VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
INFORMATIKOS KATEDRA

Vaizdų apdorojimui skirtų giliųjų neuroninių tinklų interpretavimas nuo modelio nepriklausomais metodais

(Interpreting Deep Neural Networks used for Image Processing using Model Agnostic Methods)

Magistro baigiamasis darbas

Atliko: Justas Janušauskas (parašas)

Darbo vadovė: doc. dr. Olga Kurasova (parašas)

Recenzentas: asist. dr. Vytautas Valaitis (parašas)

Vilnius
2019

Santrauka

Magistro baigiamajame darbe nagrinėjama giliųjų neuroninių tinklų, skirtų atpažinti vaizdus, interpretavimo tema. Didesnis dėmesys kreipiamas į nuo mašininio mokymosi modelių nepriklausomus interpretavimo metodus, nes šie metodai yra universalesni ir gali būti taikomi platesniam mašininio mokymosi modelių kiekiui. Atliekant literatūros analizę pagrindžiamas temos aktualumas ir pristatomi aktualūs modelių interpretavimo metodai. Dėl savo aktualumo ir naujumo darbe pasirinktas LIME [RSG16a] metodas. Nagrinėjamos dvi metodo dalys: paaiškinimų sudarymo dalis ir paaiškinimų atrinkimo vartotojui dalis.

Nagrinėjant paaiškinimų sudarymo dalį tiriami segmentacijos algoritmai, kuriuos LIME naudoja modelio paaiškinimas sudaryti. Šie algoritmai skaitiškai įvertinti, pasiūlytos rekomendacijos, kuriuos iš jų naudoti atliekant mašininio mokymosi tyrimą LIME metodu.

Nagrinėjama paaiškinimo atrinkimų dalis, kuri iš visų sudarytų paaiškinimų atrenka tam tikrą kiekį paaiškinimų, kurie geriausiai atspindi modelio veikimą. Paaiškinimų atrinkimo algoritmas pritaikytas darbui su paveikslėliais, klasterizuojant paaiškinimų segmentus. Pasiūlytos alternatyvos algoritmo maksimizuojamai funkcijai, su kuriomis išsprendžiamos problemos, kylančios taikant [RSG16a] straipsnyje siūlomą funkciją.

Raktiniai žodžiai: gilieji neuroniniai tinklai, interpretavimas, vaizdo segmentavimas, klasterizavimas, LIME.

Summary

Master thesis researches the problem of interpreting deep neural networks for image recognition. Particularly interpretation methods which are model agnostic, because they can be applied to broad array of machine learning models. Literature analysis chapter looks into relevant interpretation methods and motivate choosing of this subject. LIME "*Local Interpretable Model-agnostic Explanations"* method [RSG16a] was chosen for further analysis because of its relevancy and wide spread usage. Thesis analysis two parts of this method: segmenting an image into useful segments and extending LIME's *submodular pick* algorithm to be useful with image data.

Thesis evaluates six possible image segmentation algorithms and makes recommendations which ones are most beneficial to use with LIME. S*ubmodular pick* algorithm was extended by adding clustering step to make it useful for image data. Also a new alternative function which values S*ubmodular pick* tries to to maximize was proposed. This alternative function solves some problematic aspects of the original function.

Key words: deep neural networks, interpreting, image segmentation, clustering, LIME.

Turinys

1. Tema ir jos aktualumas

Magistro baigiamajame darbe nagrinėjama giliųjų neuroninių tinklų, skirtų vaizdų apdorojimui, analizės bei vizualizavimo problema. Pastaruoju metu ypač išpopuliarėjo giliųjų dirbtinių neuroninių tinklų taikymas sprendžiant įvairius uždavinius. Sprendžiant vaizdų atpažinimo uždavinius dėl aukšto tikslumo plačiausiai taikomi konvoliuciniai neuroniniai tinklai. Šiems tinklams populiarėjant atsirado poreikis geriau suprasti jų veikimą. Vienas iš skirtumų, kuo neuroniniai tinklai skiriasi nuo kitų populiarių mašininio mokymosi metodų, pavyzdžiui, atraminių vektorių mašinos, yra tai, kad funkcija, kurią optimizuoja neuroninis tinklas priklauso nuo labai didelio kiekio parametrų (svorių). Parametrų kiekis yra ypač didelis sprendžiant vaizdų atpažinimo uždavinį ir gali siekti dešimtis milijonų. Dėl šios savybės pasidaro sudėtinga prognozuoti, kuriomis įvedimo duomenų savybėmis tinklas labiausiai remiasi atlikdamas paveikslėlio atpažinimą. Šioje vietoje vizualizavimas gali padėti geriau suprasti, kaip įvesties duomenys veikia neuroninio tinklo parametrus.

Neuroninių tinklų vizualizavimas aktualus keliais aspektais. Gilūs neuroniniai tinklai vis dar yra „juodosios dėžės“ ir nėra tiksliai žinoma, kodėl jie taip gerai veikia bei kaip būtų galima labiau padidinti jų efektyvumą. Vidinių tinklo sluoksnių vizualizavimas gali būti vienas iš šių problemų sprendimų būdų. Vizualizacijos gali atskleisti, kokias funkcijas atpažinimo procese atlieka vidiniai sluoksniai, kurie sluoksniai bei jų dalys šias funkcijas atlieka kokybiškai, o kurie ne. Vienas iš būdų, kaip gali būti tiriamos šios savybės yra *Deconvnet* metodas. Metodo esmė – nustatyti, kurie paveikslėlio pikseliai labiausiai aktyvavo tam tikro vidinio sluoksnio neuronus. Įvesčiai einant per sluoksnius duomenų dimensijos yra mažinamos ir prarandama informacija, *Deconvnet* [ZF13] metodas stengiasi, kiek įmanoma tiksliai rekonstruoti prarastą informaciją, taip atkuriant originalias dimensijas. Remiantis šiuo metodu buvo nustatyta, kurie tinklo neuronai kreipia dėmesį į objektus, o kurie veikia neteisingai – kreipia dėmesį ne į objektus, o į foną, pavyzdžiui, žolę fone. Šio metodo autoriams pavyko ne tik nustatyti, į kokias savybes tinklas atkreipia dėmesį, bet ir remiantis rezultatais šį tinklą patobulinti. Pastebėta, kad pirmajame sluoksnyje analizuojamas tinklas labiausiai pritaikytas atpažinti aukšto ir žemo dažnio informaciją. Antrajame sluoksnyje pastebėta, kad atpažįstami artefaktai stipriai persidengia. Atsižvelgiant į šią informaciją buvo sumažinti pirmo sluoksnio filtrų dydžiai bei žingsnių dydis [ZF13].

Neuroninį tinklą vizualiai galime analizuoti keliais aspektais:

- Jei norime suprasti atskiras savybes galime ieškoti įvesties, su kuria neurono išvestis įgyja aukštas reikšmes – tirti individualius neuronus ar konvoliucinio tinklo filtrus [OMS17].
- Visą konvoliucinį sluoksnį galime vizualizuoti tokiais metodais kaip *DeepDream*. Šis metodas leidžia iš įprasto ar atsitiktinio triukšmo paveikslėlio sugeneruoti tokį vaizdą, kuris maksimaliai aktyvuoja pasirinktą sluoksnį ar dalį sluoksnio filtrų. Metodas taikomas siekiant nustatyti, kokių savybių neuroninis tinklas ieško paveikslėlyje, pavyzdžiui, bandydamas atpažinti svarmenį, tinklas ieško ir jį laikančios rankos [MOT15].

Neuroninių duomenų vizualizacija padeda ne tik suprasti, kaip veikia gilūs neuroniniai tinklai, bet ir nustatyti atvejus, kuriems esant jie veikia prastai. Atliekant vidinių sluoksnių tyrimus buvo sugebėta paveikslėlius, kuriuos tinklas atpažįsta teisingai, pakeisti taip, kad žmogaus akims jie atrodytų nepakitę, tačiau konvoliucinis neuroninis tinklas jiems priskirtų klaidingą klasę [SZS+13]. Kitas tyrimas sukonstravo paveikslėlius, kuriuos gilūs neuroniniai tinklai, apmokyti *ImageNet* [DDS+09] duomenimis, atpažįsta su labai aukštu tikslumu (>= 99,6 %), tačiau žmogui šie paveikslėliai nėra atpažįstami [NYC15].

Dauguma darbe nagrinėjamų metodų buvo pristatyti 2015 metais ar anksčiau. Daugumą jų atlikti tiriant *AlexNet* [KSH12] ar *GoogLeNet* [SLJ+15] architektūros tinklus. O apmokymui naudoti *MNIST* [LCB99] arba *ImageNet* [DDS+09] duomenys. Per paskutinius kelis metus atsirado naujos architektūros, kurios idėjiškai skiriasi nuo senesnių. *Resnet* [HZS+16] architektūros tinkle sluoksniai jungiami ne tik paeiliui, bet ir daromi šuoliai bei sujungiami tolimesni sluoksniai.

Kitas naujos architektūros tinklas – *Capsule Net* [SFH17] architektūros tinklas. Jis fundamentaliai skiriasi nuo klasikinių konvoliucinių tinklų. Šios architektūros tinkle nėra sujungimo (angl. *pooling*) sluoksnio, populiariausios *max pooling* ir *average pooling* funkcijos, neuronai jungiami „kapsulės“ struktūrose. Dėl didelio architektūrinio skirtumo, kai kurių vizualizavimo metodų taikymas kito modelio tinkluose nėra trivialus uždavinys. Nėra visiškai aišku ar esami metodai tiks šiam tinklui analizuoti.

Taip pat dauguma vizualizacijos metodų buvo kuriami analizuoti konvoliucinius tinklus, skirtus vaizdų klasifikavimo uždaviniui. Pastaruoju metu vis daugiau dėmesio skiriama ne tik objekto klasifikacijos uždaviniui, bet ir jo radimui paveikslėlyje. Nėra aišku ar metodai, kurie duodavo reikšmingus rezultatus analizuojant tinklus, skirtus spręsti klasifikavimo uždavinį, duos interpretuojamus rezultatus pritaikius juos analizuojat tinklus sprendžiančius objektų aptikimo ir

klasifikacijos uždavinį. Pavyzdžiui, YOLO [RF17] architektūros tinkle objekto nustatymas ir atpažinimas atliekamas viename neuroniniame tinkle, o ne atskirose dalyse. Tai reiškia, kad neuronai turi išmokti ir objekto paieškos logiką, nėra aišku ar vizualizavimo metodai skirti klasifikavimo uždaviniui bus interpretuojami ir šios architektūros tinklui.

1.1. Tikslas ir uždaviniai

Magistro baigiamojo darbo tikslas: ištyrus ir palyginus esamus metodus giliųjų neuroninių tinklų vizualizacijai, identifikuoti kurių metodų funkcionalumą galima praplėsti ir šį praplėtimą įgyvendinti. Atlikus literatūros analizę, praplėtimui, dėl savo aktualumo, pasirinktas LIME metodo [RSG16a] dalies, skirtos paaiškinimų atrinkimo vartotojui, praplėtimas. Ši dalis originamale straipsnyje pritaikyta teksto klasifikacijos uždaviniui, kuomet paaiškinimai yra žodžiai. Magistriniame darbe atliktos metodo modifikacijos darbui su paveikslėliais.

Darbo uždaviniai:

1. Atlikti egzistuojančių metodų, gebančių vizualizuoti mašininio mokymosi modelius, literatūros analizę, surasti ir identifikuoti atvirus uždavinius.
2. Pasirinkti vieną iš identifikuotų uždavinių ir įgyvendinti metodo modifikacijas.
3. Eksperimentiškai ištirti atliktą modifikaciją, atrasti geriausius modifikacijos parametrus.

2. Literatūros analizė

Literatūros analizei straipsniai atsirinkti atsižvelgiant į jų naujumą ir aktualumą mašininio mokymosi interpretavimo srityje. Pirmasis literatūros analizės poskyris apžvelgia vieną iš aspektų, kuris daro mašininio mokymosi interpretavimo temą aktualia. Antrasis poskyris „Mašininio mokymosi interpretavimas“ analizuoja, kaip interpretavimo metodus būtų galima skirstyti į kategorijas ir kokiais atributais juos būtų galima įvertinti. Trečiajame poskyryje pateikiami metodai, kurie modelius tiria per įvesties-išvesties sąryšius, todėl gali būti taikomi platesniam spektrui mašininio mokymosi modelių. Ketvirtajame poskyryje aptariami metodai giliųjų konvoliucinių neuroninių tinklų vizualizavimui. Trečiajame ir ketvirtajame poskyriuose pristatomi metodų veikimo principai, taikymo ypatumai, stipriosios ir silpnosios savybės.

2.1. Bendrasis duomenų apsaugos reglamentas

Nuo 2018 metų gegužės mėnesio įsigaliojęs Europos Sąjungos bendrasis duomenų apsaugos reglamentas (BDAR) paliečia ir mašininio mokymosi algoritmus. 22 reglamento straipsnyje rašoma apie automatizuotą sprendimų priėmimą ir profiliavimą. Profiliavimas šiame dokumente apibrėžiamas kaip bet koks asmens duomenų apdorojimas siekiant įvertinti tam tikrus su asmeniu susijusius aspektus. 13 ir 14 reglamento straipsniuose [EPTR06] minima, kad duomenų subjektas turi teisę gauti suprantamą, paaiškinančią profiliavimo logiką, informaciją. Iš to išplaukia poreikis interpretuoti mašininio mokymosi algoritmus. Algoritmus interpretuoti pagal BDAR reikia ne tik dėl būtinybės duomenų subjektui paaiškinti apie profiliavimo logiką, bet ir užtikrinti, kad taikomi algoritmai nediskriminuoja žmonių pagal asmens duomenis. Straipsnyje [GF17] teigiama, kad mašininio mokymosi modelių rezultatai priklauso nuo duomenų, kurie yra gauti iš visuomenės, ir, jei visuomenėje yra diskriminacija arba nelygybė požymių šie aspektai atsispindės duomenyse, kuriais remiantis bus apmokytas klasifikatorius. Tarkime, kuriame modelį, kurio tikslas yra gatvėje atpažinti žmones. Svarbu įsitikinti, kad modelis geba identifikuoti mažiau visuomenėje reprezentuotas grupes, tokias kaip neįgalieji asmenys vežimėliuose, ir identifikuojami visų rasių žmonės. Straipsnyje pažymima, kad mokslas, tiriantis įvesties duomenų įtaką „juodosios dėžės“ mašininio mokymosi algoritmams yra perspektyvus.

2.2. Mašininio mokymosi interpretavimas

Augant mašininio mokymosi taikymų spektrui mums svarbu tampa ne tik algoritmų efektyvumas, bet ir tokie aspektai kaip saugumas, nediskriminacija, teisė į paaiškinimą

[SHG+15]. Nors šie aspektai yra svarbūs, priešingai nei modelio tikslumą, juos sunku kiekybiškai pamatuoti. Mašininio mokymo bendruomenėje nėra aiškaus susitarimo, apibūdinančio interpretavimą ir interpretavimo matavimą. Straipsnyje [DK17] siūloma interpretavimą suskirstyti į kelias grupes:

- Taikymu grįstas vertinimas. Turimas algoritmas pritaikomas praktiškai, pvz: jei algoritmas skirtas nustatyti ligą, jis taikomas tikrų gydytojų, tikriems pacientais.
- Žmonėmis grįstas vertinimas. Algoritmu besinaudojantys žmonės nebūtinai yra ekspertai, kurie naudosis galutiniu produktu. Šis vertinimo būdas reikalauja mažiau resursų už pirmąjį, tačiau labiau tinka paprastesnėms modelio savybėms įvertinti.
- Funkcionalumu grįstas vertinimas. Kainuoja mažiausiai laiko ir resursų. Modelis vertinamas automatiškai, tačiau sunku apibrėžti kriterijus, kuriais jis bus vertinamas.

Straipsnio autoriai taip pat pažymi svarbą tiksliai apibrėžti interpretacijos rezultato struktūrą:

- Ar interpretacija sudaryta iš įvesties duomenų ar iš išvestinio objekto (pvz.: žodis „kėdė“ gali reprezentuoti pikselius, sudarančius kėdės objektą)?
- Koks elementų kiekis sudaro interpretaciją?
- Šių elementų hierarchija ar kita struktūrą interpretacijoje.
- Interpretacijos stochastiškumas.

Straipsnyje pabrėžiamas ne tik siekis interpretuoti mašininio mokymo algoritmus, bet ir svarba šias interpretacijas apibrėžti kuo tiksliau bei padaryti jas kuo labiau kiekybiškai vertinamas.

