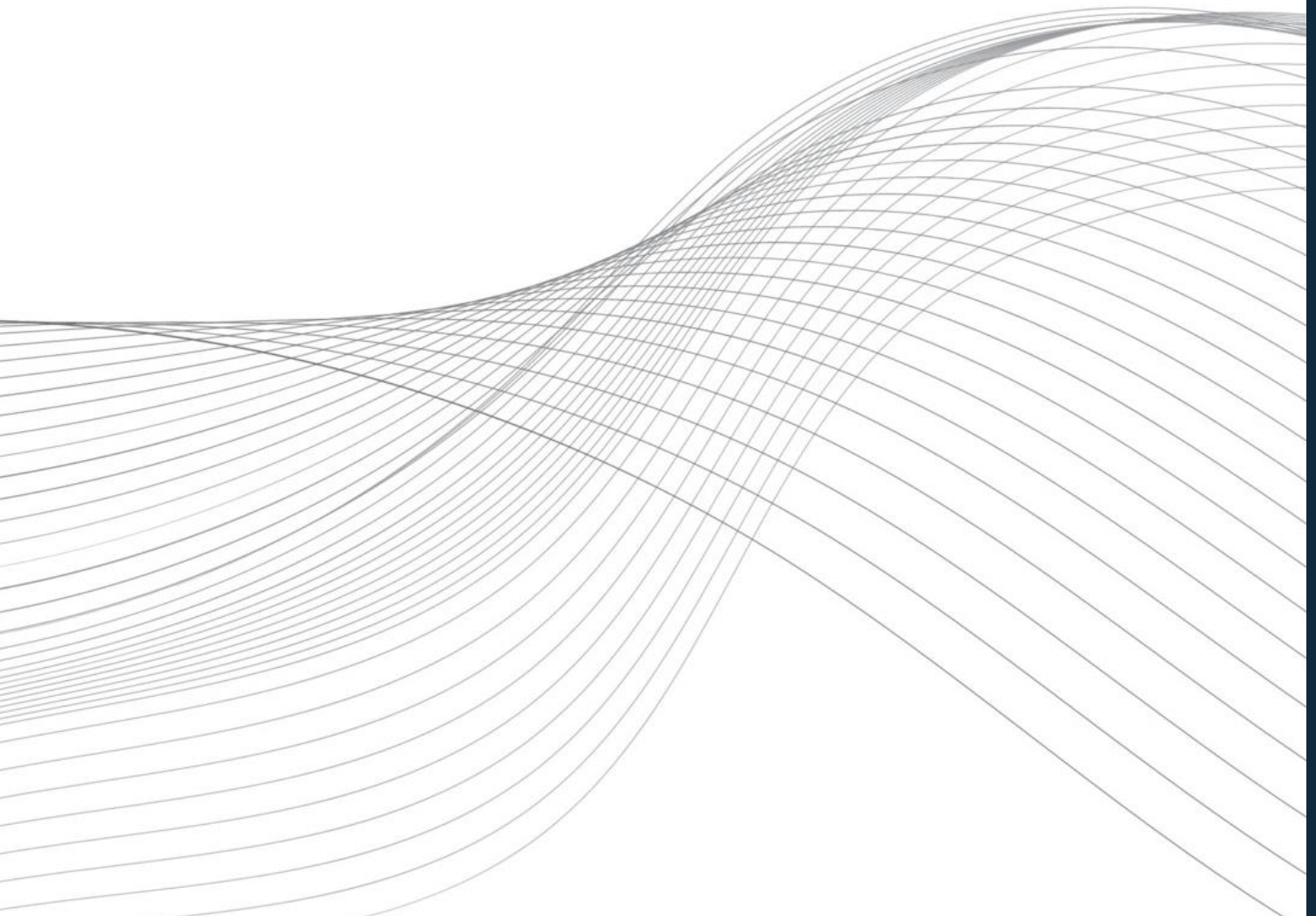


SMART inwinnen van data in de openbare ruimte

Onderzoeksrapport | object detectiemodellen

23 juni 2023, 's-Hertogenbosch | Projectcode: 22200396

In opdracht van: Ministerie van Infrastructuur en Waterstaat



HAS green academy
Onderwijsboulevard 221
PO Box 90108
5200 MA 's-Hertogenbosch
Telefoon: +31 (0)88 890 36 00

Documenttitel: SMART inwinnen van data in de openbare ruimte
Projectcode: 22200396

Status: Versie 1.0

Opdrachtgever: Ministerie van Infrastructuur en Waterstaat
Contactpersoon: Rianne Dobbelsteen

Projectleider inhoud: Paulo van Breugel
Projectteam: Rik de Hoop
Mats van der Burg

Plaats: 's-Hertogenbosch
Datum: 23 juni 2023

Voorwoord

Voor u ligt het adviesrapport behorend tot de beroepsopdracht "SMART inwinnen van data in de openbare ruimte". Dit rapport is geschreven in opdracht van het Ministerie van Infrastructuur en Waterstaat, met als doel om te voldoen aan de afstudeereisen van de opleiding Applied Geo-Information Science aan de HAS green academy. Dit project is uitgevoerd in de periode van februari 2023 tot en met juni 2023.

Wij hebben door dit project allebei persoonlijke ontwikkeling en professionele groei doorgemaakt die voor ons van grote waarde is. Dit afstudeerproject onderscheidt zich omdat het niet op individuele basis is uitgevoerd, in tegenstelling tot de meeste scripties. Het samenwerken heeft ons geholpen kennis en vaardigheden met elkaar te delen. De gezamenlijke inzichten hebben wij toe kunnen passen in het ontwikkelingsproces, wat heeft bijgedragen aan een eindproduct waar we trots op zijn.

Om te beginnen willen wij graag onze procesbegeleider, Vincent Wissink, bedanken voor zijn ondersteuning gedurende dit project. De focus op persoonlijke begeleiding heeft bijgedragen aan een prettig verloop van het project waarin wij onze kwaliteiten optimaal konden toepassen. Daarnaast willen wij Robby Veders (IenW), Frans Geurts (RWS) en Michiel Feijt (RWS) bedanken voor de ondersteuning en betrokkenheid bij dit project. Tot slot willen wij graag Rianne Dobbelsesteen (IenW), Paulo van Breugel en Maurits Dorlandt bedanken voor hun betrokkenheid bij het opstarten van het project, het delen van kennis en de prettige communicatie.

Mats van der Burg en Rik de Hoop

's-Hertogenbosch, 23 juni 2023

Samenvatting

Het REV (Register Externe Veiligheidsrisico's) bestaat uit data over activiteiten die een potentieel risico vormen voor de openbare ruimte. Daarom is het van belang dat de informatie in dit register nauwkeurig en accuraat geregistreerd is. Om deze reden wordt de data aangeleverd door gemandateerde bronhouders aan het Ministerie van Infrastructuur en Waterstaat. Door de hoeveelheid bronhouders, activiteiten en schaal is de opzet van een handmatige controle over deze data arbeidsintensief en inefficiënt.

Om inzichten te verkrijgen in de nauwkeurigheid en volledigheid van de aangeleverde data heeft het Ministerie van Infrastructuur en Waterstaat daarom belang bij een geautomatiseerde betrouwbaarheidscontrole. Een automatische controle is complex door de diversiteit van activiteiten in het REV. Het register bevat namelijk 47 elementen met ieder afwijkende fysieke eigenschappen. Deze fysieke variaties hebben invloed op de wijze waarin het element kan worden gecontroleerd. Zo zijn bepaalde objecttypen niet zichtbaar op luchtfoto's door fysieke belemmeringen van bomen of gebouwen. Voor deze problematiek is de volgende onderzoeksvraag opgesteld:

In hoeverre is het mogelijk om een kwaliteitscontrole over de activiteiten binnen het REV efficiënter te maken en te optimaliseren om zo bij te dragen aan een verhoogde betrouwbaarheid en verbeterd inzicht in de dataflow voor het Ministerie van Infrastructuur en Waterstaat?

Met dataflow wordt hierbij bedoeld op de manier hoe data van de bron naar het register wordt verplaatst.

Vanuit de onderzoeksvraag is een methodologie ontwikkeld op basis van een object detectiemodel (YOLOv5). Een object detectiemodel is een methode binnen AI (Artificial Intelligence) gericht op het detecteren van objecten uit digitaal beeldmateriaal. In dit geval op basis van luchtfoto's met een 8 centimeter resolutie. Door de hoeveelheid activiteiten in het REV is de methodologie toegepast op twee objecttypen, windturbines en propaantanks, in geselecteerde testgebieden. De methodologie is overigens ontwikkeld met de doelstelling om reproduceerbaar te zijn voor andere objecttypen in het REV.

De resultaten hebben aangetoond dat de ontwikkelde methodologie effectief is bij het lokaliseren van nieuwe objecten en het controleren van bestaande windturbines en propaantanks. Het YOLOv5 model toegepast op grote windturbines heeft een mAP score van 0.98 opgeleverd. Dit in tegenstelling tot de resultaten van kleine- en middelgrote windturbines die beduidend minder nauwkeurige resultaten hebben getoond, ondanks dat de mAP score een hoge waarde van 0.91

heeft. Voor het eerste testgebied zijn 70 windturbines gedetecteerd (op basis van een 0.5 threshold) in tegenstelling tot de 83 objecten in het REV. Dit wordt verklaard door de aanwezigheid van kleine- tot middelgrote windturbines en windturbines in opbouw die daardoor niet gedetecteerd zijn. Voor het tweede testgebied heeft het resultaat 4 windturbines gedetecteerd. Dit is in lijn met het REV op basis van een 0.6 threshold.

Ondanks de fysieke belemmeringen van propaantanks heeft het YOLOv5 model een mAP score van 0.91 bereikt. Voor het testgebied van propaantanks zijn met een threshold van 0.6, 43 propaantanks gedetecteerd in tegenstelling tot de 34 binnen het REV. Het model heeft 16 potentiële propaantanks als aanvulling op het REV gedetecteerd. Van deze detecties zijn 6 objecten als correct vastgesteld. Daarnaast toont het feit dat de methodologie effectief is toegepast op twee uiteenlopende objecttypen een mate van reproduceerbaarheid.

Op basis van de inzichten verkregen uit dit onderzoek wordt aanbevolen om de methodologie op te schalen voor de objecttypen propaantanks en windturbines. Verder wordt aanbevolen om de huidige methodologie uit te breiden naar andere objecttypen binnen het REV. Het is hierbij van belang zaken als representatieve datasets, data-augmentatie en de limitaties van het YOLOv5 model in acht te nemen. Verder wordt aanbevolen om bij zaken als thresholds of data-augmentatie exploratieve onderzoeken op te zetten. Bij dit proces blijft het ook van belang de mogelijke implicaties van privacy & data-ethiek bij de toepassing van luchtfoto's in acht te nemen. Wanneer correct toegepast kan deze automatisatie bijdragen aan de kwaliteit van het REV.

Inhoud

1. Inleiding:	10
2. Theoretisch kader:	12
2.1 Register Externe Veiligheidsrisico's (REV).....	12
2.2 Artificial Intelligence (AI)	12
2.3 Object detectiemodellen	14
2.4 YOLO	27
2.5 YOLOv5	36
2.6. Data-ethiek & privacy.....	41
3. Probleemanalyse en strategie:	43
4. Methodologie & materiaal:	44
4.1 Kwalitatief onderzoek	45
4.2 Randvoorwaarden en ontsluiting	45
4.3 Data-acquisitie (globaal)	48
4.4 Dataverwerking (globaal).....	52
4.5 Dataverwerking (object specifiek)	62
5. Resultaten:	71
5.1 YOLOv5 Modellen	71
5.2 Limitaties en uitdagingen	77
5.3 Statistiek modellen	80
5.4 Terugkoppeling van het REV.....	82
5.5 Workflow	90
6. Conclusie:	91
7. Discussie:.....	94
8. Aanbevelingen en adviezen:	98
8.1 Concrete adviezen	98
8.2 Overige aanbevelingen	101

8.3 Technische randvoorwaarden	102
8.4 Overig	104
9. Bibliografie:	105
Bijlagen	111
A.I chop.PY	111
B.I tile_extractor.PY (1/2)	113
B.II tile_extractor.PY (2/2)	114
B.III Installatie Labelimg	116
C.I Match.PY (1/3)	117
C.II Match.PY (2/3)	118
C.III Create_validation_set.PY	119
C. IV Match.PY (3/3)	120
D.I gisify.PY	121
D.II mini_mistake.PY	125
E.I alumentation.PY	126
F.I Hyperparameters (source-code YOLOv5)	129
1. Uitbreiding objecttypen	130
1.1 Kenmerkbaar EVActiviteiten	130
1.2 Niet kenmerkbaar EVActiviteiten	133
1.3 Geen onderdeel WFS	137

Terminologie

Begrip

AI (Artificial Intelligence)

Annoteren/Annotaties

Augmentatie/Data-augmentatie

Batch/Batch size

BGT (Basisregistratie Grootschalige Topografie)

CNN (Convolutional Neural Network)

Confidence threshold

CV (Computer Vision)

Dataflow

DL (Deep Learning)

Down sampling

Epoch

Definitie

Het vermogen om menselijke cognitieve eigenschappen na te bootsen met een computer.

Markeringen en labels op afbeeldingen van specifieke objecten, gebruikt als referentiegegevens

Kunstmatige uitbreiding van een dataset door afbeeldingen visueel aan te passen en daarmee de variatie te verbeteren

De hoeveelheid trainingsvoorbeelden die parallel worden berekend bij één iteratie van een trainingsalgoritme

Uitgebreid register van Nederland met daarin terreinen, spoorlijnen, wegen, waterlopen en gebouwen

Neuraal netwerk gericht op het analyseren van visuele informatie uit afbeeldingen en video's

Vooraf ingestelde drempelwaarde die bepaald hoe zeker het model moet zijn voor een resultaat wordt getoond

Een vorm van AI gericht op het analyseren van digitaal beeldmateriaal

De wijze waarin data wordt ingewonnen, verwerkt en verstuurd vanuit gemandateerde bronhouders naar het REV.

Een verdieping van Machine Learning met een focus op het begrijpen van complexe problematiek

Het verkleinen van de ruimtelijke resolutie van een afbeelding om de berekeningsefficiëntie te verbeteren

Eén iteratie van een volledige dataset in een trainingsalgoritme (neuraal netwerk)

Element	Een onderdeel van het REV, dit kan zowel duiden op een activiteit als een concreet object.
Flags	Command line argumenten gericht op het doorgeven van bepaalde configuraties aan het trainingsproces
GeoTIFF	Bestandsformaat dat ruimtelijke rastergegevens en geo referentie materiaal kan opslaan
GIS (QGIS/ArcGIS Pro)	Kort voor Geographic Information System, QGIS en ArcGIS Pro zijn software toepassingen specifiek gericht op GIS
Hyperparameters	Parameters die het de configuratie van het trainingsproces kunnen aanpassen (aantal iteraties, snelheid, grootte)
mAP	Maatstaf om te bepalen hoe nauwkeurig een model is, gaat van 0 tot 1. Hierin is hoe hoger beter.
Machine Learning (ML)	Een vorm van Artificial Intelligence (AI) die kan leren van gestructureerde data om prestaties te optimaliseren.
Objectdetectiemodel/Objectdetectiesysteem (ODS)	Een methode binnen Computer Vision dat gericht is op het detecteren van specifieke elementen uit digitale beelden
Objecttype/object	Benaming van een risicovol fysiek voorwerp/object binnen het REV.
Realtime (detectie) en FPS	De mogelijkheid om objecten te detecteren met minimale vertraging, dit is gelinkt aan een hoge FPS (Frames Per Second)
Remote Sensing	Methode om informatie te verzamelen zonder in contact te staan met een object (luchtfoto's, satellietbeelden, et cetera.)
Reproduceerbare methodologie	Methode opgesteld met als doel om herhaald te worden voor andere objecttypen
Root folder	Verwijst naar de hoofdmap waarin alle onderliggende mappen zijn opgeslagen

Register Externe Veiligheidsrisico's (REV)	Dit register bevat data over risicovolle activiteiten in de leefomgeving.
Testdata	Aparte dataset van geannoteerde afbeeldingen/video's om te evalueren hoe goed het model presteert
Trainingsdata	Onderdeel van het algoritme bedoeld om patronen te herkennen en te leren.
VRT-bestand	Bestandsformaat (Virtual Raster) gericht op rastergegevens en geografische data
YAML-bestand	Bestandsformaat waarin configuratie-informatie van het model staat genoteerd
YOLO(v5)	Een variant van een object detectie model, v5 staat hierin voor de vijfde editie

Inleiding:

Introductie, achtergrond en doelstelling

01

In de openbare ruimte worden er diverse activiteiten uitgevoerd die een potentieel risico vormen voor de omgevingsveiligheid. Om dit inzichtelijk te maken wordt hier in opdracht van het Ministerie van Infrastructuur en Waterstaat data over geïnventariseerd in het REV (Register Externe Veiligheidsrisico's). Dit gebeurt in samenwerking met (gemandateerde) bronhouders, bestaande uit de omgevingsdiensten, Provincies en Gemeenten. Het is van belang deze data te controleren om zo de nauwkeurigheid en accuraatheid te waarborgen. Dit controleproces is door het nationale schaalniveau en hoeveelheid informatie arbeidsintensief. Het toepassen van slimme technologieën, zoals AI (Artificial Intelligence) is daarmee essentieel bij het bevorderen van de volledigheid en betrouwbaarheid van het REV. Het analyseren van alle objecttypen in het REV is door de grootte van het register niet haalbaar in het opgegeven tijdbestek. Dit onderzoek zal daarom alleen in gaan op detectie van de objecttypen: windturbines en propaantanks op geselecteerde testgebieden.

De overkoepelende organisatie die deze beroepsopdracht faciliteert betreft het Ministerie van Infrastructuur en Waterstaat (minienw). Dit Ministerie is een overheidsinstantie die voor een groot deel verantwoordelijk is voor de leefbaarheid en bereikbaarheid van de openbare ruimte in Nederland. Om de leefbaarheid te waarborgen is een belangrijke kerntaak van het minienw, het beheren en publiceren van data afkomstig van verschillende bronhouders. Verder zijn ook Rijkswaterstaat en Geodan betrokken bij de ontwikkeling van de eindproducten.

Het analyseren van ruimtelijke objecten is door ontwikkelingen binnen AI, CV (Computer Vision) en RS (Remote Sensing) steeds toegankelijker. Deze ontwikkelingen hebben namelijk bijgedragen aan de opkomst van object detectiemodellen die met hoge nauwkeurigheid objecten kunnen detecteren op visueel beeldmateriaal (Namdev et al., 2022). Ondanks dat deze technologie niet nieuw is, zijn de prestaties sterk verbeterd waardoor in 2015 het eerste model beter scoorde dan mensen (Du, 2018). In de huidige samenleving wordt deze technologie daarom steeds vaker toegepast bij het analyseren van personen, gebouwen en auto's (Namdev et al., 2022). De literatuur toont een grote diversiteit object detectiemodellen. Voorbeelden hiervan zijn: YOLO (You Only Look Once, SSD (Single Shot Detection), RCNN, Fast RCNN en Faster RCNN.

Door de beschikbare literatuur is de keuze gemaakt voor het YOLOv5 model op basis van publicaties en niet op een vergelijkingsstudie. Hieruit is gebleken dat YOLOv5 een model is die toegankelijk toegepast kan worden met hoge nauwkeurigheid en snelheid. Voor de opzet en toepassing van het

gekozen model wordt wel een experimenteel onderzoek uitgevoerd. Dit onderzoek is gericht op het vaststellen van limitaties, optimale parameters en data-augmentatie. Door het model te optimaliseren voor de geselecteerde objecttypen in het REV kan het model mogelijk bijdragen aan een veilige leefomgeving. Om deze hypothese te concretiseren is de volgende onderzoeksvraag opgesteld:

In hoeverre is het mogelijk om een kwaliteitscontrole over de activiteiten binnen het REV efficiënter te maken en te optimaliseren om zo bij te dragen aan een verhoogde betrouwbaarheid en verbeterd inzicht in de dataflow voor het Ministerie van Infrastructuur en Waterstaat?

Om antwoord te geven op deze vraag is een onderzoek toegepast. Het eerste onderdeel bestaat uit het onderzoek naar relevante literatuur. Verschillende publicaties zijn geanalyseerd om inzichten te onttrekken. Daarnaast is er een functioneel object detectiemodel opgesteld op basis van YOLOv5. Dit model is opgesteld en getest voor de objecttypen: windturbines en propaantanks. Voor deze objecttypen zijn er representatieve onderzoeksomgevingen geselecteerd. Op basis van een terugkoppeling met het REV is de betrouwbaarheid van het model gecontroleerd. Ten grondslag van deze toepassing staat de reproduceerbaarheid voor andere elementen in het REV. De inzichten voortgekomen uit dit ontwikkelingsproces zijn geformuleerd als concrete adviezen en aanbevelingen.

Het volgende hoofdstuk binnen dit onderzoek toont de resultaten van het literatuuronderzoek dat is uitgevoerd gericht op object detectiemodellen en data-ethiek & privacy. In *hoofdstuk 3* wordt ingegaan op de probleemanalyse en strategie. Dit hoofdstuk fungeert als herhaling van de probleemstelling en doelstelling en als koppeling tussen de methode en de theorie. De onderzoeksmethodologie en de opbouw van het model wordt toegelicht in *hoofdstuk 4*. In dit hoofdstuk is een verdeling gemaakt tussen drie segmenten, namelijk: pre-processing, processing en post-processing. In het kort is "Processing" het onderdeel waar het model een afbeelding analyseert en hier objecten in detecteert. Met het onderdeel "Pre-processing" wordt bedoeld op de stappen die worden ondernomen voorafgaand aan "Processing". Een voorbeeld hiervan is het geschikt maken van data voor het model. Het onderdeel "Post-processing" komt na "Processing" en heeft als doel om de resultaten te analyseren en interpreteren. Verder worden in *hoofdstuk 5* de resultaten benoemd. *hoofdstuk 6* en *hoofdstuk 7* zal ingaan op de conclusie en discussies. Tot slot zal in *hoofdstuk 8* de adviezen en aanbevelingen worden gepresenteerd.

Theoretisch kader:

Inzichten en concepten voor de probleemcontext

02

Dit onderdeel van het onderzoek heeft als doelstelling om de (wetenschappelijke) literatuur gericht op dit onderzoek toe te lichten. De inzichten die hieruit voort zijn gekomen hebben de basis gevormd voor bepaalde keuzes die wij hebben gemaakt. Daarnaast heeft het als doelstelling gehad om het onderzoek in bredere context te plaatsen.

2.1 Register Externe Veiligheidsrisico's (REV)

Het Register Externe Veiligheidsrisico's, ook wel het REV genoemd, is een database bestaande uit diverse activiteiten met een potentieel risico voor de omgevingsveiligheid. Deze ingewonnen data werd voorheen centraal geïnventariseerd in het Risico Register Gevaarlijke Stoffen (RRGS), maar met de ingang van de Omgevingswet is dit vervangen door het Register Externe Veiligheidsrisico's (REV). De data in dit register is gericht op verschillende elementen in de leefomgeving, zoals de opslag van gevaarlijke stoffen en transport. Ter bevordering van transparantie wordt informatie hieruit gepubliceerd in bijvoorbeeld de Atlas leefomgeving. De centrale opslag van data op nationale schaal heeft als doel om de informatie betrouwbaar en actueel te waarborgen (Ministerie van Infrastructuur en Waterstaat, z.d.-a). Het verzamelen van de data in het REV gebeurt in samenwerking met (gemandateerde) bronhouders, bestaande uit de omgevingsdiensten, Provincies en Gemeenten (Ministerie van Infrastructuur en Waterstaat, z.d.-b). Deze data dient inzage te bieden in wat bepaalde risicovolle activiteiten kunnen betekenen voor de leefomgeving en de mensen (Dorlandt, 2022).

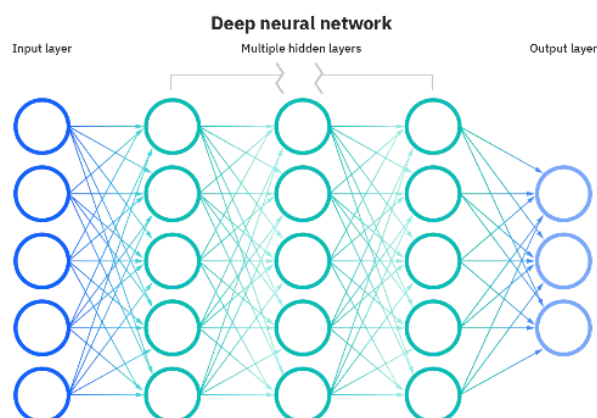
2.2 Artificial Intelligence (AI)

De automatische beeldverwerking van satellietbeelden en/of luchtfoto's is mogelijk door de toepassing van *computer vision*. Dit is de verzamelnaam voor methodieken gericht op het analyseren en extraheren van digitale beelden. De techniek 'object detection' binnen computer vision is gericht op het accuraat lokaliseren, detecteren en labelen van objecten. De genoemde methoden zijn onderdeel van de overkoepelende term voor zelflerende systemen (AI). Voordat de theorie en toepassingen van object detectie modellen verder wordt toegelicht is het van belang in te gaan op de basisprincipes van AI.

Ondanks dat deze begrippen vaak dezelfde definitie toegewezen krijgen zijn er duidelijke verschillen. Om te beginnen is AI gericht op het nabootsen van menselijke intelligentie door middel van technologie. ML is een onderdeel van AI voornamelijk gericht op het leren van ingegeven data om analytische modellen te automatiseren met minimale menselijke interventie. Binnen ML zijn er ook nog diverse sub domeinen, zoals; supervised- en unsupervised Machine Learning (IBM, z.d.). Omdat deze geen directe betrekking hebben tot dit onderzoek zal hier verder niet op in worden gegaan. Tot slot is er DL, een onderdeel van ML, waar ook gebruikt wordt gemaakt van data om te leren. Het verschil is dat er voor DL gebruik wordt gemaakt van neurale netwerken bestaande uit meerdere lagen. Deze neurale netwerken, ook bekend als Artificial Neural Networks (AANs), vormen de basis voor DL modellen. Het is technisch gezien een netwerk aan punten die aan elkaar zijn gekoppeld en op basis van trainingsgegevens zichzelf kan verbeteren, zie figuur 3 (IBM, 2022).

Naast de technische infrastructuur verschillen ML en DL met elkaar op het gebied van complexiteit. Het eerste verschil is gericht op de manieren hoe de modellen leren en de hoeveelheid data die hier voor nodig is. Om de nauwkeurigheid van een DL model te verbeteren is een grote hoeveelheid data meer van belang dan bij ML. Dit komt doordat DL op geautomatiseerde wijze patronen herkent en op basis hiervan bijvoorbeeld afbeeldingen kan categoriseren op basis van verschillen en gelijkenissen. Bij een ML model is dit minder belangrijk omdat het gebruikmaakt van datastructuren die minder afhankelijk zijn van de hoeveelheid data (IBM, 2022)

Een ander belangrijk verschil tussen DL en ML is de hoeveelheid menselijke interventie op het gebied van feature extraction (IBM, 2022). De prestaties van ML hangt voor een groot deel af van de vooraf gedefinieerde features (Janiesch et al., 2021). Dit komt omdat het verschillen in data-input probeert te begrijpen op basis van de feature hiërarchie bepaald door menselijke experts. Hiervoor zijn meestal gestructureerde datasets nodig. DL kan ook gebruikmaken van gelabelde datasets maar het is geen vereiste (IBM, 2022). DL kan met minimale interventie van mensen features automatisch herkennen. Dit geeft het ook de mogelijkheid om te werken met grote hoeveelheden ongestructureerde data, zoals teksten en afbeeldingen (Janiesch et al., 2021).



Figuur 3, schematisch voorbeeld van deep learning netwerk (IBM, 2022)

2.2.3 Introductie computer vision

In figuur 2 is te zien dat naast de domeinen AI, ML en DL ook Computer Vision (CV) staat afgebeeld. CV, onderdeel van het domein Visual AI, is gericht op het analyseren van digitale beelden om hier inzichten uit te verkrijgen. Deze technologie vormt daarmee de basis voor object detectiemodellen. Waar AI een computer het vermogen geeft tot het simuleren van nadenken, geeft CV een computer de mogelijkheid de wereld te observeren. Het verschil tussen de observaties van mensen en computers is de nauwkeurigheid en objectiviteit van de resultaten. Waar de perceptie van zicht bij mensen kan verschillen is dit bij CV consistent, bijvoorbeeld bij de inschatting hoe ver objecten van elkaar zijn verwijderd. Om deze reden is CV goed toepasbaar op Remote Sensing beelden (IBM, z.d.). Een valkuil hierbij wel is het feit dat deze methode geen context kan begrijpen en daardoor keuzes maakt op basis van de data die het ontvangt. In situaties die afwijken van deze data kan het model objecten missen die voor mensen duidelijk zichtbaar zijn. Zo heeft onderzoek bijvoorbeeld aangetoond dat CV in combinatie met ML de mogelijkheid geeft schade van gebieden te analyseren op basis van Remote Sensing (Satellite Imaging Corp, z.d.). De mogelijkheden hiervan worden ook versterkt door verbeteringen in sensor- en satelliettechnologie (hoge optische resolutie & temporele resolutie). Ontwikkelingen hiervan worden verder toegelicht in paragraaf 2.6.1.

2.3 Object detectiemodellen

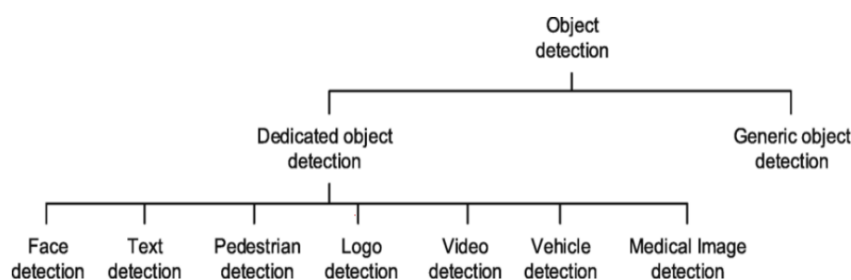
2.3.1 Introductie object detectiemodellen

Een object detectiemodel is een methode binnen Computer Vision dat gericht is op het detecteren van specifieke elementen uit digitale beelden (digitale foto's en video's). Concrete voorbeelden hiervan zijn bijvoorbeeld personen, gebouwen en auto's. In de huidige samenleving is dit van belang door de hoeveelheid aan visueel beeldmateriaal dat verzameld wordt. Om deze informatie op nauwkeurige en efficiënte wijze te analyseren kunnen object detectiemodellen worden toegepast (Namdev et al., 2022). Dit is mogelijk door de combinatie van zowel lokalisatie als classificatie binnen de domeinen van ML en DL (Joiya, 2022). Ondanks dat de toepassing van object detectiemodellen steeds relevanter wordt voor verschillende sectoren blijft het een complex thema binnen CV. Dit komt door het feit dat afbeeldingen niet eenduidig zijn. Digitale beelden bestaan namelijk uit verschillende componenten, kleuren, contrasten, resoluties, texturen en vormen. AI kan deze elementen gecombineerd tonen. Een beeld die voor het menselijke oog direct te herkennen is, maar niet voor een computer (Dixit et al., 2019). Dit wordt verklaard door het feit dat mensen de wereld van nature visueel observeren. Hierdoor is het herkennen van een object met een afwijkende positionering of kleurstelling eenvoudig (Joiya, 2022). Bovendien, waar mensen de wereld driedimensionaal observeren doen computer dit met slechts twee dimensies. Dit heeft invloed op de

manier waarop afstanden, oriëntaties, locaties, ruimtes en groottes van objecten op beelden kunnen worden gedetecteerd (Dixit et al., 2019).

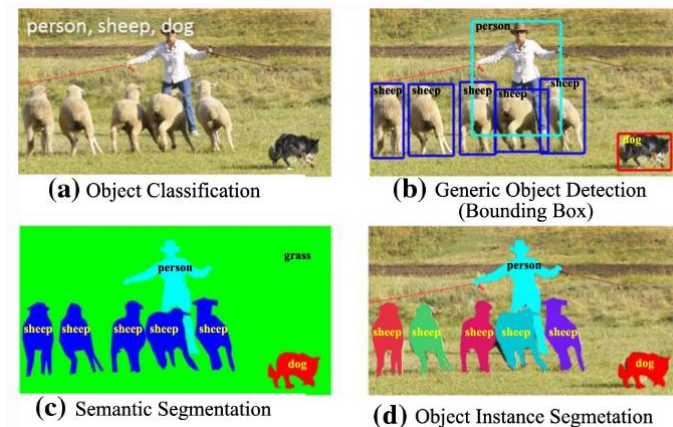
2.3.2 Dedicated- en generic object detection

Ondanks dat objectdetectie, in paragraaf 2.3.1, relatief eenduidig wordt beschreven is dit in de werkelijkheid niet het geval. Om te beginnen kan object detectie worden onderverdeeld in twee subonderdelen, namelijk: *dedicated object detection* en *generic object detection*, zie figuur 4. Het verschil hiertussen is dat generic object detection is gericht op het detecteren van objecten op basis van globale classificaties. Met eenvoudige classificaties wordt hier bedoeld op zaken als: personen, katten, auto's, gebouwen. Dedicated object detection detecteert specifieke instanties van een object, zoals: Donald Trump (persoon), Eiffeltoren (gebouw), BMW M3 (auto) (Liu et al., 2020). Dedicated object detection kan worden gezien als verdieping van generic object detection. Het is daardoor mogelijk van generic detection naar object detection te gaan. Dit proces is complex omdat voor iedere variatie een aparte klasse nodig is voor elke categorie. Bij het detecteren van specifieke beroemdheden zou er voor ieder persoon een klasse moeten worden aangemaakt met voldoende representativiteit en diversiteit. Dit maakt deze opzet complex en arbeidsintensief (Zhang et al., 2012).



Figuur 4, schematisch voorbeeld van dedicated- en generic detection (Liu et al., 2020).

Naast deze verschillende subonderdelen van object detectie zijn er ook nog verschillende methoden om dit te doen, zie figuur 5. Om te beginnen is er object classificatie, dit is de meest globale variant waarin de elementen aanwezig in een digital beeld wordt geïnventariseerd (a). Generic Object Detection in combinatie met een *bouding box* heeft als doelstelling om een vierkant te plaatsen rondom het specifieke object en hier een label aan te koppelen (b). Een verdieping op deze relatief globale lokalisaties en classificaties zijn segmentaties (c)(d). Het belangrijkste verschil tussen detectie en segmentatie is dat detectie voornamelijk gericht is op globaal lokaliseren van gedetecteerde objecten en segmentatie op het uitlijnen van deze objecten. Segmentatie is daarmee een meer geavanceerde uitwerking van object detectie, maar in de meeste instanties niet noodzakelijk voor het behalen van de gewenste resultaten (Liu et al., 2020).

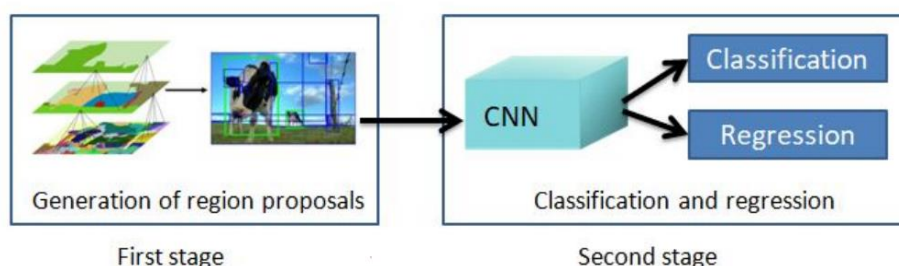


Figuur 5, object detectie en segmentatie

2.3.3 One-stage & two-stage modellen

Binnen object detectie zijn er overigens meerdere categorieën waarin modellen kunnen worden geplaatst. In de literatuur wordt uitgegaan van twee object detectie algoritmen, deze zijn als volgt:

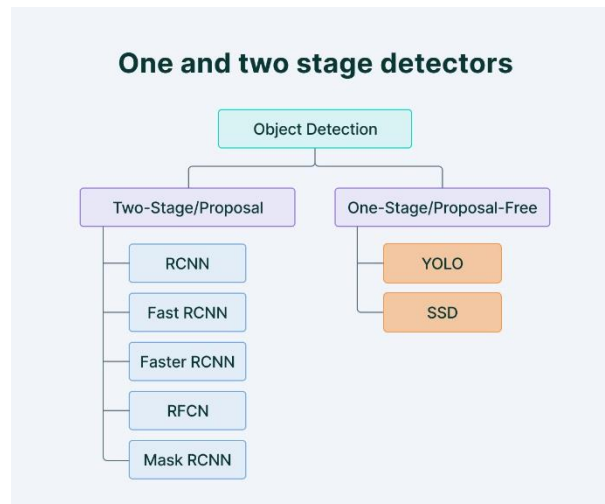
- *Two-stage detection*:
 - Een model gebaseerd op two-stage detection wordt uitgevoerd in twee verschillende fasen. De eerste fase bestaat hierin uit het selecteren van de gedetecteerde objecten uit een beeld, wat resulteert in een *generation of region proposals* (Sultana et al., 2020). In de tweede fase worden deze selecteerde objecten geclassificeerd door middel van een neurale netwerk. Het gevolg van deze twee fasen is de langere tijdsduur, aangezien voor ieder geselecteerd object uit de eerste fase een classificatie moet worden uitgevoerd in de tweede fase. De uitwerking van deze fasering is te zien in figuur 6 (Namdev et al., 2022).



Figuur 6, two-stage detection (Sultana et al., 2020)

- *One-stage detection*:
 - Bij een model gebaseerd op one-stage detection is slechts één cyclus waarin een classificatie en een regressie analyse wordt uitgevoerd. Er worden dus geen vooraf geselecteerde interessegebieden opgesteld wat leidt tot een efficiënter proces en daarmee een reductie van de tijdsduur die het nodig heeft objecten te detecteren (Namdev et al., 2022).

Per detectie categorie zijn er diverse modellen beschikbaar, zie figuur 7. Ieder model heeft andere uitdagingen, limitaties en voordelen die staan toegelicht in 2.3.5.



Figuur 7, overzicht van modellen (Sabina et al., 2022)

2.3.4 Mogelijkheden, limitaties en uitdagingen

2.3.4.1 Accuraatheid parameters

Om een zo effectief mogelijk object detectiemodel op te stellen is het van belang de mogelijkheden, limitaties en uitdagingen toe te lichten. Om te beginnen kan de effectiviteit van een object detectiemodel worden gemeten door twee onderdelen te meten, dit zijn: de kwaliteit van de resultaten (accuraatheid) en in hoeverre dit efficiënt wordt gedaan (Liu et al., 2020). In hoeverre een model presteert wordt gemeten aan de hand van prestatieparameters (performance metrics). De meest gebruikte performance metric is de AP (Average Precision), deze combineert de *recall* en *precision* om te bepalen hoe accuraat een bepaald resultaat is. Deze parameters zijn opgebouwd uit verschillende componenten, dit zijn: True Positive (TP), False Positive (FP), False Negative (FN) en True Negative (TN). In het geval van TP heeft het model een object correct gedetecteerd. FP toont een incorrecte detectie van het model. Een FN toont aan dat het model een correctie heeft gemist. Tot slot toont de TN toont aan dat het model correct foutieve objecten overslaat. Precision, ofwel

nauwkeurigheid, is gericht op de hoeveelheid TP die voortkomen uit alle detecties. Recall is hierin gericht op de hoeveelheid TP die voortkomen uit de werkelijke objecten (ground-truths). De AP is gebaseerd op de nauwkeurigheid en recall en toont aan hoe het model presteert bij verschillende recall-waarden. Bij de mAP is kortweg het gemiddelde van de AP scores van alle categorieën in de dataset. Het selecteren van een bepaalde threshold waarde heeft consequenties op het model. Een hoge confidence threshold kan ertoe leiden dat meer objecten worden gemist en daardoor met FN detecties oplevert. Een lage confidence threshold kan overigens ook leiden tot meer FP detecties (Koech, 2023).

2.3.4.2 Voordelen en mogelijkheden

Het gebruik van object detectiemodellen opent mogelijkheden voor visuele analyses. Hieronder volgt een lijst met beknopte toelichtingen per onderdeel. Het is belangrijk te benoemen dat de technologie van object detectiemodellen voortdurend in ontwikkeling is. Dit betekent dat het ook in de toekomst van belang is om onderzoek te doen naar potentiële toepassingen van object detectiemodellen.

