Kauno technologijos universitetas

Elektros ir elektronikos fakultetas

Ligotų kviečių atpažinimo metodų tyrimas

Baigiamasis magistro projektas

Vaidas Grigonis

Projekto autorius

prof. dr. Vidas Raudonis

Vadovas

Kaunas, 2021

Kauno technologijos universitetas

Elektros ir elektronikos fakultetas

Ligotų kviečių atpažinimo metodų tyrimas

Baigiamasis magistro projektas

Valdymo technologijos (6211EX014)

Vaidas Grigonis

Projekto autorius

prof. dr. Vidas Raudonis

Vadovas

lekt. dr. Vytautas Gargasas

Recenzentas

Kaunas, 2021

Kauno technologijos universitetas

Elektros ir elektronikos fakultetas

Vaidas Grigonis

Ligotų kviečių atpažinimo metodų tyrimas

Akademinio sąžiningumo deklaracija

Patvirtinu, kad:

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

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

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

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

Vaidas Grigonis

Patvirtinta elektroniniu būdu

Grigonis Vaidas. Ligotų kviečių atpažinimo metodų tyrimas. Magistro baigiamasis projektas / vadovas prof. dr. Vidas Raudonis; Kauno technologijos universitetas, Elektros ir elektronikos fakultetas.

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

Reikšminiai žodžiai: *TensorFlow* biblioteka, *Faster R – CNN , R – FCN , Inception_V2, ResNet – 101*, klaidų matrica.

Kaunas, 2021. 42 p.

Santrauka

Žemės ūkio pramonės sektorius yra svarbi šiandieninės ekonomikos sudedamoji dalis. Žemdirbystės produkcija yra nesunkiai paveikiama įvairių augalų ligų, kurios sukelia ekologinę, ekonominę žalą, todėl kaip ir daugelyje sričių, ne išimtis ir žemės ūkyje, yra siekiama taikyti vis pažangesnius techninius sprendimus norint išgauti didesnį efektyvumą. Darbe atlikta literatūros analizė, kurioje aprašoma mokslo pasaulyje įgyvendintų tyrimų apžvalga susijusia tema. Taip pat šiame darbe atlikti tyrimai, kurių pagalba yra siekiama įvertinti skirtingų neuroninių tinklų struktūrų efektyvumą, atpažįstant rūdžių ligos paveiktus kviečių lapus ir stiebus. Atliekant tyrimus, panaudotos *Faster R – CNN* ir *R – FCN* architektūros bei *Inception_V2* ir *ResNet – 101* neuroninių tinklų struktūros. Visa tai įgyvendinta naudojant *TensorFlow* biblioteką. Rezultatams pateikti panaudotos klaidų matricos, grafikai, kuriuose pavaizduotos *False / Negative, F – Score, Accuracy* reikšmių priklausomybės kintant modelio parametrams.

Vaidas Grigonis. Research of diseased wheat recognition methods. Master's Final Degree Project / supervisor prof. dr. Vidas Raudonis; Faculty of Electrical and Electronics Engineering, Kaunas University of Technology.

Study field and area (study field group): Electronics engineering (engineering science).

Keywords: *TensorFlow* library, *Faster R – CNN*, *R – FCN*, *Inception_V2*, *ResNet – 101*, *Confusion matrix.*

Kaunas, 2021. 42.

Summary

Agro – industry sector plays an important role in today‘s economy. Agricultural production is easily affected by various plant diseases that cause ecological and economical damage. So, as in many areas, no exceptions in agriculture, the aim is to apply advanced technical solutions to achieve greater efficiency. The paper analyzes the literature, which describes the review of research in the world of science on a related topic. This work also includes studies to evaluate the effectivnes of different structures in recognizing wheat leaves and stems affected by rust disease. *Faster R – CNN*, *R – FCN* architectures and *Inception_V2*, *ResNet – 101* structures were used in the research. All this was implemented using the *TensorFlow* library. The results are presented using *Confusion matrices* and graphics, showing the dependencies of *False / Negative*, *F – Score*, *Accuracy* values on changing model parameters.

Turinys

Lentelių sąrašas

Santrumpų ir terminų sąrašas

Darbe naudotų svarbiausių santrumpų sąrašas, kuris pateikiamas lentelėje (žr. 1 lentelė).

1 lentelė. Santrumpų sąrašas

Nr.	Santrumpa	Reikšmė
1	Faster R – CNN	Faster Region – based Convolutional Neural Network
2	R – FCN	Region – based Fully Convolutional Network
3	CUDA	Compute Unified Device Architecture
4	cuDNN	Deep Neural Network library
5	CNN	Convolutional Neural Network
6	RPN	Region Proposal Network
7	R – CNN	Region – based Convolutional Neural Network
8	RoI	Region of Interest
9	ResNet	Residual Neural Network
10	ReLU	Rectified Linear Unit
11	SVM	Support Vector Machine

Įvadas

Augalai yra svarbūs visai gamtai, kurie fotosintezės būdu naudojant anglies dioksidą, vandenį bei šviesą gamina deguonį. Augalai žmones aprūpina maistu, vaistais, drabužiais, anglimi, kuri yra augalinės kilmės iškastinė medžiaga. Žemės ūkio pramonės sektorius yra svarbi šiandieninės ekonomikos sudedamoji dalis. Žemdirbystės produkcija yra nesunkiai paveikiama įvairių augalų ligų, kurios sukelia ekologinę, ekonominę žalą. Vienos ligos yra lengvai pastebimos plika akimi, kitų aptikimui reikalinga įranga.

Kaip ir daugelyje sričių, ne išimtis ir žemės ūkyje, yra siekiama taikyti vis pažangesnius techninius sprendimus norint išgauti didesnį efektyvumą. Taip pat yra skatinamas ekologinis ūkis, todėl norima sumažinti pesticidų naudojimą, taip siekiant tausoti aplinką ir pagerinti produkcijos kokybę. Būtent tiksli ir ankstyva pasėlių ligų diagnozė, naudojant vaizdų atpažinimo technologijas, leidžia pagerinti produkcijos kokybę ir sumažinti galimus žemės ūkio įmonių nuostolius.

Darbe atlikta literatūros analizė, kurioje aprašoma mokslo pasaulyje atliktų tyrimų apžvalga susijusia tema. Taip pat šiame darbe atlikti tyrimai, kurių pagalba yra siekiama įvertinti skirtingų neuroninių tinklų struktūrų efektyvumą, atpažįstant rūdžių ligos paveiktus kviečių lapus ir stiebus. Atliekant tyrimus, panaudotos *Faster R – CNN* ir *R – FCN* architektūros bei *Inception_V2* ir *ResNet – 101* struktūros, o rezultatams pateikti panaudotos klaidų matricos, grafikai.

Darbo tikslas – ištirti rūdžių ligos paveiktų kviečių lapų ir stiebų atpažinimo efektyvumą, naudojant skirtingas konvoliucinių neuroninių tinklų struktūras.

Pagrindiniai darbo uždaviniai:

1. atlikti literatūros analizę ligų paveiktų pasėlių atpažinimo tema;
2. klasifikatorių treniravimui paruošti duomenų bazes, kurias sudarytų sveikų ir ligotų kviečių vaizdai;
3. eksperimentiniams tyrimams panaudoti *Inception_V2* ir *ResNet – 101* konvoliucinių neuroninių tinklų struktūras;
4. struktūroms parinkti skirtingus parametrus, siekiant išgauti kuo tikslesnius atpažinimo rezultatus;
5. palyginti rezultatus, gautus naudojant skirtingus parametrus, remiantis klaidų matricomis ir grafikais.

1. Apžvalginė dalis

Šioje dalyje yra atliekama literatūros apžvalga, kurios metu nagrinėjmi metodai, naudojami ligotų pasėlių atpažinimui pasitelkiant vaizdo apdorojimo technologijas. Analizuojami sukurti algoritmai, kurie taikomi tokio pobūdžio problemų sprendimui. Taip pat paminimos įmonės, kurios specializuojasi žemdirbystės srityje ir naudoja vaizdo apdorojimo technologijas.

1.1. Mokslinių straipsnių apžvalga

Straipsnių apžvalgos poskyryje aprašomi mokslinėje bendruomenėje naudojami metodai, siekiant atpažinti ligotus pasėlius, pateikiami gauti rezultatai. Vienuose straipsniuose aprašomi tyrimai kurie atlikti būtent su ligų paveiktais kviečių pasėliais, kituose – pateikiami tyrimai su keliomis pasėlių rūšimis.

1.1.1. Ligotų pasėlių aptikimas, naudojant atraminio vektoriaus klasifikatorių

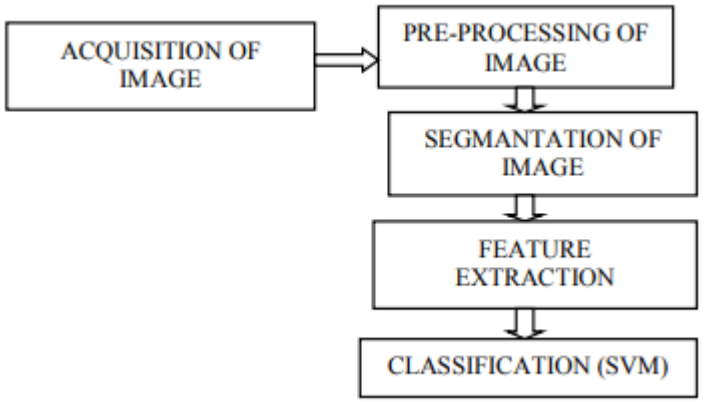
Vienas iš dažniausiai naudojamų klasifikatorių, siekiant atpažinti ligotus pasėlius, yra atraminio vektoriaus klasifikatorius *SVM* (angl. *Support Vector Machine*). Pirmajame analizuotame straipsnyje aprašomas sprendimas, kurio pagalba automatiškai aptinkami ligų pažeisti pasėliai ir klasifikuojamos ligos [1]. Sudaryti duomenų bazei buvo panaudota „Canon A3500“ vaizdo kamera. Siekiant padidinti duomenų bazę buvo paimta nuotraukų iš interneto. Tyrimo procesas yra suskirstytas į keturis etapus:

1. nuotraukos išgavimas;
2. išankstinis vaizdo apdorojimas;
3. vaizdo segmentavimas;
4. požymių išgavimas pagal spalvą, dydį ir struktūrą.

Siekiant išvengti nepageidaujamų trigdžių vaizde, prieš atliekant segmentavimą yra panaudojamas 3x3 dydžio *medianos filtras*. Kitas veiksmas – segmentavimas. Jis padeda suskirstyti vaizdus į norimas kategorijas, todėl tam panaudotas *K- vidurkių klasterizuoto skaidymo metodas* (angl. *K-means*). Šis metodas tinka apdorojant didelius duomenų kiekius. Klasteris yra paruošiamas tokiu būdu, kad to paties objekto pikseliai būtų kuo arčiau vienas kito, o pikseliai iš skirtingų objektų būtų nutolę vienas nuo kito. Po segmentavimo yra atliekamas požymių išgavimas. Tekstūros savybės yra išplėčiamos naudojant *GLCM* (angl. *Gray Level co- occurance matrix*) metodą. Jis padeda išmatuoti pilkų dalių pasiskirstymą, remiantis erdviniu pikselių santykiu, naudojant nurodytą atstumą ir kryptis vaizde. Spalvų požymių išgavimui panaudoti du metodai: vienas iš jų yra spalvų histograma, o kitas – spalvų momentai. Klasifikavimo uždaviniui spręsti panaudoti du variantai: dirbtinis neuroninis tinklas ir atraminio vektoriaus klasifikatorius.

Atlikus klasifikavimo tyrimą abiem metodais, gauti tokie rezultatai: naudojant neuroninį tinklą išgaunamas 80,21% tikslumas, o atraminio vektoriaus klasifikatorių- 89,23%.

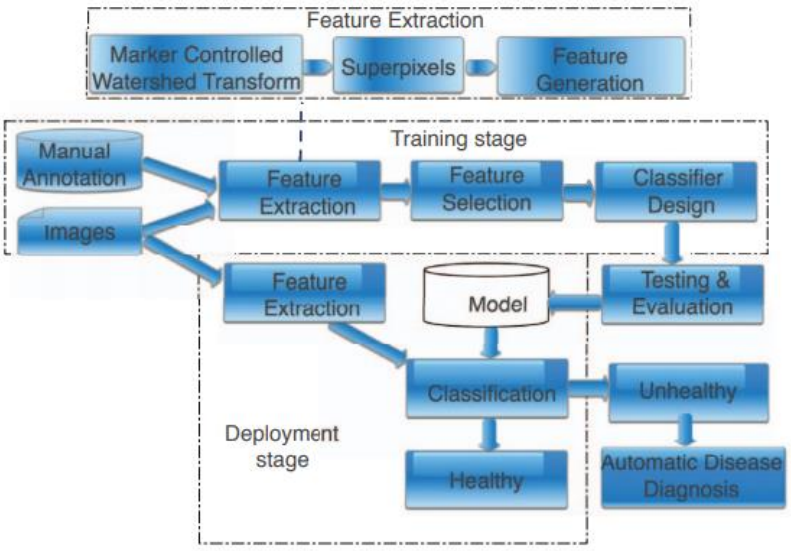
Toliau pateikiamas kitas straipsnis, kuriame aprašomas tyrimų metodas, paremtas SVM klasifikatoriaus taikymu [2]. Pirmiausiai yra išgaunamas vaizdas, po to atliekamas išansktinis jo apdorojimas, segmentavimas, savybių išgavimas. Paskutinis proceso etapas yra klasifikavimas, kurio metu naudojamas atraminio vektoriaus klasifikatorius. Toliau esančiame paveiksle yra pateikiamas metodo algoritmas (1.1 pav.).



1.1 pav. Metodo veikimo algoritmas [2]

Šios sistemos trūkumas yra tas, kad nuotraukoje esančio kviečio lapo fonas turi būti juodas, todėl realiomis aplinkos sąlygomis sistemos veikimas gali tapti komplikuotas.