2.3. Nuo modelio nepriklausomi metodai

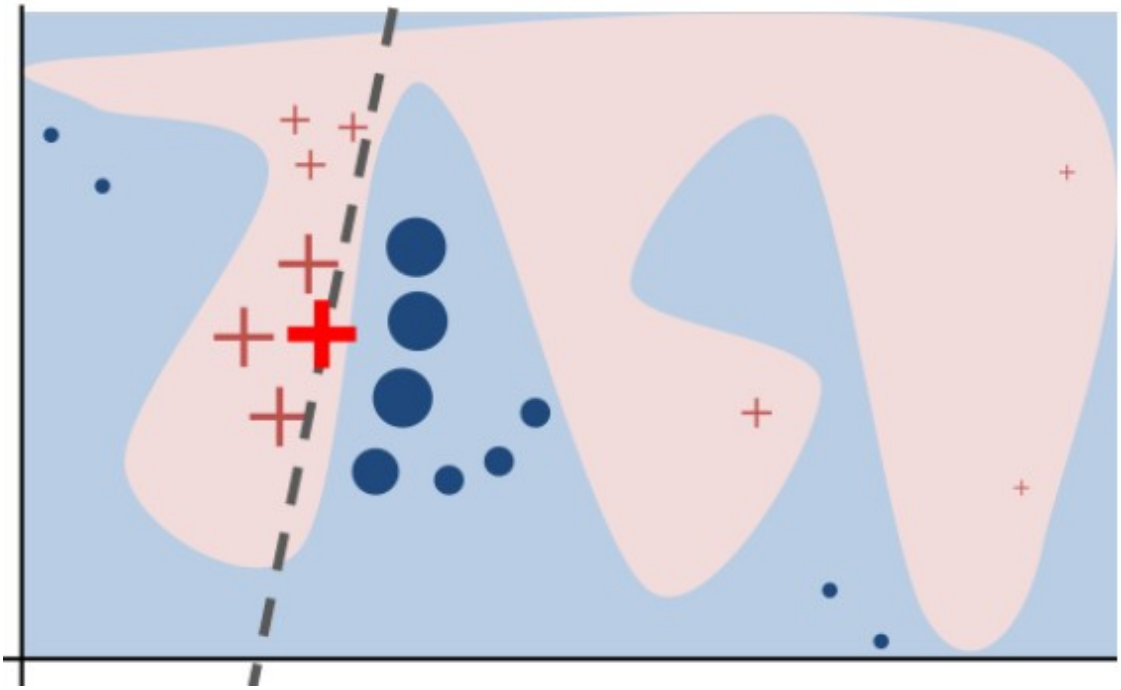
Šie metodai žvelgia į analizuojamą modelį kaip į juodąją dėžę. Modeliai analizuojami tiriant sąryšius tarp įvesties ir išvesties, o ne į tai kas vyksta modelio vidiniuose procesuose, kuriuose yra apdorojami įvesties duomenys. Didelis šių metodų privalumas yra tai, kad šie analizės metodai gali būti taikomi plačiam spektrui mašininio mokymosi modelių. Dekonvoliucijas galime taikyti tik dirbtiniams konvoliuciniams neuroniniams tinklams. Tačiau „juodosios dėžės“ modeliai gali būti taikomi tiek konvoliucinams tinklams, tiek ir kitokioms architektūroms.

2.3.1. LIME

Vienas iš šio šipo metodų yra LIME algoritmas. LIME yra akronimas angliško pavadinimo „*Local Interpretable Model-agnostic Explanations*“, lietuviškai – „Lokaliai interpretuojami nuo modelio nepriklausomi paaiškinimai“. Kaip ir teigia pavadinimas, šis metodas neprisiriša prie

konkretaus modelio. Paaiškinimai, sugeneruoti šiuo algoritmu, yra lokaliai interpretuojami, t. y. jie kalba apie konkrečius įvesties pavyzdžius, o ne apie bendrą nagrinėjamo modelio veikimą. Tačiau šie lokalūs paaiškinimai gali padėti sudaryti intuityvią nuomonę (globalią interpretaciją) ir apie bendrą modelio veikimą [RSG16a].

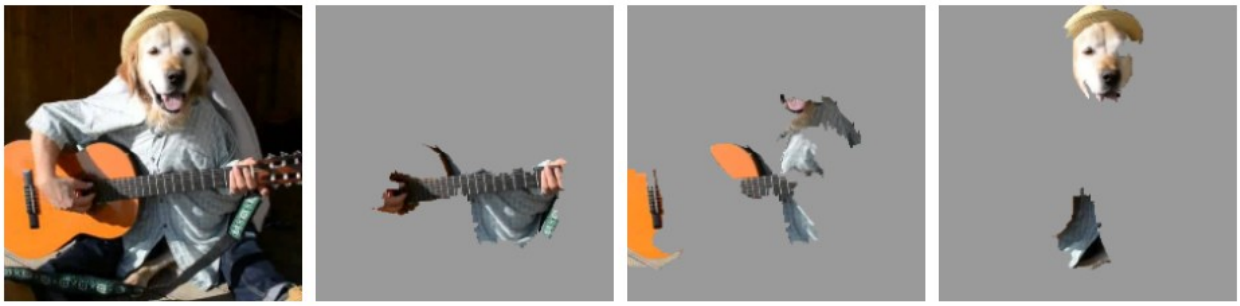
LIME algoritmo supaprastintas veikimo pavyzdys pateikiamas 1 pav.. Šiame paveikslėlyje rožinės ir mėlynos sritys žymi netiesinę juodosios dėžės modelio išmoktą funkciją. LIME algoritmas analizuoja įvesties pavyzdį, pažymėtą raudonu pliusu. Netiesinė funkcija raudonojo pliuso regione aproksimuojama tiese. Todėl gauti paaiškinimai apibūdina modelio veikimą aplink nagrinėjamą pavyzdį lokaliai, bet ne globaliai.



1 pav. Supaprastintas LIME veikimo pavyzdys. Žydros ir rožinės zonos žymi juodosios dėžės modelio išmoktą funkciją f, kuri negali būti tiesiškai aproksimuota. Raudonas pliusas – analizuojamas pavyzdys x. LIME atsirenka x kaimynus ir pagal juos aproksimuoja funkciją f tiesiškai, lokaliai pavyzdžio x erdvėje. Šaltinis: [RSG16a]

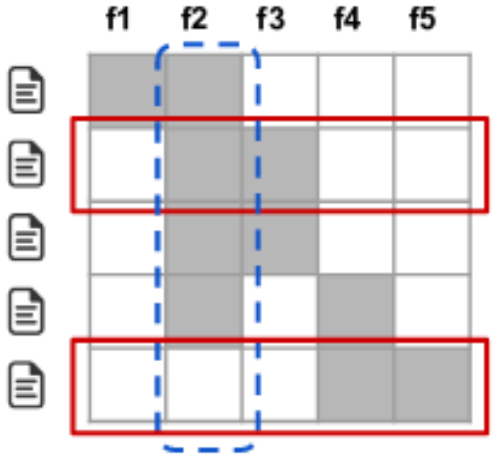
Dėl savo savybių LIME algoritmas tinka nagrinėti platų spektrą modelių, tačiau literatūros analizei aktualiausias LIME taikymas vaizdo klasifikaciją atliekantiems giliems neuroniniams tinklams tirti. Analizės rezultatui atvaizduoti pasitelkiami „super pikseliai“. „Super pikselis“ yra grupė šalia išsidėsčiusių, panašias reikšmes turinčių pikselių. „Super pikseliai“, nuo kurių labiausiai priklauso paveikslėlio klasifikacijos rezultatas parodomi, o likusieji nuspalvinami pilkai. (2 pav.) pateiktame pavyzdyje modelis atliko klaidingą klasifikaciją – didžiausią įvertį turėjo klasė „elektrinė gitara“, nors tikroji klasė buvo „akustinė gitara“. Tačiau paaiškinimas

suteikia pasitikėjimo modelio kokybe, o klaidinga klasifikacija atlikta ne be pagrindo.



2 pav. LIME paaiškinimų pavyzdys. Kairėje – analizuojamas paveikslėlis. Toliau eilės tvarka modelio spėjimo paaiškinimai: elektrinė gitara, akustinė gitara, šuo. Šaltinis: [RSG16a]

Peržiūrėti kiekvieno pavyzdžio paaiškinimus gali užtrukti daug laiko, ypač jei turimas didelis duomenų kiekis. Taip pat daugelis pavyzdžių gali būti panašūs ir turėti vienodus paaiškinimus. Nors paveikslėlių paaiškinimai gali duoti įžvalgų, šie paveikslėliai turi būti parinkti protingai, nes vartotojai gali neturėti laiko peržiūrėti didelio kiekio pavyzdžių ir paaiškinimų. Tokie įrankiai kaip *Modeltracker* [ACD+15] sprendžia panašią problemą, tačiau jis ir panašūs įrankiai atrenka neapdorotus duomenis, paaiškinimai parenkant šiuos pavyzdžius nedalyvauja. [RSG16a] pateikiamas algoritmas pritaikytas LIME metodui, kuris pavyzdžius parenka atsižvelgiant į paaiškinimus. Parenkant pavyzdžius svarbiausiais paaiškinimais laikome tuos kurie, dalyvauja paaiškinant didžiausią kiekį pavyzdžių. Taip pat pavyzdžiai renkami taip, kad paaiškinimai kuo mažiau pasikartotų. 3 pav. iliustruoja algoritmo veikimo principus, dėl paprastumo paaiškinimų reikšmės yra dvejetainės. Paaiškinimas f2 laikomas svarbiausiu, nes pasikartoja dažniausiai. Kadangi nenorima apkrauti vartotojo per dideliu duomenų kiekiu, pasirinkus pavyzdį antroje eilutėje, trečioje eilutėje esantis pavyzdys nebus pasirinktas, nes vartotojas jau matė paaiškinimus f2 ir f3. Antros ir paskutinės eilutės pavyzdžiai padengia beveik visus paaiškinimus. Straipsnyje teigiama, kad taikant šį algoritmą paveikslėliams, reikalingas būdas palyginti super-pikselius skirtinguose paveikslėliuose, pavyzdžiui pagal spalvų histogramas, tačiau autoriai tolimesnį šios problemos tyrinėjimą palieka ateities darbams.



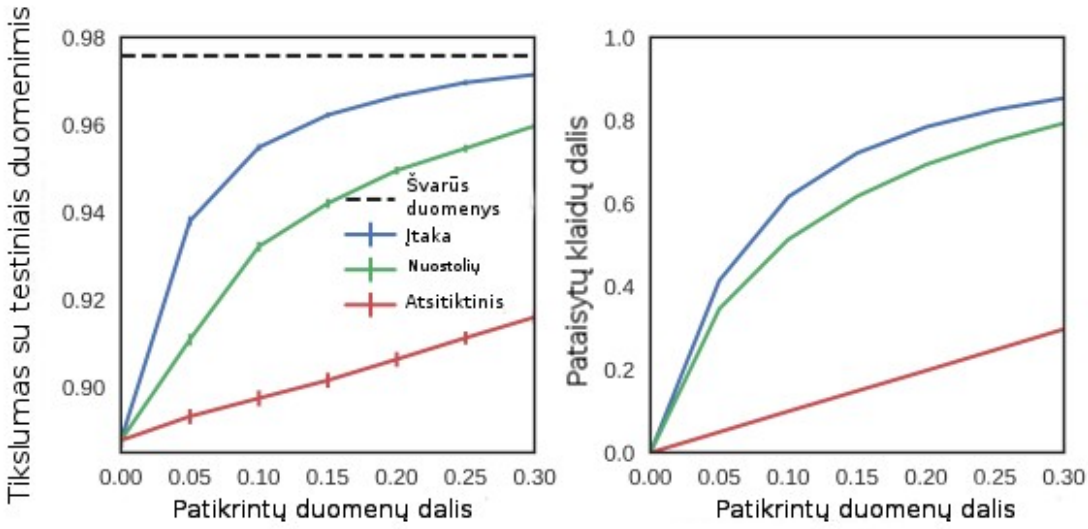
3 pav. Pavyzdžių parinkimo iliustracija. Vertikaliai – pavyzdžiai, stulpeliuose – paaiškinimai. Mėlynas punktyras – aktualiausias paaiškinimas. Algoritmo pasirinktos eilutės apibrauktos raudonai. Šaltinis: [RSG16a]

2.3.2. Įtakos funkcijos

Įtakos funkcijos savo šaknis turi statistikos srityje. Šis algoritmas tiria kaip pasikeistų modelio parametrai, jeigu jis būtų mokomas tam tikram mokymo pavyzdžiui suteikiant didesnį ar mažesnį svorį (laikant tą pavyzdį svarbesniu ar mažiau svarbiu), arba nežymiai pakeičiant mokymo pavyzdžio reikšmes. Šis metodas gali padėti tyrėjams suprasti modelio elgseną, ištaisyti modelio klaidas, nustatyti ar mokymo duomenų rinkinys nėra blogas ir sukurti modelį apgaunančias atakas [KL17].

Įtakos funkcijos sugeba nurodyti, kurie mokymo duomenys turėjo didžiausią reikšmę atliekant klasės identifikavimą. Iš šios informacijos mes galime bandyti nuspėti, kodėl modelis priėmė tam tikrą sprendimą. [KL17] straipsnyje pateikiami du mašininio mokymosi modeliai, kurie atlieka teisingą paveikslėlio klasifikaciją, tačiau iš informacijos gautos įtakos funkcijų dėka galime pamatyti, kad vienas iš jų yra geresnis. Pirmasis modelis – Atraminių vektorių mašina, naudojanti radialines funkcijas (RBF SVM), antrasis modelis – „*Inception v3*“ gilus konvoliucinis neuroninis tinklas [SVI+16]. Abu modeliai apmokyti su dalimi ImageNet [RDS+15] duomenų rinkinio. Modeliai išmoko atpažinti dviejų klasių paveikslėlius, kuriuose yra žuvys ir šunys. Įtakos funkcijų rezultatas (4 pav.) atskleidžia nemažai informacijos apie skirtumus tarp šių dviejų modelių. Iš karto pastebime, kad RBF SVM modeliui didžiausią įtaką turėjo paveikslėliai, kurių pikselių reikšmės buvo artimos testiniui paveikslėliui. *Inception* modelio atveju įtakos turėjo ir paveikslėliai kurie pikselių reikšmių erdvėje buvo nutolę nuo testinio paveikslėlio. Taip pat, RBF SVM atveju klasės „žuvis“ paveikslėliai darė teigiamą įtaką

testinio paveikslėlio klasifikacijos tikslumui, o klasės „šuo“ darė neigiamą įtaką. *Inception* modeliui klasės „šuo“ paveikslėliai buvo naudingi, t. y. priimdamas sprendimą modelis atsižvelgia ne tik į klasių panašumus, bet ir į skirtumus. Taip pat matome, kad *Inception* modelis išmoko atpažinti *žuvies klouno* rūšies išvaizdos savybes, ir būtent mokymo duomenys su šios rūšies žuvimi labiausiai padėjo teisingai klasifikuoti testinį paveikslėlį. RBF SVM modelis šių savybių išmokti nesugebėjo, klasifikacija atlikta labiau remiantis bendru spalviniu panašumu.



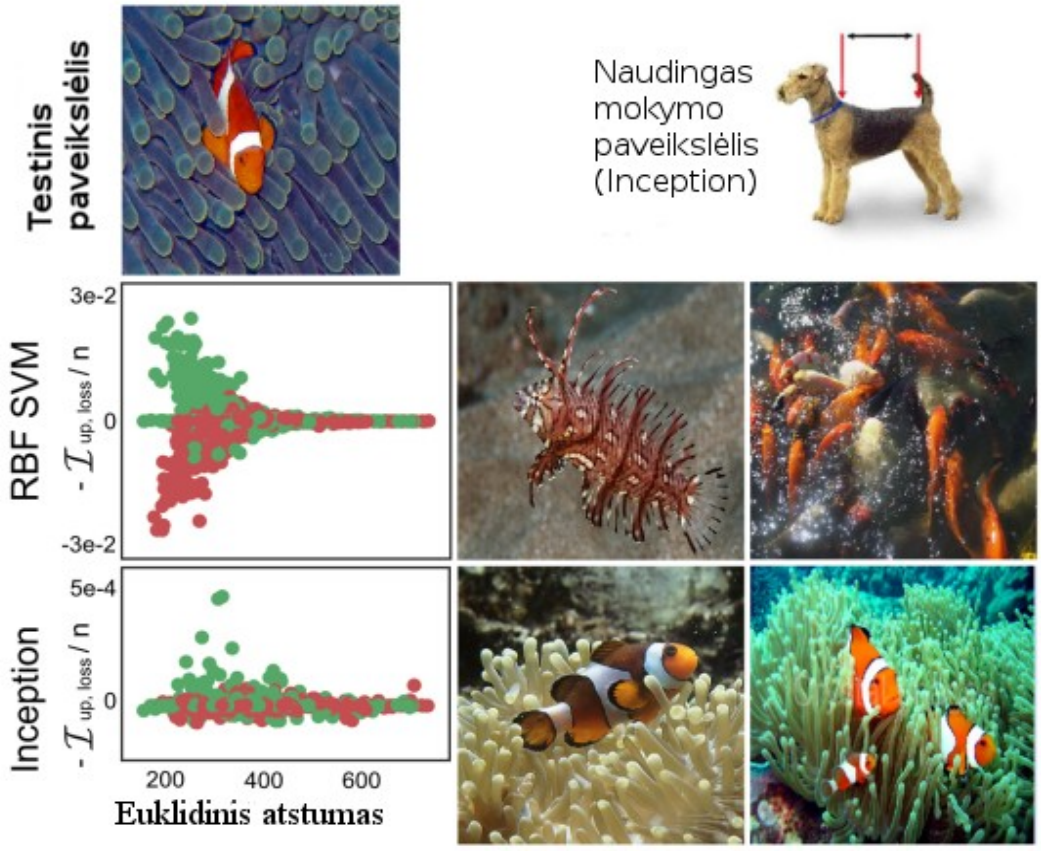
4 pav. Klaidingų pavyzdžių radimas. Kairysis grafikas nurodo kaip peržiūrint duomenis keičiasi modelio tikslumas. Dešinysis – kokia dalis klaidingų duomenų ištaisyta. Raudona kreivė – atsitiktinis pasirinkimas, žalia – pasirinkimas pagal didžiausią nuostolių funkcijos reikšmę, mėlyna – pasirinkimas pagal įtakos funkciją. Šaltinis: [KL17]

Įtakos funkcijos taip pat gali būti naudojamos kuriant atakas prieš modelį. Šios atakos pakeičia paveikslėlį taip, kad žmogus pasikeitimo jame nepastebi, tačiau mašininio mokymosi modelis jį atpažįsta klaidingai ir su aukštu pasitikėjimu. Tai įmanoma dėl to, kad įtakos funkcijos rezultatas gali mums nurodyti, kaip keisti tam tikrų pikselių reikšmes tam, kad padidinti tiriamo modelio nuostolių funkcijos reikšmę.