Snelheid (realtime prestaties)

Een belangrijk voordeel van object detectiemodellen is de snelheid waarmee objecten worden gedetecteerd. Deze snelheid is niet specifiek voor object detectiemodellen maar voor AI in het algemeen. Wilczek (2022, p.54) stelt dat: *"Computers can work much faster than brains, because transistors use the electrically driven motions of electrons, instead of the much slower processes of diffusion and chemical change that neurons rely on. By this natural measure, the limiting speed of thought for artificial intelligence is roughly a billion times faster than the speed of thought for natural intelligence"*. De snelheid van objectdetectie maakt het mogelijk om realtime analyses uit te voeren. Dit is bijvoorbeeld een relevante toepassing bij het detecteren van auto's. De snelheid waarmee een model functioneert heeft daarmee voordelen voor een efficiënt analyseproces (Kim et al., 2020)

Accuraatheid (object lokalisatie en classificatie)

Naast snelheid is de nauwkeurigheid van object detectiemodellen een essentieel aspect. De nauwkeurigheid van een model wordt beoordeeld aan de hand van verschillende parameters benoemd in 2.3.4.1. Wanneer een model nauwkeurige resultaten produceert zorgt dit ervoor dat meer geavanceerde uitwerkingen toegepast kunnen worden (Liu et al., 2020). Dit zijn toepassingen als het segmenteren van objecten en *object tracking*. Een hoge nauwkeurigheid is voornamelijk binnen remote sensing van belang. Dit komt doordat in tegenstelling tot normale camerabeelden de meeste objecten in luchtfoto's relatief klein en daardoor moeilijker te detecteren zijn (Jain et al., 2021). Overigens zijn hier binnen objectdetectie grote ontwikkelingen in geweest. Deze uitdagingen en beperkingen zullen verder worden besproken in sectie 2.3.4.3. Naast de nauwkeurigheid waarmee objecten worden gedetecteerd is ook classificatie van belang.

Een detectiemodel gericht op voertuigen krijgt namelijk geen eenzijdige informatie, er zijn hierin verschillende klassen, zie figuur 8. Hierbij is het niet alleen essentieel dat een voertuig wordt gedetecteerd maar ook dat deze correct wordt geclassificeerd.

Labeling Name	TRAIN set	TEST set
car	1447	276
mini_van	244	55
big_van	33	8
mini_truck	516	163
truck	94	27
compact	286	39

Figuur 8, classificatie van voertuigen (Kim et al., 2020)

Praktische implicaties en brede toepasbaarheid

Een belangrijk voordeel van object detectiemodellen is de brede praktische toepasbaarheid. Deze vorm van AI wordt toegepast bij het detecteren van objecten in de fysieke leefomgeving, maar wordt ook in de medische wereld steeds relevanter (Yu & Yang, 2021). Een belangrijk aspect hierbij is de veelzijdigheid. Met één model kunnen verschillende objecttypen worden getraind, waardoor bijvoorbeeld mensen, dieren en voertuigen kunnen worden gedetecteerd. Dit is van belang bij nieuwe technologische ontwikkelingen zoals zelfrijdende auto's, geautomatiseerde video surveillance en robot vision. In de afgelopen twintig jaar heeft deze technologie grote ontwikkelingen doorgemaakt en de verwachting is dat dit alleen maar zal toenemen in de toekomst (Zou et al., 2023).

2.3.4.3 Uitdagingen & limitaties

Ondanks de mogelijkheden van object detectiemodellen zijn er een aantal uitdagingen en limitaties. Een aantal van deze limitaties hebben betrekking tot objectdetectie in het algemeen terwijl anderen zijn gespecificeerd op de toepassing van luchtfoto's. Om overzicht te behouden zijn deze onderverdeeld in verschillende klassen.

Input data

Limitaties van datasets

Datasets vormen een essentieel onderdeel van object detectiemodellen, aangezien het een model in staat stelt specifieke objecttypen te herkennen. Dit is de reden dat een kleine dataset de kwaliteit van de resultaten sterk vermindert. Dit ondanks het feit dat een deep learning modellen in theorie complexere taken uit kunnen voeren (Diwan et al., 2022). Omdat een grote dataset leidt tot de beste resultaten is het opstellen hiervan kostbaar en arbeidsintensief. Hoewel er in sommige gevallen openbare datasets beschikbaar kunnen zijn, is er geen garantie dat de trainingsdata

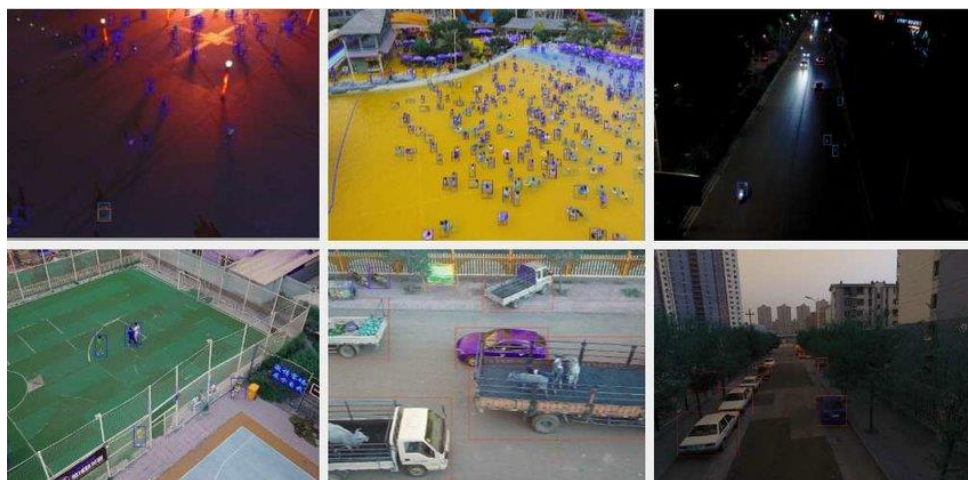
overeenkomt met nieuwe situaties of omgevingen. Overigens is niet altijd de hoeveelheid data maar ook de diversiteit belangrijk. Zo moet de trainingsdata bestaan uit verschillende perspectieven om alle dimensies van een object te kunnen herkennen in nieuwe situaties (Paullada et al., 2021).

Fysieke belemmeringen en lichtinval

Binnen het domein van objectdetectie vormen belemmeringen in het algemeen een uitdaging. Bij de analyse van luchtfoto's wordt dit probleem echter nog prominenter vanwege het perspectief. Zo kunnen bepaalde objecttypen moeilijk te detecteren zijn doordat de schaduw van een gebouw of boom erop valt. Bovendien hebben objecten verschillende kleuren, wat resulteert in verschillende reflectiewaarden. Een donkere auto in een onderbelichte situatie is bijvoorbeeld vrijwel niet detecteerbaar voor de meeste modellen. Het perspectief in combinatie met de afstand van de sensor tot het object vormen ook fysieke belemmeringen. Een klein object dat wordt geblokkeerd door een dicht bladerdak is bijvoorbeeld niet zichtbaar op standaard luchtfoto's (Jain et al., 2021).

Schaalvariaties en perceptie

Een model functioneert optimaal wanneer het wordt toegepast op situaties gelijkwaardig aan de trainingsdata. Een model dat is getraind op grote objecten zal daarom beduidend minder goed presteren op kleinere objecten (Diwan et al., 2022). Bij luchtfoto's met een bepaalde hoek zijn er grote variaties in schaal. Een auto heeft bijvoorbeeld een bepaalde dimensie. Wanneer een auto ver van de sensor is verwijderd verandert de schaal waardoor een auto op een foto kleiner kan zijn dan een fiets die dichtbij de sensor staat, zie figuur 9. Daarnaast hebben grotere objecten eigenschappen die duidelijker zichtbaar zijn. Bij een kleiner object verdwijnen deze eigenschappen relatief snel door de resolutie.



Figuur 9, schaalvariatie en perceptie (Diwan et al., 2022).

Ongelijke klassenverdeling en generalisatie van objecten

Wanneer een object detectiemodel wordt getraind op verschillende klassen is het van belang dat hier een gelijke verdeling in is. Wanneer bepaalde objecten voldoende aanwezig zijn in de trainingsdata zal deze beter presteren bij nieuwe situaties. Objecten die schaars zijn in de trainingsdata zal daardoor beduidend minder goed presteren. Deze disbalans kan leiden tot een model dat niet optimaal presteert (Diwan et al., 2022).

Lage resolutie

De lage resolutie van luchtfoto's heeft zowel betrekking op schaalvariaties als fysieke belemmeringen. In luchtfoto's met een relatief lage resolutie is het moeilijk om kleine objecten met weinig afwijkende eigenschappen te detecteren. Grotere objecten hebben hier minder last van omdat hun kenmerkende eigenschappen grotendeels behouden blijven. Dit is zelfs het geval bij kwaliteitsvermindering. Daarnaast is het ook het geval dat de object detectiemodellen zijn getraind op een bepaalde resolutie afbeelding. Wanneer de nieuwe situatie hiervan afwijkt leidt dit verminderde resultaten. Consistentie en trainingsdata met diverse omstandigheden is daarom van belang bij de ontwikkeling van een effectief object detectiemodel (Diwan et al., 2022).

Algemene limitaties object detectie

Onnauwkeurige lokalisatie/classificatie

Ondanks de relatief hoge nauwkeurigheid van bepaalde modellen blijft het een schatting. Zo is het mogelijk dat een object niet wordt herkend door kleine afwijkende fysieke eigenschappen. Daarnaast zijn de opgegeven begrenzingskaders uit trainingsdata niet exact nauwkeurig op het object. Hierdoor wordt bepaalde achtergrondinformatie meegenomen in het model wat kan leiden tot lokalisatiefouten. Dit probleem is niet van toepassing wanneer data-segmentatie wordt geïmplementeerd in tegenstelling tot *generic object detection*. Daarnaast hebben veel objecttypen gelijkwaardige fysieke eigenschappen wat tot foutieve detecties kan leiden (Diwan et al., 2022).

Technische randvoorwaarden

Computerkracht

Door de grote datasets die DL modellen nodig hebben is computerkracht een belangrijk onderdeel. Dit kan leiden tot bepaalde limitaties in bijvoorbeeld zelfrijdende auto's waar het moeilijk is om grafische kaarten te verwerken met hoog stroomverbruik. Ondanks deze mogelijke limitatie zijn er verschillende modellen en configuraties beschikbaar om de benodigde computerkracht te verminderen (Yu et al., 2018).

Praktische overwegingen

Ethiek en privacy overwegingen

Het laatste onderdeel dat een mogelijke limitatie is van object detectiemodellen zijn de implicaties die het kan hebben op privacy. Deze aspecten spelen voornamelijk een rol bij een model gericht op gezichtsherkenning en algemene surveillance. Een uitgebreid overzicht van de implicaties van object detectiemodellen in combinatie met luchtfoto's staat toegelicht in hoofdstuk 2.6.2 (Diwan et al., 2022).

2.3.5 Object detectiemodellen

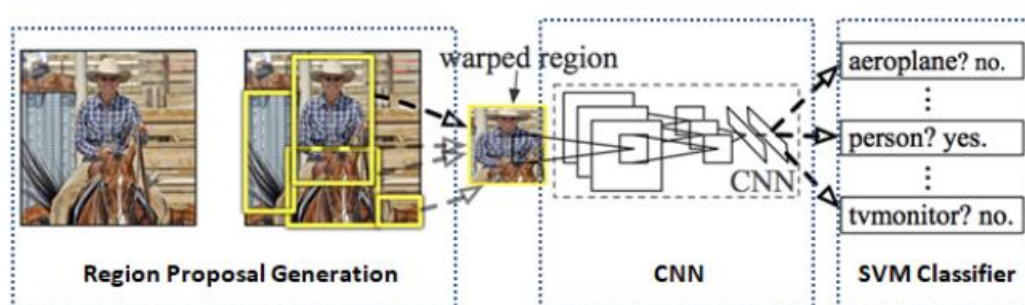
In paragraaf 2.3.3 is een overzicht gegeven van one-stage en two-stage detection modellen. In dit onderdeel zal ieder model, dat relevant is voor dit onderzoek, worden toegelicht aan de van literatuur. Deze uitleg bestaat uit een overzicht over hoe het model is opgebouwd en vervolgens zal worden ingegaan op hoe de prestaties van elkaar verschillen. Deze modellen zijn onderverdeeld onder 2.3.5.1 en 2.3.5.2.

2.3.5.1 CNN Family

Aan de basis van alle RCNN en YOLO object detectiemodellen staat de CNN family. Ondanks de verschillen tussen RCNN en YOLO (two-stage en one-stage) gebruiken deze het CNN netwerk als basis met een afwijkende technische infrastructuur. CNN staat voor Convolutional Neural Network en is gericht op het extraheren van kenmerkende eigenschappen uit een afbeelding. Dit gebeurt op basis van twee fasen (two-stages) (Dixit et al., 2019). In principe is deze technologie niet nieuw, het is namelijk al gestart in 1998. In de opvolgende decennia zijn de prestaties overigens sterk verbeterd door nieuwe ontwikkelingen, waardoor in 2015 het model zelfs beter scoorde dan mensen (Du, 2018). De opbouw van een CNN bestaat uit meerdere elementen. Dixit et al. (2019) beschrijft deze elementen als volgt: convolution layer, pooling layer en fully connected layer. De exacte innerlijke werking zal buiten beschouwing worden gelaten maar de doelstellingen van deze lagen zijn wel belangrijk. De convolution layer heeft als doelstelling om de high-level features van een input te extraheren. Tussen de uiterste twee lagen zit de pooling layer die als functie heeft om de ruimtelijke dimensies te verminderen van de feature maps (output van een laag in het netwerk). Dit leidt ertoe dat de benodigde computerkracht wordt verminderd. Tot slotte is er ook nog de fully connected layer. Deze laag heeft als doelstelling om de geanalyseerde data als output op te stellen en hierbij het proces te optimaliseren. Ondanks de mogelijkheden van CNN object detectiemodellen zoals de goede prestaties bij het detecteren van objecten in foto's zijn er ook zaken waar rekening gehouden mee dient te worden. Zo het nodig om over voldoende trainingsdata, correcte parameters voor objecten en voldoende computerkracht te beschikken (Du, 2018).

2.3.5.2 RCNN, Fast RCNN en Faster RCNN

De ontwikkeling van het RCNN is belangrijk geweest voor de totale CNN family omdat het heeft aangetoond hoe goed het presteert bij het extraheren en classificeren van objecten (Du, 2018). RCNN is opgebouwd op basis van de CNN family en bestaat uit drie verschillende onderdelen, zie figuur 10 (Sultana et al., 2020). De eerste module is gericht op het opstellen van Region Proposals op basis van opgegeven categorieën. Dit proces wordt ook wel *Selective Search* genoemd, wat op een afbeelding ongeveer 2000 keer wordt gedraaid. Dit draagt bij aan langere tijdsduuraties en niet altijd tot optimale resultaten (Srivastava et al., 2021). In de tweede module worden de geselecteerde objecten uit de eerste module geëxtraheerd. In de laatste module van het RCNN proces worden de gecategoriseerde objecten geclassificeerd (Dixit et al., 2019).



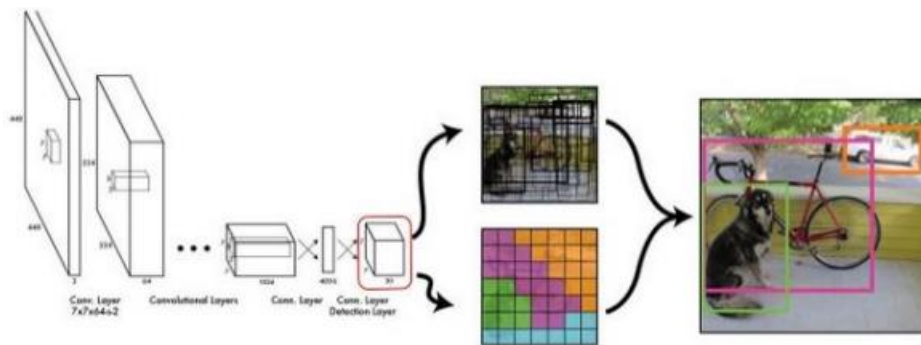
Figuur 10, region proposals (Sultana et al., 2020)

Fast RCNN past, net als RCNN, het Selective Search algoritme toe en gebruikt. Er is overigens een verschil in de toepassing ervan. Waar bij RCNN Selective Search ongeveer 2000 keer wordt toegepast op een afbeelding gebeurt dit bij Fast RCNN slechts één keer na de detectie van Region Proposals. Dit resulteert in significant lagere tijdsduuraties (Srivastava et al., 2021). Het belangrijkste verschil tussen RCNN en Fast RCNN is de mate van efficiëntie door verschillen in de technische infrastructuur. Bij RCNN worden afbeeldingen namelijk geanalyseerd op basis van pixels terwijl Fast RCNN dit doet op basis van een feature map (Dixit et al., 2019). Faster RCNN is de verbeterde versie van Fast RCNN op het gebied van detectiesnelheden, het is daardoor toepasbaar voor realtime objectdetectie (Yusro, 2022). Dit is mogelijk door een alternatieve toepassing van Selective Search waarbij een afbeelding slechts een aantal keer geanalyseerd wordt (Srivastava et al., 2021).

2.3.5.3 YOLO en SSD

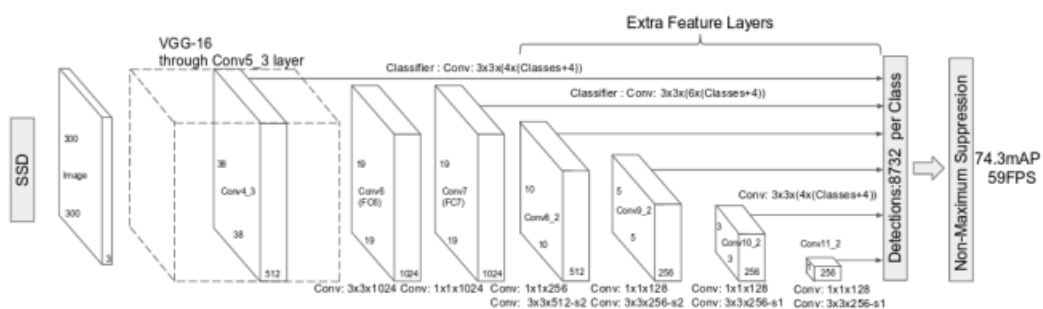
Binnen one-stage object detection zijn de twee voornaamste modellen: SSD (Single Shot Detection) en YOLO (You Only Look Once). Beide modellen zijn gericht op het efficiënt en nauwkeurig detecteren en lokaliseren van objecten. Beginnend bij YOLO. Dit detectiemodel bekijkt een afbeelding slechts één keer. Er wordt vervolgens een grid verdeeld over deze afbeelding, zie figuur

11. Voor elk vak in het raster wordt vervolgens bepaald of er een object in aanwezig. In het geval dat er een object aanwezig is wordt hier een bounding box omheen geplaatst met daarin een score over de waarschijnlijkheid van een voorspeld object. Dit proces wordt vervolgens herhaald voor ieder vak in het grid waardoor alle objecten worden gedetecteerd aanwezig op de afbeelding (Sultana et al., 2020).



Figuur 11, opzet van grid (Sultana et al., 2020)

Ondanks dat SSD ook een single shot detection systeem is heeft het een ander fundament en methodologie. Bij SSD wordt een totale afbeelding gebruikt als input voor het model. De afbeelding gaat vervolgens door verschillende lagen met verschillende resoluties en groottes, zie figuur 12. Er worden vervolgens patronen herkend die overeenkomende eigenschappen met getrainde objecten hebben. Vervolgens bepaald het model hoe accuraat de voorspelling is waarna de mindere voorspellingen niet worden meegenomen. Omdat het SSD model naar een afbeelding kijkt op verschillende schaalniveaus is het mogelijk om objecten van verschillende schaalniveaus te detecteren. Tot slot worden overlappende bounding boxes verminderd om te voorkomen dat meerdere boxes per object staan genoteerd (Sultana et al., 2020).



Figuur 12, lagen van model (Zhao et al., 2019)

2.3.6 Overzicht kwaliteit van modellen

In dit onderdeel zijn de prestaties van de verschillende object detectiemodellen geïnventariseerd aan de hand van relevante literatuur. Belangrijk hierbij is wel dat de detectiemodellen zijn getest op diverse digitale beelden en niet uitsluitend op luchtfoto's. Dit is gedaan om limitaties te voorkomen en ook inzichten te verkrijgen in randvoorwaarden, zoals de snelheden waarmee een model presteert. In onderstaande toelichting worden een aantal inzichten toegelicht.

Het YOLO object detectiemodel heeft verschillende versies (v2, v3, v4, v5, v6, v7, et cetera.). Het is overigens meer accuraat om deze versies als losse edities te zien. Dit komt omdat niet iedere versie een verbetering is op elk gebied. Later in dit hoofdstuk is bijvoorbeeld in te zien hoe het YOLOv5 model beter presteert dan het YOLOv6 model. Buiten deze observatie toont de wetenschappelijke literatuur geen eenduidige resultaten mede door het feit dat modellen als RCNN en SSD worden vergeleken met oudere edities van YOLO. In dit onderdeel zal daarom een aantal onderzoeksresultaten worden besproken.

In de publicatie van Yusro et al. (2022) is onderzoek gedaan naar de detectiemogelijkheden van overlappend keukengerei. De doelstelling hierbij is vooral om aan te tonen in hoeverre het model in staat is om objecten te classificeren die niet afzonderlijk van een ander object staat. Voor dit onderzoek zijn de algoritmen YOLOv5 en Faster RCNN toegepast. De resultaten hebben aangetoond dat het YOLOv5 model nauwkeuriger is in het plaatsen van de bounding boxes maar dat Faster RCNN meer detecties heeft gemaakt. De nauwkeurigheid in dit rapport voor Faster RCNN behaalde een mAP van 83.92% in tegenstelling tot YOLOv5 met een score van 89.12%. Het YOLO model is gebaseerd op single-stage detection wat normaliter leidt tot hogere snelheden maar lagere nauwkeurigheid. YOLOv5 is daarentegen zeer nauwkeurig en behaalt in sommige gevallen betere prestaties dan two-stage detectors.

Door de hoge snelheid van het YOLO model behaalt het hoge prestaties op real time applicaties. In het onderzoek van (Kim et al., 2020) is onderzoek gedaan naar de detectie en classificatie van voertuigen. Figuur 13 toont aan dat het YOLOv4 model beduidend betere resultaten oplevert. Het is overigens wel van belang om te benoemen dat real time situaties niet volledig representatief zijn voor dit project. Ondanks dit, is dit onderzoek gebruikt omdat het ook inzichten geeft in de mogelijkheden met geringe computerkracht. Dit onderzoek heeft namelijk aangetoond dat het model een hoge FPS behaalt met het gebruik van slechts één GPU, namelijk de GeForce RTX 2080ti. Het onderzoek van Namdev et al. (2022) heeft op basis van een literatuurstudie aangetoond dat het YOLO model in staat is op hoge snelheid objecten te detecteren met minimaal verlies van nauwkeurigheid. Een ander groot voordeel van het YOLO model heeft te maken met de relatief eenvoudige implementatie van de infrastructuur. Uiteindelijk is het YOLO model toepasbaar door de

balans tussen de hoge snelheid en het behoud van nauwkeurigheid. Deze eigenschappen maken het model zeer toegankelijk (Du, 2018).

Table 3 : Performance of YOLO v4 model

Label	Average Precision	True Positive	False Positive
car	98.08%	273	25
mini van	94.93%	52	5
big van	100.0%	8	0
mini truck	99.04%	162	4
truck	98.52%	27	5
compact	98.59%	36	1

Table 4 : Performance of Faster R-CNN model

Label	Average Precision	True Positive	False Positive
car	93.2%	262	17
mini van	87.2%	52	41
big van	100.0%	8	1
mini truck	99.7%	164	16
truck	100.0%	27	2
compact	80.3%	25	10

Table 5 : Performance of SSD model

Label	Average Precision	True Positive	False Positive
car	92.7%	257	34
mini van	84.3%	29	1
big van	87.5%	7	0
mini truck	97.9%	151	0
truck	94.9%	21	5
compact	85.8%	32	15

Figuur 13, prestaties van YOLOv4 (Kim et al., 2020)

Door uiteenlopende resultaten van de beschikbare literatuur is het kiezen van een model niet afhankelijk van één parameter. Het onderzoek van Sánchez et al. (2020) beschrijft de modellen als volgt:

YOLO en SSD zijn beiden snelle modellen door de one-stage detection infrastructuur. Deze snelheid maakt beiden modellen toepasbaar op videobeelden met een hoge FPS. SSD toont overigens wel duidelijk verminderde nauwkeurigheid in vergelijking met Fast-RCNN en Faster-RCNN. Voor YOLO geldt vrijwel hetzelfde behalve dat de nauwkeurigheid voor dit model relatief hoog is. Het voornaamste probleem van YOLO is hier de detectie van kleinere objecten. De RCNN modellen tonen verder in verschillende publicaties de hoogste nauwkeurigheid, maar hiervoor is wel een complexere technische infrastructuur voor nodig die beduidend lager in snelheid is.

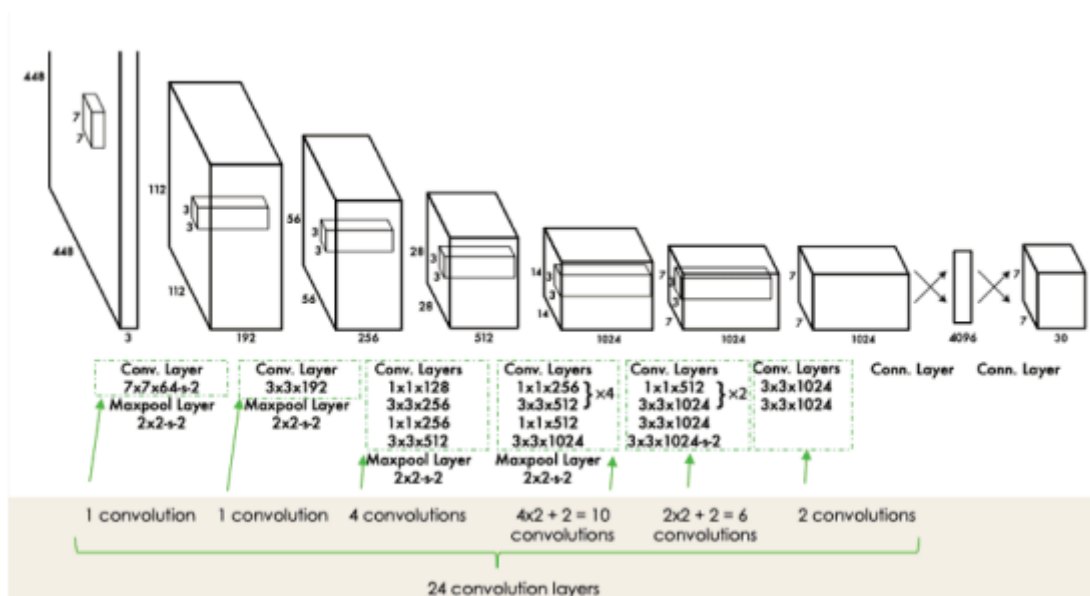
Op basis van de bestaande literatuur is gekozen om de mogelijkheden van het YOLO model verder te verkennen. Hiervoor is gekozen omdat het YOLO consistente resultaten laat zien die zowel snel als nauwkeurig zijn. Niet alleen deze balans maar ook de toegankelijke infrastructuur is in acht genomen om de doorontwikkeling mogelijk te maken voor de opdrachtgever.

2.4 YOLO

2.4.1 Achtergrond YOLO-series

YOLO wat staat voor *You Only Look Once*, beheert de eigenschap om object detectie toe te passen door een afbeelding slechts één keer aan een neurale netwerk te serveren. CNN heeft de mogelijkheid om digitale beelden *realtime* te analyseren. YOLO gebruikt dit netwerk om objecten in *realtime* te detecteren. Wat YOLO overigens zo baanbrekend maakt is de hoge snelheid en het behoud van nauwkeurigheid die het model bereikt. Dit is mogelijk door de unieke infrastructuur die het YOLO-model heeft (Palla, T. S. et., 2023). Deze baanbrekende realtime eigenschap van YOLO maakte het mogelijk deze techniek integraal toe te passen in verschillende praktijken. YOLO wordt al toegepast in een breed scala aan werkvelden waaronder het medische werkveld, beveiligingsdiensten en transport (zelfrijdende auto's) (Terven, R. J. et al., 2023).

CNN maakt dus een groot onderdeel uit van de totale YOLO-architectuur. En maakt daardoor gebruik van zogenoemde convolutionele lagen. Het aantal convolutionele lagen die geactiveerd worden bij het detectieproces is afhankelijk van de YOLO-architectuur. In figuur 14 is een volledig gevormd convolutieel netwerk van 24 convolutionele lagen afgebeeld (Palla, T. S. et., 2023)



Figuur 14: YOLO architectuur (Palla, T. S. et., 2023)

2.4.2 YOLO-architectuur (backbone, neck en head)

Een YOLO-architectuur bestaat uit drie hoofdcomponenten. De *backbone*, *neck* en *head*. Voor al deze componenten vervullen de eerder benoemde convolutionele lagen bepaalde functies. De backbone verzamelt de benodigde onderdelen uit de afbeelding en stuurt deze door naar de *neck*. De *neck* dient als koppeling tussen de backbone en de *head*. In de *neck* worden verschillende pre-processing functies zoals *up-sampling*, ruisreductie en augmentatie toegepast, ook worden in dit onderdeel andere convolutionele lagen zoals FPN of SPP (afhankelijk van het framework) ingezet om de representatie van de data te verbeteren. Het hoofd van het netwerk (de *head*) ontvangt de getransformeerde data uit de neck, voert de detectie over de afbeelding en finaliseert de output (Palla, T. S. et., 2023). Alle YOLO-iteraties die door de jaren heen zijn gepubliceerd maken gebruik van dezelfde *backbone*, *neck* en *head* structuur. De opbouw van deze componenten daarentegen verandert per iteratie.

2.4.3 Evolutie YOLO varianten

Sinds de eerste publicatie van YOLO zijn de ontwikkelingen binnen het computer vision en deep learning domein bijzonder snel gegaan. Dit onderdeel zal kort de evolutie van de YOLO-varianten toelichten. De inzichten die verkregen worden in dit onderdeel kunnen verder dienen als ondersteuning bij het bepalen van welke YOLO versie voor dit onderzoek het best geschikt is.

In 2016 werd de eerste variant gepubliceerd door Joseph Redmon. Na de publicatie van het oorspronkelijk YOLO-algoritme zijn verschillende varianten ontwikkeld. De eerste drie verbeterde varianten (YOLO, YOLOv2 en YOLOv3) zijn ontwikkeld door J. Redmon. Vanaf de vierde generatie in de YOLO-serie nam A. Bochkovskiy het werk over. Een maand nadat A. Bochkovskiy YOLOv4 had vrijgegeven, publiceerde de onderzoeker, Glen Jocher, namens Ultralytics, onafhankelijk van de oorspronkelijke uitgevers, een "vijfde generatie" van de YOLO series. YOLOv5 wijkt af van de voorgaande varianten. De vorige generaties gebruikten het zogenoemde Darknet framework dat geschreven is in de programmeertaal C. Daarentegen is YOLOv5 geschreven in Python en maakt gebruik van het zogenoemde PyTorch framework. Hoewel de runtime efficiëntie van de programmeertaal C over het algemeen sneller is, overstijgt YOLOv5 hierin zijn voorgangers (Thuan, Do 2021).

De zesde generatie in de YOLO-series is ontwikkeld in 2022 door een groep Chinese wetenschappers (Li, C. et al., 2022). Deze iteratie biedt verbeterde accuraatheid ten opzichte van voorgaande varianten maar toont gebrekkigheid aan op het gebied van opschalingsmogelijkheden en gebruiksgemak (Thuan, Do 2021). De toepassing van deze versie is complex vanwege de taalbarrière in de documentatie (Meituan team, 2022). Ook heeft YOLOv6 een limitatie met betrekking tot multi-scale detectie (verschillende formaten), met name kleine objecten (Li, Z. et al.,

2023). YOLOv7 is de laatste variant die met goedkeuring van de oorspronkelijke uitgever (Joseph Redmon) door A. Bochkovskiy gepubliceerd is (Dewi, C. et al., 2023). Hoewel het voorgaande model (YOLOv4) van deze uitgever (Alexey Bochkovskiy) in C is geschreven, is deze variant gebouwd door middel van Python en maakt net als YOLOv5 gebruik van het PyTorch framework (Wong, K.Y. 2021). De uitgever van YOLOv7 claimt van alle voorgaande varianten de beste prestaties op basis van accuraatheid en snelheid te behalen (Wang, C. et al., 2022). In januari van 2023 is YOLOv8 gepubliceerd door Ultralytics onder de naam van Glen Jocher. Aangezien dit model dusdanig recent is gepubliceerd en Ultralytics ten tijden van schrijven nog geen officieel artikel heeft gepubliceerd, wordt tijdens dit onderzoek niet verder ingegaan op YOLOv8 (Ultralytics. 2023).

2.4.4 Technische vooruitgang

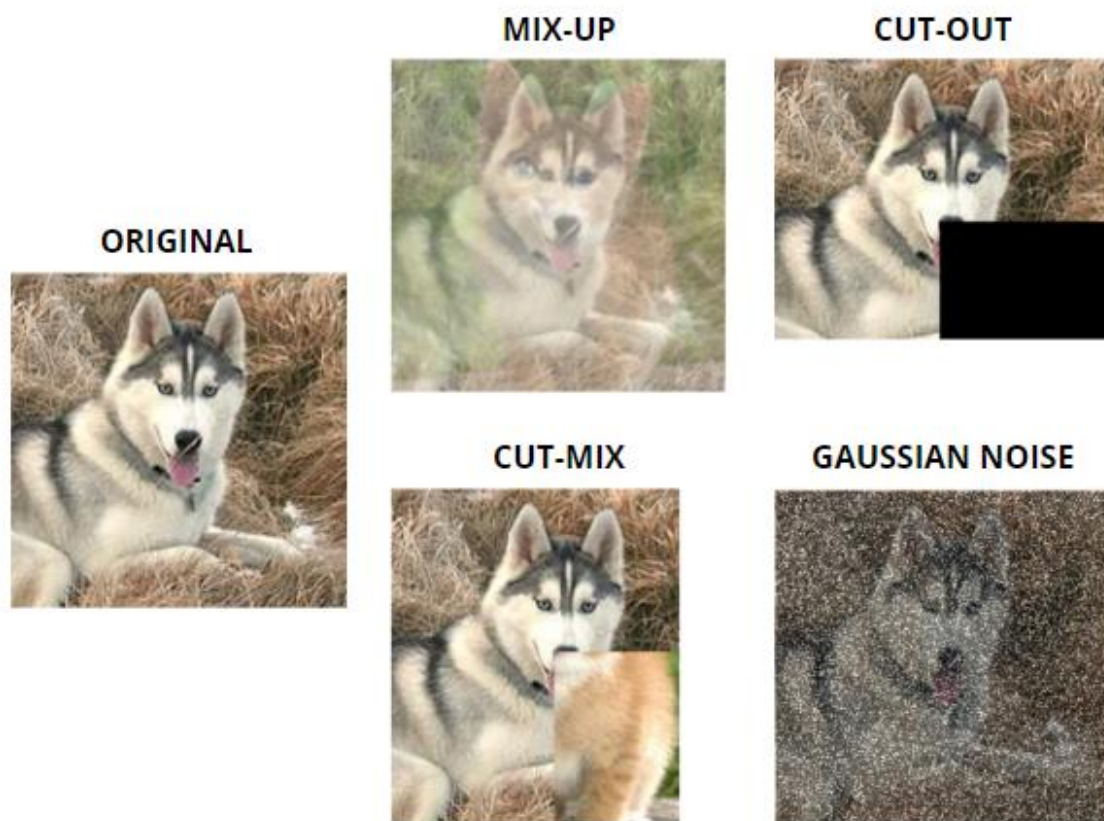
In deze alinea zal verder worden ingegaan op de technische verschillen met betrekking tot de diverse YOLO-varianten. Hierin worden de eerste drie iteraties niet verder belicht aangezien in het vorige hoofdstuk is aangetoond dat deze prestaties minder goed zijn.

Omdat YOLOv4 en YOLOv5 zo kort na elkaar ontwikkeld zijn (Maart – April, 2020), is er geen sprake van baanbrekende vernieuwingen tussen deze iteraties. Dat gezegd te hebben, zijn er wel degelijk noemenswaardige veranderingen tussen deze twee varianten. De meest invloedrijke verandering in YOLOv5 ten opzichte van YOLOv4 is wellicht de functie *Adaptive anchor boxes*. Het concept achter *anchor boxes* werd voor het eerst geïntroduceerd in YOLOv2. Deze functie genereert meerdere boxen over het voorspelde object. Vervolgens wordt de beste passende box behouden. Tot YOLOv4 zijn vormen van deze *anchor boxes* op basis van de MS COCO dataset *pre-trained* (bevat 80 verschillende objecten). De vijf meest geschikte boxen zijn ingesteld als de standaard *anchors*. Wanneer een object getraind wordt dat niet behoort tot één van de 80 objecten uit de MS COCO dataset kan het model moeite hebben met het creëren van een correcte bounding box. Dit kan ten koste gaan van de nauwkeurigheid van het model (Thuan, Do 2021).

YOLOv5 heeft als vernieuwende functie *Adaptive anchor boxes* beschikbaar gesteld. De vorming van *anchor boxes* is hiermee niet meer afhankelijk van de voor getrainde COCO *anchors*. Deze functie bepaalt de vijf beste *anchors* op basis van het betreffende object gebruikmakend van een k-means regressie tijdens het trainen van YOLOv5 (Thuan, Do 2021).

In YOLOv4 wordt het onderdeel *bag of freebies* geïntroduceerd. *Bag of freebies* verwijst naar een lijst van functionaliteiten die de prestatie en accuraatheid van het model bevordert zonder bijkomende vereiste rekenkracht. Deze optimalisaties bevatten o.a. fotometrische data-augmentatie waaronder het aanpassen van schaling, helderheid, contrast, hue, saturation en gaussian noise in de trainingsdataset. Andere (transformerende) augmentatie-technieken in YOLOv4 betreffen cutmix en mosaic (Wan, D. et al., 2023). Deze augmentatie-technieken vervangen uitsneden van originele

afbeeldingen en voegen deze samen om nieuwe afbeeldingen te creëren. Op deze manier wordt de originele dataset verrijkt met gevarieerde scenario's zonder dat nieuwe data nodig is. Dit verbetert het model waardoor deze meerdere contexten kan interpreteren wat het bereik van de detectie-capaciteit bevordert. Deze augmentatie-technieken zijn deels ook doorgevoerd in YOLOv5, hoewel gaussian noise en cutmix in YOLOv5 vervangen zijn door flip up-down/ left-right en mixup (Ultralytics. 2022-b) zie figuur 15.



Figuur 15, Augmentatie technieken; Mix-up, cut-out, cut-mix en gaussian noise

YOLOv7 heeft een verbeterde “trainable” *bag of freebies* geïntroduceerd en een state-of-the-art architectuur genaamd E-ELAN (*Extended efficient layer aggregation networks*) toegepast. De integratie van de E-ELAN architectuur expandeert, verschuift en combineert bepaalde onderdelen tijdens het leerproces. Deze manier van werken kan het gebruik van parameters verbeteren.

2.4.5 procesmatig verbetering

YOLOv5 en YOLOv7 zijn overgestapt van het oorspronkelijke Darknet framework naar de high-level Python framework genaamd PyTorch. De grootste verbetering van PyTorch is de eenvoudigheid waarmee het model getraind kan worden (Jiang. P. et al., 2022). Daarnaast is de implementatie van eigen code en het aanpassen van source-code minder complex ten aanzien van voorgaande varianten. Dit framework maakt het mogelijk om augmentatie-technieken aan te passen en te integreren, waardoor modellen gemakkelijker geoptimaliseerd kunnen worden. YOLOv5 is hierin uniek dankzij de integratie van de albumentation Python extensie (Albumentations AI, z.d.). Dankzij de actieve Python gemeenschap wordt er actief geëxperimenteerd met de verschillende YOLO-varianten. Hierdoor ontstaan doelgerichte afsplitsingen ten opzichte van de originele source-code. Dit leidt tot de ontwikkeling van nieuwe modellen gericht op specifieke doelstellingen. Een concreet voorbeeld hiervan is YOLOv5 OBB. Dit is een YOLOv5 toepassing specifiek voor georiënteerde bounding boxes zie figuur 16. Deze methode kan optimalisatie bieden wanneer sprake is van geclusterde en relatief kleine objecten. Wan, D., et al (2023); schrijven in hun onderzoek over een interessante ontwikkeling. Deze ontwikkeling is specifiek gericht op grootschalige object detectie met betrekking tot hoge resolutie luchtfoto's, gebruikmakend van YOLOv5. Daarentegen heeft YOLOv7 ook verschillende interessante ontwikkelingen op dit gebied.

Ultralytics en PyTorch hebben beide een eigen implementatie voor object detectie op platformen zoals IOS en Android-systemen inzetbaar gemaakt voor YOLOv5 (PyTorch, z.d. & Ultralytics. 2022.-c). Voor YOLOv7 is dit echter niet mogelijk met het gebruik van het Pytorch framework. Om hetzelfde te bereiken met YOLOv7 zijn een aantal extra stappen vereist waaronder een conversie naar een ander framework (Coste, S. 2022). Voor YOLOv6 zijn geen mogelijke opties gevonden voor integratie naar de eerdergenoemde platformen.



Figuur 16, (Wan et al., 2023);

2.4.6 Prestatie gedreven verbetering

De literatuur bevat veel studies waar YOLO-varianten met elkaar vergeleken worden. Hoewel dit soort studies kunnen helpen bij het selecteren van het juiste model, zijn de resultaten vaak uiteenlopend. Dit komt doordat een object detectie model voor iedere situatie anders reageert. Zodat resultaten uit andere studies gelijkwaardig met elkaar vergeleken kunnen worden, is alleen gezocht naar literatuur waar object detectie is toegepast op basis van hoge resolutie datasets. Datasets als HRRSD, SIMD, NWPU VHR-10 en RSOD zijn vergelijkbaar met de dataset waar dit onderzoek mee gewerkt zal worden (Wan, D. et al., 2023). Om deze reden zijn alleen de resultaten uit studies die deze datasets hebben toegepast meegenomen in de afweging voor het detectie model. Voor dit literair onderzoek zijn drie studies vergeleken op basis van mAP en snelheid/ms.

