

VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
INFORMATIKOS INSTITUTAS
PROGRAMŲ SISTEMŲ BAKALAURO STUDIJŲ PROGRAMA

Defektų aptikimas medžiagų paviršiuose

Material Surface Defect Detection

Bakalauro baigiamasis darbas

Atliko:	Bartas Šaltenis	(parašas)
Darbo vadovas:	dr. Vytautas Valaitis	(parašas)
Darbo recenzentas:	doc. dr. Linas Petkevičius	(parašas)

Vilnius – 2023

Padėka

Darbo autorius dėkoja Vilniaus universiteto Matematikos ir informatikos fakulteto Informacinių technologijų atviros prieigos centrui už suteiktus HPC išteklius šio darbo skaičiavimams atlikti.

Santrauka

Šiame darbe daugiausia dėmesio skiriama medžiagų paviršiaus defektų segmentavimui ir siekiama palyginti efektyvumą įvairių modelių, sukurtų ir pritaiktų plieno paviršiaus defektams segmentuoti. Šiai lyginamajai analizei naudojami modeliai yra skirtingos konvoliucinių neuroninių tinklų architektūros ir transformerių neuroninio tinklo architektūra, o jų efektyvumas lyginamas su geriausiu modeliu, kurio tikslumas siekė 98,74 %. Vertinant šių modelių efektyvumą, šiuo darbu siekiama nustatyti tinkamiausią paviršiaus defektų aptikimo ir segmentavimo modelį ir pateikti įžvalgų, padėsiančių padidinti būsimų paviršiaus defektų aptikimo sistemų tikslumą ir efektyvumą.

Kokybės patikrinimai yra labai svarbūs bet kuriame gamybos procese, o galutinio produkto kokybės užtikrinimas yra būtinas siekiant išlaikyti klientų pasitenkinimą ir prekės ženklo reputaciją. Tačiau kokybės patikrinimai gali pareikalauti daug laiko ir darbo, be to, galima žmogiškosios klaidos tikimybė. Integravus dirbtinį intelektą į kokybės patikrinimus, gaunama nauda – didesnis tikslumas, greitis ir nuoseklumas. Be to, dirbtiniu intelektu paremti kokybės patikrinimai gali aptikti defektų, kurių gali nepastebėti žmonės o tai galiausiai lemia aukštesnę produktų kokybę ir mažesnes išlaidas, susijusias su defektais.

Pakeitus svorio inicializavimą iš Xavier į Kaiming, pritaikius duomenų rinkinio normalizavimą ir pridėjus atsitiktinio praretinimo reguliarizaciją prie SegDecNet architektūros galima pagerinti šios architektūros rezultatus. Su ViT architektūros modeliais galima pasiekti geresnius rezultatus negu su CNN architektūros modeliais. Geriausia enkoderio ir dekoderio kombinacija yra PAN enkoderis su mobileVitv2 dekoderiu. Su šia kombinacija galima pasiekti geriausius rezultatus plieno defektų semantinio segmentavimo uždavinyje.

Raktiniai žodžiai: CNN, ViT, „Severstal“ duomenų rinkinys, semantinis segmentavimas, plieno defektų segmentavimas

Summary

The focus of this study is on material surface defect segmentation, with the aim of comparing the effectiveness of different models developed and adapted for defect segmentation of steel surfaces. The models used for this comparative analysis are different convolutional neural network architectures and a visual transformer architecture, and their performance is compared with the state-of-the-art model that achieved an accuracy of 98.74% . By evaluating the effectiveness of these models, this work identifies the most suitable models for surface defect detection and segmentation and provides insights for improving the accuracy and efficiency of future surface defect detection systems. The purpose is that.

Quality checks are essential in any manufacturing process and ensuring the quality of the final product is essential to maintain customer satisfaction and brand reputation. However, quality checks are time consuming, labour intensive and subject to human error. Integrating artificial intelligence into quality checks improves accuracy, speed and consistency. AI-powered quality inspections can detect defects that human inspectors might miss, ultimately improving product quality and reducing costs associated with defects and recalls.

Changing the weight initialisation from Xavier to Kaiming, applying dataset normalisation and dropout regularisation to the SegDecNet architecture can improve the performance of this architecture. The ViT architecture models can achieve better results than the CNN architecture models. The best combination of encoder and decoder is a PAN encoder with a mobileVITv2 decoder. With this combination, better results can be achieved for the semantic segmentation of steel defects task.

Keywords: CNN, ViT, „Severstal“ dataset, semantic segmentation, steel defect segmentation

TURINYS

ĮVADAS	7
1. LITERATŪROS ANALIZĖ	9
1.1. Vaizdo atpažinimas	9
1.2. Vaizdų segmentavimas	10
1.3. Plienai	11
1.4. Enkoderiai ir Dekoderiai	12
1.5. Enkoderio ir dekoderio architektūros	13
1.5.1. Unet	13
1.5.2. DeepLab	13
1.5.3. FPN ir PAN	13
1.5.4. PSPNet	14
1.5.5. Linknet	14
1.5.6. MAnet	14
1.6. Konvoliuciniai neuroniniai tinklai	14
1.6.1. CNN struktūra	15
1.7. CNN architektūros	17
1.7.1. VGG16 architektūra	17
1.7.2. ResNet architektūra	18
1.7.3. Inception architektūra	19
1.7.4. MobileNet architektūra	19
1.7.5. EfficientNet architektūra	20
1.7.6. DenseNet architektūra	20
1.7.7. Res2NeXt architektūra	20
1.8. Vaizdų transformerio modelis	21
1.8.1. MobileVit modelis	22
1.8.2. Segformer modelis	22
1.9. Didžiausią atpažinimo tikslumą pasiekusi architektūra – SegDecNet	22
2. METODOLOGIJA	24
2.1. Duomenų rinkinys	24
2.2. Tikslumo vertinimas	25
3. EKSPERIMENTINIAI TYRIMAI	27
3.1. Skirtingų tinklo architektūrų palyginimas	27
3.1.1. ResNet modeliai	27
3.1.2. VGG modelis	28
3.1.3. Inception modelis	28
3.1.4. MobileNet modelis	28
3.1.5. EfficientNet modeliai	28
3.1.6. Vaizdų transformerių modeliai	28
3.1.7. Enkoderių architektūros	28
3.1.8. Skirtingų architektūrų rezultatų įvertinimas ir palyginimas	29
3.2. SegDecNet architektūros tobulinimas	31
3.2.1. SegDecNet eksperimentinio tyrimo rezultatai	32
REZULTATAI IR IŠVADOS	33
ŠALTINIAI	35
SANTRUMPOS	38

PRIEDAI	39
1 priedas. Defektų radimas	39
2 priedas. GitHub repozitorijos nuoroda	40

Įvadas

Neapdoroto plieno pramonės augimas yra matomas visame pasaulyje. Pagal 2021 metų duomenis, neapdoroto plieno gamyba pasiekė 1958449 tūkstančių tonų per 2021 metus [Bas21]. Plieno produkcija pasaulio rinkoje 2015–2021 metų periodu išaugo 21 %. Plieno gamyba ir vartojimas dažnai laikomas šalies vystymosi rodikliu. Plienas yra metalas, kurio vientisumas gali būti pažeistas dėl įvairių priežasčių. Viena dažniausių priežasčių, dėl kurios atsiranda defektai plieno paviršiuje, yra mechaninis stresas. Mechaninis stresas gali atsirasti dėl įvairių veiksnių, tokių kaip trintis, smūgis arba spaudimas. Kitos galimos defektų atsiradimo priežastys gali būti perkaitimas arba korozija. Plieno paviršiaus defektų atradimas yra svarbi užduotis pramonėje, kadangi defektai gali sukelti rimtų problemų gamyboje ir galiausiai sumažinti produkto kokybę. Neuroniniai tinklai yra viena iš technologijų, kuri gali padėti aptikti plieno paviršiaus defektus. Šie tinklai yra specialiai sukurti analizuoti vaizdus ir yra labai efektyvūs atpažindami konkrečius bruožus ir struktūras. Jie gali būti naudojami atskirti sveiką plieno paviršių nuo defektų ir tokiu būdu padėti užtikrinti, kad gamybos procese būtų naudojamas tik kokybiškas plienas. Naudojant neuroninius tinklus, galima sutaupyti laiką ir išvengti brangių klaidų gamyboje, kadangi defektai gali būti atskleisti dar prieš pradedant gamybos procesą.

Neuroninių tinklų pritaikymas yra labai platus pramonėje, ypač defektų atradime įvairiuose gamybos procesuose. R. Augustauskas darė tyrimą apie konvoliucinių neuroninių tinklų panaudojimą defektų atradimui šaligatvčiuose [AL20]. Jo moksliniame straipsnyje naudojama U-Net architektūra atrasti plyšius ir įtrūkimus šaligatvčiuose. Taip pat konvoliuciniai neuroniniai tinklai pritaikomi defektams atrasti ir rentgeno nuotraukose. W. Du atliko tyrimą apie DeepLab, UNet ir kt. neuroninių tinklų architektūrų pritaikymą atrandant defektus įvairių objektų rentgeno nuotraukose, pavyzdžiui, automobilių dalių [DSF20]. Šiame darbe bus lyginama skirtingų neuroninių tinklų architektūrų pritaikymas segmentavimui plieno paviršiuose.

Tiriamas objektas:

Šiame darbe bus tiriami įvairios neuroninių tinklų architektūros, skirtos vaizdų segmentavimui, segmentavimo algoritmai ir plieno pramonė. Bus naudojamos šios architektūros: MobileNet [HZC⁺17], EfficientNet [TL19], ConvNeXt [LMW⁺22], U-net [RFB15], DeepLab [CPS⁺17] ir taip pat bus naudojamos transformerių architektūros modeliai tokie kaip ViT [DBK⁺20], SegFormer [XWY⁺21] ir MobileViT [MR21]. Taip pat tirama bus „Severstal“ įmonės pateiktas duomenų rinkinys. Duomenų rinkinį sudaro 12,568 nespaltotų plieno paviršiaus nuotraukų, kurie turi vieną iš keturių arba kelis defektus. Šiam duomenų rinkiniui yra pasiektas aukščiausias vidutinis preciziškumas – 98.74% [BTS21]. Bus tirama SegDecNet architektūra, su kuria buvo pasiektas aukščiausias vidutinis preciziškumas.

Darbo tikslas:

Atlikti lyginamąją analizę, lyginant savo sukurtus ir pritaikytus modelius plieno paviršiaus defektų segmentavimui su kitais sukurtais modeliais. Šiam tikslui pasiekti yra keliami tokie uždaviniai:

1. Išanalizuoti literatūrą apie skirtingas neuroninių tinklų architektūras siekiant apžvelgti ir įvertinti metodų tinkamumą sprendžiant defektų aptikimą plieno paviršiuje.

2. Išanalizuoti Severstal” įmonės pateiktą duomenų rinkinį, skirtą neuroninių tinklų mokymui. Išsiaiškinti duomenų rinkinio struktūrą, trikdžius ar iššūkius kurie galėtų atsirasti naudojant šį duomenų rinkinį.
3. Eksperimentiškai ištirti skirtingas architektūras ir sužinoti kokie gali būti pasiekti rezultatai naudojant šias architektūras plieno paviršiaus defektų segmentavime.
4. Sukurti modelius plieno paviršiaus defektų segmentavimui.
5. Eksperimentiškai palyginti sukurti modeliai su esamais sukurtais modeliais

Darbo prielaidos:

1. Bus realizuotos skirtingos architektūros plieno defektų segmentavimui.
2. Bus atlikta lyginamoji analizė su realizuotomis architektūromis.
3. Bus patobulinta SegDecNet architektūra ir pasiekti geresni rezultatai.

Darbo metodikos:

1. Buvo analizuojamos įvairios enkoderių ir dekoderių architektūros siekiant sužinoti kurios architektūros yra tinkamiausios plieno defektų segmentavimui.
2. Buvo naudojami iš anksto apmokyti (angl. *pre-trained*) modeliai ir perkeltinis mokymas (angl. *transfer learning*), pasitelkiant PyTorch segmentavimo modelių biblioteka.
3. Buvo analizuojamas SegDecNet architektūros kodas siekiant rasti kaip būtų galima pagerinti šios architektūros gautus rezultatus.

Šaltiniai:

1. „Kaggle“ konkurso dalyvių sprendimų kodas, kur buvo pristatytas duomenų rinkinys.
2. Moksliniai straipsniai, kuriuose aprašomos įvairios architektūros, kurios bus naudojamos šiame darbe.
3. „Tensorflow“, „PyTorch“ ir segmentavimo modelių dokumentacija.
4. Mokslinis straipsnis kuriame pristatoma SegDecNet architektūra.

Temos analizės tvarka:

Pirma atliekama analizė įvairių neuroninių tinklų architektūrų ir taip pat SegDecNet architektūros bei mokslinio straipsnio, kuriame pristatyta ši architektūra, analizė. Nagrinėjama segmentavimas ir metrikos padėsiančios įvertinti architektūrų segmentavimo rezultatus.

Darbo atlikimo aplinkybės:

Darbas buvo atliktas naudojantis asmeniniu kompiuteriu, „Kaggle“ platforma, „Google Colaboratory“ platforma ir Vilniaus Universiteto MIF superkompiuteriu.

Naudoti programavimo įrankiai:

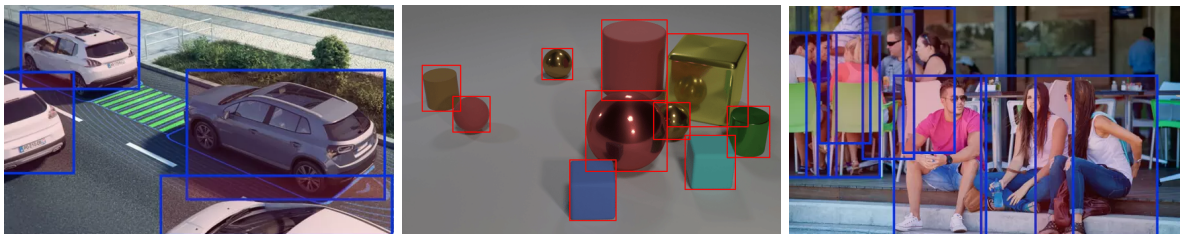
1. „Python“ programavimo kalba.
2. „Tensorflow“ ir „PyTorch“ bibliotekos.
3. „Pip“ bibliotekų valdymo sistemos.
4. „Kaggle“ platforma.
5. „Google Colaboratory“ platforma.