Šis metodas taip pat gali padėti nustatyti, ar mokymo ir testavimo duomenų rinkinių pasiskirstymai sutampa. Jei šios distribucijos nesutampa, modeliai veikiantys aukštu tikslumu su mokymo duomenimis gali prastai veikti su testavimo duomenimis [DBC+10]. Šiuo atveju mažas tikslumas yra susijęs būtent su duomenų rinkiniais, o ne su persimokymu ar netikslumu mokymo duomenyse. [KL17] pateikiamas pavyzdys, kuriame sukuriamas neatitikimas tarp testavimo ir mokymų distribucijų. Pritaikius įtakos funkcijas, klaidingai klasifikuotam testinių duomenų pavyzdžiui, mokymo duomenys, atsakingi už neatitikimą tarp duomenų aibių turėjo 30-40 kartų

didesnes įtakos reikšmes už likusius duomenis.

Dar vienas įtakos funkcijų panaudojimo atvejis yra identifikavimas mokymo pavyzdžių, kuriems buvo priskirta klaidinga kategorija. Duomenų rinkinių rinktų iš realaus pasaulio, ypač su savanorių pagalba, kategorijos gali būti sužymėtos klaidingai [FV14]. Atliktame eksperimente buvo pakeista dalis duomenų rinkinio klasių į klaidingas. Mažiausiai mokymo duomenų peržiūrėti, norint ištaisyti klaidingas klases, pavyko panaudojus įtakos funkcijas (5 pav.).



5 pav. Įtakos funkcijų grafikas. Kairėje viršuje – analizuojamas testinis paveikslėlis. Dešinėje viršuje – šuns paveikslėlis, kuris Inception tinklui padėjo teisingai klasifikuoti testinį žuvies paveikslėlį. Kairėje apačioje – įtakos pasiskirstymas pagal pavyzdžių pikselių euklidinį nuotolį. Žali taškai – žuvys, raudoni – šunys. Dešinėje apačioje – paveikslėliai, kurie modeliams labiausiai (turėjo didžiausia įtakos reikšmę) padėjo atpažinti testinį paveikslėlį. Šaltinis: [KL17]

2.3.3. Modeliams agnostinių interpretacijų privalumai

Galime išskirti kelis privalumus kuriais juodąsias dėžes interpretuojantys metodai lenkia modeliams specializuotus interpretavimo metodus.

Šie interpretavimo metodai mums suteikia lankstumo kuriant modelį, jeigu mes norime kartu su rezultatu pateikti ir interpretaciją. Galime didinti gilaus neuroninio tinklo sluoksnių

kiekį nekeičiant to, kaip atliekama interpretacija. Šiuo atveju galime rasti balansą tarp modelio sudėtingumo ir interpretacijų kokybės.

Taip pat šie metodai suteikia lankstumą interpretacijų lygmeniu. Skirtingi vartotojai gali turėti skirtingų poreikių interpretacijų sudėtingumui. Interpretuojami modeliai dažniausiai gali pateikti tik vieno tipo interpretacijas. Atskirdami interpretaciją nuo modelio galime skirtingiems vartotojams pateikti skirtingo sudėtingumo interpretacijas [RSG16b].

Interpretacijas taip pat galime išreikšti ne per mokymo duomenų atributus. Garso ar vaizdo duomenų rinkiniuose sunku rasti interpretuojamų atributų. Tačiau modeliams agnostiniai modeliai gali pateikti interpretacijas iš išvestinių atributų. Pavyzdžiui, ne atskirų pikselių, o pikselių regionų.

Naudojant šiuos metodus lengva pakeisti mašininio mokymo modelį, nekeičiant pateikiamų interpretacijų. Jeigu ankščiau naudotas atraminių vektorių modelis pakeistas į gilų dirbtinį neuroninį tinklą, nereikės keisti modelio, kuriuo atliekame interpretacijas, taip pat modelio rezultatas išliks tokio paties formato. Dėl tos pačios priežasties tyrėjams lengviau lyginti du skirtingus modelius. Jeigu interpretuojame du skirtingos architektūros modelius, galime taikyti tą patį interpretavimo metodą, taip gaudami vienodo formato interpretacijas [RSG16b].

2.3.4. Modeliams agnostinių interpretacijų trūkumai

Dauguma šių modelių paaiškinimus pateikia lokaliu lygmeniu. Jeigu modelis labai kompleksiškas, globalų supratimą apie modelį pasiekti gali būti sunku. Dėl to kai kurios lokalios interpretacijos gali prieštarauti vienai kitai, nes globaliai modelis tą patį duomenų atributą gali apdoroti skirtingai.

Kai kuriuose srityse gali būti reikalaujama labai tikslaus modelio veikimo paaiškinimo. Tuomet, kai modelio interpretavimas yra didesnis prioritetas nei modelio tikslumas, geriau naudoti tiesiogiai interpretuojamus modelius.

Modelį papildomai papildyti duomenimis iš vartotojų grįžtamojo ryšio vis dar yra lengviau jeigu naudojamas baltosios-dėžės modelis. Grįžtamojo ryšio inkorporavimas į modelį išlieka sudėtinga mokslinių tyrimų kryptis [RSG16b].

2.5. Literatūros analizės rezultatai ir išvados

Padarytos išvados, kad interpretavimo metodas, kurį geriausia pasirinkti atliekant tyrimus, priklauso nuo taikymo specifikos. Jei prioritetas yra veikimo greitis, tuomet gali būti naudingas *VisualBackProp* [BCC+16] metodas. Jeigu planuojama eksperimentuoti plačiu modelių spektru, tuomet aktualu taikyti nuo modelių nepriklausomus metodus [RSG16a, KL17]. Giliems

konvoliuciniams tinklams prasminga naudoti dekonvuliacijos [ZF13, ZTF11] metodus.

Nagrinėjant literatūrą susipažinta su atviromis mašininio mokymosi modelių interpretavimo srities problemomis. Didelių skaičiavimo pajėgumų reikalavimo problema [ZCA+17], grįžtamojo ryšio problema [RSG16b], LIME metodo pavyzdžių atrinkimo paveikslėliams problema [RSG16a].

Būtent ši pavyzdžių atrinkimo problema ir pasirinkta nagrinėti tolimesniame darbe. Problemos aktualumas grindžiamas tuo, kad atsitiktinės peržiūros metu matomos interpretacijos gali dažnai kartotis. Jei peržiūrima tik dalis duomenų, unikalios interpretacijos gali būti nepateiktos. Praplėtus šį metodą darbui su paveikslėliais, vartotojui bus pateikiamos aktualiausios ir mažiausiai besikartojančios interpretacijos.

3. Segmentacijų tyrimas

Taikant LIME metodą paveikslėliams svarbu tinkama paveikslėlių segmentacija. Taikant LIME, modeliams skirtiems teksto klasifikavimui užduočiai spręsti, modelio rezultatai paaiškinami tam tikrų žodžių buvimu. Paveikslėlių atveju naudojami paveikslėlio segmentai. Tačiau priešingai nei žodžiai kurie yra fiksuoti, segmentacijos gali kisti ir priklauso nuo algoritmo, kuriuo jos parenkamos. LIME metodą aprašančiame straipsnyje konkretūs segmentacijos algoritmai nenurodomi. Bibliotekoje kurioje realizuotas LIME metodas pateikiami trys galimi segmentacijos algoritmai. Tačiau nėra pateikiama informacija apie skirtingų algoritmų veikimo įtaką LIME metodui. Šiame skyriuje bus palyginti LIME bibliotekoje realizuoti segmentavimo algoritmai. Taip pat bus patikrintas LIME metodo veikimas su kitais segmentavimo algoritmais. LIME bibliotekoje realizuoti segmentavimo algoritmai remiasi spalvine informacija. Gali būti verta išbandyti segmentavimo algoritmus, kurie segmentacijas sudaro pagal objektų esančių paveikslėlyje kraštus, o ne pagal spalvinę informaciją.

3.1. LIME naudojami segmentavimo algoritmai

LIME bibliotekoje yra suteikiama galimybę naudotis trimis segmentacijų algoritmais: *quickshift*[VS08], *Felzenszwalb*[FH04] ir *SLIC* [ASS+12]. Šie algoritmai priklauso segmentacijų algoritmų grupei vadinamai *superpikseliais*. Šie algoritmai suskaido paveikslėlį į didelį kiekį segmentų. Atskiri segmentai neišskiria konkrečių objektų, nes segmentavimas neatsižvelgia į semantinę informaciją.

Felzenszwalb algoritmui perduodamas mastelio parametras, kuris daro įtaką segmentacijų dydžiui. Segmentacių dydis ir kiekis gali skirtis priklausomai nuo lokalaus kontrasto

paveikslėlyje.

Quickshift algoritmas pristatytas [VS08] straipsnyje. Algoritmas priklauso lokalios paieškos algoritmų šeimai. Algoritmas apdoroja spalvotus paveikslėlius, todėl įvestis yra 5 dimensijų – trys dimensijos nusako pikselio spalvą, dvi dimensijos – pikselio vietą paveikslėlyje. Viena iš pagrindinių šio algoritmo savybių yra tai, kad sugeneruojamos hierarchinės segmentacijos. Mes patys galime pasirinkti su kurio lygio segmentacijomis norime dirbti.

SLIC (angl .Simple Linear Iterative Clustering) algoritmas remiasi k-vidurkių algoritmu. Kaip parametrai algoritmui perduodamai *k-means* grupių kiekis, nuo kurio priklauso segmentacijų kiekis ir kompaktiškumo parametras kuris nurodo ar atliekant segmentaciją didesnę įtaką turės atstumas tarp spalvų ar atstumas tarp pikselių.

6 paveikslėlyje pateikiami šių trijų segmentavimo algoritmų pavyzdžiai. *Felzebszwalb* algoritmu gauti 194 segmentai, SLIC – 190 segmentai (naudotos 250 k-means grupės), *quickshift* – 695 segmentai. Segmentacijos gautos naudojant *Python* programavimo kalbos *scikit-image* biblioteką. Tos pačios realizacijos naudojamos LIME bibliotekoje.

Nors LIME bibliotekos kode realizuoti visi trys algoritmai, naudojamas yra tik vienas – *quickshift.* Jeigu naudodamiesi LIME paaiškinimų funkciją patys neperduodame segmentacijos funkcijos, kaip numatyta segmentacijos funckija parenkama būtent ši.

6 pav. Felzenszwalb, SLIC ir Quickshift segmentacijos.

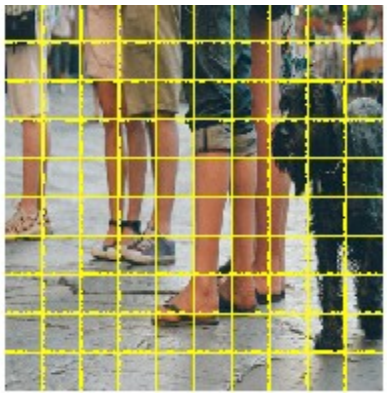
3.2. Kiti segmentavimo algoritmai

3.2.1. Vandenskyros algoritmas

Vandenskyros [RM00] (*watershed)* algoritmas taikomas nespalvotiems paveikslėliams. Algoritmas traktuoja paveikslėlį kaip topologinį žemėlapį, kuriame pikselių reikšmės nurodo

aukštį žemėlapio taške. Vandenskyros algoritmas šiame žemėlapyje segmentaciją atlieka ieškodamas keterų tarp aukščiausių topologinio žemėlapio taškų. Algoritmo euristika – žemėlapyje parenkami vandens šaltinių taškai. Tuomet žemėlapis iš šių taškų pildosi vandeniu. Vietos kur kildamas susikerta vanduo iš skirtingų šaltinių nubrėžia segmentacijų linijas.

Šis algoritmas turi keletą skirtingų realizacijų. Šio darbo tyrime panaudotos dvi iš jų. Pirmasis – kompaktinis vandenskyros algoritmas [NP14], kuriame vandens šaltiniai žemėlapyje išdėliojami tinkleliu. Vandens šaltinių kiekį nenustatomas automašikai, jis yra perduodamas kaip papildomas parametras .Algoritmas turi „kompaktiškumo“, kuris neleidžia vandeniui nutolti nuo šaltinio. Jeigu parinksime labai aukštą šio parametro reikšmę segmentacijos bus tiesiog kvadratai (7 pav.). Antrojoje algoritmo realizacijoje nenaudojamas kompaktiškumo parametras. Vandens šaltinių kiekis ir vieta žemėlapyje parenkami automatiškai. Vandens šaltinis inicijuojamas vietose, kuriuose yra mažas gradientas tarp aplinkinių pikselių, t. y. aplinkinis regionas yra pakankamai „lygus“ ir aplinkinių pikselių reikšmės yra gana panašios.



7 pav. Ekstremalus kompaktinės vandenskyros segmentacijos pavyzdys.

3.2.2. Morfologinės gyvatės

Morfologinės gyvatės [MBA14] (*morphological snakes*) priklauso geodezinių aktyvių kontūrų [CKS97] (*Geodesic Active Contours)* segmentavimo algoritmų grupei. Tačiau segmentacijos gaunamos dvejetainiame masyve naudojant morfologines operacijas. Geodezinių aktyvių kontūrų algoritmas naudoja realių skaičių masyvą ir sprendžią dalines diferencialines lygtis. Dėl šių savybių morfologinių gyvačių algoritmas yra greitesnis [MBA14]. Morfologinių gyvačių segmentavimo algoritmui reikalingi ryškūs objektų kontūrai. Todėl prieš pateikiant paveikslėlį algoritmui jis dar turi būti papildomai apdorotas stengiantis išryškinti kontūrus.

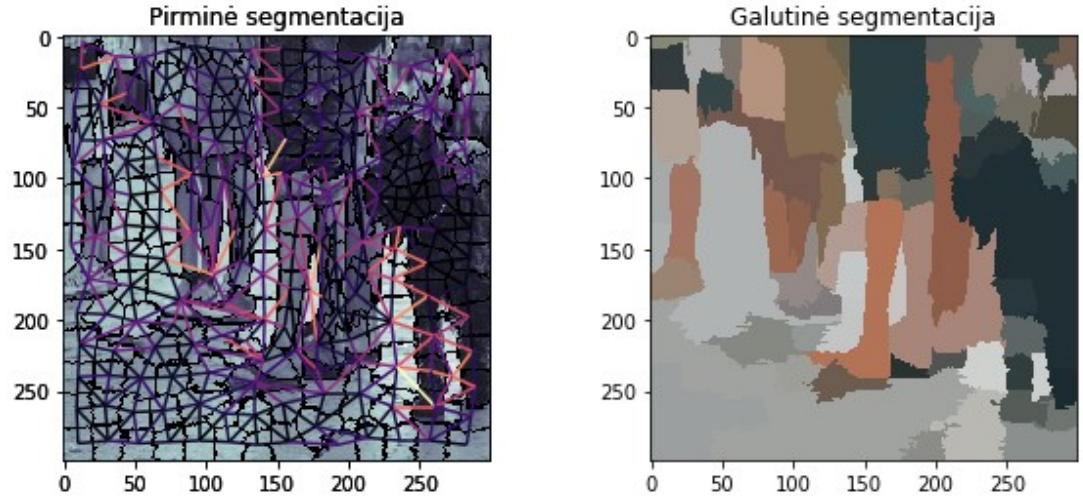
Šis segmentacijų algoritmas nebuvo panaudotas tyrime, nes atliekant pirminius bandymus segmentuojant naudojamo duomenų rinkinio paveikslėlius, daugeliui jų gautos prastos segmentacijos (8 pav.). Tai turbūt įvyko todėl, kad apdorojant paveikslėlį nebuvo pakankamai

gerai išryškinti kontūrai. Galima rasti kitą apdorojimo algoritmą kurį pritaikius gautumėme gerą segmentaciją. Tačiau mums svarbu, kad segmentacijos algoritmas būtų generalizuojamas plačiam paveikslėlių kiekiui. Tai yra todėl, kad duomenų rinkinius, kuriais apmokomi neuroniai tinklai, dažnai sudaro įvairios kokybės ir įvairaus turinio paveikslėliui. Dėl to svarbu, kad kokybiškas segmentaciajas gautume jiems visiems, o ne tik jų daliai. Būtent dėl šių priežasčių šis algoritmas nebuvo panaudotas atliekant tyrimą.

8 pav. Morfologinių gyvečių segmentavimo pavyzdys. Gauti segmentai stipriai varijuoja dydžiu ir vienas segmentas apima apie pusę paveikslėlio.

3.2.3. Hierarchinis grafo regionų jungimas

Taikant šį segmentavimo algoritmą pirmiausia paveikslėlis apdorojamas jį suskirstant į grafą, kurio kiekviena briauna turi svorį. (9 pav.) Briaunų svoris paskaičiuojamas pagal jos skiriamų regionų vidutinę pikselio vertę. Toliau algoritmas jungia tarpusavyje regionus kurių briaunų svoriai yra mažiausi. Jungimas vyksta iteraciniu būdu, iki tol kol nebelieka regionų kurios galima sujungti. [HEM+98] Svorį nuo kurio jau briaunos nebus jungiamos nustatome segmentavimo funkcijai perduodami slenksčio parametrą. Šis algoritmas veikia lėčiau už *quickshift, compact watershed, SLIC* ar *Felzenszwalb.* Tačiau skaičiuojant modelių sprendimo paaiškinus LIME metodu paveikslėlių segmentacija skaičiuojama tik kartą. Žymiai didesnę laiko dalį užima įvairių segmentacijų kombinacijų atpažinimas giliu neuroniniu tinklu.