Hoewel YOLOv7 in veel gevallen voor zowel snelheid/ms als accuraatheid (mAP) betere resultaten aantoont, is dit niet altijd het geval. Zoals in de vorige alinea is aangegeven kunnen deze resultaten verschillen per situatie. Daarnaast kan ook de gebruikte hardware invloed hebben op de resultaten. Hardware (voornamelijk de GPU en het geheugen hiervan) beïnvloedt vooral de snelheid waarmee het model berekeningen kan uitvoeren. Verder is het aantal afbeeldingen waarmee het model getraind kan worden bepalend voor de resultaten. De volgende drie onderzoeken zijn gebruikt om de prestatie van verschillende YOLO-varianten te testen; KCFS-YOLOv5: A High-Precision Detection Method for Object Detection in Aerial Remote Sensing Images (**a**) (Tian, Z. et al., 2023), Yolo-Based Improvements in Remote Sensing Image Applications (**b**) (Zhang, Y. et al., 2022) en YOLO -HR: Improved YOLOv5 for Object Detection in High-Resolution Optical Remote Sensing Images (**c**) (Wan, D. et al., 2023).

Onderzoek **a** vergelijkt Fast-RCNN, YOLOv3, YOLOv4, YOLOv5 en maakt daarbij een vergelijking met het, door de studie geïntroduceerde, KCFS-YOLOv5 model. Verder heeft dit onderzoek gebruik gemaakt van de hardware en software configuraties afgebeeld in figuur 17. Dit onderzoek heeft de prestatie van deze object detectie modellen getest op basis van de NWPU-VHR-10, RSOD en UCAS-AOD-CAR (Tian, Z. et al., 2023). Deze resultaten zijn afgebeeld in figuur 18.

Onderdeel	Type	Geheugen
GPU	Nvidia gtx 2060	8 gb
CPU	AMD ryzen 7 5800H	-
Schijfgeheugen	-	-
OS	-	-
Framework	Pytorch	-

Figuur 17, (a) hardware configuratie (Tian, Z. et al., 2023).

Dataset	Evaluation Metrics	Faster-RCNN	YOLOv3	YOLOv4	YOLOv5	Ours
NWPU VHR-10	Map (%)	81.39	82.29	86.31	91.08	94.21
	Precision (%)	90.23	88.12	89.97	91.08	94.20
	Recall (%)	85.36	84.43	86.68	86.26	89.42
	parameters (M)	60.684	61.949	62.725	7.072	7.341
	Latency (ms)	139.3	24.8	23.3	18.4	22.8
	FPS	7.2	40.3	42.9	54.3	43.9
RSOD	Map (%)	82.88	84.73	86.62	90.67	95.27
	Precision (%)	86.46	87.14	87.68	90.59	96.11
	Recall (%)	85.41	86.18	86.59	89.56	93.05
	parameters (M)	60.684	61.949	62.725	7.072	7.341
	Latency (ms)	126.4	23.7	22.9	17.7	22.6
	FPS	7.9	42.2	43.7	56.5	44.2
UCAS-AOD-CAR	Map (%)	84.54	81.66	85.03	86.48	90.13
	Precision (%)	84.75	83.47	84.15	85.30	89.80
	Recall (%)	82.81	83.38	83.11	83.36	86.51
	parameters (M)	60.684	61.949	62.725	7.072	7.341
	Latency (ms)	114.8	23.1	22.3	16.5	21.4
	FPS	8.7	43.3	44.8	60.6	46.7

Figuur 18, (a) Vergelijking mAP van objecten voor YOLOv3, YOLOv4 en YOLOv5 afkomstig uit drie verschillende datasets (Tian, Z. et al., 2023).

Voor de RSOD dataset zijn de volgende objecten getest: vliegtuigen, oliesilo's, speelplaats en oversteek (Tian, Z. et al., 2023). De resultaten voor deze dataset per objecttype zijn geïllustreerd in figuur 19.

Category	Faster-RCNN/%	YOLOv4/%	YOLOv5/%	RSyoloX[31]/%	FCOS [52]/%	DA ² FNet [50]/%	Ours/%
Aircraft	90.2	88.8	92.6	-	91.0	95.7	97.4
Oil tank	88.3	89.6	94.0	-	97.6	97.7	98.6
Playground	72.1	76.4	81.2	-	86.1	90.6	85.6
Overpass	80.9	91.5	94.9	-	99.7	95.1	99.5
Mean AP	82.9	86.6	90.7	93.1	93.6	94.8	95.3

Figuur 19, (a) Vergelijking mAP van objecten voor YOLOv3, YOLOv4 en YOLOv5 afkomstig uit RSOD (Tian, Z. et al., 2023).

Onderzoek **b** vergelijkt Fast-RCNN, YOLOv3, YOLOv5, YOLOv7 en net als in onderzoek **a**, is ook voor dit onderzoek een aangepaste YOLOv5 variant geïntroduceerd. In onderzoek **b** is een aangepast model voorgesteld dat specifiek is afgestemd voor *high-resolution* luchtfoto's (YOLO-HR). Dit onderzoek heeft gebruikgemaakt van de SIMD dataset. Deze dataset bevat kleine en middelgrote objecten. Verder heeft dit onderzoek gebruikgemaakt van de hardware en software configuraties afgebeeld in figuur 20 (Zhang, Y. et al., 2022). Deze resultaten zijn afgebeeld in figuur 21.

Onderdeel	Type	Geheugen
GPU	Nvidia geforce rtx 3090	32 gb
CPU	Intel(R) core(TM) i9-12900K	24 gb
Schijfgeheugen	SSD + HDD	500gb + 4tb
OS	Windows 10	-
Framework	Pytorch (v1.7)	-

Figuur 20, (b) (Zhang, Y. et al., 2022)

De SIMD dataset bevat in totaal 15 verschillende objecten, waaronder: vliegtuigen, lange voertuigen (aanhangwagens), bussen, auto's en boten (Zhang, Y. et al., 2022). De resultaten voor deze dataset zijn in figuur 21 voor zowel mAP, AP en Snelheid/ms weergegeven.

Name	AP50 (%)	mAP (%)	Params (M)	FLOPs (G)	Speed (ms)
YOLOv3-Tiny [32,34]	77.23	54.53	8.3	12.9	4.3
YOLOv5n [24]	79.56	60.69	1.7	4.2	4.8
YOLOv5s [24]	83.85	66.05	6.72	15.9	5.8
YOLO-HR-n	83.01	64.04	3.34	4.4	6.3
YOLO-HR-s	85.59	67.31	13.2	16.6	6.7
YOLOX-s [33]	76.63	56.83	8.94	26.79	5.7
YOLOv7s [25]	83.80	66.55	8.92	26.8	6.3
YOLOv7-tiny [25]	82.08	64.16	5.77	13.1	5.2
Faster RCNN [12]	77.74	-	41.19	198.47	26.3

Figuur 21, (b) (Zhang, Y. et al., 2022).

Onderzoek c vergelijkt YOLOv5, YOLOv6, YOLOv7 en YOLOvX (Zhang, Y. et al., 2022). YOLOvX wijkt af van de traditionele YOLO-varianten doordat deze geen anchors gebruikt om bounding boxen te definiëren (Yang, L. et al., 2022). Dit onderzoek heeft gebruikgemaakt van de "Oil Tank Detection" dataset. Dit is een open-source dataset beschikbaar gesteld op de datahub "Kaggle.com". In tegenstelling tot de datasets gebruikt in a en b is de dataset in onderzoek c slechts toegepast over één objecttype. Deze dataset bevat beelden afkomstig van de SPOT satelliet. Deze beelden bedragen een resolutie van ≈ 1.7 m. De dataset bevat oliesilo's in verschillende groottes (Zhang, Y. et al., 2022). In tegenstelling tot de andere twee onderzoeken heeft onderzoek c geen hardware configuraties vermeld. Om deze reden kunnen snelheden uit dit onderzoek niet meegenomen worden als bruikbare criteria. De mAP resultaten afgeleid uit onderzoek c zijn geïllustreerd in figuur 22.

Model	mAP@0.50	mAP@0.75	mAP
YOLOv5x	93.40%	79.43%	69.69%
YOLOX-x	85.60%	61.90%	55.60%
YOLOv6-L	90.20%	74.70%	65.60%
YOLOv7-X	94.45%	77.24%	68.11%

Figuur 22, (c) (Wan, D. et al., 2023)

Doordat tijdens dit onderzoek sprake was van oliesilo's met diverse formaten heeft onderzoeker T. Bakırman een relevante observatie gemaakt. Concluderend uit dit onderzoek blijken YOLOvX en YOLOv6 moeite te hebben met kleine objecten. Dit in tegentelling tot YOLOv5 en YOLOv7 die met hetzelfde scenario zonder problemen de oliesilo's wist te detecteren, zie ook figuur 23. Deze inzichten zijn zeer interessant voor verder onderzoek. Met name voor de toepassing op propaantanks, aangezien dit object ook relatief klein van formaat is.



Figuur 23, (Bakırman, T. 2023)

2.5 YOLOv5

YOLOv5 is een model dat gebruikmaakt van het PyTorch framework waardoor deze relatief eenvoudig in te zetten is. In het vorige onderdeel, paragraaf 2.4.6, zijn verschillende onderzoeken bekeken die object detectie toepasten over hoge resolutie luchtfoto's. Hieruit is gebleken dat YOLOv5 een hogere detectie nauwkeurigheid heeft vergeleken met voorgaande modellen. Alleen YOLOv7 presteert op dit gebied. Daarnaast is YOLOv5 ten opzichte van YOLOv7 minder intensief in geheugengebruik (lichtgewicht). Tot slot ondersteunt YOLOv5 integratie naar andere platformen zoals Android of IOS. De bovengenoemde aspecten maakt YOLOv5 zeer flexibel in gebruik en zeer geschikt voor verdere opschalingsmogelijkheden.

Dit hoofdstuk zal in meer detail ingaan op de architectuur van dit model. Daarnaast worden de specifieke augmentatie-technieken en de belangrijke hyperparameters met betrekking tot YOLOv5 toegelicht.

2.5.1 Architectuur

Over het algemeen houden alle YOLO-varianten in grote lijnen dezelfde architectuur aan. Beginnende bij de *Backbone* gevolgd door een *Neck* en een *Head*. Deze elementen bevatten intern per iteratie een andere samenstelling van lagen en netwerken.

De backbone

De backbone van YOLOv5 bestaat uit een aangepaste variant van de Darknet53 backbone genaamd CSPDarknet. Zoals de naam suggereert is deze backbone een combinatie van een cross-stage partial network (CSPNet) en Darknet. De toepassing van CSPNet verhoogt de efficiëntie en maakt het model lichtgewicht (Palla, T. S. et., 2023).

De neck

De belangrijkste aanpassingen die gemaakt zijn in de neck betreft de integratie van een snellere variant van spatial pyramid pooling (SPP), namelijk spatial pyramid pooling fast (SPPF) en een gemodificeerd path aggregation network (PANet). De SPP functie wordt gebruikt in YOLOv3 en YOLOv4. SPPF verdubbelt de snelheid waarmee de datastroom gevoed kan worden aan het netwerk PANET. YOLOv5 heeft een aantal modificaties gemaakt over dit netwerk waardoor meer informatie kan worden verzameld uit de dataset. Dit resulteert in een hogere accuraatheid en precisie met betrekking tot object lokalisatie (Palla, T. S. et., 2023).

De head

De head bestaat uit drie convolutionele lagen, zie de laatste fase in figuur 24. In deze lagen worden de anchor boxes uit het trainingsproces gebruikt om de bounding boxes te voorspellen en de finaliserende berekeningen te maken. De head van YOLOv5 bevat dezelfde samenstelling als YOLOv3 en YOLOv4. Hier zijn geen verbeteringen gemaakt (Palla, T. S. et., 2023).

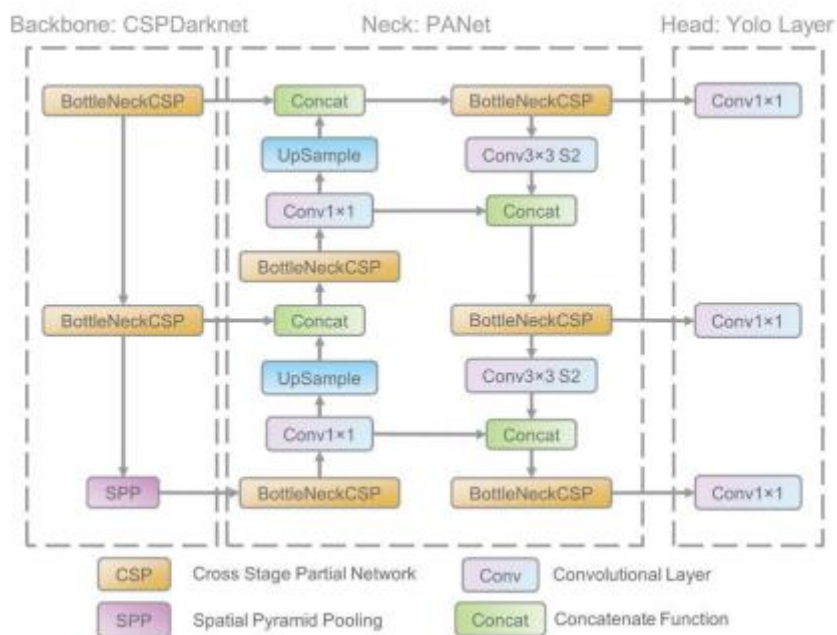


Fig. 24 (Palla, T. S. et., 2023)

2.5.2 YOLOv5 voorgetrainde modellen

Wanneer een YOLO-model getraind wordt, is het gebruikelijk om als uitgangspositie gebruik te maken van een pre-trained model. YOLOv5 bevat verschillende voorgetrainde modellen waar gebruik van gemaakt kan worden. YOLOv5n (nano), YOLOv5s (small), YOLOv5m (medium), YOLOv5l (large), YOLOv5x (extra large), weergegeven in figuur 25.

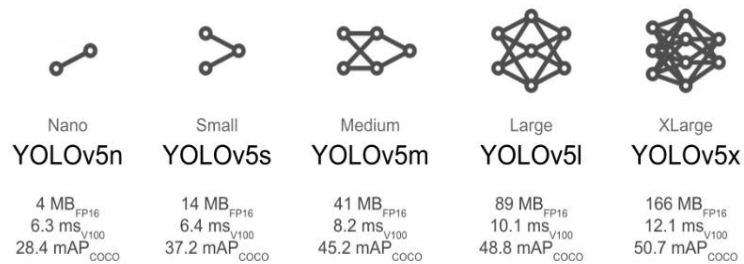


Fig. 25 voorgetrainde modellen (Ultralytics. 2023)

De verschillen tussen deze modellen zijn vooral aanwezig op het gebied van prestaties en vereiste rekenkracht, zie figuur 26. Hoewel een nano-model minder geheugen vereist en sneller is, leveren andere modellen betere accuraatheid. Het is daarom belangrijk dat het juiste model in productie wordt toegepast. Een nano-model is zeer geschikt voor testdoeleinden, echter wanneer de werkwijze daadwerkelijk in productie wordt toegepast zijn de laatste drie modellen beter geschikt (medium, large of extra large).

Model	size (pixels)	mAP ^{val} 50-95	mAP ^{val} 50	Speed CPU b1 (ms)	Speed V100 b1 (ms)	Speed V100 b32 (ms)	params (M)	FLOPs @640 (B)
YOLOv5n	640	28.0	45.7	45	6.3	0.6	1.9	4.5
YOLOv5s	640	37.4	56.8	98	6.4	0.9	7.2	16.5
YOLOv5m	640	45.4	64.1	224	8.2	1.7	21.2	49.0
YOLOv5l	640	49.0	67.3	430	10.1	2.7	46.5	109.1
YOLOv5x	640	50.7	68.9	766	12.1	4.8	86.7	205.7

Figuur 26, verschillende gemeten prestaties (Ultralytics. 2023)

2.5.3 YOLOv5 Augmentatie

In de basis bevat YOLOv5 een aantal data-augmentatie-technieken. Deze augmentatie technieken zijn als volgt gedefinieerd in de source-code: gamma, hasv-Hue, HSV-saturatie, HSV-value, rotation, fractional translation, scale, shear, flip up-down, flip left-right, mosaic, mixup en copy-paste segments. Deze technieken zijn gedefinieerd in de zogenoemde hyperparameters. De augmentatie-onderdelen bevatten waardes van 0 tot 1, deze geven de gradatie van intensiteit aan. Niet alle technieken worden meegenomen in de standaard instelling, de volgende augmentatie-technieken staan namelijk ingesteld op 0.00 in de hyperparameters: copy-paste segments, flip up-down, shear, rotation en gamma.

2.5.4 Configuratie training-proces

De YOLOv5 GitHub repository beheert door Ultralytics is een belangrijk onderdeel van het YOLOv5 framework. Deze repository bevat een aantal Python scripts die geactiveerd moeten worden om bepaalde taken uit te voeren. Voor het initiëren van het trainingsproces wordt het `train.py` script aangeroepen. Voordat dit `train.py` script kan worden geïnitieerd dienen een aantal parameters geconfigureerd te worden. De terminologie achter deze parameters zal in deze alinea worden toegelicht. Deze termen komen terug in methoden (Ultralytics, z.d.)

Img

De `img` parameter definieert het formaat van de afbeeldingen uit de trainingsdataset. YOLOv5 gebruikt als standaardwaarde 1024 x 1024 pixels. Hoewel vaak kleinere formaten worden gebruikt het proces te versnellen. Daarbij kan het knippen/croppen van deze afbeelding (e.g. in delen van 416 x 416) object detectie verbeteren doordat de objecten groter zijn afgebeeld (Thuan, 2021)

Epoch

In de praktijk wordt de gehele dataset meerdere malen door het neurale netwerk gezonden om beste resultaten te behalen. Een enkele iteratie van een volledige dataset wordt een epoch genoemd. Een epoch van bijvoorbeeld 300 waar gebruik wordt gemaakt van een dataset van 1700 afbeeldingen zorgt ervoor dat alle 1700 afbeeldingen 300 maal getraind worden door het neurale netwerk. Over het algemeen geldt de regel "des te hoger het aantal training iteraties, des te beter de nauwkeurigheid". Echter heeft ieder model een training limitatie, afhankelijk van de kwaliteit van de dataset. Wanneer dit punt bereikt is zal het mAP geen verbeteringen meer ondergaan (Thuan, 2021).

Batch

Batch representeert het aantal afbeeldingen welke als groep door het neurale netwerk worden gezonden. Een epoch dat 1700 afbeeldingen direct door het neurale netwerk stuurt is zeer geheugen-intensief. Om dit proces lichter te maken is een epoch verdeeld in kleinere "batches". De batch waarde is de waarde waardoor de dataset door gedeeld wordt. Een dataset van 1700 afbeeldingen met een epoch van 300 en een batch van 32 resulteert in $1700 \div 32 = 53,12$ batches van 32 afbeeldingen welke 300 keer aan het netwerk gevoed worden (Thuan, 2021).

Data

Deze parameter verwacht een .YAML bestand waarin gerefereerd wordt naar de locaties van alle benodigde data. Dit betreft de training-dataset, de validatie-dataset en optioneel de test-dataset. Ook het aantal klassen (`nc`) dat opgeslagen dient te worden en het labelen van deze objecten zijn gedefinieerd in dit .YAML bestand (Thuan, 2021).

Voorbeeld van .YAML bestand:

```
path: C:/datasets/turbi
train: C:/datasets/turbi/train/images
val: C:/datasets/turbi/valid/images
nc: 1
names: windturbine
```

Cfg

cfg of voluit *configuration* representeert het configuratiebestand van het model. Hierin is de architectuur van het model (backbone, neck en de head) gedefinieerd (Ultralytics, 2023).

Weights

In de weights parameter wordt bepaald welk voorgetrainde model wordt gebruikt als uitgangspunt. De keuzes hierin zijn eerder benoemd in 2.5.2 (Ultralytics, 2023).

Name

Bepaald waar de nieuw getrainde weights worden opgeslagen. Deze resulterende weging kan daarna op ieder moment gebruikt worden om object detectie toe te passen (Thuan, 2021).

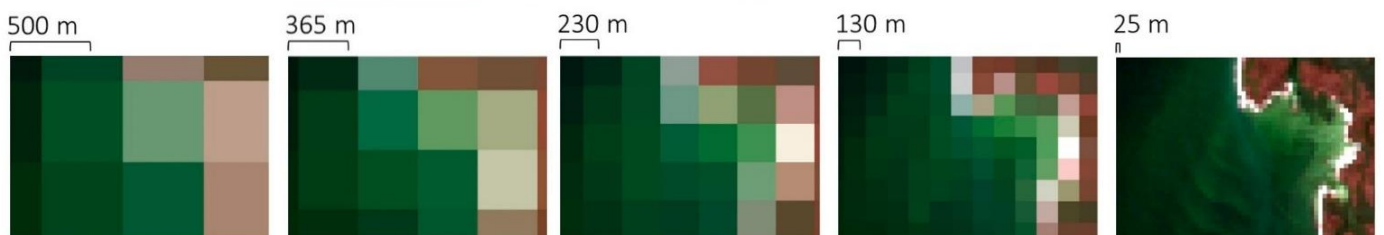
Hyps

Tot slot kan verwezen worden naar een bestand waarin de hyperparameters (hyps) in opgeslagen zijn. Wanneer deze parameter niet gebruikt wordt zal de standaard hyps gebruikt worden. (Ultralytics, z.d.-b). Hyperparameters zijn eerder toegelicht in 2.4.4.

2.6. Data-ethiek & privacy

2.6.1 Ontwikkelingen in geo-technologie

Het genereren van geospatiale gegevens is door voortuitgang in sensor- en satelliettechnologie (hoge optische resolutie & temporele resolutie) toegankelijker geworden wat heeft geleid tot een hogere dekkingsgraad en detailniveau (Bertino et al., 2008). Een concreet voorbeeld hiervan is de bevordering van de gemiddelde spatiale resolutie (pixelgrootte) die, in minder dan vijftig jaar, van 500 tot 25 meter is gegaan. De hoeveelheid ruimtelijke gegevens in een pixel van een satellietbeeld is daarmee significant gestegen wat heeft bijgedragen aan verbeterde classificatie en detectiemogelijkheden. Deze ontwikkeling is duidelijk terug te zien in figuur 27 (Coffer, 2020). Waar satellietbeelden in het begin voornamelijk werden gebruikt voor militaire doeleinden en de monitoring van het weer is deze technologie daardoor steeds relevanter geworden voor andere kritische toepassingen. Dit betreft bijvoorbeeld het monitoren van het klimaat, stedelijke gebieden, natuurrampen, ruimtelijke ontwikkelingen, et cetera. (Bertino et al., 2008; Coffer, 2020). Een onderzoek van Yeh et al. (2020) toont ook aan dat satellietbeelden voor duurzaamheidsontwikkelingen worden toegepast in combinatie met Artificial Intelligence (AI) en Machine Learning (ML) om de bodemkwaliteit te bevorderen in Afrika. Dit voorbeeld toont een grotere ontwikkeling binnen de geo-technologie en dat is de toepassing van innovatieve analysemethodieken. Vooruitgang in AI en ML hebben ertoe geleid dat complexe ruimtelijke problematiek geanalyseerd kan worden op basis van hoge resolutie satellietbeelden (Hagendorff, 2019). De vooruitgang in optische- en temporele resolutie is daarmee gerelateerd aan de voortgang in beeldverwerkingsanalyses. Deze zijn nu mogelijk door de complexe informatiestromen op geautomatiseerde wijze te structureren. Ondanks de mogelijkheden van deze hoge resolutie satellietbeelden zijn er ook kritieken, voornamelijk op het gebied van privacy. De stijging in ruimtelijke gegevens betekent namelijk ook een stijging in gegevens die gerelateerd kunnen worden aan een individu.



Figuur 27, ontwikkelingen in de resolutie van satellietbeelden (Coffer, 2020)

2.6.2 Implicaties hoge resolutie *remote sensing* beelden

Veel aspecten binnen (geo)data en AI zijn gerelateerd aan privacy en data-ethiek. Om dit onderdeel zo veel mogelijk te concretiseren zal uitsluitend worden ingegaan op ethiek gespecificeerd op hoge resolutie *remote sensing* beelden. Om te beginnen is het van belang het begrip 'privacy' te concretiseren. Privacy kan worden gedefinieerd op verschillende manieren, Hagendorff (2019) beschrijft het als de controle of regulatie die een individu heeft over het delen van persoonlijke overtuigingen, gedragingen, activiteiten, et cetera. Het beschermen van privacy zijn daarom gericht op de toegankelijkheid van de 'buitenwereld' op de persoonlijke informatie van een individu, zoals persoonlijke gegevens, woning, naam, afbeeldingen, et cetera (Safdar et al., 2020). Theoretisch gezien is dit niet in lijn met het principe van satellietbeelden, die vrijwel een constant beeld bieden van alles wat op het aardvlak plaatsvindt. Dit gebeurt tevens op een steeds gedetailleerder niveau, bijvoorbeeld in het geval van de DigitalGlobe's WorldView3 satelliet die een resolutie van 31 centimeter per pixel behaalt. Ondanks de hoeveelheid ruimtelijke gegevens die hieruit voortkomen is het onvoldoende om onderscheidende kenmerken van specifieke individuen te extraheren. De privacy van een persoon komt, zelfs met de huidige staat van satelliettechnologie, niet in gevaar. Ondanks dat in het eerste opzicht vrijwel geen directe karakteristieken van een individu te detecteren is zijn, is dit in theorie wel mogelijk. Technisch gezien kunnen algemene kenmerken zoals lichaamstype en kleding in combinatie met locatie worden ingewonnen door middel van Ultra Hoge Resolutie satellietbeelden. Dit is overigens vooral in de theorie mogelijk door randzaken en mogelijke gevolgen, zoals: beschikbaarheid van detectiemogelijkheden, vooruitstrevende technologie, voldoende budgettering en tijd (Santos, 2019).

Op basis van hoge resolutie satellietbeelden is het dus niet direct mogelijk personen te identificeren en daarmee privacy te verstoren. Santos (2019) beschrijft het als volgt: 'The combination of publicly available data pools with high resolution image data, coupled with the integration and analysis capabilities of modern Geographic Information Systems (GIS) disclosing geographic keys such as longitude and latitude, can result in a technological invasion of personal privacy'. Hiermee wordt bedoeld dat het mogelijk is om de globale kenmerken van een persoon te koppelen aan locaties of afbeeldingen die in combinatie met een analysemodel kan resulteren tot privacygevoelige informatie (Santos, 2019). Dit proces is arbeidsintensief en gericht op het analyseren van één individu voor specifieke doeleinden. Ondanks dit risico is de toepassing van hoge resolutie luchtfoto's van groot belang bij het behalen van bijvoorbeeld duurzaamheidsdoelen (Bertino et al., 2008). Het is daarom van belang om de balans te behouden tussen de positieve effecten en negatieve implicaties van hoge resolutie luchtfoto's. Het is niet zozeer de aanwezigheid van de beelden zelf, maar voornamelijk de acties die tot problematiek leiden. Daarom is het van belang verantwoordelijk om te gaan met ieder type data die vervormd kan worden tot privacygevoelige informatie. Een balans tussen analytische inzichten, ethiek, privacy en algemene dataprotectie kan bijdragen aan inzichten in een steeds veranderende wereld (Santos, 2019).

Probleemanalyse en strategie:

Een concreet overzicht

03

Dit project omvat verschillende technische elementen. Om te voorkomen dat belangrijke onderdelen onduidelijk zijn worden in dit hoofdstuk specifieke onderdelen expliciet toegelicht.

3.1 Probleemstelling

De concrete probleemstelling is dat het REV gecontroleerd moet worden om de betrouwbaarheid ervan te waarborgen. Het is complex om dit op geautomatiseerde wijze te doen omdat de elementen in het register fysiek uiteenlopende eigenschappen bedragen. Ondanks dit is het van belang dat deze nauwkeurig en accuraat zijn geregistreerd (Dorlandt, 2022).

3.2 Doelstelling

In het kort is de doelstelling om een reproduceerbare methodologie te ontwikkelen voor de controle of risicovolle objecten en activiteiten in de openbare ruimte correct in het REV staan. De methodologie zal worden toegepast op propaantanks en windturbines in geselecteerde testgebieden. Bovendien worden adviezen gegeven om de methodologie op te schalen en toe te passen op andere objecttypen. Deze automatisatie draagt daarmee bij aan de bevordering van transparantie en de kwaliteit van het register. Dit project heeft daarmee de ambitie om bij te dragen aan een inzichtelijke en veilige leefomgeving.

3.3 Globale strategische keuzes en werkwijze

Het REV bevat in totaal 47 elementen met ieder afwijkende fysieke eigenschappen. Er zijn strategische keuzes gemaakt om het gewenste resultaat te bereiken. (1) Om te beginnen is er een keuze gemaakt om een object detectiemodel op basis van luchtfoto's op te stellen. Deze technologie is effectief omdat het op efficiënte wijze objecten uit de openbare ruimte kan analyseren op basis van het YOLOv5 model. Er is specifiek gekozen voor dit model door de toegankelijke infrastructuur, zie 2.4.5 en relatief hoge nauwkeurigheid, zie 2.4.6. Een toegankelijke infrastructuur is van belang omdat dit het opschalingsproces kan vereenvoudigen. (2) Verder is de keuze gemaakt om het model te ontwikkelen voor twee objecttypen uit het REV, namelijk: windturbines en propaantanks. (3) Op basis hiervan is een methodologie ontwikkeld geoptimaliseerd voor deze twee objecttypen. (4) De methodologie vormt een fundament die op basis van concrete adviezen en aanbevelingen reproduceerbaar is voor andere objecten uit het REV.

Methodologie & materiaal:

Onderzoekopzet, dataverzameling en informatieverwerking

04

Dit hoofdstuk heeft als doelstelling om inzichten te bieden in het ontwikkelingsproces, maar ook om te dienen als handleiding bij de reproductie ervan. Onderstaande opsomming toont een overzicht van de behandelde onderdelen in dit hoofdstuk. Hierbij is een verdeling gemaakt tussen twee elementen, namelijk: globaal en object-gespecificeerd. Met globaal wordt hier bedoeld op de methodologie die ongeacht het objecttype toegepast kan worden. Het onderdeel object-gespecificeerd bevat verdiepende stappen gericht op de objecttypen: propaantanks en windturbines.

Kwalitatief onderzoek

Korte introductie van de opzet van het kwalitatieve onderzoek dat is toegepast voor dit project beschreven in het theoretisch kader.

Randvoorwaarden en ontsluiting

Beschrijving van de technische randvoorwaarden van software en hardware en selectie van objecttypen.

Data-acquisitie (globaal)

Dit onderdeel is gericht het inwinnen van relevante data en de voorwaarden die hier toe behoren.

Dataverwerking (globaal)

Bij dataverwerking staat de wijze waarop de ingewonnen data is getransformeerd en verwerkt voor het model.

Dataverwerking (object-gespecificeerd)

In dit onderdeel zijn de stappen gespecificeerd op windturbines en propaantanks uitgebreid beschreven.

SMART inwinnen van data in de openbare ruimte

4.1 Kwalitatief onderzoek

Het eerste onderdeel dat is uitgevoerd is het kwalitatieve onderzoek gericht op wetenschappelijke publicaties. De onderzoeksopzet voor het literatuuronderzoek heeft zich gericht op verschillende onderdelen binnen object detectie. De bronnen die zijn geraadpleegd zijn gevonden door middel van Google Scholar, deze zijn toegelicht in onderstaande opsomming:

1. Opdoen van fundamentele kennis over object detectiemodellen met betrekking tot AI, ML, DL, CV
2. Een vergelijkingsstudie over typen object detectiemodellen zoals; RCNN, Fast RCNN, Faster RCNN, SSD, YOLO (one-stage & two-stage detection)
3. Een verdiepend onderzoek specifiek gericht op de verschillende YOLO versies voornamelijk gericht op YOLOv4, YOLOv5, YOLOv7
4. Onderzoek naar publicaties gericht op privacy & data-ethiek gespecificeerd op hoge resolutie remote sensing beelden en de toepassing van AI

Op basis van het eerste onderdeel (1) is het fundament voor het onderzoek opgesteld. Het opstellen van de vergelijkingsstudie binnen object detectiemodellen (2) heeft de basis gevormd voor het verdiepende onderzoek van de YOLO versies (3). Tot slot is het onderzoek naar privacy & ethiek uitgevoerd om te voorkomen dat ethische grenzen worden overschreden (4).

4.2 Randvoorwaarden en ontsluiting

In dit onderdeel wordt ingegaan op het kwantitatieve onderzoek van het project. Dit bedraagt zowel het opstellen en optimaliseren van het model als het analyseren van de mogelijkheden.

4.2.1 Object selectie

Dit onderzoek is gericht op het detecteren van twee verschillende objecttypen voortkomend uit het REV. Deze objecttypen zijn geselecteerd op basis van fysieke kenmerkbaarheid en urgentie. De fysieke eigenschappen van de objecten zijn verschillend en bedragen daarmee een andere mate van complexiteit. Windturbines is een concreet detecteerbaar object door grote ruimtelijke dimensies en unieke eigenschappen. Propaantanks zijn beduidend uitdagender te detecteren door de relatief kleine ruimtelijke dimensies en fysieke belemmeringen (bladerdaken, gebouwen, et cetera.). De complexiteit van propaantanks zijn van belang om de mogelijkheden en limitaties van het model aan te tonen.

4.2.2 Materialen

Dit onderzoeksrapport maakt gebruik van een aantal technische middelen om resultaten te behalen. Deze middelen bestaan uit programma's, software, hardware en overige extensies die geïnstalleerd dienen te worden. De toepassing van de voorgestelde technologieën dienen als minimumvereiste bij de reproductie van de methodiek op andere objecten.

4.2.2.1 Hardware

Het uitvoeren van een trainingsproces voor een object detectie systeem benodigd veel rekenkracht. Ondanks de relatief eenvoudige infrastructuur is dit ook het geval voor YOLOv5.

Hoewel het mogelijk is om deze rekenkracht te verdelen over de processor (CPU) van een computer is dit niet efficiënt. Daarvoor is het beschikbaar stellen van een krachtige GPU een vereiste. Voor dit onderzoek is de hardware configuratie gebruikt getoond in figuur 28.

Onderdeel	Type	Geheugen
GPU	Nvidia GeForce RTX 1070 Ti	8 gb
CPU	Intel(R) core(TM) i5-12900K	16 gb
Schijfgeheugen	SSD + HDD	4tb + 500gb
OS	Windows 10	-
Framework	Pytorch (v1.12)	-

Figuur 28, overzicht van de hardware configuratie

De hardware beïnvloedt de snelheid waarmee een model kan worden uitgevoerd. Met hoogwaardige hardware kan de snelheid van het trainingsproces daarom worden geoptimaliseerd. Een belangrijk onderdeel binnen de snelheid van het trainingsproces is de batch-size parameter. Deze parameter staat voor de hoeveelheid iteraties die het model uitvoert op een gedeelte van de dataset voordat deze verder wordt bijgewerkt. De hardware configuratie dat in dit onderzoek is gebruikt had het vermogen om een maximaal batch-size van vier te gebruiken, wat relatief laag is. De gebruikelijke waarde voor deze parameter is acht of zestien. Om dit mogelijk te maken is het van belang een GPU met minimaal 8 GB geheugen te implementeren. De bovenstaande configuratie is daarmee het minimum dat aanwezig moet zijn bij de reproductie van de methodologie.

4.2.2.2 Software

Naast hardware componenten is de toepassing van de voorgestelde software een vereiste voor de pre-processing, processing en post-processing onderdelen. Voor het automatisatieproces in dit onderzoek is voornamelijk gebruik gemaakt van de programmeertaal Python. Er zijn voor zowel pre-processing als post-processing overigens ook afwijkende software componenten toegepast, zie onderstaande opsomming:

- Pre-processing: Voor het annoteren van data is de open source software *Labelimg* toegepast.
- Post-processing: Voor het uitsluiten van foutgevoelige gebieden is *GDAL* geïmplementeerd. Dit is een bibliotheek voor het converteren van raster en vector formaten. Wanneer GDAL als module wordt geïnstalleerd zijn niet alle functionaliteiten beschikbaar die wel in de externe GDAL shell aanwezig zijn. Om de toegang tot deze shell te verkrijgen is een extra installatie nodig. Dit kan op een van de twee volgende methoden;
 - De GDAL shell is als alleenstaand programma te downloaden via de officiële GDAL website
 - Als alternatief kan de GDAL shell ook geïnstalleerd worden tijdens de installatie van de GIS software QGIS.
- Post-Processing: ArcGIS Pro is toegepast om de dichtstbijzijnde punten te berekenen ten opzichte van de punten in het REV.

4.2.2.3 Overige extensies

Python heeft een uitgebreide distributie van extensies en modules die relevant zijn bij dit onderzoek. In de bijlage van ieder script en in het tekstbestand dependencies.txt zijn alle modules genoteerd die nodig zijn om de Python scripts uit te kunnen voeren.

4.3 Data-acquisitie (globaal)

Dit onderdeel benadrukt de vereiste data voor het gebruik van deze werkwijze. Ook wordt globaal omschreven welke databronnen geraadpleegd kunnen worden voor de acquisitie hiervan. De fundamentele data-onderdelen zijn te verdelen in drie subgroepen:

- (4.3.1) Data gebruikt om objecten effectief te voorspellen
- (4.3.2) Data ter ondersteuning van het objectdetectiesysteem (ODS)
- (4.3.3) Data ter ondersteuning van optimaliserende activiteiten

4.3.1 Testdata

Na het trainen van een ODS kan deze gebruikt worden om onbekende scenario's te voorspellen. Hiervoor is data nodig. Deze data neemt de vorm aan van afbeeldingen, in dit geval luchtfoto's. De objecten die voorspeld moeten worden dienen visueel duidelijk herkenbaar te zijn in deze scenario's. Aangezien dit onderzoek objecten verwerkt van verschillende formaten is het essentieel dat deze data kwalitatief hoogwaardig is. Ook moeten de voorspelde resultaten geplot kunnen worden in GIS software om verder analyses over uit te kunnen voeren. Daarom is gekozen om voor de datagroep testdata gebruik te maken van hoge resolutie GeoTIFF bestanden. Deze bestanden behouden een resolutie van 8 cm x 8 cm waardoor zelfs de kleinere objecten zoals propaantanks zichtbaar zijn.

Deze GeoTIFF bestanden worden gedownload over een vooraf geselecteerd testgebied. Hierbij is het belangrijk dat gebieden die gebruikt zijn tijdens het trainingsproces niet worden gebruikt als testdata om bias uitkomsten te voorkomen. In dit onderzoek zijn deze testgebieden zorgvuldig geselecteerd op gebieden zonder bias, maar met de aanwezigheid van potentieel foutgevoelige objecten. Op deze manier kan het model getest worden op mogelijke limitaties.

Het downloaden van de afbeeldingen aan de hand van het testgebied resulteert in meerdere GeoTIFF bestanden van 12500 x 12500 pixels. Dit is te groot voor YOLOv5 om te kunnen verwerken daarom worden deze afbeeldingen in kleinere delen van 1280 x 1280 pixels geknipt. Dit is gedaan door op een iteratieve wijze een GeoTIFF bestand op te delen en op te slaan in kleinere TIF bestanden middels het Python script in bijlage **A.I**. Voordat deze afbeeldingen gebruikt kunnen worden voor detectie vinden een aantal preventieve optimalisaties plaats, hier wordt in meer detail ingegaan in paragraaf 4.4.5.

4.3.2 Dataset ODS

Dit onderdeel benadrukt welke data voor het YOLOv5 objectdetectiesysteem aanwezig moet zijn om uitgevoerd te kunnen worden. Een zelflerend objectdetectiesysteem zoals YOLOv5 kan voorspellende resultaten leveren gebaseerd op feitelijke data. Deze feitelijke *input* data neemt de vorm aan van voorbeelden van het object dat gedetecteerd dient te worden. De objecten binnen deze afbeelding dienen aangegeven te worden door middel van zogenoemde annotaties. Dit zijn notaties die aangeven waar op de afbeelding het object zich bevindt. Een hoog aantal voorbeeld afbeeldingen en de mate van variërende scenario's waarin het object zich bevindt verhogen het detectiebereik van het model. Het is daarom van belang dat veel en gevarieerde data wordt verzameld ter voorbereiding van de initiatie van het YOLOv5 trainingsproces. Voor dit onderzoek zijn daarom hoge resolutie luchtfoto's van de objecten vereist.

Afhankelijk van het type algoritme van het ODS dient deze data in verschillende formaten worden opgeslagen. Een trainingsdataset van YOLOv4 verwacht bijvoorbeeld dat afbeeldingen en annotaties in dezelfde folder opgeslagen zijn. YOLOv5 daarentegen neemt aan dat de annotatiedata en afbeeldingen aangeleverd worden in aparte folders. Ook de vorm van annoteren verschilt per ODS. De vier dominerende annotatie formaten zijn pascal_voc, albumentations, coco en YOLO. Het ODS dat gebruikt wordt in dit onderzoek neemt aan dat de annotaties in het YOLO formaat zijn opgeslagen. Is dit niet het geval dienen deze geconverteerd te worden.