1. Literatūros analizė

Kompiuterinė rega ir vaizdo atpažinimas yra svarbios ir labai sparčiai populiarėjančios sritys technologijų srityje. Šios sritys siekia suteikti kompiuteriams galimybę suprasti ir analizuoti vaizdinę informaciją taip, kaip tai daro žmogaus rega. Tam reikalingos įvairios technologijos ir algoritmai, leidžiantys kompiuteriams atpažinti objektus, tekstūras, raides ir skaičius vaizduose, taip pat suprasti ir interpretuoti vaizdinius įvykius. Kompiuterinė rega ir vaizdo atpažinimas turi svarbių pritaikymų įvairiose srityse, pavyzdžiui, automobilių vairavime, robotikoje, kokybės kontrolėje ir medicinoje. Kompiuterinė rega ir vaizdo atpažinimas yra sparčiai vystomos ir plačiai naudojamos technologijos.

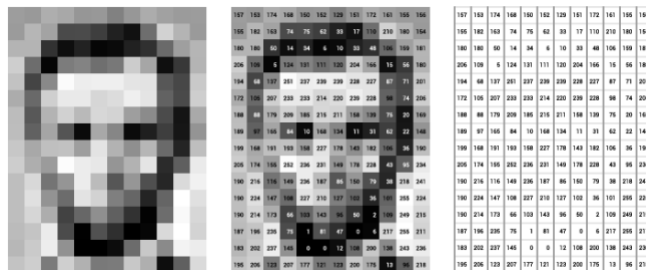
1.1. Vaizdo atpažinimas

Vaizdo atpažinimas yra metodas, naudojamas atpažinti objektus vaizduose ir juos grupuoti pagal specifines kategorijas, priklausomai nuo to kaip žmonės atpažįsta tuos objektus.



1 pav. Vaizdo atpažinimo panaudojimas: a) automobilių aptikimas, b) įvairių objektų aptikimas, c) žmonių aptikimas [Pot22].

Vaizdo atpažinimas žmonėms veikia pasąmoningai. Bet tai yra kur kas labiau komplikauta kai vaizdo atpažinimą turi atlikti kompiuteris. Kompiuteriui, paveikslėlis yra sudarytas iš duomenų. Tai yra masyvas, sudarytas iš pikselio verčių. Pikselis yra skaitmeninio atvaizdavimo elementas, apibūdinantis tam tikros vaizdo srities ryškumą. Pikseliai gali būti vieno arba kelių kanalų. Juodai baltos nuotraukos turi tik vieną kanalą, kuriame kinta pikselių intensyvumas nuo juodos iki baltos spalvos, o spalvotos nuotraukos turi kelis kanalus, kuriems varijuojant nuo šviesiausių iki tamsiausių atspalvių, susidaro spalvotas vaizdas.



2 pav. Kompiuteris mato paveikslėlį kaip skaičių masyvą [Bah23].

Kompiuteriams būtent ir yra svarbu pikselių vertės. Jie paprasčiausiai lygina pikselių sekas su kitomis sekomis, jei sekos yra panašios, tada kompiuteris susieja šias dvi sekas ir patalpina į tą

pačią kategoriją. Retai kada programa atpažins paveikslėlį 100 %, nes pikselių sekos labai retai kada būna visiškai panašios. Šiame darbe būtent vaizdo atpažinimas yra panaudojamas plieno paviršiaus defektams segmentuoti.

1.2. Vaizdų segmentavimas

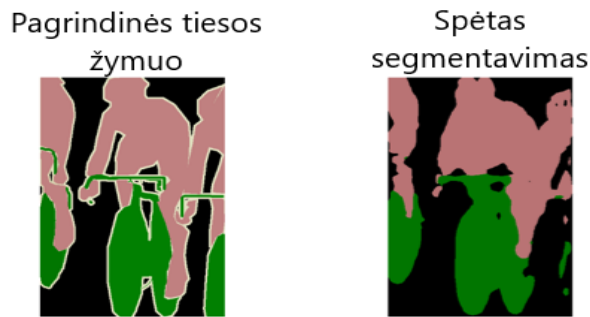
Vaizdų segmentavimas yra vienas iš pagrindinių kompiuterinės regos uždavinių, kai vaizdas padalijamas į atskirus ir reikšmingus regionus arba segmentus. Tikslas – kiekvienam pikseliui ar pikselių grupei priskirti žymenį (angl. *label*), nurodantį segmentą, kuriam jis priklauso. Šis procesas leidžia kompiuteriams išsamiau ir tiksliau analizuoti ir suprasti vaizdo turinį. Vaizdų segmentavimas plačiai taikomas įvairiose srityse, pavyzdžiui, objektų atpažinimo, vaizdų redagavimo, medicininių vaizdų, autonominio vairavimo ir vaizdo stebėjimo srityse.

Vaizdo segmentavimo procesas apima kelis etapus. Iš pradžių įvesties vaizdas iš anksto apdorojamas, kad būtų pagerinta jo kokybė ir jis būtų paruoštas segmentavimui. Segmentavimo tikslumui pagerinti gali būti taikomi įvairūs metodai, pavyzdžiui, triukšmo mažinimas, kontrasto koregavimas ir spalvų normalizavimas. Paruošus vaizdą, segmentavimui galima naudoti įvairius algoritmus ir metodus.

Egzistuoja kelios vaizdų segmentavimo metodų rūšys. Slenksčių nustatymas (angl. *thresholding*) yra paprastas, bet veiksmingas metodas, kuriuo pikseliai priskiriami skirtingiems segmentams pagal nustatytą slenksčio vertę. Jis veikia lyginant pikselių vertes su slenkstine verte ir atitinkamai juos atskiriant. Kitas metodas – segmentavimas pagal kraštus (angl. *edge-based*, kai nustatomos ribos arba kraštai tarp skirtingų regionų, aptinkant reikšmingus pikselių intensyvumo arba gradientų pokyčius. Regionais pagrįsti (angl. *Region-based*) metodai grupuoja panašius požymius turinčius pikselius ir sudaro segmentus pagal tokius požymius kaip spalva, tekstūra ar forma. Šiems metodams dažnai naudojami klasterizavimo algoritmai arba grafais grindžiami metodai panašioms pikseliams grupuoti.

Semantinis segmentavimas – tai vaizdo segmentavimo rūšis, kai kiekvienam vaizdo pikseliui priskiriama semantinę žymenį ir taip suteikiama galimybė išsamiai suprasti scenos turinį. Skirtingai nuo kitų segmentavimo metodų, kuriais objektai atskiriami tik pagal jų vizualinius požymius, semantinis segmentavimas kiekvieną pikselį priskiria konkrečiai objekto kategorijai arba klasei. Tai leidžia atlikti smulkesnę vaizdo analizę, todėl kompiuteriai gali suvokti skirtingų objektų kontekstą ir ryšius.

Semantinio segmentavimo procese paprastai naudojami gilaus mokymosi modeliai, ypač konvoliuciniai neuroniniai tinklai, toliau CNN. Šie modeliai mokomi naudojant didelius pažymėtų duomenų rinkinius, kuriuose kiekvienas vaizdo pikselis rankiniu būdu pažymimas atitinkama klasės žymenį. Mokymo metu CNN išmoksta iš įvesties vaizdo išgauti aukšto lygio požymius ir sukuria pikselių prognozavimo žemėlapi, kuriame nurodoma tikimybė, kad kiekvienas pikselis priklauso tam tikrai klasei.



3 pav. Semantinio segmentavimo pavyzdys [Jor18].

Vaizdų segmentavimas svarbus tuo, kad iš vaizdų galima išgauti tikslią informaciją, leidžiančią kompiuteriams suprasti ir interpretuoti vaizdinius duomenis. Jis atlieka lemiamą vaidmenį objektų atpažinime, kai segmentuotus objektus galima tiksliai atpažinti ir klasifikuoti. Medicininuose vaizduose segmentavimas padeda nustatyti ir apibrėžti organus ar anomalijas, padeda diagnozuoti ir planuoti gydymą. Segmentavimas naudingas vaizdų redagavimo programoms, nes leidžia tikslingai keisti ar tobulinti tam tikrus vaizdo regionus. Taip pat vaizdų segmentavimas svarbus pramonėje aptikti defektus medžiagų paviršiuose.

Apibendrinant galima teigti, kad vaizdo segmentavimas – tai vaizdo padalijimo į reikšmingus regionus procesas, kuris atlieka svarbų vaidmenį įvairiose kompiuterinės regos programose. Vaizdo segmentavimas leidžia atlikti išsamią analizę, atpažinti objektus ir tiksliai suprasti vaizdo turinį. Segmentavimo svarba yra tai, kad leidžia kompiuteriams efektyviau suvokti ir interpretuoti vaizdus.

1.3. Plienas

Plienas yra tvirta ir ilgaamžė medžiaga, plačiai naudojama statybų, automobilių ir kitose pramonės šakose. Plienas gaminamas krosnyje lydant geležies rūdą ir kitas medžiagas, o paskui pridant anglies ir kitų elementų, kad būtų gautas norimas lydinys. Tada mišinys pilamas į formas, iš kurių formuojami įvairių formų lakštai, plokštės ar strypai.

Plienas, palyginti su kitomis medžiagomis, turi daug privalumų. Jis tvirtas, ilgaamžis, atsparus korozijai ir nusidėvėjimui. Be to, jis yra palyginti nebrangus ir lengvai pagaminamas, todėl yra populiarus pasirinkimas daugeliui paskirčių. Tačiau, kaip ir bet kuri kita medžiaga, plienas gali turėti defektų, kurie gali turėti įtakos jo veikimui ir patikimumui. Kai kurie dažniausiai pasitaikantys plieno defektai yra įtrūkimai, intarpai ir paviršiaus defektai. Plieno įtrūkimų gali atsirasti dėl įtempimo, nuovargio arba korozijos. Šie defektai gali susilpninti plieną ir padaryti jį labiau linkusį į gedimus. Įterpimai – tai priemaišos arba pašalinės medžiagos, kurios gamybos proceso metu patenka į plieną. Šie defektai taip pat gali susilpninti plieną ir padidinti jo gedimo tikimybę. Plieno paviršiaus defektai gali būti įbrėžimai, įlenkimai ir įskilimai. Šie defektai gali turėti įtakos plieno išvaizdai, taip pat gali susilpninti jo paviršių, dėl to jis tampa jautresnis korozijai ir nusidėvėjimui. Plieno defektai gali atsirasti dėl įvairių veiksnių, įskaitant netinkamus gamybos procesus, nekokybiškų žaliavų naudojimą ir prastų aplinkos sąlygų poveikį.

Paprastai plieno gaminių kokybės tikrinimas atliekamas rankiniu būdu. Plieno defektų

tikrinimas rankiniu būdu gali lemti gamybos proceso vėlavimą, be to, tai nėra veiksmingas būdas garantuoti, kad plienas bus be defektų. Plieno defekto nustatymo tikslumas gali priklausyti nuo tikrintojų patirties. Plieno defektų tikrinimas rankiniu būdu yra varginantis ir daug laiko reikalaujantis procesas, kai inspektoriai turi rankiniu būdu patikrinti kiekvieną plieno gaminį. Vieneto per valandą rodiklis, kuriuo inspektoriai aptinka plieno defektus, yra mažas. Paprastai plieno gaminių kokybės tikrinimas atliekamas rankiniu būdu. Plieno defektų tikrinimas rankiniu būdu gali lemti gamybos proceso vėlavimą, be to, tai nėra veiksmingas būdas garantuoti, kad plienas bus be defektų. Plieno defekto nustatymo tikslumas gali priklausyti nuo tikrintojų patirties. Plieno defektų tikrinimas rankiniu būdu yra varginantis ir daug laiko reikalaujantis procesas, kai inspektoriai turi rankiniu būdu patikrinti kiekvieną plieno gaminį. Vieneto per valandą rodiklis, kuriuo inspektoriai aptinka plieno defektus, yra mažas. Dėl vėlavimo sumažėja gamybos sparta, todėl gamybos įmonės patiria didelių ekonominių nuostolių. Be to, jei neaptikti plieno defektai pateks galutiniams naudotojams ar klientams, tai gali lemti blogą bendrovės reputaciją [AAL⁺21].

1.4. Enkoderiai ir Dekoderiai

Semantiniam segmentavimui naudojamoje CNN architektūroje dažnai naudojama enkoderio ir dekoderio (angl. *encoder-decoder*) struktūra. Enkoderio komponentas palaipsniui mažina įvesties vaizdo erdvinius matmenis, išsaugodamas esminius požymius. Šis matmenų mažinimas leidžia tinklui užfiksuoti daugiau globalios informacijos. Tuomet dekoderio komponentas išplečia užkoduotus požymius, palaipsniui atkuria erdvinę raišką ir sukuria galutinį segmentavimo žemėlapi.

Vaizdų segmentavimo modeliuose dekoderių ir enkoderių vaidmenys yra labai svarbūs norint užfiksuoti ir atvaizduoti tiek aukšto, tiek žemo lygio įvesties vaizdo požymius. Šie komponentai veikia kartu, kad būtų išgauta reikšminga informacija ir sukurtos tikslios pikselių prognozės.

Enkoderis yra atsakingas už aukšto lygmens požymių išgavimą iš įvesties vaizdo, palaipsniui mažinant jo erdvinius matmenis. Jis apdoroja vaizdą naudodamas keletą konvoliucinių sluoksnių, sutelkimo operacijų ir netiesinių aktyvacijos funkcijų. Tokiu būdu enkoderis išmoksta užfiksuoti svarbias vaizdines struktūras kurios yra svarbios segmentavimo užduočiai. Sumažinus erdvinius matmenis enkoderis gali surinkti daugiau globalios informacijos apie vaizdą. Tai padeda suprasti kontekstą ir bendrą vaizdo struktūrą.

Dekoderio paima užkoduotus požymius ir atkuria pradinę erdvinę raišką. Jis atlieka išplėtimo operacijas, kad palaipsniui padidintų užkoduotų požymių erdvinius matmenis. Tikslas yra sukurti pikselio dydžio tikimybių žemėlapi, kuris atitiktų įvesties vaizdo matmenis. Dekoderis tai pasiekia naudodamas išmokus aukšto lygio požymius kartu su atkurta erdvine informacija. Šis procesas padeda užfiksuoti smulkias detales ir vietinį kontekstą, kuris galėjo būti prarastas enkoderio etape.

