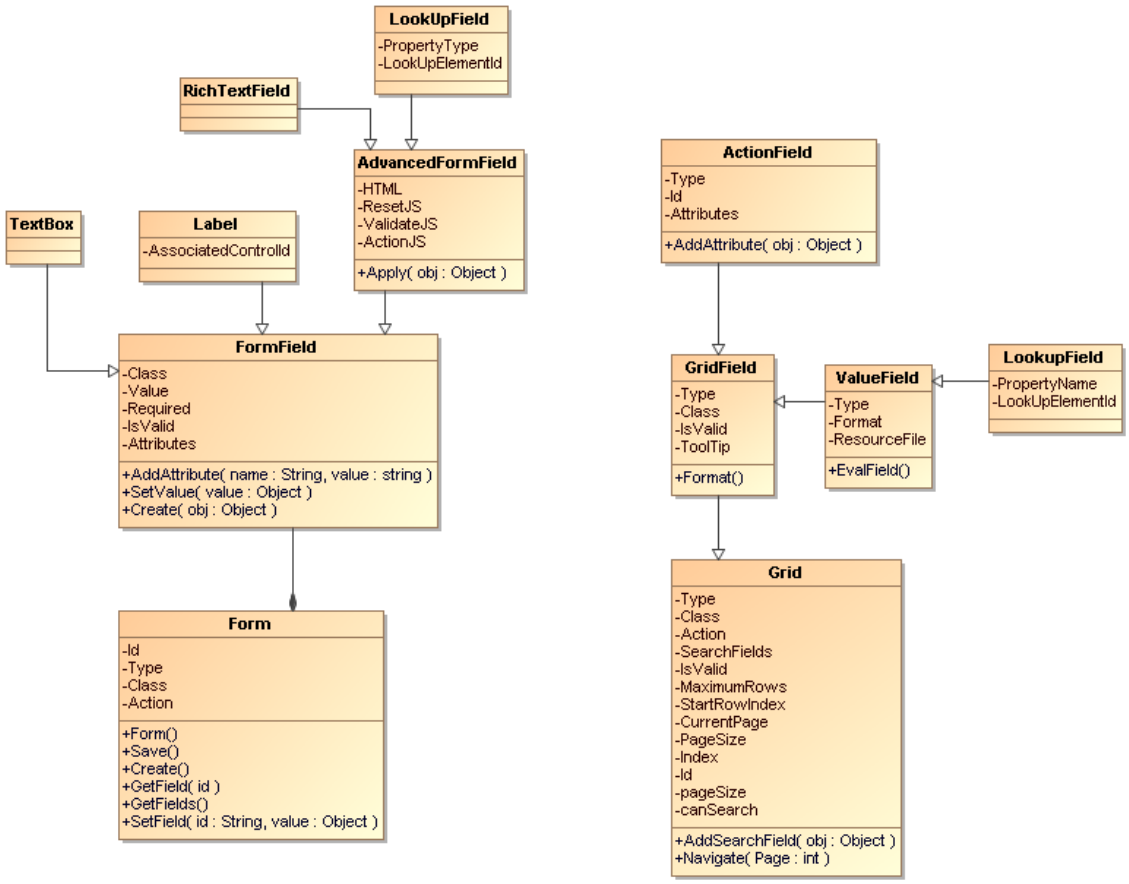
3.5.4. Components.NLP



3.7 pav. „Components.NLP“ paketo detalizuota klasių diagrama

Natūraliosios kalbos analizavimo komponentas, skirtas nagrinėti pateiktus tekstus, bei žodžius, priskirti jiems meta duomenis, atributus.

3.5.5. Components.View



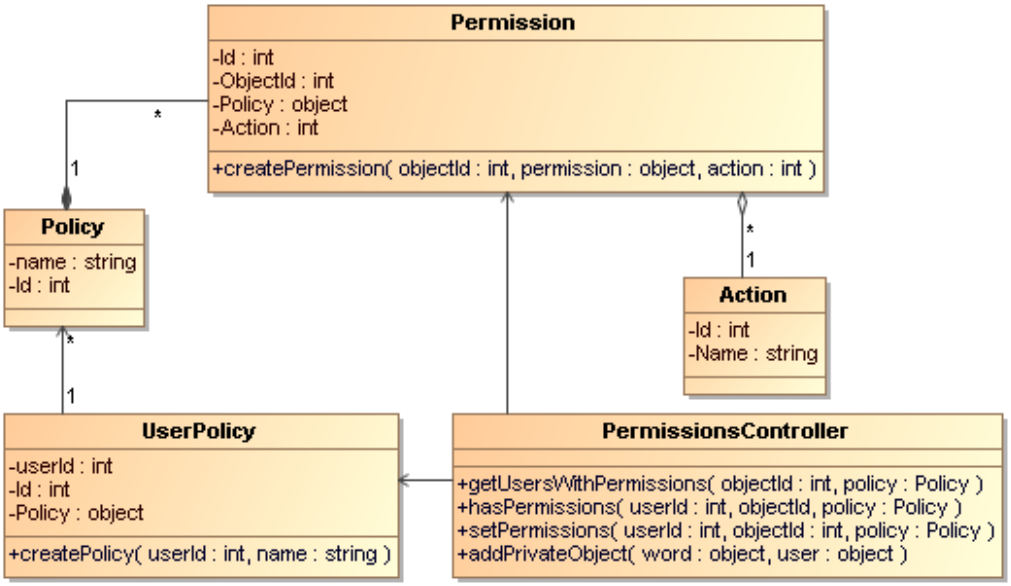
3.8 pav. „Components.View“ paketo detalizuota klasių diagrama

Komponentas skirtas atvaizduoti duomenims. Jame yra pagrindinės dvi dalis:

1. Komponentus skirtus valdymui –„Form“ ir „Grid“;
2. Komponentus skirtus aprašyti ir apdoroti duomenų laukus – visi kiti likę komponentai.

Šioje vietoje yra palikta didelė plėtimo laisvė, esant poreikiui įgyvendinus abstrakčius metodus galimas nesunkiai papildyti esamą funkcionalumą.

3.5.6. Components.URM



3.9 pav. „Components.URM“ paketo detalizuota klasių diagrama

„URM“ komponentas yra atsakingas už elementų priėjimo teisių valdymą. Šio paketo funkcionalumas apima teisių suteikimą ir vartotojų valdymą priėjimo teisių požiūriu.

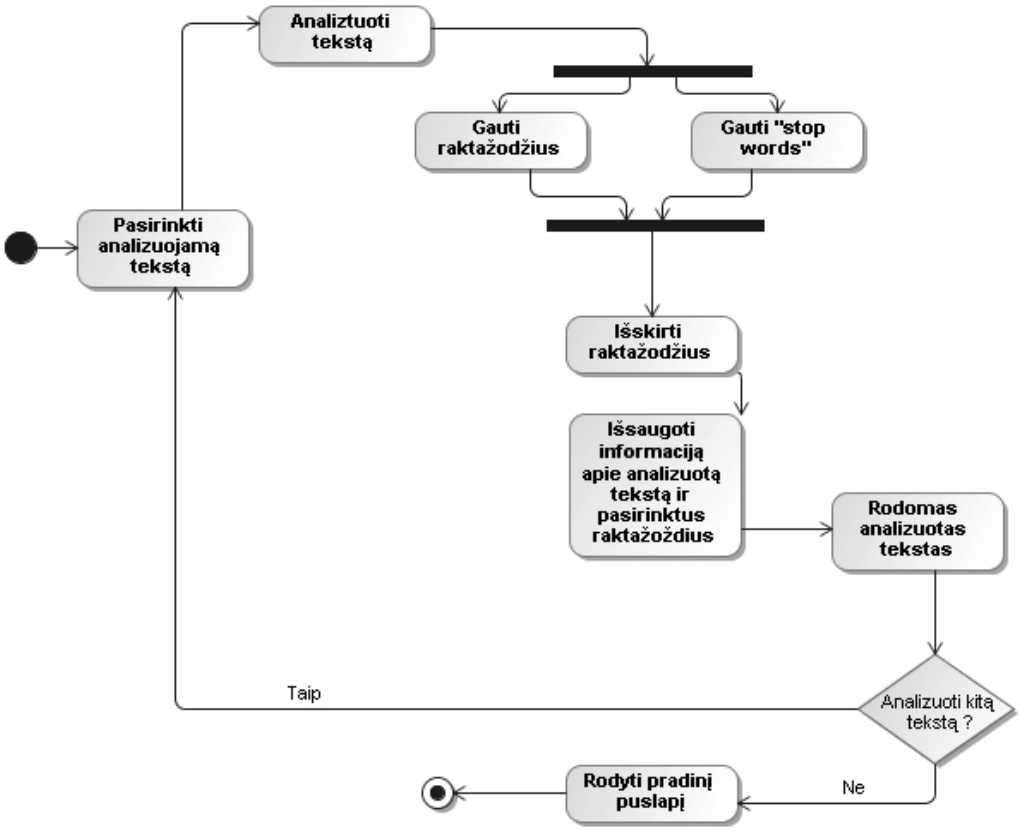
3.6. SISTEMOS VEIKLOS DIAGRAMA

3.6.1. Teksto pateikimas analizei

Teksto analizavimo procesas gali prasidėti dviem būdais:

1. Vartotojas apsilankęs analizavimo puslapyje įkopijuoja norimą analizuoti tekstą, bei spaudžia analizavimo mygtuką,
2. Vartotojas pelės pagalba pažymi norimą analizuoti tekstą interneto naršyklėje, bei iš kontekstinio menu pasirenka teksto analizavimą.

Sistema priima norima analizuoti tekstą, tuomet gauna vartotojo jau analizuotų tekstų raktažodžius, bei nepageidaujamus žodžius. Abu šie veiksmai yra nepriklausomi, todėl gali būti vykdomi lygiagrečiai. Surinkus visą pradinę informaciją algoritmas analizuoja tekstą. Rasti raktažodžiai yra išsaugojami duomenų bazėje, bei gražinami vartotojui.



3.10 pav. Veiklos diagrama. Pateikti tekstą analizei

3.7. PASIRINKTOS TECHNOLOGIJOS

Kuriamas interneto naršyklės papildinys skirtas „Google Chrome“ interneto naršyklei. Ši naršyklė pasirinkta dėl kelių priežasčių:

1. populiarumo;
2. naujų technologijų palaikymo;

Kuriant papildinį bus naudojamos interneto technologijos:

- HTML kalba;
- JavaScript kalba;
- CSS stiliai.

Serveryje bus naudojamasi pagrinde „Microsoft“ kuriamomis ir palaikomomis technologijomis. Jos pasirinktos, nes:

1. labai greitas projektų įgyvendinimas;
2. turima darbo patirtis su šiomis technologijomis.

Naudojamos tokios „Microsoft“ technologijos:

1. Windows Server 2008 R2;
2. SQL Server Express;
3. C#;
4. ASP.NET;

5. Internet Information Services 7.5.

3.8. IŠVADOS

Pasirinkta komponentų architektūra, kurios esminės dalys yra teksto analizavimas, klientų prieigos kontrolė, naujomis technologijomis grįstas duomenų pateikimas. Pasirinkta žiniatinklio automatinio dokumentų žymėjimo architektūra yra tinkama keliamiems reikalavimams, ji leis sukurti mažai jungią sistemą, kurią esant poreikiui bus galima plėsti. Esant poreikiui, bei dideliam sistemos apkrautumui, sistemą galima išplėsti.

Pasirinktas sprendimas kurti interneto naršyklės papildinį leis vartotojui iš bet kurios vietos, patogiai, pasiekti sistemą. Vartotojui nebereikės papildomai įdiegti programinės įrangos.

4. ŽINIATINKLIO DOKUMENTŲ AUTOMATINIO ŽYMĖJIMO EKSPERIMENTINIS ALGORITMŲ TYRIMAS

4.1. TIKSLAI

Eksperimento tikslas yra nustatyti žiniatinklio dokumentų automatinio žymėjimo algoritmų efektyvumą, tiriamų algoritmų panaudojimą automatiniame teksto žymėjime. Nors sistemos tikslas yra kuo tiksliau pažymėti analizuojamą dokumentą, analizavimo algoritmai bus naudojami individualių žiniatinklio naudotojų. Todėl siekiam nustatyti, ar analizuojami algoritmai gali išskirti tokius raktažodžius, kuriuos natūraliai pasirinktų vartotojas.

Eksperimentas atliekamas su žiniatinklyje rastais kitų autorių tekstais suskirstytais į penkias kategorijas:

1. technologijos,
2. gamta,
3. karjera,
4. mokslas,
5. kulinarija.

Visi analizuojami tekstai jau yra pažymėti autorių. Tai yra, jiems yra priskirti meta-duomenys apibūdinantys tekstus. Eksperimento metu taip, pat norima nustatyti kokią įtaką daro analizuojamo teksto ilgis. Todėl analizuojami tekstai taip pat yra suskirstyti į skirtingo teksto ilgio grupes:

1. trumpi – mažiau nei 150 žodžių,
2. vidutiniai – nuo 150 iki 600 žodžių,
3. ilgi – virš 600 žodžių.

Trečioji charakteristika yra randamų raktažodžių kiekio pasirinkimas. Analizuojami tekstai, kuriems autoriai jau yra priskyrę nevienodą skaičių raktažodžių. Atliekant eksperimentą yra naudojami du raktažodžių parinkimo kriterijai:

1. pasirenkamas fiksuotas skaičius – sistema parenka 5 geriausius raktažodžius apibūdinančius tekstą,
2. raktažodžių skaičius priklauso nuo analizuojamo teksto ilgio. Parenkant raktažodžius tuomet yra naudojama formulė:

$$n = \log(N) + 1 \quad (8)$$

Čia:

- n – parenkamų raktažodžių skaičius,
- N – analizuojamo teksto žodžių skaičius.

Šis būdas leidžia dinamiškai parinkti raktažodžių skaičių. Ilgiems tekstams priskiriama daugiau raktažodžių, nei trumpiems tekstams. Skaičiuojant pridedamas vienas, jog visi analizuojami tekstai turėtų nors vieną raktažodį.

Atliekamo eksperimento tikslas ištirti algoritmų tinkamumą, naudoti automatiniai žiniatinklio dokumentų žymėjimo sistemai. Tiriami algoritmai:

1. teksto dažnumo algoritmas (TF),
2. teksto dažnumo, atvirkštinio dokumentų dažnumo algoritmas(IDF),
3. entropija grįstas algoritmas(Ent),
4. sąrašo algoritmas(List).