9 pav. Kairėję – pirminiai paveikslėlio segmentai. Dešinėje – galutinė segmentacija po grafų briaunų jungimo.

3.3. Tyrimo techninės charakteristikos

Skaičiavimai atlikti Ubuntu 14.04 operacinėje sistemoje. Naudotas procesorius: Intel® Core™ i5-5300U CPU, 2.30GHz × 4. Neuroninio tinklo realizacijai naudotas tensorflow karkasas.

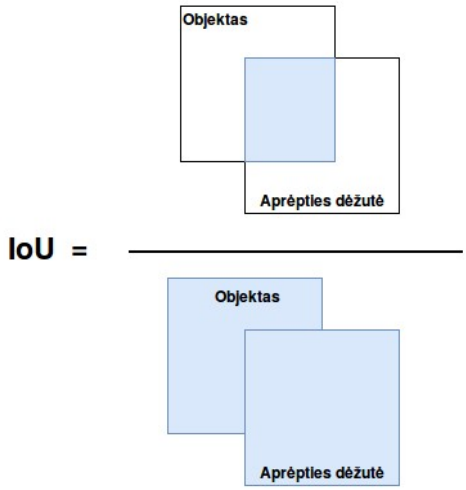
Analizuojamas neuroninis tinklas – *Inception v3* [SVI+16]. Šis tinklas apmokytas 2012 metų *ImageNet* [RDS+15] duomenų rinkinio duomenimis. *ImageNet* duomenų rinkinį sudaro 1000 klasių, klasifikuojamas visas paveikslėlis. *Inception v3* modelis su 2012 *ImageNet* validacijos duomenis tarp 5 aukščiausių spėjimų teisingos klasės nebuvo 3,46 % . Palyginimui: AlexNet [KSH12] - 15,3 % , *GoogLeNet* [SLJ+15] - 6,67 %.

Analizuojamas duomenų rinkinys – *Microsoft COCO* (*Common Objects in Context)* duomenų rinkinys. Naudoti 2017 metų validavimo duomenys, juos sudaro 5000 paveikslėliai. Šis duomenų rinkinys pasirinktas, nes duomenų rinkinyje yra informacija apie objektus paveikslėlyje ir būtent kurioje paveikslėlio vietoje šie objektai yra. Lyginant šią informaciją su LIME rezultatais bus bandomo skaitiškai įvertinti LIME veikimą. Naudojamas *Inception v3* modelis nebuvo apmokytas su COCO duomenimis, tačiau daugelis klasių sutampa todėl galime *Inception v3* taikyti ir COCO duomenims.

3.4. Tiriamos segmentacijos

Jeigu turime giliųjų neuroninių tinklų modelį, kuris yra apmokytas kokybiškai ir šis modelis, kurią nors klasę paveikslėlyje atpažino su aukštu pasikliovimu. Galime tikėtis, kad modelis aukštą pasikliovimą gavo kreipdamas dėmesį į atpažintą objektą. Kiti paveikslėlyje

esantys objektai ar fonas sprendimui įtakos daug nepadarė. Tokiu atveju galime daryti prielaidą, kad tokiam paveikslėliui pritaikytas LIME metodas išskirs segmentus, kuriuose yra atpažįstamas objektas, kaip darančius teigiamą įtaką sprendimui. O segmentai priklausantys fonui nebus išskirti arba bus išskirti kaip darantys neigiamą įtaką modelio sprendimui. Galime tikėtis, sutaps regionai kuriuose yra objektas ir kuriuos identifikavo LIME metodas. Idealiu atveju šie paveikslėlio regionai sutaptų. Kaip tiksliai šie du regionai sutampa galima išmatuoti jų sankirtos plotį padalinus iš sąjungos ploto. Šis matavimas dažnai taikomas objektų aptikimo uždaviniams. Literatūroje dažnai žymimas kaip „IoU“ (*Intersection over union) (10 pav.).*



10 pav. IoU skaičiavimo vizualizacija

Pasitelkiant IoU kaip kokybės matą, ištirtas LIME veikimas su 6 skirtingais segmentavimo algoritmais:

1. *Felzenszwalb* – parametrai parinkti taip, kad segmentų kiekis būtų panašus į gaunamus *QuickShift* algoritmu. Parametrai: (mastelis=300, sigma=0.5, minimalus dydis=150)
2. *SLIC* – naudojama 120 k-means centrų. Parametrai: (segmentai=120, kompaktiškumas= 10, sigma=1)
3. *Kompaktinė vandenskyra* – visada gauname 100 segmentų. Parametrai: (segmentai=100, kompaktiškumas=0.005)
4. *Vandenskyra* – Segmentų kiekis varijuoja ir priklauso nuo spalvų gradiento nespalvotama paveikslėlyje paveikslėlyje. Parametrų nėra
5. *Hierarchinis grafo regionų jungimas* – Segmentų kiekis taip pat varijuoja. Parametrai: (slenkstis=0.1)

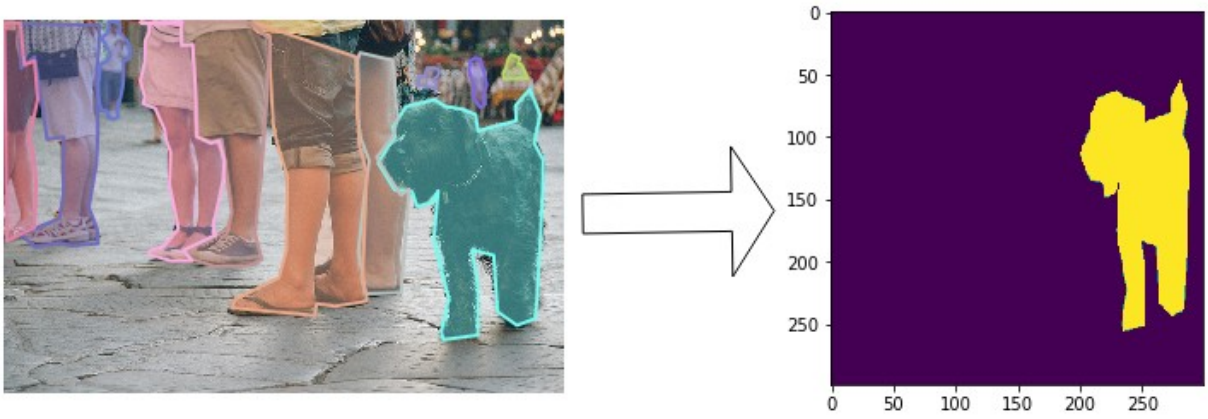
Atliekant modelio sprendimo paaiškinimo skaičiavimus LIME metodu naudojami du parametrai – nurodomas segmentavimo algoritmas ir pavyzdžių kiekis. Kokią įtaką sprendimui daro individualus segmentas LIME nustato neuroniniui tinklui siųsdamas paveikslėlį, kur rodoma tik dalis segmentų. Pavyzdžių kiekis nusako, kiek kartų bus šių skirtingų segmentų kombinacijų bus parodyta neuroniniui tinklui. Kuo didesnė šio parametro reikšmė, tuo tiksliau LIME geba nustatyti individualių segmentų daromą įtaką.

6. *QuickShift* – naudoti tokie patys parametrai kurie standartiškai nustatyti LIME karkase. Parametrai: (branduolio dydis=4, maksimalus atstumas =200, santykis=0.2)

Tyrimo metu panaudotos dvi pavyzdžių kiekio reikšmes – 500 ir 100. Ši kombinacija pasirinkta tam, kad patikrinti kiek suprastės rezultatai sumažinus pateikiamų pavyzdžių kiekį ir ar šis sumažėjimas priklausys nuo taikomo segmentavimo algoritmo.

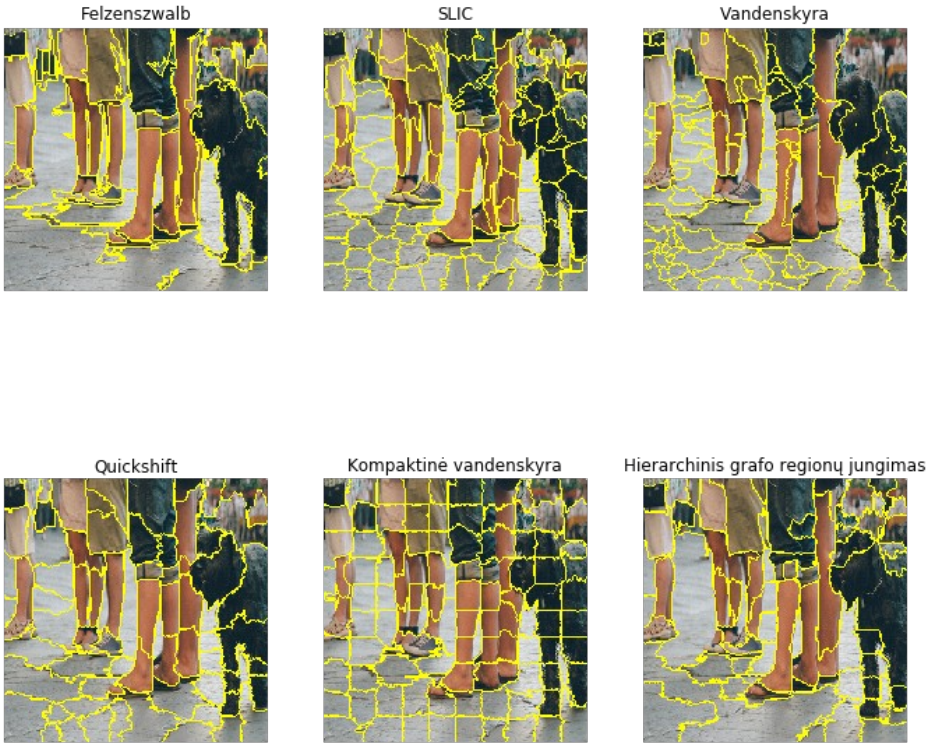
3.5. Tyrimo eiga

Inception v3 modelis atpažįsta 300x300 dimensijų paveikslėlius. Kitokių dimensijų paveikslėliai transformuojami į šias dimensijas. COCO duomenų rinkinio objektą identifikuojantys regionai taip pat turi būti transformuoti į šias dimensijas. Prieš atliekant transformacijas pašalinami neaktualūs regionai (11 pav.). *Inception v3* šuns klasę atpažįsta su 91% pasikliovimo procentu.

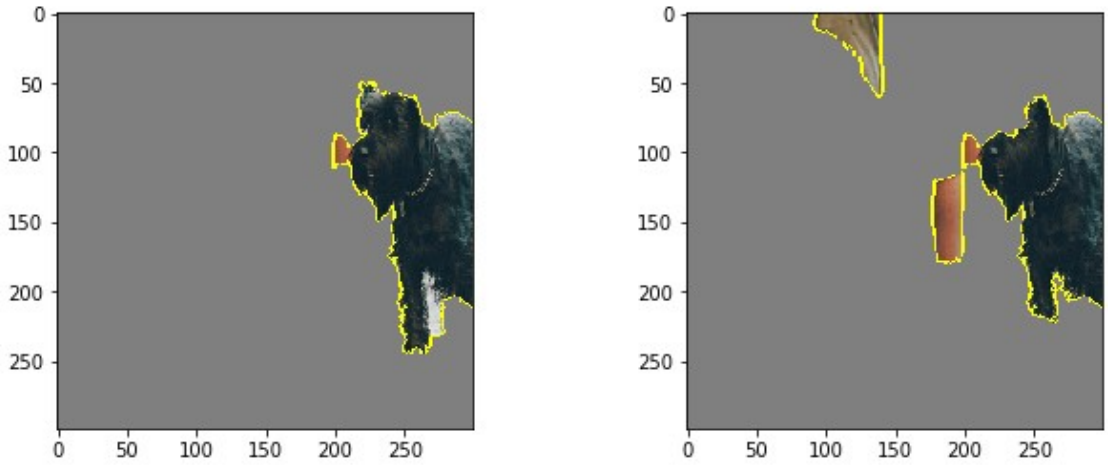


11 pav. Objekto regiono išskyrimas ir matmenų transformacija

Transformuotam paveikslėliui pritaikome segmentavimo algoritmus (12 pav.) Kiekvienam jų suskaičiuojame gautų segmentų skaičių. Galiausiai, atliekame paaiškinimų skaičiavimus LIME karkase. Kiekvienai iš 6 segmentacijų paaiškinimai paskaičiuoti naudojant 500 ir 100 pavyzdžių (13 pav.).



12 pav. Segmentavimo algoritmai pritaikyti vienam iš tyrimo metu naudotų paveikslėlių



13 pav. LIME paaiškinimų palyginimas. Naudotas quickshift segmentavimo algoritmas. Kairėje pusėje rezultatas po 500 iteracijų, dešinėje rezultatas po 100 iteracijų

3.6. Rezultatai

Tyrimui pasirinkti 10 skirtingų klasių paveikslėlių kuriuose *Inception v3 k*lases atpažino vidutiniškai su 0.89 pasikliautimi. Vidutinis segmentų skaičius skyrėsi priklausomai nuo to, koks buvo naudojamas algoritmas. Vidutiniškai mažiausiai segmentų sudarydavo *quickshift* ir *Felzenszwalb* algoritmai. Taikant q*uickshift* algoritmą nebuvo didelių variacijų tarp segmentų kiekio skirtinguose paveikslėliuose – tą galime pamatyti iš standartinio nuokrypio. Vidutiniškai daugiausia segmentų sudarė vandenskyros algoritmas. Jis taip pat pasižymėjo didžiausia variaciją tarp tiriamų algoritmų. Vandenskyros algoritmu gauti tiek mažiausias (14) tiek didžiausias segmentų kiekis (410).

1 lentelė. Segmentų kiekis

Segmentų kiekis						
Klasė	**Quickshift**	**Felzenszwalb**	**SLIC**	**Kompaktinė vandenskyra**	**Vandenskyra**	**Grafo regionų jungimas**
Dramblys	69	66	98	100	14	122
Zebras	61	93	73	100	53	95
Kelio ženklas	61	65	92	100	135	66
Mob. telefonas	62	64	94	100	136	63
Beisbolo žaidėjas	91	21	108	100	410	20
Lokomotyvas	58	106	89	100	58	86
Bananas	59	47	94	100	111	54
Lėktuvas	68	55	97	100	91	66
Autobusas	66	68	101	100	70	92
Šuo	70	68	97	100	122	66
Vidurkis	**66,5**	**65,3**	**94,3**	**100**	**120**	**73**
Standartinis nuokrypis	9,11	22,01	8,65	0,00	103,72	26,14

Naudojantis vandenskyros segmentacijos algoritmu gautos geriausios IoU vidurkio ir standartinio nuokrypio reikšmės. Nors ir segmentų kiekis stipriai varijavo taikant šį algoritmą, tačiau segmentai buvo pakankamai kokybiški, kad esant dideliam jų kiekiui LIME 500 iteracijų rasti teigiamą įtaką darančius segmentus. Blogiausiai pasirodė kompaktinės vandenskyros algoritmas. Tai galėjo nutikti todėl, kad šis algoritmas nėra pakankamai lankstus – sudaro fiksuotą segmentacijų skaičių. Norint šiuo algoritmu pasiekti geresnių rezultatų reikėtų sumažint į kompaktiškumo parametrą, kad segmentacijos būtų mažiau lokalizuotos. *Felzenszwalb* algoritmas pagal IoU vidurkį yra antroje vietoje, tačiau jo rezultatai labiausiai varijuoja. Skaičiuodamas kiekvieną paaiškinimą ir taikant 500 iteracijų, LIME užtruko vidutiniškai 187 sekundes.

2. lentelė. IoU reikšmės kai LIME pritaikytas naudojant 500 pavyzdžių

Klasė	**Quickshift**	**Felzenszwalb**	**SLIC**	**Kompaktinė vandenskyra**	**Vandenskyra**	**Grafo regionų jungimas**
Dramblys	0,25	0,54	0,11	0,08	0,43	0,13
Zebras	0,22	0,17	0,59	0,16	0,35	0,27
Kelio ženklas	0,24	0,47	0,06	0,05	0,46	0,46
Mob. telefonas	0,21	0,42	0,20	0,10	0,39	0,12
Beisbolo žaidėjas	0,29	0,06	0,22	0,19	0,47	0,42
Lokomotyvas	0,35	0,16	0,19	0,17	0,27	0,16
Bananas	0,46	0,78	0,34	0,20	0,51	0,50
Lėktuvas	0,27	0,46	0,20	0,25	0,38	0,27
Autobusas	0,25	0,22	0,14	0,13	0,23	0,13
Šuo	0,59	0,37	0,36	0,43	0,39	0,54
Vidurkis	0,313	0,365	0,241	0,176	0,388	0,3
Standartinis nuokrypis	0,12	0,20	0,15	0,10	0,08	0,16

Kaip ir buvo galima tikėtis sumažinus LIME iteracijų kiekį vidutiniškas IoU sumažėjo (3 lentelė). Didžiausias sumažėjimas pastebėtas taikant vandenskyros algoritmą, mažiausias – *quickshift.* Galima atkreipti dėmesį į vandenskyros segmentacijos algoritmą taikytą „Beisbolo žaidėjo“ klasei. Taikant 100 LIME iteracijų gauta nulinė IoU reikšmė. Šiam paveikslėliui Vandenskyros objektas sugeneravo 410 segmentų. To pasekoje 100 iteracijų nepakako identifikuoti teigiamą įtaką sprendimui darančius segmentus. Todėl renkantis LIME iteracijų skaičių reikia atsižvelgti į tai kiek segmentų sudaro naudojamas algoritmas. Taikant 100 iteracijų LIME algoritmas paaiškinimams rasti užtruko vidutiniškai 39 sekundes.