Apibendrinant, enkoderiai ir dekoderiai atlieka esminius vaidmenis vaizdų segmentavimo modeliuose. Enkoderis išgauna aukšto lygio požymius, kartu sumažindamas erdvinius matmenis, kad būtų užfiksuotas visuotinis kontekstas. Dekoderis savo ruožtu atkuria erdvinę rezoliuciją ir generuoja pikselio dydžio spėjimus, į kuriuos įtraukiama ir aukšto, ir žemo lygio požymius. Siekiant sukurti veiksmingas ir efektyvias enkoderio ir dekoderio struktūras, kurios leistų tiksliai ir išsamiai segmentuoti vaizdą, naudojamos įvairios architektūros ir metodai.

1.5. Enkoderio ir dekoderio architektūros

Vaizdų segmentavimo užduočiai spręsti buvo sukurtos įvairios enkoderio ir dekoderio architektūros, kurios turi savų privalumų ir savybių. Šiomis architektūromis siekiama iš įvesties vaizdo surinkti kontekstinę informaciją ir aukšto lygio požymius, kad būtų galima atlikti tikslų segmentavimą. Tokių architektūrų pavyzdžiai: U-Net, DeepLab, FPN, PSPNet, PAN, LinkNet ir kt. Šiuose modeliuose naudojami praleidimo jungtys (angl. *skip connections*), siekiant integruoti informaciją iš kelių mastelių ir palengvinti tikslų aptikimą vaizduose. Pasinaudodamos enkoderio ir dekoderio struktūra, šios architektūros generuoja pikselių prognozes, kiekvienam pikseliui priskirdamos žymenis pagal jo semantinę reikšmę. Architektūros pasirinkimas priklauso nuo konkrečios užduoties reikalavimų, skaičiavimo efektyvumo ir turimų resursų.

1.5.1. Unet

U-Net architektūra yra plačiai naudojama enkoderio ir dekoderio architektūra vaizdų segmentavimo srityje. Ją 2015 m. pristatė Ronneberger et al. Pristatyta U-Net architektūra sukėlė esminį pokytį semantinio segmentavimo srityje dėl kodavimo ir dekodavimo struktūros ir praleidimo jungčių. Jos unikalus konstrukcinis pasirinkimas apima modifikuotos VGG-16 architektūros naudojimą kaip enkoderį, kuris leidžia išgauti požymius iš įvesties. U-Net plačiai taikomas medicininių vaizdų segmentavimo uždaviniuose, pavyzdžiui, auglių segmentavimui ir ląstelių segmentavimui. Šios architektūros privalumai yra tikslus lokalizavimas naudojant praleidimo jungtis, efektyvus parametrų naudojimas ir gebėjimas dirbti su ribotais mokymo duomenimis [RFB15].

1.5.2. DeepLab

Chen et al. 2017 m. pristatytame DeepLab architektūroje, kuri turėjo kelis patobulinimus per metus, semantinei segmentacijai buvo naudojamos praretintos konvoliucijos operacijos ir pilnai sujungtų sąryšių sąlyginiai atsitiktiniai laukai (angl. *fully-connected Conditional Random Field (CRF)*). Jos pagrindinis architektūros pasirinkimas yra išsiplėtusių konvoliucijų naudojimas, kurios leidžia užfiksuoti kelių mastelių informaciją per daug nepadidinant skaičiavimo sudėtingumo. DeepLab puikiai fiksuoja smulkias detales ir nustato tikslas objektų ribas. Ji plačiai naudojama įvairiose srityse, įskaitant miesto scenų supratimą, palydovinių vaizdų analizę ir medicininius vaizdus. DeepLab privalumai – išplėstinių konvoliucijų ir CRF panaudojimas siekiant tikslų ir išsamių segmentavimo rezultatų [CPS⁺17].

1.5.3. FPN ir PAN

Lin et al. 2017 m. pasiūlyta požymių piramidės tinklo (angl. *Feature Pyramid Network, FPN*) architektūra sprendžia daugiamatį požymių atvaizdavimo problemą segmentavimo modeliuose. FPN naudoja iš viršaus į apačią ir šoninį sujungimą, kad sujungtų požymius iš skirtingų požymių piramidės lygių. Šios architektūros savotiškumas yra kontekstinės informacijos sujungime ir erdvių detalių išsaugojime. FPN plačiai naudojamas objektų aptikimo ir segmentavimo uždaviniuose, nes leidžia tiksliai nustatyti vietą ir atpažinti įvairaus mastelio objektus [LDG⁺17]. FPN privalumai –

gebėjimas tiksliai segmentuoti skirtingo mastelio objektus. Yan et al. 2018 metais pasiūlyta Path Aggregation Network (PAN) architektūra remiasi požymių piramidės tinklo (FPN) architektūra, įtraukdama papildomus šoninius ir iš viršaus į apačią sujungimus. Ši architektūra yra panaši, tačiau patobulinta FPN versija [LQQ⁺18].

1.5.4. PSPNet

Zhao et al. 2017 metais pristatytame piramidės scenos analizės tinkle (angl. *Pyramid Scene Parsing Network*) (PSPNet) yra piramidės sutelkimo modulis, kuris leidžia veiksmingai suprasti vaizdą platesniu mastu. PSPNet suskirsto požymių žemėlapi į nepersidengiančius regionus ir fiksuoja globalų kontekstą kiekvienoje srityje keliais lygmenimis. Šios architektūros unikalumas yra kelių lygmenų kontekstinės informacijos integravimas į dekodierio procesus. PSPNet puikiai fiksuoja platų kontekstą ir buvo taikomas tokioms užduotims, kaip miesto scenų supratimas ir palydovinių vaizdų analizė [KRS⁺21]. PSPNet privalumai – gebėjimas kurti tikslus segmentus, visapusiškai suprantant sceną [ZSQ⁺17].

1.5.5. Linknet

Chaurasia et al. 2017 m. pristatytame LinkNet tinkle daugiausia dėmesio skiriama segmentavimo modelių tikslumo ir skaičiavimo efektyvumo pusiausvyrai pasiekti. Ši siūloma giliojo neuroninio tinklo architektūra bando efektyviai pasidalinti enkoderio įgyta informacija su dekodieriu po kiekvieno mažesnio imties mažinimo bloko. Šis požymių persiuntimo metodas ne tik leidžia pasiekti didesnius tikslumo rezultatus, bet ir leidžia turėti nedaug dekodierio parametrų. Šios architektūros privalumai – efektyvumas, tikslus segmentavimas naudojant praleidimo jungtis ir esant ribotiems ištekliams [CC17].

1.5.6. MAnet

Fan et al. 2020 m. pristatytame MAnet (angl. *Multi-scale Attention Net*) pateikiami keli unikalūs dizaino pasirinkimai. Naudojant šią architektūrą galima užfiksuoti daugybę kontekstinių priklausomybių pasitelkiant dėmesio sutelkimo mechanizmu naudojant du blokus: padėties dėmesio bloką (angl. *Position-wise Attention Block*)(PAB), kuris fiksuoja pikselių erdvines priklausomybes bendrame vaizde ir bendro dėmesio sutelkimo apjungimo bloką (angl. *Mutual-attentional fusion block*) [FWL⁺20].

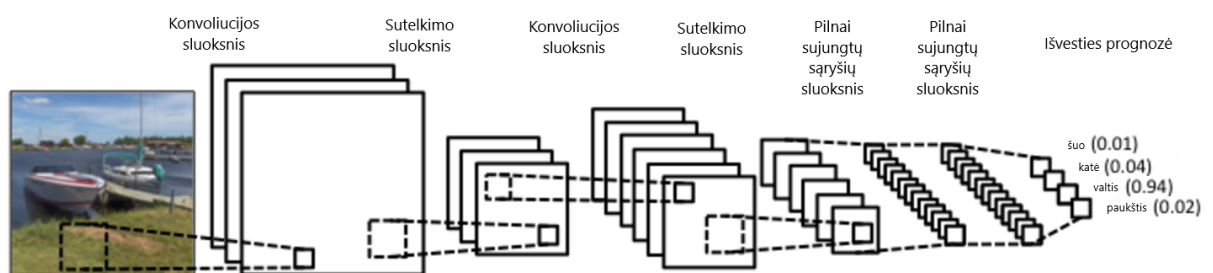
1.6. Konvoliuciniai neuroniniai tinklai

Konvoliuciniai neuroniniai tinklai (angl. *Convolutional Neural Networks*), yra gilaus mokymosi modelis, skirtas apdoroti duomenis, turinčius tinklėlio struktūrą (angl. *Grid Pattern*), pavyzdžiui, vaizdus, ir skirtas automatiškai ir adaptyviai mokytis erdvinių požymių hierarchijų, pradedant žemo ir baigiant aukšto lygio požymiais. CNN paprastai sudaro trijų tipų sluoksniai (arba statybiniai blokai): konvoliucijos, sutelkimo ir visiškai sujungti sluoksniai. Pirmieji du – konvoliucijos ir sutelkimo sluoksniai – atlieka požymių išskyrimą, o trečiasis – visiškai sujungtas

sluoksnis – išskirtus požymius perkelia į galutinę išvestį, pavyzdžiui, klasifikaciją. Konvoliucijos sluoksnis atlieka pagrindinį vaidmenį CNN, kurį sudaro matematinių operacijų rinkinys, pavyzdžiui, konvoliucija – specializuota tiesinės operacijos rūšis. Skaitmeniniuose vaizduose pikselių reikšmės saugomos skaičių masyve (2 pav.), o kiekvienoje vaizdo vietoje taikomas nedidelis parametrų tinklėlis, vadinamas filtro branduoliu. Filtro branduolys – tai optimizuotas požymių išskyrimo būdas, todėl CNN labai efektyviai apdoroja vaizdus, nes požymis gali atsirasti bet kurioje vaizdo vietoje. Kai vienas sluoksnis perduoda savo išvestį kitam sluoksniui, išgauti požymiai gali būti hierarchizuoti ir palaipsniui tapti sudėtingesni. Parametrų, pavyzdžiui, branduolių, optimizavimo procesas vadinamas mokymu, kuris vykdomas taip, kad būtų sumažintas skirtumas tarp išvesties ir realių duomenų, taikant optimizavimo algoritmus, tokius kaip atgalinio sklido ar gradientinio nusileidimo algoritmais [YND⁺18].

1.6.1. CNN struktūra

CNN architektūrą sudaro kelios sudedamosios dalys, tokios kaip konvoliucijos sluoksniai, sutelkimo sluoksniai ir pilnai sujungtų sąryšių sluoksniai. Tipinę architektūrą sudaro pakartojimai iš kelių konvoliucinių sluoksnių ir sutelkimo sluoksnio, po kurių seka vienas ar daugiau pilnai sujungtų sąryšių sluoksnių. Žingsnis, kuriame įvesties duomenys per šiuos sluoksnius paverčiami išvesties duomenimis, vadinamas tiesioginiu sklidimu [Bri15] (4 pav.).

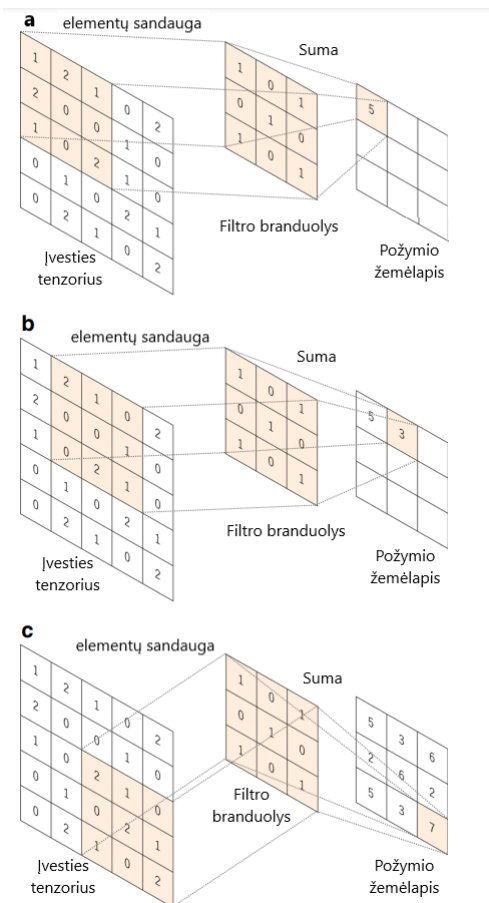


4 pav. Konvoliucinio neuroninio tinklo struktūra [Bri15].

Konvoliucinio neuroninio tinklo sluoksnių veikimo principai:

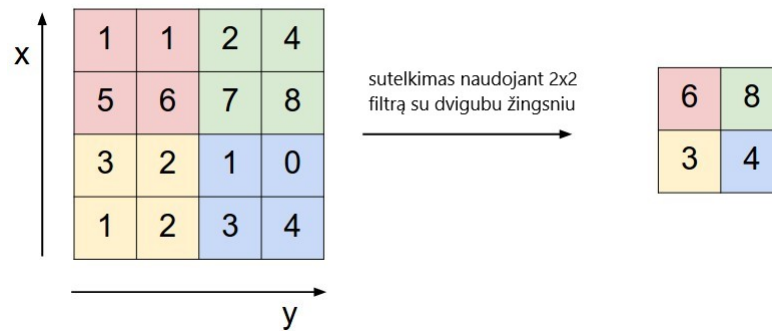
- Konvoliuciniai sluoksniai yra pagrindiniai CNN blokai. Šie sluoksniai yra atsakingi už tokių požymių, kaip briaunos, tekstūros ir formos, išskyrimą iš įvesties duomenų. Jie tai daro įvesties duomenims taikydami išmoktų filtrų rinkinį, kuris nuskaito duomenis ir aptinka tam tikrus požymius. Konvoliucinių sluoksnių išvestis yra požymių žemėlapių rinkinys, kuriame aptikti požymiai pateikiami kompaktiškesne forma. Konvoliucija – tai specializuota tiesinė operacija, naudojama požymių išskyrimui, kai nedidelė skaičių aibė, vadinama filtro branduoliu, taikoma įvesties duomenims, t. y. skaičių aibe, vadinamai tenzoriumi. Kiekvienoje tenzoriaus vietoje apskaičiuojama kiekvieno branduolio elemento ir įvesties tenzoriaus elementų sandauga ir sumuojama, kad atitinkamoje išvesties tenzoriaus vietoje būtų gauta išvesties reikšmė, vadinama požymių žemėlapiu (5a–c pav.). Ši procedūra kartojama taikant keletą filtro branduolių, kad būtų sudarytas bet koks kiekis požymių

žemėlapių, atspindinčių skirtingas įvesties tenzorių charakteristikas. Taigi skirtingus branduolius galima laikyti skirtingais požymių išskyrėjais. Du pagrindiniai hiperparametrai, apibrėžiantys konvoliucijos operaciją, yra dydis ir branduolių skaičius. Dydis paprastai būna 3×3 , bet gali būti ir 5×5 ar 7×7 . Branduolių skaičius yra laisvai pasirenkamas ir lemia išvesties požymių žemėlapių gylį [YND⁺18].