Dar viename straipsnyje, paremtame SVM klasifikatoriaus naudojimu, aprašomas metodas remiasi šiais etapais: požymių išgavimu, jų parinkimu ir klasifikavimu [7]. Požymių išgavimo metu yra naudojamas *vandenskyros segmentacijos* (angl. *watershed segmentation*) algoritmas siekiant atskirti lapą nuo fono. Po to kai išgaunamas ligos pažeistas lapas, požymiams atskirti panaudojamas *superpikselių* metodas, kuris sugrūpuoja vaizdo taškus į reikšmingas sritis (taškų grupes kurios turi panašias charaktersitikas). Požymių parinkimo metu yra išskiriamos keturios grupės: tekstūriniai požymiai, gradientiniai požymiai, *Gaboro* filtro požymiai ir biologiniai požymiai. Klasifikavimas atliekamas naudojant atraminio vektoriaus klasifikatorių, o tam, kad būtų galima palyginti rezultatus, buvo panaudotas ir dirbtinis neuroninis tinklas. Bendras sistemos algoritmas pateiktas toliau esančiame paveiksle (1.3 pav.).



1.3 pav. Bendras sistemos modelis

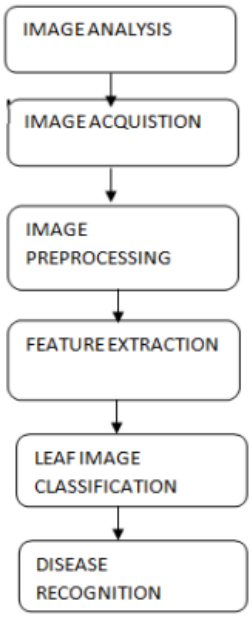
Atlikus bandymus, naudojant atraminio vektoriaus klasifikatorių gautas 70% *septoriozės* ir 95% *geltonųjų rudžių* (angl. *yellow rust*) ligų aptikimo tikslumas. Išbandžius neuroninį tinklą gautas 72% ir 53% tikslumas.

Tyrimo metu naudota įranga – Intel Core -i7-3630 QM procesorius ir 8GB fizinė bei 16GB virtuali atmintis.

Ketvirtame nagrinėjamame straipsnyje pateiktas metodas yra paremtas šiais pagrindiniais etapais [11]:

- vaizdo analizės atveju yra iš anksto apdorojamas įvesties vaizdas ir pagal duomenų rinkinį išgaunamos ypatybės. Po to yra panaudojamas klasifikatorius pagal konkretų duomenų rinkinį.
- vaizdo išgavimo metu jis yra konvetuojamas į norimą išvesties formatą. Pirmiausiai yra užfiksuojamas analoginis vaizdas ir po to yra konvetuojamas į skaitmeninį – tolimesniam apdorojimui.
- išankstinio apdorojimo metu yra padidinamas vaizdo kontrastingumas, sumažinama vaizdo rezoliucija, *RGB* spalvų gama pakeičiama į pilką.
- požymių išgavimo atveju išskiriamos vaizdo ypatybės, tokios kaip splavos ir kontūrai.
- atliekant lapų vaizdų klasifikavimą, naudojami klasifikatoriai, kurie paremti *Bayeso* teorema ir atraminio vektoriaus metodu.

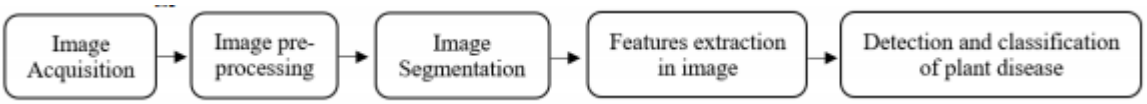
Metodo blokinė diagrama pateikta 1.4 paveiksle.



1.4 pav. Metodo blokinė diagrama [11]

Pirmiausiai vaizde užfiksuoti lapai suskirstomi į sveikus ir pažeistus ligos. Ligos nepažeistų lapų spalvos pasiskirsto tolygiai, tačiau pažeistų lapų spalvų pasisikrstymas nėra vienodas. Taip yra todėl, kad ligotų lapų pikselių vertės skiriasi nuo įprastų lapų. Vaizdo kokybė pagerinama pritaikius *vidurkinimo* filtrą. Tolimesnis žingsnis – vaizdo segmentavimas, pasitelkus *Otso* slenkstinės vertės algoritmą. Po to kai vaizdo ypatybės yra išryškintos, reikia atpažinti ligą naudojantis duomenų bazės vaizdais. Šiuo atveju naudojamas *atbulinės sklaidos tinklas* (angl. *Back Propagation Network*), o tikslumas yra padidinamas naudojant skirtingas vaizdų apdorojimo technologijas, tokias kaip vaizdo analizė, vaizdo išgavimas, išankstinis apdorojimas ir klasifikavimas. Naudojant šį metodą yra pasiektas 92% tikslumas.

Tolimesniame nagrinėtame darbe pateikiamas lapų ligos nustatymo tyrimas, taip pat paremtas SVM metodu [12]. Segmentavimui panaudotas *k – vidurkių klasterizavimo algoritmas*, kuris skirtas suskirstyti augalo lapus į skirtingas grupes. Bendras metodo modelis pateiktas 1.5 paveiksle.



1.5 pav. Sistemos modelis [12]

Straipsnyje pateiktas metodas suskirtytas į penkias dalis: vaizdo išgavimo, pirminio apdorojimo, segmentavimo, ypatybių išgavimo ir klasifikavimo.

Pirmiausiai vaizdo išgavimo metu parenkamas augalas, paveiktas ligos ir jo nuotrauka patalpinama sistemoje. Siekiant vaizdą paversti lengviau analizuojamu, naudojamas segmentavimas, kurio metu vaizdas padalijamas į kelis skyrius, kurie gali būti apibrėžti kaip „superpikseliai“. Toliau naudojamas *mažo kontrasto* metodas, siekiant pikselių vertes sukoncentruoti siaurame diapazone. Tyrimui atlikti panaudotas atraminio vektoriaus metodas.

1.1.2. Konvoliuciniais neuroniniais tinklais pagrįstas ligų paveiktų pasėlių atpažinimas

Pirmasis nagrinėtas straipsnis, kuriame aprašomas metodas paremtas *CNN* taikymu yra suskirstytas į šiuos etapus [3]:

- duomenų rinkinio sudarymas;
- išanksktinis vaizdo apdorojimas;
- duomenų didinimas;
- modelio mokymas naudojant skirtingas konvoliucinio neuroninio tinklo struktūras;
- skaičiavimų ir rezultatų gavimas.

Panaudotos skirtingos konvoliucinių neuroninių tinklų struktūros siekiant išgauti kuo geresnį rezultatą: *VGG16, VGG19, AlexNet, ResNet – 34, Resnet – 101, ResNet – 50* ir *ResNet – 18*. Treniravimui panaudota 600 nuotraukų, o testaimui – 150. Kiekvienos nuotraukos dydis buvo pakeistas į 224x224 rezoliuciją. Tinklo treniravimo metu buvo atliekamas duomenų didinimas sukiojant vaizdą, o naudotas *batch* dydis yra 64 vnt.

Tiksliausi rezultatai (98,6% tikslumas) gauti naudojant *ResNet – 101* struktūrą. Prasčiausi rezultatai gauti bandymus atliekant su *VGG-16* struktūra. Šiuo atveju gautas 92,6% tikslumas.

Kitame straipsyje aprašomas kviečių rūdžių ligos atpažinimo metodas naudojant konvoliucinius neuroninius tinklus, kurio duomenų bazę sudaro 876 nuotraukos: 143 sveikų kviečių, 359 su ligos pažeistais lapais ir 377 su ligos pažeistais stiebais [8]. Didžioji dalis nuotraukų buvo surinktos Etiopijos ir Tanzanijos šalyse. Likusi dalis buvo paimta iš viešai prienamų „Google“ vaizdų duomenų bazių. Nuotraukos išgautos naudojant įvairius fotografavimo įrenginius, todėl turi skirtingą kokybę ir rezoliuciją. Duomenų bazė atsitiktine tvarka buvo padalinta į treniravimo ir testavimo duomenis santykiu 80:20.

Kuriant modelį, buvo naudojama *Densenet201* konvoliucinio neuroninio tinklo architektūra. Konvoliucijos metu naudojamas *Kernel* filtras (angl. *Kernel matrix*) kuris taikomas apdorotam

sluoknsiui taip suformuojant aktyvacijos signalą kito sluoksnio neuronui. Tokiu būdu yra gaunami ypatybių sąrašai. Toliau yra sumažinami šių sąrašų matmenys, naudojant sutelkimo funkciją.

Tinklo treniravimo metu buvo panaudotos šios funkcijos: atsitiktinis pasukimas, atsitiktinis šviesos kontrastas, atspalvio sodrumas, suliejimo funkcija. Kiekviena iš šių transformacijų buvo panaudota su 0,3 tikimybe. Pradinis mokymosi greitis buvo parinktas 0,001. Modelio treniravimo trukmė – 25 epochos. Po 8 – os ir 20 – os epochų mokymosi greitis buvo mažinamas 10 kartų. Atlikus testavimo tyrimą buvo gautas 90% tikslumas.

Toliau pateikiamas dar vienas kviečių ligų aptikimo metodas, paremtas gilaus mokymosi sistema [9]. Duomenų bazės suskirstytos į keturias ligų grupes: stiebo rūdžių, geltonųjų rūdžių, miltigės ir sveikų pasėlių. Kiekvieną kategoriją sudaro 2207 nuotraukos. Vaizdai buvo suskirstyti į treniravimo ir testavimo duomenis santykiu 80:20. Kiekvienos nuotraukos dydis buvo pakeistas į vienodą – 227x227 pikselių dydžio formatą. Funkcijos išgautos per konvoliucinius sluoksnius pritaikant *ReLU* (angl. *Rectified Linear Unit*) aktyvavimo bei *maksimalaus sutelkimo* (angl. *Max- Pooling*) funkcijas.

Klasifikatoriaus treniravimui panaudotas konvoliucinis neuroninis tinklas pasitelkiant *AlexNet* architektūrą. Tokio tinklo vienas iš pagrindinių privalumų – automatiškai išskirti bruožus tiesiogiai apdorojant pirminius vaizdus. Visą procesą galima suskirstyti į keturis etapus:

- išankstinis mokymas;
- parametrų derinimas;
- treniravimas;
- ligų klasifikavimas.

Treniruojant modelį, panaudotas *batch* dydis yra 16, o mokymosi greitis – 0.0001. Naudojant šį modelį buvo išgautas 84,54% tikslumas.

Dar viename straipsnyje aprašomas augalų ligų atpažinimo metodas, naudojant gilius konvoliucinius neuroninius tinklus [13]. Duomenų bazė sudaryta iš atsisiųstų nuotraukų, kurios suskirstytos į 15 klasių. Siekiant atskirti sveikus lapus nuo ligos pažeistų, pridėta papildoma klasė, kurią sudaro tik sveikų augalų lapų vaizdai. Tinklo treniravimui sukurta 30880 vaizdų, o testavimui – 2589. Kadangi vaizdai yra skirtingų formatų, tam, kad išgauti tikslesnę informaciją, atliekamas išankstinis apdorojimas.

Pagrindinis duomenų kiekio didinimo tikslas yra padidinti duomenų rinkinį ir sukurti nedidelius vaizdų iškraipymus kurie padeda sumažinti persimokymą tinklo treniravimo metu. Vaizdo papildymo metu įtraukiamas vienas iš kelių transformacijos metodų bei paprastas vaizdo pasukimas.

Darbe panaudota *CaffeNet* struktūra, kuri buvo modifikuota siekiant pritaikyti penkiolikai klasių. Taip pat panaudota *ReLU* aktyvavimo funkcija. Tinklas su šia funkcija apsimoko žymiai greičiau. Tikslumo testui pritaikyta *kryžminio patvirtinimo* (angl. *cross validation*) funkcija kuri pakartojama kas tūkstantį mokymo iteracijų. Atliekant tyrimą, pasiektas bendras 96.3% sistemos tikslumas.

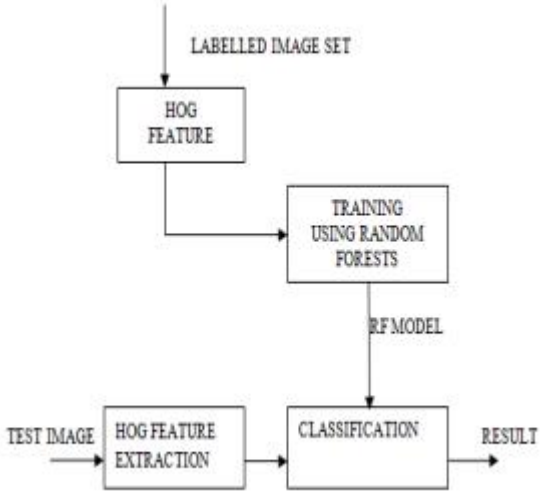
1.1.3. Kitais metodais paremtas ligų pažeistų pasėlių atpažinimas

Šiame nagrinėjamame straipsnyje yra aprašomas ligų pažeistų augalų atpažinimas naudojant *atsitiktinio miško* (angl. *Random forest*) metodą [4]. Siekiant išsiaiškinti ar augalo lapai yra sveiki, ar pažeisti, yra naudojami šie tyrimo žingniai: pirminis duomenų apdorojimas, ypatybių išgavimas,

klasifikatoriaus mokymas ir klasifikavimas. Pirminio duomenų apdorojimo metu vaizdai yra atkuriami į vienodą, sumažintą dydį. Toliau yra išgaunamos apdoroto vaizdo savybės naudojant orientuotų gradientų histogramą *HOG* (angl. *Histogram of oriented gradients*). Naudojant *HOG* funkciją, išskiriamos trys dalys:

1. *Hu* momentai, kurie išduoda svarbiausias vaizdo taškų charakterstikas. Momentai padeda apibūdinti tam tikro lapo kontūrą. Skaičiuojant momentus, *RGB* spalvų gama yra konveruotuojama į pilką;
2. *Haraliko* tekstūra. Paprastai sveikų ir ligotų lapų tekstūra yra skirtinga, todėl siekiant ją išgauti, naudojama ši funkcija. Ji paremta gretimybių matrica, kurioje saugoma vaizdo taškų padėtis dvimatėje koordinačių ašyje. Norint apskaičiuoti tekstūrą, reikalinga *RGB* vaizdą konvertuoti į pilkos spalvos vaizdą.
3. spalvų histograma reprezentuoja vaizdo spalvas. Pirmiausiai *RGB* vaizdas paverčiamas į *HSV* (*Hue, Saturation, Value*) spalvų spektrą, o po to yra apskaičiuojama splavų histograma.

Algoritmas realizuojamas naudojant *atsitiktinio miško* klasifikatorių. Šio modelio struktūra pateikiama 1.2 paveiksle.