4.2. EKSPERIMENTO ATLIKIMAS

Naudojant žiniatinklio duomenis iš duomenų agregavimo svetainių:

4. Delicious [21],
5. BlogFlux [19],
6. Technorati [20],

Pasirinkti tekstai patenkantys į numatytas kategorijas, bei grupes pagal žodžių kiekį tekste. Nuspręstas pasirinkti po 15 tekstų kiekvienai kategorijai. Kiekvienas tekstas yra analizuojamas 4 skirtingais algoritmais, bei 2 skirtingais būdais parinkti raktažodžių kiekį tekste.

Išskiriant žodžių kiekį yra remiamasi šiomis taisyklėmis. Pirmiausia tekstas yra suskirstomas į sakinius. Laikoma, jog sakinius skiria šie skyrybos ženklai: „.“(taškas), „!“, „?“. Tuomet sakiniai yra suskirstomi į žodžius. Laikoma, jog žodžius sakinyje skiria: „ “ (tarpas), „,“ (kablelis), „:“.

Gauti rezultatai yra įvertinami rankiniu būdų.

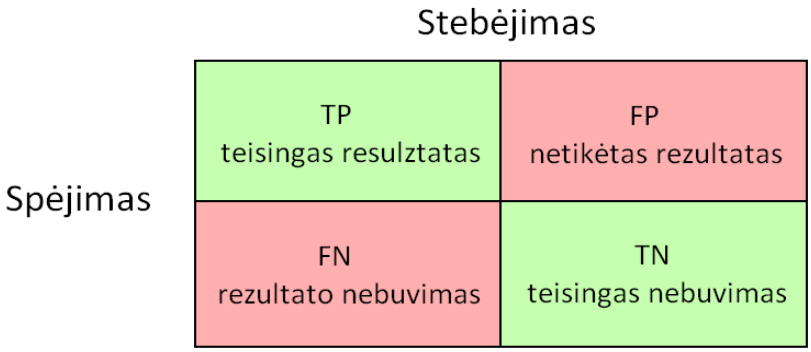
4.3. REZULTATŲ VERTINIMAS

Įvertinant atlikto eksperimento rezultatus yra lyginama kaip tiksliai algoritmų rasti raktažodžiai sutampa su tais raktažodžiais, kuriuos buvo pasirinkę tekstų autoriai. Atliekant rezultatų vertinimą naudojamos standartinės metrikos:

1. tikslumas (angl. „precision”) – tai skaičius, parodantis, kuri dalis iš rastų raktažodžių yra prasmingi, atitinkantys teksto turinį, bei gali būti naudojami apibūdinant analizuojamą tekstą,
2. atkūrimas (anlg. “recall”) – tai skaičius, parodantis, kiek iš vartotojo anksčiau išskirtų raktažodžių, buvo rasta analizuojant tekstą.
3. bendra „F“ metrika – apjungia tikslumą, bei atkūrimą. Šios metrikos tikslas yra gauti bendrą rezultatą apibūdinantį visą sistemą.

Algoritmo didelis atkūrimas parodo, jog algoritmas gali rasti reikalingą informaciją tekste. Atkūrimas taip pat parodo kiekybinę charakteristiką. Didelis tikslumas nurodo, kad rasti duomenys yra teisingi. Tikslumas taip pat parodo kokybinę charakteristiką.

Lyginant stebėjimo, pasirinktų jau žymėtų duomenų, bei spėjimo, atliktos analizės, rezultatus galima išskirti keturias skirtingas grupes, kuriomis gali baigtis eksperimentas. Galimi atvejai, pavaizduoti 4.1 paveikslėlyje.



4.1 pav. Rezultatų eksperimento matrica

Galimi rezultatai:

1. teisingas rezultatas (TP, angl. – „true positive“) – kuomet rastas raktažodis yra tinkamas, bei sutampa su išskirtu vartotojo,
2. teisingas nebuvimas (TN, angl. – „true negative“) – kuomet vartotojo priskirtas raktažodis turi būti nerastas, nes jis nepriklauso tekstui. Šis kriterijus leidžia įvertinti žmogiškąjį faktorių, aptikti neteisingai, per klaidą, pasirinktus autoriaus raktažodžius. Dokumentų algoritmo žymėjimo tikslas kuo tiksliau apibūdinti tekstą,
3. netikėtas rezultatas (FP, angl. – „false positive“) – kuomet žymėjimo algoritmas surado raktažodį, kuris neturėtų būti rastas,
4. rezultato nebuvimas (FN, angl. – „false negative“) – kuomet nebuvo rastas vartotojo išskirtas raktažodis.

Sudarius tokią matricą galima apskaičiuoti kiekvieno nagrinėjamo algoritmo charakteringąsias metrikas:

- tikslumas yra apskaičiuojamas remiantis formule:

$$Tikslumas = \frac{tt}{tt + nt} \tag{9}$$

- atkūrimas yra apskaičiuojamas remiantis formule:

$$Atkūrimas = \frac{tt}{tt + nn} \tag{10}$$

- bendra F metrika yra apskaičiuojama remiantis formule:

$$F = 2 * \frac{Tikslumas * Atkūrimas}{Tikslumas + Atkūrimas} \tag{11}$$

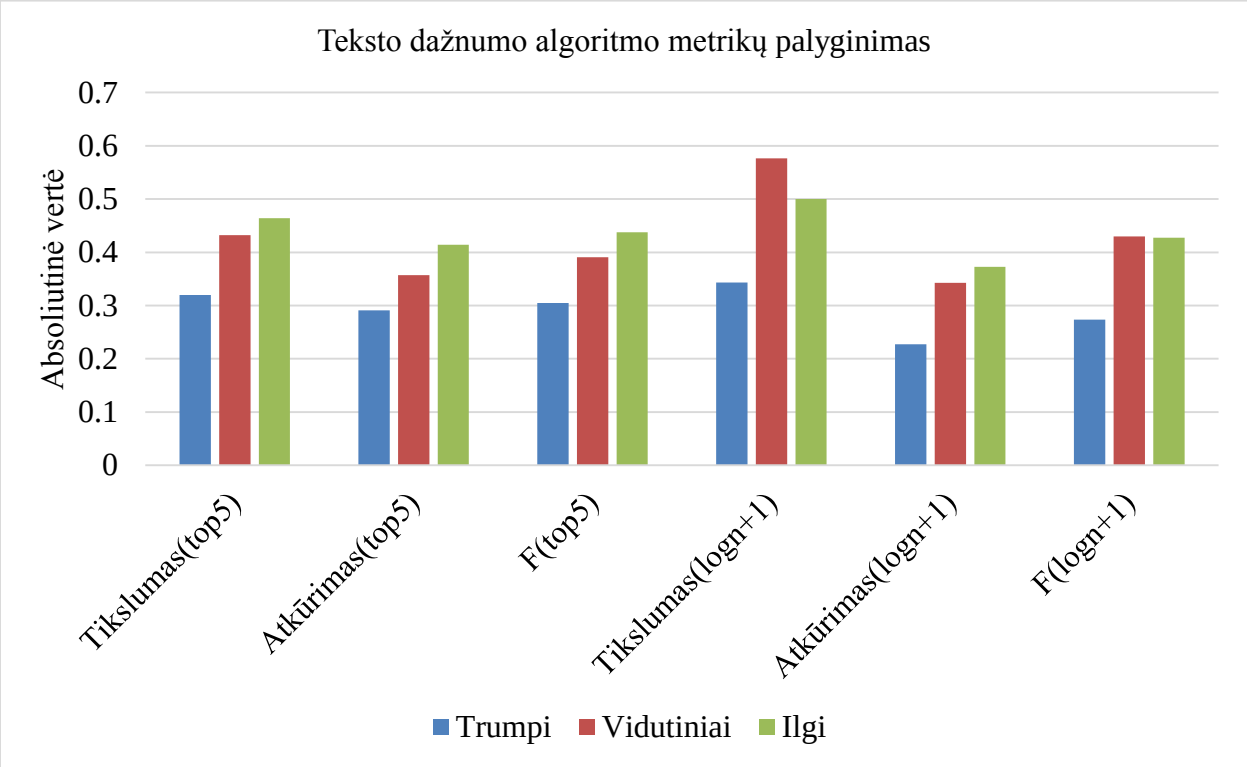
4.4. ANALIZUOTŲ ALGORITMŲ PALYGINIMAS

Skirtingų algoritmų metrikų bendras palyginimas pateikiamas toliau esančiose lentelėse, bei diagramose.

4.4.1. Teksto dažnumo algoritmo rezultatai

4.1 lentelė. Teksto dažnumo algoritmo rezultatai

Rasti(pagal top5)	Tikslumas	Atkūrimas	F	Rasti(pagal logn+1)	Tikslumas	Atkūrimas	F
Trumpi	0.30	0.29	0.30	Trumpi	0.33	0.22	0.26
Vidutiniai	0.43	0.35	0.39	Vidutiniai	0.57	0.34	0.42
Ilgi	0.46	0.41	0.43	Ilgi	0.50	0.37	0.42



4.2 pav. Teksto dažnumo algoritmo rezultatai

Grafikas 4.2 rodo teksto dažnumo algoritmo metrikas analizuojant įvairaus ilgio tekstus. Pateiktas grafikas rodo, didėjant analizuojamo teksto ilgiui didėja ir algoritmo metrikos: tikslumas, bei atkūrimas. Tai tiesiogiai sąlygoje F metrikos didėjimą. Didžiausias šuolis yra gaunamas lyginant trumpų ir vidutinio ilgumo tekstus. Lyginant vidutinio ilgio, bei ilgus matomas geresnis ilgų tekstų analizavimo rezultatas, tačiau lyginant padidėjusių žodžių skaičių, santykinis padidėjimas yra mažas.

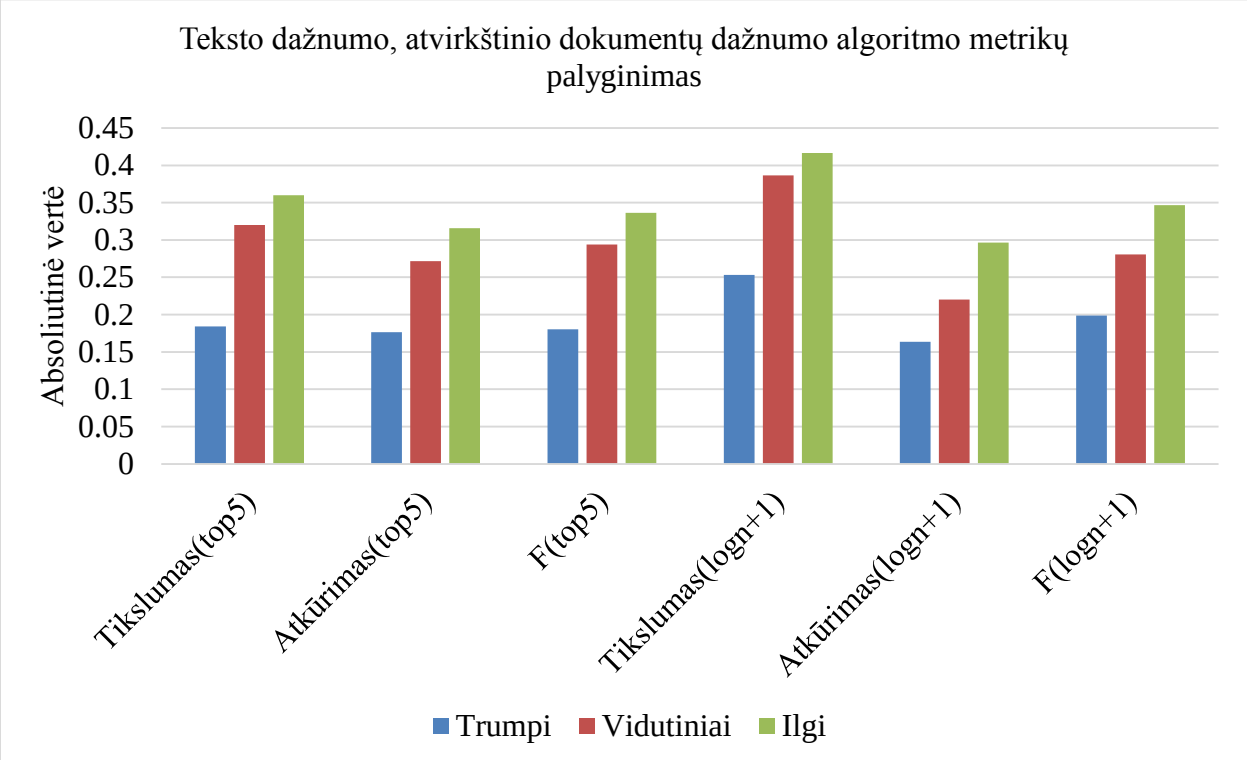
Negalima išskirti aiškaus būdo pasirinkti raktažodžiams. Nors dinaminio pasirinkimo būdas gauna geresnius rezultatus tikslumui, fiksuotas būdas yra geresnis atkūrimui.

Teksto dažnumo algoritmo tikslas yra išskirti dažniausiai tekste naudojamus žodžius, kurie galėtų būti naudojami kaip raktažodžiai. Rezultatai rodo, jog šis išskyrimas yra pakankamo tikslumo, jog algoritmas galėtų būti naudojamas realioje aplinkoje.

4.4.2. Teksto dažnumo, atvirkštinio dokumentų dažnumo algoritmo rezultatai

4.2 lentelė. Teksto dažnumo, atvirkštinio dokumentų dažnumo algoritmo rezultatai

Rasti(pagal top5)	**Tikslumas**	**Atkūrimas**	**F**	**Rasti(pagal logn+1)**	**Tikslumas**	**Atkūrimas**	**F**
Trumpi	0.184	0.17	0.18	Trumpi	0.25	0.16	0.19
Vidutiniai	0.32	0.27	0.29	Vidutiniai	0.38	0.22	0.28
Ilgi	0.36	0.31	0.33	Ilgi	0.41	0.29	0.34



4.3 pav. Teksto dažnumo, atvirkštinio dokumentų dažnumo algoritmo rezultatai

Grafikas 4.3 rodo teksto dažnumo, atvirkštinio dokumentų dažnumo algoritmų metrikų palyginimą. Iš rezultatų matyti, jog didesnis tikslumas, bei atkūrimas yra gaunamas analizuojant didesnės apimties tekstus.