5 pav. Konvoliucijos operacijos pavyzdys, kai branduolio dydis yra 3×3 , be pildymo nuliais ir su vienetiniu žingsniu [YND⁺18].

- Sutelkimo sluoksniai (angl. *Pooling layers*) naudojami duomenų dydžiui sumažinti, tam tikru būdu sujungiant konvoliucinių sluoksnių išvestį. Yra kelios jungimo rūšys: jungimas imant maksimalią reikšmę ir jungimas vidurkinant. Jungimas imant maksimalią reikšmę iš gretimų reikšmių grupės paima didžiausią reikšmę, o jungimas vidurkinant – vidutinę reikšmę. Grupavimas padeda sumažinti tinklo skaičiavimo naštą ir taip pat padeda konvoliucinių sluoksnių aptiktus požymius padaryti patikimesnius. Naudojant maksimalų sujungimą, tinklas greičiau išmokinamas ir tiksliau sprendžia atpažinimo uždavinius [YND⁺18]. Sutelkimo operacija imant maksimalią reikšmę pavaizduota 6 paveiksle.



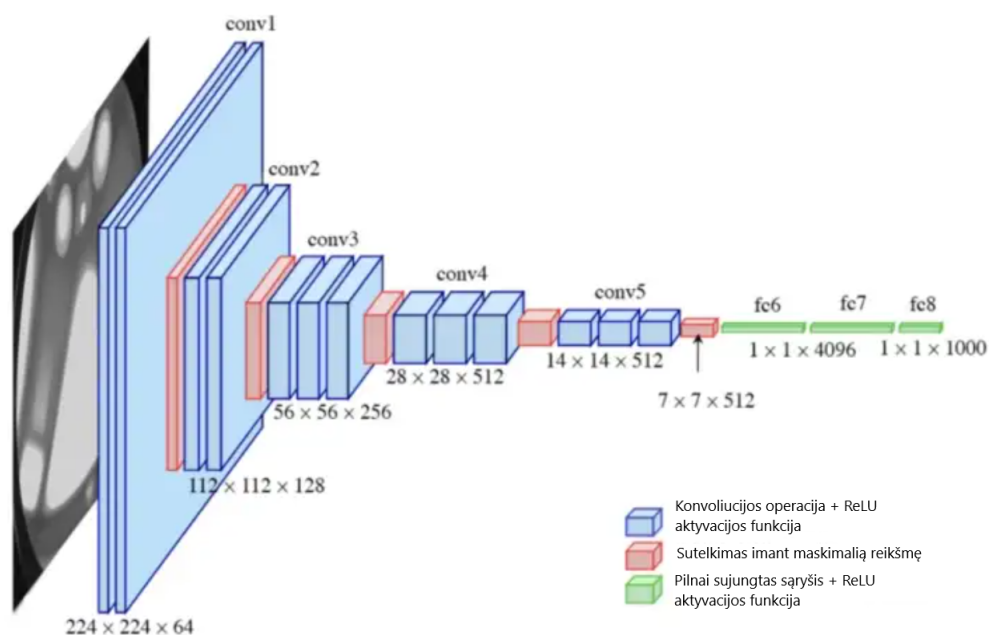
6 pav. Sutelkimo operacija imant maksimalią reikšmę.

- Pilnai sujungtų sąryšių sluoksniai (angl. *Fully Connected Layer*), dar vadinami tankiaisiais sluoksniais, sujungia konvoliucinių ir sutelkimo sluoksnių išskirtus požymius ir naudoja juos prognozėms arba sprendimams priimti. Šie sluoksniai vadinami visiškai sujungtais, nes visi vieno sluoksnio neuronai yra sujungti su visais kito sluoksnio neuronais. Visiškai sujungtų sluoksnių išvestis yra tikimybių arba klasių prognozių rinkinys, rodantis tikimybę, kad įvesties duomenys priklauso tam tikrai klasei [YND⁺18].

1.7. CNN architektūros

1.7.1. VGG16 architektūra

VGG16 architektūra pirmą kartą buvo pristatyta 2014 metais. VGG16 yra CNN architektūra skirta vaizdų klasifikavimo uždavimams. Ši architektūra išsiskiria tuo, kad yra gana paprasta, tačiau veiksminga. VGG16 architektūra yra dažnai naudojama kaip bazinis modelis atliekant testavimus vaizdų klasifikavime. VGG16 architektūrą sudaro 16 sluoksnių. 13 iš jų yra konvoliucinių sluoksniai ir 3 pilnai sujungtų sąryšių sluoksniai. VGG16 naudojama paprasta konstrukcija, pagal kurią naudojami keli maži konvoliuciniai filtrai (3x3) ir sudedami vienas po kito, o siekiant sumažinti požymių žemėlapių erdvinę rezoliuciją, naudojami maksimaliai sujungti sluoksniai. Galutinis VGG16 modelio išvesties rezultatas yra klasifikavimo prognozė, pagrįsta iš įvesties vaizdo išskirtais požymiais [SZ14].

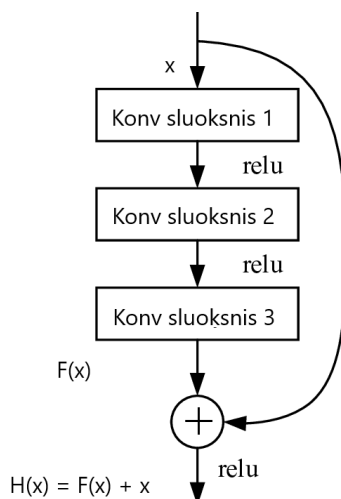


7 pav. VGG16 architektūra. ReLU aktyvacijos funkcija: $f(x) = \max(0, x)$ [Le21].

1.7.2. ResNet architektūra

ResNet yra CNN architektūra, kurią 2015 m. sukūrė „Microsoft Research“, skirta vaizdų klasifikavimo užduotims. Ši architektūra išsiskiria tuo, kad yra daug gilesnė už daugumą kitų tuo metu sukurtų CNN. Taip pat ši architektūra išsprendžia nykstančio gradiento (angl. vanishing gradient) problemą, naudodama praleidžiamąsias jungtis (angl. skip connection). Ši problema būdinga tinklams kurie yra per gilūs ir vėlesniuose sluoksniuose tinklas tiesiog nebesimoko [HZR⁺16].

Pagrindinis ResNet tinklo blokas yra liekanų blokas (angl. residual block), kurį sudaro du arba trys konvoliuciniai sluoksniai, su papildoma galimybe pridėti praleidžiamąją jungtį (8 pav.). Praleidžiamoji jungtis apeina konvoliucinius sluoksnius ir leidžia įvesties reikšmę tiesiogiai pridėti prie bloko išvesties, iš esmės sukurdamą liekaną, kurią išmoksta tinklas. Tai leidžia tinklui išmokti įvesties liekanas arba pataisas, o ne bandyti išmokti visą požymių žemėlapi nuo įvesties iki išvesties [HZR⁺16].



8 pav. Liekanų bloko architektūra [ZYW⁺19].

ResNet34 ir ResNet101 yra ResNet architektūros variantai, kurie skiriasi savo sluoksnių skaičiumi. ResNet34 turi 34 sluoksnius, o ResNet101 – 101 sluoksnį. Tai reiškia, kad ResNet101 yra gilesnis ir potencialiai galingesnis nei ResNet34, tačiau jam taip pat reikia daugiau skaičiavimų ir jis gali būti labiau linkęs į persimokymą.

SeResNet34 yra dar vienas ResNet architektūros variantas. Jis panašus į ResNet34 savo sluoksnių skaičiumi, tačiau jame naudojamas kitokio tipo liekanų blokas, vadinamas suspaudimo ir sužadavimo (angl. squeeze-and-excitation) bloku. Suspaudimo ir sužadavimo blokas naudoja visuotinį sutelkimo vidurkinant sluoksnį, kad apskaičiuotų kanalo parametrus, kurie vėliau naudojami konvoliucinių sluoksnių išvesties masteliams pakeisti. Tai gali pagerinti modelio gebėjimą sutelkti dėmesį į svarbius požymius ir pagerinti jo rezultatus sprendžiant vaizdų klasifikavimo užduotį [HSS18].

Apibendrinant, ResNet34, ResNet101 ir SeResNet34 yra ResNet architektūros variantai, skirti vaizdų klasifikavimo uždaviniams spręsti. ResNet34 ir ResNet101 skiriasi savo sluoksnių skaičiumi: ResNet34 turi 34 sluoksnius, o ResNet101 – 101 sluoksnį. Abu jie, kaip ir VGG16, paprastai naudojami kaip baziniai modeliai vaizdų klasifikavimo užduotyse. SeResNet34 yra panašus į ResNet34, nes turi tą patį sluoksnių skaičių, tačiau turi kitokį likutinį bloką.

1.7.3. Inception architektūra

Inception architektūra yra CNN architektūra, kurią sukūrė „Google“. Ji buvo pristatyta 2014 m [SLJ⁺15].

Inception architektūra žinoma dėl to, kad joje naudojami pradžios (angl. Inception) moduliai, t. y. statybiniai blokai, kuriuos sudaro keletas lygiagrečių konvoliucinių sluoksnių su skirtingo dydžio filtro branduoliais ir jų išvestis sujungiantis sujungimo sluoksniu. Pradinio modulio idėja – fiksuoti skirtingo mastelio požymius ir juos sujungti, kad tinklas iš įvesties vaizdo galėtų išmokti įvairesnių ir sudėtingesnių požymių [SLJ⁺15].

Inceptionv3 yra Inception architektūros variantas, kuris buvo pristatytas 2015 m. Inceptionv3 yra patobulinta originalioji Inception architektūra, joje yra keletas pakeitimų, pavyzdžiui, mažesnio dydžio ir asimetrinės konvoliucijos, taip pat įtraukti papildomi inception moduliai ir pagalbiniai klasifikatoriai, kad būtų pagerintas tinklo mokymas ir sumažinta persimokymo tikimybė [SVI⁺16].

Apibendrinant, Inception architektūra yra CNN architektūra, žinoma dėl to, kad joje naudojami inception moduliai, kurie fiksuoja skirtingo mastelio požymius ir juos sujungia. Inceptionv3 yra Inception architektūros variantas, kuriame yra keletas patobulinimų, palyginti su pradine architektūra.

1.7.4. MobileNet architektūra

MobileNet yra CNN architektūra, kurią „Google“ sukūrė mobiliesiems ir integruotiems įrenginiams su ribotais skaičiavimo ištekliais. MobileNet sukurtas taip, kad būtų efektyvus, turėtų mažesnę parametrų skaičių ir mažesnius skaičiavimo reikalavimus nei kitos CNN architektūros. Šį efektyvumą ji pasiekia naudodama pagal gylį konvoliucijas, kurios tradicines konvoliucijas išskaido į pagal gylį ir taškinę konvoliuciją [HZC⁺17].

Pagal gylį konvoliucijoje kiekvienas filtras yra atsakingas už tam tikro požymio mokymąsi iš vieno įvesties kanalo. Pagal gylį atliekamos konvoliucijos išvestis yra požymių žemėlapių rinkinys, po vieną kiekvienam įvesties kanalui. Tada pagal gylį taikomos konvoliucijos išėjimui taikoma taškinė konvoliucija, kad požymių žemėlapiai būtų sujungti į vieną išvesties požymių žemėlapi. Tai leidžia tinklui išmokyti veiksmingesnių filtrų, kurie efektyviau fiksuoja svarbius įvesties paveikslėlio požymius [HZC⁺17].

MobileNetV2 yra patobulinta MobileNet architektūra ir apima keletą pakeitimų, pavyzdžiui, invertuotų liekanų blokų ir tiesinių butelio kaklelių naudojimą, kurie pagerina tinklo efektyvumą [SHZ⁺18].

1.7.5. EfficientNet architektūra

EfficientNet yra CNN architektūra, kurią sukūrė „Google“, kad būtų galima efektyviai ir tiksliai klasifikuoti vaizdus. Ji gana panaši į MobileNet architektūrą – abi jos sukurtos taip, kad būtų efektyvios ir nereikalaujančios daug resursų. Abiejose architektūrose naudojama konvoliucija pagal gylį ir invertuoti liekanų blokai. Esminis skirtumas yra tas, kad EfficientNet architektūroje naudojamas jungtinis mastelio didinimas. EfficientNet sukurta taip, kad būtų galima keisti jos parametrus. Didinant parametrus yra efektyviai plečiamas tinklas. Šis efektyvumas pasiekiamas naudojant jungtinį mastelio didinimą, kuris subalansuotai keičia tinklo matmenis keičiant gylį, plotį ir paveikslėlio raišką. Tai leidžia tinklui išmokyti veiksmingesnius filtrus, kurie efektyviau fiksuoja svarbius įvesties vaizdo požymius [TL19].

EfficientNetB0 ir EfficientNetB2 yra EfficientNet architektūros variantai, kurie skiriasi savo dydžiu. Šiems architektūros variantams pritaikytas skirtingas jungtinis mastelio didinimas. EfficientNetB0 yra mažiausias ir mažiausiai sudėtingas variantas, o EfficientNetB2 yra didesnis, reikalaujantis daugiau resursų, bet taip pat pasiekiantis geresnius rezultatus.