1.2 pav. Naudojamas sistemos algoritmas[4]

Algoritmas buvo gretinamas su kitais mašininio mokymo modeliais, siekiant išgauti didesnį tikslumą. Naudojant *atsitiktinio miško* klasifikatorių, modelis buvo apmokytas naudojant 160 atvaizdų. Modelis gali klasifikuoti apie 70% tikslumu. Rezultatai galėtų būti pagerinti naudojant daugiau vaizdų mokymo procesui, taip pat kitas funkcijas, tokias kaip *SIFT* (angl. *Scale Invariant Feature Transform*) ar *SURF* (angl. *Speeded – Up Robust Features*).

Kitame straipsnyje aprašomas metodas remiasi šiais etapais [5]:

- *RGB* vaizdo išgavimas;
- išansktinis apdorojimas, kurio metu yra padidinama vaizdo kokybė atliekant vaizdo iškirpimą, lyginimą ir patobulinimą;
- *K- vidurkių klasterizuoto skaidymo* metodo panaudojimas. Vaizdas yra suskaidomas į skirtingas grupes;

- žalių pikselių, esančių nuotraukoje, maskavimas. Šie pikseliai užmaskuojami ir paliekamos tik užkrėstos lapo vietos. Šio veiksmo pagalba išgaunamas didesnis tikslumas. Toks procesas dar vadinamas segmentavimu;
- užkrėstos lapo vietos *RGB* vaizdas paverčiamas į *HSI* (angl. *Hue, Saturation, Intensity*);
- sugeneruojama *SGDM* (angl. *Spacial Gray- level Dependence*) matrica;
- panaudojama *GLCM* (angl. *Gray- level Co-occurence Matrix*) funkcija požymių skaičiavimui;
- apskaičiuojama tekstūros statistika ir klasifikavimui pritaikomas tikimybinis neuroninis tinklas.

Naudojant šią sistemą gautas 96% tikslumas. Tolimesnis šio projekto darbas – padidinti duomenų bazę, siekiant išgauti tikslesnius rezultatus. Kaip klasifikatorių planuojama panaudoti *fuzzy* logiką ir palyginti rezultatus, gautus abiem atvejais.

Dar viename darbe, kuriame sprendžiamas ligotų pasėlių atpažinimo uždavinys, aprašoma sistema, kurią sudaro ARM9 procesorius su įterptine Linux operacinės sistemos platforma, o programa sukurta naudojantis integruota Qt aplinka [6].

Metodas paremtas tuo, kad pirmiausiai užfiksuotas ligoto lapo vaizdas iš *RGB* spalvinės gamos yra transformuojamas į vieno kanalo pilką vaizdinį. Remiantis *Sobelio* metodu (naudojamas 3x3 matricos filtras), atliekamas vertikalių kraštų aptikimas ir taip eliminuojamas nereikalingas fonas, paliekant tik ligoto lapo vietą. Po to yra išfiltruojamos nereikalingos vietos naudojant *srauto užpildymo* (angl. *flood filling*) algoritmą. Galiausiai yra apskaičiuojamas ligotų vietų ploto santykis.

Naudojant šią sistemą, gautas 96,2% tikslumas.

Paskutiniajame nagrinėtame moksliniame tiriamojo pobūdžio darbe aprašomi vaizdų atpažinimo metodai, kurių pagalba yra išskiriamos ligotos lapo vietos [10].

Ligos pažeisto kviečio lapo savybių suradimo procesas yra suskaidytas į keturis etapus:

1. aiškiausiai vaizde matomos vietos išskyrimas. Tikslas – panaikinti vaizde esantį foną, jo spalvą pakeičiant juoda. Šiam procesui naudojamas *GrabCut* metodas;
2. kontūrų radimas. Siūlomas būdas – *Cany Edge Detection*, o rekomenduojamos slenkstinės vertės lapo tekstūrų aptikimui yra šios – 250 apatinė ir 300 – viršutinė;
3. spalvų filtravimas. Šiam procesui siūloma naudoti *inRange* metodą, kuris aprašytas *OpenCV 2* biliotekoje;
4. dviejų ar daugiau metodų sujungimas. Siekiant išgauti kuo tikslensius rezultatus, gali būti naudojamas metodų sujungimas, todėl šiame darbe pateikiamas kontūrų radimo ir spalvų filtravimo funkcijų panaudojimas.

Tokio tipo sistemos trūkumas yra tas, kad lapas turi būti vaizdo centre, o fone neturi būti kitų lapų, todėl realiomis sąlygomis tokios sistemos veikimas taptų komplikuotas.

Atlikus mokslinių straipsnių apžvalgą, pastebėta, kad ligų pažeistų pasėlių atpažinimo tyrimuose dažnai naudojami konvoliuciniais neuroniniais tinklais paremti klasifikatoriai. Taip pat, sprendžiant tokio tipo uždavinius, yra naudojamas algoritmas, kuris susideda iš tokių etapų, kaip vaizdo analizė, vaizdo išgavimas, išankstinis apdorojimas, požymių išgavimas.

1.2. Įmonių apžvalga

Šioje dalyje pateikiamos įmonės, kurios specializuojasi ligų pažeistų pasėlių atpažinimo srityje.

1.2.1. *BitRefine Group*

BitRefine Group – tai kompanija, įkurta 2013 m. Honkonge, kuri specializuojasi gilaus mokymo tyrimuose [14]. Ji savo veiklą vykdo Azijoje, Europoje bei JAV. Įmonė dirba duomenų analizės, vaizdinės informacijos apdorojimo, žemdirbystės srityse. Tyrimai daugiausia skirti šiems augalams: kukurūzams, medvilnei, kavai, vaismedžiams. Kompanijos kuriama produkcija, kuri skirta pasėlių ligų atpažinimui, paremta procesu, kada ant transporto priemonės sumontuotos kameros pagalba yra išgaunamas vaizdo srautas ir naudojant mašininio mokymosi metodus aptinkami ligų pažeisti augalai.

1.2.2. *Taranis*

Taranis įmonė buvo įkurta 2015 m. [15]. Jos kuriama įranga naudoja kompiuterinę regą, duomenų mokslą bei gilaus mokymo algoritmus tam, kad būtų galima išgauti išsamias įžvalgas apie derlių. Kompanija naudoja AI^2 technologiją, kuri leidžia išgauti 0.3mm per pikselį rezoliuciją.

2. Metodinė dalis

Metodinėje dalyje yra pateikiamas tyrimų objektas, aprašoma tyrimams atlikti naudota kompiuterinė bei programinė įranga. Tiriamojo pobūdžio darbui atlikti buvo panaudota *TensorFlow* biblioteka, kuria naudojantis įgyvendinti tyrimai su *Faster – RCNN* ir *R – FCN* architektūromis. Atliekant tyrimus su *Faster – RCNN* architektūra, naudota *Inception V_2* struktūra. *R – FCN* architektūrai pasirinkta *ResNet – 101* struktūra. Šios konvoliucinių neuroninių tinklų struktūros buvo paimtos iš programinės įrangos dalinimosi sveitainės „GitHub“ [28]. Metodinėje dalyje surašytos naudotos duomenų bazės, aprašyta *TensorFlow* biblioteka, jos veikimo principai. Taip pat aprašomi *Inception_V2* ir *ResNet – 101* struktūrų veikimo principai, pagrindinės jų charakteristikos. Modelių efektyvumui įvertinti nuspręsta naudoti *klaidų matricas,* kuriomis remiantis galima apskaičiuoti *False / Negative* koeficienus*, Accuracy, F- Score* reikšmes. Pateikiamos formulės, kuriomis remiantis yra gaunamos šių reikšmių vertės.

2.1. Tyrimų objektas

Atlikus literatūros analizę, tiriamąjį darbą buvo nuspręsta daryti remiantis konvoliuciniais neuroniniais tinklais pagrįstais metodais. Tiriamojo pobūdžio darbo įgyvendinimui buvo pasirinktas ligotų kviečių pasėlių atpažinimo uždavinys. Tokio tipo užduočiai atlikti yra reikalinga duomenų bazė, kuri buvo paimta iš mokslinės bendruomenės tinklapio „kaggle“. Jame pateikti duomenys, kuriais pasidalino Ibrahim Olagoke ir kuriuos sudaro treniravimo ir testavimo grupės [16]. Testavimo duomenų grupę sudaro 610 nuotraukų, o treniravimo yra suskirstyta į tris dalis:

1. sveiki kviečiai (142 nuotraukos);
2. rūdžių ligos paveikti lapai (338 nuotraukos);
3. rūdžių ligos paveikti stiebai (376 nuotraukos).

Tam, kad būtų galima gauti kuo tikslesnius rezultatus, buvo atrinktos tik tos nuotraukos, kuriose aiškiausiai matomi tyrimo objektai. Ruošiant duomenis, buvo sudaryta tokia duomenų bazė:

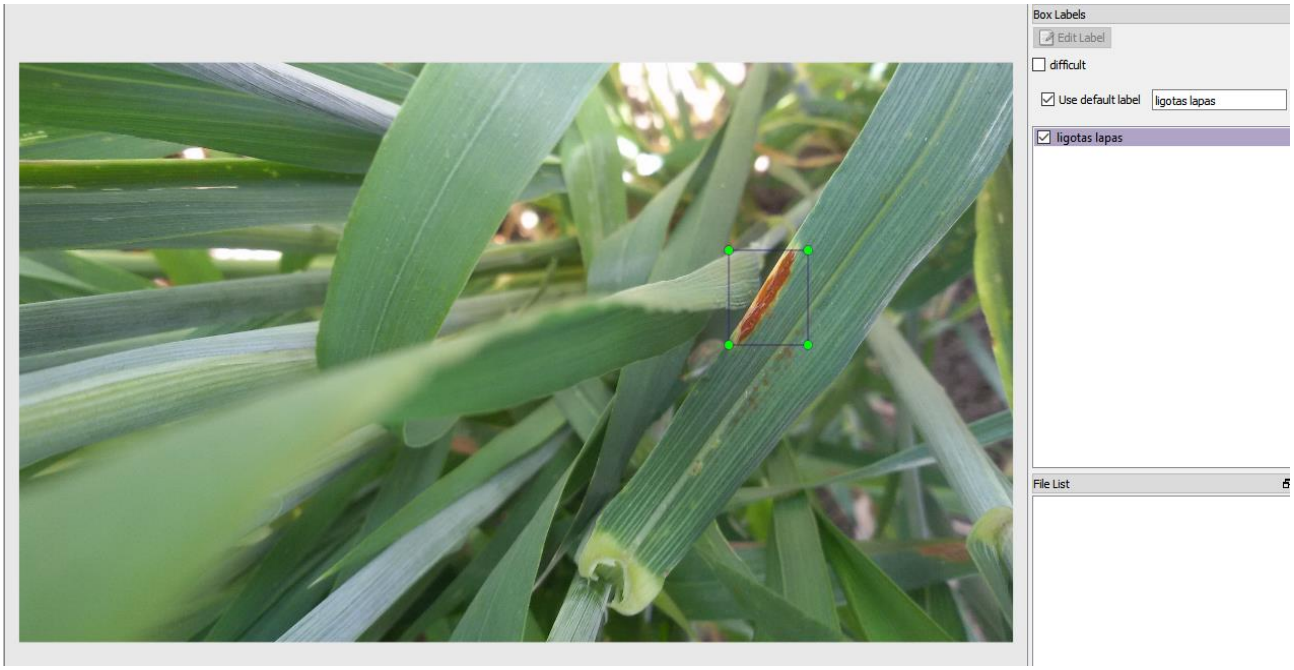
1. treniravimo duomenys (612 nuotraukų);
2. patvirtinimo duomenys (350 nuotraukų);
3. testavimo duomenys (55 nuotraukos).

Toliau esančiame paveiksle pateikiami kviečių vaizdai, turintys rūdžių ligos paveiktų lapų ir stiebų (2.1 pav.).

2.1 pav. Rūdžių ligos paveikti stiebo ir lapo vaizdai

Ruošiant duomenis mokymui buvo naudojama „LabelImg“ programa, kurios pagalba nuotraukose rankiniu būdu yra iškerpamos reikalingos sritys (šio tyrimo atveju iškerpamos dalys, kuriose yra ligotų kviečių lapų arba stiebų). Ši programa failus išsaugo .xml formatu, kurie vėliau yra panaudojami neuroninio tinklo mokymui. Rudžių ligos paveikti kviečių pasėliai yra aptinkami pagal du kriterijus – vaizde ieškoma ligotų stiebų arba ligotų lapų.

Toliau esančiame paveiklse pateikiamas reikalingos informacijos iš nuotraukos išgavimo procesas (2.2 pav.)



2.2 pav. Duomenų ruošimo procesas, naudojant „LabelImg“ programą

2.2. Kompiuterinė įranga

Atliekant ligų pažeistų kviečių atpažinimo tyrimą buvo naudojamas *Acer* gamintojo kompiuteris, kurio modelis – *VN7 – 571G – 57C8*. Šis kompiuteris turi penktos kartos *Intel Core i5 – 5200U* procesorių ir *NVIDIA GeForce 940M* vaizdo plokštę. Žemiau esančiose lentelėse pateikiamos pagrindinio procesoriaus (2.1 lentelė) ir grafinio procesoriaus (2.2 lentelė) specifikacijos.

2.1 lentelė. Pagrindinio procesoriaus specifikacijos [17]

Branduolių skaičius	2
Taktinis dažnis	2,2 GHz
Maksimalus taktinis dažnis	2,7 GHz
Maksimalus atminties dydis	16 GB
Šiluminė projektavimo galia	15 W

2.2 lentelė. Grafinio procesoriaus specifikacijos [18]

Architektūra	Maxwell
Atminties tipas	DDR3
Atminties takrinis dažnis	2000 MHz
Maksimalus atminties dydis	4096 MB
Branduolio taktinis dažnis	1072 MHz

2.3. *Anaconda* programinis įrankis

Darbui su *TensorFlow* biblioteka atlikti buvo panaudotas *Anaconda* programinis įrankis, kurio pagalba sukuriama *Phyton* programavimo kalbos aplinka. Ji suteikia galimybę naudotis *Phyton* bibliotekomis išvengiant programų versijų skirtumų. *Anaconda* įrankis leidžia dirbti *Windows* operacinės sistemos aplinkoje ir naudotis bibliotekomis, kuriomis įprastai būtų galima dirbti tik *Linux* sistemoje. Šiuo atveju buvo parinkta 4.8.3 aplinkos versija.