4.3.3 Data acquisitie ODS

4.3.3.1 Openbaar beschikbare datasets

Een trainingsdataset zoals beschreven in 4.3.2 is voor bepaalde gevallen als openbare dataset online beschikbaar. Veelbelovende open databronnen zoals Roboflow, Kaggle en GitHub hebben veel ODS datasets openbaar vrijgegeven. Wanneer gebruik wordt gemaakt van een soortgelijke dataset moet extra aandacht besteed worden aan het controleren van de formatering van de annotaties en de kwaliteit en context van de afbeeldingen. Idealiter is de context van een ODS dataset vergelijkbaar met de testdata zodat het ODS geen onverwachte (ofwel niet aangeleerde) situaties zal herkennen. Wanneer gebruik wordt gemaakt van openbare datasets is dit vaak niet het geval. Dit kan leiden in een reductie van de detectiecapaciteit over de geselecteerde testdataset.

Deze manier van data acquisitie is toegepast voor de eerste uitvoering van het YOLOv5 model met betrekking tot het object "windturbines". Voor deze uitvoering zijn de volgende openbare datasets gebruikt;

- Wind Turbine Detection **(1)** geraadpleegd van Kaggle.com
- Wind-Turbine-Detector **(2)** geraadpleegd van GitHub.com

Deze datasets bevatten luchtfoto's met variërende resolutie tussen 0.5 en 1.3 cm en bevatten samen in totaal 1700 beelden waar windturbines op weergegeven zijn. De annotaties van dataset **1** staan in het gewenste formaat conform de YOLO standaarden, dit is niet het geval voor dataset **2**. Voordat deze dataset gebruikt kan worden met YOLOv5 is het noodzakelijk dat bepaalde pre-processen uitgevoerd worden. In hoofdstuk 4.4.3 Pre-processing wordt verder ingegaan op dit proces.

Een andere methode om een trainingsdataset te verzamelen is door handmatig een dataset samen te stellen. Deze vorm van acquisitie is arbeidsintensief en tijd consumerend. Maar wanneer correct uitgevoerd levert dit proces een dataset op waar formatering vooraf bepaald kan worden en kwaliteit en context gewaarborgd kunnen worden.

4.3.3.2 Op maat gemaakte datasets

Het samenstellen van een trainingsdataset vergt een aantal dataverzameling en dataverwerking stappen. Ten eerste zal een dataset vergaard moeten worden van verschillende afbeeldingen dat het te detecteren object weergeeft. Voor de tweede uitvoering van het YOLOv5 model gericht op windturbines en propaantanks zijn om deze reden afbeeldingen gedownload afkomstig van Beeldmaterialen.nl. Dit zijn afbeeldingen met een resolutie van 8 cm x 8 cm. Via deze bron kunnen tegels van 1 km x 1km als individuele bestanden worden gedownload. Het proces op onderstaande wijze uitgevoerd:

1. Om tegels met gevarieerde situaties te verkrijgen zijn gebieden geselecteerd verspreid over heel Nederland.
2. Deze tegels zijn vervolgens samengevoegd tot één enkelvoudig VRT-bestand zie hiervoor bijlage **B.I**.
3. Daarna zijn de bestaande punten uit de REV gebruikt als referentiepunten om de locaties van specifieke objecten uit de gebieden van het VRT-bestand te extraheren.
 - a. Dit is gedaan door een vierhoekig gebied van 1280 x 1280 pixels rondom het REV punt uit het VRT-bestand te knippen. Voor de exacte methode zie bijlage **B.II**.

Het is hierbij noodzakelijk om de exacte locatie van het betreffende object(en) op alle 1280 x 1280 afbeelding te annoteren. Omdat er sprake is van een groot aantal afbeeldingen is dit een tijdrovend proces. Om dit proces efficiënt te laten verlopen bestaan verschillende freeware programma's. Voor het annotatieproces in dit onderzoek is gebruik gemaakt van Labelimg. De installatie en het gebruik hiervan is terug te lezen in bijlage **B.III**.

4.3.3.3 Limitaties en uitdagingen

Het uitsnijden van de ingewonnen luchtfoto's, toegelicht in 4.3.1, kan als gevolg hebben dat objecten op de scheidingslijnen worden gesplitst. Dit is met name het geval bij objecten met grotere fysieke dimensies. Ook is het mogelijk dat het splitsen van een object wordt veroorzaakt doordat een object simpelweg groter is dan 102,4 m (pixels (1280) * resolutie (8 cm)). Een andere verklaring van objectsplitsing kan de camerahoek zijn die afwijkt van het gewenste top down perspectief.

Om objectsplitsing te voorkomen kan het formaat van de rasteruitsneden vergroot worden. In plaats van 1280 x 1280 pixels is het bijvoorbeeld mogelijk om uitsneden van 2580 x 2580 pixels te maken. Dit zal resulteren in een lagere kans dat objectsplitsing voor zal komen tijdens de constructie van trainingsdata.

Een ODS trainen met hogere resolutie trainingsdata vereist een GPU met meer dan 8 GB VRAM. Wanneer deze vereiste hardware niet aanwezig is kan downsampling worden toegepast. Bij deze methode is het mogelijk de resolutie van uitsneden terug schalen naar bijvoorbeeld 1280 x 1280. Dit gaat wel ten koste van het detailniveau. Hierbij is het wel van belang resolutie van de trainingsdata moet gelijkwaardig zijn aan het geselecteerde testgebied om hoogwaardige resultaten te behalen. Zoals in paragraaf 3.2.1 is toegelicht worden deze afbeeldingen op een iteratieve wijze geknipt.

Objectsplitsing is in dit proces ongeacht het formaat van de uitsnede een risico. Er zijn twee manieren om hier mee om te gaan.

1. Bij het formaat van trainingsdata kan worden gekozen voor grotere uitsneden. Op deze manier bevat deze data vrijwel geen gesplitste objecten. In dit geval is het model niet/onvoldoende getraind op gesplitste objecttypen en kan het deze minder goed detecteren. Het risico in dit geval is dat wanneer gesplitste objecttypen alsnog voorkomen deze hoogstwaarschijnlijk niet correct worden gedetecteerd.
2. De andere optie bestaat uit het toepassen van een formaat waarbij objectsplitsing veel voorkomt. Wanneer deze data wordt getraind in het model kan het model gesplitste objecttypen mogelijk correct detecteren. Het model leert hierbij namelijk niet alleen het totale object te herkennen, maar ook losse componenten van het objecttype. Voor dit onderzoek is deze methode gebruikt.

4.3.4 Data acquisitie optimalisatie taken

Het laatste onderdeel binnen data acquisitie is gericht op de toepassing van externe databronnen voor optimalisatie. De taken tot deze activiteit behoren worden verder toegelicht bij paragraaf 4.4.5

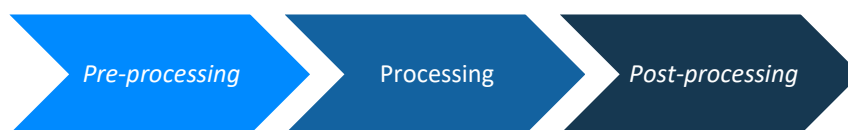
Post-processing. De data die benodigd is voor deze stappen komt voort uit de BGT (Basisregistratie Grootschalige Topografie). De BGT bevat door de overheid beheerde data afkomstig uit de openbare ruimte. Hieronder vallen ruimtelijke gegevens zoals autowegen, fietspaden, kunstwerken en overige openbare faciliteiten.

Deze data is op te vragen via de officiële website van PDOK. Aangezien de testgebieden (zie 4.5.1.1 en 4.5.2.1) voor dit onderzoek relatief kleinschalig zijn, is data extraheren door middel van het PDOK voldoende. De data dat is verkregen middels de handmatige download en omvat data uit de datagroep “onbegroeidterreindeel” en “wegdeel”. Uit deze datagroepen zijn verder selecties gefilterd, deze zijn verder beschreven bij paragraaf 4.5.1.3 en 4.5.2.2.

4.4 Dataverwerking (globaal)

4.4.1 Indeling dataverwerking

Het hoofdstuk data-acquisitie is ingegaan op een aantal globale onderdelen van de inwinning van data. In dit onderdeel zal ingegaan worden op de dataverwerking hiervan. Om de reproduceerbaarheid van dit onderzoek te bevorderen is er een dataverwerkingsproces opgesteld bestaande uit drie onderdelen, zie figuur 29.



Figuur 29, dataverwerkingsproces

Pre-processing is hierbij gericht op alle vormen van inwinning- en manipulatie van data die plaats heeft gevonden voorafgaand aan de analyse. In het geval van het ontwikkelde objectdetectiemodel was dit gericht op de ETL (Extract, Transform, Load) van trainingsdata en de toepassing van diverse augmentatie methodieken. Het processing onderdeel is gericht op het trainen van het model op basis van specifieke parameters. Tot slot is het post-processing segment gericht op het detecteren van getrainde objecten op basis hoge resolutie remote sensing beelden en de terugkoppeling met het REV. De resultaten hiervan tonen inzichten in de nauwkeurigheid en bruikbaarheid van model.

Belangrijk hierbij is te benoemen dat dit op globale methodiek in zal gaan. Per gespecificeerd objecttype zijn er in de pre- en post processing segmenten namelijk verschillen. Deze worden toegelicht in het onderdeel 3.5.1.

4.4.3 Pre-processing

Het eerste onderdeel binnen het datamodel betreft de pre-processing van het YOLOv5 objectdetectiemodel. Het proces begon bij het selecteren van een objecttype, voortkomend uit het REV. Vervolgens was het van belang om trainingsdata te verzamelen zodat het model getraind kon worden om de specifieke objecttypen te detecteren. Hier is uitgegaan van één classificatie, maar technisch gezien kan hier nog een verdeling worden gemaakt tussen positieve- en negatieve objecten. Een negatief object is hierin een object dat fysieke overeenkomsten heeft met het gewenste object. Het model kan deze objecten daardoor foutief detecteren. Door de foutgevoelige objecten te trainen in het model als negatief object kunnen deze beter onderscheiden kunnen worden van elkaar.

Na het verzamelen van de trainingsdata is het van belang te controleren of de annotaties beschreven staan in het YOLOv5 formaat. Wanneer dit niet het geval is, is het noodzakelijk om de originele *bounding boxes* te converteren naar het correct formaat. De exacte stappen die genomen zijn om deze conversie toe te passen zijn in detail toegelicht in paragraaf 4.5.1. Hierbij dient rekening gehouden te worden aan de voorwaarden: een YOLOv5 coördinaat is opgebouwd uit minimaal één tekstregel, bestaand uit een numerieke waarde dat een object klasse reflecteert en vier genormaliseerde dimensies tussen 0 en 1, zie figuur 30. Deze coördinaten vormen de *bounding boxes*.

0 0.480109 0.631250 0.692969 0.713278

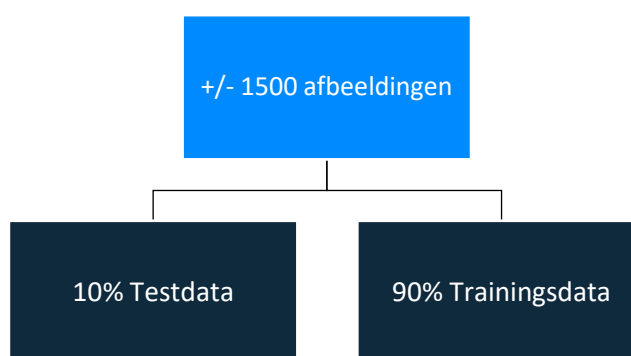
Figure 30, class center_x, center_y, width, height

Het YOLO formaat gebruikt genormaliseerde waarden van het verticale en horizontale middelpunt en de genormaliseerde breedte en hoogte van de bounding box. Dit in tegenstelling tot het pascal_voc formaat die de absolute pixel waarden van de bounding box hoekpunten gebruikt om de bounding box op te slaan. Figuur 31 geeft verschillende formaten weer van de meest gebruikte formaten in objectdetectie.

pascal_voc [x_min, y_min, x_max, y_max] → [98, 345, 420, 462]
albuementations normalized [x_min, y_min, x_max, y_max] → [0.153125, 0.71875, 0.65625, 0.9625]
coco [x_min, y_min, width, height] → [98, 345, 322, 117]
yolo normalized [x_center, y_center, width, height] → [0.4046875, 0.8614583, 0.503125, 0.24375]

Figuur 31, De vier meest gebruikte formaten waarin bounding boxes in worden opgeslagen.

Het volgende preparatieproces is gericht op de hoeveelheid afbeeldingen binnen een dataset. Er is geen objectief minimum, maar in dit model is uitgegaan van 1500 afbeeldingen. Wanneer dit aantal niet wordt bereikt is het mogelijk de hoeveelheid trainingsdata uit te breiden door gebruik te maken van data augmentatie. In dit proces worden afbeeldingen visueel vervormd om het model te optimaliseren, uitgebreide toelichting hierover is in te zien in 2.4.4. Om het overzicht van de dataset te behouden is aangehouden om de benaming van de afbeeldingen en annotatie bestanden te standaardiseren. Het is hierbij belangrijk dat de benaming van de annotaties en de afbeeldingen hetzelfde zijn. De naamgeving van deze bestanden is de enige manier waarmee de annotatie data referentie houdt met de specifieke afbeeldingen. Dit standaardisatieproces is geautomatiseerd en verder beschreven in bijlage **C.I**. Wanneer aan deze stappen zijn voldaan kan de ingewonnen data worden verdeeld over trainingsdata (90%) en test/validatiedata (10%), zie figuur 32. Tot slot moet de folderstructuur worden ingericht conform het YOLOv5 formaat.



Figuur 32, test en validatiedata

Folderstructuren kunnen net als annotatieformaten afwijken per ODS algoritme. YOLOv5 vereist dat de dataset opgedeeld is in een "train" folder en een "valid" folder. Beide folders moeten de folders "images" en "labels" bevatten. De exacte benaming van deze folders zijn belangrijk, net als de verdeling tussen train- en testdata:

- De "train\\images" en "train\\labels" folders bevatten de afbeeldingen en annotaties die gebruikt zullen worden tijdens het trainingsproces.
- De "valid\\images" en "valid\\labels" folder worden tijdens het trainingsproces gebruikt ter validatie.

Het opsplitsen van de trainingsdata is in detail beschreven in bijlage **C.III**. De automatisering van het opstellen van data en de datamigratie naar de correcte folderstructuur is geautomatiseerd. Dit is gedaan door middel van een Python script beschreven in bijlage **C.II**. In bijlage **X.IV** is verder nog

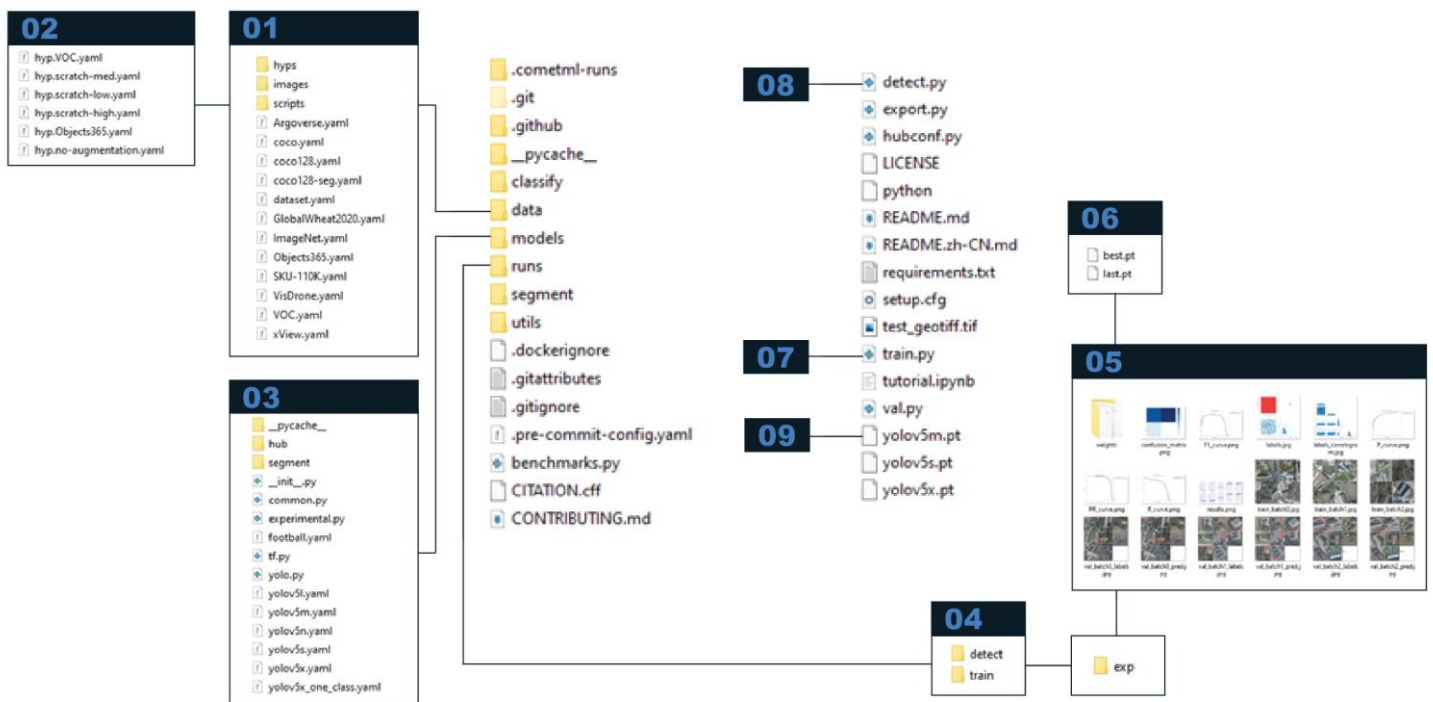
een optioneel script toegelicht waarmee de geannoteerde data en de afbeeldingen visueel gecontroleerd kunnen worden.

4.4.4 Processing

Het onderdeel *processing* is gericht op het installeren van YOLOv5, het prepareren en bepalen van parameters, het initiëren van het trainingsproces en de voorspelling van objecten op basis van het getrainde model. Deze taken zijn onderverdeeld in drie delen; het installatieproces, het preparatieproces en het initiatieproces.

4.4.4.1 Installeren van YOLOv5

YOLOv5 is gedistribueerd via een open source GitHub. Deze GitHub installatie bevat een folderstructuur waarin de volledige code voor de backbone, neck en head zijn opgebouwd. Daarnaast bevat deze folderstructuur alle benodigde functies om bepaalde processen te initiëren zoals `train.py` en `detect.py`. In figuur 33 zijn de belangrijkste elementen van het algoritme omschreven. Conventioneel gebruik leert dat het praktisch is deze YOLOv5 folderstructuur op te slaan in de root folder van het OS systeem.



Figuur 33, overzicht folderstructuur

- ① De "data" folder beheert de metadata van training en validatie datasets. Deze metadata verwijst naar het aantal klassen binnen de dataset en de locaties waar deze zijn opgeslagen.
- ② De "data" folder bevat de sub folder "hyps". De "hyps" folder beheert verschillende varianten van *pre-defined* hyperparameters.
- ③ De "models" folder bevat de complexe architectuur van YOLOv5. Deze folder beheert zes verschillende varianten van de YOLOv5 backbone, neck en head structuur zoals beschreven in hoofdstuk x.
- ④ Deze folder bevat twee sub folders; "detect" en "train". In deze folders worden de resultaten opgeslagen wanneer een trainingsproces of detectie proces wordt geïnitieerd.
- ⑤ Nadat een trainingsproces is afgerond, is een exp folder aangemaakt met hierin bepaalde relevante statistische gegevens zoals de F1 curve, een confusion matrix en de mAP.
- ⑥ Tijdens het trainen van een YOLOv5 model, bijvoorbeeld geïnitieerd voor 300 epochs, wordt de statistische voornaamste iteratie en de laatste iteratie opgeslagen als zogenoemde pt bestanden. Deze bestanden komen in de "weights" folder terecht.
- ⑦ Het train.py bestand is een Python script waarmee een trainingsproces gestart kan worden.
- ⑧ Het detect.py bestand is een Python script waarmee een weging van een model kan worden ingezet om bepaalde objecten te detecteren.
- ⑨ Het yolovm.pt bestand is een *pre-trained* model dat bij het initiëren van een trainingsproces gebruikt kan worden als uitgangspunt (hoofdstuk verwijzing).

YOLOv5 vereist een aantal afhankelijkheden (*dependencies*) om te kunnen functioneren. De Python modules, opgesomd in figuur 34, inclusief het Pytorch framework zijn vereist correct te worden geïnstalleerd.

<code>gitpython>=3.1.30</code>	<code>scipy>=1.4.1</code>
<code>matplotlib>=3.2.2</code>	<code>thop>=0.1.1</code>
<code>numpy>=1.18.5</code>	<code>torch>=1.7.0</code>
<code>opencv-python>=4.1.1</code>	<code>torchvision>=0.8.1</code>
<code>Pillow>=7.1.2</code>	<code>tqdm>=4.64.0</code>
<code>psutil</code>	<code>tensorboard>=2.4.1</code>
<code>PyYAML>=5.3.1</code>	<code>pandas>=1.1.4</code>
<code>requests>=2.23.0</code>	<code>seaborn>=0.11.0</code>

Figuur 34, afhankelijkheden

4.4.4.2 Preparatie parameters

In dit stadium van de methode zijn alle benodigde data verzameld, de datasets geprepareerd en de YOLOv5 architectuur en het framework gedownload en geïnstalleerd. De laatste stap dat moet worden uitgevoerd voordat het trainingsproces gestart kan worden is het voorbereiden van de parameters. Termen die eerder benoemt zijn in hoofdstuk 2.5.4 komen terug in dit onderdeel.

Ten eerste dient een YAML-bestand geschreven te worden die verwijst naar alle relevante datafolders. Deze moet de root folder bevatten van de datasets, de trainingsdataset en de validatiedataset zoals ook omschreven is in 2.5.4. Daarnaast moet het YAML-bestand het aantal objecttypen bevatten die getraind worden op een numerieke referentie en een tekstuele definitie van het object. Daarna moeten de juiste hyperparameters ingesteld worden. De folderstructuur van YOLOv5 bevat een aantal voorgestelde hyperparameter bestanden zie figuur 33. Het is ook mogelijk om hyperparameters te optimaliseren naar eigen invulling, echter is de impact van deze parameters niet daadwerkelijk meetbaar. Daarom wordt in dit onderzoek gebruik gemaakt van de voorgestelde `hyp.scratch-low` en `hyp.scratch-med` parameters. Voor een voorbeeldweergave de source-code van deze hyperparameters wordt gerefereerd naar bijlage **F.I**.

De daaropvolgende parameter betreft het vaststellen van de juiste voorgetrainde weging als uitgangspositie waarmee verder getraind zal worden. Zoals in hoofdstuk 2.5.2 is uitgelegd bestaat de keuze uit nano, small, medium, large en extra-large. Voor het testen van de modellen in dit onderzoek is gebruik gemaakt van medium aangezien deze een goed balans houdt tussen efficiëntie en prestatie. Mocht de werkwijze in productie worden toegepast kan besloten worden om verder te trainen met grotere *pre-trained models*.

Daarna zal de afbeelding grootte dat het model zal gebruiken bepaald moeten worden. Wanneer de afbeelding niet de aangegeven grootte bevat zal YOLOv5 de ingevoerde data automatisch vervormen. Dit kan leiden tot verlies van kwaliteit van het object. Daarom is tijdens de productie van de dataset zie hoofdstuk x vooraf bepaald gebruik te maken van het afbeelding formaat 1280 x 1280 pixels. Volgens de uitgever van YOLOv5 is dit formaat het beste voor prestatiegerichtes doeleindes. Een andere standaard maat die veel gebruik wordt is 640 x 640 pixels. Dit standaard formaat zou de beste balans tussen prestatie en snelheid opleveren.

Vervolgens moet de batch-size bepaald worden. De batch, zie 2.5.4, is voor dit onderzoek ingesteld op 4. Deze waarde is vrij laag, echter door de limitatie in aanwezige hardware was dit de beste optie. Tot slot dient een epoch aantal vastgesteld te worden. Voor dit onderzoek is de epoch ingesteld op 3000. Dit betekent dat de volledige dataset van 1100 tot 1500 afbeeldingen, 3000 keer als input aan het CNN wordt gevoed. Het is daarentegen mogelijk dat het model eerder stopt. Dit gebeurt wanneer het model geen vooruitgang heeft gedetecteerd in de laatste 100 iteraties.

4.4.4.3 Initiëren training en detectie

Met de parameters vastgesteld in 4.4.4.1 is het mogelijk om het trainingsproces te initiëren. Hiervoor wordt de `train.py` functie aangeroepen met de volgende *flags*:

```
Python train.py --img 1280 --epochs 3000 --data data\dataset.YAML --weights yolov5m.pt
--batch-size 4 --hyp data/hyps/hyp.scratch-med.YAML
```

Dit begint een proces dat afhankelijk van de beschikbare hardware tussen 5 en 25 uur kan duren. Nadat dit proces succesvol is afgerond zal de beste weging opgeslagen worden en kunnen bepaalde statistieken opgevraagd worden zoals de mAP en AP. Deze weging zal gebruikt worden om over een vooraf geselecteerd testgebied het geleerde object te detecteren. Om het detectieproces te initiëren wordt de `detect.py` functie aangeroepen. Dit proces wordt in dit onderdeel nog niet gestart. Voordat deze geïnitieerd zal worden dienen een aantal post-processing stappen vooraf uitgevoerd te worden. Deze zijn beschreven in de paragraaf 4.4.5. Net als de `train.py` functie heeft ook deze functie verschillende *flags*. Deze regel ziet er als volgt uit:

```
Python detect.py --weights \yolov5\runs\train\exp\weights\best.pt --img 1280 --source
\dataset_locatie\ --save-txt --save-conf
```

De eerste parameter verwijst naar de beste weging die tijdens het trainingsproces geleerd is. De tweede parameter stelt het formaat van de afbeeldingen van de testdataset in. De derde parameter verwijst naar de locatie van de testdataset omschreven in 4.3.2 *Testdata*. De vierde parameter zorgt ervoor dat de *bounding boxes* die gevonden worden tijdens het detectieproces als tekst bestand

worden opgeslagen. Deze *flag* is noodzakelijk zodat de *bounding boxes* in post-processing kunnen worden geëxporteerd naar GIS software. Tot slot de vijfde parameter, "--save-conf", zorgt ervoor dat de *confidence threshold*, die is ingesteld bij parameter vier, tijdens het detectieproces in hetzelfde tekstbestand wordt opgeslagen als de bounding boxes, eerder genoemd in parameter vijf. Deze waarde is bij het onderdeel post-processing nodig om een confidence threshold in te stellen.

4.4.5 Post-processing

In 4.4.4, pre-processing, is het ODS systeem voor het betreffende object getraind en zijn die parameters vastgesteld om dit model te gebruiken bij de detectie in nieuwe situaties. Deze paragraaf verwerkt preventieve uitsluiting van gebieden over de testdata en het verknippen van de data in kleinere delen (met behoudt van metadata). Tot slot wordt het model uitgevoerd en deze resultaten geconverteerd naar GIS software.

Deze stappen zijn gevormd zodat optimalisatie kan worden uitgevoerd ten opzichte van de ODS resultaten. Daarnaast wordt in dit onderdeel de afstand berekenend tussen de overeenkomende objecten van het detectiemodel en het REV om de nauwkeurigheid te controleren en daarmee de foutmarge te kunnen doorvoeren.

4.4.5.1 Preventief uitsluiten van gebieden

De testdata, voordat deze is verknipt tot kleinere delen, bevat de begrenzing van het vooraf vastgestelde testgebied voor het betreffende object. Dit vooraf bepaalde gebied kan mogelijk bepaalde locaties of typerende gebieden bevatten waar het betreffende object fysiek, wettelijk of logische benaderd onmogelijk aanwezig kan zijn.

De testlocaties en gebieden zijn geselecteerd op basis van bestaande wetgeving, fysieke onmogelijkheid en aanwezigheid van foutgevoelige objecten. Door verschillende gebieden te testen kan geobserveerd worden waar het model foutieve detecties maakt. Vervolgens zal gereflecteerd worden op deze informatie om te bepalen of deze gebieden mogelijk kunnen worden uitgesloten voordat een tweede of derde detectie wordt uitgevoerd. Zo zijn bijvoorbeeld sportvelden frequent gedetecteerd als windturbines. Deze locaties of gebieden worden uit de originele GeoTIFF raster gefilterd om zo mogelijke foutieve detectie te voorkomen. Deze handelingen zijn geautomatiseerd in het Python script dat is bijgeleverd in bijlage **D.II**. In paragraaf 4.5 worden deze weg gefilterde gebieden verder gedefinieerd.

Zodra het excluderen van gebieden is afgerond zal het rasterbestand in kleinere delen moeten worden geknipt. Zoals eerder benoemd is in paragraaf 4.3.2 is voor dit onderzoek besloten een

formaat van 1280 x 1280 pixels aan te houden. Dit is gedaan door middel van een Python script bijgeleverd in bijlage **A.I**. Dit script maakt gebruik van de GDAL functionaliteit waardoor de metadata van de geknipte delen behouden blijft. Het behoud van de metadata is hoofdzakelijk tijdens de conversie naar GIS-software. De rasterdelen dienen namelijk als input voor het detectie model dat in dit stadium van het onderzoek wordt gestart. De volgende paragraaf zal ingaan op het converteren van de resultaten van het detectie model, ofwel, de gedetecteerde objecten.

4.4.5.2 Conversie van bbx naar GIS

De gedetecteerde resultaten zijn afkomstig uit het getrainde ODS model (4.4.4.3). Voor iedere afbeelding waar detectie over is uitgevoerd is een tekstbestand met data opgeslagen. Deze data bevat de object klasse, genormaliseerde coördinaten die de gedetecteerde *bounding boxes* vormen en een confidence waarde. De coördinaten verwijzen naar vier locaties van de betreffende afbeelding. Dit zijn geen ruimtelijke coördinaten en kunnen daarom niet direct in GIS software gebruikt worden. Deze coördinaten dienen daarvoor eerst omgezet te worden in ruimtelijke coördinaten zodat hier een polygoon mee gevormd kan worden. Dit is gedaan door gebruik te maken van de metadata van de geknipte GeoTIFF bestanden die gebruikt zijn als input van het objectdetectie model. Deze metadata bevat referentie naar ruimtelijke data zoals projectiesystemen en locaties.

Doordat de metadata van de geknipte TIFF bestanden behouden is gebleven is het mogelijk de ruimtelijke coördinaten te extraheren op basis van de gestandaardiseerde coördinaten. Hiervoor is het nodig om de gestandaardiseerde coördinaten om te zetten naar pixel-waarden. Door binnen het TIFF bestand te navigeren naar deze pixel-waarden, is het mogelijk om op basis van de metadata achter het TIFF bestand de reflecterende ruimtelijke coördinaten te extraheren. Nadat de correcte ruimtelijke coördinaten zijn weggeschreven naar een tijdelijke variabele wordt ook de *confidence* waarde uit het tekstbestand in een tijdelijke variabele opgeslagen.

Vervolgens worden deze ruimtelijke coördinaten gebruikt om polygoon op te slaan in shapefile formaat en zal een kolom aangemaakt worden voor deze polygoon waarin de confidence waarde kan worden weggeschreven. In het script "gisify.py" zijn deze stappen geautomatiseerd, zie bijlage **D.I**. Doordat de polygoon de *confidence-waarde* toegewezen krijgt is het in een later stadium mogelijk om bepaalde filter functies of graduele symbologie toe te passen. Hiermee kan de *confidence threshold* worden bepaald.

4.4.5.3 Berekenen van nauwkeurigheid

De polygonen resulterend uit 4.4.5.2 zijn vierhoekige shapefile objecten. Deze polygonen geven de *bounding boxes* weer op basis van het objectdetectie-bestand waar de objecten gevonden zijn. In deze paragraaf wordt de afstand berekend van het dichtstbijzijnde REV object ten opzichte van het middelpunt van de polygoon. Alle gevonden afstanden langer dan een specifieke afstand is gelabeld als "geen match". Vanaf deze, object afhankelijke, bodemwaarde tot aan het minimum gevonden afstand zal met evenredige intervallen gegradeerd worden op nauwkeurigheid en waarschijnlijkheid. Deze GIS taken zijn uitgevoerd in ArcGIS Pro doormiddel van de *arcpy.analysis.near* functie. Hoe groter de afstand tot een gevonden REV object de groter de kans dat dit een niet geregistreerd object bevat.

4.4.5.4 Controleren van resultaten

In het vorige onderdeel zijn de gedetecteerde objecten als punt geometrie toegevoegd in een GIS platform. Deze zijn vervolgens gevisualiseerd op basis van de afstand tot de huidige REV objecten. De volgende taak betreft het controleren van de correctheid van de punten die niet overeenkomen met de objecten uit het REV. Door gebruik te maken van Google Street view kan het bestaan van het objecten bevestigd worden. Hoewel deze methode veelal effectief werkt zijn er situaties waarin de Street View functie niet kan worden ingezet. In deze situaties kan als alternatief een controle uitgevoerd worden door een hoge resolutie luchtfoto's te bekijken. Deze methode is minder betrouwbaar. Dit komt doordat bij een vogelvlucht perspectief bepaalde vormen en kleuren minder duidelijk te definiëren zijn.

4.5 Dataverwerking (object specifiek)

In dit onderdeel worden bepaalde zaken toegelicht waar de globale methodologie afwijkt afhankelijk van het specifieke object. Dit omvat zaken zoals gebiedsbegrenzing, annotatie methodes, dataset samenstelling, uitsluiting van specifieke gebieden, het bepalen van de *confidence threshold* waardes en de toepassing van augmentatie-technieken.

4.5.1 Windturbines

4.5.1.1 Gebiedsbegrenzing / testgebied

Voor ieder object is een specifiek gebied geselecteerd waarbij rekening wordt gehouden met bias, object aanwezigheid en de aanwezigheid van potentieel foutgevoelige objecten. Omdat ieder object fundamenteel anders is in fysieke vorm en ruimtelijke plaatsing is het noodzakelijk deze gebiedsbegrenzing per object te bepalen.

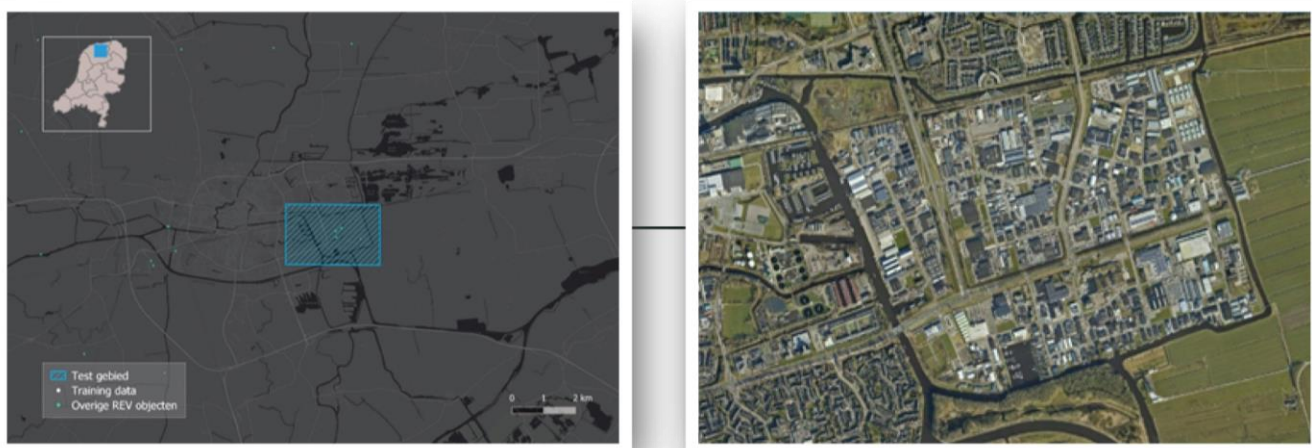
Voor windturbines zijn twee aparte gebieden geselecteerd. Het eerste testgebied voor het object windturbines omvat een rechthoekige begrenzing van 8 x 12 km in het noorden van Flevoland zoals figuur 35 weergeeft. Hiervoor zijn 96 raster tegels van 1 x 1 km gedownload.



Figuur 35, testgebied 1 van windturbines

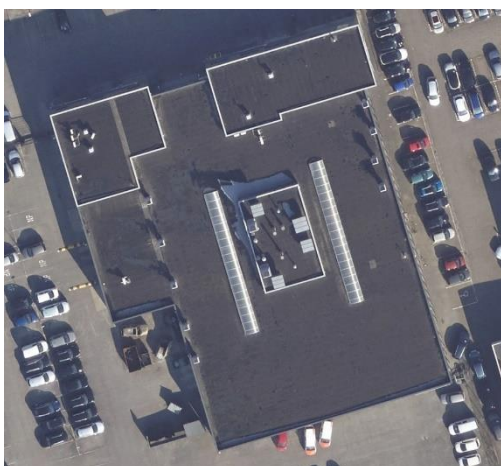
In figuur 35 zijn de objecten weergegeven die gebruikt zijn voor het trainingsproces (witte punten) en welke niet gebruikt zijn voor het trainingsproces (blauwe punten). Hier is te zien dat het geselecteerde gebied alleen objecten bevat die niet gebruikt zijn voor training. Zoals eerder benoemt is dit gedaan zodat de resultaten niet beïnvloed worden door bias data. Het gebied bevat

83 windturbines volgens de registratie van het REV. Mocht het model meer of minder punten vinden betekent dit dat ofwel het REV ofwel het model foutieve gegevens bevat. Voor het eerste testgebied met betrekking tot windturbines is specifiek gekozen voor een gebied uit de provincie Flevoland aangezien hier hoge cluster van windturbines aanwezig is. Echter, bevat dit gebied niet veel "lastige" objecten doordat het hoofdzakelijk uit akkerbouw bestaat. Dit gebied is relatief monotoon en consistent in fysieke kenmerken waardoor het model niet voldoende getest wordt op potentiële limitaties. Daarvoor is een tweede gebied geselecteerd, zie figuur 36.

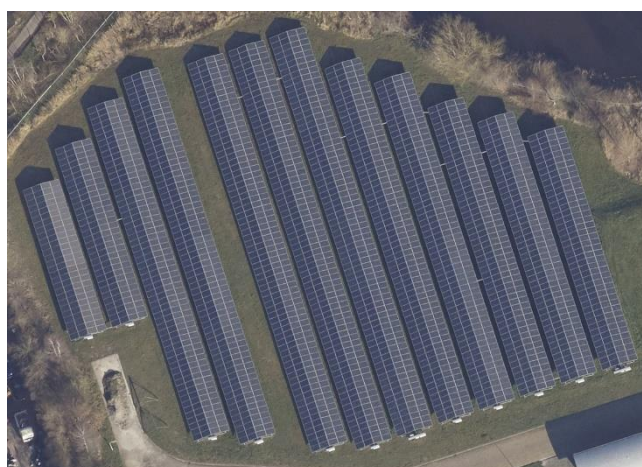


Figuur 36, testgebied 2 van windturbines

Het tweede Testgebied voor het object windturbines omvat een rechthoekige begrenzing van 2 x 3 km. Het gebied omvat het dorp Hemrik gelegen ten zuidoosten van de stad Leeuwarden in de provincie Friesland. Het gebied bevat slechts vier windturbines volgens de registratie van het REV. Echter, door de dichtbebouwde eigenschap van dit gebied zijn hier veel potentieel foutgevoelige objecten aanwezig zoals verticaal geplaatste zonnecellen en dakramen van warenhuizen en kunstgras sportvelden zie figuur 37 en 38.



Figuur 37, dakramen



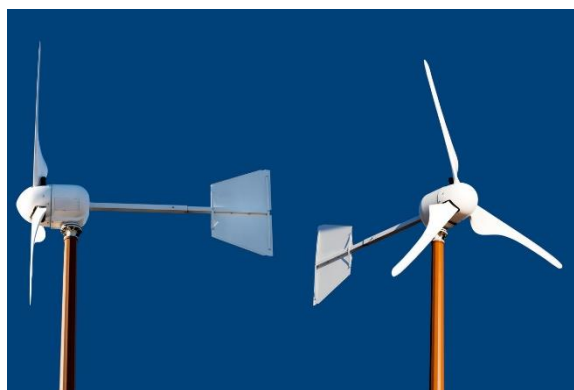
Figuur 38, zonnecellen

4.5.1.2 Afwijkende objectklassen gedefinieerd

Tijdens het handmatige annoteren van de data zijn specifiek afwijkende windturbine modellen gevonden. Deze afwijkende modellen zijn minder groot in formaat en hebben afwijkende fysieke eigenschappen, zie figuur 39 en 40. Door deze unieke fysieke karakteristieken is besloten deze windturbine modellen onder een andere object klassen te annoteren, deze klassen is onder de noemer "windturbine_1" gelabeld.