1.7.6. DenseNet architektūra

DenseNet yra CNN architektūra, kurią pristatė Huang et al. 2017 m. Ši architektūra siekia išspręsti nykstančio gradiento problemą kuri atsiranda su CNN. Ši architektūra sprendžia šią problemą sukurdamą ryšius su kiekvienu sluoksniu neuroniniame tinkle. DenseNet architektūroje kiekvienas sluoksnius gauna įvestis iš visų ankstesnių sluoksnių ir perduoda savo požymių žemėlapius kaip įvestis visiems vėlesniems sluoksniams. Toks sluoksnių jungimas leidžia užtikrinti tiesioginį informacijos srautą ir pakartotinai naudoti požymius visame tinkle. DenseNet architektūros pagrindinis komponentas yra tankūs (angl. „dense“) blokai. Šiuose blokuose kiekvieno sluoksnio išvesties požymių žemėlapiai yra sujungiami ir perduodami kaip įvesties duomenys to paties bloko sluoksniams, tai padeda sumažinti nykstančio gradiento problemą.[HLV⁺17]

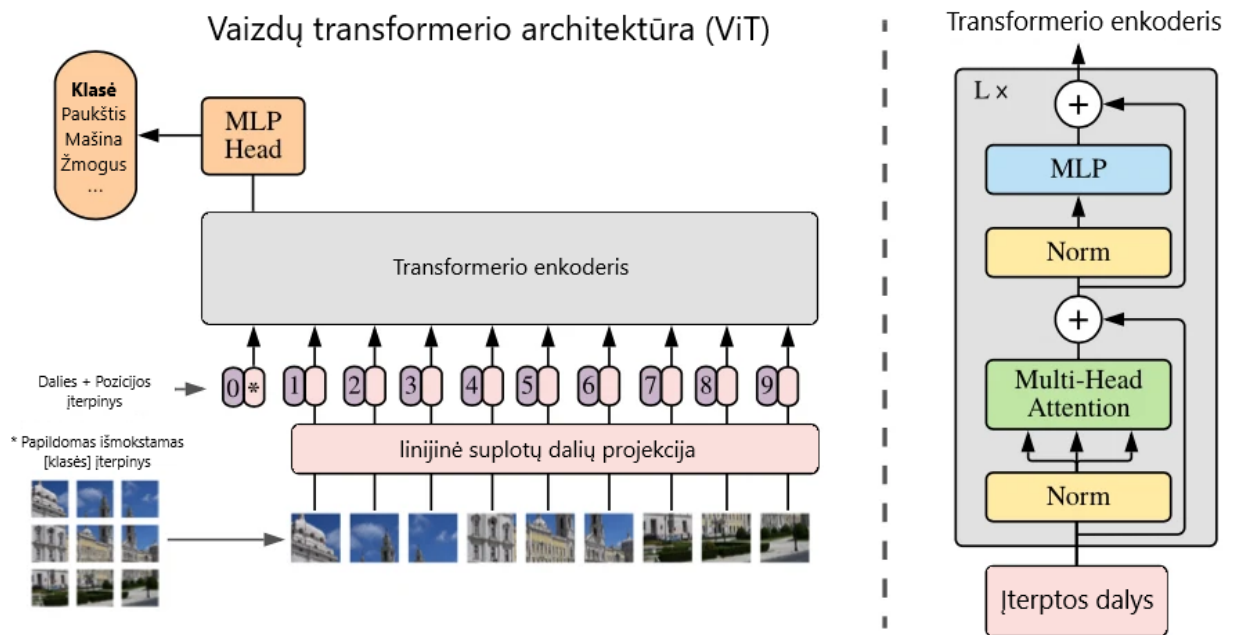
1.7.7. Res2NeXt architektūra

Gao et al. 2019 metais savo moksliniame straipsnyje pristatė Res2NeXt architektūrą [GCZ⁺19]. Ši architektūra yra patobulinta ResNeXt architektūros versija. ResNeXt. Šioje architektūroje yra naudojama nauja dimensija – kardinalumas. Ši dimensija leidžia praplėsti

architektūra nepadidinant skaičiavimo sudėtingumo. ResNeXt, palyginti su ResNet, pasižymėjo geresniu našumu sprendžiant įvairias kompiuterinės regos užduotis, todėl tai yra galinga architektūra tokioms užduotims, kaip vaizdų klasifikavimas, objektų aptikimas ir semantinis segmentavimas, spręsti.[XGD⁺17].

1.8. Vaizdų transformerio modelis

Transformatoriaus architektūra, iš pradžių sukurta skaitmeniniam natūraliosios kalbos apdorojimui [VSP⁺17], buvo sėkmingai pritaikyta kompiuterinės regos sričiai, todėl atsirado vaizdų transformerių (angl. *Visual Transformer*) modelis. Transformerių neuroninio tinklo architektūra yra savaiminiu dėmesiu pagrįsta architektūra. Ši architektūra sugeba įvesties sekoje užfiksuoti globalias priklausomybes. Kompiuterinės regos kontekste transformerių neuroninio tinklo architektūra naudoja dėmesio sutelkimo mechanizmą, kad užfiksuotų tolimąsias priklausomybes vaizduose ir išmokyti prasmingų požymių tiesiogiai iš pikselių duomenų [DBK⁺20]. Vaizdų transformeris, toliau ViT, – tai architektūra, kurioje vaizdams apdoroti naudojami dėmesio sutelkimo mechanizmai. Vaizdų transformerio architektūrą sudaro keletas transformerių blokų. Kiekvieną transformerio bloką sudaro du posluoksniai: kelių galvučių (angl. *multi-head*) dėmesio sutelkimo sluoksnis ir tiesioginio sklidimo (angl. *feed-forward*) sluoksnis. Dėmesio sutelkimo sluoksnis apskaičiuoja kiekvieno vaizdo pikselio dėmesio svorius pagal jo ryšį su visais kitais pikseliais, o tiesioginio sklidimo sluoksnis dėmesio sutelkimo sluoksnio išėjimui taiko netiesinę transformaciją. Kelių galvučių dėmesys išplečia šį mechanizmą, leisdamas modeliui vienu metu atkreipti dėmesį į skirtingas įvesties sekos dalis. ViT taip pat apima papildomą dalių (angl. *patch*) įterpimo sluoksnį, kuris vaizdą suskirsto į fiksuoto dydžio dalis ir kiekvieną dalį atvaizduoja į vektorinį pavidalą. Tada šie dalių įterpiniai perduodami į transformerių blokus tolesniam apdorojimui. Galutinė ViT architektūros išvestis yra klasės prognozė, gaunama perduodant paskutinio transformerio bloko išvestį per klasifikatorių, kurį paprastai sudaro vienas pilnai sujungtų sąryšių sluoksnis (9 pav.) [Boe23].



9 pav. Vaizdų transformerio architektūra. (MLP – Daugiasluoksnis perceptronas)

1.8.1. MobileViT modelis

MobileViT modelis yra ViT architektūros variantas, specialiai sukurtas efektyviai ir nereikalaujant daug resursų spręsti vaizdų apdorojimo uždaviniams. MobileViT architektūroje yra naudojamas sluoksnis, kuriame vietinis apdorojimas konvoliucijomis pakeičiamas visuotiniu apdorojimu naudojant transformerius. MobileViT sujungia transformerių galią su CNN efektyvumu, kad būtų pasiektas puikus vaizdo segmentavimo užduočių našumas. Skirtingai nuo tradicinių ViT modelių, kurie veikia su nuosekliais duomenimis, MobileViT veikia tiesiogiai su duomenų rinkiniu, todėl tinka vaizdų užduotims spręsti. Dėl to, kad MobileViT nereikalaujantis yra daug resursų, jis ypač naudingas, kai skaičiavimo ištekliai ir atmintis yra riboti, nesumažinant segmentavimo modelio tikslumo [MR21].

1.8.2. Segformer modelis

SegFormer modelis yra ViT architektūros variantas. Segformer naudoja enkoderio ir dekodero architektūrą. Tačiau vietoj to, kad dekodieriui būtų naudojami konvoliuciniai sluoksniai, Segformer modelis naudoja transformerio sluoksnius. Tai leidžia užfiksuoti priklausomybes ir kontekstinę informaciją visame įvesties vaizde, taip padidinant modelio gebėjimą suprasti sudėtingus erdvinius ryšius. Taip pat Segformer modelis naudoja Daugiasluoksnį perceptroną dekodero dalyje. SegFormer modelis parodo, kad paprastas ir lengvas modelio dizainas taip pat gali veiksmingai gauti gerus segmentavimo rezultatus [XWY⁺21].

1.9. Didžiausią atpažinimo tikslumą pasiekusi architektūra – SegDecNet

Jakob Božič et al. savo pristatytame darbe 2021 metais daugiausia dėmesio skiria gilaus mokymosi metodui, tinkamam pramoninėms kokybės kontrolės problemoms spręsti, kai yra

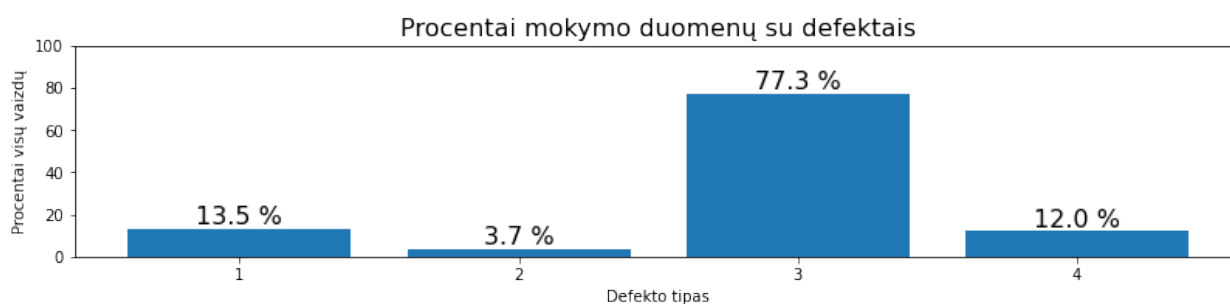
skirtingas defektuotų pavyzdžių ir jų anotacijų prieinamumas. Autoriai siūlo gilaus mokymosi modelį defektams aptikti naudojant visišką pikselių lygmens žymenimis priežiūrą, kai ji yra prieinama. Siūlomas metodas pranoksta visus susijusius metodus visiškai priežiūroje aplinkoje ir taip pat pranoksta silpnai priežiūrimus metodus, kai prieinamos tik vaizdo lygmens etiketės. Autoriai pristato sukurtą SegDecNet CNN architektūrą, ši architektūra yra defektų aptikimui paviršiuose. Pristatytas modelis buvo įvertinamas pagal vidutinį tikslumą (AP). Šiame darbe tiriamas modelio mokymas su silpnai pažymėtais žymenimis iki pilnai priežiūrimo modelio mokymo ir yra lyginami rezultatai. Priežiūrėtai apmokytas modelis pasiekia labai gerus rezultatus su įvairiais duomenų rinkiniais. SegDecNet architektūroje semantiniam segmentavimui atlikti naudojama dekoderio struktūra. SegDecNet architektūrą sudaro keletas konvoliucinių ir sutelkimo sluoksnių, kartu su praleidimo jungtimis ir požymių normalizacija. Taip pat SegDecNet naudoja dinamiškai subalansuotų nuostolių funkcijas, gradiento srauto koregavimo funkcija bei atstumo transformavimo algoritmą žymeniui. Pritaikius šią architektūrą autoriai sugebėjo išgauti beveik tobulus rezultatus semantinio segmentavimo uždavinyje [BTS21].

2. Metodologija

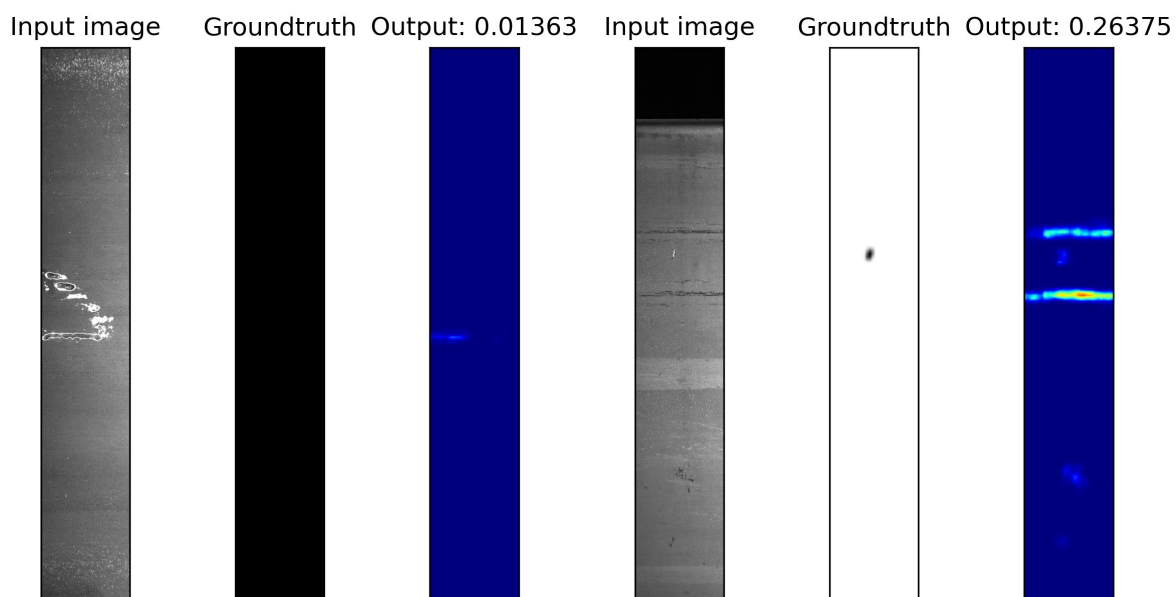
Šiame bakalauro darbe buvo tiriamas defektų aptikimas plieno paviršiuje. Atlikti lyginamąją analizę buvo pasirinkta su enkoderio ir dekoderio architektūromis, aprašytomis 1.5 skyriuje pritaikant į architektūrą skirtingas CNN architektūras, kurios aprašytos 1.7 skyriuje ir vaizdų transformerio modelius, aprašytus 1.8 skyriuje. Šios visos architektūros apmokytos segmentavimo uždaviniui spręsti ir palygintos pagal metrikas. Šiame skyriuje aprašoma naudota duomenų bazė bei modelių tikslumo vertinimo kriterijai.