3. lentelė . IoU vidurkių skirtumas tarp LIME pritaikymo su 500 ir 100 iteracijų

	Quickshift	**Felzenszwalb**	**SLIC**	**kompaktinė Vandenskyra**	**Vandenskyra**	**Grafo regionų jungimas**
vidurkių skirtumas	0,038	0,052	0,069	0,048	0,114	0,084

4. lentelė. IoU reikšmės kai LIME pritaikytas naudojant 100 pavyzdžių

100 pavyzdžių						
Klasė	**Quickshift**	**Felzenszwalb**	**SLIC**	**Kompaktinė vandenskyra**	**Vandenskyra**	**Grafo regionų jungimas**
Dramblys	0,21	0,5	0,07	0,08	0,4	0,1
Zebras	0,15	0,03	0,44	0,12	0,32	0,16
Kelio ženklas	0,22	0,45	0,06	0,04	0,37	0,34
Mob. telefonas	0,22	0,28	0,12	0,07	0,2	0,1
Beisbolo žaidėjas	0,36	0,06	0,16	0,27	0	0,04
Lokomotyvas	0,28	0,03	0,17	0,13	0,27	0,24
Bananas	0,34	0,68	0,29	0,15	0,25	0,16
Lėktuvas	0,34	0,43	0,21	0,19	0,37	0,41
Autobusas	0,19	0,27	0,11	0,1	0,25	0,09
Šuo	0,44	0,4	0,09	0,13	0,31	0,52
Vidurkis	0,275	0,313	0,172	0,128	0,274	0,216
Standartinis nuokrypis	0,09	0,21	0,11	0,06	0,11	0,15

3.7. Segmentacijų tyrimo išvados

Apibendrinant segmentacijų tyrimo rezultatus galima išskirti Vandenskyros algoritmą, kuris pademonstravo aukščiausią IoU vidurkį ir mažiausią standartinį nuokrypį taikant 500 LIME iteracijų, tačiau mažinant iteracijų skaičių jo efektyvumas krenta greičiausiai. LIME karkase numatytas *quickshift* algoritmas veikė stabiliausiai, IoU reikšmių standartiniai nuokrypiai buvo vieni mažiausių. Taip pat jį silpniau įtakoja iteracijų kiekio mažinimas. Galime spėti, kad būtent dėl šių savybių iš LIME realizuotų segmentavimo algoritmų *quickshift* pasirinktas kaip numatytasis.

Quickshift, f*elzenszwal* ir vandenskyros algoritmai veikė efektyviausiai. Pasitelkiant vandenskyros algoritmą magistro baigiamajame darbe bus ištirtas LIME paaiškinimų atrinkimo metodas. Šio metodo veikimui svarbu, kad analizuojami segmentai būtų kuo prasmingesni. Šis tyrimas padėjo nustatyti segmentavimo algoritmus, su kuriais paaiškinimų atrinkimų metodas pasieks geresnius rezultatas.

4. LIME Submodulinio pasirinkimo algoritmo tyrimas

Paaiškinimai gauti LIME metodu gali padėti tyrėjui gauti įžvalgų apie modelio veikimą su konkrečiais paveikslėliais. Norint susidaryti intuiciją apie bendrą modelio veikimą, remiantis tik individualiais paaiškinimais reikia, kad šie paaiškinimai tarpusavyje neštų kuo galima daugiau skirtingos informacijos. Pavyzdžiui, jeigu tiriamas modelis apmokytas klasifikuoti tam tikrą kiekį skirtingų klasių, mažai sužinosime apie modelio veikimą, jei didžioji dalis paaiškinimų bus paveikslėlių, priklausančių vienai klasei. Taip pat, net ir analizuojant paveikslėlius, priklausančius vienai klasei, svarbu, kad paaiškinimus sudarytų skirtingi atributai. Pavyzdžiui, analizuojant paveikslėlius su automobiliais ir pamačius vieno paveikslėlio paaiškinimą, būtų naudinga, jei sekančiame paveikslėlyje esantis automobilio objektas būtų kitokios spalvos, būtų pasisukęs kitokiu kampu ar skirtųsi kokia kita savybė. Ši informacija padėtų suprasti kokius atributus modelis yra išmokęs atpažinti, o kokių – ne.

Straipsnyje, kuriame pristatomas LIME, taip pat aprašytas algoritmas, skirtas iš visų paaiškinimų, gautų LIME metodu, išrinkti aktualiausius (Algoritmas 1). Tačiau algoritmo realizacija, pateikiama straipsnyje, pritaikyta darbui su tekstiniais arba lentelėse esančiais duomenimis.

Jeigu duomenys yra lentelėse, egzistuoja tam tik tikras fiksuotas kiekis atributų. Pavyzdžiui, IRIS [Fis36] duomenų rinkinyje kiekvienas pavyzdys turi po 4 atributus. Analizuojant tokius pavyzdžius LIME metodu, gaunami paaiškinimai tarpusavyje turės daug bendrų atributų, skirsis tik jiems priskiriamas pasikliovimo dydis.

Dirbant su tekstiniais duomenimis, atributais tampa žodžiai. Šiuo atveju LIME paaiškinimai mažiau persidengs bendrais atributais. Tačiau, jeigu naudojamų žodžių aibė nebus didelė, persidengimas bus didesnis. Taip pat persidengimą galima padidinti, pasitelkiant teksto apdorojimo metodikas. Žodžio kamieno gavimo (angl. S*temming)* algoritmais galime skirtingus žodžius (atributus) suvienodinti, taip kelis atributus sujungdami į vieną.

Analizuojant paveikslėlius, susiduriama su nepakankamo atributų persidengimo tarp LIME paaiškinimų problema. LIME mašininio mokymosi modelių, skirtų vaizdų atpažinimui, paaiškinimai yra paveikslėlių segmentai. Tikimybė, kad pora segmentų bus vienodi, yra artima nuliui. Dėl to, kad visų paaiškinimų atributai yra unikalūs, tampa sunku nustatyti, būtent kurie paaiškinimai yra aktualiausi.

Šiame skyriuje bus pristatytas LIME submodulinio pasirinkimo algoritmas ir pasiūlytos jo

modifikacijos pritaikymui darbui su paveikslėlių duomenimis. Algoritmo modifikacijos bus pritaikytos analizuojant tą patį duomenų rinkinį kaip ir trečiajame skyriuje – COCO 2017 validavimo duomenys. Naudojamas mašininio mokymosi modelis taip pat nesikeičia – *Inception v3* [SVI+16]. Detali informacija apie šį duomenų rinkinį ir modelį aprašoma 3.3. skyriuje.

4.1. Submodulinio pasirinkimo algoritmas

Submodulinio pasirinkimo algoritmui (Algoritmas 1) reikalingi du parametrai. X – rinkinys duomenų, kuriems bus sugeneruoti paaiškinimai, B – dydis, nusakantis kokį kiekį paaiškinimų atrinks algoritmas. Algoritmą galima suskirstyti į 3 pagrindines dalis.

2-4 eilutėse sukonstruojama $n \times d'$ dimensijų matrica - W, kur $n = |X|$ ir d' = skirtingų atributų skaičius. Jeigu atributo *j* nėra pavyzdžio x_i paaiškinime, tuomet $W_{ij} = 0$.

5-7 eilutėse konstruojamas „svarbumo“ vektorius *I*. *I* apskaičiuojamas susumuojant matricos *W* stulpelių elementų absoliučias vertes ir ištraukiant iš sumos šaknį. Gautas vektorius atspindi, kurie atributai paaiškinimuose pasikartoja dažniau ir įgyja didesnės vertes, t. y. svarbesni.

8-11 eilutėse konstruojamas aktualiausių paaiškinimų indeksų masyvas $|V| = B$. Indeksų pasirinkimas atliekamas maksimizuojant funkciją (1). Funkcijai perduodami trys parametrai: indeksų masyvas – V , paaiškinimų matrica – W ir *svarbumo* vektorius – *I*. funkcija sudaryta remiantis intuicija, kad parinkti paaiškinimai neneštų perteklinės informacijos. Tai įgyvendinama panaudojant indikatorinę funkciją. W_{ij} skaitinės reikšmės dydis, kai jis yra didesnis už 0, rezultatui įtakos neturi. Dėl to jeigu V papildysime indeksu paaiškinimo, kurio atributai sutampa su atributais jau esančiais kituose V indeksuose, funkcijos reikšmė nepasikeis. Funkcijos reikšmė augs tik kai V bus papildomais indeksais paaiškinimų kurie dar nebuvo matyti. 10 eilutėje pateikiamas godus pasirinkimo funkcijos (2) maksimizavimo algoritmas. Maksimizuoti aprėpties funkciją su svoriais yra NP sunki problema [Fei98]. Dėl funkcijos (1) submodulinių savybių savybių $c(V \cup \{i\}, W, I) - c(V, W, I) \geq 0$ esame garantuoti, kad skirtingų atributų kiekis, kurį padengs V_{godus} gautas godžiu algoritmu, nuo optimalaus $V_{optimalus}$ skirsis ne daugiau nei $1 - \frac{1}{e}$.

$$c(V, W, I) = \sum_{j=1}^{d'} 1_{[\exists i \in V : W_{ij} > 0]} I_j \quad (1)$$

$$Pick(W, I) = \underset{V, |V| \leq B}{argmax}\, c(V, W, I) \quad (2)$$

Algoritmas 1: Submodulinis pasirinkimas

$X \leftarrow$ Analizuojami duomenys, $B \leftarrow$ Biudžetas

```
1: procedure SUBMODULINIS PASIRINKIMAS
2:     for x_i ∈ X do
3:         W_i ← explain(x_i, x'_i)
4:     end for
5:     for j ∈ {1...d'} do
6:         I_j ← sqrt(Σ_{i=1}^{n} |W_ij|)
7:     end for
8:     W ← {}
9:     while |V| < B do
10:        V ← V ∪ argmax_i c(V ∪ {i}, W, I)
11:    end while
12:    return V
13: end procedure
```

Optimizacijos algoritmas gerai veikia, kai turimas *Biudžetas* santykinai mažas, palyginus su unikalių atributų skaičiumi. Tačiau, jeigu vykdant 9-11 eilutėse esantį ciklą, išrenkami paaiškinimai, į kurių aibę patenka visi unikalūs atributai, tuomet jau pasiekta maksimali funkcijos (1) reikšmė ir tolesniuose cikluose galioja lygybė $c(V \cup \{i\}, W, I) - c(V, W, I) = 0$. Taigi, sekantys elementai bus pasirenkami atsitiktine tvarka. Atsitiktinio pasirinkimo pageidautina išvengti, nes gali būti parinkti tokie paaiškinimai, kurie neturi nei vieno atributo. Šią problemą galima spręsti keliais būdais:

- Sekti funkcijos (1) reikšmę tarp ciklo iteracijų ir ciklą nutraukti jeigu (1) reikšmės nustoja didėti.
- Funkcijos (1) reikšmėms nustojus didėti, ją pakeisti modifikuota (1) funkcijos versija, kuri atsižvelgtų į W_{ij} reikšmės, pavyzdžiui (3). Tokiu būdu išvengiama, kad į V bus pridėti indeksai paaiškinimų kurie neturi atributų.

$$c(V, W, I) = \sum_{j=1}^{d'} \left(1_{[\exists i \in V : W_{ij} > 0]} I_j + \sum_{i \in V} W_{ij} I_j\right) \quad (3)$$

4.2. Submodulinio pasirinkimo algoritmo paveikslėliams modifikavimas

Šiame skyriuje bus pristatytos galimos submodulinio pasirinkimo algoritmo modifikacijos, atsižvelgiant į problemines sritis, kylančias algoritmą siekiant pritaikyti darbui su paveikslėliais. LIME straipsnyje pateikiama idėja, kad dirbant su paveikslėliais *svarbumo* vektorius turėtų būti apskaičiuojamas, atsižvelgiant į spalvų histogramas ar kokias kitas segmentų savybes. Tačiau tai neadresuoja to, kad nėra nustatoma, kurie paaiškinimai turi bendrus atributus.

Dėl to, išanalizavus submodulinio pasirinkimo algoritmą, nuspręstą jį papildyti, įvedant klasterių sudarymo žingsnį. Kiekvienas klasteris atitiktų skirtingą atributą. Klasteriai galėtų būti sudaromi pagal spalvų histogramas. Tokiu atveju galime nekeisti *svarbumo* vektoriaus sudarymo formulės, nes segmentų spalvos informacija atsispindės atributuose (panašaus atspalvio segmentai pateks į tą patį klasterį). *Svarbumo* vektoriaus funkcionalumas nepakinta, vektorius didesnes reikšmes priskiria tiems atributams, kurie paaiškinimuose pasitaiko dažniau ir įgyja didesnes svorio reikšmes.

Taip pat bus atliktas tyrimas panaudojant (3) formulę. Tokios formulės taikymas gali pasitarnauti tokiais atvejais, kai naudojamas didelis *Biudžeto* dydis. Tokiu atveju, kai $B=|X|$ submodulinio pasirinkimo algoritmu gauto sąrašo, pirmieji elementai padėtų nustatyti pagrindinius atributus, kuriuos moka atpažinti analizuojamas mašininio mokymosi modelis. O paskutiniai sąrašo elementai nurodytų, kuriems paveikslėliams paaiškinimų nebuvo rasta arba jų buvo rastas mažas kiekis. Ši informacija taip pat yra naudinga, nes svarbu suprasti ne tik kokie paveikslėlio segmentai yra svarbūs modeliui, bet ir kokių modelis nemoka atpažinti ar laiko nepakankamai svarbiais.

4.2.1. Klasterizavimo algoritmai

Siekiant paveikslėlio segmentus suskirstyti į prasmingus klusterius, svarbu parinkti gerą klasterizavimo algoritmą ir nustatyti su kuriais segmentų atributais geriausia atlikti klasterizaciją. Pasirinkti keli galimi klasterizavimo algoritmai, iš kurių eksperimentiškai bus pasirinktas tas, kuris labiausiai tinka naudojamiems duomenims. Renkantis kandidatus, pagrindinis keliamas kriterijus buvo dinamiškas klasterių kiekio parinkimas, nes iš anksto nėra žinoma koks kiekis klasterių geriausiai atskirs duomenis. Kaip kandidatai pasirinkti *Affinity propagation, DBSCAN, MeanShift* ir *Birch* algoritmai

Affinity propagation [FD07] algoritmas grįstas „žinučių perdavimu“ tarp duomenų. Klasterių centrų kiekis parenkamas automatiškai ir šio algoritmo vykdymo metu centrai jungiasi, kol pasiekiamas konvergavimas ir daugiau naujų sujungimų neatliekama. Naudojamoje algoritmo realizacijoje konvergavimas laikomas pasiektu, kai klasterių kiekis nepakinta 15 iteracijų.

DBSCAN [EKS+96] algoritmas klasterius formuoja kaip didelio tankumo zonas, atskirtas mažo tankumo zonomis. Dėl šios savybės klasteriai gali būti įvairios formos, priešingai nuo *K-means,* kur klasteriai yra išgaubtos formos. 2014 metais algoritmas buvo apdovanotas *test of time* apdovanojimu KDD konferencijoje, taip pažymint dažną algoritmo naudojimą mokslo

bendruomenėje [STOT14].

MeanShift [Che95] yra kopimo į kalną principu grįstas algoritmas, kiekvieno klasterio centro kandidatas yra atnaujinamas didžiausio tankumo zonos kryptimi. Algoritmas taikomas klasterių analizės ir vaizdų apdorojimo srityje [CM02]. Dėl dažno naudojimo vaizdų apdorojimo srityje šis algoritmas ir pasirinktas kaip vienas iš kandidatų.

Birch [ZRL96] algoritmas atlieka hierarchinį klasterizavimą ir gerai veikia esant net ir dideliems duomenų rinkiniams. Algoritmui galima nurodyti fiksuotą klasterių kiekį arba jis gali būti parenkamas automatiškai.

4.2.2. Klasterizavimo algoritmų vertinimo metrikos

Klasterizuojami duomenys sudaryti iš daugelio dimensijų, todėl sudarytus klasterius sunku įvertinti duomenų vizualizavimo technikomis. Todėl, siekiant iš praeitame skyrelyje išvardintų algoritmų išrinkti labiausiai tinkantį analizuojamiems duomenims, jie įvertinti keliomis skaitinėmis metrikomis. Tam, kad būtų sudarytas bendresnis vaizdas nesiklaujama tik viena metrika, nes siekiant maksimizuoti vieną klasterių sudarymo aspektą gali nukentėti kiti. Tyrimo metu pasirinktos trys metrikos.