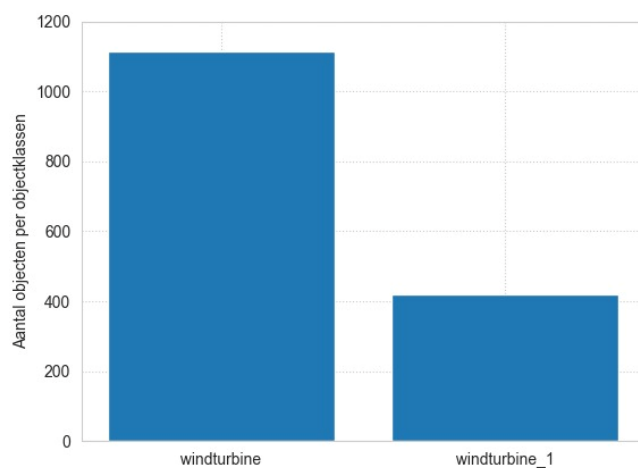


Figuur 39, windturbine_1



Figuur 40, windturbine_1

Doordat deze klasse pas werd vastgesteld tijdens het annotatieproces is hier geen rekening mee gehouden bij het inwinnen van trainingsdata. Dit verklaart het verschil in de hoeveelheid geannoteerde afbeeldingen tussen de objectklassen. Met de data die gedownload is voor annotatie doeleindes was het mogelijk annotaties te maken van windturbines en 418 annotaties te maken van windturbine_1. In figuur 41 is een grafiek te zien wat deze totaal verdeling per objectklassen aantoont.



Figuur 41, verdeling in objectklassen

4.5.2.1 Preventieve Augmentatie

Uitvoering 2 van windturbines is een tweede keer getraind echter ditmaal met een door augmentatie-technieken aangevulde trainingsdataset. Verder in dit document zal naar het geaugmenteerde model gerefereerd worden als “uitvoering 2 (aug)”. Reflecterend op de resultaten van uitvoering 2 zijn zeven verschillende augmentatie-technieken geselecteerd gebruikmakend van de albumentation module in Python. De volgende technieken zijn hiervoor gebruikt;

- RandomCrop
- RandomSizeBBoxSafeCrop
- ElasticTransform,
- GridDistortion,
- ToGray
- RandomBrightnessContrast
- Flip

Deze technieken zijn geselecteerd op basis van een aantal limitaties waar uitvoering 2 mee te maken had.

RandomCrop en RandomSizeBBoxSafeCrop zijn gebruikt om het model robuuster te maken voor ook gedeeltelijk zichtbare objecten. Door de crop techniek toe te passen wordt de kwantiteit van deze gedeeltelijk zichtbare objecten vergroot waardoor het model deze beter kan trainen en daardoor beter leert omgaan met dit soort scenario's. Daarnaast is besloten GridDistortion en ElasticTransform toe te passen zodat het model beter afwijkende geometrieën kan leren herkennen. Dit zou ervoor moeten zorgen dat het model een middellijn van een sportveld bijvoorbeeld beter kan onderscheiden van een wiel.

Tot slot is de augmentatietechniek RandomBrightnessContrast toegepast. Deze techniek past de helderheid aan. Dit kan ervoor zorgen dat het model minder afhankelijk wordt van kleur. Hierdoor wordt de vorm van bepaalde geometrieën meer leidend bij het detecteren van een object. Op deze manier kan het model een achtergrond, waar met hoge frequentie kleinere witte objecten op zijn weergegeven, beter negeren.

Op basis hiervan zijn twee augmentatietechnieken geselecteerd. In de code van het script waarin deze transformaties worden geïnitieerd is een procentuele waarde ingesteld. Deze waarde is representatief voor de waarschijnlijkheid dat de techniek wordt toegepast. Wanneer deze waarde op 0.5 is ingesteld, is de waarschijnlijkheid dat deze 50% van de keren zal worden toegepast. Daarnaast is gekozen om slechts één techniek te gebruiken per afbeelding. Dit is gedaan omdat te veel transformaties op een afbeelding te veel ruis zal veroorzaken.

Tot slot is er voor iedere transformatie een rotatie toegepast zodat geen enkele afbeelding dezelfde oriëntatie bevat als het niet-getransformeerde origineel. Deze zeven technieken hebben het aantal geannoteerde trainingsdata uitgebreid met 1011 getransformeerde afbeeldingen. Alle stappen die zijn ondernomen om deze augmentaties toe te passen zijn verder toegelicht in bijlage **E.I**.

4.5.1.3 Annotatie conversie: pascal_voc / YOLOv5

Zoals eerder in dit document is uitgelegd bestaan er verschillende annotatieformaten die ieder compatibel zijn met enkel het hiervoor gespecificeerde ODS. Voor de eerste uitvoering van het detectiemodel met betrekking tot het object, windturbine, zijn openbare datasets gebruikt. Om een zo groot mogelijke dataset samen te stellen is gekozen om de twee gevonden openbare datasets te combineren tot één dataset. Van de twee datasets waren de annotaties van de tweede dataset in een ander formaat, namelijk in pascal_voc. Daarnaast was deze dataset oorspronkelijk gebruikt voor tensorflow waardoor deze opgeslagen was in een csv bestandsvorm, zie figuur 42, in plaats van een tekstbestandsvorm wat YOLOv5 vereist. Om deze dataset om te zetten naar het juiste annotatie formaat en de juiste bestandsvorm zijn een aantal stappen genoodzaakt. Zoals in figuur 42 te zien is, heeft de kolom "class" en tekstuele aanduiding voor de objectklasse. Deze zal ten eerste vervangen moeten worden door een numerieke waarde, in dit geval 0. Vervolgens zal de conversie van *bounding box* dimensies plaatsvinden. Hiervoor zijn de volgende formules gebruikt:

Genormaliseerde x-center (YOLOv5) = $((x_{max}+x_{min}) / 2) / tot_x_{max}$,

Genormaliseerde y-center (YOLOv5) = $((y_{max}+y_{min}) / 2) / tot_y_{max}$,

Genormaliseerde breedte (YOLOv5) = $(x_{max}-x_{min}) / tot_x_{max}$,

Genormaliseerde hoogte (YOLOv5) = $(y_{max}-y_{min}) / tot_y_{max}$

De waardes (xmax, xmin, ymax en ymin) verwijzen naar de (xmax, xmin, ymax en ymin) waardes uit het csv bestand, zie figuur 42. Deze waarden tonen de absolute breedte en hoogte uit de afbeeldingen van de dataset. Nadat deze berekeningen gemaakt zijn dienen de nieuwe waardes weggeschreven te worden naar een tekstbestand zoals in figuur 42 is afgebeeld. Daarnaast moet de naam overeenkomen met de waarde uit de kolom "filename". Dit komt doordat deze verwijst naar de afbeelding waarvan de data afkomstig is.

filename	width	height	class	xmin	ymin	xmax	ymax
m_4209449_sw_15_1_20170701_13_06.jpg	300	300	wind turbine	51	211	83	264
m_4209449_sw_15_1_20170701_13_11.jpg	300	300	wind turbine	1	127	29	187
m_4209426_ne_15_1_20170707_10_07.jpg	300	300	wind turbine	55	120	126	149

Figuur 42, originele csv bestand

4.5.1.3 Preventief uitsluiten van gebieden

Uit de eerste testen met betrekking tot het model uitvoering 2 is gebleken dat sportvelden kunnen worden aangezien als windturbines. Dit wordt verklaard doordat de belijning van het veld in combinatie met de middencirkel gelijkwaardige zichtbare eigenschappen hebben. Hierbij is het belangrijk te benoemen dat dit vrijwel alleen een risico is in het geval dat de middencirkel een contrast heeft met de rest van het veld. Dit soort sportvelden zijn veelal gemaakt van kunstgras in tegenstelling tot echt gras. Figuur 43 geeft een voorbeeld weer waarbij het model een soortgelijk sportveld fout geïnterpreteerd heeft als een windturbine. In figuur 44 is overigens te zien dat een standaard sportveld correct is genegeerd. Om foutieve detectie zoals in het eerste voorbeeld is weergegeven te voorkomen worden deze objecten uit de data gefilterd voordat detectie geïnitieerd wordt. De data die is beschreven in 4.3.5 is hiervoor toegepast. Deze data bevat de data groep "onbegroeid terreindeel". De selectie-filter, "bgt-fysiekVoorkomen" = 'gesloten verharding' AND "plus-fysiekVoorkomen" = 'kunststof', extraheert enkel de objecten waarbij het materiaal van kunststof gemaakt is en de bestaande verharding gesloten is. Dit resulteert in hoofdzakelijk polygonen van kunststof sportvelden.



Figuur 43, kunstgrasveld (foutieve detectie)

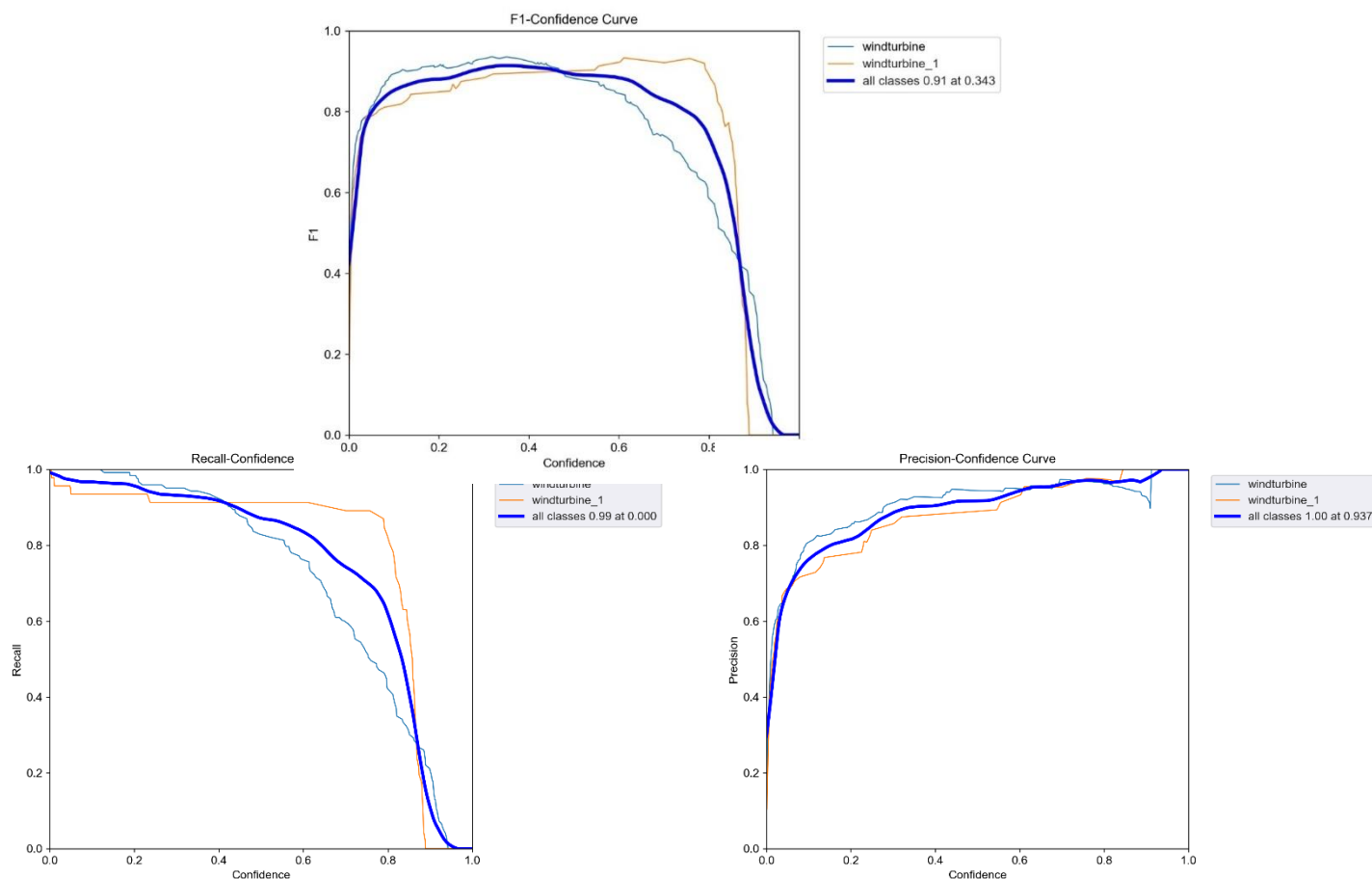


Figuur 44, normaal gras (geen detectie)

4.5.1.3 Confidence Threshold

Na het trainen van het YOLOv5 model worden verschillende statistieken gegenereerd waaronder een grafiek die de F1 en de confidence waarde plot. Een ideale F1-score betekent dat weinig fout-positieven en weinig fout-negatieven objecten gedetecteerd worden. De grafiek in figuur 45 geeft voor verschillende confidence thresholds, de F1-score weer. Deze grafiek bevat gegevens die tijdens het trainingsproces zijn berekend op basis van de recall en precisie waarden. In theorie is de ideale threshold waarde het punt waar de F1-score en de confidence waarde beide het hoogste zijn. Daarentegen is het ook mogelijk op basis van de recall-confidence curve of precision-confidence curve een threshold vast te stellen, zie figuur 45. Door deze grafieken te gebruiken kan worden bepaald welke threshold het meest geschikt is op basis van de behoeften. Zo kan worden gekozen

meer aandacht te besteden aan de precision (nauwkeurigheid) of recall (volledigheid). Bij hoge precisie is het hier mogelijk dat bepaalde objecten worden gemist, maar bij een hoge recall kan het model juist te veel objecten opnemen. Een handmatig controleproces over deze data zal door de hoeveelheid punten arbeidsintensief zijn. Daarnaast is het per studiegebied belangrijk om kritisch om te gaan met deze grafieken en verschillende waarden te testen.



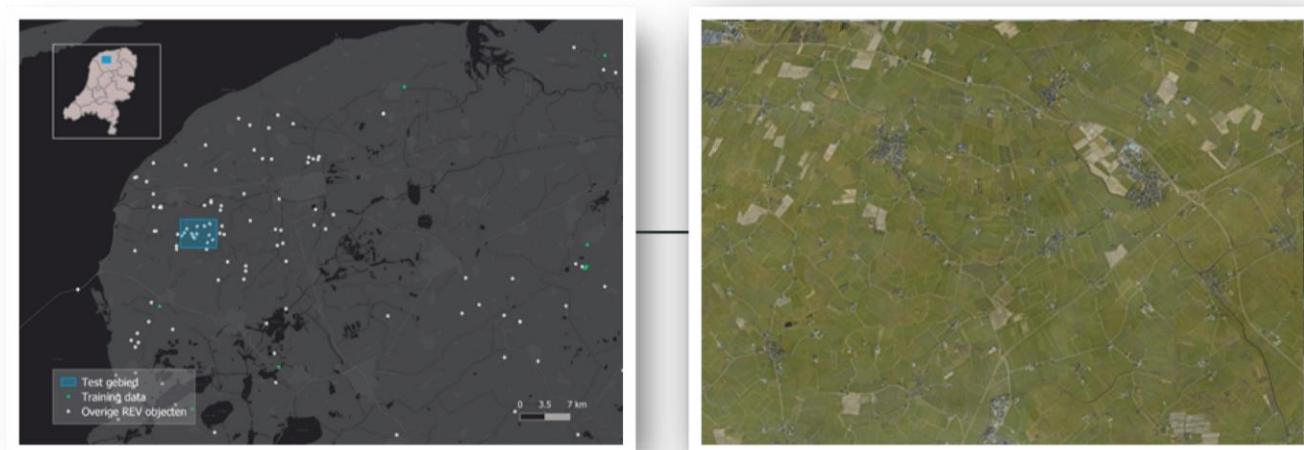
Figuur 45, F1 confidence, recall-confidence en precision confidence curve

Hoewel de *F1-confidence curve* een *threshold* van 0.34 suggereert is besloten af te wijken van deze waarde. Voor windturbines in studiegebied 1 is uitgegaan van een confidence threshold van 0.5 (50%). Deze waarden komen voort uit verschillende experimenten waarin onderzoek is gedaan naar het percentage van correct gedetecteerde objecten. Er is gekozen voor relatief hoge percentages ten opzichte van de F1-confidence curve omdat de kans hierdoor kleiner is dat incorrecte objecten worden getoond. Uit het exploratieve onderzoek over studiegebied 1 is ondervonden dat alle objecten onder de 0.6 % incorrect worden gedetecteerd. Omdat gedetecteerde objecten ongeacht de confidence waardes uiteindelijk overgezet worden naar een GIS platform is het mogelijk om naderhand andere threshold waardes in te laden middels een filter functie of door het toepassen van een graduele kleurenschema.

4.5.2 Propaantanks

4.5.2.1 Gebiedsbegrenzing / testgebied

Voor het testen van propaantanks zijn net zoals dit gedaan is bij windturbines een testgebied geselecteerd. Het testgebied voor het objecttype propaantanks omvat een rechthoekige begrenzing van 4 x 5 km ten zuiden van Leeuwarden zoals figuur 46 weergeeft. Hiervoor zijn 20 raster tegels van 1 x 1 km gedownload.



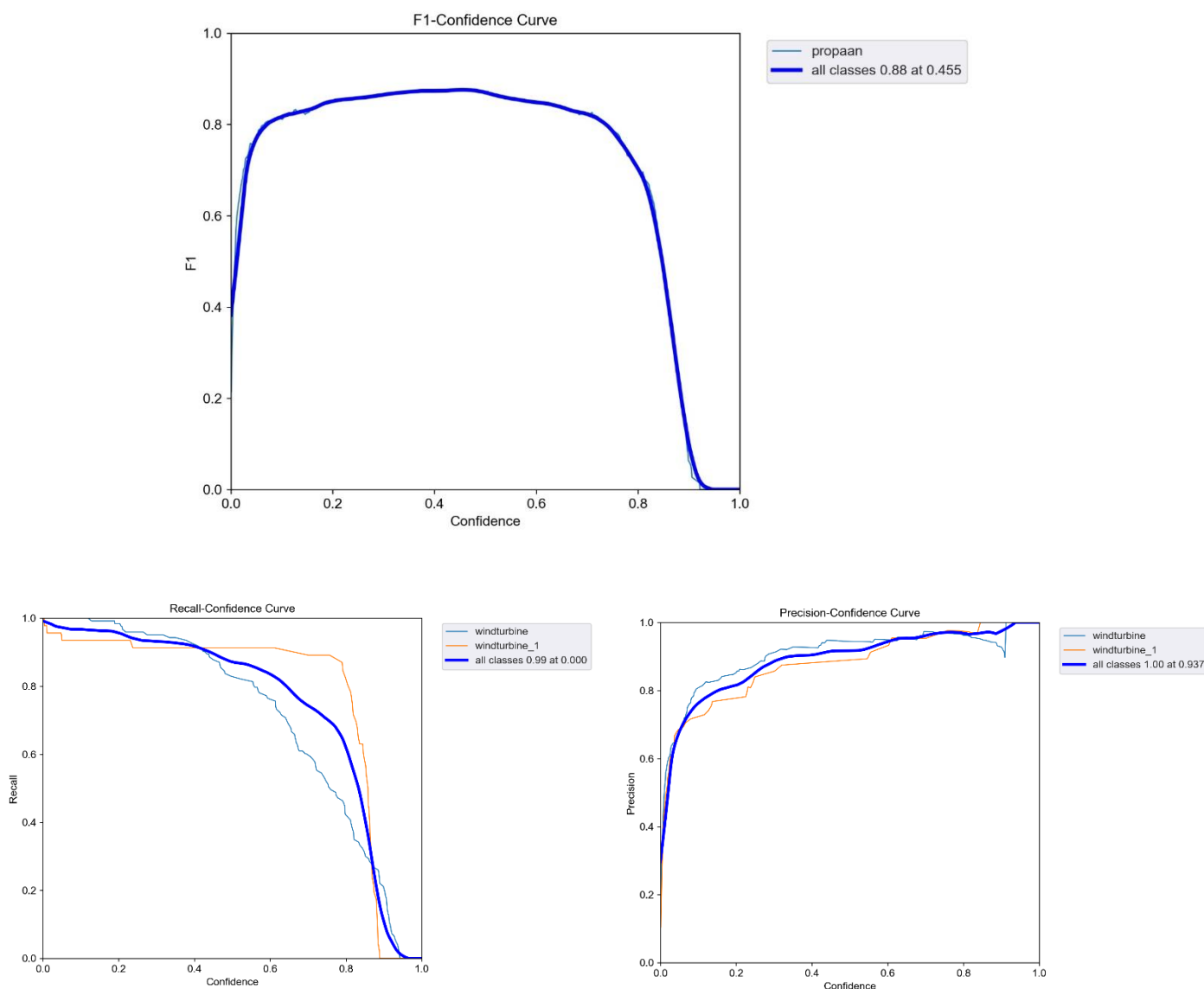
Figuur 46, testgebied propaantanks

4.5.2.2 Preventief uitsluiten van gebieden

Uit de eerste testen met betrekking tot het model voor propaantanks is gebleken dat middelgrote tot grote bussen en aanhangers van vrachtwagens een potentiële bron kan zijn voor foutieve detectie. Aangezien dit type object geen vaste ligging heeft bestaat hier geen directe actuele bruikbare data dat gebruikt kan worden voor gebiedsuitsluiting. Om toch optimalisatie toe te passen door het minimaliseren van foutieve detectie is de keuze gemaakt parkeervlakken, regionale en lokale autowegen preventief weg te filteren. De data dat is beschreven in 4.3.5 "optimalisatie taken" is hiervoor gebruikt. Deze data bevat de data groep "bgt_wegdeel". De selectie-filter, "function" = 'parkeervlak' OR "function" = 'rijbaan autoweg' OR "function" = 'rijbaan regionale weg', extraheert enkel de infrastructurele voorzieningen waar de zogenoemde "lastige objecten" zich zullen bevinden. Hierbij is bewust gekozen inritten en transitiewegen niet mee te nemen aangezien op deze wegen sporadisch propaantanks kunnen bevinden.

4.5.1.3 Confidence threshold

Hoewel de F1-confidence curve een threshold van 0.455 suggereert, is voor propaantanks uitgegaan van een threshold van 0.6 (60%). Deze waarden komen voort uit verschillende experimenten waarin onderzoek is gedaan naar het percentage van correct gedetecteerde objecten. Uit het exploratieve onderzoek over studiegebied 1 is ondervonden dat alle objecten onder de 0.6 % incorrect worden gedetecteerd. Op basis van deze gegevens is een overweging gemaakt en is gekozen om af te wijken van de F1-confidence curve, zie figuur 47. Ook voor propaantanks zijn de recall-confidence en precision confidence curve opgesteld, zie figuur 47. In dit geval is dus uitgegaan van een hogere confidence (precision-confidence curve) dan aangegeven in de F1-confidence curve.



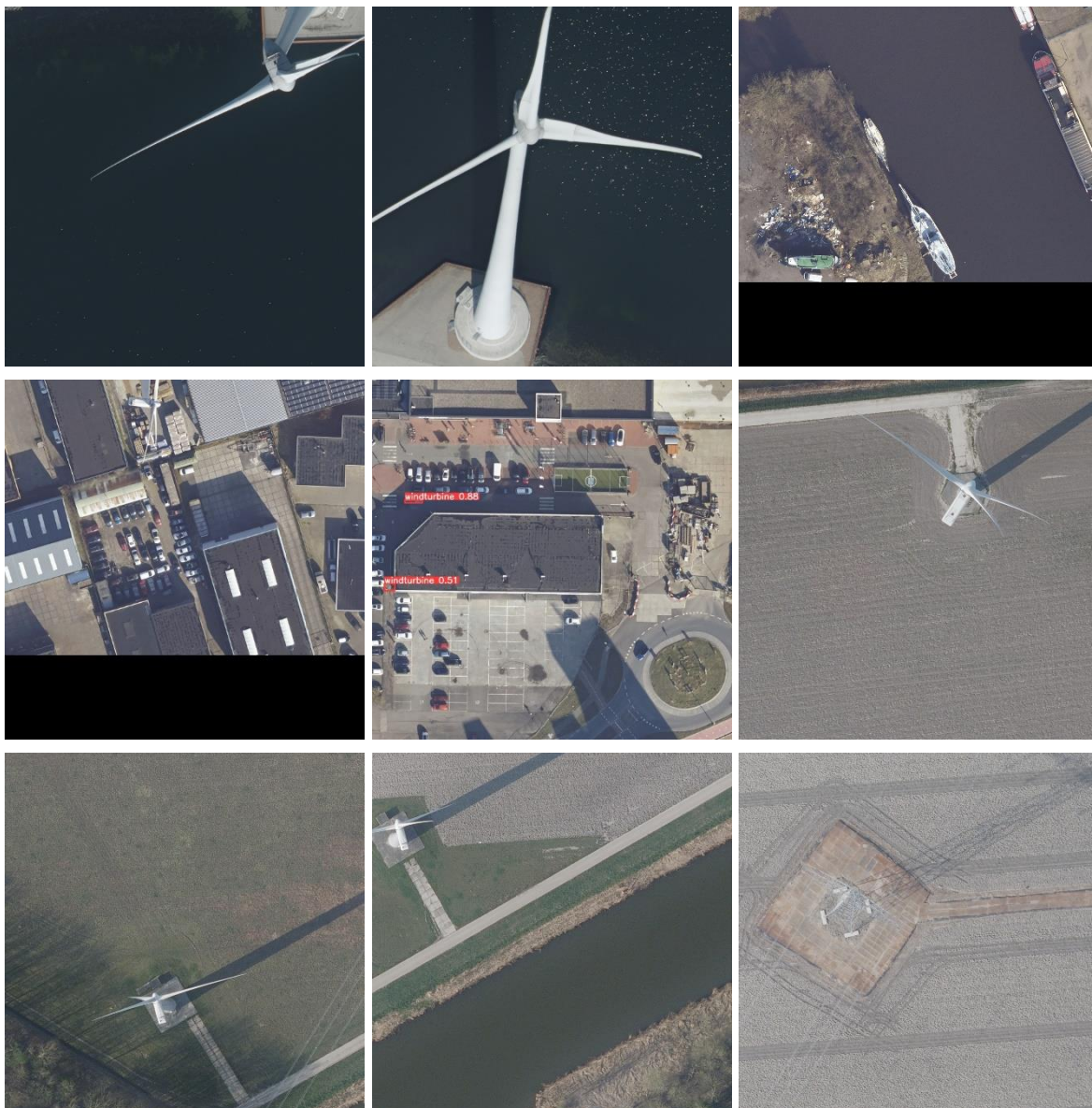
Figuur 47, F1 confidence, recall-confidence en precision confidence curve

5.1 YOLOv5 Modellen

In dit hoofdstuk zullen de resultaten voortkomend uit de methodologie en de verschillende variaties van YOLOv5 modellen worden getoond. Uitgebreidere interpretaties en observaties staan beschreven in Hoofdstuk 6: Conclusie en Discussie.

5.1.1 Windturbines uitvoering 1 (Openbare dataset)

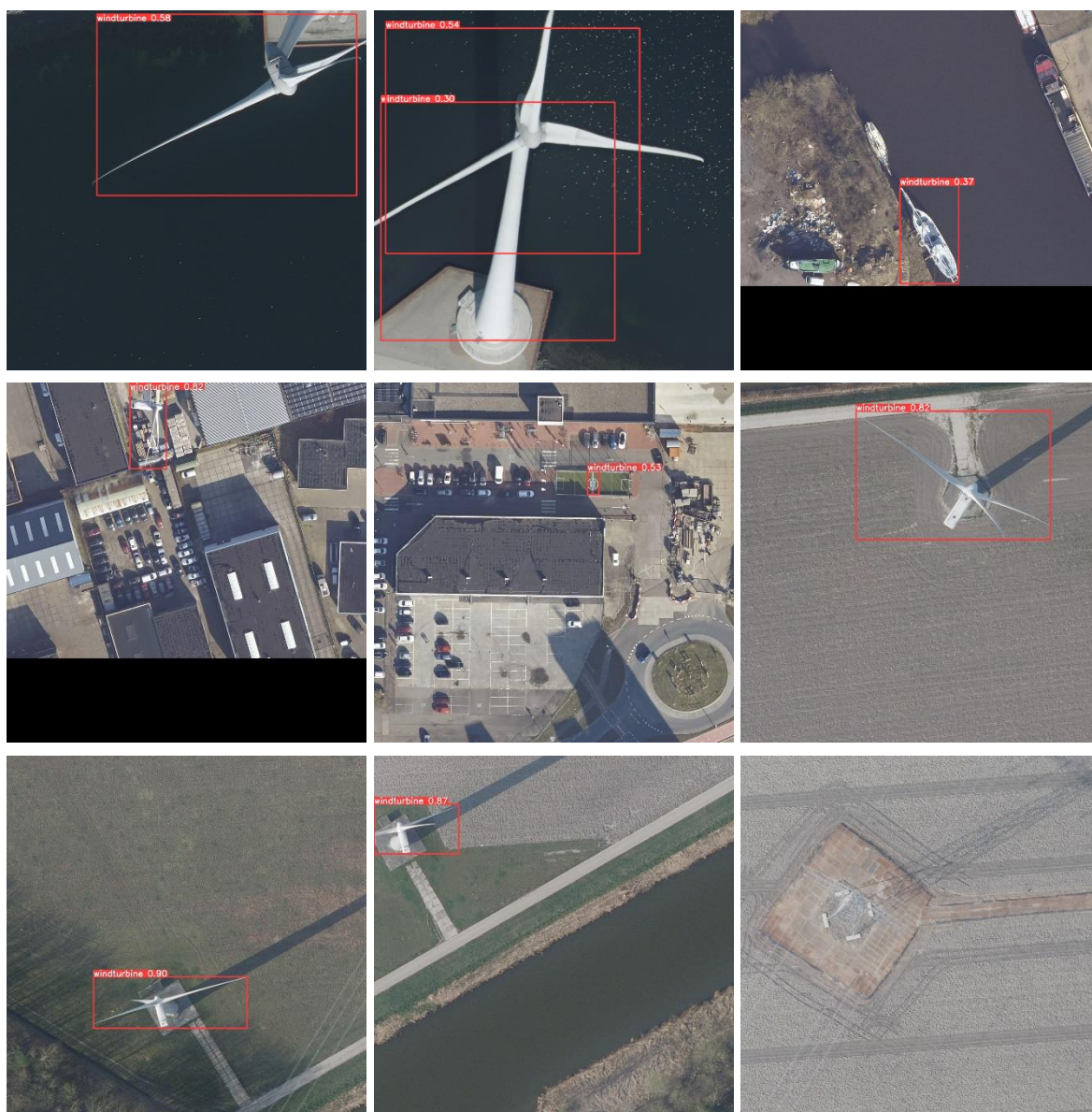
Figuur 48 toont een aantal visualisaties afkomstig uit de eerste uitvoering van het windturbine model. Deze versie is opgesteld op basis van openbaar beschikbare trainingsdata. Het resultaat toont geen detectie.



Figuur 48, uitvoering door middel van openbare data

5.1.2 Windturbines uitvoering 2 (geaugmenteerd)

Figuur 49 toont een aantal visualisaties afkomstig uit een variatie van uitvoering 2 (uitvoering 2 aug) van het windturbine detectiemodel. Deze versie is opgesteld op basis van handgemaakte datasets waarbij augmentatietechnieken zijn toegepast. Deze resultaten tonen wel detecties in tegenstelling tot de eerste uitvoering (Openbare dataset).



Figuur 49, detecties met augmentatietechnieken

5.1.3 Windturbines uitvoering 2

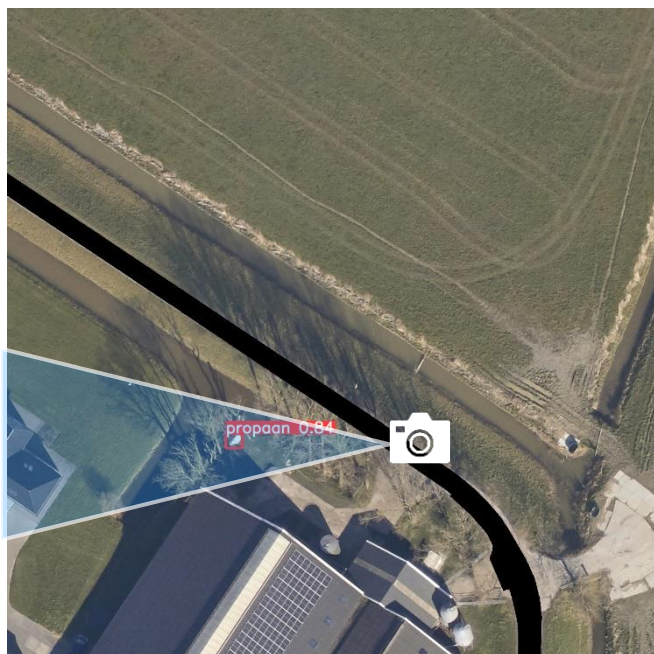
Figuur 50 toont een aantal visualisaties afkomstige van de tweede uitvoering van het windturbine detectiemodel aan. Deze versie is opgesteld op basis van handgemaakte datasets. In tegenstelling tot de eerste uitvoering (Openbare dataset) toont dit model correctie detectie. Daarbij weet dit model ook potentieel foutgevoelige objecten te negeren anders dan het geval was bij uitvoering 2 (aug).



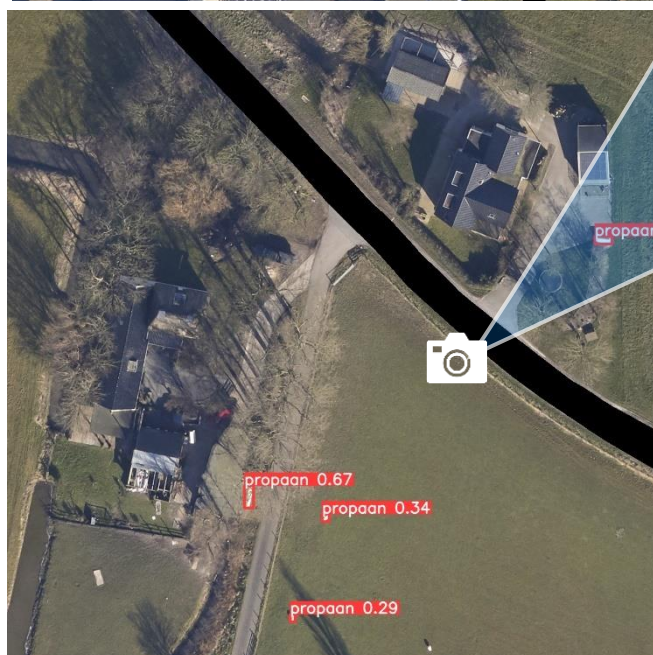
Figuur 50, detecties met op maat gemaakte data (uitvoering 2)

5.1.4 Propaantanks uitvoering 1

Figuur 51, 52 en 53 tonen visualisaties afkomstige van het model dat getraind is voor propaantank detectie aan. Dit model is opgesteld op basis van handgemaakte datasets. De resultaten tonen aan dat dit model in staat is propaantanks correct te detecteren.



Figuur 51, Propaantank detectie



Figuur 52, Propaantank detectie

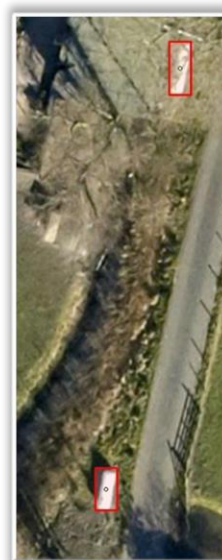
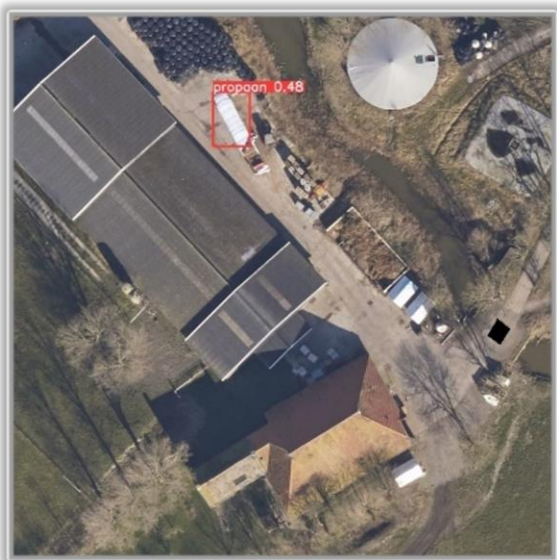


Figuur 53, Propaantank detectie

5.2 Limitaties en uitdagingen

5.2.1 Limitatie propaantanks

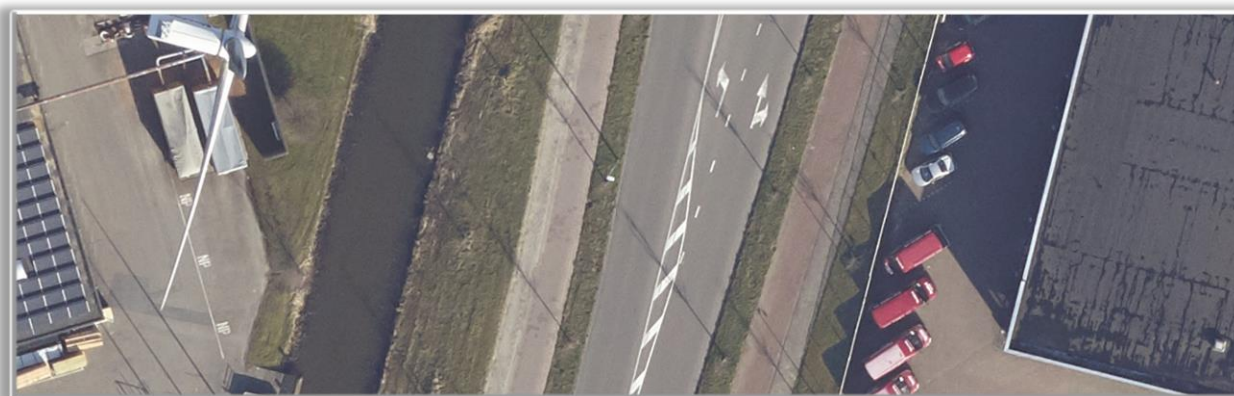
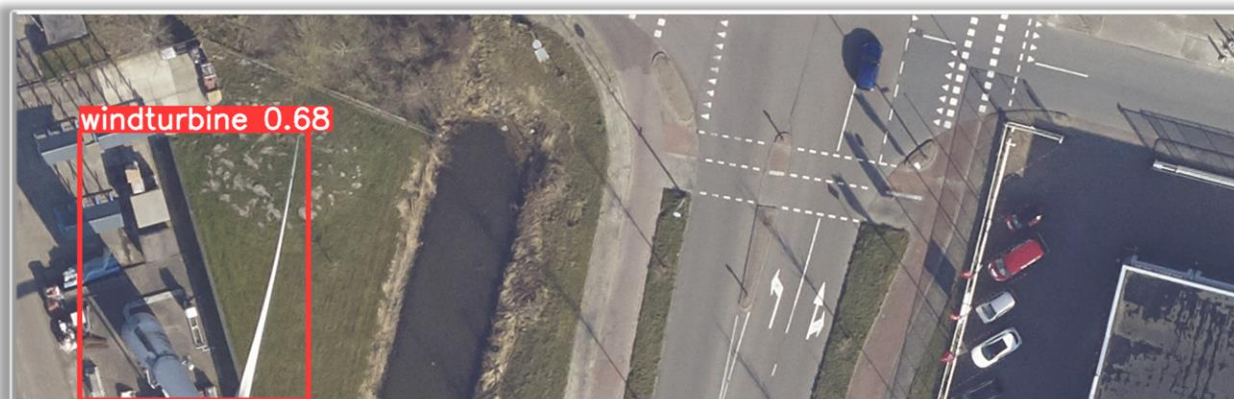
Het model is niet foutloos en toont ook bepaalde limitaties en uitdagingen aan. Zo is het mogelijk dat schapen, bussen en vrachtwagens worden gezien als propaantanks, zie figuur 54 en 55. Daarentegen laat figuur 56 zien dat het model ook met de aanwezigheid van een licht bladerdak functioneert.



Figuur 54, limitatie van vrachtwagens Figuur 55, limitatie van schapen Figuur 56, bladerdak

5.2.2 Limitatie windturbines

Ook het windturbine model is niet foutloos en toont ook bepaalde limitaties en uitdagingen. In situaties waar een windturbine niet volledig op de afbeelding staat is het mogelijk dat objecten niet gedetecteerd worden of met een lagere *confidence* waarde gedetecteerd worden. Daarnaast zijn kunstmatige sportvelden en windturbines en opbouw aangemerkt als potentieel foutgevoelig object. In figuur 57 en 58 zijn voorbeelden te zien van deze scenario's.



Figuur 57, limitatie van object splitsing

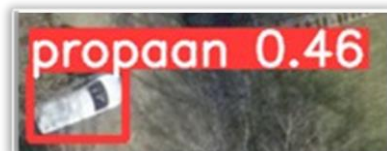


Figuur 58, limitatie van kunstmatige sportvelden & aanbouw windturbines

5.2.3 Exclusie gebieden

Om te voorkomen dat bepaalde objecten incorrect gedetecteerd worden zijn stappen ter preventie hiervan toegepast. De resultaten hiervan zijn gevisualiseerd in figuur 60, 61, 62 en 63.