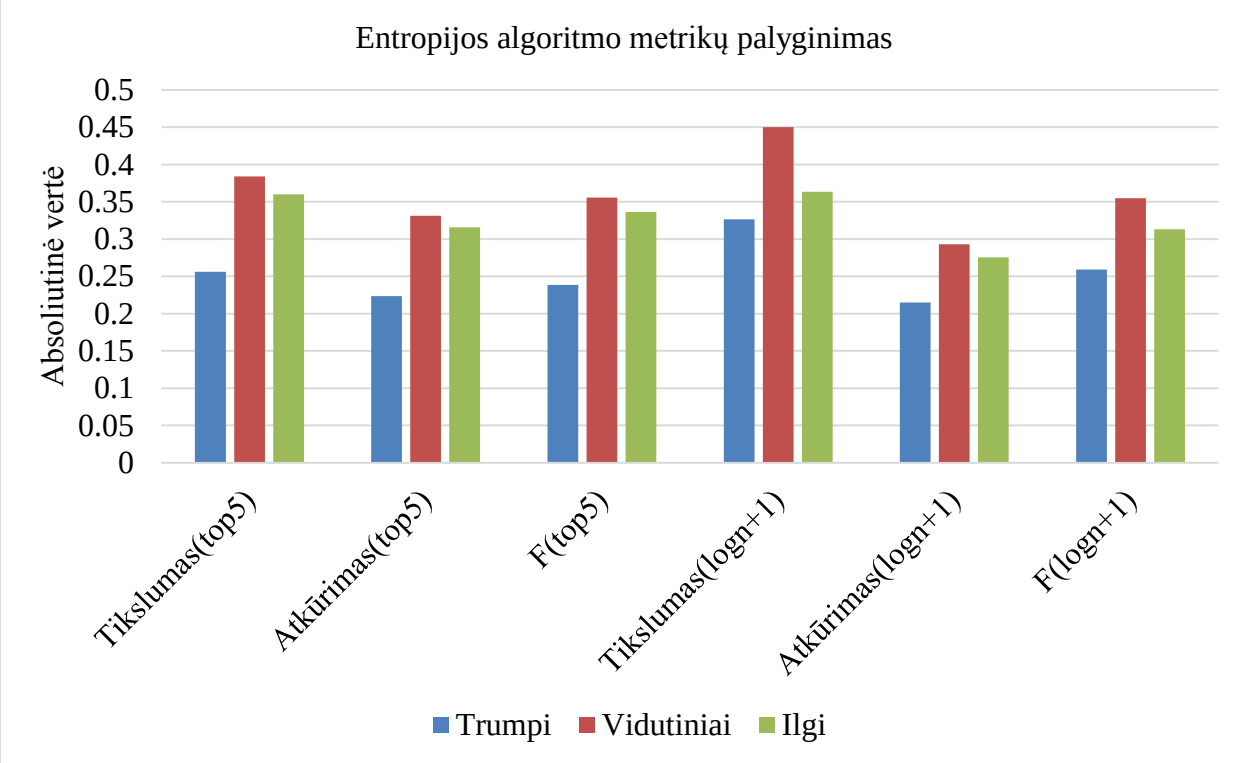
Lyginant raktažodžių pasirinkimo būdus nors ir nežymiai, tačiau naudojant dažnumo, atvirkštinio dokumentų dažnumo algoritmą geriau naudoti dinaminį būdą pasirinkti randamų raktažodžių skaičių tekste. Tuomet gaunamas aukštesnis tikslumo įvertinimas, bei šiek tiek geresnis atkūrimas.

Teksto dažnumo, atvirkštinio dažnumo algoritmo tikslas yra išskirti dažniausiai tekste naudojamus žodžius, tačiau retai naudojamus specifinėje srityje, kurie galėtų būti naudojami kaip raktažodžiai. Rezultatai rodo, jog šis išskyrimas yra pakankamo tikslumo, jog algoritmas galėtų būti naudojamas realioje aplinkoje.

4.4.3. Entropijos algoritmo rezultatai

4.3 lentelė. Entropijos algoritmo rezultatai

Rasti(pagal top5)	Tikslumas	Atkūrimas	F	Rasti(pagal logn+1)	Tikslumas	Atkūrimas	F
Trumpi	0.24	0.21	0.23	Trumpi	0.31	0.20	0.24
Vidutiniai	0.38	0.33	0.35	Vidutiniai	0.45	0.29	0.35
Ilgi	0.36	0.31	0.33	Ilgi	0.36	0.27	0.31



4.4 pav. Entropijos algoritmo rezultatai

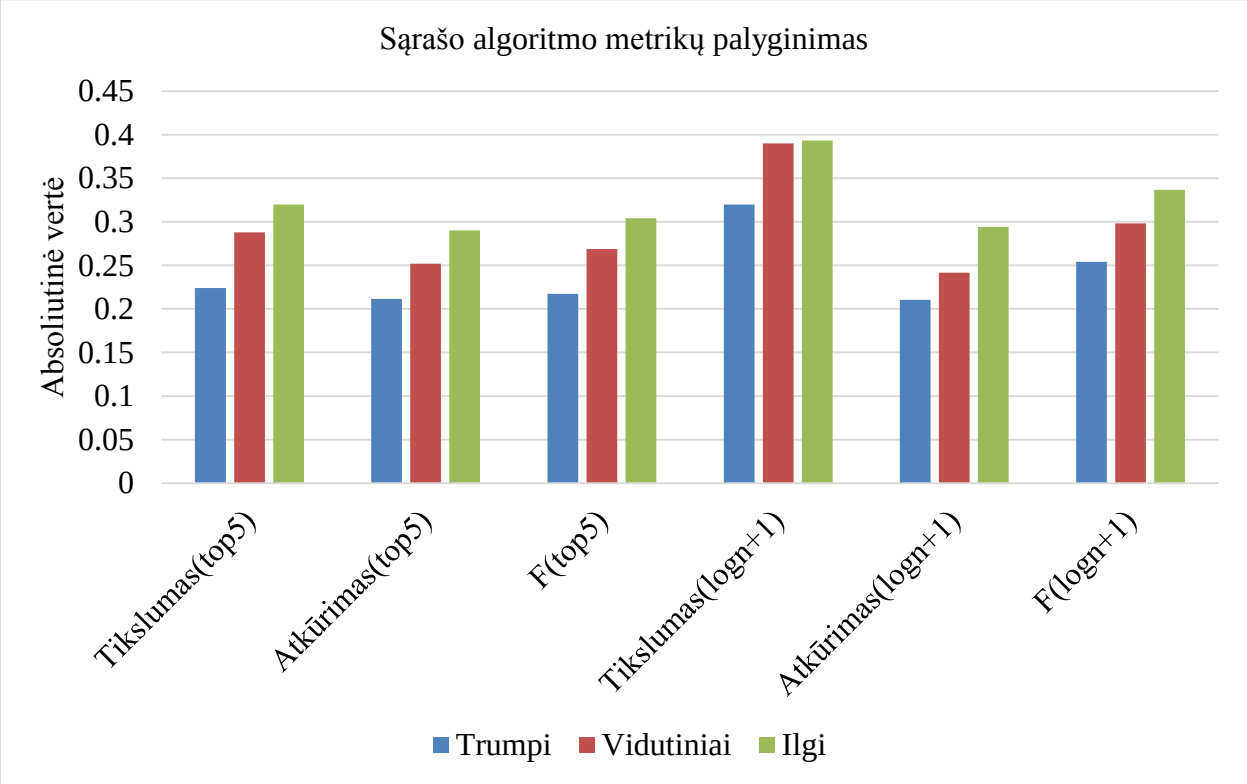
Grafikas 4.4 rodo entropijos algoritmo metrikų palyginimą. Iš rezultatų matyti, jog didesnis tikslumas, bei atkūrimas yra gaunamas analizuojant didesnės apimties tekstus. Tačiau priešingai, nei lyginant su teksto dažnumo ar teksto dažnumo, atvirkštinio dokumentų dažnumo algoritmais, vidutinio ilgumo tekstai turi geriausius rezultatus. Entropijos algoritmas paremtas sąryšio tarp tekste naudojamų žodžių nagrinėjimu, todėl esant per mažai informacijos, analizuojant trumus tekstus, algoritmas negali pakankamai tiksliai parinkti raktažodžių. Priešingu atveju, analizuojant didelį kiekį informacijos, randama daug ryšių, todėl algoritmui sunku tiksliai parinkti raktažodžius. Naudojant entropijos algoritmą geriausi rezultatai gaunami analizuojant vidutinės apimties tekstus.

Entropijos algoritmo tikslas nagrinėjant ryšius tarp žodžių pasirinkti žodžių grupę, iki keturių žodžių, kuri geriausiai galėtų apibūdinti tekstą. Nors šis algoritmas gali būti naudojamas visiškai automatiniu būdu, tačiau gaunami rezultatai turėtų būti peržiūrimi žmogaus.

4.4.4. Sąrašo algoritmo rezultatai

4.4 lentelė. Sąrašo algoritmo rezultatai

Rasti(pagal top5)	Tikslumas	Atkūrimas	F	Rasti(pagal logn+1)	Tikslumas	Atkūrimas	F
Trumpi	0.21	0.21	0.21	Trumpi	0.30	0.20	0.24
Vidutiniai	0.28	0.25	0.26	Vidutiniai	0.39	0.24	0.29
Ilgi	0.32	0.29	0.30	Ilgi	0.39	0.29	0.34



4.5 pav. Sąrašo algoritmo rezultatai

Grafikas 4.5 rodo sąrašo algoritmų metrikų palyginimą. Iš rezultatų matyti, jog didesnis tikslumas, bei atkūrimas yra gaunamas analizuojant didesnės apimties tekstus. Lyginant žodžių pasirinkimo būdą, šiek tiek geresnius rezultatus galima gauti naudojant dinaminį parinkimą. Kaip ir ankstesniuose rezultatuose didžiausias skirtumas gaunamas tikslumo metrikos reikšmėje.

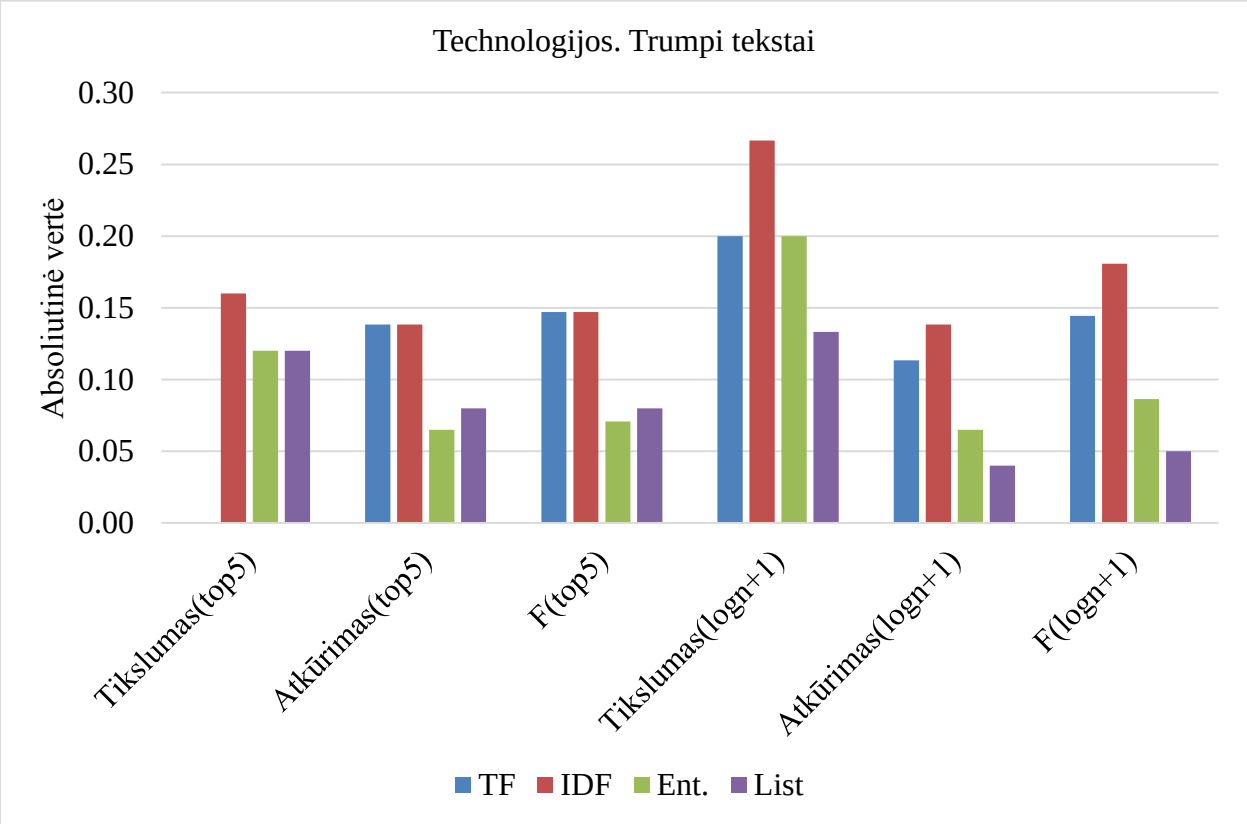
Sąrašo algoritmas, priešingai nei likusieji, turi iš anksto nustatytus raktažodžius. Tekstui bus priskirti raktažodžiai bus pasirinkti tik iš šio sąrašo. Todėl pagrindinis algoritmo tikslas yra suskirstyti tekstus į kategorijas. Būtent dėl to algoritmas turi prasčiausią atkūrimą. Vartotojų priskirti raktažodžiai, tiesiog nėra įtraukti į galimų raktažodžių sąrašą.

Nors algoritmas gali būti naudojamas automatiškai, tačiau rekomenduojama rezultatus, rastus raktažodžius, įvertinti patiems teksto autoriams.

4.5. SRITIES ANALIZIŲ PALYGINIMAS

4.5 lentelė. Technologijos kategorijos trumpų tekstų rezultatai

Žodžių	**Rasti(pagal top5)**	**Tikslumas**	**Atkūrimas**	**F**	**Rasti(p agal logn+1)**	**Tikslumas**	**Atkūrimas**	**F**
95	TF	0.00	0.14	0.15	TF	0.20	0.11	0.14
	IDF	0.16	0.14	0.15	IDF	0.27	0.14	0.18
	Ent.	0.12	0.07	0.07	Ent.	0.20	0.07	0.09
	List	0.12	0.08	0.08	List	0.13	0.04	0.05



4.6 pav. Technologijos kategorijos trumpų tekstų rezultatai

Pateiktas 4.6 grafikas rodo technologijos srities vidutinius, trumpų tekstų rezultatus. Pagrindinė žemo tikslumo priežastis yra ta, kad trumpuose tekstuose pateikiama per mažai informacijos tiksliai nustatyti teksto raktažodžius. Pastebėta pagrindinė žemo atkūrimo priežastis yra ta, kad autoriai pateikdami trumpus tekstus dažnai siekia pritraukti, sudominti skaitytojus raktažodžiais, kurie galėtų būti įdomus, tačiau ne kuo tiksliau apibūdinti tekstą. Mažas tikslumo įvertinimas, bet mažas atkūrimas tiesiogiai įtakoja žemą F metriką.