Silhouette koeficientas [Rou87] vienam duomenų taškui apskaičuojamas pagal formulę (4), kur a – vidutinis atstumas nuo taško iki visų kitų taškų priklausančių tam pačiam klasteriui, b – vidutinis atstumas nuo taško iki visų kitų artimiausio klasterio taškų. *Silhouette koeficientas* duomenų rinkiniui skaičiuojamas kaip visų atskirų taškų koeficientų vidurkis. Koeficientas įgyja reikšmes rėžiuose [-1;1]. Neigiamos koeficiento reikšmės indikuoja, kad taškai priskirti klaidingiems klasteriams. Reikšmės arti 0 indikuoja, kad klasteriai persidengia.

$$s=\frac{b-a}{max(a,b)} \quad (4)$$

Calinski and Harabaz [CH74] indeksas, dar žinomas kaip dispersijos santykio kriterijus, skaičiuojamas kaip santykis tarp klasterių dispersijos vidurkio ir klasterio dispersijos. Didesnės indekso reikšmės reiškia, kad klasteriai tankūs ir gerai atskirti.

Davies-Bouldin [DB79] indeksas apibrėžiamas kaip vidutinis panašumas tarp klasterio $C_i, i=1,..,k$ ir į jį panašiausio klasterio C_j . Panašumas apskaičiuojamas kaip R_{ij} (5). Kur s_i yra vidutinis atstumas tarp kiekvieno klasterio C_i taško ir klasterio centro, d_{ij} yra atstumas tarp klasterių C_i ir C_j centrų. Tuomet indeksas apskaičuojamas pagal formulę (6). Nulis yra mažiausia galima indekso reikšmė. Reikšmės artimos nuliui indikuoja gerą klasterių

atskyrimą.

$$R_{ij}=\frac{s_i+s_j}{d_{ij}} \quad (5)$$

$$DB=\frac{1}{k}\sum_{i=1}^{k} \max_{i\neq j} R_{ij} \quad (6)$$

4.2.3. Klasterizavimo tyrimas

Klasterius sudaryti pasirinkta iš kelių galimų paveikslėlių segmentų savybių. Kiekvienam segmentui galime apskaičiuoti spalvų histogramas. Spalvotiems paveikslėliams skaičiuojamos trys histogramos kiekvienam spalvos kanalui (raudonai, žaliai, mėlynai). Histogramas gali sudaryti nuo 1 iki 256 stulpelių. Atliekant tyrimą naudotos skirtingos stulpelių kiekio reikšmės, nes naudojant 256 stulpelių histogramas, klasterizacija atliekama 3x256 dimensijų erdvėje. Ir nors informacijos išsaugoma daugiausiai, tačiau klasterizacijai per didelis taškų išsibarstymas gali padidinti anomalių taškų skaičių.

Klasterizacijos atliktos su trijų tipų histogramomis:

1. Įprastomis spalvos histogramomis, kur x ašyje žymime pikselių reikšmes ir y ašyje šias reikšmes įgyjančių pikselių kiekį. Šiose histogramose atsispindi ne tik informacija apie segmento pikselių spalvų reikšmes, bet ir apie segmento dydį.
2. Pasiskirstymo funkcijos histogramoje stulpelio reikšmė yra tikimybė, kad pikselio reikšmė pateks į reikšmių diapazoną nuo iki tam tikros reikšmės. Stulpelių diapazonai auga ir apima kiekvieno prieš tai buvusio stulpelio reikšmių diapazoną. Todėl stulpelių reikšmės visada didėja, paskutinio stulpelio reikšmė visada yra 1.
3. Įprastos spalvos histogramų reikšmes padaliname iš pikselių kiekio. Taip gauname tikimybes, kad atsitiktinis pikselis pateks į reikšmių rėžį, apibrėžtą stulpelio. Tokio tipo histogramoje nebelieka informacijos apie segmento dydį.

Kiekviena histograma klasterizuota su trimis stulpelių kiekio dydžiais: 10, 100 ir 256. Klasteriai sudaryti iš 717 segmentų, gautų iš 200 LIME paaiškinimų. Iš paaiškinimų atrinkti tik tie segmentai, kuriems LIME algoritmas priskyrė pasikliovimo vertę didesnę už 0,1. Pagal naudotas metrikas kokybiškiausi rezultatai pasiekti naudojanti 256 stulpelių spalvų tikimybių histogramą (5 lentelė), visos lentelės pateikiamos (žr. 1 priedą). *DBSCAN* ir *Birtch* algoritmai sugeneravo sugeneruoti klasteriai lyginant su likusiais dvejais algoritmais buvo prastesni (5 lentelė). Šie algoritmai analizuojamus duomenis suskirsto į mažą kiekį klasterių kiekį (5, 6

lentelės). Nors pagal pagal kai kurias metrikas jie lenkia *Affinity prapogation* ir *MeanShift* algoritmus. (5 lentelėje) *DBSCAN Silhouette* metrika pati geriausia, *Birtch* algoritmo *Davies-Bouldin* indeksas geriausias. Tačiau šios metrikos, kaip ir *Calinski and Harabaz,* priklauso ir nuo sudarytų klasterių kiekio. Todėl patikimiausia būtų lyginti šias metrikas, kai sudarytas klasterių kiekis vienodas. Šiuo atveju to padaryti negalime, bet galime atsižvelgti į metrikų santykį. *MeanShift* klasterizacijos *Calinski and Harabaz* įvertis tik nežymiai mažesnis už *Affinity propagation*, nors klasterių skaičius skiriasi daugiau nei pusantro karto.

5. lentelė. Klasterių sugeneruotų iš spalvų tikimybių histogramos metrikos. Pagal metrikas geriausios klasterizacijos atliktos MeanShift ir Affiity propagation algoritmais

Spalvų tikimybių histograma 256 stulpelių				
	Silhouette	*Calinski and Harabaz*	*Davies-Bouldin*	Klasterių kiekis
Affinity propagation	0,057	96,421	0,947	109
DBSCAN	0,735	758,042	0,974	4
MeanShift	0,367	88,576	0,273	66
Birtch	0,621	218,339	0,163	12

6. lentelė. Klasteriai sugeneruoti naudojant pasiskirstymo funkcijos histogramas pasižymi mažu klasteriu kiekiu.

Pasiskirstymo funkcija 256 stulpelių				
	Silhouette	*Calinski and Harabaz*	*Davies-Bouldin*	Klasterių kiekis
Affinity propagation	0,188	184,108	1,301	30
DBSCAN	0,277	36,335	0,583	3
MeanShift	0,319	41,481	0,582	2
Birtch	0,302	461,457	1,004	3

MeanShift identifikuotas kaip geriausiai klasterius su duotais duomenimis sudaręs klasteris. Gautas klasterių kiekis, atsižvelgiant į tai kad dauguma segmentų sudaryti iš vieno atspalvio, maždaug tokio dydžio kokio buvo galima tikėtis. *Affinity propagation* algoritmu taip pat sudaryta pakankamai geras klasterių skaičius, tačiau *Silhouette* ir *Davies-Buoldin* metrikos buvo prastesnės už *MeanShift* gautas metrikas. Todėl *MeanShift* algotirmas pasirinktas naudoti atliekant LIME submodulinio pasirinkimo pritaikymą paveikslėliams.

4.3. Modifikuoto submodulinio pasirinkimo algoritmo paveikslėliams taikymas

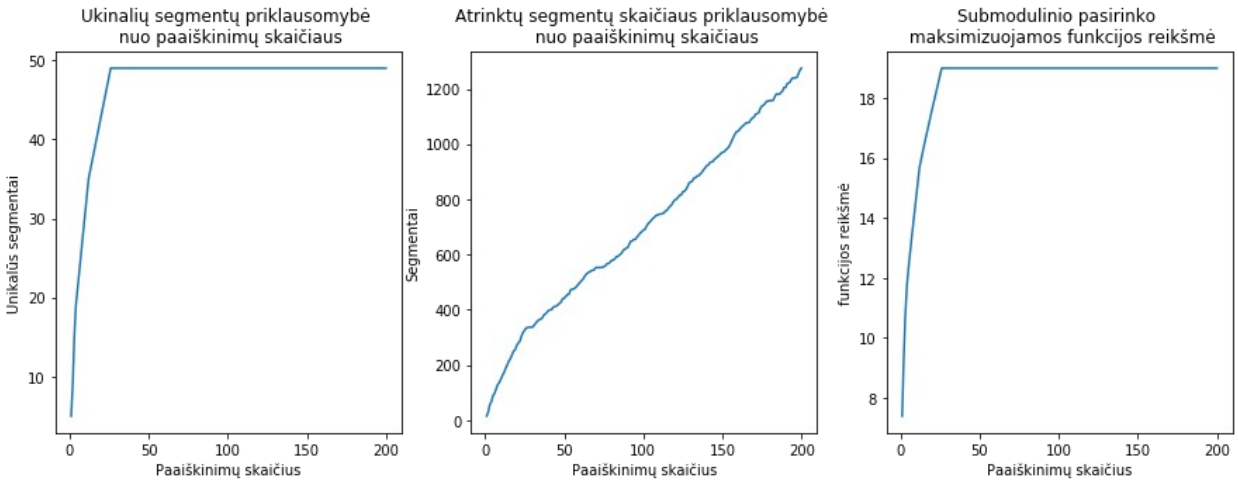
Modifikuotas algoritmas pritaikytas 200 atsitiktinai parinktų paveikslėlių iš COCO 2017 validavimo duomenų rinkinio. LIME algoritmu gauti dviejų tipų parametrai: LIME algoritmą taikant 100 iteracijų kiekį, ir 500 iteracijų kiekį. Taikant 500 iteracijų kiekį gauti paaiškinimai turėtų būti kokybiškesni. Tyrimu bus bandoma nustatyti kaip skiriasi submodulinio pasirinkimo rezultatas, kai paaiškinimai gauti taikant skirtingą iteracijų kiekį. Taip pat šiame skyriuje

pristatyti ir aptarti modifikuoto submodulinio pasirinkimo algoritmo rezultatai. Dėl tikslesnių paaiškinimų rezultatų atliekant modifikacijos tyrimą naudoti LIME paaiškinimai gauti pritaikius 500 iteracijų. Paaiškinimai gauti pritaikius 100 iteracijų naudoti tik rezultatų palyginimui.

Paaiškinimai, gauti trečiajame darbo skyriuje aprašytu *vandenskyros* algoritmu, nes nustatyta, kad juo sugeneruoti segmentai kokybiškai įtakoja LIME paaiškinimų radimo algoritmą. Atrenkant segmentus, kurie bus naudojami submodulinio pasirinkimo algoritmu, nustatyti kriterijai, kad segmento svorio įvertis būtų didesnis už 0.05 arba mažesnis už -0.05. Neigiamas svorio įvertis indikuoja, kad segmentas turi neigiamą įtaką atpažįstant nustatytą klasę. Gauti 1275 segmentai, kurie *MeanShift* algoritmu suklasifikuoti į 49 klases.

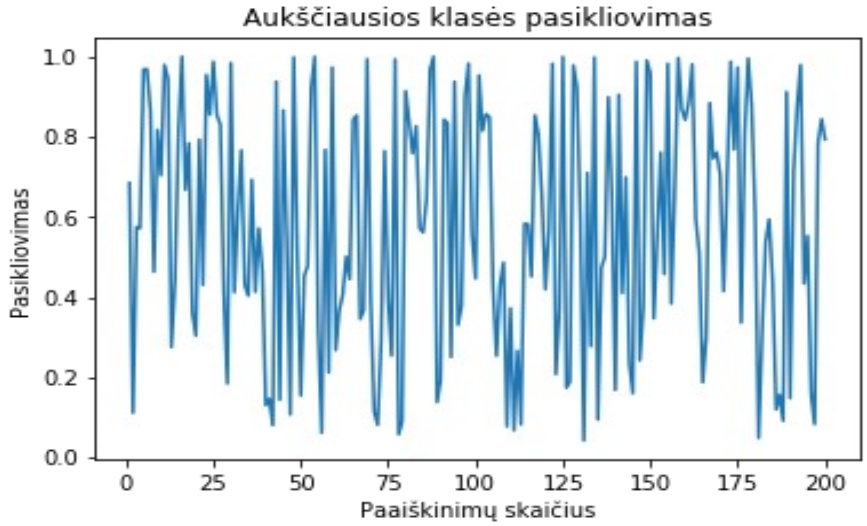
Taip pat buvo išbandyta atrinkti segmentus, kurių svorio įvertis didesnis už 0,1 arba mažesnis už 0,1. Tačiau gautas salyginai mažesnis kiekis segmentų (470), kurie buvo suskirstyti į mažą klasterių kiekį (7) naudojant *MeanShift* algoritmą. Buvo išbandyti ir kiti algoritmai, tačiau nors ir gautas didesnis klasterių kiekis (33 klasteriai su *Affinity propagation*) gautos metrikos indikavo prastai sudarytus klasterius. Pagal gautus klasterizavimo rezultatus galime daryti išvadą, kad tam, kad pasiekti geresnius segmentų klasterius, rekomenduotina naudoti kuo didesnį segmentų kiekį.

Pasirinktiems duomenims pritaikius submodulinio pasirinkimo algoritmą, naudojant nemodifikuotą aprėpties funkciją, visi unikalūs segmentai padengti pirmaisiais 26 paaiškinimais (14 pav. dešinysis grafikas). Toliau sekantys paaiškinimai atrenkami atsitiktiniu būdu. Analizuojant, kaip atrinkti pasirinkimai koreliuoja su modelio spėjimų pasikliautimi, pastebimas nežymus pasikliovimo vidurkių padidėjimas pirmuose 26 atrinktuose paaiškinimuose lyginant su visų 200 paaiškinimų gauta pasikliautimi. Pirmų 26 paaiškinimų aukščiausios klasės



14. pav. Submodulinio pasirinkimo charakteristikų grafikai.

pasikliauties vidurkis lygus 0,70, visos 200 paaiškinimų imties aukščiausios klasės pasikliauties vidurkis lygus 0,58 (15 pav.). Tarp neatsitiktine tvarka atrinktų pirmųjų paaiškinimų dominuoja paaiškinimai tų paveikslėlių, kuriuose modelis su aukštu pasikliovimu atpažino vieną klasę. Tačiau atrinkta ir keletas paaiškinimų paveikslėlių su žemu aukščiausios klasės atpažinimo pasikliautini (15 pav.). Jeigu modelio rezultatams būtų taikomas slenkstis, šie pavyzdžiai galėtų būti identifikuoti kaip klaidingi.

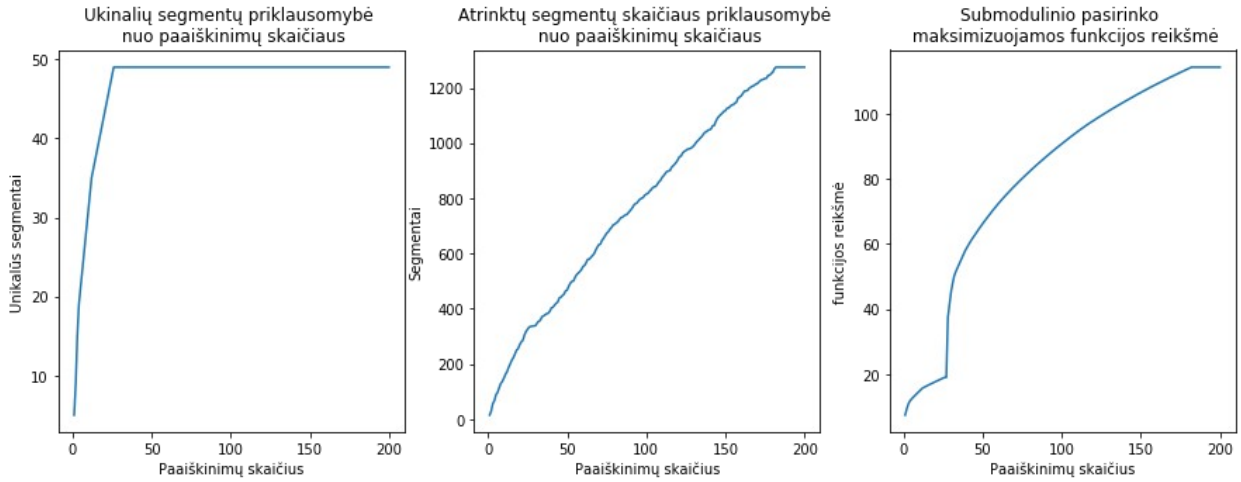


15. pav. Paaiškinimų aukščiausios klasės pasikliovimo reikšmės

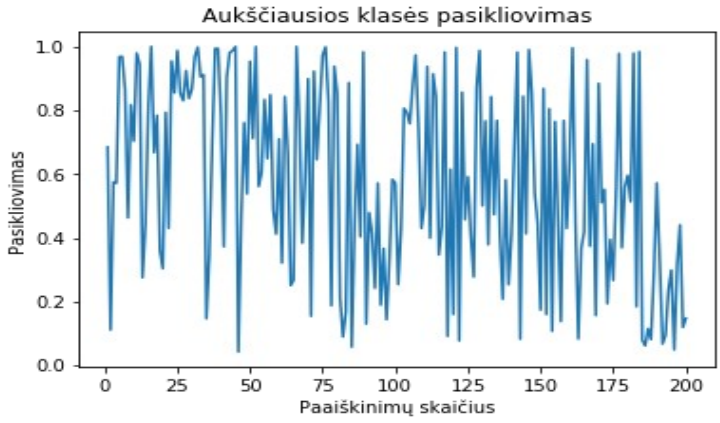
Pirmieji paaiškinimai, atrinkti submodulinio pasirinkimo algoritmu, vartotojui pateikia pavyzdžių su dideliu kiekiu unikalių atributų, pirmi 5 atrinkti paaiškinimai sudaryti iš 89 segmentų, kurie reprezentuoja 21 unikalų atributą (žr. 2 priedą). Tačiau, jei galime peržiūrėti didelį kiekį pavyzdžių, susiduriame su problema, kad kai atrinkti pavyzdžiai padengia visus atributus, kiti pavyzdžiai parenkami atsitiktinai. Šią problemą atspindi ir (14 pav.) dešinėje esantis grafikas – pasiekus 26 paaiškinimus maksimizuojama funkcija nustoja augti. Siekiant išspręsti šią problemą, tol parenkami pavyzdžiai padengs visus atributus bus taikoma (1) funkcija. Kai visi atributai bus padengti ir (1) funkcijos reikšmės nustos augti, bus pradėta taikyti (3) funkcija.