Propaantanks:



Figuur 60, limitatie van bussen



Figuur 61, preventie limitatie

Windturbines:



Figuur 62, limitatie van kunstmatige sportvelden

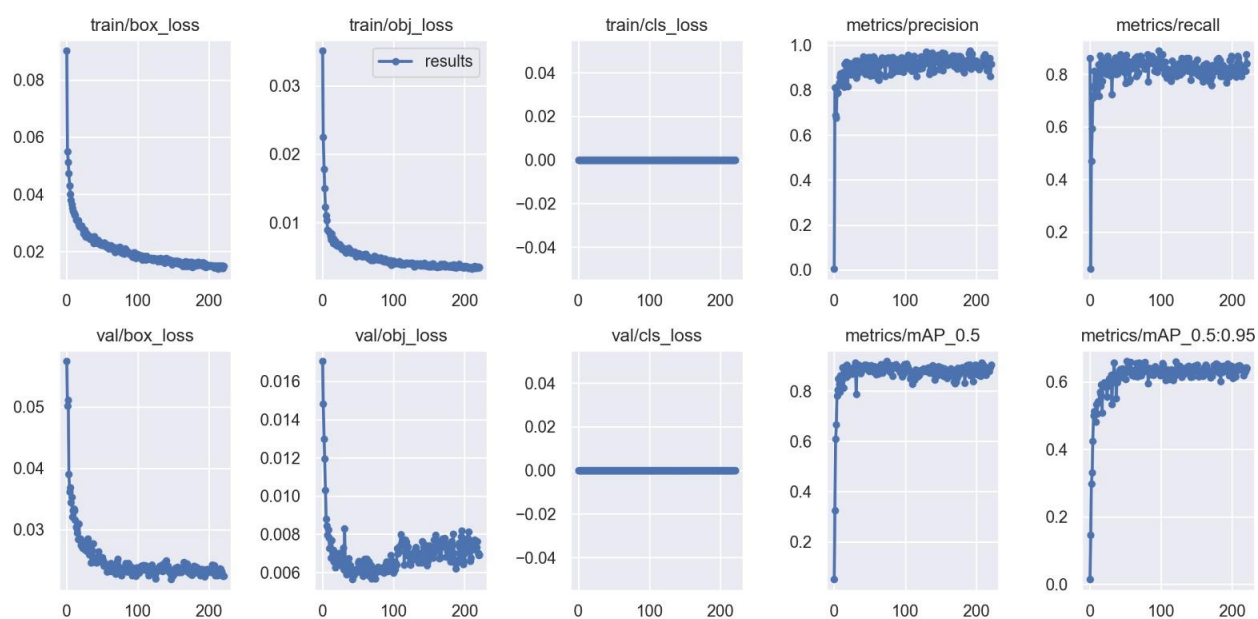


Figuur 63, preventie limitatie

5.3 Statistiek modellen

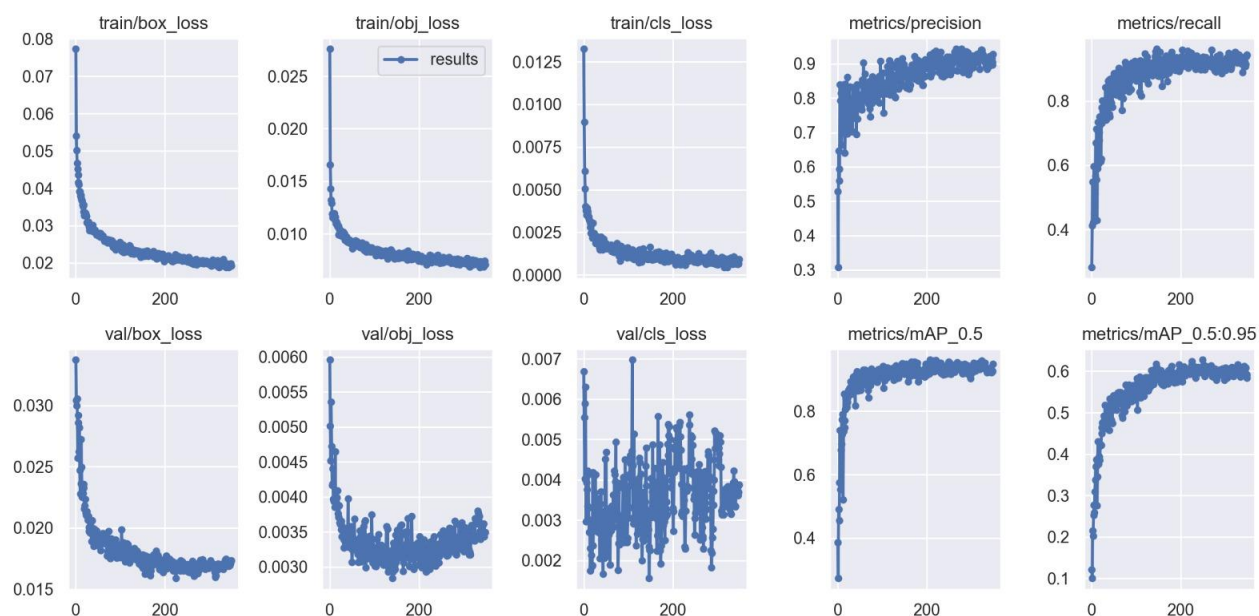
In dit onderdeel zijn verschillende grafieken te zien die het trainingsproces van een specifiek model van het begin tot einde weergeeft, zie figuur 64 en 65. Een van de belangrijkste statistieken hiervan zijn de mAP 0.5 en mAP 0.5:0.95. De gemiddelde eindwaardes hiervan zijn in meer detail weergegeven in de tabel in paragraaf 5.3.3, figuur 66.

5.3.1 Grafieken propaantanks



Figuur 64, trainingsproces propaantanks

5.3.2 Grafieken windturbines



Figuur 65, trainingsproces windturbines

5.3.3 Mean-average-precision per model

Model	mAP-0.5	mAP-0.5-0.95
Mean-max Windturbine	0.96181	0.62726
Windturbine	0.982	0.87
Windturbine_1	0.914	0.448
Mean-max Windturbine (aug)	0.96114	0.59415
Windturbine (aug)	0.982	0.849
Windturbine_1 (aug)	0.946	0.5
Mean-max Windturbine (Open data)	0.92447	0.58218
Mean-max Propaantanks	0.91852	0.66095

Figuur 66, mean-average-precision model

5.4 Terugkoppeling van het REV

5.4.1 Windturbines, studiegebied 1 (met confidence threshold)

De kaart in figuur 67 toont het resultaat van het uitvoering 2 voor windturbines over studiegebied 1 met de terugkoppeling naar het REV. Voor deze visualisatie is een *confidence threshold* van 0.5 ingesteld. De REV objecten zijn aangegeven met een rode driehoek. De gedetecteerde punten zijn aangegeven met de overige punten waar het kleurenschema de afstand ten opzichte van het dichtstbijzijnde REV object reflecteert.



Figuur 67, terugkoppeling naar het REV (windturbines studiegebied 1)

5.4.2 Windturbines, studiegebied 1 (zonder confidence threshold)

De kaart in figuur 68 toont het resultaat van het uitvoering 2 voor windturbines over studiegebied 1 met de terugkoppeling naar het REV. Voor deze visualisatie is geen *confidence threshold* ingesteld.

Windturbines gedetecteerd zonder confidence threshold | Studiegebied 1



Figuur 68, terugkoppeling naar het REV (windturbines studiegebied 1 zonder threshold)

5.4.3 Windturbines, studiegebied 2 (met confidence threshold)

De kaart in figuur 69 toont het resultaat van het uitvoering 2 voor windturbines over studiegebied 2 met de terugkoppeling naar het REV. Voor deze visualisatie is een *confidence threshold* van 0.6 ingesteld.



Figuur 69, windturbines naar het REV (windturbines studiegebied 2 met threshold)

5.4.4 Windturbines, studiegebied 2 (zonder confidence threshold)

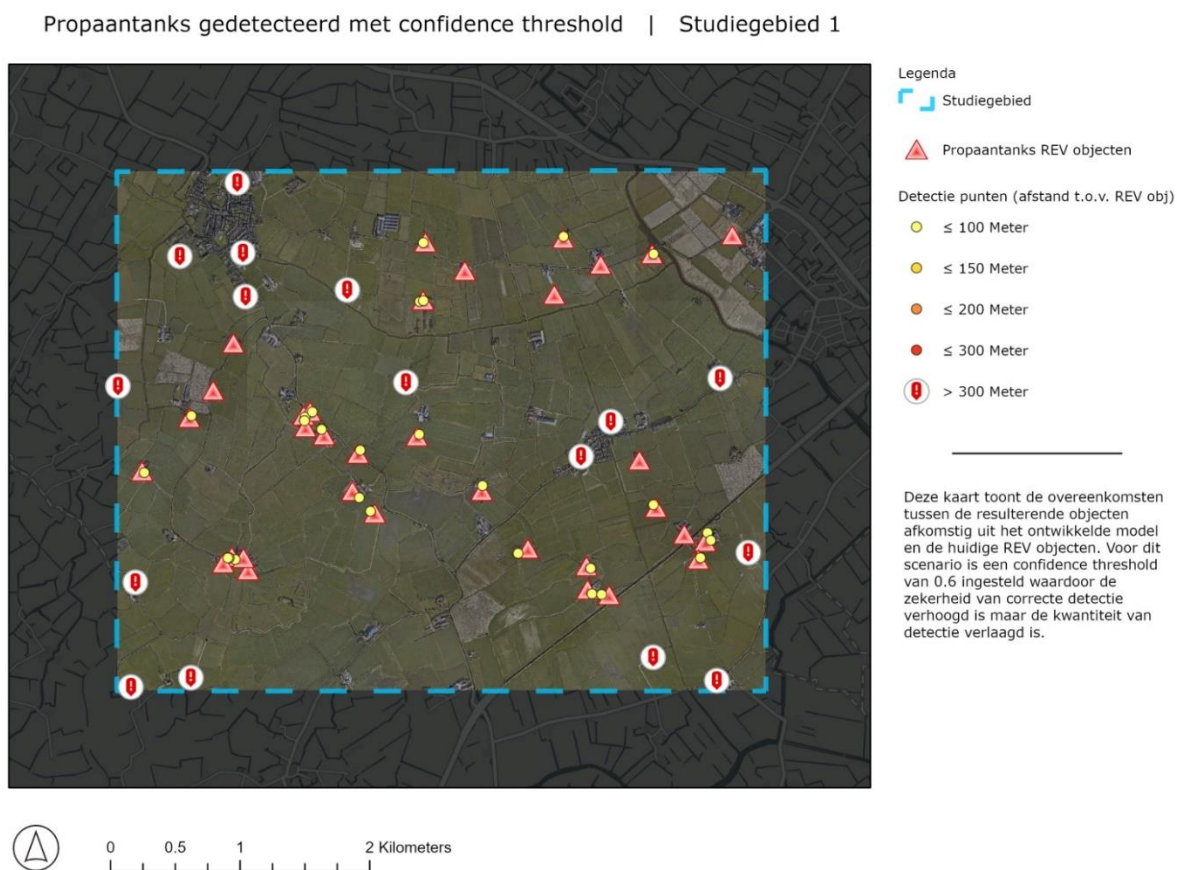
De kaart in figuur 70 toont het resultaat van het uitvoering 2 voor windturbines over studiegebied 2 met de terugkoppeling naar het REV. Voor deze visualisatie is geen *confidence threshold* ingesteld.



Figuur 70, terugkoppeling windturbines naar het REV (windturbines studiegebied 2 zonder threshold)

5.4.5 Propaantanks, studiegebied 1 (met confidence threshold)

De kaart in figuur 71 toont het resultaat van uitvoering 1 voor propaantanks over studiegebied 1 met de terugkoppeling van het REV. Voor deze visualisatie is een *confidence threshold* van 0.6 ingesteld.

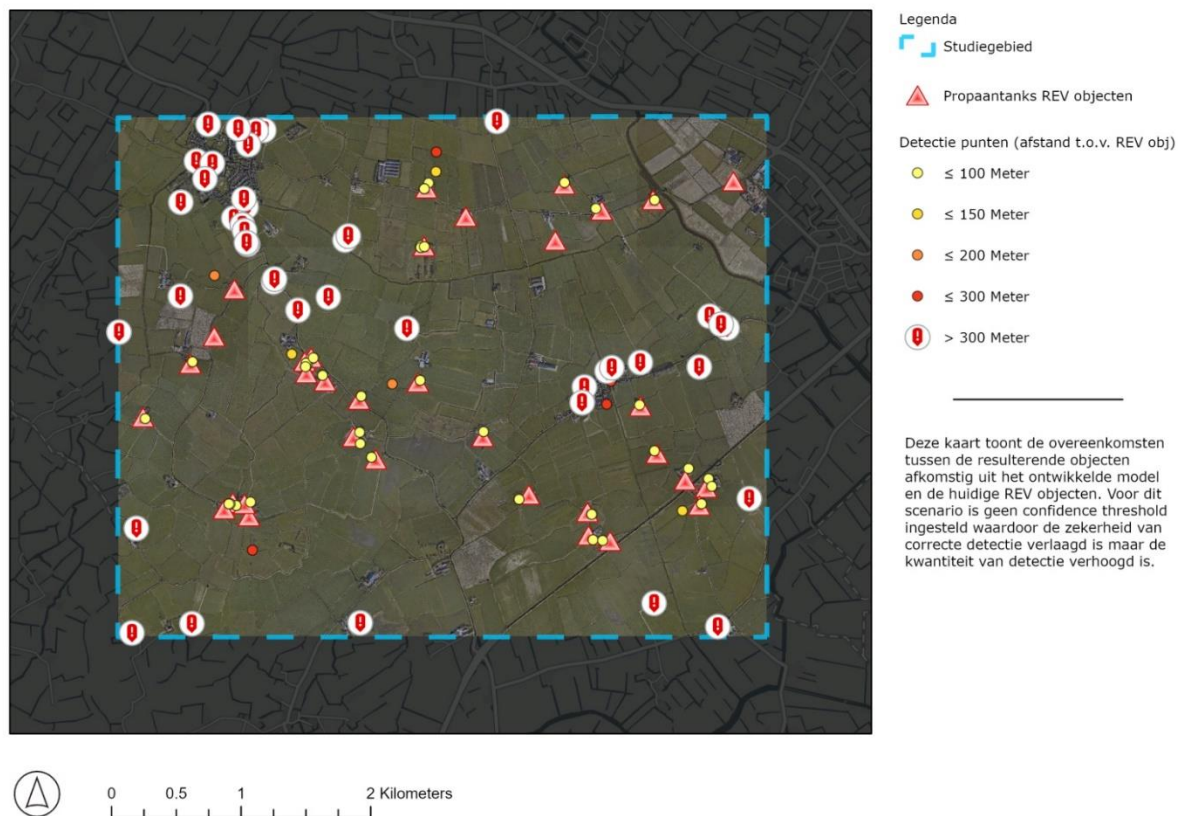


Figuur 71, terugkoppeling propaantanks naar het REV (propaantanks studiegebied 1 met threshold)

5.4.6 Propaantanks, studiegebied 1 (zonder confidence threshold)

De kaart in figuur 72 toont het resultaat van het uitvoering 1 voor propaantanks over studiegebied 1 met de terugkoppeling naar het REV. Voor deze visualisatie is geen *confidence threshold* ingesteld.

Propaantanks gedetecteerd zonder confidence threshold | Studiegebied 1

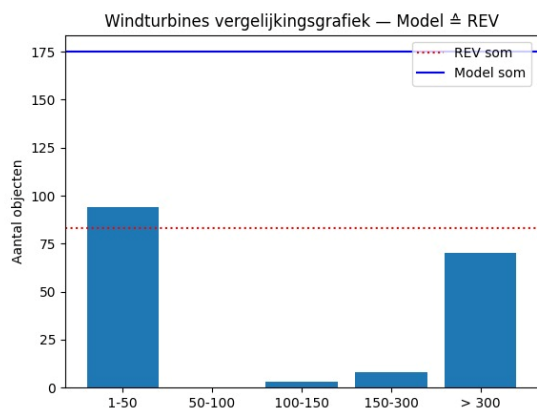


Figuur 72, terugkoppeling propaantanks naar het REV (propaantanks studiegebied 1 zonder threshold)

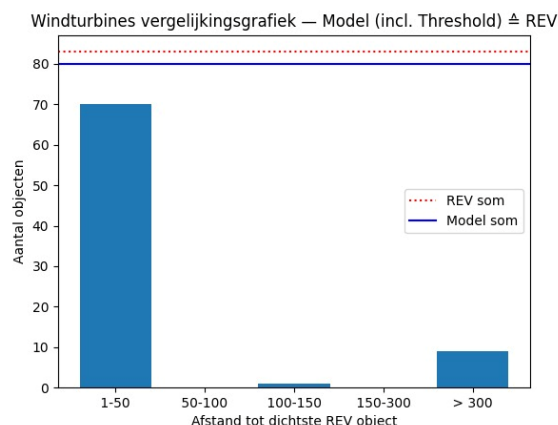
5.4.7 Vergelijkingstabellen windturbines

In figuur 73, 74, 75 en 76 zijn de resultaten weergegeven voor windturbines per studiegebied en threshold waarden. De x-as reflecteert de afstand van het gedetecteerde object ten opzichte van het dichtstbijzijnde REV object. Hiervoor zijn vijf categorieën gevormd beginnend bij 50 meter of minder tot en met 300 meter of hierboven met stappen van 50 daartussen. En de y-as reflecteert het aantal objecten gevonden per categorie.

Studiegebied 1

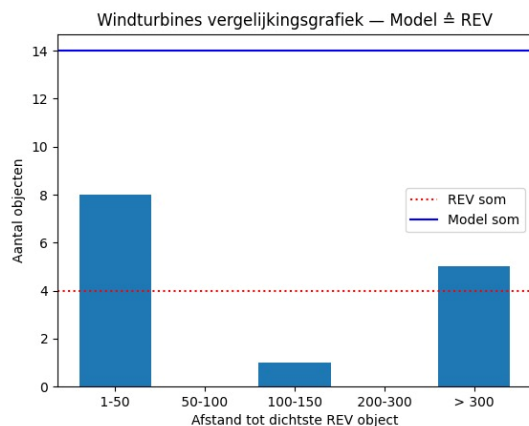


Figuur 73

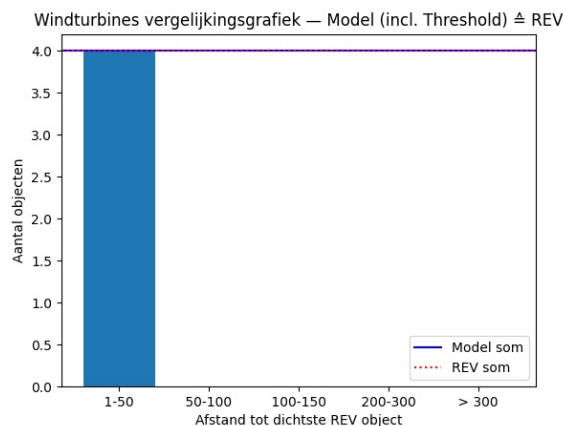


Figuur 74

Studiegebied 2



Figuur 75

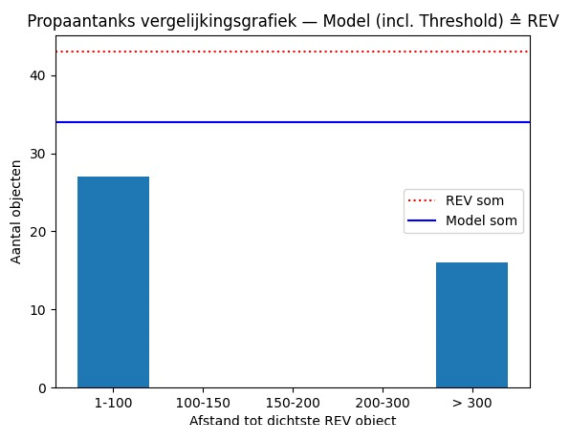


Figuur 76

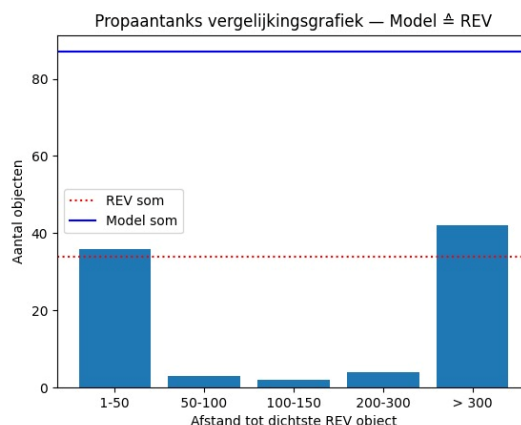
5.4.8 Vergelijkingstabellen propaantanks

In figuur 77 en 78 zijn de resultaten weergegeven voor propaantanks per studiegebied en threshold waarden. De x-as reflecteert de afstand van het gedetecteerde object ten opzichte van het dichtstbijzijnde REV object. Hiervoor zijn vijf categorieën gevormd beginnende bij 100 meter of minder tot en met 300 meter of hierboven met stappen van 50. En de y-as reflecteert het aantal objecten gevonden per categorie.

Studiegebied 1



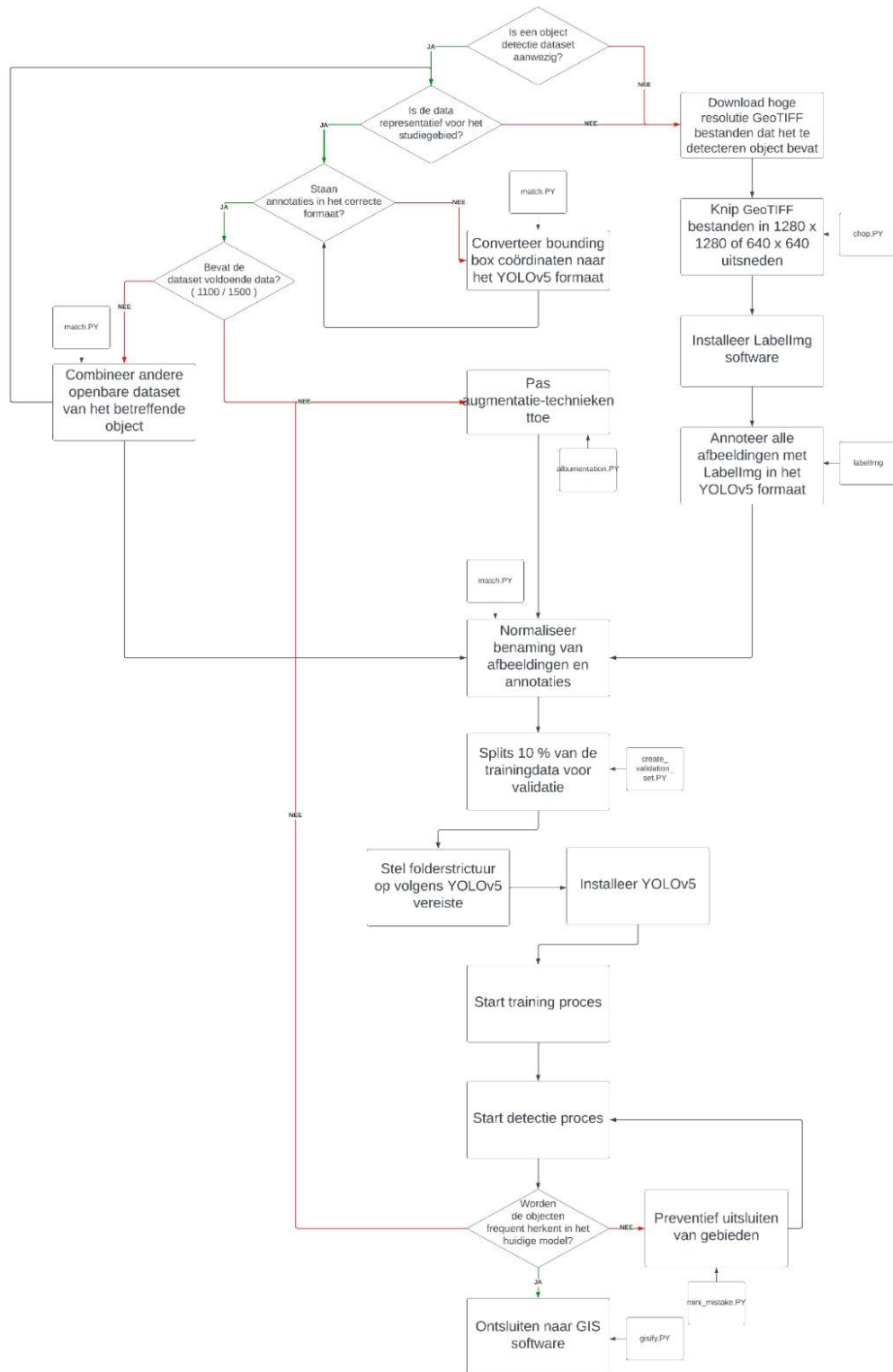
Figuur 77



Figuur 78

5.5 Workflow

Figuur 79 visualiseert de stappen uit de methoden waardoor dit onderzoek gereproduceerd kan worden.



Figuur 79, workflow

Conclusie:

Observaties van de resultaten en methodologie

06

Voor deze beroepsopdracht is onderzoek gedaan naar de vraag: 'In hoeverre is het mogelijk om een kwaliteitscontrole over de activiteiten binnen het REV efficiënter te maken en te optimaliseren om zo bij te dragen aan een verhoogde betrouwbaarheid en verbeterd inzicht in de dataflow voor het Ministerie van Infrastructuur en Waterstaat?'. Om deze onderzoeksvraag te beantwoorden, is er een onderzoek uitgevoerd dat gericht was op de toepassing van objectdetectie op de objecttypen windturbines en propaantanks binnen het REV.

Binnen het domein van AI is een objectdetectiemodel op basis van luchtfoto's een effectieve methode gebleken voor het detecteren van objecten in de openbare ruimte. Met name het YOLO(v5)-model levert nauwkeurige resultaten op. Het onderzoek heeft ook aangetoond dat verschillende beschikbare modellen andere kwaliteiten hebben, waardoor het optimale model mogelijk kan variëren afhankelijk van de situatie. Bovendien wijst de literatuur op beperkingen van objectdetectiemodellen op luchtfoto's met zeer lage resoluties of kleine objecten. Een ander aspect dat in het onderzoek is onderzocht, is de ethiek van gegevens en privacy. Hieruit blijkt dat luchtfoto's theoretisch gezien implicaties kunnen hebben voor de privacy. Echter, in de praktijk blijkt dat de ruimtelijke gegevens onvoldoende onderscheidende kenmerken bevatten om de privacy van een individu in gevaar te brengen.

De resultaten uit de studie hebben aangetoond dat het YOLOv5 model op basis van 8 centimeter luchtfoto's effectief is in het herkennen van grote windturbines. Zo geeft de mAP een score van 0.98 procent. De resultaten van kleine- en middelgrote windturbines (windturbine_1) tonen beduidend minder nauwkeurige resultaten. Ondanks dit toont het object een relatief hoge mAP score van 0.91 aan. Voor zowel studiegebied 1 en 2 was het model in staat alle grote windturbines te detecteren die overeenkwamen met het REV. De objecten die niet zijn gedetecteerd, zijn ofwel kleine tot middelgrote windturbines (windturbine_1) of windturbines die tijdens het maken van de luchtfoto's in opbouw waren. Het model toont overigens ook uitdagingen op het gebied van sportvelden. Deze foutieve resultaten zijn succesvol geminimaliseerd door de exclusie van deze gebieden vanuit de BGT zoals te zien is in 5.2.3.

YOLOv5 is ook effectief in het herkennen van de objecten propaantanks op basis van hoge resolutie luchtfoto's. Zo geeft de mAP een score van 0.91. Daarnaast heeft testgebied 1 objecten

gedetecteerd die niet in het REV staan. Hoewel de uitkomsten veelbelovende resultaten tonen zijn er ook uitdagingen op het gebied van bussen en aanhangers van vrachtwagens zoals aangetoond zijn in paragraaf 5.2.1. Deze foutieve resultaten zijn geminimaliseerd door de exclusie van parkeervlakken, regionale en lokale autowegen vanuit de BGT zoals afgebeeld is in 5.2.3.

Verder is uitgewezen dat de toepassing van data-augmentatie niet gegarandeerd tot hogere nauwkeurigheid leidt. In de eerste resultaten van studiegebied 1 voor windturbines zijn een aantal observaties gemaakt die de keuze heeft onderbouwt om augmentatietechnieken toe te passen. Ten eerste viel op dat het model moeite had met het detecteren van gedeeltelijk zichtbare objecten. Met name wanneer een wiek en kopstuk deels zichtbaar waren, gaf het model een minder hoge *confidence* waarde aan. Ten tweede viel op dat rechte, witte strepen als wiek gezien kunnen worden. Deze situaties resulteerde vaak in foutieve detectie. Tot slot presteerde het model minder goed bij een achtergrond waar in hoge frequentie kleinere witte objecten aanwezig waren. Reflecterend op deze informatie is de keuze gemaakt een tweede model te trainen, uitvoering 2 (aug), waarbij specifieke augmentatietechnieken zijn toegepast. De resultaten van het geaugmenteerde model tonen aan dat er sprake is van overfitting. Dit leidt tot een hogere frequentie van foutieve objectdetecties in tegenstelling tot het model zonder data-augmentatie gespecificeerd op windturbines. De resultaten voor beide modellen zijn gevisualiseerd in paragraaf 5.1.2 en 5.1.3. Het model zonder augmentatie gaf de beste resultaten en is daardoor voor het onderzoek verder toegepast. Deze uitvoering toont een hoge nauwkeurigheid ($mAP = 0.98$), mindere foutieve detectiefrequentie en bredere toepasbaarheid met betrekking tot verschillende achtergronden, zie paragraaf 5.4.7.

Uit het vergelijkende onderzoek tussen de verschillende getrainde modellen is gebleken de toepassing van openbare datasets kan leiden tot een verminderde nauwkeurigheid van het YOLOv5 object detectiemodel. Dit verschil is duidelijk zichtbaar in de afbeeldingen bij paragraaf 5.1. Ook tonen de resultaten in de Mean-average-precision tabel in paragraaf 5.3.3 aan dat deze toepassing een lagere mAP geeft dan andere getrainde uitvoeringen, namelijk een mAP score van 0.92. Een mAP van 0.92 is niet direct te interpreteren als overeenkomend met de werkelijkheid. Daarentegen liet dit model grote limitatie zien op verschillende achtergronden zoals water en weilanden in tegenstelling tot andere twee getrainde modellen voor windturbines.

De effectiviteit van de ontwikkelde methodologie is ook aangetoond door de resultaten uit de testgebieden. In testgebied 1 voor windturbines waar geen *confidence threshold* is ingesteld, zie paragraaf 5.4.2, zijn 175 windturbines gedetecteerd waar het REV slechts 83 objecten bevat. Voor hetzelfde testgebied met een *threshold* van 0.5, te zien in paragraaf 5.4.1, zijn 80 windturbines gedetecteerd waarvan 70 windturbines in een straal van vijftig meter verwijderd zijn van een REV object. Deze resultaten komen niet overeen met het aantal objecten geïnventariseerd in het REV. De 13 objecten die niet zijn gedetecteerd door twee redenen. Ten eerste bestond een deel van deze

windturbines uit kleine- en middelgrote windturbines. Zoals eerder in dit hoofdstuk is benoemt, is dit een van de gevonden limitaties van dit model. Ook waren een aantal van deze 13 objecten nog in aanbouw op het moment dat de luchtfoto's gemaakt werden zoals terug te zien in paragraaf 5.2.2. De fysieke representatie van windturbines zoals deze getraind zijn kwam hiermee niet overeen waardoor deze niet succesvol gedetecteerd zijn. Voor testgebied 2, waar geen *threshold* ingesteld is, hebben 14 objectdetecties plaatsgevonden waar het REV slechts 4 objecten aangeeft zoals is gevisualiseerd in paragraaf 5.4.4. Voor hetzelfde gebied met een *threshold* van 0.6 detecteerde het model overeenkomend met het REV 4 objecten, te zien in paragraaf 5.4.3. Ook voor propaantanks is de methodologie effectief gebleken. Voor testgebied 1 zijn zonder *confidence threshold* 87 exemplaren gedetecteerd zoals is weergegeven in paragraaf 5.4.6. Voor hetzelfde testgebied met een *threshold* van 0.6, te zien in paragraaf 5.4.5, detecteerde het model 43 propaantanks in tegenstelling tot de 34 objecten aanwezig binnen het REV. Het model heeft 16 potentiële propaantanks als aanvulling op het REV gedetecteerd. Van deze detecties zijn 2 objecten correct gevalideerd middels een Street view perspectief en 4 objecten gevalideerd middels een vogelvluchtperspectief. Dit betekent dat er ook met een *threshold* waarde van 0.6 nog steeds een aanzienlijke foutmarge plaatsvindt.

Uit dit onderzoek is gebleken dat het mogelijk is om de kwaliteitscontrole van objecttypen binnen het REV te optimaliseren om zo de betrouwbaarheid te verhogen. Dit is mogelijk door de implementatie van de methodologie die is ontwikkeld voor dit onderzoek voor de objecttypen van windturbines en propaantanks. De toepassing van innovatieve technologieën zoals AI staat hier centraal. Hierbij is het optimaal om het YOLOv5 objectdetectiemodel te implementeren op basis van hoge resolutie luchtfoto's. Hiermee is het mogelijk op geautomatiseerde wijze objecten in de openbare ruimte nauwkeurig te detecteren en classificeren. Ook maakt de implementatie van de methodologie de terugkoppeling met het REV mogelijk. Als gevolg hiervan kan de betrouwbaarheid van het REV worden gecontroleerd. De stappen voor reproductie van deze methode zijn gevisualiseerd in 5.5. Deze visualisatie is bedoeld als hulpmiddel om de methoden reproduceerbaar te maken zodat ook andere detecteerbare objecten binnen het REV gecontroleerd kunnen worden.

07

Validiteit aantonen

Voor dit onderzoek zijn propaantanks en windturbines uit het REV gecontroleerd op basis van de opgestelde methodologie. De trainingsdata is handmatig opgesteld voor beide objecttypen met voldoende diversiteit van landschappen, perspectieven en lichtformaties. Het uiteindelijke objectdetectiemodel is getest met een steekproef op gebieden die representatief zijn voor het Nederlandse landschap. Omdat de testgebieden niet overeenkomen met de geselecteerde trainingsgebieden is een potentiële bias in de data te verminderen. Op basis van deze observaties kan gesteld worden dat bij herhaling van dit onderzoek op dezelfde objecttypen, de resultaten overeen zouden komen, wat impliceert dat de resultaten van dit onderzoek valide zijn.

Resultaten interpreteren

De resultaten uit de studie hebben aangetoond dat het YOLOv5 model op basis van 8 centimeter luchtfoto's effectief is in het herkennen van de objecten. Dit resultaat is in overeenstemming met de verwachting, omdat uit de literatuur is gebleken dat het YOLOv5 model een relatief hoge nauwkeurigheid heeft. Deze hoge nauwkeurigheid in combinatie met hoge resolutie luchtfoto's waarin een hoog detailniveau aanwezig verklaard de nauwkeurigheid van de resultaten.

Windturbines

Voor grote windturbines kunnen de positieve resultaten ook worden verklaard doordat het een groot object is met unieke fysieke eigenschappen. Hierdoor is de kans op belemmeringen van omliggende objecten niet aanwezig. Daarnaast staan windturbines niet in stedelijke gebieden maar voornamelijk op open velden wat ertoe leidt dat er weinig ruis op de achtergrond aanwezig is. Tot slot zijn de objecten getraind op basis van handmatig opgestelde datasets die voldoende diversiteit en representativiteit van de omgeving bevatten. Ondanks de positieve resultaten zijn deze niet foutloos. Zo kunnen sportvelden en bepaalde daken die witte lijnen bevatten door het model foutief herkend worden als windturbine. Daarnaast worden afgesneden afbeeldingen waar windturbines slechts gedeeltelijk zichtbaar zijn gelimiteerd gedetecteerd.

De resultaten van kleine- en middelgrote windturbines (windturbine_1) tonen beduidend minder nauwkeurige resultaten. In tegenstelling tot grote windturbines zijn deze objecten kleiner van formaat en daardoor slecht zichtbaar vanuit een vogelvluchtperspectief. Dit in combinatie met de ruis van de omgeving op de achtergrond maakt het object gelimiteerd detecteerbaar. De belangrijkste verklaring waarom dit object minder goed presteert is het gebrek aan trainingsdata (N = 418) in tegenstelling tot de grote windturbines (N = 1401). Hierdoor heeft het model minder instanties en daarmee minder variatie van het object kunnen leren.

Datasets

De resultaten hebben aangetoond dat openbare datasets bij de toepassing van windturbines hebben geleid tot mindere resultaten. Voor de eerste versie van windturbines is een openbare dataset opgesteld met voldoende grootte en diversiteit. Een verklaring waarom de resultaten significant verbeterden met handmatige trainingsdata heeft te maken met representativiteit. Niet alleen de objecttypen zijn namelijk van belang maar ook de omgevingskenmerken, schaal, perspectief en lichtinval. De openbare dataset is afkomstig uit de Verenigde Staten waar het landschap verschilt van het Nederlandse landschap. Bepaalde gebieden zoals windturbines op zee waren bovendien niet/onvoldoende aanwezig. Daarnaast zijn de afbeeldingen getraind op een ander schaalniveau die niet representatief is voor de afbeeldingen van het testgebied. De handmatig opgestelde datasets (N = 1401) zijn afkomstig uit verschillende provincies in Nederland met diverse landschapskenmerken om te zorgen voor voldoende diversiteit. Dit is de reden dat de tweede versie van het model betere resultaten toont op verschillende testgebieden, ook op zee.

Data-augmentatie

Ondanks dat data-augmentatie positieve effecten kan hebben op de resultaten, is aangetoond dat de toepassing van data-augmentatie niet gegarandeerd tot hogere nauwkeurigheid leidt. Een verklaring waarom uitvoering 2 (aug) van windturbines minder goed presteert is het principe van overfitting. Met overfitting wordt bedoeld op het feit dat het model te afhankelijk wordt van trainingsdata en daardoor niet geschikt is om toe te passen op nieuwe situaties. Dit leidt ertoe dat het model minder goede prestaties oplevert. Augmentatie is geadviseerd toe te passen wanneer een model moeite heeft met het trainen van een object. In uitvoering 2 van windturbines is augmentatie toegepast op een goedwerkend model, hierdoor heeft overfitting waarschijnlijk plaats kunnen vinden. Hierbij moet wel worden benoemd dat data-augmentatie niet eenduidig is en kan bestaan uit verschillende methoden en technieken. Elke vorm van data-augmentatie kan zowel positieve als negatieve effecten op de nauwkeurigheid van het model hebben.

Propaantanks

De resultaten van propaantanks zijn ook zeer nauwkeurig. Ondanks de hoge nauwkeurigheid van resultaten zullen er altijd limitaties zijn met dit objecttype. De grootste limitatie is de zichtbaarheid van het object door belemmeringen zoals daken en bomen. Naast de directe belemmering kunnen omliggende objecten ook bijdragen aan een sterke schaduw waardoor het object niet duidelijk wordt belicht. Ondanks deze uitdagingen toont het YOLOv5 model hoogwaardige resultaten op dit kleine objecttype. Zelfs in het geval van gedeeltelijke belemmeringen zijn propaantanks gedetecteerd. Een verklaring hiervoor is de nauwkeurigheid van het YOLOv5 model op (kleine) objecten. Daarnaast is voor dit objecttype handmatig hoogwaardige trainingsdata (N = 1057) en validatiedata (N = 117) opgesteld met voldoende diversiteit en representativiteit. Ondanks de hoge nauwkeurigheid zijn er ook foutieve detecties. Dit komt voornamelijk voort uit bussen, aanhangers van vrachtwagens en schapen. Dit komt omdat deze op luchtfoto's relatief gelijkwaardige fysieke eigenschappen hebben.

Externe registers

Potentiële foutieve resultaten zijn geminimaliseerd door de toepassing van een extern register, namelijk de BGT. Het uitsluiten van bepaalde gebieden waar foutgevoelige objecten zich bevinden heeft geleid tot de optimalisatie van het model. In het geval van propaantanks zijn parkeervlakken, regionale en lokale autowegen preventief gefilterd om te voorkomen dat bepaalde objecttypen worden meegenomen in de analyse. Voor windturbines zijn de resultaten geoptimaliseerd door de uitsluiting van de gesloten verharding en kunststof vanuit de BGT. De toepassing van externe registers zoals het BGT kan de resultaten optimaliseren door de verminderde kans dat foutgevoelige objecten worden meegenomen in de analyse.

Beperkingen bespreken

Reproduceerbaarheid methodologie

De methodologie heeft de detectie van windturbines als basis gebruikt bij de uitwerking ervan. Het feit dat de methodologie vervolgens effectief toegepast is op propaantanks toont de reproduceerbaarheid van de methodologie aan. Het feit dat het model slechts is getest op twee objecttypen toont overigens de eerste beperking van het onderzoek aan. De reproduceerbaarheid kan niet worden gegarandeerd voor andere objecttypen omdat deze niet getest zijn. De kans dat de methodologie correct functioneert op concrete objecttypen met zichtbare fysieke eigenschappen is groot. De limitatie komt voornamelijk voort bij objecten die niet zichtbaar zijn op luchtfoto's doordat deze ondergronds of in een gebouw staan of te klein zijn.

Negatieve objecten

Een andere beperking van het onderzoek is het feit dat er geen negatieve detectie objecten zijn getraind. In plaats hiervan zijn geselecteerde gebieden preventief ontsloten. Voor het objecttype windturbines is het mogelijk dat daken of sportvelden worden getraind om ervoor te zorgen dat het model deze herkent als andere objecten. In het geval van propaantanks had dit gedaan kunnen worden voor objecten als auto's, vrachtwagens en schapen. Dit is mogelijk een verklaring dat bepaalde foutieve objecten zijn gedetecteerd.