Iš keturių nagrinėtų algoritmų, trumpus tekstus tiksliau apibūdino teksto dažnumo, atvirkštinio dažnumo algoritmas. Prasčiausiai nagrinėjant trumpus tekstus yra naudoti tekstų dažnumo algoritmą arba sąrašo algoritmą. Žemus tekstų dažnumo algoritmo rezultatus sąlygoje faktas, jog autoriai apibūdinti trumpus tekstus naudoja raktažodžius, kurie yra nenaudojami tekste, arba naudojami tekste retai. Žemus sąrašo rezultatus sąlygoje faktas, jog algoritmui neužtenka informacijos tinkamai nustatyti teksto raktažodžius.

Gauti rezultatai, taip pat leidžia teikti, jog trumpiems tekstams analizuoti tinkamesnis dinaminis raktažodžių parinkimo būdas. Geriau pateiki mažiau, tačiau tikslesnius rezultatus, nei fiksuotą raktažodžių skaičių.

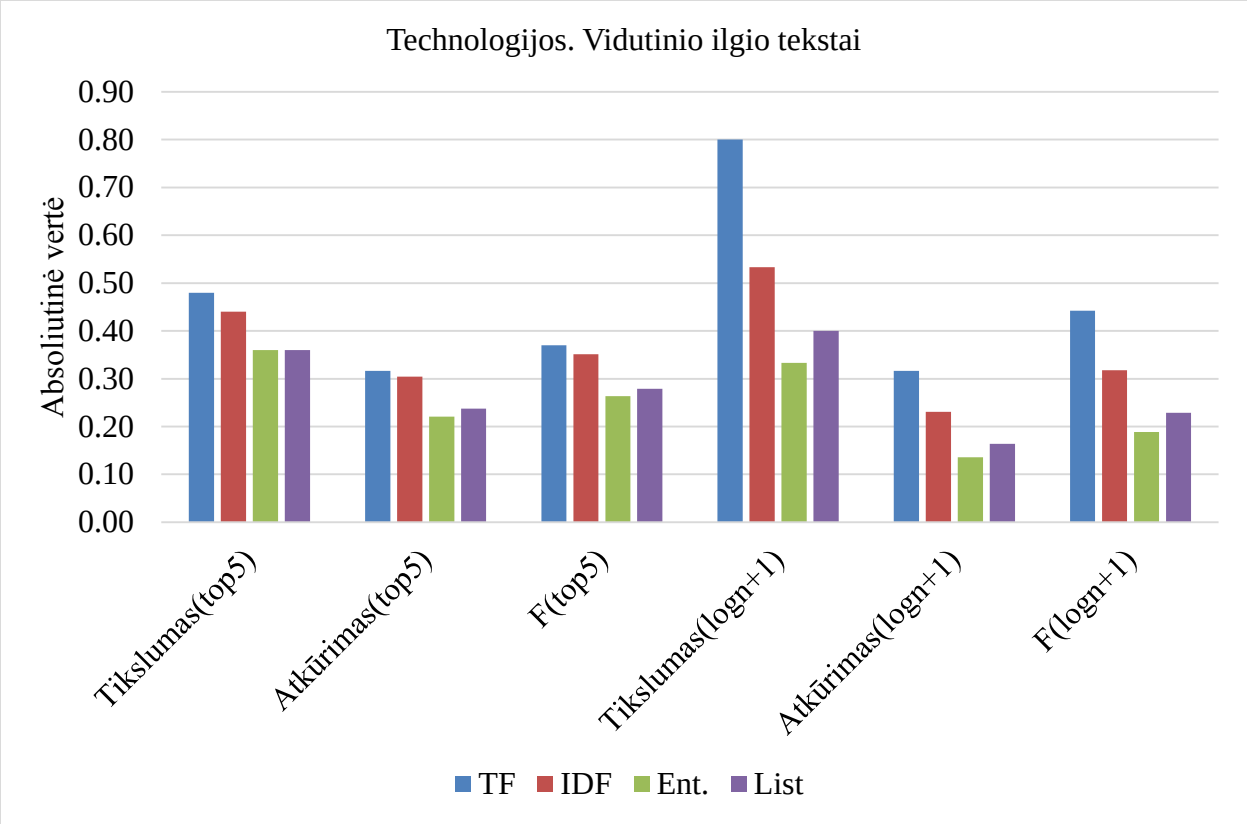
4.6 lentelė. Technologijos kategorijos vidutinių tekstų rezultatai

Žodžių	**Rasti(pagal top5)**	**Tikslumas**	**Atkūrimas**	**F**	**Rasti(p agal logn+1)**	**Tikslumas**	**Atkūrimas**	**F**
404	TF	0.48	0.32	0.37	TF	0.80	0.32	0.44
	IDF	0.44	0.30	0.35	IDF	0.53	0.23	0.32
	Ent.	0.36	0.22	0.26	Ent.	0.33	0.14	0.19
	List	0.36	0.24	0.28	List	0.40	0.16	0.23

Grafikas 4.7 rodo technologijos srities vidutinio ilgumo tekstų analizavimo rezultatus. Rezultatai rodo, jog ilgesnius tekstus analizuodami algoritmai gali geriau išskirti raktažodžius iš vidutinio ilgio tekstų lyginant su trumais tekstais. Ilgesnis tekstas suteikia daugiau informacijos apie autoriaus rašomą sritį. Lyginant su trumpais tekstais pagerėjo abi metrikos: tikslumas ir atkūrimas. Aukštesnis šių metrikų rezultatais tiesiogiai įtakoja ir tai, kad yra gaunama ir aukštesnė bendroji, F metrika.

Iš keturių nagrinėtų algoritmų, vidutinio ilgio tekstus tiksliau apibūdino teksto dažnumo, bei teksto dažnumo, atvirkštinio dažnumo algoritmai.

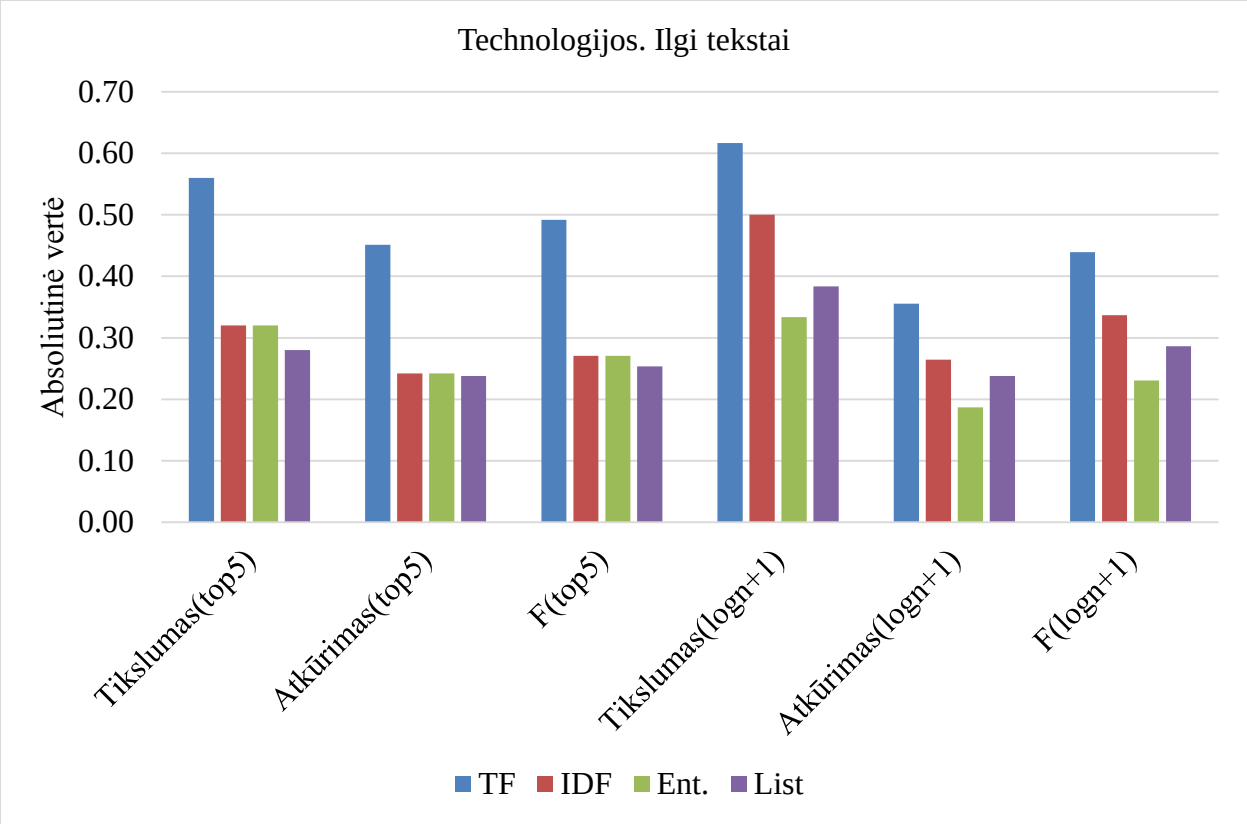
Gauti rezultatai, taip pat leidžia teikti, jog vidutinio ilgio tekstams analizuoti tinkamesnis dinaminis raktažodžių parinkimo būdas. Tačiau skirtumas tarp fiksuoto ir dinaminio raktažodžių skaičiaus nėra didelis.



4.7 pav. Technologijos kategorijos vidutinio ilgio tekstų rezultatai

4.7 lentelė. Technologijos kategorijos ilgų tekstų rezultatai

Žodžių	**Rasti(pagal top5)**	**Tikslumas**	**Atkūrimas**	**F**	**Rasti(p agal logn+1)**	**Tikslumas**	**Atkūrimas**	**F**
816	TF	0.56	0.45	0.49	TF	0.62	0.36	0.44
	IDF	0.32	0.24	0.27	IDF	0.50	0.26	0.34
	Ent.	0.32	0.24	0.27	Ent.	0.33	0.19	0.23
	List	0.28	0.24	0.25	List	0.38	0.24	0.29



4.8 pav. Technologijos kategorijos ilgų tekstų rezultatai

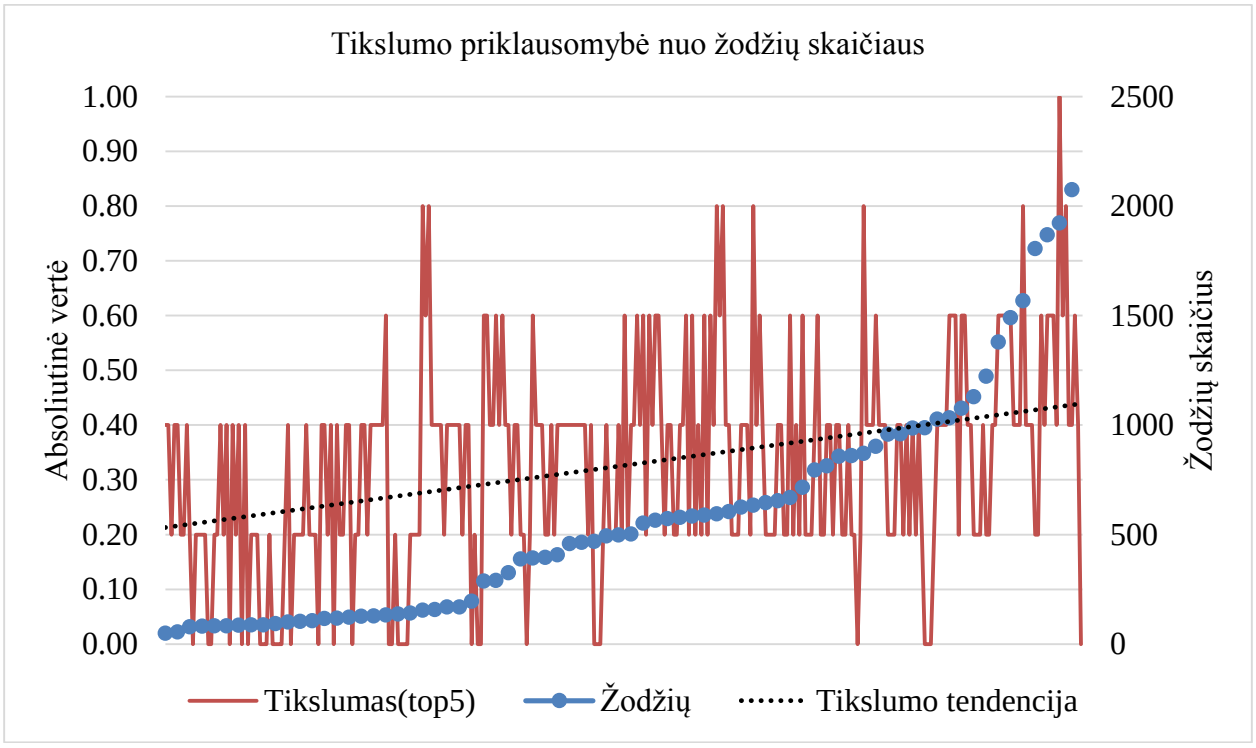
Grafikas 4.8 rodo technologijos srities ilgų tekstų analizavimo rezultatus. Rezultatai rodo, jog ilgesnius tekstus analizuodami algoritmai gali geriau išskirti raktažodžius iš ilgų tekstų. Ilgesnis tekstas suteikia daugiau informacijos apie autoriaus rašomą sritį. Lyginant su vidutinio ilgio tekstais pagerėjo abi metrikos: tikslumas ir atkūrimas. Aukštesnis šių metrikų rezultatais tiesiogiai įtakoja ir tai, kad yra gaunama ir aukštesnė bendroji, F metrika.

Iš keturių nagrinėtų algoritmų, vidutinio ilgio tekstus aukštesnį tikslumo įvertinimą gauna teksto dažnumo algoritmas. Tačiau atkūrimo rezultatai yra beveik vienodi.