TensorFlow biblioteka, treniruojant konvoliucinius neuroninius tinklus, suteikia galimybę naudotis tiek grafiniu, tiek ir pagrindiniu kompiuterio procesoriumi. Tam reikia įrašyti *CUDA* lygiagrečios skaičiavimo platformos ir programavimo sąsajos modelį bei *cuDNN* biblioteką.

2.4. *TensorFlow* biblioteka

Ligotų kviečių pasėlių atpažinimo tyrimui atlikti buvo pasirinkta *TensorFlow* atvirojo kodo biblioteka, kuri yra sukurta „Google“ kompanijos ir skirta gilaus mokymosi programoms [19]. Sistema taip pat palaiko ir tradicinį mašininį mokymąsi. Iš pradžių *TensorFlow* buvo sukurtas didelės apimties skaičiavimams vykdyti, o ne nesiekiant pritaikymo gilaus mokymo aplikacijose, tačiau sistema pasirodė naudinga ir gilaus mokymosi plėtojimui, todėl „Google“ padarė ją prienamą viešai.

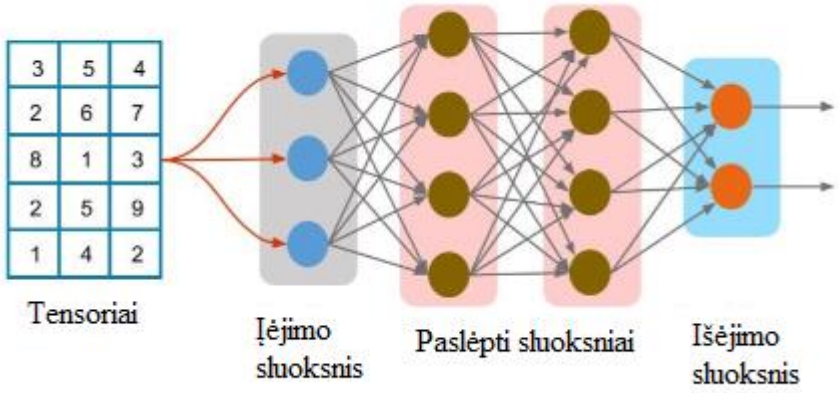
TensorFlow veikimas pagrįstas tuo, kad yra priimami aukštesnių matmenų, vadinamų *tensoriais* (angl. *tensors*) daugiamačių masyvų duomenys. Daugiamačiai masyvai yra patogus įrankis tvarkant didelius duomenų kiekius. *TensorFlow* sistema veikia remiantis duomenų srautų grafais, turinčiais mazgus ir kraštus. Kadangi mechanizmas yra vykdomas grafų pagalba, norint vykdyti mokymo programą galima pasitelkti grafinį procesorių *GPU* (angl. *graphichs processing unit*).

Prieš kuriant bibliotekas, programų rašymas mašininiam ir giliam mokymuisi buvo daug sudėtingesnis procesas. *TesorFlow* biblioteka suteikia aukšto lygio aplikacijų programavimo sąsają *API* (angl. *application programming interface*), todėl norint paruošti neuroninį tinklą, sukonfigūruoti neuroną ar jį užprogramuoti, komplikuotas programų rašymas tampa nebereikalingas.

Gilaus mokymosi programos yra labai sudėtingos, o mokymosi procesas reikalauja daug skaičiavimų. Proceso vykdymas trunka ilgai dėl didelio duomenų kiekio, o jis apima matematinius skaičiavimus, matricų dauginimus ir panašiai, todėl atliekant šias operacijas pasitelkus tik pagrindinį procesorių *CPU* (angl. *central processing unit*), tai užtrunka ilgiau. Vienas iš pagrindinių *TensorFlow* aplikacijos naudojimo privalumų ir yra, kad ji palaiko tiek pagrindinio procesoriaus, tiek ir grafinio procesoriaus naudojimo galimybę, todėl darbe buvo panaudotas būdas pasitelkiant abu procesorius. *TensorFlow* aplikacija taip pat turi greitesnį kompiliavimo laiką, lyginant su kitomis gilaus mokymosi bibliotekomis, tokiomis kaip *Keras* ar *Torch*.

2.4.1. Tensorių aprašymas

Tensorius, kuris naudojamas *TensorFlow* aplikacijoje, yra aukštesnių matmenų vektorių ir matricų apibendrinimas, o duomenų masyvai su skirtingais matmenimis ir lygiais, kurie tiekiami kaip įvesties duomenys į neuroninį tinklą, vadinami *tensoriais*. Giliam mokymuisi, ypač mokymo procese, reikia turėti daug duomenų. Kai duomenys yra patalpinami į *tensorius* ir įvedami į neuroninį tinklą, gaunama išvestis yra tokia, kaip parodyta toliau esančiame paveiksle (2.3 pav.).



2.3 pav. Tensoriaus įvedimas į neuroninį tinklą

Tensorius apibūdina matmenys ir lygiai. Matmenys nusako iš kelių dimensijų sudaryti *tensoriai*. Jie gali būti vienmačiai, dvimačiai ir t.t. Lygiai – tai dimensijų skaičius, naudojamų pateikti duomenims:

- 0 lygis – tai vienas elementas, arba kitaip - skaliaras;
- 1 lygis – vienos dimensijos vektorius;
- 2 lygis – tai dviejų dimensijų vektorius, arba matrica;
- 3 lygis – kelių dimensijų vektorius.

2.4.2. Duomenų srauto grafas

Kai turime *tensoriuose* saugomus duomenis, yra atliekami skaičiavimai, kurie vyksta grafų pavidalu. Skirtingai negu tradiciniame programavime, kai parašytas kodas yra vykdomas nuosekliai, naudojant

TensorFlow aplikaciją yra kuriami duomenų srautai, kurie sudaryti iš mazgų. Tada grafai yra vykdomi sesijos forma.

2.5. Konvoliuciniai neuroniniai tinklai

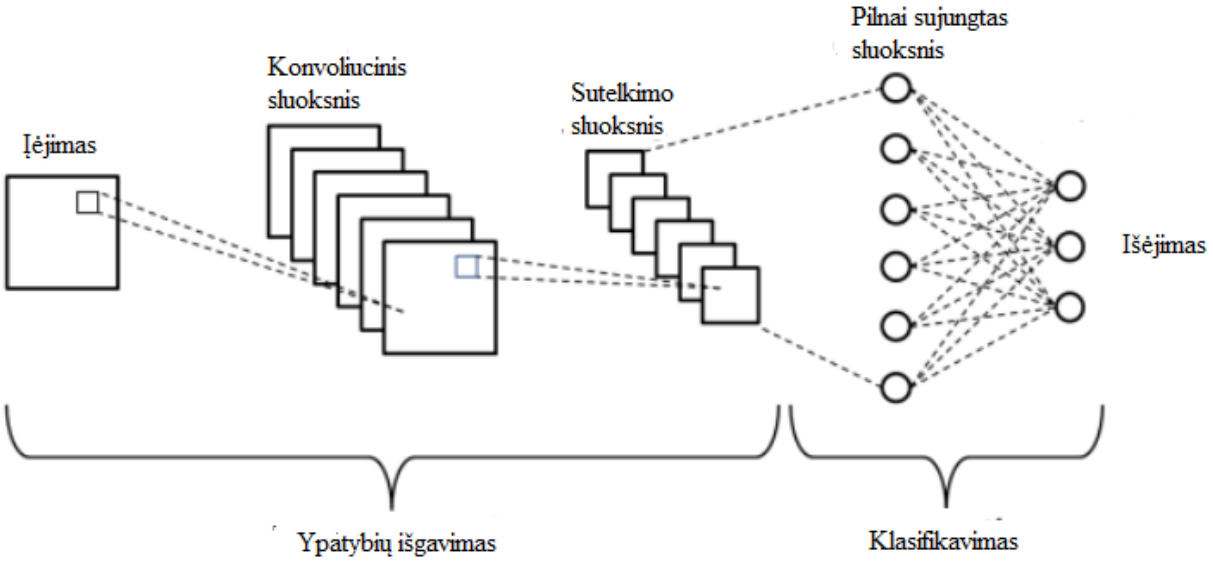
Konvoliucinis neuroninis tinklas *CNN* (angl. *convolutional neural network*) yra gilaus mokymosi algoritmas, kuris įėjime gaudamas vaizdą, mokymosi proceso metu pritaiko mokymosi svorius ir poslinkius (angl. *bias*) ir tokiu būdu sprendžia kalsifikavimo uždavinius [20]. Naudojant šio tipo tinklus, reikalingas išankstinio apdorojimo informacijos kiekis yra mažesnis lyginant su kitais klasifikavimo algoritmais.

Konvoliucinio neuroninio tinklo vaidmuo yra sumažinti vaizdus į lengviau apdorojamą formą, neprarandant ypatybių, kurios yra būtinos siekiant išgauti kuo tikslesnius rezultatus.

CNN veikimo principas paremtas tuo, kad yra naudojamas *Kernel* filtras (tam tikro dydžio matrica). Ši matrica po vieną žingsnį yra perstumiama per visą vaizdo plotį, po to paslenkama aukščio kryptimi ir procesas kartojamas tol, kol apdorojamas visas vaizdas. Konvoliucijos proceso tikslas yra iš vaizdo išgauti svarbias ypatybes, tokias kaip objektų kraštai.

Sutelkimo sluoksnis (angl. *pooling layer*), kuris yra panšus į konvoliucinį sluoksnį, atsakingas už erdvinio dydžio sumažinimą. Jį naudojant siekiama sumažinti resurus, reikalingus skaičiavimo operacijoms atlikti. Be to, tai naudinga norint išgauti vyraujačias savybes, taip išlaikant efektyvų modelio mokymą.

Dar vienas sluoksnis, naudojamas konvoliuciniuose neuroniniuose tinkluose, yra *pilnai sujungtas sluoksnis* (angl. *fully – connected layer*). Jis modeliui padeda išmokti netiesinius aukšo lygio ypatybių derinius, pateiktus konvoliucinio sluoksnio išvestyje. Konvoliucinio neuroninio tinklo struktūra yra pateikta 2.4 paveiksle.



2.4 pav. Konvoliucinio neuroninio tinklo struktūra [21]

2.6. Naudotos architektūros

2.6.1. *Faster R – CNN* architektūra

Viena iš dviejų architektūrų, panaudotų atliekant tiriamąjį darbą, yra *Faster R – CNN*, kuri sukurta 2015 m. [22]. Ji buvo išvystyta iš prieš tai sukurtų *R – CNN* (2014 m.) ir *Fast R – CNN* (2015 m.) architektūrų. *Faster R – CNN* sudaro tokios dalys: *RPN* (angl. *Region Proposal Network*), *RoI* (angl. *Region of Interest*) bei *R – CNN* (angl. *Region Based Convolutional Neural Network*). Naudojant šią architektūrą, pagrindinis tikslas yra iš vaizdo išgauti šias dalis:

- besiribojančių sričių sąrašą;
- kiekvienai sričiai priskirtą žymę;
- kiekvienos srities ir žymės tikimybę.

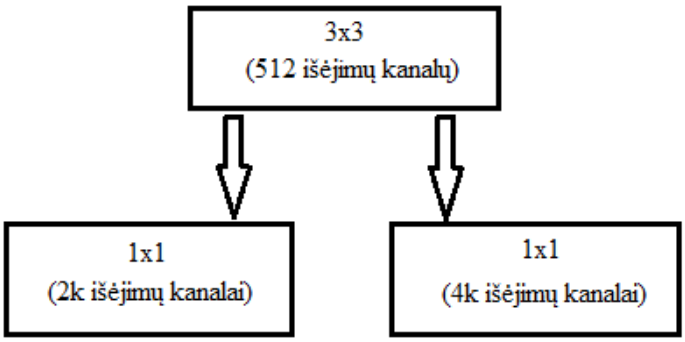
Įvesties vaizdai yra apibūdinami aukščio, pločio ir gylio *tensoriais* (daugiamačiais masyvais), kurie persiunčiami per iš anksto apmokytą konvoliucinį neuroninį tinklą iki tarpinio sluoksnio, sudarant *ypatybių sąrašą* (angl. *feature map*). Viena didžiausių problemų su kuria susiduriama sprendžiant vaizdų atpažinimo uždavinius naudojantis gilaus mokymo algoritmais, yra kintamo dydžio besiribojančių sričių *sąrašo* sukūrimas. Ši problema sprendžiama taikant *RPN* metodą, panaudojant fiksuoto dydžio besiribojančių sričių rinkininius, kurie tolygiai išdėstyti visame originaliame vaizde. Vietoj to, kad būtų siekiama surasti objektų padėtį, problema yra suskirstoma į dvi dalis:

- siekiama išsiaiškinti ar šiame fiksuoto dydžio besiribojančių sričių rinkinyje yra atitinkamas objektas;
- siekiama išsiaiškinti kaip būtų galima pritaikyti šį rinkinį, kad jis kuo geriau atitiktų objektą.

RPN metodo taikymo metu yra priimami fiksuoto dydžio besiribojančių sričių rinkininiai ir pateikiami tinkami variantai. Tai atliekama turint du skirtingus išėjimus kievienam rinkiniui:

- pirmasis išėjimas yra tikimybė, kad besiribojančių sričių rinkinys yra objektas.
- antroji išvestis yra besiribojančios srities regresija, kuri skirta koreguoti sritims, kad jos kuo tiksliau atitiktų spėjamą objektą.

RPN metodas yra įgyvendinamas konvoliuciniu būdu, naudojant bazinio tinklo grąžintą konvoliucinių *ypatybių sąrašą*. Pirmiausiai naudojamas konvoliucinis sluoksnis su 512 kanalų ir 3x3 dydžio filtru. Toliau, panaudojus 1x1 fitrą, gaunami du lygiagretūs konvoliuciniai sluoksniai. *RPN* architektūros įgyvendinimo schema pateikta 2.5 paveiksle, kur *k* raidė žymi besiribojančių sričių rinkinių skaičių.



2.5 pav. *RPN* architektūra

Įgyvendinus *RPN* etapą, lieka daugybė objektų, kuriems nėra priskirta jokia klasė, todėl besiribojančias sritis reikia suskirstyti į norimas kategorijas. Paprasčiausias būdas būtų iškirpti kiekvieną variantą ir perduoti per iš anksto apmokytą bazinį tinklą. Tada išgautas ypatybes būtų galima naudoti kaip įėjimus vaizdų klasifikatoriui. Tačiau pagrindinė problema yra ta, kad visų variantų skaičiavimas yra neefektyvus ir lėtas procesas.