Literatuurstudie

De keuze voor het YOLOv5 model is voortgekomen uit literatuurstudies gericht op de detectie van gelijkwaardige producten. De verschillende object detectiemodellen zijn dus niet opgesteld en naast elkaar getest op deze exacte datasets om de verschillende resultaten in te zien. Een limitatie hiervan is dat andere object detectiemodellen in specifieke gevallen kunnen leiden tot afwijkende resultaten. Daarnaast functioneert dit model positief op windturbines en propaantanks, maar is dit niet gegarandeerd voor andere objecttypen binnen het REV

Vervolgonderzoek

Op basis van de huidige resultaten van dit onderzoek kan worden gesteld dat de methodologie in combinatie met het objectdetectiemodel kan bijdragen aan het controleren van het REV. Dit kan daarmee bijdragen aan de betrouwbaarheid van het REV en daarmee aan een veilige leefomgeving. Om dit te bereiken wordt overigens wel aanbevolen om de methodologie op te schalen naar nationaal schaalniveau en naar andere objecttypen binnen het REV

Aanbevelingen en adviezen:

Praktische vervolgstappen en concrete adviezen

08

In dit hoofdstuk zijn de praktische vervolgstappen en concrete adviezen beschreven. De doorontwikkelingsmogelijkheden zijn onderverdeeld in drie segmenten. Om te beginnen zijn er concrete adviezen gericht op zowel de opschaling als de uitbreidingsmogelijkheden van het huidige model. Verder is er een overzicht gemaakt van overige aanbevelingen die betrekking hebben tot technische limitaties van het model. Technische randvoorwaarden met betrekking tot het project wordt toegelicht in onderdeel 7.3 Overige aspecten.

8.1 Concrete adviezen

8.1.1 Implementatie model & methodologie

De resultaten hebben aangetoond dat de methodologie in combinatie met het YOLOv5 detectiemodel op basis van 8 centimeter luchtfoto's effectief is bij het controleren van de objecttypen windturbines en propaantanks. Het model heeft aangetoond dat het zowel toegankelijk als nauwkeurig is, wat in lijn is met bevindingen uit de bestaande wetenschappelijke literatuur. Daarom wordt aanbevolen om de reproduceerbare methodologie, het objectdetectiemodel en de technische inzichten uit dit rapport toe te passen op het REV. Er zijn twee mogelijke benaderingen om dit toe te realiseren, namelijk: schaalvergroting naar nationaal niveau en uitbreiding naar andere objecttypen binnen het REV.

8.1.2 Opschaling

Dit onderzoek heeft op geselecteerde gebieden een controle uitgevoerd op de objecttypen windturbines en propaantanks. Deze objecttypen hebben een relatief hoge prioriteit en daarom wordt aangeraden het model hiervoor op te schalen naar nationaal schaalniveau. Het proces van opschaling is normaliter uitdagend door de grote hoeveelheid data, computerkracht en expertise die is vereist. De implementatie van de geautomatiseerde methodologie in combinatie met het toegankelijke object detectiemodel kan dit proces vereenvoudigen. Voor de onderzochte objecttypen in dit onderzoek zijn bepaalde processtappen al uitgevoerd zoals het verzamelen en voorbereiden van de data. Daarnaast is het huidige model getraind en zijn de resultaten gevalideerd op basis van (neutrale) testgebieden. De grootste uitdagingen binnen de opschaling liggen mogelijk op het gebied

van beschikbaarheid van voldoende technische infrastructuur, monitoring en het proces van verbetering. Naast het model en de methodologie is het belangrijkste element voor opschaling de aanwezigheid van luchtfoto's van veel Nederland.

8.1.3 Uitbreiding objecttypen

Voor dit project zijn de methodologie en het model toegepast op twee objecten. Naast de opschaling van deze objecttypen wordt aanbevolen om het model uit te breiden voor andere objecttypen in het REV. Ondanks dat een AI object detectiemodel mogelijkheden geeft tot het controleren van objecten zijn er ook limitaties. Het model gaat namelijk uit van fysieke eigenschappen. Wanneer objecttypen niet zichtbaar zijn vanuit een vogelvluchtperspectief doordat het bijvoorbeeld in een gebouw staan is dat een limitatie van de methodologie.

Het eerste onderdeel binnen de toepassing van de methodologie op andere objecttypen is het uitvoeren van een prioriteitenanalyse. Hierbij is het doel om een volgorde in prioriteiten te maken binnen de objecttypen op basis van urgentie en bruikbaarheid. Wanneer het totale aantal elementen van een objecttype namelijk zeer gering is kan het minder relevant zijn om de methodologie toe te passen. Zo staat voor de activiteit *Opslag-Brandbare-Vloeistof-Geen-Diesel-Ondergronds* bijvoorbeeld maar 1 object vastgelegd, zie bijlagen 1.2.4. Daarnaast is het belangrijk om de limitaties van objecten in acht te nemen. In de bijlagen zijn de elementen uit het REV geclassificeerd op basis van twee klassen.

De eerste klasse bestaat uit objecten die fysiek kenmerkbaar eigenschappen hebben. Voorbeelden hiervan zijn mestvergistingsinstallaties, opslag brandstof bunkerstations, seveso inrichtingen, enzovoort, zie bijlagen 1.1. Deze klasse zelf kan ook nog worden opgedeeld in meerdere onderdelen. Zo is een mestvergistingsinstallatie en concreet object net als windturbines en propaantanks. Dit in tegenstelling tot de gebouwen en tankstations die worden gekenmerkt door een gebied. Omdat deze objecten geen vaste fysieke eigenschappen hebben is het detecteren hiervan complexer.

In de tweede klasse staan objecten die technisch gezien zichtbaar zijn maar geen fysiek kenmerkende eigenschappen hebben. Zo worden elementen als vuurwerk, giftige baden en gasopslag voornamelijk geplaatst in warenhuizen, zie bijlagen 1.2. Vanuit technisch standpunt zijn deze objecten niet eenvoudig te detecteren door de homogeniteit van de fysieke eigenschappen. Ondanks deze beperkingen is het niet totaal onmogelijk deze objecten te detecteren. Het is hierbij wel van belang meer contextuele informatie bij te voegen. Dit is mogelijk door ruimtelijke patronen te combineren met vakinhoudelijke kennis om zo nieuwe inzichten te creëren. Ondanks dat dit soort objecten een technische verdieping nodig hebben om te realiseren kan het mogelijk zijn deze objecten te analyseren. Het is hierbij wel van belang realistisch om te gaan met de mogelijkheden.

Als het object geen fysieke kenmerken, contextuele eigenschappen of andere bruikbare informatie heeft kunnen deze objecten mogelijk niet geautomatiseerd worden gecontroleerd.

8.1.4 Windturbines en propaantanks

De adviezen in dit onderdeel hebben betrekking tot inzichten van windturbines en propaantanks. Ondanks dat deze criteria hiertoe behoren zijn deze ook toepasbaar en relevant voor andere objecttypen.

8.1.4.1 Threshold

Om een balans tussen nauwkeurigheid en correct gedetecteerde objecten te bewaren is een threshold van belang. Om een optimale threshold te selecteren wordt aangeraden om experimenten op te zetten waarin onderzoek wordt gedaan naar het percentage van correct gedetecteerd objecten. Het is hierbij van belang inzichten te krijgen in de statistieken van het model als ook naar vakinhoudelijke kennis. De *F1-confidence curve* kan ook als hulpmiddel gebruikt worden om de ideale threshold te bepalen, hoewel dit niet voor iedere situatie het gewenste resultaat op zal leveren.

8.1.4.2 Externe registers

Een belangrijk onderdeel bij het analyseren van objecten in de openbare ruimte is de toevoeging van externe registers. De contextuele informatie over objecten kunnen bijdragen aan het verminderen van benodigde computerkracht en de hoeveelheid handmatige controle die nodig is. Externe registers kunnen op verschillende momenten in het proces worden geïmplementeerd. Ten eerste is het mogelijk gebieden bijvoorbeeld preventief uit te sluiten. Dit zorgt ervoor dat minder gebieden geanalyseerd hoeven te worden. Het kan overigens ook na het analyseren van de data worden toegepast om het handmatige controleproces te vereenvoudigen. Net als bij thresholds vraagt de manier hoe externe registers worden toegepast om een genuanceerde aanpak. Bij preventieve uitsluiting van gebieden is er een mogelijkheid dat correcte objecten in die gebieden niet worden geanalyseerd. Anderzijds leidt het achteraf toevoegen van registers ook tot meer handmatig werk.

Er wordt aangeraden om externe registers toe te passen. Voor het huidige model is alleen de BGT gebruikt, maar dit kan per objecttype verschillen. Op basis van deze registers is het niet alleen mogelijk elementen uit te sluiten, maar ook ruimtelijke patronen in de data te ontdekken. Wanneer een object bijvoorbeeld niet voor kan komen in stedelijk gebied kan deze informatie worden gebruikt bij de uitzet en opschaling van het model.

8.1.4.3 Klassen binnen objecttypen

Een laatste belangrijke element is het analyseren van klassen binnen objecttypen. Ondanks dat windturbines eenduidige objecten lijken is dat het niet. Binnen windturbines zijn er bijvoorbeeld grote en kleine/middelgrote variaties. Deze klassen zijn fysiek sterk verschillend van elkaar en kunnen daardoor bijna worden gezien als twee objecttypen. Een gebrek aan kennis over deze klassen kan ertoe leiden dat er een disbalans in de trainingsdata is die bijdraagt aan verminderde resultaten. Er wordt daarom aangeraden om bij ieder objecttype onderzoek te doen naar de mogelijke variaties van objecten en de verschillende omgevingen waarin deze voorkomen. Deze inhoudelijke kennis kan gebruikt worden om een representatieve dataset op te stellen en het model daarmee te optimaliseren.

8.2 Overige aanbevelingen

8.2.1 Vergelijken object detectiemodellen

Het model is gekozen op basis van literatuurstudies gericht op de detectie van gelijkwaardige producten. De verschillende object detectiemodellen zijn dus niet opgesteld en naast elkaar getest op deze exacte datasets om de verschillende resultaten in te zien. Ondanks dat de resultaten nauwkeurig zijn is het mogelijk dat andere objectdetectiemodellen betere resultaten opleveren in nieuwe situaties. Een aanbeveling is daarom om in het geval dat het YOLOv5 model onnauwkeurige resultaten oplevert, terwijl de trainingsdata en parameters correct zijn geconfigureerd, een vergelijkingsstudie op te zetten. In deze studie kunnen meerdere objectdetectiemodellen worden geconfigureerd en worden getest op exact dezelfde situatie.

8.2.2 Data augmentatie

Data augmentatie kan een waardevolle toevoeging zijn aan het pre-processing onderdeel van de methodologie. Niet alleen kan data-augmentatie de hoeveelheid beschikbare afbeeldingen verveelvoudigen, ook draagt het bij aan een hogere diversiteit. Data-augmentatie kan daardoor bijdragen aan een effectiever model dat beter presteert op verschillende situaties. Dit is mogelijk zonder de arbeidsintensieve manier om meer trainingsdata op te stellen. Hierbij dient wel rekening te worden gehouden met het feit dat data-augmentatie niet automatisch leidt tot een optimaal functionerend model. Deze methode kan een toevoeging zijn op een trainingsdataset die relatief veel afbeeldingen bevat met voldoende diversiteit.

Met de methoden toegepast op het windturbine en propaantank model heeft de implementatie van data-augmentatie niet tot betere resultaten geleid. Data-augmentatie is overigens geen vaste

methode. Er zijn verschillende data-augmentatie technieken die tot andere resultaten kunnen leiden. De aanbeveling voor andere objecttypen is om een optimalisatieonderzoek op te zetten specifiek gericht op data-augmentatie. Op deze manier kan worden bepaald welke augmentatie methoden en parameters een positieve invloed kunnen hebben op de eindresultaten.

8.2.3 YOLOv5 OBB

De implementatie van YOLOv5 OBB (Oriented Bounding Box) kan een toevoeging zijn aan het model omdat objecten nauwkeurig kunnen worden gedefinieerd. Dit komt doordat objecten in meerdere oriëntaties kunnen worden geplaatst. De toevoeging van deze methode kan ertoe bijdragen dat complexere scenario's (clusters van kleinere objecten) ook nauwkeurige resultaten opleveren. Er wordt daarom aanbevolen om YOLOv5 OBB toe te passen wanneer objecten complexe vormen aannemen met verschillende oriëntaties.

8.2.4 Seizoensgebonden luchtfoto's

De resultaten van de propaantanks heeft aangetoond dat een bladerdak invloed heeft op de zichtbaarheid van dit objecttype. De staat van een bladerdak heeft overigens veel betrekking tot specifieke seizoenen. Om belemmering van kleine objecten te vermijden wordt aanbevolen om seizoensgebonden luchtfoto's toe te passen om zo de accuraatheid van het model te verbeteren.

8.3 Technische randvoorwaarden

Het literatuuronderzoek heeft een aantal limitaties van object detectiemodellen aangetoond. In dit onderdeel zullen deze worden toegelicht.

8.3.1 Datasets

De resultaten van windturbines hebben aangetoond dat een kwalitatief hoogwaardige dataset essentieel is bij het opstellen van een effectief objectdetectiemodel. Er wordt daarom aangeraden om grote diverse trainingsdatasets te gebruiken met voldoende representativiteit. Met grote datasets wordt bedoeld op een geschat minimum van 1500 afbeeldingen als trainingsdataset. Representativiteit en diversiteit is van belang om objecten herkennen in verschillende nieuwe situaties.

8.3.2 Resolutie en schaalvariatie

De kwaliteit en consistentie van de gebruikte dataset is ook belangrijk. In het geval van lage resolutie afbeeldingen is het mogelijk dat voornamelijk kleine objecten of bepaalde fysieke

eigenschappen niet zichtbaar zijn. Ook is consistentie van de resolutie belangrijk. De situaties die worden gedetecteerd dienen namelijk grotendeels overeen te komen met de trainingsdata voor optimale resultaten. Er wordt daarom aanbevolen gebruik te maken van hoge resolutie luchtfoto's waarin voldoende variëteit in de schaal van objecten aanwezig is.

8.3.3 Computerkracht

Het YOLOv5 object detectiemodel is gedeeltelijk gekozen door de toegankelijkheid van de infrastructuur en de snelheid waarmee het detecteert. Desondanks kan computerkracht een belangrijk component vormen bij de opschaling en uitbreiding van het huidige model. De beschikbaarheid van een betrouwbare technische architectuur wordt daarom aanbevolen te gebruiken om het model optimaal te laten presteren.

8.3.4 Handmatige controle

De resultaten tonen aan dat het model nauwkeurige resultaten biedt, maar ook dat het niet foutloos is. Ondanks dat de methodologie grotendeels geautomatiseerd is wordt daarom aanbevolen om ook een vorm van handmatige controle uit te voeren. Zo kan het zijn dat het model bij foutieve detectie onterechte meldingen geeft. Een handmatige controle kan een cruciale stap zijn in het controleren van de betrouwbaarheid van het REV en het verminderen van onrechtmatige resultaten.

8.3.5 Monitoringsperiode

De openbare ruimte is dynamisch en maakt daardoor veranderingen door. Het wordt daarom aangeraden een periode op te stellen waarin wordt gecontroleerd of objecttypen zijn veranderd. In het geval dat een objecttype veranderd van fysieke eigenschappen kan dit invloed hebben op de nauwkeurigheid van het model. Het inplannen van een periodieke controle waarin de effectiviteit van het model wordt beoordeeld is daarom essentieel om de kwaliteit te behouden.

8.3.6 Iteratieve verbeteringen

Ondanks dat een objectdetectiemodel statisch lijkt is dat het in de werkelijkheid niet. Het is altijd mogelijk nieuwe iteraties te maken die het huidige model verbeteren. Daarom wordt aangeraden de nauwkeurigheid van de resultaten te monitoren om nieuwe limitaties en mogelijkheden in te zien. Zo kan een objecttype over een langere periode veranderen waardoor deze niet meer worden gedetecteerd. Daarnaast is het mogelijk dat het objecttype in een gebied staat die sterk afwijkt van de data waarop het model is getraind. Deze regionale verschillen kunnen door middel van optimalisaties worden verbeterd.

8.3.7 Fysieke belemmeringen en lichtinval

De resultaten van het model op propaantanks heeft aangetoond dat fysieke belemmeringen een grote uitdaging kan zijn bij de toepassing van een object detectiemodel. Bij voornamelijk kleinere objecten is de kans groot dat een ander object het beeld belemmerd. Dit fenomeen in combinatie met een sterke lichtinval kan ertoe leiden dat bepaalde objecten niet zichtbaar zijn. Daarom wordt aanbevolen om luchtfoto's van momenten op te nemen waar zonlicht een minder grote invloed heeft. Voor belemmerende objecten zijn er vrijwel geen concrete oplossingen, maar het is belangrijk om hier rekening mee te houden bij het inschatten van de betrouwbaarheid van een model.

8.4 Overig

8.4.1 Data-ethiek & privacy

De bestaande wetenschappelijke literatuur heeft aangetoond dat de combinatie van een object detectiemodel met luchtfoto's niet direct leidt tot een inbreuk van privacy. Ondanks dat het risico gering is wordt aanbevolen om op een veilige en verantwoordelijke manier om te gaan met het model. Veelal komen de negatieve gevolgen van data niet uit de data zelf, maar uit de intenties achter de verwerking ervan. Bewust omgaan met de gedetecteerde afbeeldingen en inwinningstrategie is daarom een essentieel onderdeel van het datacontroleproces.

Bibliografie:

Overzicht wetenschappelijke literatuur en referenties

09

Albumentations AI. (z.d.). Frameworks and Libraries.

https://albumentations.ai/docs/frameworks_and_libraries. Geraadpleegd op 13 april 2023

applications. arXiv:2209.02976v1. 1-17. <https://doi.org/10.48550/arXiv.2209.02976>

-b Ultralytics. (2022). Ultralytics [Computer software]. GitHub.

<https://github.com/ultralytics/yolov5/discussions/4487>. Geraadpleegd op 11 april 2023

Bakirman, T. (2023). An Assessment of YOLO Architectures for Oil Tank Detection from SPOT

Bertino, E., Thuraisingham, B., Gertz, M., & Damiani, M. L. (2008). *Security and privacy for geospatial data*. <https://doi.org/10.1145/1503402.1503406>

-c Ultralytics. (2022). Ultralytics [Computer software]. GitHub.

<https://github.com/ultralytics/yolov5/discussions/4487>. Geraadpleegd op 11 april 2023

Coffer, M. M. (2020). Balancing Privacy Rights and the Production of High-Quality Satellite Imagery.

Environmental Science & Technology, 54(11), 6453–6455.

<https://doi.org/10.1021/acs.est.0c02365>

Coste, S. 2022. Crowd Analysis on Edge Devices: A Comparative Study of Neural

Dewi, C. et al., (2023). Deep Learning for Highly Accurate Hand Recognition Based on Yolov7

Diwan, T., Anirudh, G., & Tembhurne, J. V. (2022). Object detection using YOLO: challenges, architectural successors, datasets and applications. *Multimedia Tools and Applications*, 82(6), 9243–9275. <https://doi.org/10.1007/s11042-022-13644-y>

Dixit, K. G., Chadaga, M. G., & Savalgimath, S. S. (2019). Evaluation and Evolution of Object Detection

Techniques YOLO and R-CNN. *International Journal of Recent Technology and Engineering*, 8(2S3), 824–829. <https://doi.org/10.35940/ijrte.b1154.0782s319>

Dorlandt, M. (2022, 30 november). *SMART inwinnen van data in de openbare ruimte*. sharepoint.

Geraadpleegd op 2 maart 2023, van <https://haskennisplein->

my.sharepoint.com/personal/540274992_has_nl/_layouts/15/onedrive.aspx?id=%2Fpersonal%2F540274992%5Fhas%5FnI%2FDocuments%2FBijlagen%2FDor%2D2022%2D012547%20MinIW%20SMART%20inwinnen%20van%20data%5Fdef%2Epdf&parent=%2Fpersonal%2F540274992%5Fhas%5FnI%2FDocuments%2FBijlagen

Du, J. (2018). Understanding of Object Detection Based on CNN Family and YOLO. *Journal of physics, 1004*, 012029. <https://doi.org/10.1088/1742-6596/1004/1/012029>

for Side Scan Sonar with Embedded GPU. *Marine Science and Engineering*. 2-20.

Hagendorff, T. (2019). From privacy to anti-discrimination in times of machine learning. *Ethics and Information Technology, 21*(4), 331–343. <https://doi.org/10.1007/s10676-019-09510-5>
<https://doi.org/10.3390/app13010649>
<https://doi.org/10.3390/jmse11030487>

IBM. (2022). AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What's the Difference? *IBM*. <https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks>

IBM. (2022). AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What's the Difference? *IBM*. <https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks>

IBM. (z.d.). *What is Computer Vision?* Geraadpleegd op 10 juni 2023, van

<https://www.ibm.com/topics/computer-vision>

Imagery, *International Journal of Environment and Geoinformatics (IJEGEO)*, 10(1):009-000. Doi.10.30897/ijegeo.1196817

Jain, A., Ramaprasad, R., Narang, P., Mandal, M., Chamola, V., Yu, F. R., & Guizan, M. (2021). AI-Enabled Object Detection in UAVs: Challenges, Design Choices, and Research Directions.

IEEE Network, 35(4), 129–135. <https://doi.org/10.1109/mnet.011.2000643>

Jain, A., Ramaprasad, R., Narang, P., Mandal, M., Chamola, V., Yu, F. R., & Guizan, M. (2021). AI-Enabled Object Detection in UAVs: Challenges, Design Choices, and Research Directions.

IEEE Network, 35(4), 129–135. <https://doi.org/10.1109/mnet.011.2000643>

Janiesch, C., Zschech, P., & Heinrich, K. (2021). Machine learning and deep learning. *Electronic Markets, 31*(3), 685–695. <https://doi.org/10.1007/s12525-021-00475-2>

- Jiang, P. et al., (2022). A Review of Yolo Algorithm Development. *Procedia Computer Science*, vol. 199. 1066-1073. <https://doi.org/10.1016/j.procs.2022.01.135>
- Joiya, F. (2022). OBJECT DETECTION: YOLO VS FASTER R-CNN. *International Research Journal of Modernization in Engineering Technology and Science*. <https://doi.org/10.56726/irjmets30226>
- Kim, J., Sung, J., & Park, S. (2020, 1 november). *Comparison of Faster-RCNN, YOLO, and SSD for Real-Time Vehicle Type Recognition*. IEEE Conference Publication | IEEE Xplore.
Geraadpleegd op 21 juni 2023, van <https://ieeexplore.ieee.org/document/9277040>
- Koech, K. E. (2023, 7 juni). Object Detection Metrics With Worked Example - Towards Data Science. *Medium*. <https://towardsdatascience.com/on-object-detection-metrics-with-worked-example-216f173ed31e>
- Li, C. et al., (2022). YOLOv6: A Single-Stage Object Detection Framework for Industrial
- Li, Z. et al., (2023). Sparsity Regularization-Based Real-Time Target Recognition
- Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., & Pietikäinen, M. (2020). Deep Learning for Generic Object Detection: A Survey. *International Journal of Computer Vision*, 128(2), 261–318. <https://doi.org/10.1007/s11263-019-01247-4>
- Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., & Pietikäinen, M. (2020). Deep Learning for Generic Object Detection: A Survey. *International Journal of Computer Vision*, 128(2), 261–318. <https://doi.org/10.1007/s11263-019-01247-4>
- Meituan team. (2022). YOLOv6 概述. yolov6-docs.readthedocs.io. https://yolov6-docs.readthedocs.io/zh_CN/latest/YOLOv6%E6%A6%82%E8%BF%B0.html. Geraadpleegd op 19 april 2023
- Ministerie van Infrastructuur en Waterstaat. (z.d.-a). *Over het Register Externe Veiligheidsrisico's (REV)*. [registerexterneveiligheid.nl](https://www.registerexterneveiligheid.nl). Geraadpleegd op 2 maart 2023, van <https://www.registerexterneveiligheid.nl/overrev>
- Ministerie van Infrastructuur en Waterstaat. (z.d.-b). *Register Externe Veiligheidsrisico's | Register externe veiligheid*. [registerexterneveiligheid.nl](https://www.registerexterneveiligheid.nl). Geraadpleegd op 2 maart 2023, van <https://www.registerexterneveiligheid.nl/>

- Model. Big Data Cognitive Computing. 2-16. <https://doi.org/10.3390/bdcc7010053>
- Namdev, U., Agrawal, S., & Pandey, R. (2022). Object Detection Techniques based on Deep Learning: A Review. *Computer science and engineering : an international journal*, 12(1), 125–134. <https://doi.org/10.5121/cseij.2022.12113>
- Networks on Blurred Images. University of Twente. 1-6.<https://purl.utwente.nl/essays/94360>
- Palla, T. S.; Chethireddy, M. (2023). A comparison of Advanced deep learning algorithms for multi-digit recognition in historical documents. Department of Computer Science Engineering. 1-51. DiVA id: diva2:1740872
- Paullada, A., Raji, I. D., Bender, E. M., Denton, E., & Hanna, A. (2021). Data and its (dis)contents: A survey of dataset development and use in machine learning research. *Patterns*, 2(11), 100336. <https://doi.org/10.1016/j.patter.2021.100336>
- Pytorch. (z.d.). PYTORCH MOBILE. <https://pytorch.org/mobile/ios/#pytorch-ios-tutorial-and-recipes>. Geraadpleegd op 13 april 2023
- Sabina, N., Aneesa, M. P., & Haseena, P. V. (2022, 14 juni). *Object Detection using YOLO And Mobilenet SSD: A Comparative Study*. www.ijert.org. Geraadpleegd op 21 juni 2023, van <https://www.ijert.org/research/object-detection-using-yolo-and-mobilenet-ssd-a-comparative-study-IJERTV11IS060065.pdf>
- Safdar, N. M., Banja, J. D., & Meltzer, C. C. (2020). Ethical considerations in artificial intelligence. *European Journal of Radiology*, 122, 108768. <https://doi.org/10.1016/j.ejrad.2019.108768>
- Sánchez, S. F., Romero, H. J., & Morales, A. D. (2020). A review: Comparison of performance metrics of pretrained models for object detection using the TensorFlow framework. *IOP Conference Series: Materials Science and Engineering*, 844, 012024. <https://doi.org/10.1088/1757-899x/844/1/012024>
- Santos, C. (2019). Satellite Imagery, Very High-Resolution and Processing-Intensive Image Analysis: Potential Risks Under the GDPR. *Air and Space Law*, 44(Issue 3), 275–295. <https://doi.org/10.54648/aila2019018>

- Sarker, I. H. (2022). AI-Based Modeling: Techniques, Applications and Research Issues Towards Automation, Intelligent and Smart Systems. *ResearchGate*, 3(2).
<https://doi.org/10.1007/s42979-022-01043-x>
- Satellite Imaging Corp. (z.d.). *Computer Vision | Satellite Imaging Corp.* Geraadpleegd op 21 juni 2023, van <https://www.satimagingcorp.com/applications/artificial-intelligence-ai/computer-vision/>
- Srivastava, S., Divekar, A. A., Anilkumar, C., Naik, I., Kulkarni, V. P., & Pattabiraman, V. (2021). Comparative analysis of deep learning image detection algorithms. *Journal of Big Data*, 8(1).
<https://doi.org/10.1186/s40537-021-00434-w>
- Sultana, F., Sufian, A., & Dutta, P. (2020). A Review of Object Detection Models Based on Convolutional Neural Network. *Advances in intelligent systems and computing*, 1–16.
https://doi.org/10.1007/978-981-15-4288-6_1
- Terven, R. J.; Cordova-Esparaza, M. D. (2023). A COMPREHENSIVE REVIEW OF YOLO: FROM YOLOV1 TO YOLOV8 AND BEYOND. arXiv:2304.00501v1. 2-27.
<https://doi.org/10.48550/arXiv.2304.00501>
- Thuan, Do. 2021. EVOLUTION OF YOLO ALGORITHM AND YOLOV5: THE STATE-OF-THE-ART OBJECT DETECTION ALGORITHM. Oulu University of Applied Sciences. 3-16. Corpus ID: 235341555
- Tian, Z.; Huang, J.; Yang, Y.; Nie, W. KCFS-YOLOv5: A High-Precision Detection Method for Object Detection in Aerial Remote Sensing Images. *Appl. Sci.* 2023, 13, 649.
- Ultralytics. (2023). Ultralytics [Computer software]. GitHub.
<https://github.com/ultralytics/ultralytics>. Geraadpleegd op 11 april 2023
- Wan, D.; Lu, R.; Wang, S.; Shen, S.; Xu, T.; Lang, X. (2023). YOLO-HR: Improved YOLOv5 for Object Detection in High-Resolution Optical Remote Sensing Images. *Remote Sensing*. 2-17.
<https://doi.org/10.3390/rs15030614>
- Wang, C. et al., (2022). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. arXiv:2207.02696. 1-15. <https://doi.org/10.48550/arXiv.2207.02696>
- Wilczek, F. (2022). *Fundamentals: Ten Keys to Reality*.

Wong, K.Y. (2021). YOLOv7 [Computer software]. GitHub. <https://github.com/WongKinYiu/yolov7>.

Geraadpleegd op 11 april 2023

Yu, J., Guo, K., Hu, Y., Ning, X., Qiu, J., Mao, H., & Yao, S. (2018, 1 maart). *Real-time object detection towards high power efficiency*. IEEE Conference Publication | IEEE Xplore.

Geraadpleegd op 21 juni 2023, van <https://ieeexplore.ieee.org/document/8342100>

Yu, Y., & Yang, R. (2021, 9 maart). *Artificial Convolutional Neural Network in Object Detection and Semantic Segmentation for Medical Imaging Analysis*. Frontiers. Geraadpleegd op 21 juni 2023, van <https://www.frontiersin.org/articles/10.3389/fonc.2021.638182/full>

Yusro, M. M. (2022). Comparison of Faster R-CNN and YOLOv5 for Overlapping Objects Recognition. *Baghdad Science Journal*. <https://doi.org/10.21123/bsj.2022.7243>

Zhang, X., Zhang, L., Wang, X., & Shum, H. (2012). Finding Celebrities in Billions of Web Images. *IEEE Transactions on Multimedia*, 14(4), 995–1007.

<https://doi.org/10.1109/tmm.2012.2186121>

Zhang, Y. et al., (2022). dawi Mathematical Problems in Engineering Volume 2022, Article ID 1272896, 1-15. <https://doi.org/10.1155/2022/1272896>

Zhao, Z., Zheng, P., Xu, S., & Wu, X. (2019, 16 april). *Object Detection with Deep Learning: A Review*. Geraadpleegd op 21 juni 2023, van <https://arxiv.org/pdf/1807.05511.pdf>

Zou, Z., Chen, K., & Shi, Z. (2023, 18 januari). *Object Detection in 20 Years: A Survey*.

Geraadpleegd op 21 juni 2023, van <https://arxiv.org/pdf/1905.05055.pdf>

A.I chop.PY

```

1 import subprocess
2 import os
3 def gdalsubprocess( path, file, outp, outn, padding, startx, starty ):
4
5     cmd = f'C:\\OSGeo4W\\bin\\gdal_translate -srcwin {startx} {starty} {padding} {padding} {path}\\{file} {outp}\\{outn}'
6     print(cmd)
7
8     subprocess.run(cmd,shell=True)
9
10 path=r"C:\Users\Rik\Downloads\wind_turbi_test_data"
11 outp = r"C:\Users\Rik\Downloads\wind_turbi_test_res"
12
13 for file in os.listdir(path):
14     startx=0
15     starty=0
16     padding = 1280
17     starty=0
18     for i in range(80):
19         outn = f"{file[:-4]}patch_{str(i)+}.tif"
20
21         if startx+1280 > 12500:
22             starty += 1280
23             startx = 0
24
25         gdalsubprocess(path, file, outp, outn, padding, startx, starty)
26         startx+=1280
27

```

Regels

Beschrijving

- | | | |
|-------|-------|---|
| 1t/m2 | | Importeert vereiste modules; subprocess en os. |
| 3 | | Creëert een functie dat zeven parameters vereist. De eerste parameter is de locatie van de folder waarin de afbeeldingen staan welke geknipt dienen te worden. De tweede parameter neemt als input de bestandsnaam dat geknipt dient te worden. De volgende twee parameters bevatten de folder waarin de nieuwe bestanden kunnen worden opgeslagen en het bestandsnaam dat geknipt dient te worden. Vervolgens verwacht de parameter padding het formaat wat geknipt zal worden. Tot slot vult de gebruiker bij startx en starty de beginnende pixel vanwaar de uitsneden gemaakt kan worden. |
| 5t/m8 | | Zorgt ervoor dat de prompt samengesteld met de parameters benoemt bij regel 3 uitgevoerd wordt in de OSGeo4W shell. |
| 10 | | Definieert de eerste parameter voor de functie genoemd bij regel 3. Namelijk de locatie van de folder waarin de afbeeldingen staan welke geknipt dienen te worden. |
| 11 | | Definieert de derde parameter voor de functie genoemd bij regel 3. Namelijk de locatie van de folder waarin de afbeeldingen nadat deze geknipt zijn dienen te worden opgeslagen. |

Regels	Beschrijving
13	De eerste for loop; deze for loop itereert over alle bestanden in de folder welke aangegeven is bij regel 11.
14t/m17	In deze regels worden de variabelen startx en starty eerder benoemt in regel 3 aangemaakt zodat deze gebruikt kunnen worden als parameters voor de functie genoemd in regel 3. Ook wordt de variabelen padding aangemaakt en gedefinieerd met een integer van 1280.
18	De tweede for loop; deze for loop itereert over het aantal delen dat uit het bestand geknipt kan worden. Dit aantal is berekent door de totale lengte van de x-as en y-as van het originele bestand te delen door de padding en deze afgerond met elkaar te vermenigvuldigen.
19	De output naam van het geknipte deel wordt in deze regel gedefinieerd. Deze naam krijgt als unieke. identificatie de naam van het originele bestand en een nummering verwijzend naar de iteratie.
20t/m23	Wanneer startx + 1280 groter is dan de totale lengte van de x-as van het originele bestand zal startx gereset worden naar de startpositie 0. Dit betekent namelijk dat het einde van de eerste rij, van links naar rechts, bereikt is. De starty wordt daarentegen opgeteld met 1280 zodat de volgende iteratie een rij naar onder schuift.
25	Initieert de functie gdalsubprocess() met de parameters beschreven in regel 3 en gedefinieerd in regels regel 10, regels 14 t/m 17, regel 19 en regels 20 t/m 23.
26	Tot slot wordt startx opgeteld met 1280 zodat de startpositie voor de volgende knip 1280 pixels naar rechts is geschoven.

B.I tile_extractor.PY (1/2)

```
1 # first use the following command within the OSGeo4W Shell to make a vrt raster from the 4 terabytes of imagery:
2 '''C:\OSGeo4W>gdalbuildvrt E:\stacked_raster.vrt E:\turbi_SAT\*.tif'''
```

Regels

Beschrijving

1t/m2 De commando tussen  dient ingevoerd te worden in de OSGeo4W shell met als eerste parameters de output locatie en vrt bestandsnaam en als tweede parameter de folder waarin de GeoTIFF bestand en zich bevinden met daarachter *.tif.

B.II tile_extractor.PY (2/2)

```
3
4 from haishoku.haishoku import Haishoku
5 import arcpy, os
6 from arcpy.ia import *
7 import subprocess
8
9 AOI = r"E:\turbi_dataset\detecteer_data\turbi_laag.shp"
10 spatial_ref_poly = arcpy.Describe(AOI).spatialReference
11
12 def main():
13     arcpy.env.overwriteOutput = True
14     nid = 0
15     # inputs
16     fc_in = AOI
17     width = 51.2 # with of the output rectangle
18     height = 51.2 # height of the output rectangle
19
20     # output
21     fc_out = r'C:\Users\Rik\Documents\python_script\pytorch_custom_code\arcpy_workspace\object_in_tile_check\object_in_tile_check.gdb\rectangles_prod'
22
23     # create empty output featureclass
24     sr = arcpy.Describe(fc_in).spatialReference
25     out_ws, out_name = os.path.split(fc_out)
26     arcpy.CreateFeatureclass_management(out_ws, out_name, "POLYGON", fc_in, "DISABLED", "DISABLED", sr)
27
28     # make fields list
29     flds_use = correctFieldList(arcpy.ListFields(fc_in))
30
31     # create insert cursor
32     with arcpy.da.InsertCursor(fc_out, flds_use) as curs_out:
33
34         # loop through input features
35         with arcpy.da.SearchCursor(fc_in, flds_use) as curs_in:
36             for row_in in curs_in:
37                 nid+=1
38                 polygon = createRectangleFromPoint(row_in[0].firstPoint, width, height, sr)
39                 row_out = buildRow(row_in, flds_use, polygon)
40                 curs_out.insertRow(row_out)
41                 arcpy.management.CopyFeatures(polygon, rf"C:\Users\Rik\Documents\python_script\pytorch_custom_code\arcpy_workspace\minipolys\{str(nid)}polypart.shp")
42                 poly = rf"C:\Users\Rik\Documents\python_script\pytorch_custom_code\arcpy_workspace\minipolys\{str(nid)}polypart.shp"
43                 gdalsubprocess( poly , nid )
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59 def createRectangleFromPoint(pnt, width, height, sr):
60     arrPnts = arcpy.Array()
61
62     pnt2 = arcpy.Point(pnt.X - width, pnt.Y - height)
63     arrPnts.add(pnt2)
64     pnt2 = arcpy.Point(pnt.X - width, pnt.Y + height)
65     arrPnts.add(pnt2)
66     pnt2 = arcpy.Point(pnt.X + width, pnt.Y + height)
67     arrPnts.add(pnt2)
68     pnt2 = arcpy.Point(pnt.X + width, pnt.Y - height)
69     arrPnts.add(pnt2)
70
71     return arcpy.Polygon(arrPnts, sr)
72
73
74 def correctFieldList(flds):
75     flds_use = ['Shape@']
76
77     fldtypes_not = ['Geometry', 'Guid', 'OID']
78     for fld in flds:
79         if not fld.type in fldtypes_not:
80             flds_use.append(fld.name)
81     return flds_use
82
83
84 def buildRow(row_in, flds_use, polygon):
85     # try:
86     lst_in = list(row_in)
87     lst_in[0] = polygon
88
89     return tuple(lst_in)
```

Regels

Beschrijving

4t/m7		Importeert vereiste modules; haichoku, os, arcpy, arcpy.ai en subprocess.
9		Creëer de variabel AOI (area of intrest) en geeft deze variabelen een shapefile met punt geometrieën.
10		Extraheert de ruimtelijke informatie (spatial reference) als object op in een variabelen.

Regels	Beschrijving
12	Creëert de functie main(). Deze functie wordt o.a. aan het eind gebruikt om alle andere functie waar nodig ook mee uit te voeren.
13	Stelt de arcpy.env.overwrite variabelen in op True.
14	Creëert een variabelen (nid) dat in een later stadium van dit script zal dienen als unieke identificator.
17t/m18	Creëert twee variabelen waarin de breedte en hoogte in meter van een 1280 x 1280 vierkant / 2 bij een pixel resolutie van 8 x 8 cm. De volgende berekening geven de breedte en hoogte waarde $51.2 = ((1280*0.8)/10)/2$.
21	Creëert een variabelen waar een ESRI feature class in een geodatabase opgeslagen kan.
25	Splitst de ESRI geodatabase locatie en de feature class naam en slaat deze op in aparte variabelen.
26	Creëert de feature class met de punten van de shapefile genoemd in regel 9 met de naam en locatie gesplitst in regel 25 en de spatial reference opgeslagen in regel 10 gebruikmakend van de CreateFeautreclass_management() functie.
29	Maakt een variabelen met daarin de kolommen van de feature class.
32	Creëert een object dat zal dienen als de manier van inlezen, een zogenoemde cursor.
35t/m36	Itereert met de cursor uit regel 32 door alle items in de feature class, dit betekent alle punt geometrieën.
37	De variabelen nid telt iedere iteratie op.
38	Roept de functie CreateRectangleFromPoint() aan. Deze functie is later in dit script geschreven. Deze functie vereist als parameters vier elementen. Ten eerste een punt geometrie, ten tweede een breedte waarde, ten derde een hoogte waarde en tot slot een spatial reference. De vereiste parameters zijn in de afgelopen regels verzameld en worden in deze regel gegeven als input voor de functie. De CreateRectangleFromPoint() creëert een vierhoekige polygoon object. Dit polygoon object wordt in deze regel opgeslagen als variabele polygoon.
41	Met het polygoon object wordt in deze regel een shapefile copy gemaakt.
43	Roept de functie gdalsubprocess() aan. Deze functie is later in dit script geschreven. Deze functie vereist twee parameter. Ten eerste een shapefile van een polygoon en ten tweede een unieke identificatie. Deze twee parameters zijn gegeven.
45	Creëert de CreateRectangleFromPoint() functie.
46	Maakt een leeg arcpy.array welke gebruikt zal worden om tijdelijke punt objecten in om te slaan
48t/m55	Berekent tijdelijke punt objecten door de x-coördinaat – en + (breedte/2), y-coördinaat – en + (hoogte/2). Met de juiste combinatie van uit de berekeningen resulterende coördinaten blijven vier x, y-coördinaten over, ofwel punt objecten.
57	Deze regel maakt van de vier tijdelijke punt objecten, gemaakt in regels 28 t/m 55, tot een polygoon object.
60	Creëert de gdalsubprocess() functie.
61t/m65	Stelt een gdal prompt in wat als input het polygoon object meeneemt. Op basis van deze polygoon zorgt de gdal functie gdalwarp met als extra flag -cutline ervoor dat de correcte uitsneden worden gemaakt. Deze worden opgeslagen als vrt bestand.
67t/m71	De prompt in deze regel converteert de vrt of virtual raster tiles welke zijn gecreëerd in regels 61 t/m 65 naar geotif bestanden. Op deze manier zijn het losstaande bestanden en kunnen deze makkelijk verplaatst worden.