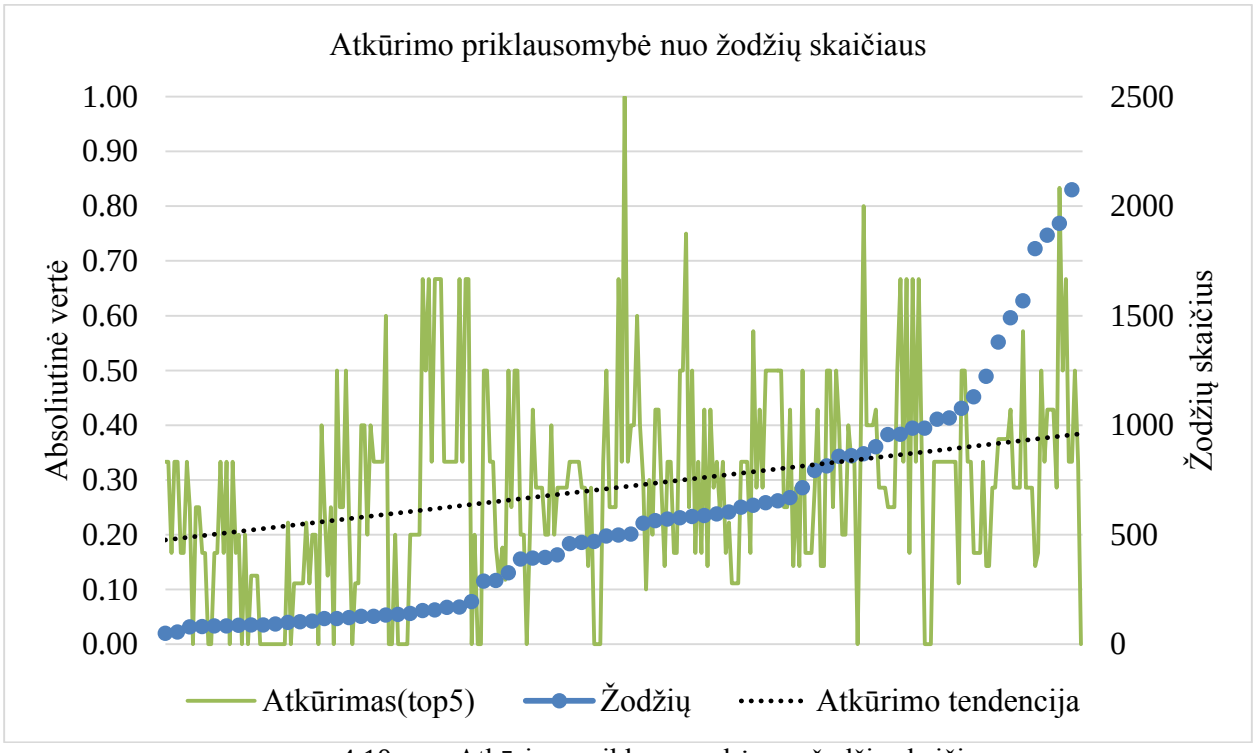
Gauti rezultatai, taip pat leidžia teikti, jog ilgiems tekstams analizuoti nebėra vieno tinkamiausio būdo pasirinkti raktažodžių. Abu pasirinkimo būdai gauna gana panašius rezultatus.

Vienos teksto kategorijos analizė rodo, jog norint gauti tikslesnius rezultatus, reikia pateikti daugiau duomenų algoritmams, analizuoti ilgesnius tekstus. Taip pat geriau naudoti dinaminį, nuo teksto ilgio priklausantį parenkamų raktažodžių skaičių. Tokie pat rezultatai yra gaunami ir kitose dokumentų srityse. Algoritmai gana panašiu tikslumu gali analizuoti skirtinų sričių dokumentus. Šie rezultatai yra pateikiami prieduose.

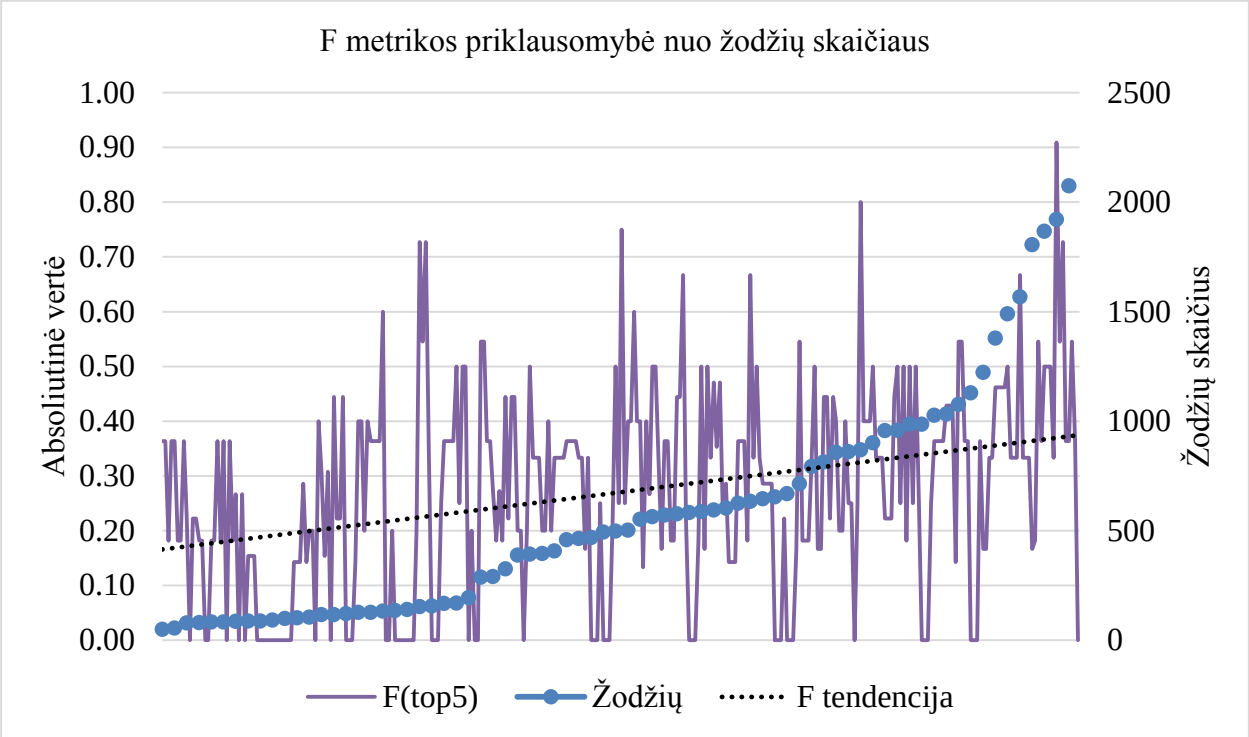
4.6. BENDRAS DOKUMENTŲ ŽYMĖJIMO PALYGINIMAS



4.9 pav. Tikslumo priklausomybė nuo žodžių skaičiaus



4.10 pav. Atkūrimo priklausomybė nuo žodžių skaičiaus



4.11 pav. F metrikos priklausomybė nuo žodžių skaičiaus

Pateikti grafikai 4.9, 4.10 ir 4.11 rodo analizuojamų teksto ilgio, žodžių skaičiumi, bei gaunamų metrikų priklausomybę. Kiekvienas grafikas turi dvi ašis. Kairioji ašis atvaizduoja parametro priklausomybę, dešinioji žodžių skaičių. Taip pat, kiekvienas grafas pateikia konkretaus parametro kitimo tendenciją. Galima matyti, jog pradžioje, esant mažam žodžių skaičiui lėtai auga nagrinėjamos metrikos. Vėliau, kai nagrinėjamo teksto ilgis pasiekia apytiksliai 100 žodžių, didėjant žodžių skaičiui kartu auga ir algoritmų metrikos (tikslumas, atkūrimas, bei F metrika). Vis didėjant žodžių skaičiui pasiekiama riba, teksto ilgiui esant apytiksliai 550 žodžių gaunamų metrikų reikšmės pradeda didėti kur kas lėčiau, nei analizuojamų žodžių skaičius.

Šis grafikas parodo, jog naudojant automatinio žymėjimo algoritmus optimaliausia rinktis vidutinio ilgio tekstus. Vidutinės apimties tekstai pateikia pakankamą informacijos kiekį išskirti tinkamus raktažodžius.

4.7. IŠVADOS

Pagal atliktą eksperimentinį tyrimą daromos išvados iškeltiems eksperimento tikslams yra tokios:

1. Algoritmai gali nagrinėti visų nagrinėtų sričių dokumentus, gaudami panašų analizuojamų dokumentų tikslumą, bei atkūrimą.
2. Nėra vieno algoritmo tinkančio visiems atvejams.
3. Egzistuoja priklausomybė, tarp analizuojamų žodžių skaičiaus, bei gaunamų algoritmo metrikų. Augant žodžių skaičiui, tikslėja algoritmų parenkami raktažodžiai. Rekomenduojama analizuoti vidutinės apimties tekstus.

4. Algoritmai negali tinkamai analizuoti trumpų tekstų.
5. Pasirenkant raktažodžių skaičių trumpiems, bei vidutinio ilgio tekstams geriau naudoti dinaminį būdą.
6. Nors ir remiasi paprasčiausiu būdų, teksto dažnumo, bei teksto dažnumo, atvirkštinio dokumentų dažnumo algoritmai gauna geriausius rezultatus tarp nagrinėjamų algoritmų.

5. IŠVADOS

Automatinis žiniatinklio dokumentų žymėjimas yra galimas pasinaudojant tyrimo metu nagrinėtais algoritmais. Analizuoti algoritmai skirti gauti skirtingo tipo raktažodžius, nėra vieno algoritmo tinkamo visiems atvejams.

Pasirinkta realizuoti kliento-serverio architektūra yra patogi naudoti, vartotojui nereikia papildomai įdiegti programinės įrangos. Naudojant sukurtą papildinį tekstai analizuojami vartotojo jau naudojamos interneto naršyklės papildinio pagalba. Pirmoji sistemos versija yra orientuota tik į anglų kalbą. Sistema kurta taip, kad ją būtų galima plėsti, tačiau norint palaikyti daugiau kalbų reikia turėti kalbų žodynus.

Teksto dažnumu grįsti algoritmai išskiria raktažodžius, iš nagrinėjamo teksto. Kadangi dauguma realių vartotojų naudojamų raktažodžių yra naudojami tekste, tokia algoritmo euristika gauna tikslius rezultatus. Entropija grįstas algoritmas analizuodamas ryšius tarp žodžių, pateikia raktažodžius susidedančius iš žodžių grupių. Nors rezultatus sudaro analizuojamo teksto žodžiai, šio algoritmo rezultatų teisingumą, turėtų įvertinti teksto autorius. Sąrašo algoritmas iš anksto analizuodamas tekstą, parenka geriausiai tinkančius raktažodžius iš anksto parinkto sąrašo. Toks parinkimo būdas geriausiai tinka priskirti tekstus jau žinomoms kategorijoms, todėl atkūrimas visuomet bus žemas.

Nagrinėti algoritmai gali analizuoti tekstus, bei autoriams rekomenduoti pasirinkti raktažodžius. Šis tikslumas priklauso nuo analizuojamų žodžių skaičius, eksperimento metu pastebėta, jog geriausia nagrinėti vidutinės apimties tekstus. Tokio ilgio tekstai turi pakankamai informacijos išskirti raktažodžius, bei aprašo vieną sritį.

Analizuojant atsitiktinius autorių tekstus pastebėta, jog algoritmai gali analizuoti skirtingų sričių tekstus, gaudami panašaus tikslumo rezultatus. Be to dauguma autorių naudoja raktažodžius tuos žodžius, kurie yra tiesiogiai naudojami tekste. Toks pobūdis leidžia teksto dažnumo algoritmams pateikti rezultatus, kurie turi aukštą atkūrimą. Tokio tipo algoritmai be didelių skaičiavimų, tačiau tik po to kai yra sukaupta didelė žodžių dažnumo informacija, gali pateikti tikslius rezultatus.

6. LITERATŪRA

[1] Nirmala Pudota, Antonina Dattolo, Andrea Baruzzo, Felice Ferrara, Carlo Tasso: Automatic keyphrase extraction and ontology mining for content-based tag recommendation, 2010

[2] Scott Golder and Bernardo A. Huberman. The structure of collaborative tagging systems. Journal of Information Science, 198–208, 2006.

[3] Florin Gorunesc: Data Mining Concepts, Models and Techniques. 1-38, 2011.

[4] Ronen Feldman,James Sanger: The Text Mining Handbook - Advanced Approaches In Analyzing Unstructured Data. 64-80, 2007.

[5] Hércules Antonio do Prado: Edilson Ferneda: Emerging Technologies of Text Mining: Techniques and Applications. 54-75, 2005.

[6] Michael W. Berry,Jacob Kogan: Text Mining Applications and Theory. 3-30, 2010.

[7] Gilad Mishne. Using blog properties to improve retrieval. In ICWSM 2007 – International Conference on Weblogs and Social Media, 2007.

[8] Min Song , Yi-fang Brook Wu Handbook of Research on Text and Web Mining Technologies(2). 733-740, 2005.

[9] Gilad Mishne. AutoTag: A collaborative approach to automated tag assignment for weblog posts. In WWW ’06: Proceedings of the 15th international World Wide Web conference on Alternate track papers & posters, 2006.

[10] Hsinchun Chen and Michael Chau: Web Mining: Machine learning for Web Applications. 289-308, 2007.

[11] Xu,Yanchun Zhang,Lin Li: Web Mining and Social Networking Techniques and Applications. 20-86, 2011.

[12] M.F. Porter, 1980, An algorithm for suffix stripping, Program, 14(3) pp 130−137.

[13] Leksikos duomenų bazė anglų kalba [Žiniatinklis]. Nuoroda: http://wordnet.princeton.edu/ [Peržiūrėta: Balandžio 10, 2014]

[14] Stephen Robertson: Understanding Inverse Document Frequency: On theoretical arguments for IDF. Journal of Documentation 60, 503-520, 2004

[15] Thorsten Joachims. Text categorization with suport vector machines: Learning with many relevant features. In ECML ’98: Proceedings of the 10th European Conference on Machine Learning, pages 137–142, 1998.

[16] Kilian Q. Weinberger , John Blitzer , Lawrence K. Saul: Distance metric learning for large margin nearest neighbor classification. 2006

[17] K. Y. Matsuo and M. Ishizuka, "Keyword Extraction From A Single Document Using Word Co-Occurrence Statistical Information" International Journal on Artificial Intelligence Tools, pp. 157-169 , 2004.