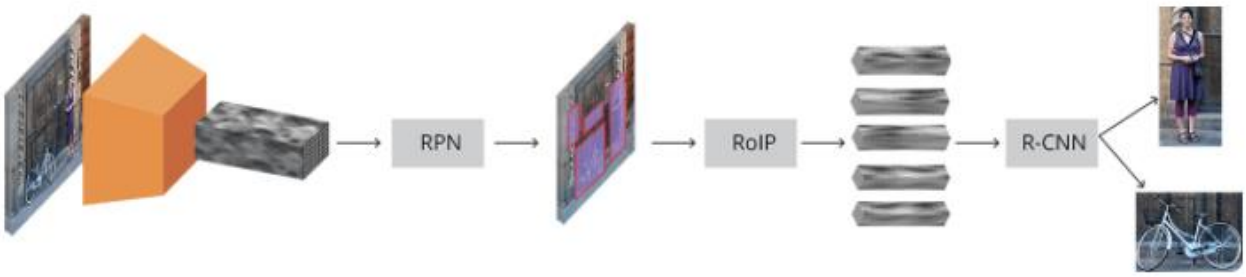
Faster R- CNN architektūros pagalba yra bandoma išspęsti šią problemą pakartotinai naudojant esamą *ypatybių sąrašą*. Naudojantis konvoliucinio tinklo išgautomis savybėmis ir besiribojančiomis sritimis su atitinkamais objektais, pritaikomas *RoI sutelkimo* metodas ir yra išskiriamos tos ypatybės, kurios atitiktų objektus naujame daugiamačiame masyve.

Galiausiai pritaikomas *R – CNN* modelis, kurį naudojant turinys suklasifikuojamas besiribojančiose srityse ir yra sugeneruojamos tų sričių koordinates. Pasiteliant *R – CNN* modelį yra imituojami paskutiniai konvoliucinio neuroninio tinklo etapai, kai panaudojamas *pilnai sujungtas sluoksnis*, siekiant išgauti kiekvienos galimos objektų klasės rezultatą.

R – CNN modeliui priskiriami du pagrindiniai tikslai:

- suskirstyti variantus į vieną iš klasių ir į foninę klasę (siekiant pašalinti netinkamus variantus);
- kuo tiksliau pritaikyti besiribojančią sritį pagal numatomą klasę.

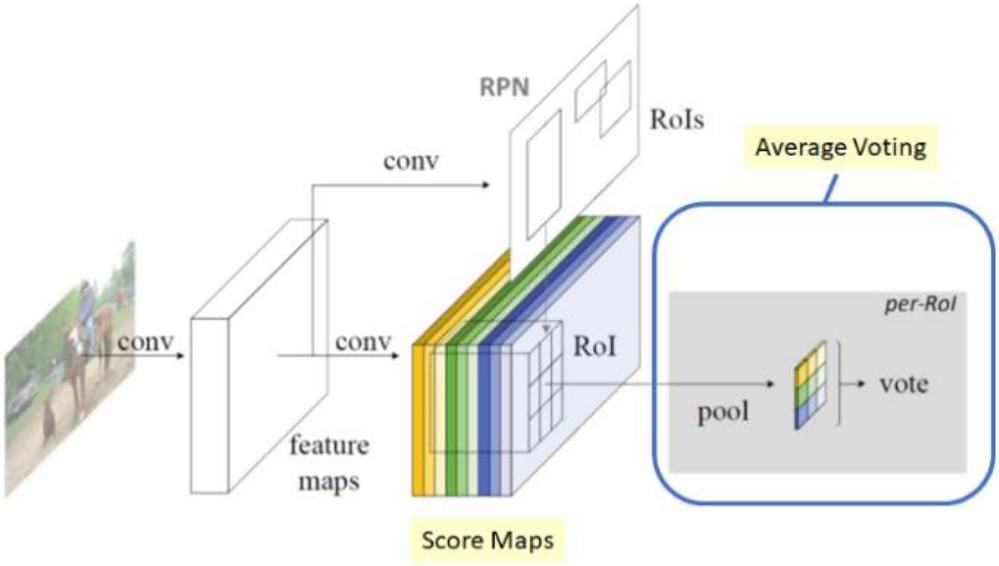
Pilna modelio struktūra yra pateikta toliau esančiame 2.6 paveiksle.



2.6 pav. Bendra *Faster R – CNN* modelio architektūra [22]

2.6.2. *R – FCN* architektūra

Kita architektūra, panaudota atliekant darbą, yra *R – FCN* (angl. *Region- based Fully Convolutional Network*). Ji buvo sukurta „Microsoft“ kompanijos, kartu bendradarbiaujant su Tsinghua universitetu 2016 metais [23]. Naudojant įprastinius *RPN* metodus, tokius kaip *R – CNN*, *Fast R – CNN* ar prieš tai aprašytą *Faster R – CNN*, *region proposal* sritys pirmiausiai sugeneruojamos *RPN* etapo vykdymo metu. Tada pritaikomas *RoI sutelkimo* metodas ir yra pereinama per pilnai sujungtus sluoksnius siekiant spręsti klasifikavimo uždavinius. *Pilnai sujungtų sluoksnių* proceso vykdymo metu duomenimis nėra keičiamasi su *RoI* modeliu, todėl *RPN* metodai tampa neefektyvūs laiko atžvilgiu. Tačiau naudojant *R – FCN* architektūrą, pilnai sujungti sluoksniai po *RoI sutelkimo* yra pašalinami. Tai leidžia sudėtingiausius skaičiavimus vykdyti prieš *RoI* procesą. Toliau esančiame 2.7 paveiksle pateikta *R – FCN* architektūra.



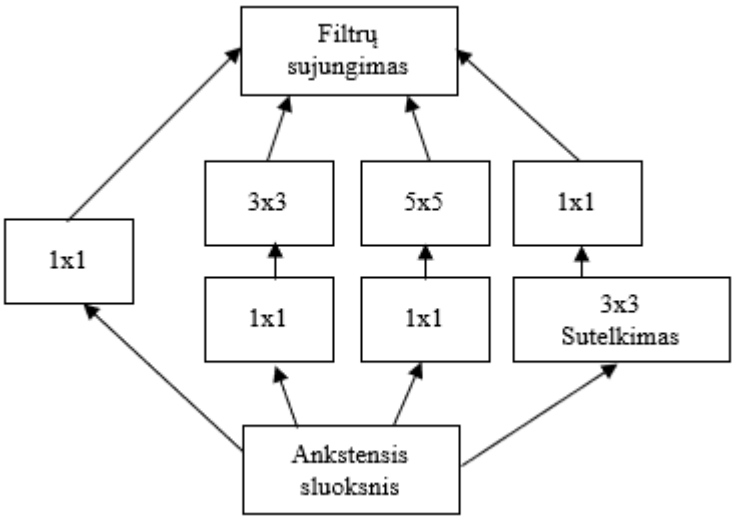
2.7 pav. Bendra *R – FCN* architektūra [23]

2.7. Naudotos neuroninių tinklų struktūros

2.7.1. *Inception_V2* struktūra

Naudojant *Faster – RCNN* architektūrą, tyrimams atlikti buvo pasirinkta *Inception_V2* neuroninių tinklų struktūra, kuri yra patobulinta *Inception_V1* versija. Sprendžiant vaizdų atpažinimo uždavinius dažna problema tampa tai, kad objektai esantys vaizduose yra nevienodo dydžio, todėl atliekant konvoliucinių tinklų operacijas, sunku parinkti tinkamą *kernelio* filtro dydį [24]. Didesnis šio filtro dydis geriau pritaikomas platesniame vaizdo diapazone, o maženis filtras yra tinkamesnis apdoroti informacijai kuri yra lokalesnė. Šio tipo problemai spręsti gali būti panaudotas kelių dydžių filtrų susiejimas, kada panaudojami trijų dydžių filtrai (1x1, 3x3, 5x5) ir *maksimalaus sutelkimo* procesas.

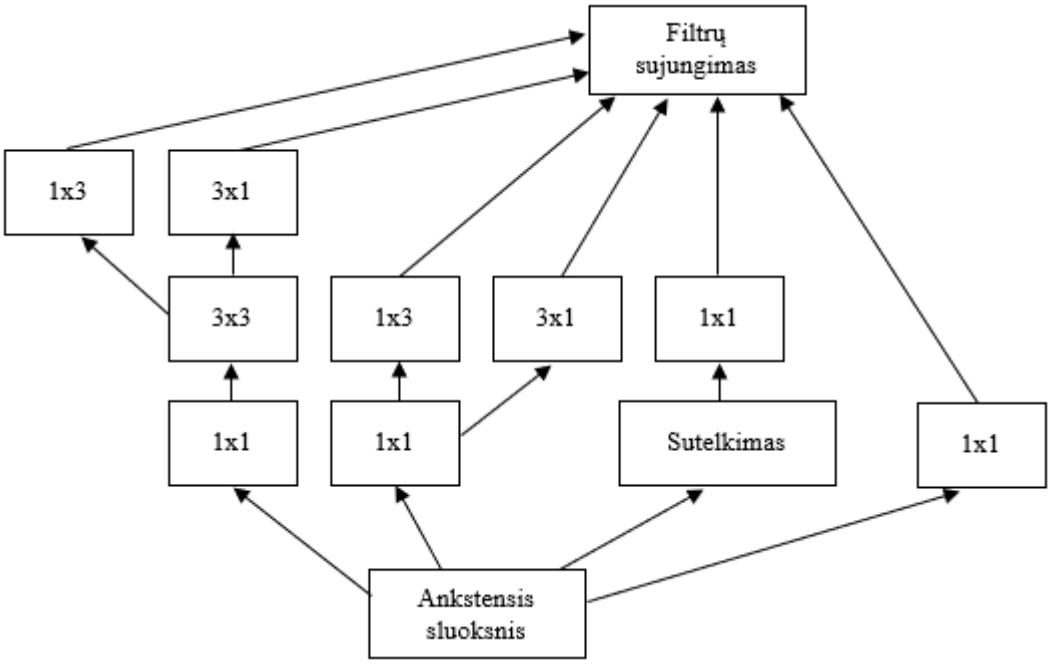
Siekiant sumažinti skaičiavimų resursus, vykdomas įvesties kanalų skaičiaus apribojimas papildomai pridedant 1x1 dydžio filtus prieš 3x3 ir 5x5 filtrus. Bendras *Inception_V1* sistemos modelis pateiktas toliau esančiame 2.8 paveiksle.



2.8 pav. *Inception_V1* modelis

Neuroninių tinklų modelių veiksmingumas yra didesnis, kai konvoliucijų operatoriai drastiškai nepakeičia įvesties duomenų dydžio, kadangi per didelis jų sumažinimas įtakoja informacijos praradimą. Siekiant išvengti šios problemos buvo sukurtas *Inception_V2 modelis*, kurio esmė – 5x5 konvoliucijų faktorizavimas į du atskirus 3x3 operatorius, taip paspartinant skaičiavimų atlikimus, kadangi 5x5 operatoriaus vykdymas užtrunka 2.78 karto ilgiau negu 3x3. Taigi dviejų 3x3 operatorių panaudojimas vietoje 5x5 pagreitina skaičiavimų atlikimą.

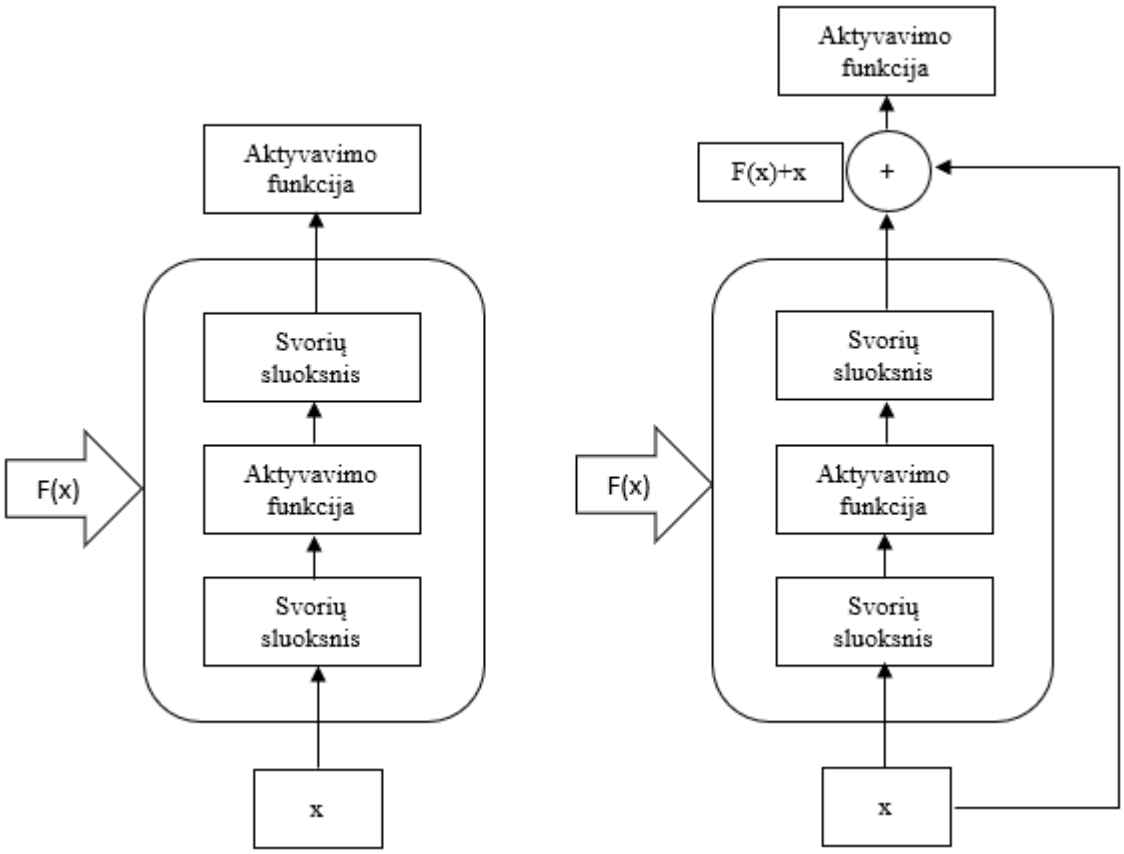
Dar vienas šio modelio sistemos patobulinimas – operatoriaus faktorizavimas į dviejų operatorių kombinaciją. Pavyzdžiui, 3x3 operatoriaus apdorojimas yra ekvivalentiškas procesui, kada pirmiau apdorojamas 1x3 operatorius, o po to – 3x1. Naudojant šį procesą, skaičiavimų resusrai yra sumažinami 33%. Bendras *Inceprion_V2* modelis pateikiamas 2.9 paveiksle.