Pritaikius tokią dviejų funkcijų kombinaciją su naudojamais duomenimis, pirmieji 26 segmentai nepakito, nes buvo atrinkti pagal tą pačią funkciją, tai matyti ir iš (16 pav.) kairiojo grafiko. Tačiau sekantys pasirinkimai skiriasi, nes atrinkti maksimizuojant (3) funkciją. Kada buvo pradėta taikyti ši funkcija, matyti iš staigaus šuolio (16 pav.) dešiniajame grafike. Likus paskutiniams keliems paaiškinimams maksimizuojama funkcija nustoja augti, tai nutiko dėl to, kad likę paaiškinimai yra „tušti“ – neturi nei vieno segmento. Tai patvirtina ir (16 pav.)

vidurinysis grafikas, matome, kad pridedant paskutinius pavyzdžius atrinktų segmentų kiekis nedidėja. Iš (17 pav.) matyti, kad paveikslėlių, kuriems LIME algoritmas nesugebėjo sugeneruoti paaiškinimų, aukščiausios klasės pasikliovimo įverčiai yra žemi. 18 paskutinių paaiškinimų neturi segmentų, jų vidutinis aukščiausios klasės pasikliovimo įvertis – 0,25, kai bendras visų duomenų vidurkis lygus 0,58.



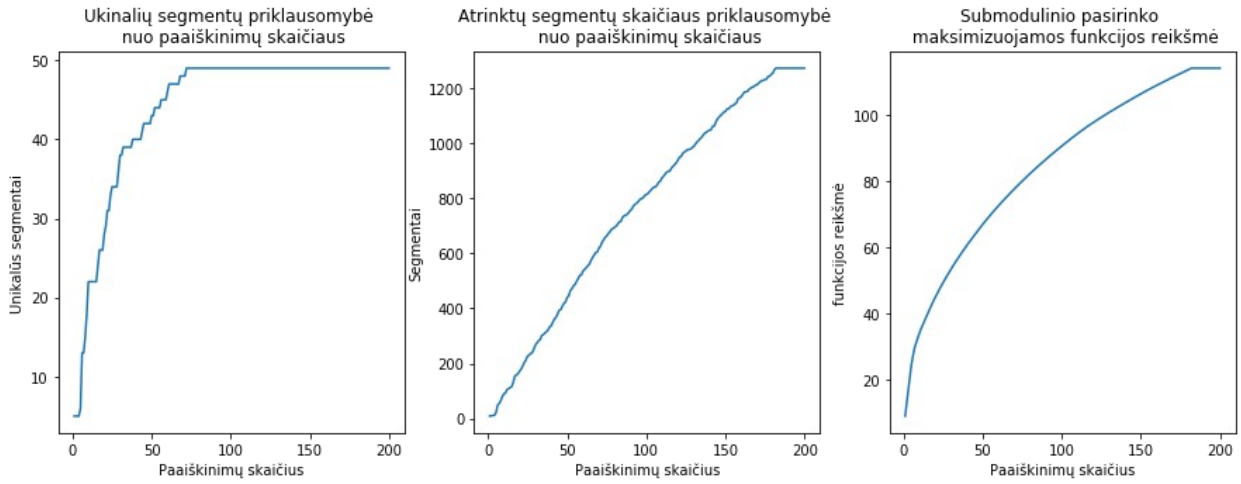
16 pav. Submodulinio pasirinkimo algoritmo maksimizuojančio (1) ir (3) funkcijas charakteristikos



17 pav. Paaiškinimų atrinktų submodulinio pasirinkimo algoritmo maksimizuojančio (1) ir (3) funkcijas aukščiausios klasės pasikliovimo reikšmės.

Jeigu maksimizuojama tik (3) funkcija nuo pat algoritmo pradžios unikalūs segmentai užsipildo lėčiau. Visi 49 atributai pirmuose 72 atrinktuose paaiškinimuose. (18 pav.) kairėje esantis grafikas pasidaro laiptuotas, atsiranda atvejų, kai submodulinio parinkimo algoritmas prideda paaiškinimus su jau esamais atributais. Tai įvyksta dėl, to kad paaiškinimo, turinčio aukštus esamų atributų svorius pridėjimas nusvėrė paaiškinimo su naujais atributais pridėjimą. Nors, pridedant pavyzdžius su naujais atributais. ir taikomas prioritetas. Prioritetas (3) funkcijoje

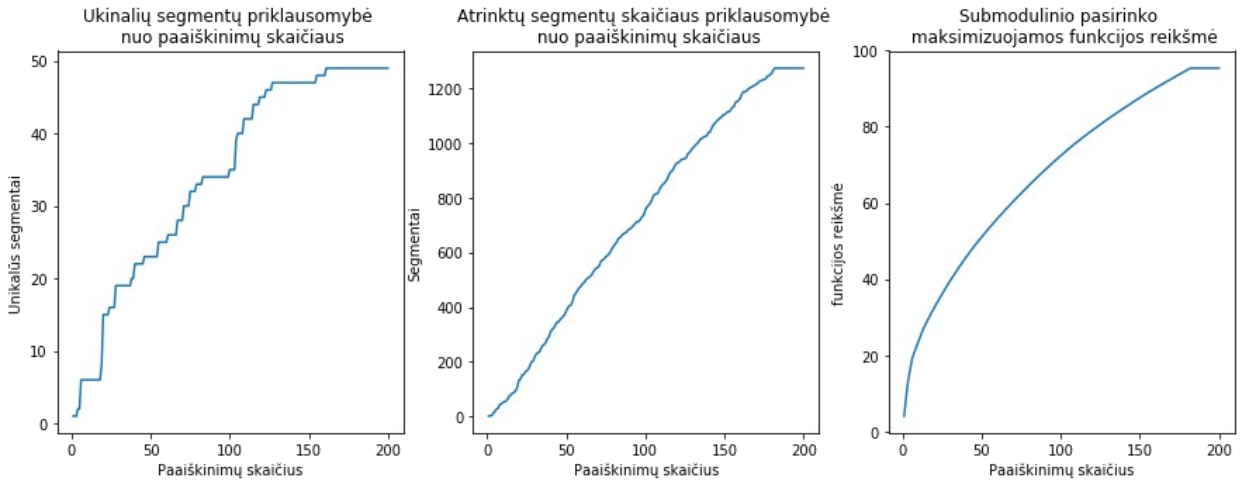
pasireiškia tuo, kad pridėjus pavyzdį su nauju atributu i, (3) funkcijos reikšmė padidės bent $1*I_j+W_{ij}*I_j$, o jeigu atributas i, jau yra pridėtas ankstesniuose pavyzdžiuose, tuomet funkcija padidės $W_{ij}*I_j$. Nors ir taikomas prioritetas, matome, kad jau atrinkti atributai su didesnėms $W_{ij} ir I_j$ reikšmėmis gali nustelbti naujų atributų pridėjimą.



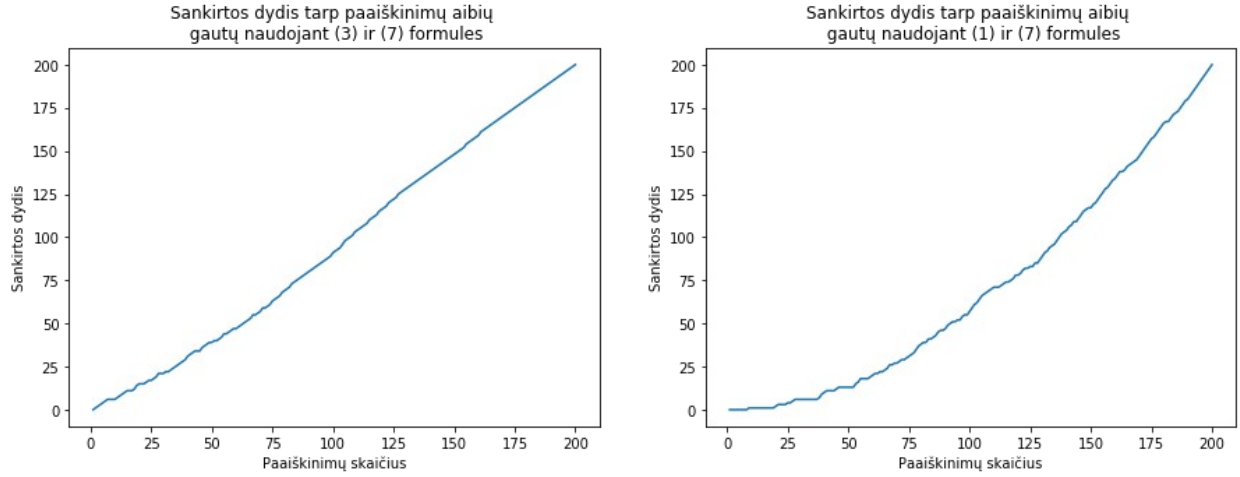
18 pav. Submodulinio pasirinkimo algoritmo maksimizuojančio (3) funkciją charakteristikos

Jeigu formulę (3) pakeistumėm į formulę (7), kuri atsižvelgia tik į atributų svarbumą. Taikant šią formulę pirmiausia atrenkami paaiškinimai, kurie turi dažniausiai pasikartojančius atributus su didžiausiais svoriais svoriais. Dėl to visi unikalūs atributai atrenkami tik su 162 paaiškinimais (19 pav. kairysis grafikas). Matyti, kad pirmieji paaiškinimai, atrinkti maksimizuojant šią formulę (žr. 3 priedą), skiriasi nuo paaiškinimų, gautų maksimizuojant straipsnyje aprašytą (1) formulę (žr. 2 priedą). Maksimizuojant (1) formulę pirmuosiuose paaiškinimuose gausu skirtingų segmentų, maksimizuojant (7) formulę paaiškinimą sudaro vienas ar keli segmentai, jie dažniausiai padengia didelį atpažįstamo objekto plotą ir įvertinti kaip turintys didelį svorį. Maksimizuojant (3) ir (7) formules atrenkami gana panašūs rezultatai. Jeigu paimtumėm sankirtą tarp pirmų atrinktų 30 paaiškinimų, gauname, kad šias aibes sudaro 21 bendras paaiškinimas (20 pav. kairysis grafikas). Tuo tarpu, jeigu skaičiuojame tą pačią sankirtą tarp algoritmų, maksimizuojančių (1) ir (7) formules, sankirtos dydis yra 6 (20 pav. dešinysis grafikas) .

$$c(V,W,I)=\sum_{j=1}^{d'}\sum_{i\in V} W_{ij}I_j \quad (7)$$

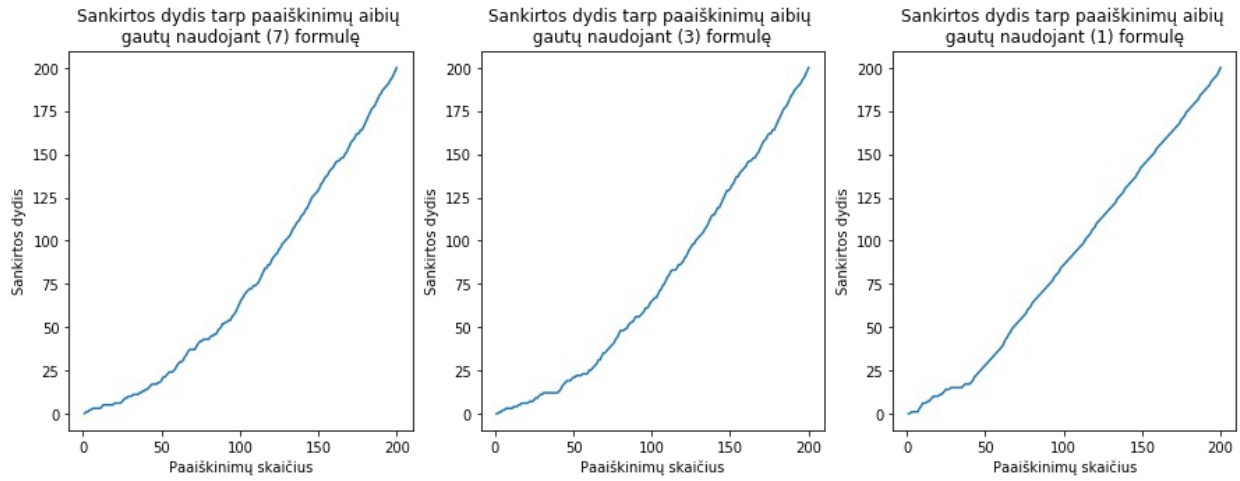


19 pav. Submodulinio pasirinkimo algoritmo, maksimizuojančio (7) funkciją charakteristikos



20 pav. Sankirtos tarp atrinktų paaiškinimų aibių maksimizuojant skirtingas formules

Lyginant atrinktų paaiškinimų panašumus maksimizuojant tą pačią formulę, bet naudojant skirtingus duomenis – paaiškinimus gautus LIME algoritmu atlikus 100 iteracijų ir atlikus 500 iteracijų. Nustatyta, kad panašiausi pavyzdžiai atrinkti maksimizuojant (1) funkciją (21 pav. grafikas dešinėje). Maksimizuojant (1) funkciją, atrinkus 100 paaiškinimų sutapo 84 iš jų, maksimizuojant (3) ir (7) sutapo atitinkamai 64 ir 65 paaiškinimai. Naudojant (3) ir (7) funkcijas submodulinio atrinkimo algoritmas pasidaro jautresnis paaiškinimų segmentams suteiktiems svoriams, šių svorių reikšmės yra labai priklausomos nuo LIME algoritmo iteracijų skaičiaus. Naudojant (1) funkciją didesnis dėmenys skiriamas unikaliems segmentams, atrinkus visus unikalius segmentus sekantys parenkami iš eilės. Dėl to naudojant (1) funkciją pastebima didesnė sankirta.



21 pav. Sankirtos tarp paaiškinimų sugeneruotu LIME algoritmu atlikus 100 iteracijų ir atlikus 500 iteracijų.

Rezultatai

1. Atliekant literatūros analizę ištirti ir palyginti giliųjų neuroninių tinklų vizualizavimo metodai. Nustatyti atviri uždaviniai.
2. Atliktas vaizdo segmentavimo tyrimas, skaitiškai palyginti ir įvertinti šeši segmentavimo algoritmai.
3. Submodulinio pasirinkimo algoritmas, kuris originaliai pritaikytas darbui su skaitiniais tekstiniais duomenimis, praplėstas darbui su vaizdais įvedant klasterizavimo žingsnį. Atliktas klasterizavimo algoritmų tyrimas. Pasiūlytos alternatyvos submodulinio pasirinkimo maksimizuojamai funkcijai.

Išvados

1. Dėl savo naujumo ir aktualumo tolimesniam tyrimui pasirinktas LIME metodas. Šio metodo vaizdų segmentavimo uždavinys ir pavyzdžių atrinkimo algoritmas identifikuoti kaip kandidatai įgyvendinti praplėtimus.
2. Atlikus segmentavimo algoritmų tyrimą, nustatyta, kad su LIME paaiškinimų sudarymo algoritmu *quickshift*, *felzenszwal,* ir vandenskyros segmentavimo algoritmai veikė efektyviausiai. Atliekant paaiškinimų sudarymą rekomenduotina remtis šiais segmentavimo algoritmais, taip padidinant tikimybę gauti geresnius rezultatus.
3. Įgyvendintas praplėtimas suteikia galimybę naudoti submodulinio pasirinkimo algoritmą su vaizdų duomenimis. Pasiūlytos maksimizuojamos funkcijos modifikacijos garantuoja, kad maksimizuojamos funkcijos reikšmės, pridedant naujus netuščius paaiškinimus,

didės viso algoritmo metu. Ši savybė leidžia atrinkti didelį kiekį paaiškinimų ir užtikrina, kad jie bus išrikiuoti pagal prasmingumą. Kelios skirtingos modifikacijos suteikia vartotojui galimybę pasirinkti ar prioritizuoti kuo didesnį skirtingų atributų skaičių ar didelius svorius turinčius svarbius atributus.

Literatūra

[ASS+12] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua. Sabine Süsstrunk ir k.t. Slic superpixels compared to state-of-the-art superpixel methods. Ieee transactions on pattern analysis and machine intelligence, 34(11):2274–2282, 2012.

[ACD+15] S. Amershi, M. Chickering, S. M. Drucker, B.Lee, P. Y. Simard, J. Suh. ModelTracker: Redesigning Performance Analysis Tools for Machine Learning. Annual ACM Conference on Human Factors in Computing Systems, p. 337-346, Seoul, Republic of Korea. CHI 2015.

[BCC+16] M. Bojarski, A.Choromanska, K. Choromanski, B. Firner, L. D. Jackel, U. Muller, K. Zieba. VisualBackProp: visualizing CNNs for autonomous driving. CoRR, abs/1611.05418, 2016

[CH74] Tadeusz Caliński ir Jerzy Harabasz. A dendrite method for cluster analysis. Communications in statistics-theory and methods, 3(1):1–27, 1974.