[18] A. K. Mehran, "Fuzzy model predictive control using Takagi-Sugeno model " in Control, Automation and Systems, 2008. ICCAS 2008. International Conference on, pp. 632 – 637

[19] „Most Popular Blog Tags - Blog Flux“ [Žiniatinlio nuoroda]. Adresas: http://dir.blogflux.com/tags.php [Peržiūrėta: Kovo. 27, 2014]

[20] „Blog Directory – Technorati“ [Žiniatinlio nuoroda]. Adresas: http://technorati.com/blogs/directory. [Peržiūrėta: Kovo. 27, 2014]

[21] „A social bookmarks manager“ [Žiniatinlio nuoroda]. Adresas: http://delicious.com/. [Peržiūrėta: Kovo. 27, 2014]

[22] Salton G, Buckley C (1988). "Term-weighting approaches in automatic text retrieval". Information Processing and Management 24 (5): 513–523
[23] Robertson, Stephen. "Understanding inverse document frequency: On theoretical arguments for IDF". Journal of Documentation 60 (5): 503–520.
[24] Press, William H.; Teukolsky, Saul A.; Vetterling, William T.; Flannery, B. P. (2007). "Section 16.5. Support Vector Machines". Numerical Recipes: The Art of Scientific Computing (3rd ed.).
[25] Altman, N. S. (1992). "An introduction to kernel and nearest-neighbor nonparametric regression". The American Statistician 46 (3): 175–185.

7. PRIEDAI

7.1. SKIRTINGŲ TEKSTO STILIŲ ANALIZAVIMO REZULTATAI

7.1 lentelė. Karjeros kategorijos trumpų tekstų rezultatai

Žodžių	Rasti(pagal top5)	Tikslumas	Atkūrimas	F	Rasti(pagal logn+1)	Tikslumas	Atkūrimas	F
87	TF	0.32	0.21	0.25	TF	0.35	0.16	0.21
	IDF	0.20	0.15	0.17	IDF	0.27	0.14	0.19
	Ent.	0.36	0.23	0.28	Ent.	0.40	0.20	0.26
	List	0.16	0.12	0.14	List	0.15	0.09	0.11

7.2 lentelė. Karjeros kategorijos vidutinio ilgio tekstų rezultatai

Žodžių	Rasti(pagal top5)	Tikslumas	Atkūrimas	F	Rasti(pagal logn+1)	Tikslumas	Atkūrimas	F
418	TF	0.48	0.43	0.45	TF	0.57	0.34	0.42
	IDF	0.36	0.32	0.34	IDF	0.38	0.23	0.28
	Ent.	0.48	0.44	0.45	Ent.	0.48	0.28	0.36
	List	0.40	0.37	0.38	List	0.63	0.37	0.46

7.3 lentelė. Karjeros kategorijos ilgų tekstų rezultatai

Žodžių	Rasti(pagal top5)	Tikslumas	Atkūrimas	F	Rasti(pagal logn+1)	Tikslumas	Atkūrimas	F
1253	TF	0.36	0.48	0.39	TF	0.45	0.48	0.44
	IDF	0.28	0.35	0.29	IDF	0.35	0.35	0.33
	Ent.	0.44	0.55	0.47	Ent.	0.45	0.48	0.44
	List	0.28	0.32	0.28	List	0.37	0.31	0.31

7.4 lentelė. Kulinarijos kategorijos trumpų tekstų rezultatai

Žodžių	Rasti(pagal top5)	Tikslumas	Atkūrimas	F	Rasti(pagal logn+1)	Tikslumas	Atkūrimas	F
174	TF	0.44	0.45	0.44	TF	0.48	0.38	0.42
	IDF	0.20	0.20	0.20	IDF	0.15	0.10	0.12
	Ent.	0.44	0.48	0.45	Ent.	0.53	0.45	0.47
	List	0.28	0.28	0.28	List	0.40	0.28	0.32

7.5 lentelė. Kulinarijos kategorijos vidutinio ilgio tekstų rezultatai

Žodžių	Rasti(pagal top5)	Tikslumas	Atkūrimas	F	Rasti(pagal logn+1)	Tikslumas	Atkūrimas	F
638	TF	0.60	0.40	0.46	TF	0.65	0.31	0.41
	IDF	0.44	0.31	0.35	IDF	0.48	0.25	0.31
	Ent.	0.56	0.39	0.44	Ent.	0.58	0.34	0.41
	List	0.32	0.21	0.24	List	0.38	0.18	0.23

7.6 lentelė. Kulinarijos kategorijos ilgų tekstų rezultatai

Žodžių	Rasti(pagal top5)	Tikslumas	Atkūrimas	F	Rasti(pagal logn+1)	Tikslumas	Atkūrimas	F
1477	TF	0.68	0.51	0.58	TF	0.70	0.41	0.52
	IDF	0.48	0.36	0.41	IDF	0.55	0.32	0.41
	Ent.	0.52	0.39	0.44	Ent.	0.55	0.32	0.41
	List	0.44	0.32	0.37	List	0.55	0.32	0.41

7.7 lentelė. Gyvūnų kategorijos trumpų tekstų rezultatai

Žodžių	Rasti(pagal top5)	Tikslumas	Atkūrimas	F	Rasti(pagal logn+1)	Tikslumas	Atkūrimas	F
131	TF	0.40	0.46	0.42	TF	0.45	0.32	0.36
	IDF	0.20	0.22	0.21	IDF	0.32	0.20	0.24
	Ent.	0.20	0.25	0.22	Ent.	0.25	0.20	0.21
	List	0.36	0.42	0.38	List	0.57	0.38	0.44

7.8 lentelė. Gyvūnų kategorijos vidutinio ilgio tekstų rezultatai

Žodžių	Rasti(pagal top5)	Tikslumas	Atkūrimas	F	Rasti(pagal logn+1)	Tikslumas	Atkūrimas	F
392	TF	0.32	0.25	0.28	TF	0.42	0.22	0.29
	IDF	0.24	0.21	0.22	IDF	0.32	0.18	0.22
	Ent.	0.28	0.22	0.25	Ent.	0.42	0.22	0.29
	List	0.24	0.19	0.21	List	0.30	0.17	0.21

7.9 lentelė. Gyvūnų kategorijos ilgų tekstų rezultatai

Žodžių	Rasti(pagal top5)	Tikslumas	Atkūrimas	F	Rasti(pagal logn+1)	Tikslumas	Atkūrimas	F
1159	TF	0.40	0.32	0.35	TF	0.37	0.22	0.28
	IDF	0.48	0.38	0.42	IDF	0.47	0.29	0.35
	Ent.	0.36	0.29	0.32	Ent.	0.32	0.19	0.24
	List	0.24	0.19	0.21	List	0.27	0.16	0.20

7.10 lentelė. Mokslo kategorijos trumpų tekstų rezultatai

Žodžių	Rasti(pagal top5)	Tikslumas	Atkūrimas	F	Rasti(pagal logn+1)	Tikslumas	Atkūrimas	F
131	TF	0.28	0.26	0.00	TF	0.27	0.27	0.27
	IDF	0.12	0.17	0.00	IDF	0.20	0.20	0.20
	Ent.	0.20	0.22	0.00	Ent.	0.33	0.33	0.33
	List	0.16	0.15	0.15	List	0.27	0.27	0.27

7.11 lentelė. Mokslo kategorijos vidutinio ilgio tekstų rezultatai

Žodžių	Rasti(pagal top5)	Tikslumas	Atkūrimas	F	Rasti(pagal logn+1)	Tikslumas	Atkūrimas	F
574	TF	0.40	0.39	0.00	TF	0.52	0.53	0.52
	IDF	0.20	0.21	0.00	IDF	0.28	0.28	0.28
	Ent.	0.24	0.22	0.00	Ent.	0.35	0.35	0.35
	List	0.20	0.23	0.21	List	0.28	0.30	0.29

7.12 lentelė. Mokslo kategorijos ilgų tekstų rezultatai

Žodžių	Rasti(pagal top5)	Tikslumas	Atkūrimas	F	Rasti(pagal logn+1)	Tikslumas	Atkūrimas	F
949	TF	0.20	0.25	0.18	TF	0.27	0.28	0.27
	IDF	0.20	0.25	0.18	IDF	0.22	0.23	0.22
	Ent.	0.12	0.13	0.09	Ent.	0.17	0.17	0.17
	List	0.32	0.40	0.35	List	0.43	0.47	0.45

7.2. STRAIPSNIS „ŽINIATINKLIO DOKUMENTŲ AUTOMATINIO ŽYMĖJIMO SISTEMA“

KAUNO
TECHNOLOGIJOS
UNIVERSITETAS

DALYVIO PAŽYMĖJIMAS

2014-04-24 Nr. ST25-F-14-12

Kaunas

Pažymima, kad

Remigijus Valys (KTU)

2014 m. balandžio 24 d. dalyvavo tarpuniversitetinėje konferencijoje „Informacinė visuomenė ir universitetinės studijos“ (IVUS 2014) ir perskaitė pranešimą

Žiniatinklio dokumentų automatinio
žymėjimo sistema

Informatikos fakulteto dekanas **prof. Eduardas Bareiša**

Programų komiteto pirmininkas **prof. Robertas Damaševičius**

Žiniatinklio dokumentų automatinio žymėjimo sistema

Remigijus Valys
Informatikos fakultetas
Kauno technologijos universitetas
Kaunas, Lietuva
El. paštas: remigijus.valys@stud.ktu.lt

***Santrauka*—Straipsnio tikslas supažindinti su žiniatinklio dokumentų automatiniu žymėjimu. Proceso metu kylančiomis problemomis ir sprendimo būdais Aprašomi kuriamoje sistemoje naudojami algoritmai, atliktas eksperimentas, bei gauti rezultatai.**

Raktažodžiai—žiniatinklio dokumentai; automatinis žymėjimas;

1. ĮVADAS

Pastaraisiais metais internete atsirado daug naujų programų, kurios leidžia vartotojams būti ne tik vien tik stebinčiais vartotojais, bet ir dalyvauti sistemos kūrime. Tokios programos leidžia vartotojams pateikti ne vien tik savo duomenis, tuos duomenis kategorizuoti, suteikti meta duomenis. Taip naudojant meta duomenis skirtingų vartotojų duomenys gali būtų susiejami. Siejant duomenis bei žmones gaunamas tinklas siejantis informaciją. Vartotojai kurdami tinklą gali įtakoti sistemą. Tokios sistemos susilaukia didelio vartotojų dėmesio, yra populiarios.

Tokios sistemos leidžia vartotojams sužinoti tik tam tikros norimos srities informaciją arba prisidėti plečiant srities dokumentus. Kadangi informacijos kiekis yra didelis, vartotojams yra siūloma prie pateikiamo teksto pateikti ir kelis raktažodžius, kurie apibūdintų patį turinį. Taip kitiems vartotojams lengviau surasti norimą informaciją ar pačiai sistemai teikti rekomendacijas apie turinį, kuris galėtų būti aktualus kitiems vartotojams [11]. Egzistuoja kelios pagrindinės problemos:

- toks turinio kategorizavimas nėra privalomas, yra tik pageidautinas. Tai sąlygoja, kad didelė informacijos dalis yra nekategorizuota,
- antra problema yra ta, kad vartotojai nesistengia panaudoti jau naudotų raktažodžių, taip sistemoje atsiranda sinonimų.

Vienas iš būdų kaip spręsti tokias problemas yra naudoti automatinio dokumentų žymėjimo sistemą. Šios sistemos įvestis yra vartotojo norimas analizuoti tekstas, o gaunami rezultatai yra analizuojamo teksto raktažodžiai. Sistema gali sujungti raktažodžius su informacija, vartotojais [7].

Sistemai pateikus siūlomus raktažodžius vartotojas gali pasirinkti iš pateikto sąrašo. Jei vartotojas mano, kad pateikiami raktažodžiai nekategorizuoja teksto jis gali koreguoti

raktažodžius. Taip galima spręsti problemas dėl nekategorizuotų duomenų, sinonimų naudojimo. Didesnio tikslumo sistemos naudoja algoritmus, kurie analizuoja ne tik šiuo metu pateikiamą tekstą, tačiau atsižvelgia ir į anksčiau nagrinėtus tekstus, taip pat ankstesnius šio vartotojo raktažodžius.

2. TEKSTO ANALIZĖ

2.1. DUOMENŲ GAVYBA

Bendru atveju duomenų gavyba yra procesas, kurio metu yra įvairiapusiškai analizuojami duomenys ir rezultatus apjungiant į naudingą informaciją. Teksto duomenų gavyba automatizuotai atranda naują, prieš tai nežinoma informaciją, analizuojamuose šaltiniuose [2]. Teksto duomenų gavyba prasideda nuo to, kad analizuojamame tekste yra išskiriami faktai ar įvykiai, kurie vėliau yra analizuojami tradiciniais duomenų gavybos metodais. Galima išgauti reikalingą informaciją apie norimą tekstą, susieti žmones, vietas, panašius objektus, identifikuoti, organizuoti ir grupuoti dokumentus pagal jų turinį.

Naudojant analizavimo algoritmus sprendimai remiasi „skaldyk ir valdyk“ principu [1]. Kai yra žinoma konkreti užduotis, kurią reikia išspręsti, algoritmai sudalina sudėtingą ir didelę užduotį į mažesnes, bei paprastesnes. Mažesnės užduotys yra išsprendžiamos atskirai ir tuomet gauti rezultatai yra apjungiami. Egzistuoja keli pagrindiniai žingsniai [3][5]:

- teksto paruošimo algoritmams žingsnis. Šio etapo metu yra formuojamas tekstas, filtruojami nereikalingi, nenaudingi simboliai,
- papildomos informacijos radimo žingsnis. Iš atskirų tekste esančių žodžių yra ieškoma susijusi informacija. Šiame etape norint išgauti kuo tikslesnę informaciją be paprastų, šablonais grįstų algoritmų yra naudojami protingesni algoritmai. Analizuojami kelių žodžių junginiai, ar žodynai, norint išgauti žodžių prasmes, susieti sinonimus. Žingsnis taip pat įvertinta, kad keli žodžiai gali sudaryti junginį, kurie sudaro pavadinimą (pvz.: „saulės slėnis“)
- neinformacinių žodžių pašalinimo žingsnis. šiame etape yra pašalinami žodžiai kurie egzistuoja tekste, tačiau nesuteikia informacijos, bei nebus naudojami kaip raktažodžiai (pvz.: „ar“, „arba“, „ir“, „irgi“). Taip padidinamas būsimų raktažodžių tikslumas, bei sumažinamas būsimų skaičiavimų laikas.
- normalizavimo etapas. Šiame etape iš atrinktų žodžių yra išskiriamas žodžio kamienas. Toliau skaičiuojant žodžių dažnumą, bus naudojamas žodžio kamienas.
- svorių skaičiavimo žingsnis. Šiame etape stengiamasi priskirti žodžiams svorius. Didesnis svoris parodo jpg naudojamas žodis yra svarbesnis konkrečiame kontekste.
- panašumo įvertinimo žingsnis. Norint gauti tikslesnius raktažodžius galima įvertinti vieno teksto fragmento panašumą su kitais tekstais. Tokie skaičiavimai yra sudėtingi, bei reikalauja daug laiko, tačiau leidžia panaudoti jau naudotus raktažodžius [10].
- raktažodžių parinkimo žingsnis. Šiame žingsnyje yra pasirenkami konkretūs raktažodžiai, kurie geriausiai apibūdina tekstą.
- testavimas. Žingsnis nenaudojamas, kai algoritmas yra naudojamas realioje sistemoje, tačiau būtinas kiekvienam algoritmui. Žingsnyje yra įvertinamas algoritmo tikslumas ir patikimumas.