2.9 pav. *Inception_V2* modelis

2.7.2. *ResNet – 101* struktūra

Problema, su kuria susiduriama kuriant neuroninius tinklus su vis daugiau sluoksnių, yra ta, kad tampa sudėtinga tokius modelius treniruoti, tikslumas pradeda *persisotinti* ir blogėti [25]. Būtent dėl šių priežasčių buvo sukurta *ResNet* neuroninių tinklų struktūra. Pagrindinis dalykas, kuris skiria *liekamuosius* (angl. *residual*) tinklus nuo įprastinių, yra tiesioginis ryšys, kuris praleidžia kai kuriuos sluoksnius. Šis ryšys yra vadinamas praleidimo ryšiu, kuris išsprendžia nykstančio grandiento problemą, leisdamas jam pereiti. Šie ryšiai naudojami tam, kad būtų galima išmokti tapatumo funkcijas, užtikrinančias, kad aukštesnis sluoksnis veiks bent jau taip pat gerai, kaip ir ankstesnis. Darbo metu, naudojant *R – FCN* architektūrą, tyrimams atlikti buvo pasirinkta *ResNet – 101* (*angl. Residual Network*) struktūra. *ResNet – 101* struktūros skaičiai nusako kiek sluoksnių panaudota sudarant modelį. Toliau esančiame 2.10 paveiksle pavaizduota įprasta konvoliucinių neuroninių tinklų struktūra (kairėje) ir *ResNet* (dešinėje).



2.10 pav. Standartinė ir *ResNet* struktūros

2.8. Tyrimams atlikti naudoti skaičiavimai

2.8.1. Klaidų matrica

Atlikus klasifikatoriaus treniravimo procesą, siekiant įvertinti jo patikimumą, yra sudaromos lentelės, vadinamos klaidų (angl. *confusion*) matricomis, kurios naudojamos apibūdinti klasifikatoriaus modelio efektyvumą [26]. Šioje lentelėje yra atvaizduojami dviejų arba daugiau klasių gauti rezultatai.

Standartiniu atveju, matricoje išskiriamos šios dalys:

- tikroji klasė (yra žinoma, ar pavyzdžiui augalas paveiktas ligos, ar ne);
- prognozuojama klasė (tai, ką nustato klasifikatoriaus modelis);
- TP (True Positive)- tai atvejis, kada yra žinoma, kad augalas yra paveiktas ligos ir taip „teigia“ klasifikatoius;
- TN (True Negative)- atvejis, kada yra žinoma, kad augalas nėra paveiktas ligos ir taip „teigia“ klasifikatorius;
- FP (False Positive)- atvejis, kada yra žinoma, kad augalas nėra paveiktas ligos, tačiau klasifikatorius „teigia“, kad augalas yra ligotas;
- FN (False Negative)- tai atvejis, kada yra žinoma, kad augalas yra paveiktas ligos, tačiau klasifikatorius „teigia“, kad augalas yra sveikas.

Kaip tokia sistema atrodo bendruoju atveju, pateikiama 2.3 lentelėje.

2.3 lentelė. Klaidų matrica bendruoju atveju

		Prognozuojama klasė	
		Taip	Ne
Tikroji klasė	Taip	TP	FN
	Ne	FP	TN

Kadangi atliekant ligotų kviečių pasėlių atpažinimo tyrimą vaizduose yra ieškoma ligotų lapų bei stiebų, klaidų matrica įgyja kitą struktūrą. Vietoje sveikų ir ligos paveiktų augalų klasių atsiranda ligotų lapų, ligotų stiebų ir sveikų augalų klasės. Sveikų augalų klasę į klaidų matricą reikia įtraukti todėl, kad vaizde esančių ligotų lapų ar stiebų klasifikatoriaus modelis gali neatpažinti, todėl šiuos atvejus būtina užfiksuoti. Darbe naudotos matricos struktūra, skirta įvertinti rezultatus ligotų lapų klasės atžvilgiu, pateikiama 2.4 lentelėje.

2.4 lentelė. Klaidų matrica, sudaryta pirmai klasei

		Prognozuojama klasė		
		Ligoti lapai	Ligoti stiebai	Sveiki kviečiai
Tikroji klasė	Ligoti lapai	TP	FN	FN
	Ligoti stiebai	FP	TN	TN
	Sveiki kviečiai	FP	TN	TN

Kita matricos struktūra, skirta įvertinti rezultatams ligotų stiebų klasės atžvilgiu, pateikta 2.5 lentelėje.

2.5 lentelė. Klaidų matrica, sudaryta antrai klasei

		Prognozuojama klasė		
		Ligoti lapai	Ligoti stiebai	Sveiki kviečiai
Tikroji klasė	Ligoti lapai	TN	FP	TN
	Ligoti stiebai	FN	TP	FN
	Sveiki kviečiai	TN	FP	TN

Į šias klaidų matricas yra surašomi visi atvejai (teisingai ar neteisingai atpažinti objektai). Pavyzdžiui, jei vaizde yra ligotas lapas ir klasifikatorius „teigia“, kad tai ligotas lapas, šis atvejis užfiksuojamas ir įrašomas į atitinkamą lentelės vietą. Atliekant tolimesnius tyrimus visi tokie patys gauti atvejai yra susumuojami.

Remiantis galutinėmis suskaičiuotomis TN, TP, FP ir FN vertėmis, siekiant įvertinti klasifikatoriaus veiksmingumą, tyrimuose dažnai yra naudojamos tokios reikšmės kaip *Recall*, *Precision*, *Accuracy*, *F – Score*, *ROC Curve*, *False / Positive* koeficientas, *False / Negative* koeficientas. Šiame darbe atliekant klasifikatorių patikimumo tyrimus panaudotas *False/Negative* koeficientas, *Accuracy* bei *F – Score* reikšmės.

2.8.2. *False/Negative* koeficientas

Vienas iš dažniausiai naudojamų įverčių, siekiant nustatyti klasifikavimo modelio veiksmingumą, yra *False/Negative* koeficientas. Šis koeficientas parodo, kokia dalis tam tikros klasės objektų buvo neteisingai suklasifikuota. Pavyzdžiui, vienu iš darbe naudojamų atvejų, False / Negative koeficientas parodo kokia dalis ligotų lapų buvo neteisingai suklasifikuota (ligoti lapai buvo priskirti ligotų stiebų ir sveikų kviečių klasėms). Šio koeficiento vertė yra apskaičiuojama naudojantis toliau pateikta 1 formule:

$$False/Negative\ koef. = \frac{FN}{FN+TP} \quad (1)$$

Darbe šis koeficientas yra atvaizduojamas grafikuose, parodant jo priklausomybę nuo kintančių struktūrų paramatrų.

2.8.3. *Accuracy* reikšmė

Siekiant įvertinti klasifikatoriaus modelio patikimumą, naudojama tikslumo (angl. *accuracy*) reikšmė, kuri apskaičiuojama naudojantis toliau pateikta 2 formule:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (2)$$

Ši reikšmė nurodo bendrą modelio tikslumą. Darbo metu atlikus tyrimus, grafikuose *Accuracy* ir *F – Score* reikšmės atvaizduojamos grafikuose parodant jų priklausomybę nuo kintančių struktūrų parametrų (vaizdo dydžio ir *batch* reikšmės).

2.8.4. *F- Score* reikšmė

F- Score reikšmė yra *Precision* ir *Recall* verčių vidurkis [27]. Šio metodo naudojimas yra naudingas, kai skirtingų klasių testavimo duomenų kiekiai nėra vienodi. *Recall* vertė gaunama naudojant 3 formulę, o *Precision* vertė apskaičiuojama pagal 4 formulę:

$$Recall = \frac{TP}{TP+FN} \tag{3}$$

$$Precision = \frac{TP}{TP+FP} \tag{4}$$

F- Score reikšmė gaunama imant *Recall* ir *Precision* įverčius, kurie panaudojami taikant 5 formulę:

$$FScore = 2 * \frac{(Precision*Recall)}{(Precision+Recall)} \tag{5}$$

3. Tyrimo rezultatų dalis

Šioje dalyje pateikiamos struktūros, su kuriomis atlikti tyrimai, aprašomi naudoti jų įverčiai. Klasifikatorių rezultatai pateikiami klaidų matricose, kuriomis remiantis apskaičiuojama *Fasle / Negative* koeficiento priklausomybė nuo modelio parametrų pokyčių, *Accuracy, F-Score* reikšmės.

3.1. Atliktų tyrimų variantai

Darbo metu buvo atlikti tyrimai naudojant *Inception_V2* ir *ResNet – 101* struktūras, keičiant vaizdo dydžio apdorojimo ir *batch* parametrus. Naudoti variantai pateikiami žemiau esančiose lentelėse (3.1 ir 3.2 lentelės), kuriose surašyti ir treniravimo žingsnių skaičiai.

3.1 lentelė. *Inception_V2* struktūros variantai

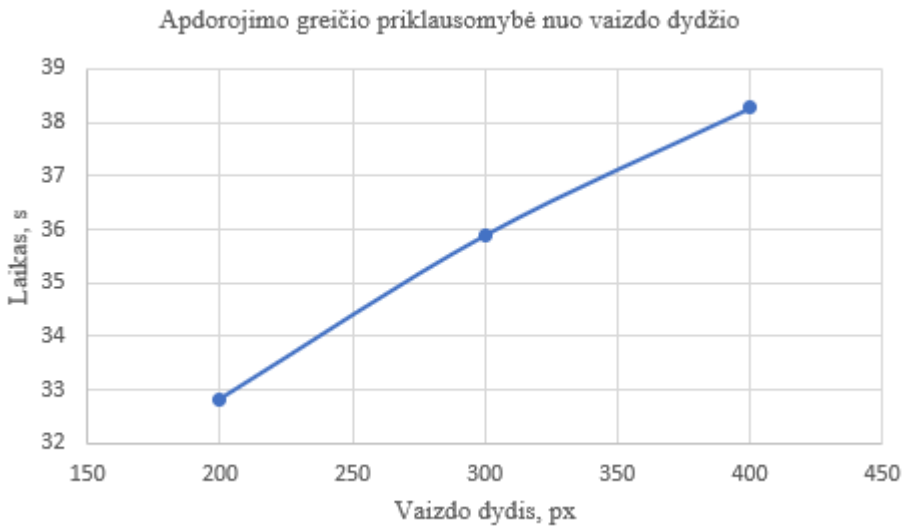
Varianto nr.	Pradinė mokymosi reikšmė	Vaizdo dydis, px	*Batch* dydis	Treniravimo žingsnių sk.
1	0,001	200x200	2	275940
2	0,001	300x300	2	260025
3	0,001	400x400	2	227624
4	0,001	300x300	1	324191
5	0,001	300x300	4	199074

3.2 lentelė. *ResNet – 101* struktūros variantai

Varianto nr.	Pradinė mokymosi reikšmė	Vaizdo dydis, px	*Batch* dydis	Treniravimo žingsnių sk.
1	0,001	300x300	1	203630
2	0,001	300x300	2	118172
3	0,001	300x300	4	39905

3.2. Atlikti tyrimai naudojant *Inception_V2* struktūrą

Pirmiausiai klasifikatorius buvo treniruojamas naudojant *Inception_V2* struktūrą ir *Faster R – CNN* architektūrą. Pradinė mokymosi vertė buvo parinkta 0,001, vaizdo dydis 200x200 px., o *batch* dydis - 2. Modelio treniravimo trukmė – 275940 žingsnių. Toliau buvo atlikti dar du modelio treniravimai keičiant vaizdo dydį į 300x300 px. ir 400x400 px. Atlikus bandymus su šiais modeliais, buvo fiksuojamas laikas, reikalingas suklasifikuoti visus 55 vaizdus. Gauti rezultatai pateikiami toliau esančiame 3.1 paveiksle.



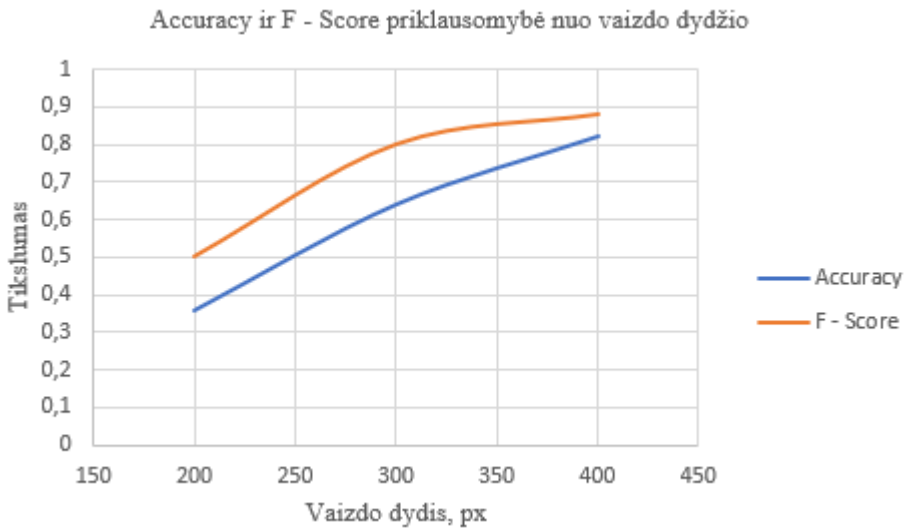
3.1 pav. Duomenų apdorojimo spartos priklausomybė nuo vaizdo dydžio

Atlikus modelių treniravimo procesą, rezultatų patikrinimui buvo parinkta 0,6 slenkstinė vertė. Kiekvieno varianto rezultatai buvo surašyti į klaidų matricas. *Inception_V2* konvoliucinių neuroninių tinklų struktūros pirmo varianto matrica pateikta žemiau esančioje 3.3 lentelėje, o kitų variantų matricos, taip pat ir *ResNet – 101* struktūros, pateiktos priede (žiūrėti 1 pr.).