2.1. Duomenų rinkinys

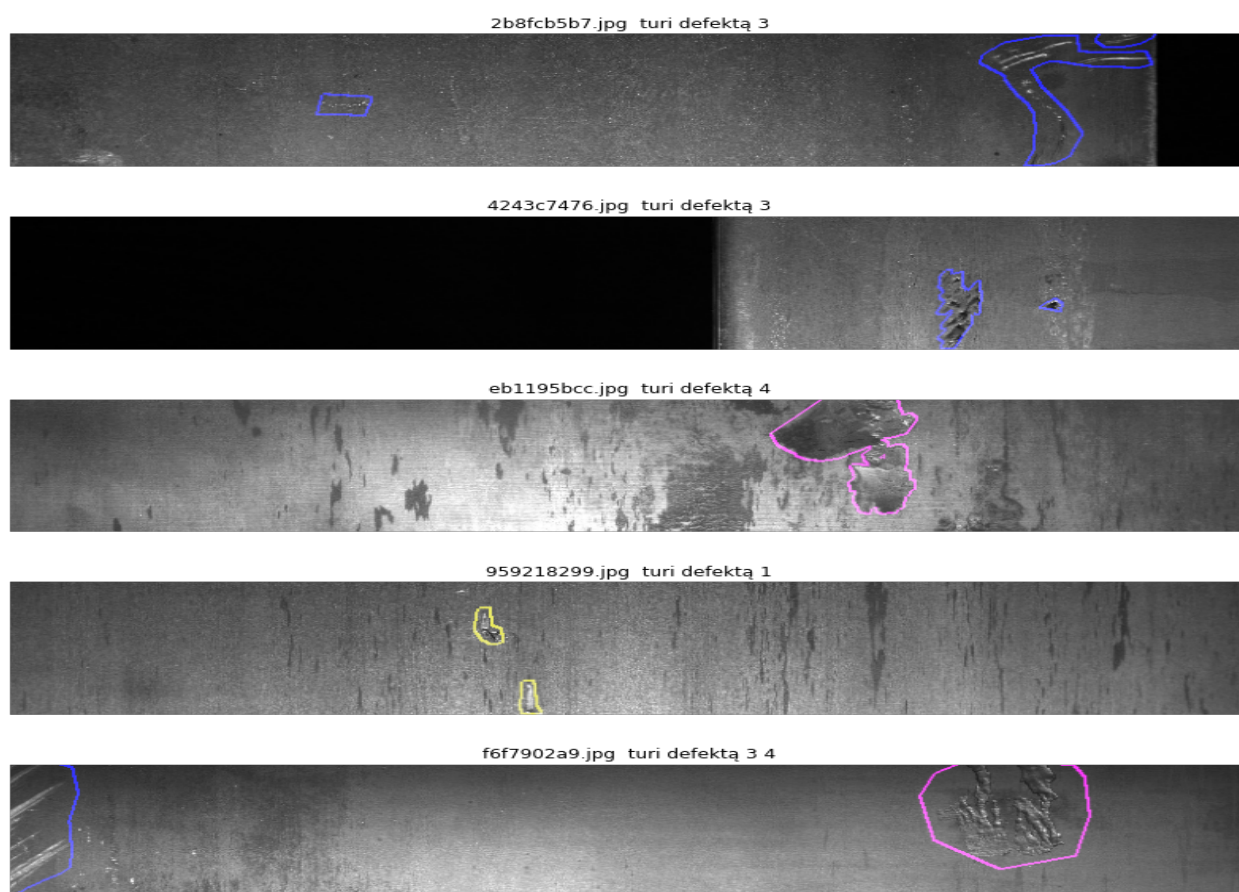
Modelių apmokymui ir testavimui buvo naudojama „Severstal“ įmonės pateiktas duomenų rinkinys. „Severstal“ yra įmonė, daugiausia veikianti plieno ir kasybos pramonėje. Duomenų rinkinį sudaro 12568 nespaltotų plieno paviršiaus nuotraukų, kurie turi vieną iš keturių arba kelis defektus. Nuotraukų rezoliucija yra 1600 x 256 pikselių. Taip pat duomenų rinkinyje yra nurodyta užkoduotos pikselių vertės, kurios nurodo nuotraukose esančias plieno paviršiaus plotą, kuris yra laikomas kaip defektas ir defekto tipą tame plote (10 pav.). Defektuoto paviršiaus plotas užkoduotas yra naudojant „run-length“ kodavimą. Šiame kodavime yra rašomos verčių poros, kuriose yra pradžios padėtis ir eigos ilgis. Pvz., „1 3“ reiškia, kad pradedama nuo 1 pikselio ir iš viso seka trunka 3 pikselius (1, 2, 3). Duomenų rinkinyje esančių defektų pasiskirstymas pavaizduotas pavaizduotas 10 pav. Plieno paviršiaus nuotrauka su defektu pavaizduota 12 pav. Iš nuotraukų galima matyti, kad pirmo tipo defektams priklauso pliene susidarę burbuliukai ir purslai, antro tipo defektams – sulenkimai, susisluoksniavimai pliene, trečio tipo defektams – įbrėžimai atsiradę judinant plieną, ketvirto tipo defektams – žymės, dideli įbrėžimai. Defektai yra labai įvairaus dydžio, formos ir išvaizdos – nuo įbrėžimų ir įlenkimų iki medžiagos pertekliaus. Nors duomenų rinkinys yra gana didelis ir įvairus, kai kurie pažymėti defektai yra gana dviprasmiški ir gali būti netinkamai pažymėti 11 pav.



10 pav. Defektų pasiskirstymas duomenų rinkinyje.



11 pav. Netinkamai pažymėti defektai



12 pav. Plieno nuotraukos su pažymėtais defektų plotais.

2.2. Tikslumo vertinimas

Modelių tikslumo vertinimui pasirinkta naudoti šias formules: Dice'o koeficientas, susikirtimas padalintas iš sąjungos (angl. *Intersection over union (IOU)*), Vidutinis preciziškumas ir

sritis po veikimo charakteristikos (angl. *receiver operating characteristic*) kreivę.

Tikrai teigiamas rezultatas (TP) yra segmentavimo kaukės susikirtimo sritis su pažymėtu defektuotu plotu.

Klaidingas teigiamas rezultatas (FP) yra segmentavimo kaukės plotas, esantis už pažymėto defektuoto ploto.

Klaidingas neigiamas rezultatas (FN) yra plotas pažymėtame defektuotame plote, kurio nepavyko modeliui segmentuoti.

Dice'o koeficiento formulė:

$$Dice = \frac{2|X \cap Y|}{|X| + |Y|}. \quad (1)$$

Formulėje, X žymi prognozuotą defekto sritį, o Y - realią defekto sritį. Ši formulė matuoja kokia dalis persidengia modelio nustatyto defekto ploto ir realaus pliene esančio defekto ploto.

F1 statistika:

$$F1 = \frac{2 * P * R}{P + R} = \frac{2 * TP}{2 * TP + FP + FN} \quad (2)$$

Susikirtimas padalintas iš sąjungos:

$$IoU = \frac{TP}{(TP + FP + FN)}. \quad (3)$$

Preciziškumas:

$$P = \frac{TP}{TP + FP} \quad (4)$$

Atkūrimo statistika:

$$R = \frac{TP}{TP + FN} \quad (5)$$

Vidutinis preciziškumas (AP) yra plotas po tikslumo ir atkūrimo kreivę. Tikslumo ir atkūrimo kreivė rodo santykį tarp tikslumo ir atkūrimo statistikos. Vidutinis preciziškumas apibendrina tikslumo-atitikimo kreivę į vieną reikšmę, rodančią visų tikslumų vidurkį. Bendrai paėmus, vidutinis preciziškumas yra plotas po tikslumo ir atkūrimo kreivę:[Dem23]

Vidutinis preciziškumas:

$$AP = \int_R^1 P(R) dR \quad (6)$$

Sritis po veikimo charakteristikos kreivę (AUC). Ši metrika parodo, kiek modelis geba atskirti klases. Kuo didesnė AUC, tuo geriau modelis prognozuoja ar plotas defektuotas ar ne:

$$AUC = \int_R^1 R(FPR) dFPR \quad (7)$$

$$FPR = \frac{FP}{TN + FP} \quad (8)$$

3. Eksperimentiniai tyrimai

Eksperimentiniai tyrimai buvo atlikti naudojant *Kaggle* platformoje esančią techninę įrangą, kuri yra prieinama visiems registruotiems naudotojams ir Vilniaus universiteto Matematikos ir informatikos fakulteto Informacinių technologijų atviros prieigos centro suteiktus HPC išteklius. *Kaggle* įrangos specifikacija: GPU – *NVIDIA Tesla P100* su 16 GB operatyviosios atminties. Vilniaus universiteto suteiktos įrangos specifikacija: GPU – *NVIDIA Tesla V100* su 32 GB operatyviosios atminties. Eksperimentiniai tyrimai atlikti naudojant tiek TensorFlow tiek PyTorch mašininio mokymo bibliotekas. Iš pradžių eksperimentiniai tyrimai buvo atliekami su TensorFlow biblioteka, bet buvo pereita prie PyTorch bibliotekos dėl didesnio modelių pasirinkimo. Pradžioje su TensorFlow buvo realizuotos 5 skirtingos architektūros: ResNet, VGG, Inception, MobileNet, EfficientNet. Kaip dekoderį buvo pasirinkta naudoti Unet architektūrą. Su šiomis architektūromis buvo pasirinkti šie specifiniai modeliai kaip enkoderiai: ResNet34, ResNet101, SeResNet34, VGG16, InceptionV3, MobileNetV2, EfficientNetB0 ir EfficientNetB2. Po tyrimų buvo pasirinkta naudoti PyTorch biblioteką kad būtų galima panaudoti kitas dekoderio ir enkoderio architektūras. Taip pat buvo pasirinkta pagal daugiau metrikų lyginti šias architektūras. Pasinaudojant PyTorch biblioteka buvo pasirinkti palyginti šias dekoderių architektūras: Unet, UnetPlusPlus, Linknet, FPN, MAnet, PSPNet, PAN, DeepLabV3, DeepLabV3Plus. Su šiais dekoderiais buvo pasirinktos šios enkoderių architektūros: SegFormer, MobileVit kaip vaizdų transformerių modeliai ir ResNeXt, Res2Next ResNet, MobileOne, DenseNet kaip CNN modeliai.

3.1. Skirtingų tinklo architektūrų palyginimas

Visi modeliai buvo apmokyti naudojant perkėlimo mokymąsį. Perkėlimo mokymasis – tai metodas, kai iš anksto apmokytas modelis, kuris buvo apmokytas naudojant didelį duomenų rinkinį, naudojamas kaip pradinis taškas kitai užduočiai atlikti. Šis metodas padeda panaudoti jau išmokytą modelį ir jį perpanaudoti kitai specifiniai užduočiai. Modelis jau būna išmokęs specifinius požymius ir jis juos gali perpanaudoti reikiamai užduočiai. Visi modeliai prieš tai buvo apmokyti naudojant ImageNet duomenų rinkinį. ImageNet duomenų rinkinyje yra daugiau kaip 14 milijonų vaizdų ir daugiau kaip 20000 objektų kategorijų. Modelių mokymui duomenų rinkinys buvo padalintas 80:20, tai reiškia, kad mokymui buvo naudojama 10054 plieno paviršiaus nuotraukų, o validacijai – 2514. Optimizavimo funkcija buvo naudota Adam, nuostolių funkcija buvo naudota binarinė kryžminė entropija, aktyvacijos funkcija buvo naudota sigmoidinė.

3.1.1. ResNet modeliai

Iš ResNet architektūros buvo pasirinkti ResNet34, ResNet101, SeResNet34. Buvo sudaryta hipotezė, kad iš šių 3 modelių, geriausius rezultatus pasieks ResNet101 modelis, dėl savo didesnio kiekio sluoksnių. Taip pat buvo manoma, kad SeResNet34, pasieks geresnius rezultatus už ResNet34, kaip minima moksliniame straipsnyje [HSS18], eksperimentuojant su SeResNet, SeResNet pasiekdavo geresnius rezultatus negu ResNet34 modelis. Tačiau plieno defektų radime, geriausiai pasirodęs modelis iš šių trijų buvo būtent ResNet34.

3.1.2. VGG modelis

Iš VGG architektūros buvo pasirinktas tik vienas modelis – VGG16. Kadangi šis modelis yra paprastos architektūros (tik 16 sluoksnių), buvo manyta, kad šio modelio mokymas, truks mažai laiko, bet lyginant su ResNet34 modeliu, užtruko ilgiau. Tai yra dėl to, nes VGG16 modelis mokymo metu atlieka sudėtingesnius ir daugiau skaičiavimų, nepaisant to, kad turi mažiau sluoksnių [HSS18]. Tikslumo atžvilgiu, VGG16 turėjo prastesnius rezultatus negu ResNet34.

3.1.3. Inception modelis

Iš Inception architektūros buvo pasirinktas InceptionV3 modelis. Kadangi šio modelio architektūra skiriasi nuo kitų modelių, šis modelis pasirinktas buvo, kad išsiaiškinti ar su pradžios moduliais, kurie būtent yra būdingi šiam modeliui, pavyks pasiekti geriausią tikslumą šioje užduotyje ir išsiaiškinti ar ši architektūra yra tinkama šiam uždaviniui.

3.1.4. MobileNet modelis

Iš MobileNet architektūros buvo pasirinktas MobileNetV2 modelis. Nors šis modelis buvo sukurtas mobiliesiems ir integruotiems įrenginiams, jis vis tiek pasiekia aukštus rezultatus, lyginant su kitais modeliais [HZC⁺17]. Šis modelis buvo pasirinktas, kad būtų galima pažiūrėti, kaip jis gerai atlieką defektų atradimą, lyginant su kitais modeliais, kurie nebuvo specifiskai sukurti mobiliesiems įrenginiams.

3.1.5. EfficientNet modeliai

Iš EfficientNet architektūros buvo pasirinkti keli modeliai, EfficientNetB0 ir EfficientNetB2. Šie modeliai pasirinkti, nes pasiekia labai gerus rezultatus, lyginant su kitais modeliais [TL19]. Pasirinkta būtent du modeliai, kad būtų palyginta jų tarpusavio rezultatai ir kaip jie kinta, kai yra keičiamas EfficientNet būdingas jungtinis mastelis ir taip pat palyginti šio modelio, kuris laikomas vienu iš geriausių, rezultatus, su kitais esamais modeliais.

3.1.6. Vaizdų transformerių modeliai

Iš ViT architektūros modelių buvo pasirinkta naudoti MobileViT architektūros modelį MobileViTv2 ir iš SegFormer architektūros mit-b2 modelį. Buvo pasirinkta ištirti vaizdų transformerių modelius, kad būtų galima palyginti jų rezultatus su CNN modeliais ir pažiūrėti kokius rezultatus vaizdų transformerių modeliams pavyks pasiekti plieno paviršiaus defektų segmentavimo uždavinyje.