2.2. APMOKYMAS

Kadangi kompiuterizuota sistema negali logiškai suprasti, kas tiksliai yra norima pasakyti konkrečiame dokumente ar jo ištraukoje, yra atliekami spėjimai, kurie remiasi ankstesnių teksto analizių rezultatais. Jau sukaupta algoritmo "patirtis" yra panaudoti realių, vartotojų pateiktų duomenų analizei [9].

Norint sukaupti pradinius duomenis algoritmai yra apmokomi. Todėl analizei yra pateikiami tekstai, kai kuriems algoritmams, taip pat reikalingi ir raktažodžiai išskirti žmogaus. Mašininio mokinimosi algoritmams pateikti duomenys leidžia suprasti dalykinę sritį. Kuo didesnis, bei kuo tikslesnis mokinimo duomenų yra pateikiama algoritmams tuo geresnių rezultatų galima tikėtis naudojant algoritmą. Priklausomai nuo pačio algoritmo šis žingsnis gali būti ne tik naudojamas pradinėje fazėje, prieš pradedant naudoti sistemą, tačiau ir pačios eksploatacijos metu, taip tik padidinant vėlesnių duomenų tikslumą.

3. DUOMENŲ GAVYBOS SUNKUMAI

Nors teksto duomenų gavyba pradėta studijuoti prieš dešimtmečius, šiandien dar egzistuoja nemažai iššūkių, bei sunkumų, kurie turi būti įvertinti:

- Greitis – idealiai, kiekvienam duotam tekstui norima rasti visą susijusią informaciją, kuri leistų kuo tiksliau apibūdinti sritį. Greitis glaudžiai susijęs su analizuojamu ir jau analizuotų duomenų kiekiu.
- Tikslumas – nagrinėjami algoritmai yra skirtingi. Mašininio mokinimosi algoritmams gauti tikslesniu rezultatus reikia gerų duomenų „apsimokyti". Realioje aplinkoje tikslumas priklauso nuo galimybės aptikti susietus duomenis.
- Prisitaikomumas – algoritmas gali puikiai veikti vienoje aplinkoje, tačiau blogai kitoje. Pavyzdžiui jei naudojamas algoritmas buvo „apmokytas" analizuojant su muzika susijusius įrašus jam bus sunku analizuoti medicininius dokumentus. Dažniausiai, analizavimo algoritmai turi skirtingą veikimo greitį priklausomai nuo duomenų kiekio toje pačioje sistemoje, todėl sunku sukurti algoritmą ir jį „apmokyti" visiems gyvenimo atvejams.
- Personalizavimas – susijusios informacijos gavimas priklausomai nuo to koks vartotojas pateikė užklausą yra nauja tyrimo kryptis. Internetinių dokumentų srityje yra norima, kad algoritmas sugebės susidoroti su plačiu vartotojų ratu, bei su dokumentais, kurių turinys gali būti labai platus. Pavyzdžiui personalizuotas raktažodžių radimo algoritmas, galėtų būti papildytas funkcija, kuri leistų vartotojams į sistemą įtraukti savo kategorijas, o algoritmas automatiškai galėtų „apsimokyti" ir suprasti koks tekstas gali būti šiuo raktažodžiu apibūdinamas.
- Nereikšmingi žodžiai – kalboje yra naudojami žodžiai, kurie netinkami būti naudojami kaip raktažodžiai, tačiau egzistuoja tekste. Pavyzdžiui „aš", „ir", „arba" ir t.t. Kad tokie žodžiai nebūtų naudojami kaip raktažodžiai yra sudaromas specialus sąrašas (angl. terminas „stop words list")[6]. Šie žodžiai priklausomai nuo algoritmo analizuojant yra pašalinami arba neįtraukiami į atitinkamus sąrašus.
- Linksniai, bei sinonimai – išskiriant raktažodžius yra tikslinga naudoti jau esamus ir naudojamus raktažodžius. Todėl dažnai žodžiai yra normalizuojami. Iš duoto žodžio yra gaunama jo šaknis ir tik tuomet žodis yra įvertinamas. Taip tokie žodžiai kaip „greitaveika" ar „greitumas" yra normalizuojami ir toliau analizuojamas bendras žodis. Žodžio normalizavimo algoritmai nebūtinai gražins žodį, kuris yra teisingas, todėl

normalizuoti žodžiai nėra naudojami konkrečių raktažodžių pasirinkimui. Norint pašalinti sinonimus naudojami žodynai. Pašalinant, dažniausia pasirenkamas tas sinonimas, kuris yra dažniau naudojamas šnekamojoje kalboje. Vienas iš tokių žodynų yra „WordNet“. Procesas yra sudėtingesnis, tačiau reikalingas norint sumažinti informacijos dubliavimą..

4. EKSPERIMENTO TIKSLAS

Eksperimentas orientuotas į anglų kalbą. Projekto tikslas sukurti internetinių duomenų automatinio žymėjimo sistemą, kuri iš pateikto angliško teksto galėtų išskirti raktažodžius ir juos pateiktų vartotojams. Kuriamas interneto naršyklės plėtinys, kuris leistų vartotojams greitai ir patogiai nenaudojant papildomų programų analizuoti rašomus tekstus. Vartotojas gali pele pažymėti tekstą, tuomet kontekstinio menu pagalba pasirinkti teksto analizavimą. Atlikus analizę vartotojui bus pateikiamas sąrašas su siūlomais raktažodžiais.

Sistema paremta kliento ir serverio architektūra, ją sudaro sudaro du pagrindiniai komponentai. Tai interneto naršyklės papildinys, bei serveris. Interneto naršyklės papildinys yra skirtas tam, kad vartotojas galėtų patogiu būtu, nenaudodamas pašalinių programų, o tiesiog kurdamas ar redaguodamas interneto dokumentą galėtų jį analizuoti. Papildinyje realizuojamas didelių skaičiavimų nereikalaujantis, tačiau galintis gauti gerus rezultatus, žodžių dažniu grįstas algoritmas. Serveryje yra realizuoti sudėtingesni teksto analizavimo algoritmai. Taip pat naudojamas mašininis mokinimasis. Jis analizuoja kokius žodžius vartotojas naudoja tekste, bei kaupia apie tai duomenis. Vėliau, kai vartotojas pateiks analizuoti tekstą, į žodžio svorio skaičiavimą bus įtrauktas ir požymis, kuris parodys kaip dažnai žodis yra naudojamas konkretaus vartotojo.

5. SISTEMOS PRISITAIKYMAS

Norint gauti tikslesnius rezultatus, sukurta sistema renka informaciją apie vartotojo analizuojamus tekstus, bei rezultatus.

Vartotojui analizavus tekstą yra išsaugomi rasti raktažodžiai. Šiuos raktažodžius vartotojas gali reitinguoti, taip keisdamas algoritmų veikimą. Vartotojo suteiktas didesnis reitingas suteks žodžiui pradinį svorį. Taip pat jei po analizės sistema gali rinktis iš kelių panašaus svorio žodžių, bus pasirinktas tas kurio reitingas yra didesnis. Vartotojai priskiria reitingus individualiai ir tik po to, kai sistema jau yra išskyrus žodį kaip raktažodį.

Sistemos pradiniai duomenys yra nustatyti, taip kad raktažodžiais negali būti tik nereikšmingi žodžiai. Tačiau jei vartotojas nori, jis gali į sistemą įvesti savo žodžius, kurie jam yra nereikalingi. Šie žodžiai nebebus naudojami kaip sistemos išskiriami raktažodžiai.

Surinkta informacija yra naudojama vartotojui analizuojant tekstus. Taip sistema gali mokytis, bei prisitaikyti prie konkretaus vartotojo viso proceso metu.

6. ALGORITMAI

6.1. DAŽNIU GRĮSTAS ALGORITMAS

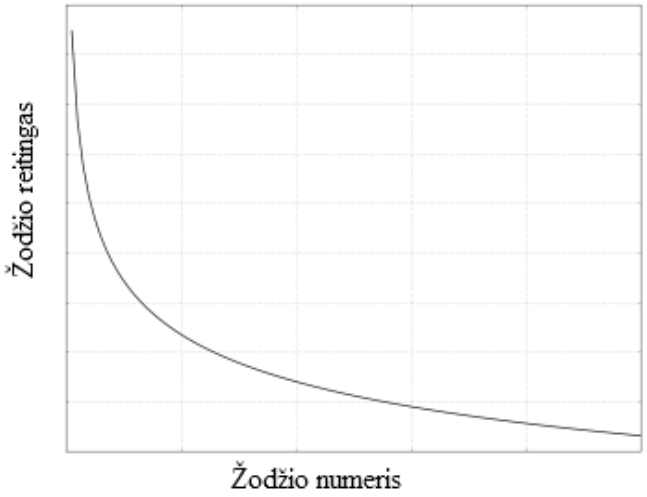
Žingsniai:

1. Suskaidyti tekstą į žodžius,

2. Išskirti kiekvieno žodžio kamieną naudojant Porter [12] algoritmą,
3. Suskaičiuoti žodžių svorius pagal dažnumą,
4. Pasirinkti dažniausiai tekste pasikartojančius raktažodžius..

Pasirinkta realizuoti dažniu grįstą metodą skaičiavimams, kiek kartų žodžius pasikartoja tekste [8]. Toks algoritmai prieš pradedami dirbti realioje aplinkoje turi būti "apmokytas". Yra pateikiami tekstai, kurie yra analizuojami automatiškai. Analizuojant žodis yra normalizuojamas, bei yra skaičiuojama, kiek kartų žodis pasikartojo tekste. Išanalizavus didelį duomenų kiekį, algoritmas gali daryti prielaidą, kaip dažnai žodžiai yra naudojami kalboje. Dažniau naudojami žodžiai yra geresni kandidatai į raktažodžius. Nors anglų kalboje kasmet atsiranda apytiksliai po porą tūkstančių naujų žodžių, tačiau žmonių įpratimai nesikeičia ir egzistuoja žodžiai, kurie yra naudojami dažniau nei kiti.

Siekiant pritaikyti algoritmą prie konkretaus vartotojo, skaičiuojant žodžių svorius yra atsižvelgiama į vartotojo suteiktą reitingą žodžiui. Taip pat pasirenkant geriausius raktažodžius, jei yra žodžių su panašiais svoriais, bus pasirinkti tie, kurių reitingas yra didesnis.



1 pav. Zipf dėsnio grafikas

Galima nupiešti žodžių dažnumo grafiką. Surinkus visus kalboje naudojamus žodžius, bei kiekvienam žodžiui priskyrus jo numerį, bei reitingą. kuris parodys kaip dažnai šis žodis yra naudojamas kalboje. Nupieštas žodžių dažnumo grafikas būtų panašus į „Zipf" dėsnio grafiką, pateiktą 1 paveikslėlyje. „Zipf" dėsnis teigia, kad žodžio reitingas yra atvirkščiai proporcingas žodžio numeriui. Tai parodo, kad tik nedidelė žodžių dalis yra naudojama dažnai [12]. Dažniu grįsti algoritmai remdamiesi šia taisykle gali generuoti pakankamai tikslius rezultatus.

6.2. ŽODŽIO DAŽNUMO – ATVIRKŠTINIO DOKUMENTO DAŽNUMO ALGORITMAS

Žingsniai:
1. Suskaidyti tekstą į žodžius,
2. Išskirti kiekvieno žodžio kamieną naudojant Porter [14] algoritmą,
3. Suskaičiuoti žodžių svorius pagal dažnumą,
4. Išskirti teksto kontekstą,
5. Sumažinti dažnai pasikartojančių žodžių kontekste svorius.

6. Pasirinkti geriausius raktažodžius.

Žodžio dažnumo – atvirkštinio dokumento dažnumo algoritmas skaičiuoja konkretaus žodžio svarbumą konkrečiame tekste. Skaičiavimui yra naudojami du pagrindiniai dydžiai, kuriai sudaro bendrą žodžio svorį. Skaičiuojant kiekvienas žodžio pasikartojimas didina svorį. Tačiau kiekvieno žodžio svoris yra sumažinamas pagal tai, kaip dažnai šis žodis yra naudojamas konkrečioje srityje, kontekste. Tai reiškia, kad kontekste dažniau naudojami žodžiai, turės mažesnį svorį tekste. Kuo svoris yra didesnis, tuo konkretus žodis yra laikomas svarbesnis tekstui ir jį yra prasminga naudoti kaip raktažodį. Toks skaičiavimo būdas padeda valdyti žodžių reikšmingumo skaičiavimą konkrečioje srityje, nes tam tikri žodžiai tiesiog yra naudojami dažniau viename kontekste nei kiti [12]. Šis algoritmas taip pat remiasi „Zipf“ taisykle, tačiau skirtingai nei vien tik dažniu grįstas algoritmas, stengiamasi parinkti tuos raktažodžius, kurie kuris yra aktualūs tekstui, bet ne kontekstui.

Skaičiuoti žodžių dažnumą tf(w, d) tekste galima įvairiai. Paprasčiausiu atveju skaičiuojant žodžių svorį skaičiuojamas vien tik konkretus dažnumas f(w, d), t.y. kiek kartų jis pasikartojo tekste. Egzistuoja kelios šio algoritmo atmainos:

- loginė – kai įvertinamas tik faktas jog žodis pasikartoja tekste:

  tf(w, d) = 1, jei žodis w yra dokumente d, bei 0 priešingu atveju, (1)

- logaritminė – kai skaičiuojant dažnumą, svoris yra logaritmuojamas:

  tf(w, d) = log(f(w, d) +1), (2)

- normalizuotas – kai svoris priklauso nuo to, ne tik kiek kartų žodis panaudojamas tekste, bet ir nuo to, kiek iš vis tekste yra žodžių. Toks skaičiavimo būdas yra naudojamas, kai yra analizuojami didelės apimties tekstai:

$$tf(w,d) = 0{,}5 + \frac{0{,}5 * f(w,d)}{\max\{f(t,d) : t \in d\}} \quad (3)$$

Internetinių dokumentų žymėjimo sistemoje yra naudojamas trečiasis, normalizuotas skaičiavimo būdas, nes šis būdas leidžia tiksliau įvertinti dažnumą esant didelės apimties dokumentams.