3.3 lentelė. *Inception_V2* 1 varianto klaidų matrica

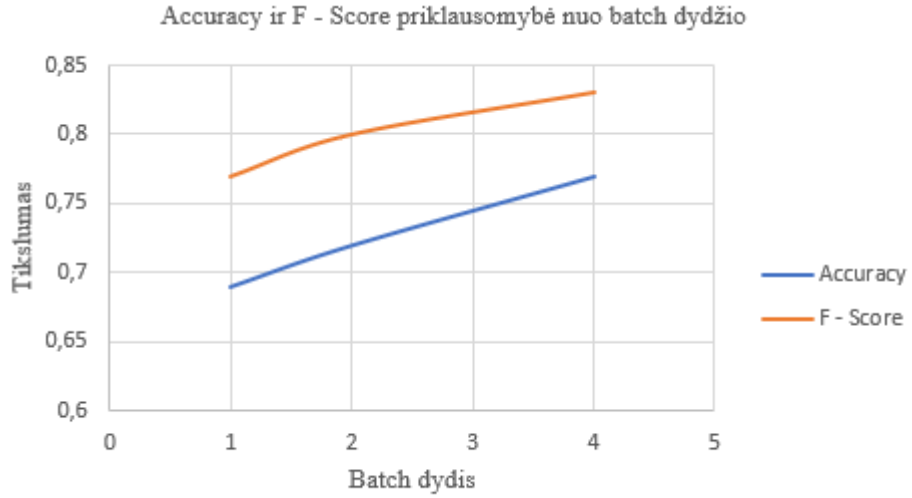
		Prognozuojama klasė		
		Ligoti lapai	Ligoti stiebai	Sveiki kviečiai
Tikroji klasė	Ligoti lapai	20	1	31
	Ligoti stiebai	2	12	22
	Sveiki kviečiai	0	0	0

Naudojantis pirmų trijų variantų klaidų matricomis, ir metodinėje dalyje aprašytomis formulėmis, buvo apskaičiuotos *Accuracy* ir *F – Score* reikšmės. Jos buvo atvaizduotos grafike, kuris pateiktas toliau esančiame 3.2 paveiksle. Grafike yra atvaizduotos šių reikšmių priklausomybės nuo konfigūraciniuose failuose parinktų vaizdo dydžių. Iš šio grafiko matyti, kad didinant šį parametrą, *Accuracy* ir *F – Score* vertės taip pat didėja.



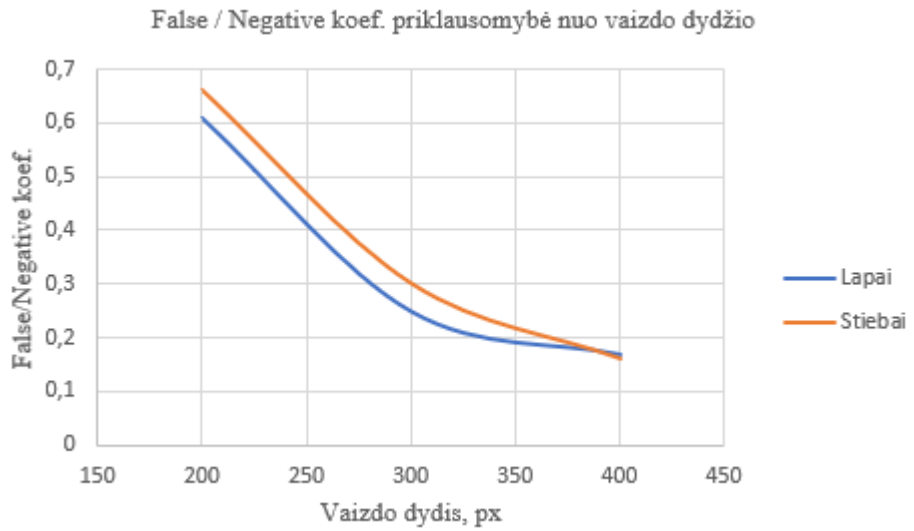
3.2 pav. *Inception_V2* struktūrų tikslumo priklausomybė nuo vaizdo dydžio

Įvertinus modelio greitaveiką ir *Accuracy* bei *F – Score* verčių priklausomybę nuo vaizdo dydžio parametrų, buvo nuspręsta tolimesniems tyrimams naudoti 300x300 px. vaizdo dydžio parametrą. Šį parametrą išlaikant vienodą, buvo keičiamas *batch* dydis. Atlikti dar 2 bandymai, *batch* reikšmę keičiant į 1 ir 4. Variantai pateikti 3.1 lentelėje (4 ir 5 variantai). Atlikus šiuos tyrimus, nustatyta, kad didinant modelio konfigūraciniame faile esantį *batch* dydį, *Accuracy* ir *F – Score* reikšmės taip pat didėja. Šių parametrų priklausomybės pateiktos toliau esančiame grafike (3.3 pav.).

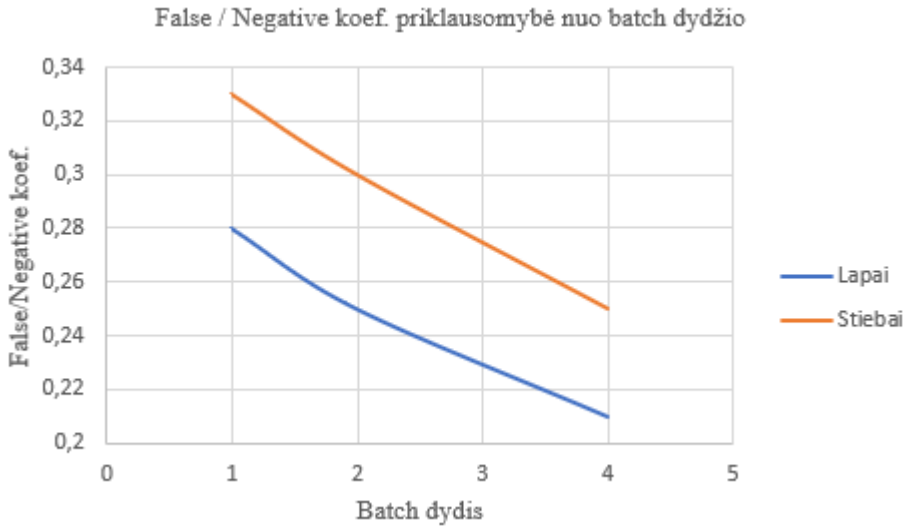


3.3 pav. Inception_V2 modelių tikslumo priklausomybė nuo batch dydžio

Naudojant *Inception_V2* struktūrą taip pat buvo atlikti tyrimai, siekiant išsiaiškinti *False / Negative* koeficiento priklausomybę nuo vaizdo ir *batch* dydžių. Šis koeficientas parodo, kokia dalis tam tikros klasės objektų buvo neteisingai suklasifikuota. Žemiau esančiuose grafikuose (3.4 ir 3.5 paveikslai) pateikiami gauti rezultatai. Iš jų matyti, kad didinant vaizdo ir batch dydžius, neteisingai suklasifikuotų objektų skaičius mažėja. Grafikuose pateikti rezultatai lapų ir stiebų klasėms atskirai.

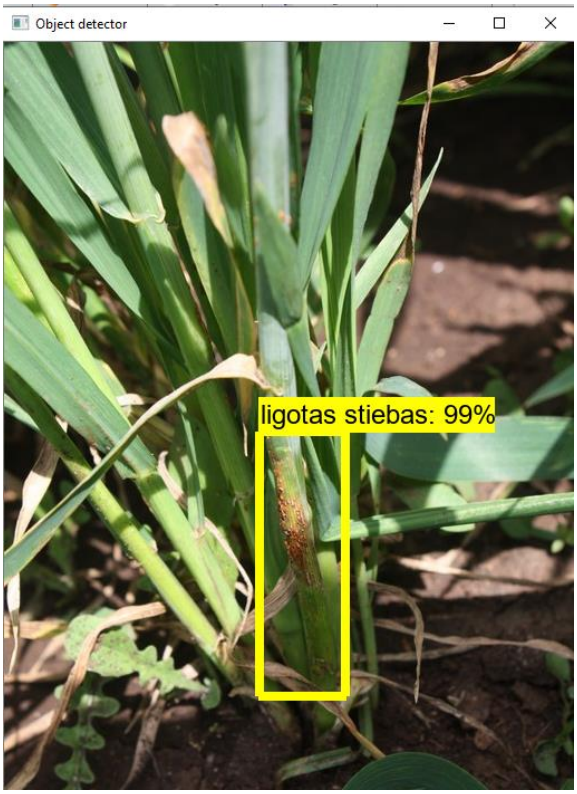


3.4 pav. *Inception_V2* struktūros *False / Negative* koef. priklausomybė nuo vaizdo dydžio



3.5 pav. *Inception_V2* struktūros *False / Negative* koef. priklausomybė nuo batch dydžio

Toliau esančiame 3.6 paveiksle, pateikiama kaip atrodo objekto atpažinimo procesas. Rezultatų lange yra nurodoma objekto klasė ir klasifikatoriaus „įsitikinimo“ reikšmė. Šiuo atveju buvo teisingai nustatytas rūdžių ligos paveiktas kviečio stiebas.



3.6 pav. Atpažintas rūdžių ligos paveiktas stiebas

Taip pat buvo atliktas tyrimas į *Inception_V2* struktūros konfigūracinį failą įtraukiant kontrasto ir šviesos keitimo funkciją. Naudojant šią funkciją, galima teigti, kad modelis nepasiteisino, kadangi gauti prastesni rezultatai lyginant su kitais modeliais. 3.7 paveiksle yra pateikiamas vaizdas kaip atrodo klaidingai atpažįstamų objektų procesas. 3.4 lentelėje pateikta šio varianto klaidų matrica, iš kurios apskaičiuota *Accuracy* (0,12) ir *F – Score* (0,18) reikšmės.

3.4 lentelė. Klaidų matrica, į mokymą įtraukus kontrasto ir šviesos keitimo funkciją

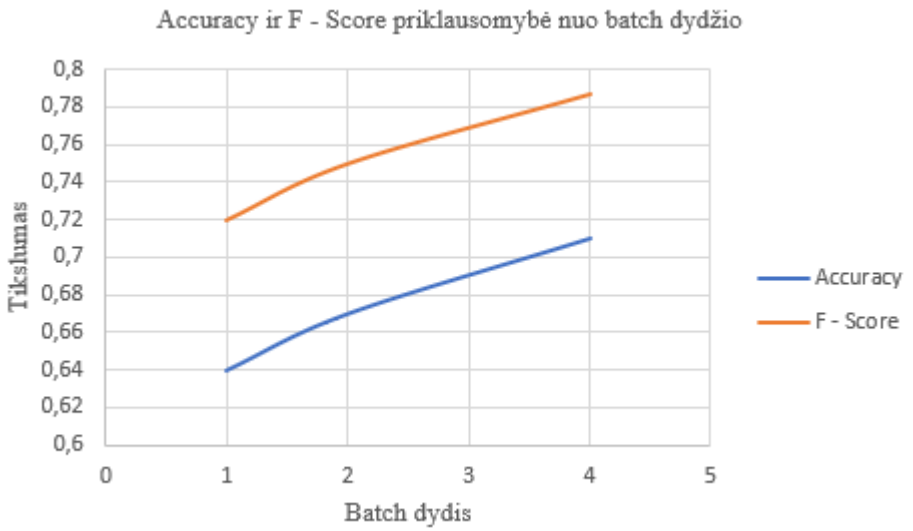
		Prognozuojama klasė		
		Ligoti lapai	Ligoti stiebai	Sveiki kviečiai
Tikroji klasė	Ligoti lapai	8	2	42
	Ligoti stiebai	16	3	17
	Sveiki kviečiai	0	0	0

***3.7* pav.** Neteisingai atpažinti objektai

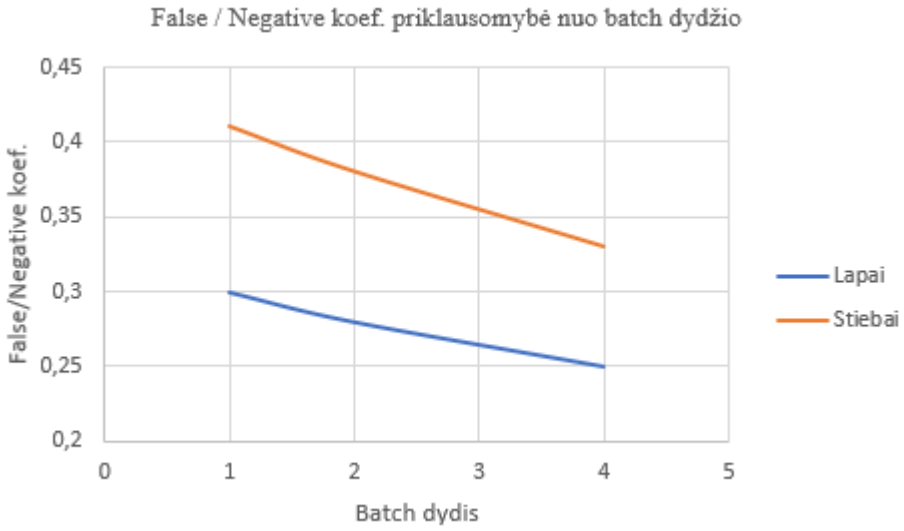
Taip pat *Inception_V2* struktūrą buvo bandoma treniruoti *batch* parametro reikšmę pakeitus į 8. Esant šiai reikšmei, treniravimo procesas nebuvo pradėtas dėl nepakankamų kompiuterio resursų. Tada *batch* dydis buvo sumažintas iki 6, tačiau modeliui treniravimosi metu pasiekus 128 žingsnį, treniravimo procesas buvo nutrauktas dėl tos pačios priežasties (nepakankami kompiuterio resursai).

3.3. Atlikti tyrimai naudojant *ResNet – 101* struktūrą

Atlikus bandymus su *Inception_V2* struktūra, toliau buvo treniruojamas klasifikatorius naudojant *ResNet – 101* struktūrą ir *R - FCN* architektūrą. Pradinė mokymosi vertė buvo parinkta 0,001, vaizdo dydis 300x300 px., o *batch* dydis - 1. Modelio treniravimo trukmė – 203630 žingsnių. Toliau buvo atliekami dar du šio modelio treniravimai, keičiant konfigūraciniame faile esantį *batch* dydį. Atlikus objektų atpažinimo bandymus, buvo sudarytos klaidų matricos, iš kurių apskaičiuotos *Accuracy*, *F – Score* ir *False / Negative* koeficientų reikšmės. *Accuracy* ir *F – Score* verčių priklausomybės nuo batch dydžio pateiktos toliau esančiame grafike (3.8 pav.). *False / Negative* priklausomybė nuo batch dydžio pateikiama 3.9 paveiksle. Remiantis šiais grafikais, galima teigti, kad modelio tikslumas, didinat batch dydį, didėja. *Batch* vertei, esant 4, gauta 0,71 *Accuracy* reikšmė ir 0,78 *F – Score* vertė. Iš *False / Negative* koeficiento priklausomybės nuo *batch* dydžio matoma, kad rūdžių ligos paveiktų lapų atpažinimas yra tikslesnis negu stiebų.