3.1.7. Enkoderių architektūros

Lyginamajai analizei atlikti buvo pasirinkta ištirti ir palyginti kelių populiarių vaizdo segmentavimo architektūrų našumą: UNet, LinkNet, FPN, MAnet, PSPNet, PAN ir DeepLab. Šios architektūros buvo pasirinktos dėl jų plataus naudojumo, gerų rezultatų sprendžiant semantinio segmentavimo uždutis ir išskirtinių architektūros savybių. UNet yra labai plačiai naudojama ir

populiari architektūra. LinkNet yra žinoma architektūra dėl savo architektūros, kuri nereikalauja daug skaičiavimo resursų, bet užtat pasiekia konkurencingus rezultatus. FPN ir PAN pasižymi savo specifine architektūra turint iš viršaus į apačią ir šoninius sluoksnių sujungimus. MANet architektūroje naudojama padėties dėmesio blokai ir bendro dėmesio sutelkimo apjungimo blokai. PSPNet naudoja piramidės sutelkimo modulį, kuris leidžia veiksmingai suprasti vaizdą platesniu mastu. Lyginant šias architektūras, yra siekiama išsiaiškinti kokius galima pasiekti skirtingų metrikų rezultatus sprendžiant plieno paviršiaus defektų segmentavimo užduotį.

3.1.8. Skirtingų architektūrų rezultatų įvertinimas ir palyginimas

Visi duomenys apie modelių mokymą ir modelių tikslumą surašyti lentelėse žemiau (1, 2, 3 lentelės).

Kai modeliai buvo mokinti naudojant Tensorflow biblioteką, iš 1 lentelės duomenų galima pamatyti, kad mokant modelius 30 epochų, geriausiai pasirodė ResNet34 modelis ir EfficientNetB2, kurie atitinkamai pasiekė 0,66 ir 0,71 Dice'o koeficientą validacijos metu. Būtent šie modeliai buvo pasirinkti toliau mokinti daugiau epochų. Bendrai paėmus visi modeliai turėjo gan panašų Dice'o įvertį, blogiausią turėjo ResNet101. Lyginant EfficientNetB2 ir ResNet34 modelius, išmokintus 60 epochų, iš 1 lentelės duomenų matome, kad EfficientNetB2 turėjo geresnius rezultatus, tačiau mokymo laikas užtruko beveik dvigubai. Mokinant ResNet34 120 epochų, įvertis pasiekė tiek pat kaip ir EfficientNetB2 su 60 epochų. Taigi, per trumpiausią laiką, pasiektą geriausią įvertį yra geriausia naudoti EfficientNetB2, nors ir šis modelis ilgiau mokosi, jis pasiekia geresnius rezultatus, negu ResNet34. Didžiausią preciziškumą pavyko pasiekti MobileNetV2 modeliui – 0,64.

1 lentelė. Modelių mokymo rezultatai (Tensorflow, Unet enkoderis)

Modelis	Epochų sk.	Val. Dice'o įvertis	Preciziškumas
ResNet34	30	0,66	0,55
ResNet101	30	0,61	0,57
SeResNet34	30	0,66	0,57
VGG16	30	0,64	0,59
InceptionV3	30	0,65	0,53
MobileNetV2	30	0,63	0,64
EfficientNetB0	30	0,62	0,59
EfficientNetB2	30	0,71	0,57
EfficientNetB2	60	0,72	0,58
ResNet34	60	0,69	0,61
ResNet34	120	0,72	0,60

Kai modeliai buvo mokinti naudojant PyTorch biblioteką, iš 2 lentelės duomenų galima pamatyti, kad mokant modelius 20 epochų, geriausiai pasirodė MobileVit architektūros mobileViTv2 modelis naudojant PAN ir DeepLabV3Plus enkoderius, kurie atitinkamai pasiekė 0.773 ir 0.763 Dice'o koeficientą validacijos metu. Būtent šis vaizdų transformerio modelis buvo pasirinktas toliau mokinti daugiau epochų su PAN, DeepLabV3Plus, UnetPlusPlus ir PSPNet enkoderiais. Mokinant 50 epochų (3 lentelė) modeliai turėjo gan panašų Dice'o įvertį. Geriausius rezultatus pavyko pasiekti naudojant PAN enkoderį ir mobileViTv2 dekodery: Dice'o įvertis pasiektas 0,78, F1 pasiektas 0,476 ir IoU pasiektas 0,64.

2 lentelė. Modelių mokymo rezultatai (Pytorch)

Enkoderis	Dekoderis	Epochų sk.	Val. Dice	Val. F1	Val. IoU
Unet	mit_b2	20	0,681	0,397	0,518
Unet	resnet101	20	0,721	0,426	0,565
Unet	resnext50_32x4d	20	0,734	0,438	0,582
Unet	timm-res2next50	20	0,751	0,452	0,603
Unet	mobileone_s4	20	0,741	0,444	0,589
Unet	densenet121	20	0,746	0,448	0,596
UnetPlusPlus	mobileone_s0	20	0,731	0,435	0,578
Linknet	densenet121	20	0,737	0,44	0,584
Linknet	res2next50	20	0,738	0,441	0,586
FPN	densenet121	20	0,762	0,461	0,616
FPN	res2next50	20	0,748	0,449	0,599
MAnet	mit_b2	20	0,707	0,415	0,549
MAnet	densenet121	20	0,734	0,437	0,579
MAnet	res2next50	20	0,726	0,431	0,572
PSPNet	mit_b2	20	0,72	0,427	0,565
PSPNet	densenet121	20	0,755	0,454	0,607
PSPNet	res2next50	20	0,747	0,449	0,597
PAN	mobilevitv2	20	0,773	0,47	0,631
PAN	lambda_resnet50ts	20	0,672	0,39	0,511
DeepLabV3	resnet34	20	0,735	0,44	0,583
DeepLabV3Plus	mobilevitv2_150	20	0,763	0,462	0,618
DeepLabV3Plus	lambda_resnet50ts	20	0,522	0,289	0,361

3 lentelė. Modelių mokymo rezultatai su MobileVit dekoderiu

Enkoderis	Dekoderis	Epochų sk.	Val. Dice	Val. F1	Val. IoU
PAN	mobilevitv2_150	50	0,78	0,476	0,64
DeepLabV3Plus	mobilevitv2_150	50	0,774	0,471	0,633
UnetPlusPlus	mobilevitv2_150	50	0,777	0,472	0,635
PSPNet	mobilevitv2_150	50	0,726	0,433	0,572

3.2. SegDecNet architektūros tobulinimas

Šiame bakalauro darbe buvo atlikti eksperimentai, siekiant pagerinti SegDecNet architektūros rezultatus naudojant „Severstal“ plieno duomenų rinkinį. Iš pradžių buvo atkartoti rezultatai pateikti moksliniame straipsnyje, kuriame ši architektūra buvo pristatyta. Kadangi apmokinti modeliai paruošti testavimui tokie, kokie buvo naudojami straipsnyje nebuvo pateikti, buvo atlikta architektūros analizė ir apmokyti modeliai. Naudojant straipsnyje nurodytus parametrus, sėkmingai atkartoti rezultatai.

Siekiant pagerinti architektūros rezultatus, buvo įgyvendinti šie pakeitimai:

1. Svorio inicializavimas – buvo pakeistas konvoliucinio ir rinkinio normalizavimo sluoksnių svorio inicializavimo metodas. Vietoj Xavier inicializacijos buvo panaudota Kaiming inicializacija, kuri labiau tinka modeliams su ReLU aktyvacijomis. Xavier inicializacija pristatyta anksčiau negu Kaiming inicializacija. Xavier inicializacija yra pristatyta atsižvelgiant į sigmoidinę aktyvacijos funkciją. Sigmoidinę aktyvacijos funkciją savo rezultatais pralenkė ReLu aktyvacijos funkcija ir taip pat atsižvelgiant į ReLu aktyvacijos funkciją, buvo pristatyta Kaiming inicializacija, kuri geriau tinka su ReLu aktyvacijos funkcija. Kadangi SegDecNet architektūroje naudojamos ReLu aktyvacijos funkcijos, buvo pasirinkta pakeisti Xavier inicializaciją į Kaiming inicializaciją.
2. Duomenų rinkinio (angl. *batch*) normalizavimas – mokymo metu nustatytas duomenų rinkinio normalizavimo sluoksnių einamosios statistikos sekimas. Su šiuo normalizavimu modelis gali pritaikyti normalizavimo parametrus pagal esamą duomenų rinkinio statistiką. Tai leidžia modeliui mokymosi metu dinamiškai mokytis ir koreguoti normalizavimo parametrus.
3. Regularizacija – regularizacija, tai baudos/papildomų kaštų pridėjimas prie skaičiuojamos nuostolių funkcijos, įgalinanti atsižvelgti į iškeltus apribojimus. Prie modelio buvo pridėtas atsitiktinis praretinimas pridedant atsitiktinio praretinimo sluoksnius. Modelio mokymo metu atsitiktine tvarka dalis įvesties elementų nustatoma į 0. Tai padeda išvengti persimokymo, nes sumažina modelio priklausomybę nuo konkrečių požymių. Atsitiktinai išmesdamas elementus, atsitiktinis praretinimas įveda triukšmą ir skatina modelį mokytis patikimesnių ir bendresnių reprezentacijų. Pridėjus atsitiktinio praretinimo regularizaciją, galima pagerinti modelio rezultatus ir sumažinti persimokymą.

3.2.1. SegDecNet eksperimentinio tyrimo rezultatai

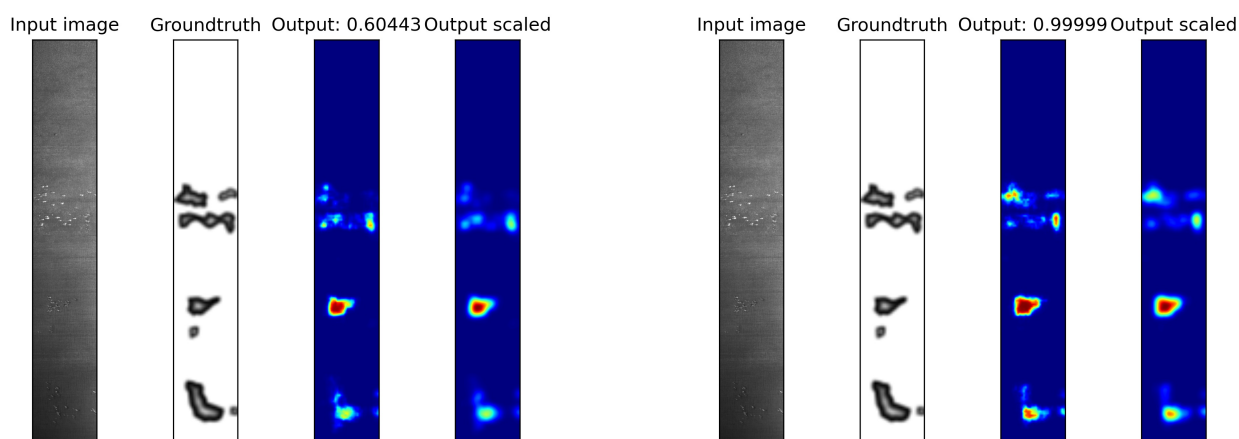
Straipsnyje pateikti tokie vidutinio preciziškumo rezultatai: 99,3%, kai modelis mokintas su pilna priežiūra, 98,6 %, kai vykdoma mišri priežiūra, ir 98,74 %, kai vykdoma mažesnė imtis su pilna priežiūra. Rezultatai buvo sėkmingai atkartoti naudojant tuos pačius parametrus kurie pateikti straipsnyje. Tačiau mokant modelį pagal mažesnę imtį, vidutinis tikslumas buvo 97,97 %, t. y. mažesnis, nei nurodyta straipsnyje.

Atlikus pakeitimus, buvo pagerintas vidutinis preciziškumas ir kiti rezultatai, palyginti su straipsnyje pateiktais rezultatais. Konkrečiai, vidutinis preciziškumas buvo 99,45 %, kai taikoma visiška priežiūra, ir 98,98 %, kai taikoma mišri priežiūra. Apmokant mažesnę imtį, vidutinis tikslumas padidėjo nuo 97,97 % iki 98,98 %.

4 lentelė. SegDecNet eksperimentinio tyrimo rezultatai

	Prižiūrimas mokymas				Dalinai prižiūrimas mokymas				Prižiūrimas (mažesne imtis)			
	AUC	AP	F1	FP+FN	AUC	AP	F1	FP+FN	AUC	AP	F1	FP+FN
Straips. rez.	-	0,993	-	-	-	0,986	-	-	-	0,9874	-	59+40
Atkartoti rez.	0,992493	0,992964	0,968	38+38	0,989694	0,989383	0,96	41+55	0,978089	0,979653	0,938	57+66
Modifikacijų rez.	0,994028	0,994463	0,967	36+44	0,991424	0,992281	0,964	35+50	0,988607	0,989827	0,952	24+70

Iš 4 lentelės rezultatu galima matyti, kad po modifikacijų architektūroje prižiūrimo mokymo metu rezultatai pagerėjo labai minimaliai: vidutinis preciziškumas padidėjo 0,1 procentiniais punktais, dalinai prižiūrimo mokymo metu: AUC pagerėjo 0,2 procentiniais punktais, AP pagerėjo 0,63 procentiniais punktais ir kai modelis buvo mokintas pilnai prižiūrimas, bet su mažesne imtimi (1000 nuotraukų): AUC pagerėjo 1,1 procentiniais punktais, AP pagerėjo 1 procentiniais punktu nuo atkartoto rezultato, bet tik 0,24 procentiniais punktais nuo straipsnyje pateikto rezultato ir F1 pagerėjo 1,4 procentiniais punktais.



(a) Pilnai prižiūrimo mokymo straipsnio segmentavimo pvz. AP = 0,604

(b) Pilnai prižiūrimo mokymo modifikuotos architektūros segmentavimo pvz. AP = 1

13 pav. Segmentavimo palyginimas

Rezultatai ir išvados

Darbo eigoje išanalizuotos skirtingos CNN ir ViT architektūros bei eksperimentiškai ištirtas plieno paviršiaus defektų semantinio segmentavimo uždavinys. Praktinis įvairių architektūrų modelių realizavimas ir literatūros analizė parodė šių architektūrų ir jų specifinių modelių tinkamumą, sprendžiant plieno paviršiaus defektų segmentavimo uždavinį. Darbo eigoje taip pat ištirtas geriausia pristatyta architektūra plieno paviršiaus defektų semantinio segmentavimo uždaviniui spręsti. Eksperimentinių tyrimu metu sėkmingai pagerinti šios architektūros rezultatai pritaikius modifikacijas.