[Che95] Yizong Cheng. Mean shift, mode seeking, and clustering. Ieee transactions on pattern analysis and machine intelligence, 17(8):790–799, 1995.

[CKS97] Vicent Caselles, Ron Kimmel ir Guillermo Sapiro. Geodesic active contours. International journal of computer vision, 22(1):61–79, 1997.

[CM02] Dorin Comaniciu ir Peter Meer. Mean shift: a robust approach toward feature space analysis. Ieee transactions on pattern analysis & machine intelligence, (5):603–619, 2002.

[DB79] David L Davies ir Donald W Bouldin. A cluster separation measure. Ieee transactions on pattern analysis and machine intelligence, (2):224–227, 1979.

[DBC+10] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, J. W. Vaughan. A theory of learning from different domains. Machine Learning, 79:151–175, 2010.

[DDS+09] J. Deng, W. Dong, R. Socher, Li-Jia Li, K. Li, Fei-Fei Li. ImageNet: A large-scale hierarchical image database. Computer Society Conference on Computer Vision and Pattern Recognition, Miami, Florida, USA. 2009, p. 248-255.

[DK17] F. Doshi-Velez, B. Kim. Towards A Rigorous Science of Interpretable Machine

Learning. CoRR, abs/1702.08608, 2017.

[EBC+09] D. Erhan, Y. Bengio, A. Courville, P. Vincent. Visualizing Higher-Layer Features of a Deep Network. echnical report, University of Montreal, p. 1341, 2009.

[EKS+96] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu ir k.t. A density-based algorithm for discovering clusters in large spatial databases with noise. Kdd. Tom. 96. (34), 1996, p.p. 226–231.

[EPTR06] Europs parlamento ir tarybos Reglamentas (ES) 2016/679. 2016 m. balandžio 27d. [žiūrėta2018-06-10]. Prieiga per Internetą: <https://eur-lex.europa.eu/legal-content/LT/TXT/?uri=celex%3A32016R0679>

[FD07] Brendan J Frey ir Delbert Dueck. Clustering by passing messages between data points. Science, 315(5814):972–976, 2007.

[Fei98] Uriel Feige. A threshold of ln n for approximating set cover. Journal of the acm (jacm), 45(4):634–652, 1998.

[FH04] Pedro F Felzenszwalb ir Daniel P Huttenlocher. Efficient graph-based image segmentation. International journal of computer vision, 59(2):167–181, 2004.

[Fis36] Ronald A Fisher. The use of multiple measurements in taxonomic problems. Annals of eugenics, 7(2):179–188, 1936.

[FV14] B. Frénay, M. Verleysen. Classification in the Presence of Label Noise: A Survey. IEEE Transactions on Neural Networks and Learning Systems, Volume 25, 2014,p. 845-869.

[GF17] B. Goodman, S. Flaxman. European Union regulations on algorithmic decision-making and a "right to explanation". AI Magazine 38(3) 2017, p. 50-57.

[HEM+98] Kostas Haris, Serafim N Efstratiadis, Nikolaos Maglaveras ir Aggelos K Katsaggelos. Hybrid image segmentation using watersheds and fast region merging. Ieee transactions on image processing, 7(12):1684–1699, 1998.

[HZS+16] K. He, X. Zhang, S. Ren, J. Sun. Deep Residual Learning for Image Recognition. IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA. p. 770-778.

[KL17] P. W. Koh, P. Liang. Understanding Black-box Predictions via Influence Functions. International Conference on Machine Learning, ICML 2017, Sydney, Australia, p. 1885-1894.

[KSH12] A. Krizhevsky, I. Sutskever, G. E. Hinton. ImageNet Classification with Deep

Convolutional Neural Networks. 26th Annual Conference on Neural Information Processing Systems, Lake Tahoe, Nevada, USA, 2012. p. 1106-1114.

[LCB99] Yann LeCun, Corinna Cortes, Christopher J.C. Burges. The MNIST Dataset Of Handwritten Digits. [žiūrėta 2017-12-05]. Prieiga per Internetą: <http://yann.lecun.com/exdb/mnist/>

[MBA14] Pablo Marquez-Neila, Luis Baumela ir Luis Alvarez. A morphological approach to curvature-based evolution of curves and surfaces. Ieee transactions on pattern analysis and machine intelligence, 36(1):2–17, 2014.

[MOT15] A. Mordvintsev, C. Olah, M. Tyka. Inceptionism: Going Deeper into Neural Networks. [žiūrėta 2017-12-05]. Prieiga per Internetą: <https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html>

[NP14] Peer Neubert ir Peter Protzel. Compact watershed and preemptive slic: on improvingtrade-offsof superpixelsegmentationalgorithms. Pattern recognition (icpr), 2014 22nd international conference on. IEEE, 2014, p.p. 996–1001.

[NYC15] A. M. Nguyen, J. Yosinski, J.Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, p. 427-436.

[OMS17] C. Olah, A. Mordvintsev, L. Schubert. Feature Visualization. [žiūrėta2017-12-05]. Prieiga per Internetą: <https://distill.pub/2017/feature-visualization/>

[Rou87] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of computational and applied mathematics, 20:53–65, 1987.

[RDS+15] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, Fei-Fei Li. ImageNet Large Scale Visual Recognition Challenge. 2015, International Journal of Computer Vision 115(3), p. 211-252.

[RF17] J. Redmon, A. Farhadi. YOLO9000: Better, Faster, Stronger. Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA. p. 6517-6525.

[RM00] Jos BTM Roerdink ir Arnold Meijster. The watershed transform: definitions,

algorithmsand parallelization strategies. Fundamenta informaticae,41(1, 2):187–228, 2000.

[RSG16a] M. T. Ribeiro, S. Singh, C. Guestrin. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. 2016. Knowledge Discovery and Data Mining (KDD), San Francisco, CA, USA, p. 1135-1144.

[RSG16b] M. T. Ribeiro, S. Singh, C. Guestrin. Model-Agnostic Interpretability of Machine Learning. CoRR, abs/1606.05386, 2016.

[SFH17] S. Sabour, N. Frosst, G. E. Hinton. Dynamic Routing Between Capsules. Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA, p. 3859-3869.

[SHG+15] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J. Crespo, D. Dennison. Hidden Technical Debt in Machine Learning Systems. Advances in Neural Information Processing Systems, 2015, p. 2503-2511.

[SLJ+15] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich. Going deeper with convolutions. IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015,Boston, MA, USA, p. 1-9.

[STOT14] 2014 SIGKDD Test of Time Award [žiūrėta 2007-04-15]. Prieiga per Internetą: <https://www.kdd.org/News/view/2014-sigkdd-test-of-time-award>

[SVI+16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna. Rethinking the Inception Architecture for Computer Vision. Computer Vision and Pattern Recognition. IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, p. 2818–2826.

[SZS+13] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, R. Fergus. Intriguing properties of neural networks. CoRR. Abs/1312.6199, 2013.

[VS08] Andrea Vedaldi ir Stefano Soatto. Quick shift and kernel methods for mode seeking. European conference on computer vision. Springer, 2008, p.p. 705–718.

[ZCA+17] L. M. Zintgraf, T. S. Cohen, T. Adel, M. Welling. Visualizing Deep Neural Network Decisions: Prediction Difference Analysis. CoRR, abs/1702.04595, 2017.

[ZF13] M. D. Zeiler, R. Fergus. Visualizing and Understanding Convolutional

Networks. CoRR, abs/1311.2901, 2013.

[ZRL96] Tian Zhang, Raghu Ramakrishnan ir Miron Livny. Birch: an efficient data clustering method for very large databases. Acm sigmod record. Tom. 25. (2). ACM, 1996, p.p. 103–114.

[ZTF11] M. D. Zeiler, G. W. Taylor, R. Fergus. Adaptive deconvolutional networks for mid and high level feature learning. ICCV 2011, Barcelona, Spain. p. 2018-2025.

Priedai

1 priedas (klasterių įvertinimo lentelės)

Pasiskirstymo funkcija 10 stulpelių				
	Silhouette	Calinski and Harabaz	Davies-Bouldin	Klasterių kiekis
Affinity propagation	0,175	130,875	0,959	46
DBSCAN	0,157	25,131	2,09	4
MeanShift	0,271	42,552	0,849	2
Birtch	0,306	456,247	1,001	3

Pasiskirstymo funkcija 100 stulpelių				
	Silhouette	Calinski and Harabaz	Davies-Bouldin	Klasterių kiekis
Affinity propagation	0,18	128,147	0,972	49
DBSCAN	-0,437	13,261	1,645	10
MeanShift	0,318	41,421	0,583	2
Birtch	0,297	469,864	1,085	3

Pasiskirstymo funkcija 256 stulpelių				
	Silhouette	Calinski and Harabaz	Davies-Bouldin	Klasterių kiekis
Affinity propagation	0,188	184,108	1,301	30
DBSCAN	0,277	36,335	0,583	3
MeanShift	0,319	41,481	0,582	2
Birtch	0,302	461,457	1,004	3

Spalvų histograma 10 stulpelių				
	Silhouette	Calinski and Harabaz	Davies-Bouldin	Klasterių kiekis
Affinity propagation	0,034	99,615	0,869	207
DBSCAN	-0,46	4,367	1,292	3
MeanShift	0,562	151,761	0,627	39
Birtch	0,691	711,805	0,809	3

Spalvų histograma 100 stulpelių				
	Silhouette	Calinski and Harabaz	Davies-Bouldin	Klasterių kiekis
Affinity propagation	0,326	236,726	0,693	100
DBSCAN	-0,464	4,092	1,337	3
MeanShift	0,5	119,116	0,582	58
Birtch	0,703	581,403	0,865	3

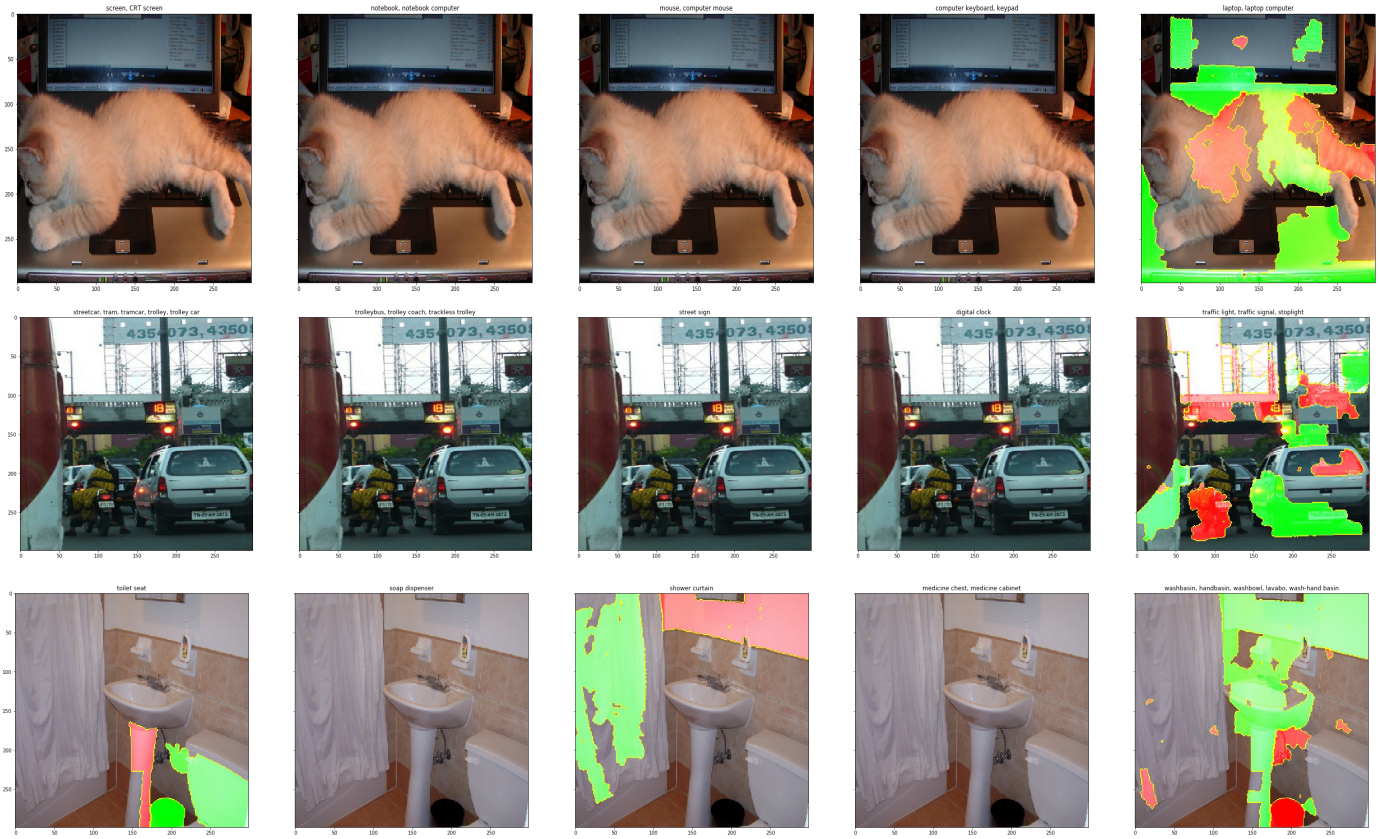
Spalvų histograma 256 stulpelių				
	Silhouette	Calinski and Harabaz	Davies-Bouldin	Klasterių kiekis
Affinity propagation	0,348	216,92	0,689	102
DBSCAN	-0,463	4,066	1,343	3
MeanShift	0,492	107,483	0,606	59
Birtch	0,694	573,317	0,864	3

Spalvų tikimybių histograma 10 stulpelių				
	Silhouette	Calinski and Harabaz	Davies-Bouldin	Klasterių kiekis
Affinity propagation	0,104	42,215	1,165	73
DBSCAN	0,346	101,636	1,303	4
MeanShift	0,304	91,09	0,898	5
Birtch	0,454	157,633	0,728	3

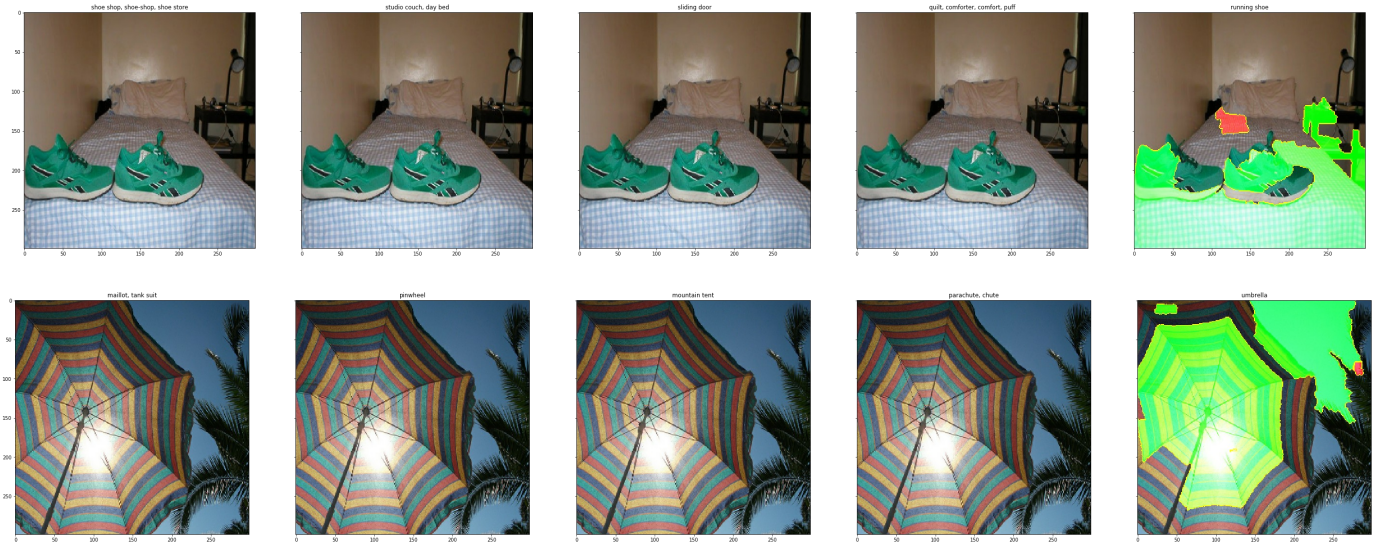
Spalvų tikimybių histograma 100 stulpelių				
	Silhouette	Calinski and Harabaz	Davies-Bouldin	Klasterių kiekis
Affinity propagation	0,078	68,872	0,738	105
DBSCAN	0,66	503,353	1,002	4
MeanShift	0,288	53,634	0,362	51
Birtch	0,179	106,357	0,43	26

Spalvų tikimybių histograma 256 stulpelių				
	Silhouette	Calinski and Harabaz	Davies-Bouldin	Klasterių kiekis
Affinity propagation	0,057	96,421	0,947	109
DBSCAN	0,735	758,042	0,974	4
MeanShift	0,367	88,576	0,273	66
Birtch	0,621	218,339	0,163	12

2 priedas (pirmi 5 paaiškinimai gauti LIME submodulinio pasirinkimo algoritmu)



3 priedas (pirmi 5 paaiškinimai gauti LIME submodulinio pasirinkimo algoritmu maksimizuojant (7) formulę)


shoe shop, shoe-shop, shoe store
studio couch, day bed
sliding door
quilt, comforter, comfort, puff
running shoe
maillot, tank suit
pinwheel
mountain tent
parachute, chute
umbrella