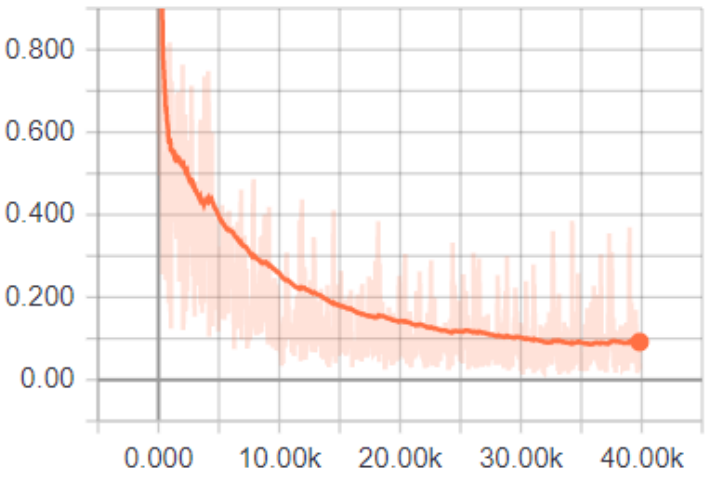


3.4 pav. *ResNet – 101* struktūrų tikslumo priklausomybė nuo batch dydžio



3.9 pav. *ResNet – 101* struktūrų *False / Negative* koef. priklausomybė nuo batch dydžio

Toliau esančiame paveiksle yra pavaizduota kaip atrodo klasifikatoriaus mokymosi grafikas (3.10 pav.). Jame parodyta, kaip keičiasi *Total Loss* reikšmė, mokymosi proceso eigoje. Šiuo atveju pateiktas *ResNet – 101* struktūros mokymosi grafikas, naudojant 3 varianto parametrus (žiūrėti 3.2 lentelė).



3.10 pav. *ResNet- 101* struktūra su batch 4 parametru

Išvados ir rezultatai

1. Darbo metu atlikta literatūros analizė, kurios metu apžvelgti moksliniai straipsniai, ligų paveiktų pasėlių atpažinimo srityje. Atsižvelgiant į literatūros analizę nustatyta, kad dažnai naudojami *SVM* ir *CNN* klasifikatoriai. Tiksliausi rezultatai gaunami taikant *CNN* klasifikatorius.
2. Sudarytos duomenų bazės klasifikatorių mokymui ir rezultatų patikrinimui. „kaggle“ tinklapyje pateiktą žaliavinę medžiagą klasifikatorių mokymui reikėjo paruošti naudojant „LabelImg“ programą.
3. Tyrimams atlikti panaudotos *Inception_V2* ir *ResNet – 101* neuroninių tinklų struktūros naudojant *Faster R – CNN* ir *R – FCN* architektūras.
4. Atlikus rezultatų patikrinimą, sudarytos klaidų matricos. Jomis naudojantis, apskaičiuotos *Accuracy, F – Score* ir *False / Negative* koeficientų reikšmės. Šios reikšmės atvaizduotos grafikuose, nustatant jų priklausomybę nuo struktūros konfigūracinio failo parametrų.
5. Įvertinus gautus rezultatus, didžiausios *Accuracy* ir *F - Score* reikšmės gautos naudojant *Inception_V2* struktūrą su didžiausiu vaizdo dydžiu (400x400 px.). Atitinkamai gautos reikšmės yra 0,82 ir 0,88. Mažiausi *False / Negative* koeficientai gauti naudojant tuos pačius struktūros parametrus. Koeficiento skirto lapų klasei reikšmė yra 0,17, o stiebų – 0,16.
6. Naudojant *Inception_V2* neuroninių tinklų struktūrą ir tyrimus atliekant keičiant *batch* dydį, geriausi rezultatai gauti *batch* dydžiui esant 4. Gauta *Accuracy* reikšmė yra 0,77, o *F – Score* – 0,83. Tai rodo, kad didinant *batch* dydį išauga klasifikatoriaus tikslumas.
7. Tyrimus atlikus su *ResNet – 101* struktūra, geriausi rezultatai taip pat gauti esant didžiausiai *batch* vertei. *Accuracy* reikšmė – 0,71, *F – Score* – 0,78. *False / Negative* koeficientas – stiebams 0,33, o lapams – 0,25. Naudojant *Inception_V2* struktūrą gauti šiek tiek geresni klasifikatoriaus rezultatai, todėl ji yra tinkamesnė ligotų pasėlių atpažinimo uždaviniams spręsti negu *ResNet* struktūra.

Literatūros sąrašas

1. Varsha P. Gaikwad. Assist Prof. IT dept. Govt. College of Engg. Aurangabad, India. Wheat Disease Detection Using Image Processing [žiūrėta 2020 balandžio 6d.]. Prieiga per internetą: https://ieeexplore.ieee.org/document/8122158
2. Sumit Nema Department of computer engineering Global nature group of sangathan group of institution Jabalpur, India. Wheat Leaf Detection and Prevention Using Support Vector Machine [žiūrėta 2020 balandžio 6d.]. Prieiga per internetą: https://ieeexplore.ieee.org/document/8821098
3. Anshuman Singh, Monika Arora. Department of Electronic and Communication Engineering ASET, Amity University Noida, Uttar Pradesh, India. CNN Based Detection of Healthy and Unhealthy Wheat Crop [žiūrėta 2020 balandžio 6d.]. Prieiga per internetą: https://ieeexplore.ieee.org/document/9215340
4. Shima Ramesh. Assistant Professor: department of electronics and communication, MVJ college of Engineering. Bangalore, India. Plant Disease Detection Using Machine Learning [žiūrėta 2020 balandžio 6d.]. Prieiga per internetą: https://www.researchgate.net/publication/327065422_Plant_Disease_Detection_Using_Machine_Learning
5. Mrinal Kumar. ME Student, Department of ECE, BIT Mesra, Ranchi Wheat Leaf Disease Detection Using Image Processing [žiūrėta 2020 balandžio 9d.]. Prieiga per internetą: https://www.ijltemas.in/DigitalLibrary/Vol.6Issue4/73-76.pdf
6. Peifeng Xu. Engineering and Technology Center for Modern Horticulture Jiangsu Polytechnic College of Agriculture and Forestry, Jurong, 212400, China. Automatic Wheat Leaf Rust Detection and Grading Diagnosis via Embedded Image Processing System [žiūrėta 2020 balandžio 9d.]. Prieiga per internetą: https://www.sciencedirect.com/science/article/pii/S1877050917304520
7. Han, L., Haleem, M. S., & Taylor, M. (2015). A novel computer vision-based approach to automatic detection and severity assessment of crop diseases. 2015 Science and Information Conference (SAI) [žiūrėta 2020 balandžio 9d.]. Prieiga per internetą: https://ieeexplore.ieee.org/document/7237209
8. Genaev, M., Ekaterina, S., & Afonnikov, D. (2020). Application of neural networks to image recognition of wheat rust diseases. 2020 Cognitive Sciences, Genomics and Bioinformatics (CSGB) [žiūrėta 2020 balandžio 11d.]. Prieiga per internetą: https://ieeexplore.ieee.org/document/9214703
9. Altaf Hussain Department of Computer Science, Islamia Collage University Peshawar, Peshawar, Pakistan. Automatic Disease Detection in Wheat Crop using Convolution Neural Network [žiūrėta 2020 balandžio 11d.]. Prieiga per internetą: https://www.researchgate.net/publication/343206552_Automatic_Disease_Detection_in_Wheat_Crop_using_Convolution_Neural_Network
10. Rashid, M., Ram, B., Batth, R. S., Ahmad, N., Elhassan Ibrahim Dafallaa, H. M., & Burhanur Rehman, M. (2019). Novel Image Processing Technique for Feature Detection of Wheat Crops using Python OpenCV. 2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE) [žiūrėta 2020 balandžio 18d.]. Prieiga per internetą: https://ieeexplore.ieee.org/abstract/document/9004432
11. V. Ramya, M. Anthuvan Lydia. Assistant Professor, Department of ECE, Saranathan College of Engineering, Trichy, India. Leaf Disease Detection and Classification using Neural Networks

[žiūrėta 2020 balandžio 18d.]. Prieiga per internetą: https://www.ijarcce.com/upload/2016/november-16/IJARCCE%2044.pdf

12. Sujatha R, Y Sravan Kumar and Garine Uma Akhil School of Information Technology and Engineering, VIT University, Vellore. Leaf disease detection using image processing [žiūrėta 2020 balandžio 18d.]. Prieiga per internetą: https://www.researchgate.net/publication/318109025_Leaf_Disease_Detection_using_Image_Processing
13. Srdjan Sladojevic, Department of Industrial Engineering and Management, Faculty of Technical Sciences, University of Novi Sad, Trg Dositeja Obradovica 6, 21000 Novi Sad, Serbia. Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification [žiūrėta 2020 balandžio 20d.]. Prieiga per internetą: https://www.hindawi.com/journals/cin/2016/3289801/
14. BitRefine įmonės aprašymas [žiūrėta 2020 balandžio 20d.]. Prieiga per internetą: https://bitrefine.group/industries/precision-agriculture/88-industries/agriculture-food/agriculture-solutions/184-plant-disease-detection
15. Taranis įmonės aprašymas [žiūrėta 2020 balandžio 20d.]. Prieiga per internetą: https://taranis.ag/
16. Tyrimų objektas [žiūrėta 2020 kovo 3d.]. Prieiga per internetą: https://www.kaggle.com/ibrahimgoke/cgiar-computer-vision-for-crop-disease-dataset
17. Pagrindinio kompiuterio procesoriaus aprašymas [žiūrėta 2021 balandžio 2d.]. Prieiga per internetą: https://ark.intel.com/content/www/us/en/ark/products/85212/intel-core-i5-5200u-processor-3m-cache-up-to-2-70-ghz.html
18. Grafinio kompiuterio procesoriaus aprašymas [žiūrėta 2021 balandžio 2d.]. Prieiga per internetą:] https://www.notebookcheck.net/NVIDIA-GeForce-940M.138027.0.html
19. Tensorflow bibliotekos aprašymas [žiūrėta 2020 lapkričio 27d.]. Prieiga per internetą: https://www.simplilearn.com/tutorials/deep-learning-tutorial/what-is-tensorflow
20. Konvoliucinių neuroninių tinklų aprašymas [žiūrėta 2021 gegužės 3d.]. Prieiga per internetą: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53
21. Konvoliucinių tinklų architektūra [žiūrėta 2021 gegužės 3d.]. Prieiga per internetą: https://www.researchgate.net/figure/Schematic-diagram-of-a-basic-convolutional-neural-network-CNN-architecture-26_fig1_336805909
22. Fastet R – CNN architektūros aprašymas [žiūrėta 2020 gruodžio 7d.]. Prieiga per internetą: https://tryolabs.com/blog/2018/01/18/faster-r-cnn-down-the-rabbit-hole-of-modern-object-detection/
23. R – FCN architektūros aprašymas [žiūrėta 2021 kovo 3d.]. Prieiga per internetą: https://towardsdatascience.com/review-r-fcn-positive-sensitive-score-maps-object-detection-91cd2389345c
24. Inception_V2 modelio aprašymas [žiūrėta 2021 balandžio 27d.]. Prieiga per internetą: https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202
25. ResNet modelio aprašymas [žiūrėta 2021 balandžio 27d.]. Prieiga per internetą: https://www.mygreatlearning.com/blog/resnet/
26. Klaidų matricos aprašymas [žiūrėta 2021 gegužės 4d.]. Prieiga per internetą: https://www.dataschool.io/simple-guide-to-confusion-matrix-terminology/

27. *F – Score* reikšmės aprašymas [žiūrėta 2021 gegužės 4d.]. Prieiga per internetą: https://medium.com/analytics-vidhya/accuracy-vs-f1-score-6258237beca2
28. Darbo atlikimui naudoti modeliai [žiūrėta 2020 kovo 3d.]. Prieiga per internetą: https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf1_detection_zoo.md

Priedai

1 priedas. Klaidų matricos

2 lentelė. *Inception_V2* 2 varianto klaidų matrica

		Prognozuojama klasė		
		Ligoti lapai	Ligoti stiebai	Sveiki kviečiai
Tikroji klasė	Ligoti lapai	39	2	11
	Ligoti stiebai	4	25	7
	Sveiki kviečiai	0	0	0

3 lentelė. *Inception_V2* 3 varianto klaidų matrica

		Prognozuojama klasė		
		Ligoti lapai	Ligoti stiebai	Sveiki kviečiai
Tikroji klasė	Ligoti lapai	43	2	7
	Ligoti stiebai	1	30	5
	Sveiki kviečiai	0	0	0

4 lentelė. *Inception_V2* 4 varianto klaidų matrica

		Prognozuojama klasė		
		Ligoti lapai	Ligoti stiebai	Sveiki kviečiai
Tikroji klasė	Ligoti lapai	37	3	12
	Ligoti stiebai	3	24	9
	Sveiki kviečiai	0	0	0

5 lentelė. *Inception_V2* 5 varianto klaidų matrica

		Prognozuojama klasė		
		Ligoti lapai	Ligoti stiebai	Sveiki kviečiai
Tikroji klasė	Ligoti lapai	41	2	9
	Ligoti stiebai	3	27	6
	Sveiki kviečiai	0	0	0

6 lentelė. *ResNet – 101* 1 varianto klaidų matrica

		Prognozuojama klasė		
		Ligoti lapai	Ligoti stiebai	Sveiki kviečiai
Tikroji klasė	Ligoti lapai	36	6	13
	Ligoti stiebai	6	21	9
	Sveiki kviečiai	0	0	0

7 lentelė. *ResNet – 101* 2 varianto klaidų matrica

		Prognozuojama klasė		
		Ligoti lapai	Ligoti stiebai	Sveiki kviečiai
Tikroji klasė	Ligoti lapai	37	3	12
	Ligoti stiebai	5	22	9
	Sveiki kviečiai	0	0	0

8 lentelė. *ResNet – 101* 3 varianto klaidų matrica

		Prognozuojama klasė		
		Ligoti lapai	Ligoti stiebai	Sveiki kviečiai
Tikroji klasė	Ligoti lapai	39	2	11
	Ligoti stiebai	5	24	7
	Sveiki kviečiai	0	0	0