Kad algoritmas galėtų sumažinti dažnai kontekste pasitaikančių žodžių svorį, pirmiausia reikia nustatyti teksto kontekstą. Kontekstas pasirenkamas pagal dažniausiai naudojamus tekste žodžius. Konteksto pasirinkimas atliekamas taip:

5. Pasirenkama dešimt daugiausiai naudojamų žodžiai,
6. Ieškomas kontekstas kuriame dominuotų pasirinkti žodžiai,
7. Jei kontekstas nerastas, sumažinamas žodžių skaičius, bei grįžtama į pirmą punktą,
8. Jei nepavyko rasti konteksto yra naudojamas bendras kontekstas.

Žinant kontekstą galima suskaičiuoti atvirkštinį dokumentų dažnumą naudojant formulę:

$$idf(w,D) = \log\frac{N}{1 + |\{d \in D : w \in d\}|} \quad (4)$$

čia:

- N – dokumentų skaičius analizuojamoje srityje,
- $|\{d \in D : w \in d\}|$ – skaičius dokumentų kuriuose žodis d yra rastas.

Žinant abu dydžius, žodžių dažnumą, bei atvirkštinį dokumentų dažnumą, galima apskaičiuoti bendrą dydį:

$$tfidf(w,d,D) = tf(w,d) * idf(w,D) \quad (5)$$

Siekiant pritaikyti algoritmą prie konkretaus vartotojo, skaičiuojant žodžių svorius yra atsižvelgiama į vartotojo suteiktą reitingą žodžiui. Taip pat pasirenkant geriausius raktažodžius, jei yra žodžių su panašiais svoriais, bus pasirinkti tie, kurių reitingas yra didesnis.

6.3. ENTROPIJA GRĮSTAS ALGORITMAS

Žingsniai:

1. Suskaidyti tekstą į sakinius,
2. Suskaidyti tekstą į žodžių junginius,
3. Išskirti kiekvieno žodžio kamieną naudojant Porter [14] algoritmą,
4. Atrinkti iki trisdešimties procentų dažniausiai pasikartojančių junginių,
5. Apskaičiuoti dydį X pasirinktiems junginiams,
6. Apjungti junginius,
7. Pasirinkti raktažodžius.

Algoritmas pirmiausia sudalina tekstą į atskirus sakinius (laikoma, kad sakiniai yra skiriami ženklas: („.“, „?”, „!”). Tuomet atskiruose sakiniuose yra grupuojami žodžiai. Eksperimentai parodė, jog geriausia grupę sudaryti nuo vieno iki keturių žodžių. Taip sugrupuojami visi žodžiai w. Iš visų junginių g, yra atrenkama trisdešimt procentų dažniausiai pasikartojančių žodžių junginių. Atrinkti junginiai laikomi geriausiais kandidatais G. Skaičiavimo laikui sumažinti toliau yra naudojamos tik atrinkti junginiai.
Tuomet kiekvienam žodžiui yra skaičiuojamas dydis X[16]:

$$X(w)^2 = \sum_{g \in G} \frac{(freq(w,g) - n_w p_g)^2}{n_w p_g} \quad (6)$$

$freq(w,g)$ - pakartotinio pasikartojimo dažnumo funkcija,

n_w - kaip dažnai žodis w pasikartojo su geriausiu raktažodžiu g,

p_g - kaip dažnai geriausias raktažodis g pasikartoja tekste.

Apskaičiavus dydį X, tikslumui pagerinti junginiai yra apjungiami:

- pagal panašumą: jeigu junginiai w_1 ir w_2 kartu tekste kartojasi panašiai kaip ir kiti junginiai, jie yra laikomi vienoje grupėje,
- pagal pakartotinį pasikartojimą, jeigu junginiai w_1 ir w_2 tekste pakartotinai kartojasi panašiai, jie yra laikomi toje pačioje grupėje.

Atrinkus raktažodžius ir suskirsčius juos į grupes yra atliekamas rezultatų pasirinkimo žingsnis. Pasirenkami tie junginiai, kurie turi didesnį dydį X, bei dažnai kartojasi tekste. Taip pat pasirenkant jei yra žodžių su panašiais svoriais, bus pasirinkti tie žodžiais, kuriems didesnį reitingą suteikė vartotojas.

6.4. SĄRAŠO ALGORITMAS

Žingsniai:

1. Suskaidyti tekstą į žodžius,
2. Išskirti kiekvieno žodžio kamieną naudojant Porter [14] algoritmą,
3. Apskaičiuoti žodžio miglotąją funkciją dažnumui tekste,
4. Apskaičiuoti žodžio miglotąją funkciją dažnumui ir pakartojantį pasikartojimą naudojant pateiktą žodžių sąrašą,
5. Apskaičiuoti dydį X,
6. Pasirinkti raktažodžius.

Algoritmui iš anksto yra pateikiamas sąrašas žodžių, kurie gali būti naudojami kaip raktažodžiai. Tuomet iš visų pateiktų žodžių yra pasirenkami tie, kurie geriausiai tinka tekstui.

Kiekvienam žodžiui yra skaičiuojama „Takagi-Sugeno (Type-3)"[15] taisyklė rasti dydį X:

$$X(w) = \lambda(w) * (1 - \mu(w)) * \eta(w) \qquad (7)$$

λ – miglotoji funkcija nusakanti žodžio dažnumą tekste,

μ – miglotoji funkcija nusakanti pasikartojimų dažnumą kalboje,

η – miglotoji funkcija nusakanti pakartotinį pasikartojimą tekste.

Algoritmas parinks raktažodžius tik iš anksto paruošto sąrašo, todėl didelis algoritmo tikslumas priklauso nuo gerai paruošto sąrašo.

Rezultatų pasirinkimą įtakoja dydis X, bei kaip dažnai žodis pasikartojo tekste. Didesnė X reikšmė parodo sąrašo žodžio tikslumą tekstui, o didesnis pasikartojimas, jog žodis yra svarbus tekste.

Siekiant pritaikyti algoritmą prie konkretaus vartotojo iš anksto nustatytas žodžių sąrašas yra išplečiamas ankstesniuose vartotojo tekstuose rastais raktažodžiais. Taip pat pasirenkant jei yra žodžių su panašiais svoriais, bus pasirinkti tie žodžiais, kuriems didesnį reitingą suteikė vartotojas.

7. REZULTATAI

Prieš pradedant naudoti algoritmus realioje aplinkoje, juos reikia įvertinti [8]. Įvertinant atlikto eksperimento rezultatus yra lyginama kaip tiksliai algoritmų rasti raktažodžiai sutampa su tais raktažodžiais, kuriuos buvo pasirinkę tekstų autoriai. Tai atlikti buvo pasirinkti įvairių sričių, skirtingų autorių tekstai, straipsniai[17][18]. Atliekant eksperimentą buvo pasirinktos penkios kategorijos: technologijos, karjera, gyvūnai, mokslas, kulinarija. Atsitiktinai pasirinkta po dvidešimt tektų iš kiekvienos kategorijos, kurie turi autorių priskirtus raktažodžius. Šie tekstai buvo analizuojami pasirinktais algoritmais, bei gauti rezultatai buvo lyginami rankiniu būdų.

Automatinės internetinių dokumentų žymėjimo sistemos tikslumui naudojamos standartinės metrikos:

- Tikslumas (angl „precision") – tai skaičius parodantis kuri dalis iš rastų raktažodžių yra teisinga,
- Atkūrimas (anlg "recall") – tai skaičius parodantis kiek iš viso teksto yra rasta teisingų raktažodžių.
- Bendra „F" metrika apjungianti tikslumą, bei atkūrimą. Šios metrikos tikslas yra apjungti abi metrikas, bei gauti bendrą rezultatą apibrėžiantį visą sistemą.

Algoritmo didelis atkūrimas parodo, jog algoritmas gali rasti daugumą reikalingos informacijos tekste, parodo kiekybinę charakteristiką. Didelis tikslumas nurodo, kad rasti duomenys yra teisingi, parodo kokybinę charakteristiką.
Lyginant gautus raktažodžius su tais, kuriuos išskyrė vartotojai yra išskiriami keturi kriterijai:

- teisingas-teigiamas (tt) – kuomet rastas raktažodis yra tinkamas,
- teisingas-neigiamas (tn) – kuomet vartotojo priskirtas raktažodis turi būti nerastas, nes jis nepriklauso tekstui. Šis kriterijus leidžia įvertinti žmogiškąjį faktorių, aptikti neteisingai, per klaidą, pasirinktus autoriaus raktažodžius,
- neteisingas-teigiamas (nt) – kuomet randamas raktažodis, kuris neturėtų būti rastas,
- neteisingas-neigiamas (nn) – kuomet nebuvo rastas vartotojo išskirtas raktažodis.

Sudarius tokią matricą galima apskaičiuoti kiekvieno nagrinėjamo algoritmo charakteringąsias metrikas:

$$Tikslumas = \frac{tt}{tt + nt} \tag{8}$$

$$Atkūrimas = \frac{tt}{tt + nn} \tag{9}$$

$$F = 2 * \frac{Tikslumas + Atkūrimas}{Tikslumas * Atkūrimas} \tag{10}$$

Atlikus eksperimentą gauti rezultatai pateikti pirmoje lentelėje.

LENTELĖ I. METODŲ REZULTATAI

Algoritmas	**Tikslumas**	**Atkūrimas**	**F**
Dažnumo	0,75	0,42	0,54
Atvirkštinio dokumentų dažnumo	0,66	0,47	0,55
Entropijos	0,7	0,4	0,5
Sąrašo	0,74	0,38	0,5

Rezultatai parodo bendrą algoritmų analizavimo kokybę. Visi nagrinėjami algoritmai turi panašius rezultatus. Aukštas tikslumas parodo, kad randami raktažodžiai yra atitinkantys tekstą, bei gali būti naudojami kategorizuojant tekstą. Vidutinis atkūrimas rodo, jog nagrinėjamuose tekstuose net šiek tiek mažiau nei puse žmonių priskirtų raktažodžių sutapo su algoritmų parinktais raktažodžiais.

8. IŠVADOS

Vis didėjantis duomenų kiekis verčia ieškoti naujų sprendimų kaip palengvinti duomenų ieškojimą. Vienas iš būdų duotiems tekstas priskirti raktažodžius. Priskirti raktažodžiai galėtų susieti tekstus, bei taip palengvintų informacijos kategorizavimą. Asmenys kuriems svarbi viena ar kita sritis galėtų lengviau surasti norimą informaciją.

Vartotojai ne visada nori patys apibūdinti savo įkeliamą informaciją patys, todėl vienas iš galimų sprendimų yra automatinis teksto analizavimas išskiriant raktažodžius. Norint, kad sistema galėtų prisitaikyti prie besikeičiančių srities reikalavimų, bei vartotojų poreikių yra naudojami mašininio mokinimosi algoritmai. Papildomai analizuojant, norint gauti kuo tikslesnius rezultatus, gali būti naudojami žodynai, leidžiantys atskirti žodžių sinonimus, bei žodžių grupei priskirti bendrinį žodį.

Atliekant analizę pastebėta, kad dauguma egzistuojančių sprendimų naudoja žodžių dažnių skaičiavimu grįstus algoritmus, jie yra pakankamai greiti, bei tikslūs. Nors mašininio mokinimosi algoritmai gauna tikslesnius rezultatus, jie reikalauja didesnių sąnaudų skaičiavimams.

9. LITERATŪRA

[1] Ronen Feldman,James Sanger: The Text Mining Handbook - Advanced Approaches In Analyzing Unstructured Data. 64-80, 2007.

[2] Florin Gorunesc: Data Mining Concepts, Models and Techniques. 1-38, 2011.

[3] Hércules Antonio do Prado: Edilson Ferneda: Emerging Technologies of Text Mining: Techniques and Applications. 54-75, 2005.

[4] Min Song , Yi-fang Brook Wu Handbook of Research on Text and Web Mining Technologies(2). 733-740, 2005.

[5] Michael W. Berry,Jacob Kogan: Text Mining Applications and Theory. 3-30, 2010.

[6] Xu,Yanchun Zhang,Lin Li: Web Mining and Social Networking Techniques and Applications. 20-86, 2011.

[7] Scott Golder and Bernardo A. Huberman. The structure of collaborative tagging systems. Journal of Information Science, 198–208, 2006.

[8] Mary Hodder. A comparison of how some blog aggregation and RSS search tools work, 2005.

[9] Gilad Mishne. AutoTag: A collaborative approach to automated tag assignment for weblog posts. In WWW ’06: Proceedings of the 15th international World Wide Web conference on Alternate track papers & posters, 2006.

[10] Gilad Mishne. Using blog properties to improve retrieval. In ICWSM 2007 – International Conference on Weblogs and Social Media, 2007.

[11] Nirmala Pudota, Antonina Dattolo, Andrea Baruzzo, Felice Ferrara, Carlo Tasso: Automatic keyphrase extraction and ontology mining for content-based tag recommendation, International Journal of Intelligent Systems, 25(12): 1158-1186, 2010

[12] Stephen Robertson: Understanding Inverse Document Frequency: On theoretical arguments for IDF. Journal of Documentation 60, 503-520, 2004

[13] Federico Michele Facca and Pier Luca Lanzi. Mining interesting knowledge from weblogs: a survey. Data & Knowledge Engineering, 225–241, 2005.

[14] M.F. Porter, 1980, An algorithm for suffix stripping, *Program*, 14(3) pp 130−137.

[15] A. K. Mehran, "Fuzzy model predictive control using Takagi-Sugeno model " in Control, Automation and Systems, 2008. ICCAS 2008. International Conference on, pp. 632 – 637

[16] K. Y. Matsuo and M. Ishizuka, "Keyword Extraction From A Single Document Using Word Co-Occurrence Statistical Information" International Journal on Artificial Intelligence Tools, pp. 157-169 , 2004.

[17] Most Popular Blog Tags - Blog Flux [Online]. Available: http://dir.blogflux.com/tags.php [Accessed: Nov. 27, 2013]

[18] Blog Directory - Technorati [Online]. Available: http://technorati.com/blogs/directory. [Accessed: Nov. 27,2013]