Rezultatai:

1. Išanalizuotos skirtingos giliųjų neuroninių tinklų architektūros ir priimtas sprendimas dėl jų naudojimo lyginamojoje analizėje ir priimtas sprendimas naudoti dekoderių architektūras Unet, UnetPlusPlus, Linknet, FPN, MAnet, PSPNet, PAN, DeepLabV3, DeepLabV3Plus. Su šiais dekoderiais buvo pasirinktos šios enkoderių architektūros: SegFormer, MobileVit kaip vaizdų transformerių modeliai ir ResNeXt, Res2Next ResNet, MobileOne, DenseNet kaip CNN modeliai.
2. Išanalizuotas „Severstal“ įmonės pateiktą duomenų rinkinys, kuriame pastebėtas netinkamas defektų žymėjimas ir duomenų disbalansas.
3. Eksperimentiškai ištirti realizuoti CNN ir ViT modeliai. Jie palyginti pagal Sorenseno-Dice'o panašumo koeficientą, F1 įvertį ir IoU. PAN architektūra su MobileVit dekoderiu pasiekė geriausius rezultatus pagal šias metrikas.
4. Išanalizuota SegDecNet architektūra, kuri pasiekia geriausius rezultatus segmentuojant „Severstal“ plieno duomenis ir pritaikius modifikacijas pagerinti rezultatai.

Iš gautų rezultatų galima daryti šias išvadas:

1. Pakeitus svorio inicializavimą iš Xavier į Kaiming, pritaikius duomenų rinkinio normalizavimą ir pridėjus atsitiktinio praretinimo regularizaciją prie SegDecNet architektūros galima pagerinti šios architektūros rezultatus. Prižiūrimo mokymo metu rezultatai: vidutinis preciziškumas padidėjo 0,1 procentiniais punktais, dalinai prižiūrimo mokymo metu: AUC pagerėjo 0,2 procentiniais punktais, AP pagerėjo 0,63 procentiniais punktais ir kai modelis buvo mokintas pilnai prižiūrimas, bet su mažesne imtimi (1000 nuotraukų): AUC pagerėjo 1,1 procentiniais punktais, AP pagerėjo 1 procentiniu punktu nuo atkartoto rezultato, bet 0,24 procentiniais punktais nuo straipsnyje pateikto rezultato ir F1 pagerėjo 1,4 procentiniais punktais.
2. Buvo atrasta, kad ViT tinkami naudoti plieno defektų segmentavimui ir su ViT architektūros modeliais galima pasiekti geresnius rezultatus negu su CNN architektūros modeliais.
3. Lyginamosios analizės metu, buvo atrasta, kad geriausius rezultatus pasiekęs dekoderis yra MobileVit architektūros modelis mobilevitv2. Geriausius rezultatus pasiekusios architektūros yra PAN, DeepLabV3Plus, UnetPlusPlus, PSPNet ir FPN. Iš šių penkių architektūrų aukščiausius rezultatus pavyko pasiekti su PAN enkoderiu. Geriausia enkoderio

ir dekoderio kombinacija yra PAN enkoderis su mobileVitv2 dekoderiu. Naudojant PAN enkoderį su mobilevitv2 dekoderiu pasiekti geriausi rezultatai: Dice'o įvertis 0,78; F1 – 0,476; IoU – 0,64.

4. Jei „Severstal“ plieno paviršiaus defektų duomenų rinkinyje visų nuotraukų defektai būtų tinkamai sužymėti, galimai būtų galima pasiekti geresnius modelių rezultatus.

- [AAL⁺21] M. Abu, A. Amir, Y. Lean, N. Zahri, S. Azemi. The performance analysis of transfer learning for steel defect detection by using deep learning [<https://iopscience.iop.org/article/10.1088/1742-6596/1755/1/012041.pdf>]. 2021. J. Phys. Conf. Ser. 1755(1), 012041.
- [AL20] R. Augustauskas, A. Lipnickas. Improved pixel-level pavement-defect segmentation using a deep autoencoder. *Sensors*. 2020, tomas 20, numeris 9, p. 2557.
- [Bah23] P. Baheti. Image Recognition: Definition, Algorithms & Uses [<https://www.v7labs.com/blog/image-recognition-guide>]. 2023.
- [Bas21] E. Basson. World Steel in Figures 2021 [<https://worldsteel.org/wp-content/uploads/2021-World-Steel-in-Figures.pdf>]. 2021.
- [Boe23] G. Boesch. Vision Transformers (ViT) in Image Recognition [<https://viso.ai/deep-learning/vision-transformer-vit/>]. 2023.
- [Bri15] D. Britz. Understanding Convolutional Neural Networks for NLP [<https://dennybritz.com/posts/wildml/understanding-convolutional-neural-networks-for-nlp/>]. 2015.
- [BTS21] J. Božič, D. Tabernik, D. Skočaj. Mixed supervision for surface-defect detection: From weakly to fully supervised learning. *Computers in Industry*. 2021, tomas 129, p. 103459.
- [CC17] A. Chaurasia, E. Culurciello. Linknet: Exploiting encoder representations for efficient semantic segmentation. Iš: *2017 IEEE visual communications and image processing (VCIP)*. IEEE, 2017, p. 1–4.
- [CPS⁺17] L.-C. Chen, G. Papandreou, F. Schroff, H. Adam. *Rethinking Atrous Convolution for Semantic Image Segmentation*. 2017.
- [DBK⁺20] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn ir kiti. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*. 2020.
- [Dem23] K. Dementavičiūtė. *Inkstų auglių segmentavimas kompiuterinės tomografijos vaizduose taikant vienos ir dviejų pakopų metodus, esant mažoms duomenų imtims*. 2023. Disertacija. Vilniaus universitetas.
- [DSF20] W. Du, H. Shen, J. Fu. Automatic defect segmentation in X-ray images based on deep learning. *IEEE Transactions on Industrial Electronics*. 2020, tomas 68, numeris 12, p. 12912–12920.
- [FWL⁺20] T. Fan, G. Wang, Y. Li, H. Wang. MA-Net: A Multi-Scale Attention Network for Liver and Tumor Segmentation. *IEEE Access*. 2020, tomas 8, p. 179656–179665. Prieiga per internetą: <https://doi.org/10.1109/ACCESS.2020.3025372>.

- [GCZ⁺19] S.-H. Gao, M.-M. Cheng, K. Zhao, X.-Y. Zhang, M.-H. Yang, P. Torr. Res2net: A new multi-scale backbone architecture. *IEEE transactions on pattern analysis and machine intelligence*. 2019, tomas 43, numeris 2, p. 652–662.
- [HLV⁺17] G. Huang, Z. Liu, L. Van Der Maaten, K. Q. Weinberger. Densely connected convolutional networks. Iš: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, p. 4700–4708.
- [HSS18] J. Hu, L. Shen, G. Sun. Squeeze-and-excitation networks. Iš: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, p. 7132–7141.
- [HZC⁺17] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*. 2017.
- [HZR⁺16] K. He, X. Zhang, S. Ren, J. Sun. Deep residual learning for image recognition. Iš: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, p. 770–778.
- [YND⁺18] R. Yamashita, M. Nishio, R. K. G. Do, K. Togashi. Convolutional neural networks: an overview and application in radiology. *Convolutional neural networks: an overview and application in radiology*. 2018.
- [Jor18] J. Jordan. An overview of semantic image segmentation [[https : / / www . jeremyjordan.me/semantic-segmentation/](https://www.jeremyjordan.me/semantic-segmentation/)]. 2018.
- [KRS⁺21] A. V. Kakade, S. Rajkumar, K. Suganthi, L. Ramanathan. Object Detection in Satellite Images Using Modified Pyramid Scene Parsing Networks. Iš: *Sensor Data Analysis and Management: The Role of Deep Learning*. 2021, p. 147–160. Prieiga per internetą: <https://doi.org/10.1002/9781119682806.ch9>.
- [LDG⁺17] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, S. Belongie. Feature pyramid networks for object detection. Iš: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, p. 2117–2125.
- [Le21] K. Le. An overview of VGG16 and NiN models [<https://medium.com/mlearning-ai/an-overview-of-vgg16-and-nin-models-96e4bf398484>]. 2021.
- [LMW⁺22] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, S. Xie. A convnet for the 2020s. Iš: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, p. 11976–11986.
- [LQQ⁺18] S. Liu, L. Qi, H. Qin, J. Shi, J. Jia. *Path Aggregation Network for Instance Segmentation*. 2018.
- [MR21] S. Mehta, M. Rastegari. Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. *arXiv preprint arXiv:2110.02178*. 2021.
- [Pot22] R. Potter. How Bounding Box Annotation Helps Object Detection in Machine Learning: Use Cases [<https://worldsteel.org/wp-content/uploads/2021-World-Steel-in-Figures.pdf>]. 2022.

- [RFB15] O. Ronneberger, P. Fischer, T. Brox. U-net: Convolutional networks for biomedical image segmentation. Iš: *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5–9, 2015, Proceedings, Part III 18*. Springer, 2015, p. 234–241.
- [SHZ⁺18] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.-C. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. Iš: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, p. 4510–4520.
- [SLJ⁺15] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich. Going deeper with convolutions. Iš: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, p. 1–9.
- [SVI⁺16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna. Rethinking the inception architecture for computer vision. Iš: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, p. 2818–2826.
- [SZ14] K. Simonyan, A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*. 2014.
- [TL19] M. Tan, Q. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. Iš: *International conference on machine learning*. PMLR, 2019, p. 6105–6114.
- [VSP⁺17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin. *Attention Is All You Need*. 2017.
- [XGD⁺17] S. Xie, R. Girshick, P. Dollár, Z. Tu, K. He. Aggregated residual transformations for deep neural networks. Iš: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, p. 1492–1500.
- [XWY⁺21] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, P. Luo. SegFormer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*. 2021, tomas 34, p. 12077–12090.
- [ZYW⁺19] M. Zhang, Z. Yu, H. Wang, H. Qin, W. Zhao, Y. Liu. Automatic digital modulation classification based on curriculum learning. *Applied Sciences*. 2019, tomas 9, numeris 10, p. 2171.
- [ZSQ⁺17] H. Zhao, J. Shi, X. Qi, X. Wang, J. Jia. *Pyramid Scene Parsing Network*. 2017.

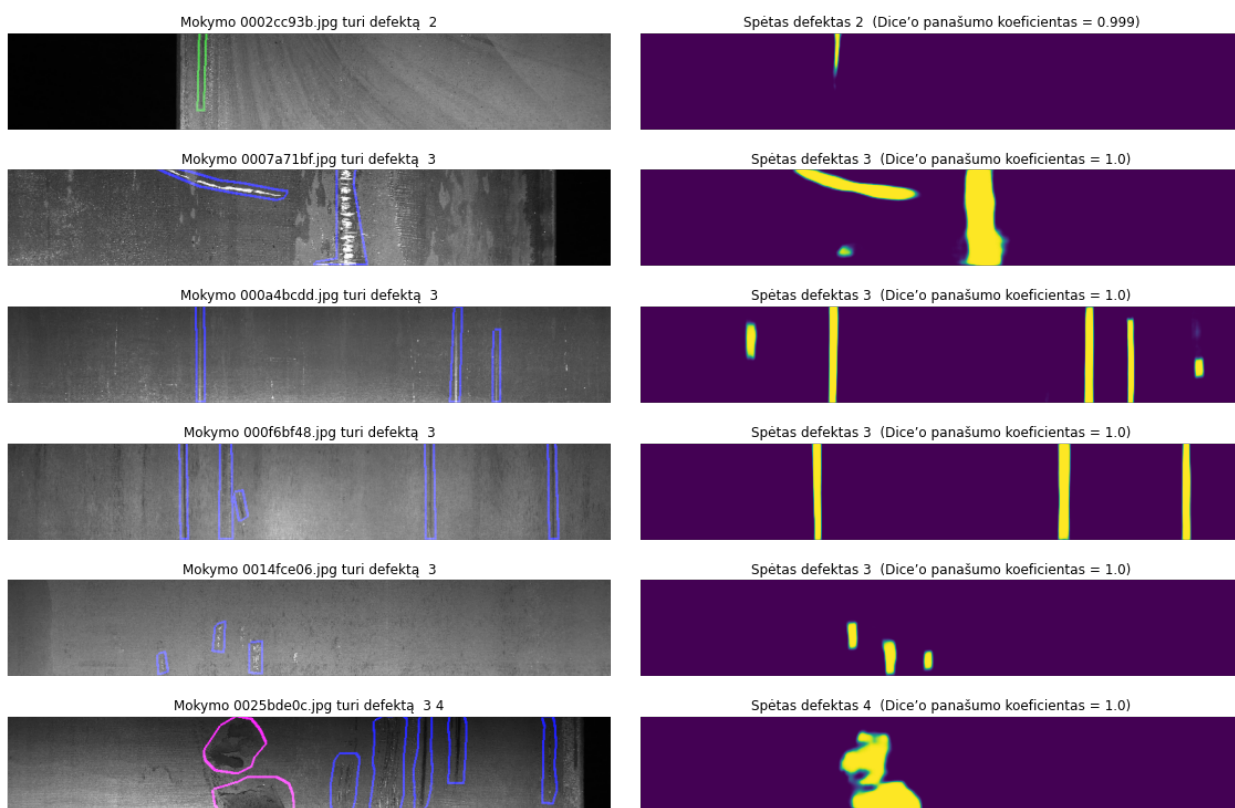
Santrumpos

- **CNN** konvoliuciniai neuroniniai tinklai
- **ViT** vaizdų transformeris
- **Dice** Sorenseno-Dice'o panašumo koeficientas
- **F1** F1 statistika
- **IoU** susikirtimas padalintas iš sąjungos
- **P** preciziškumas
- **R** atkūrimo statistika
- **AP** vidutinis preciziškumas
- **AUC** sritis po veikimo charakteristikos kreive
- **FPR** klaidingai teigiamo rezultato dažnis

Priedai

Priedas nr. 1

Defektų radimas



14 pav. defektų radimas

Priedas nr. 2

GitHub repozitorijos nuoroda

Nuoroda į kodo repozitoriją. Repozitorijoje patalpintas programinis kodas, kuris buvo naudotas tiramojo darbo metu: <https://github.com/CombatPanda/Bakalaurinis>