B.III Installatie Labelimg

Beschrijving:

1. instaleer labelimg

Met pip:

conda create labelimg python (alleen als gebruik wordt gemaakt van een conda omgeving)

pip install lxml

pip install PyQt5

Of met conda:

conda create -n labelimg_conda python pyqt=5 lxml

Download github: <https://github.com/heartexlabs/labelImg/archive/refs/heads/master.zip> en pak deze uit

Navigeer naar de folder labelimg-master

Compileer de tool via de volgende prompt: `pyrcc5 -o libs/resources.py resources.qrc`

Open de gui via de volgende prompt: `python labelimg.py`

2. Annoteren met labelimg tooling

Open de folder met de bestanden die geannoteerd dienen te worden

Stel het juiste annotatie formaat in (YOLOv5)

Annoteer alle objecten in deze folder (handmatig proces)

C.I Match.PY (1/3)

```
1 import os
2 import shutil
3 import cv2
4 import pandas as pd
5 #check which images match the annotation name
6 def rename_files(dir="train"):
7     nn = 0
8     #rename names to a more readable normalized title:
9     for i in os.listdir(dir):
10         if i.endswith(".tif"):
11             im = i
12             continue
13         if i.endswith(".txt"):
14             if i[:-4] == im[:-4]:
15                 nn += 1
16                 n = f"{nn}"
17                 n = n.zfill(10)
18                 os.rename(f"{dir}/{im}", f"{dir}/{n}.tif")
19                 os.rename(f"{dir}/{i}", f"{dir}/{n}.txt")
20     else: print("nope")
```

Regels	Beschrijving
1t/m4	 Importeert vereiste modules; os, shutil, cv2, pandas.
6	 Creëert de functie rename_files(), deze neemt een folder locatie als parameter.
7	 Maakt een variabele welke zal dienen als unieke identificator.
9	 Een for loop. Deze for loop itereert over alle bestanden in de folder welke als parameter aangegeven dient te worden bij de rename_files() functie.
10 t/m12	 Controleert of het geïtereerde bestand een .TIFF extensie bevat, is dit het geval, wordt de naam van dit bestand opgeslagen in de im variabele en zal het script voortzetten.
13	 Controleert of het geïtereerde bestand een .txt extensie bevat, is dit het geval zal het script vervolgt Worden en het bestand zal onder het iteratie object i opgeslagen blijven.
14	 Controleert of im gelijk is aan i. Dit wil zeggen, als de TIFF bestandsnaam gelijk is aan de tekstbestandsnaam zal het script vervolgen.
15	 telt de unieke identificatie genoemd in regel 7 op met 1.
16	 Slaat deze waarde op als string data type.
17	 Vult deze string op met, 0 karakters, tot de volledige string bestaat uit 10 karakters.
18t/m19	 Hernoemt zowel het bestand dat opgeslagen is onder de variabele im als het bestand dat opgeslagen is in de variabele i met de string gedefinieerd in regel 17.

C.II Match.PY (2/3)

```
22 def if_match_move(im_path="images", anno_path="annotations", out="train"):
23     imgdir = im_path
24     annodir = anno_path
25     annolist = []
26     count = 0
27     # Write annotation names to list
28     for i in os.listdir(annodir):
29         # checking if it is a file
30         count += 1
31         if i.endswith('.txt') and os.path.isfile(annodir+"\""+i):
32             annolist.append(i[:-4])
33     print(count)
34     print(len(annolist))
35
36     count = 0
37     # Check if images in annotation and move
38     for i in os.listdir(imgdir):
39         if i.endswith('.tif') and os.path.isfile(imgdir+"\""+i):
40             if i[:-4] in annolist:
41                 count += 1
42                 shutil.move(imgdir+"\""+i, out)
43                 shutil.move(annodir+"\""+i[:-4]+".txt", "train/anno")
44
45     print(count)
46
```

Regels	Beschrijving
22	Creëert de functie if_match_move(), deze functie vereist drie parameters. Ten eerste de folder locatie naar afbeeldingen, ten tweede de folder locatie van annotatie bestanden en tot slot de folder waar de training data ofwel de afbeeldingen in opgeslagen dienen te worden.
25	Creëert een leeg lijst-object genaamd annolist. Hierin worden de annotatie namen tijdelijk in opgeslagen.
28	De eerste for loop. Deze for loop itereert over alle bestanden in de folder welke als tweede parameter aan is gegeven bij de if_match_move() functie zie regel 22.
31t/m32	Wanneer het geïtereerde object eindigt met een tekst extensie en is in feite een bestandstype en geen folder, zal deze zonder extensie definitie toegevoegd worden aan het legen lijst-object dat is aangemaakt in regel 25.
38	De tweede for loop. Deze for loop itereert over alle bestanden in de folder welke als eerste parameter. aan is gegeven bij de if_match_move() functie.
13	Wanneer het geïtereerde object eindigt met een tif extensie en is in feite een bestandstype en geen folder, zal gecontroleerd word en of de bestandsnaam zonder extensie definitie bestaat in het lijst-object waar in regels 31 t/m 32 invulling aan is gegeven. Is dit het geval zal het script voortzetten.
31t/m32	Verplaatst zowel de annotatie bestanden als de afbeeldingen naar de folder //train en //valid zoals YOLOv5 dit verwacht.

C.III Create_validation_set.PY

```
1 import glob, os
2 import shutil
3
4 tr_obj_dir = r"C:\datasets\windturbine\train\img"
5 tr_lbl_dir = r"C:\datasets\windturbine\train\anno"
6 val_obj_dir = r"C:\datasets\windturbine\valid\img"
7 val_lbl_dir = r"C:\datasets\windturbine\valid\anno"
8 # Percentage of images to be used for the test set
9 percentage_test = 10
10
11
12 # Populate train.txt and test.txt
13 counter = 1
14 index_test = round(100 / percentage_test)
15 for i in glob.iglob(os.path.join(tr_obj_dir, "*.tif")):
16     title, ext = os.path.splitext(os.path.basename(i))
17
18     if counter == index_test:
19         counter = 1
20         shutil.move(tr_obj_dir + "\\" + title + '.tif', val_obj_dir + "\\" + title + '.tif')
21         shutil.move(tr_lbl_dir + "\\" + title + '.txt', val_lbl_dir + "\\" + title + '.txt')
22     else:
23         counter = counter + 1
```

Regels	Beschrijving
1t/m2	 Importeert vereiste modules; os, glob, shutil.
4t/m7	 Creëert een vier variabelen die ieder naar een specifieke folder wijzen. Het eerste variabel wijst naar de folder waar de afbeeldingen in opgeslagen zijn welke gebruikt kunnen worden als training data. Het tweede variabel wijst naar de annotatie data dat bij de training data hoort. Het derde variabel wijst naar de folder waar een deel van de training data naar toe wordt getransporteerd tijdens dit script. En het vierde variabel wijst naar de folder waar een deel van de annotatie data naar toe wordt getransporteerd horend bij de getransporteerde afbeeldingen.
9	 creëert een variabel dat een numerieke waarde is gegeven. Deze waarde verwijst naar het percentage data dat als validatie data gebruikt zal worden.
13	 Creëert de variabel counter. Deze variabel is de numerieke waarde 1 gegeven. Deze waarde zal in een later stadium van dit script dienen als hulpmiddel om het aantal iteraties bij te houden.
14	 Berekent het percentage over de gegeven waarde in regel 13 en bewaart de nieuwe waarde in een de variabele index_test.
15	 Een for loop. Deze for loop itereert over alle bestanden in de folder welke eindigen met een tif extensie.
16	 Splitst de bestandsnaam van de extensie en bewaart deze in een tuple object.
18t/m23	 Controleert of het variabel counter gelijk is aan de waarde van index_test. Is dit het geval wordt het variabel counter gereset naar de waarde 1 en het iteratie item verplaatst naar de validatie folder opgegeven bij regels 4 t/m 7. Dit zelfde wordt gedaan met het annotatie bestand met dezelfde naam. Is counter niet gelijk aan het variabel index_test wordt counter met 1 opgeteld.

C. IV Match.PY (3/3)

```
104 import glob, os
105 import shutil
106 import cv2
107 # convert xxyy coords to pixel values:
108 SCREEN_DIMENSIONS = (1279, 1279)
109 TEST_IMGAGE = r"E:\python_script\pytorch_custom_code\train\anno\000000070.txt"
110 def xxyy2pixel_(coords):
111     return tuple(round(coord * dimension) for coord, dimension in zip(coords, SCREEN_DIMENSIONS))
112
113 #check if annotations are correctly matched with images:
114 def check_annotations(img_in=TEST_IMGAGE):
115     # Reading an image in default mode
116     image = cv2.imread(img_in)
117     # Window name in which image is displayed
118     window_name = 'Image'
119
120     # represents the top left corner of rectangle
121     start_point = xxyy2pixel_(XminYmin)
122
123     # represents the bottom right corner of rectangle
124     end_point = xxyy2pixel_(XmaxYmax)
125     # Blue color in BGR
126     color = (255, 0, 0)
127     # Line thickness of 2 px
128     thickness = 2
129
130     # Using cv2.rectangle() method
131     # Draw a rectangle with blue line borders of thickness of 2 px
132     image = cv2.rectangle(image, start_point, end_point, color, thickness)
133
134     # Displaying the image
135     imS = cv2.resize(image, (540, 540))
136     cv2.imshow(window_name, imS)
137     k = cv2.waitKey(0)
138     if k==ord('q'):
139         cv2.destroyAllWindows()
140
141
142 check_annotations(r"E:\python_script\pytorch_custom_code\train\img\000000070.tif")
143
```

Regels	Beschrijving
--------	--------------

Dit script is optioneel en is daarom niet nader toegelicht. Dit script bevat daarentegen voldoende notities in de code zelf om de functionaliteit van dit script in terug te vinden.	
--	--

D.I gisify.PY

```
1 from osgeo import osr, ogr, gdal
2 import os
3 from shapely.geometry import mapping, Polygon
4 from shapely.ops import unary_union
5 import fiona
6 import geopandas as gpd
7
8 cnt = 0
9 polylist = []
10 confv = []
11
12 def pixel_to_world(geo_matrix, x, y):
13     ul_x = geo_matrix[0]
14     ul_y = geo_matrix[3]
15     x_dist = geo_matrix[1]
16     y_dist = geo_matrix[5]
17     _x = x * x_dist + ul_x
18     _y = y * y_dist + ul_y
19     return _x, _y
20
21
22 SCREEN_DIMENSIONS = (1280, 1280)
23 def xywh2xyxy(x,y,w,h):
24     x1, y1 = x-w/2, y-h/2
25     x2, y2 = x+w/2, y+h/2
26     return x1, y1, x2, y2
27
28 def xyxy2pixel(coords):
29     return tuple(round(coord * dimension) for coord, dimension in zip(coords, SCREEN_DIMENSIONS))
30 TEST_ANNO = r"C:\\yolov5\\runs\\detect\\exp2\\labels"
31 for i in os.listdir(TEST_ANNO):
32
33
34
35     with open((r"C:\\yolov5\\runs\\detect\\exp2\\labels\\"+i), "r") as test_txt:
36         for line in test_txt:
37             test_txt = line.split(" ")
38
39             if int(test_txt[0])==0:
40
41                 xyxy = xywh2xyxy(float(test_txt[1]),float(test_txt[2]),float(test_txt[3]),float(test_txt[4]))
42                 XminYmin = (xyxy[0],xyxy[1])
43                 XmaxYmax= (xyxy[2],xyxy[3])
44
45                 XminYmax = (xyxy[0],xyxy[3])
46                 XmaxYmin = (xyxy[2],xyxy[1])
```

Regels

Beschrijving

- | | | |
|-------|-------|---|
| 1t/m6 | | Importeert vereiste modules; osgeo (gdal), os, shapely.geometry (mapping, polygon), fiona, geopandas. |
| 8 | | Definieert een variabelen onder de noemer cnt, deze variabelen zal gebruikt worden om een telling bij te houden in een for loop later in het script. |
| 9 | | Creëert een leeg lijst-object dat later in het script gebruikt zal worden om polygoon gegevens in op te slaan. |
| 10 | | Creëert een leeg lijst-object dat later in het script gebruikt zal worden om confidence waardes in op te slaan. |
| 12 | | creëert de functie pixel_to_world(), deze functie neemt het extent van een tif bestand als eerste parameter, een pixel-waarde langs de x-as van het tif bestand als tweede parameter en een pixel-waarde langs de y-as van het tif bestand als derde parameter. |

Regels	Beschrijving
13	Extraheert de waarde welke is opgeslagen in de eerste indexing van het GetGeoTranform object. Deze waarde representeert de ruimtelijke x-coördinaat van de hoge linkse hoek van de pixel-waarde op deze locatie.
14	Extraheert de waarde welke is opgeslagen in de vierde indexing van het GetGeoTranform object. Deze waarde representeert de ruimtelijke y-coördinaat van de hoge linkse hoek van de pixel-waarde op deze locatie.
15	Extraheert de waarde welke is opgeslagen in de tweede indexing van het GetGeoTranform object. Deze waarde representeert de pixel breedte.
16	Extraheert de waarde welke is opgeslagen in de tweede indexing van het GetGeoTranform object. Deze waarde representeert de pixel hoogte waarde representeert de x-coördinaat van de hoge linkse hoek van de pixel-waarde op deze locatie.
17	Deze regel berekent het ruimtelijke x-as coördinaat door de x-as pixel-waarde te vermenigvuldigen met de pixel breedte en vervolgens de ruimtelijke x-coördinaat van de hoge linkse hoek van het tif bestand hierbij op te tellen.
17	Deze regel berekent het ruimtelijke y-as coördinaat door de y-as pixel-waarde te vermenigvuldigen met de pixel hoogte en vervolgens de ruimtelijke y-coördinaat van de hoge linkse hoek van het tif bestand hierbij op te tellen.
17	Deze regel berekent het ruimtelijke x-as coördinaat door de x-as pixel-waarde te vermenigvuldigen met de pixel breedte en vervolgens de ruimtelijke x-coördinaat van de hoge linkse hoek van het tif bestand hierbij op te tellen.
18	Deze regel berekent het ruimtelijke x-coördinaat door de x-as pixel-waarde te vermenigvuldigen met de pixel breedte en vervolgens de ruimtelijke x-coördinaat van de hoge linkse hoek van het tif bestand hierbij op te tellen.
19	Als output geeft deze functie het ruimtelijke x-coördinaat en ruimtelijke y-coördinaat resulterend out regel 17 en 18.
22	Creëert een variabelen genaamd dat het aantal pixels over de x en y-as opslaat in een tuple object.
23	Creëert de functie xywh2xyxy_(), deze functie converteert het middelpunt van een x-as en het middelpunt van een y-as, de breedte en de hoogte van een bounding box naar de minimale en maximale x en y-as waardes van de bounding box. Hiervoor neemt deze functie de genormaliseerde x en y-as middelpunten en de breedte en hoogte van de bounding box als parameters.
24t/m25	Om de genormaliseerde minimale en maximale x en y-as waardes te krijgen worden de middelpunten van een x en y-as minus de breedte en hoogte gedeeld door 2 berekent.
26	Als output geeft deze functie de minimale en maximale x en y-as waardes.
28	Creëert de functie xyxy2pixel_(), deze functie converteert de minimale en maximale x en y-as waardes naar de pixel-waarde van de tif afbeelding. Deze functie neemt tuple object van een set van een x en y coördinaten als paramater.
28	Creëert de functie xyxy2pixel_(), deze functie converteert de minimale en maximale x en y-as waardes naar de pixel-waarde van de tif afbeelding. Deze functie neemt een enkele genormaliseerde coördinaat of meerdere genormaliseerde coördinaten in een lijst object als parameter.
29	berekent de pixel waarde door een genormaliseerde coördinaat te vermenigvuldigen met de maximale pixel dimensie.
30	Definieert de locatie van de folder "labels". Deze folder bevat alle annotaties afkomstig uit de resultaten van het YOLOv5 model.
31	Itereert over alle annotatie bestanden.
35	Voor elke iteratie open het geselecteerde bestand.
36	Itereert over alle tekst regels van per annotatie bestand.
37	Voor ieder annotatie bestand splitst iedere regel op in delen waar een spatie voorkomt en slaat deze opgesplitste regel op als lijst object.

Regels	Beschrijving
39	 Controleert voor iedere iteratie of de eerste index van het lijst object genoemd in regel 37 gelijk is aan 0 (ofwel object klasse 0), is dit het geval, vervolgt het script verder (Wanneer meer object klassen geconverteerd dienen te worden naar GIS dient deze if statement niet uitgevoerd te worden).
41	 Initieert per iteratie de functie xywh2xyxy_() eerder genoemd in regel 23 en neemt als input Parameters de lijst elementen voor de tweede, derde, vierde en vijfde indexering uit het lijst object afkomstig uit de opgesplitste tekst regel en doet dit voor iedere iteratie ofwel ieder annotatie bestand. Deze lijst elementen bevat het middelpunt van een x-as en het middelpunt van een y-as, de breedte en de hoogte van een bounding boxen. De output dat resulteert de initiatie van de functie wordt opgeslagen in een variabele.
42	 De genormaliseerde minimale x en y-as worden opgeslagen als een set in een tuple object.
43	 De genormaliseerde maximale x en y-as worden opgeslagen als een set in een tuple object.

```

49     # convert xyxy coords to pixel values:
50     xmin=(xyxy2pixel_(XminYmin)[0])
51     ymin=(xyxy2pixel_(XminYmin)[1])
52     xmax=(xyxy2pixel_(XmaxYmax)[0])
53     ymax=(xyxy2pixel_(XmaxYmax)[1])
54
55
56
57     ds = gdal.Open(r"E:\\detect_data\\wind_turbi_test_data_000\\"+i[: -4]+".tif")
58
59     world_Xmin, world_Ymin = pixel_to_world(ds.GetGeoTransform(), xmin, ymin)
60     world_Xmax, world_Ymax = pixel_to_world(ds.GetGeoTransform(), xmax, ymax)
61     world_Xmin, world_Ymax = pixel_to_world(ds.GetGeoTransform(), xmin, ymax)
62     world_Xmax, world_Ymin = pixel_to_world(ds.GetGeoTransform(), xmax, ymin)
63
64     coord1 = (world_Xmax, world_Ymax)
65     coord2 = (world_Xmax, world_Ymin)
66     coord3 = (world_Xmin, world_Ymin)
67     coord4 = (world_Xmin, world_Ymax)
68
69     # Here's an example Shapely geometry
70     poly = Polygon([coord1, coord2, coord3, coord4])
71     polylist.append(poly)
72     confv.append(float(test_txt[5]))
73     print(len(polylist), " ", len(confv))
74     # Define a polygon feature geometry with one attribute
75
76     schema = {
77         'geometry': 'Polygon',
78         'properties': {'id': 'int',
79                        'confv': 'float'},
80     }
81
82     # Write a new Shapefile
83
84     with fiona.open(f'E:\\python_script\\gisified_results\\square_bbxs_turb2_conf.shp', 'w', 'ESRI Shapefile', schema) as c:
85         ## If there are multiple geometries, put the "for" loop here
86         for i, cnf in zip(polylist, confv):
87             cnt+=1
88             c.write({
89                 'geometry': mapping(i),
90                 'properties': {'id': cnt,
91                              'confv': cnf},
92             })
93

```

50t/m53	 Initieert de xyxy2pixel() functie individueel voor de minimale en maximale x en y-as gecollecteerd uit de tuple objecten gecreëerd in regels 42 en 43 en slaat deze op in aparte variabelen.
57	 Opent een geotif bestand dat is gebruikt om de annotatie data te genereren door middel van het. detectie model en slaat dit bestand op als variabelen zodat deze later terug gerefereerd kan worden.
59t/m62	 Initieert de Pixel_to_world() functie en gebruikt als input parameters de pixel-waardes die in regels 50 t/m 53 zijn opgeslagen in aparte variabelen om de ruimtelijke coördinaten te collecteren voor de vier hoekpunten van de bounding boxen voor iedere iteratie (ofwel voor iedere annotatie bestand).

Regels	Beschrijving
64t/m67	Slaan de x en y-as coördinaten per set van twee (ofwel een punt) met de combinatie die de Vierhoekpunten vormen als vier aparte variabelen.
70	Vormt voor iedere iteratie (ofwel voor iedere annotatie bestand) met de vierhoekpunten die in regel 64 t/m 67 zijn opgeslagen in aparte variabelen een polygoon object dat de bounding box representeert
71	Voegt voor iedere iteratie het gemaakte polygoon object toe aan het lijst object die is gedefinieerd in regel 9.
72	Voegt voor iedere iteratie de vijfde indexering van het lijst object afkomstig uit iedere geïtereerde regel uit het annotatie tekstbestand toe aan het lijst object dat in regel 10 is gedefinieerd. Deze index waarde representeert de confidence waarde van het gedetecteerde object. Dit is het einde van de for loop welke is geïnitieerd in regel 36.
76t/m80	Creëert een schema dat gebruikt wordt om in een later stadium een shapefile te kunnen creëren, hier worden eigenschappen zoals kolommen vooraf gedefinieerd.
84	Creëert met de fiona module een lege shapefile.
86	Initieert een for loop dat itereert over alle polygoon objecten en confidence waardes die zijn opgeslagen in twee lijst objecten in regels 71 en 72.
87	Telt voor iedere iteratie een enkele waarde bij de variabelen cnt welke is gedefinieerd in regel 8.
88t/m92	Schrijft de geïtereerde objecten, het polygoon object en de confidence waarde per iteratie in de kolommen die zijn aangemaakt in het schema in regels 76 t/m 80. Dit is het einde van de derde geïnitieerde for loop (geïnitieerd in regel 86).

```

94
95     # GeoDataFrame creation
96     poly = gpd.read_file("E:\\python_script\\gisified_results\\square_bbxes_turb2_conf.shp")
97     points = poly.copy()
98     points.geometry = points['geometry'].centroid
99     # same crs
100    points.crs=poly.crs
101    # save the shapefile
102    points.to_file('E:\\python_script\\gisified_results\\bbxs_centroid_turb2_conf.shp')
103

```

96	Slaat de shapefile op inbegrepen met de polygoon objecten en bijhorende confidence waardes doormiddel van de geopandas module.
97	Maakt een kopie van dit shapefile bestand.
98	Berekent het middelpunt voor ieder polygoon object in de shapefile van het gekopieerde bestand
100	Zorgt ervoor dat hetzelfde coördinaten systeem behouden blijft.
102	Slaat het nieuwe shapefile bestand op met de berekent middelpunten (centroids) die zijn berekent in regel 98.

D.II mini_mistake.PY

```
1 import subprocess
2 import os
3 def gdalsubprocess( shp_name, shp_filepath, tif_path, file):
4
5     cmd = f'''C:\OSGeo4W\bin\gdal_rasterize -b 1 -b 2 -b 3 -burn 0 -burn 0 -burn 0 -l {shp_name} {shp_filepath} {tif_path + "/" + file}'''
6     print(cmd)
7
8     subprocess.run(cmd,shell=True)
9 tif_path = r'E:\detect_data\wind_turbi_test_data_000'
10 shp_filepath = r"E:\\Data\\uitsluitinggebieden\\kunststof_sportvelden_.shp"
11 shp_name = os.path.split(shp_filepath)
12 for file in os.listdir(tif_path):
13     gdalsubprocess(shp_name[1][:-4], shp_filepath, tif_path,file)
14
```

- 1t/m2 Importeert vereiste modules; subprocess en os.
- 3 Initieert de gdalsubprocess() functie. Deze neemt als input vier parameters, een shapefile, de locatie van de shapefile, een geotif bestand en de locatie van dit bestand. De shapefile levert de begrenzing van de uitsneden die toegepast zal worden over het geotif bestand.
- 5 Slaat de command prompt voor de OSGeo4W GDAL shell op in een variabele.
- 8 Voert een sub proces uit met de opgeslagen command prompt.
- 9 Slaat de locatie op van het geotif bestand(en) dat gebruikt dient te worden voor dit script.
- 10 Slaat de locatie op van het shapefile bestand dat gebruikt dient te worden voor dit script.
- 11 Koppelt de shapefile bestandnaam los van het volledige pad.
- 12t/m13 Initieert een for loop over de folder dat het geotif bestand (of bestanden) bevat en gebruikt de bestandsnaam en locatie van de geotif bestanden het shapefile bestand om de gdalsubprocess functie aan te roepen, zie regel 3.

E.I alumentation.PY

```
1 import os
2 import cv2
3 import albumentations as A
4 from PIL import Image as im
5 import numpy as np
6 # Define the path to the input directory containing the images
7 input_dir = "C:\\datasets\\windturbine\\traindata_backup\\images"
8
9 # Define the path to the output directory where augmented images will be saved
10 output_dir = "C:\\datasets\\windturbine\\traindata_backup\\images_augmented"
11 output_annotation_path = "C:\\datasets\\windturbine\\traindata_backup\\labels_augmented"
12 bbxes = []
13 cl_nr = []
14 # Define the augmentation transformations you want to apply
15 transform = A.Compose([
16     A.OneOf([
17         A.RandomCrop(896,896,p=0.7),
18         A.RandomSizedBBBoxSafeCrop(896,896,p=0.3),
19     ], p=1.0),
20     A.OneOf([
21         A.ElasticTransform(p=0.3,alpha=4,sigma=50),
22         A.GridDistortion(p=0.3, num_steps=5, distort_limit=0.3),
23     ], p=1.0),
24     A.OneOf([
25         A.ToGray(p=0.6),
26         A.RandomBrightnessContrast(p=0.6),
27     ], p=1.0),
28     A.Flip(always_apply=True)
29     # Add more desired transformations here
30 ], bbox_params=A.BboxParams(format="yolo", label_fields=[]))
31
32
33 # Iterate over the images in the input directory
34 for filename in os.listdir(input_dir):
35     box = []
36     cl_nr = []
37     with open(input_dir[:-6]+"labels"+"\\ "+filename[:-4]+".txt", "r") as f:
38         for line in f:
39             test_txt = line.split(" ")
40
41             if int(test_txt[0])==0:
42                 n=0
43             if int(test_txt[0])==1:
44                 n=1
45             box = [float(test_txt[1]),float(test_txt[2]),float(test_txt[3]),float(test_txt[4])]
46             print(box)
47             cl_nr.append(n)
48             bbxes.append(box)
```

Regels	Beschrijving
1t/m5	Importeert vereiste modules; os, cv2, alumentations, PIL, numpy.
7	Definieert een variabelen dat de locatie van de folder met training input / afbeeldingen bevat.
10	Definieert een variabelen dat de locatie van de folder waarin geaugmenteerde training input / afbeeldingen in opgeslagen kunnen worden.
11	Definieert een variabelen dat de locatie van de folder waarin geaugmenteerde annotatie input / labels in opgeslagen kunnen worden.
12	Creëert een leeg lijst object dat gebruikt kan worden om bounding box coördinaten in op te slaan.

Regels	Beschrijving
15	Creëert een structuur waarin augmentatie transformaties kunnen worden aangeroepen.
16	Stelt in dat van twee of meerdere technieken een enkele techniek doorvoert gebaseerd op de p waarde aangegeven in regel 19.
17t/m18	Initieert de RandomCrop en RandomSizesBBBoxSafeCrop technieken toe voor een kleinere resolutie dan het origineel zodat bepaalde windturbines worden opgesplitst in delen. De p waarde bepaalt hier de percentuele kans dat deze techniek zal worden doorgevoerd.
19	Stelt een kans voorwaarde in dat van twee of meerdere technieken een enkele techniek doorvoert . gebaseerd op de p waarde aangegeven.
20	Stelt in dat van twee of meerdere technieken een enkele techniek doorvoert gebaseerd op de p waarde aangegeven in regel 23.
21t/m22	Initieert de ElasticTransform en GridDistortion technieken toe voor een kleinere resolutie dan het origineel zodat bepaalde windturbines worden opgesplitst in delen. De p waarde bepaalt hier de percentuele kans dat deze techniek zal worden doorgevoerd.
23	Stelt een kans voorwaarde in dat van twee of meerdere technieken een enkele techniek doorvoert gebaseerd op de p waarde aangegeven.
24	Stelt in dat van twee of meerdere technieken een enkele techniek doorvoert gebaseerd op de p waarde aangegeven in regel 27.
25t/m26	Initieert de ToGray en RandomBrightnessContrast technieken toe voor een kleinere resolutie dan . het origineel zodat bepaalde windturbines worden opgesplitst in delen. De p waarde bepaalt hier de percentuele kans dat deze techniek zal worden doorgevoerd.
27	Stelt een kans voorwaarde in dat van twee of meerdere technieken een enkele techniek doorvoert gebaseerd op de p waarde aangegeven.
28	Initieert met 100% kans waarde de Flip techniek toe.
30	Stelt in dat de aangepaste afbeeldingen de originele annotaties behoudt en definieert het formaat waarin de geaugmenteerde annotaties dienen opgeslagen te worden.
34	Creëert een for loop dat itereert over de annotaties van de afbeeldingen uit de folder gedefinieerd in regel 7.
35	Creëert een leeg lijst object dat gebruikt kan worden om de bounding boxes in op te slaan per afbeelding.
36	Creëert een leeg lijst object dat gebruikt kan worden om object klas nummerwaardes in op te slaan.
37	creëert een while loop en itereert over de originele annotatie bestanden van de originele training data.
41t/m44	Als de numerieke object klasse uit de originele annotaties gelijk is aan 0 word deze waarde in een tijdelijke variabele opgeslagen en als deze waarde gelijk is aan 1, worde deze waarde hierin opgeslagen.
45	Slaat de originele bounding boxes per iteratie op in het lijst object gedefinieerd in regel 35.
47	Slaat de tijdelijke variabele opgeslagen in regels 41 t/m 44 op in het lijst object aangemaakt in regel 36.
48	Slaat het lijst object, waar in regel 45 de originele bounding boxes per iteratie in opgeslagen zijn, op in het lijst object dat is aangemaakt in regel 12 en beëindigt de while loop.

```

49
50 image_path = os.path.join(input_dir, filename)
51
52 if os.path.isfile(image_path):
53
54     # Load the image
55     image = im.open(image_path)
56
57
58 image = np.array(image)
59 # Apply the augmentation transformations
60 augmentation = transform(image=image, bboxes=bboxes)
61 augmented_image = augmentation["image"]
62 augmented_bbx = augmentation["bboxes"]
63 # Save the augmented image to the output directory
64 output_path = os.path.join(output_dir, f"augmented_{filename}")
65 cv2.imwrite(output_path, augmented_image)
66
67
68
69 with open(output_annotation_path+"//"+f"augmented_{filename[:-4]}.txt", "w") as f:
70     for bbox, cl_nr in zip(augmented_bbx, cl_nr):
71         f.write(f"{cl_nr} {bbox[0]} {bbox[1]} {bbox[2]} {bbox[3]}\n")

```

Regels	Beschrijving
50	creëert voor iedere for loop iteratie de volledige locatie van de afbeelding door gebruik te maken van de variabele in regel 7.
52	Controleert of de bestand dat in de vorige regel een bestand is en geen folder, is dit het geval vervolgt het script.
55	Laad de afbeelding gebruikmakend van de PIL module in een tijdelijke variabele.
58	Laad de tijdelijke variabele in als een array gebruikmakende van de numpy module.
60	Pas de augmentatie technieken toe over de afbeelding die in regels 17 t/m 28 zijn geformuleerd.
61	Slaat de geaugmenteerde afbeelding data op in een variabelen.
62	Slaat de geaugmenteerde annotaties op in een variabelen.
65	Slaat de geaugmenteerde afbeelding op als bestand met de volgende naam: "augmented_" + origineel Bestandsnaam.
69	Opent een nieuwe textbestand met dezelfde naam als de nieuwe opgeslagen afbeelding (zie regel 64)
70 t/m 72	Initeert een for loop over de geaugmenteerde bounding boxes en de numerieke verwijzing naar de objectklasse en gebruikt deze gegevens om het nieuwe tekstbestand te schrijven.

F.I Hyperparameters (source-code YOLOv5)

```
1 # YOLOv5 🚀 by Ultralytics, GPL-3.0 license
2 # Hyperparameters for medium-augmentation COCO training from scratch
3 # python train.py --batch 32 --cfg yolov5m6.yaml --weights '' --data coco.yaml --img 1280 --epochs 300
4 # See tutorials for hyperparameter evolution https://github.com/ultralytics/yolov5#tutorials
5
6 lr0: 0.01 # initial learning rate (SGD=1E-2, Adam=1E-3)
7 lrf: 0.1 # final OneCycleLR learning rate (lr0 * lrf)
8 momentum: 0.937 # SGD momentum/Adam beta1
9 weight_decay: 0.0005 # optimizer weight decay 5e-4
10 warmup_epochs: 3.0 # warmup epochs (fractions ok)
11 warmup_momentum: 0.8 # warmup initial momentum
12 warmup_bias_lr: 0.1 # warmup initial bias lr
13 box: 0.05 # box loss gain
14 cls: 0.3 # cls loss gain
15 cls_pw: 1.0 # cls BCELoss positive_weight
16 obj: 0.7 # obj loss gain (scale with pixels)
17 obj_pw: 1.0 # obj BCELoss positive_weight
18 iou_t: 0.20 # IoU training threshold
19 anchor_t: 4.0 # anchor-multiple threshold
20 # anchors: 3 # anchors per output layer (0 to ignore)
21 fl_gamma: 0.0 # focal loss gamma (efficientDet default gamma=1.5)
22 hsv_h: 0.015 # image HSV-Hue augmentation (fraction)
23 hsv_s: 0.7 # image HSV-Saturation augmentation (fraction)
24 hsv_v: 0.4 # image HSV-Value augmentation (fraction)
25 degrees: 0.0 # image rotation (+/- deg)
26 translate: 0.1 # image translation (+/- fraction)
27 scale: 0.9 # image scale (+/- gain)
28 shear: 0.0 # image shear (+/- deg)
29 perspective: 0.0 # image perspective (+/- fraction), range 0-0.001
30 flipud: 0.0 # image flip up-down (probability)
31fliplr: 0.5 # image flip left-right (probability)
32 mosaic: 1.0 # image mosaic (probability)
33 mixup: 0.1 # image mixup (probability)
34 copy_paste: 0.0 # segment copy-paste (probability)
35
```

1. Uitbreiding objecttypen

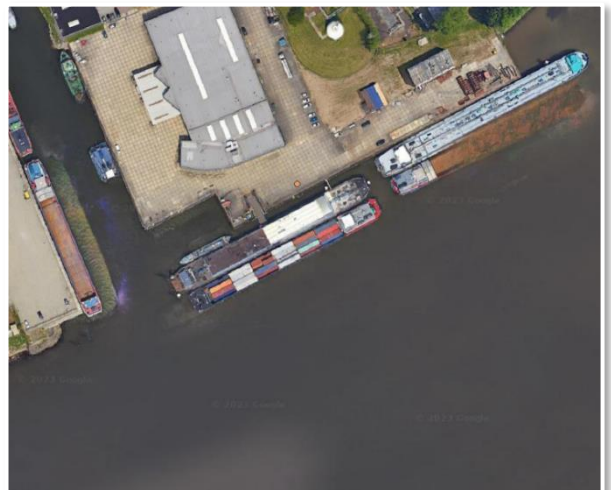
1.1 Kenmerkbare EVActiviteiten

Dit eerste onderdeel bestaat uit objecten die naar kennis zichtbaar zijn op luchtfoto's. Dit betekent dat de methodologie hoogstwaarschijnlijk effectief toegepast kan worden op deze.

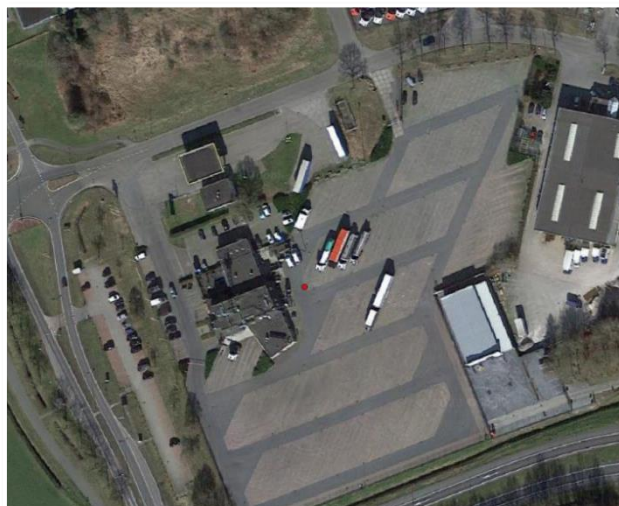
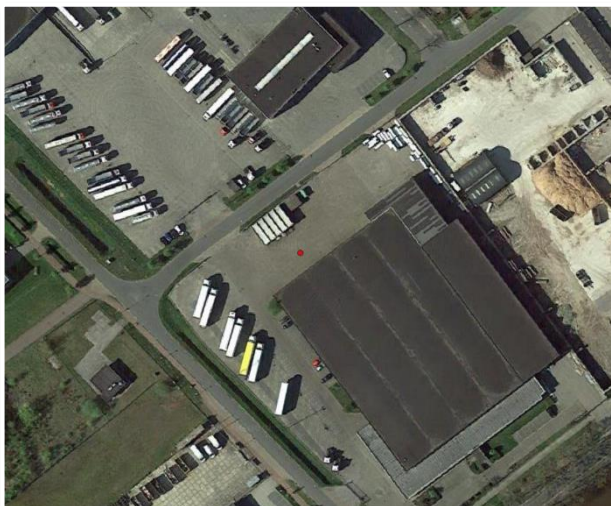
1.1.1 Mestvergisting



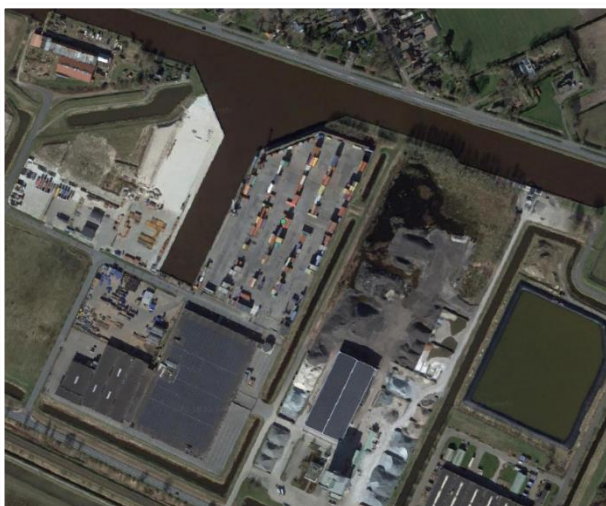
1.1.2 Opslag-Brandstof-Bunkerstation



1.1.3 Opstellen-Voertuigen



1.1.4 Opslag-Seveso-Hoeveelheid-Buiten-Seveso-Inrichting



1.1.5 Tanken (LPG, LNGVoertuig-Werktuig. Waterstof-Voertuig-Werktuig, CNG)



1.1.6 Aardgas (Behandelen-Regelen-Meten-Aardgas)

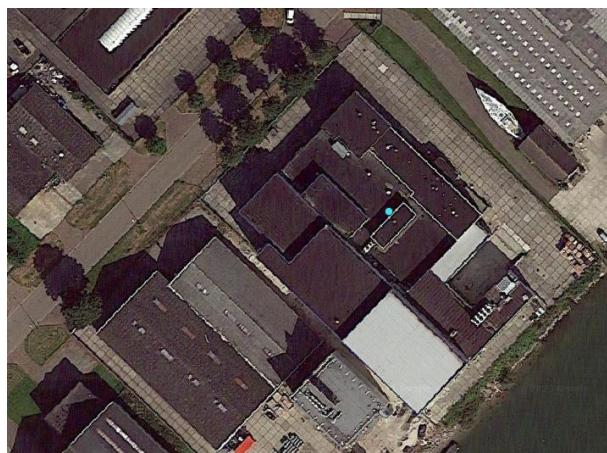


1.1.7 Seveso-Inrichting

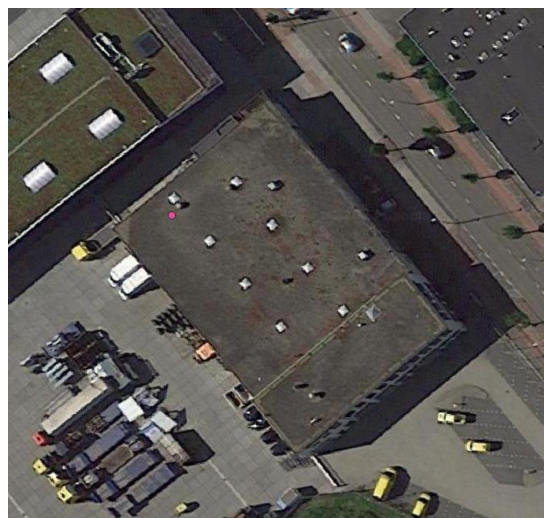
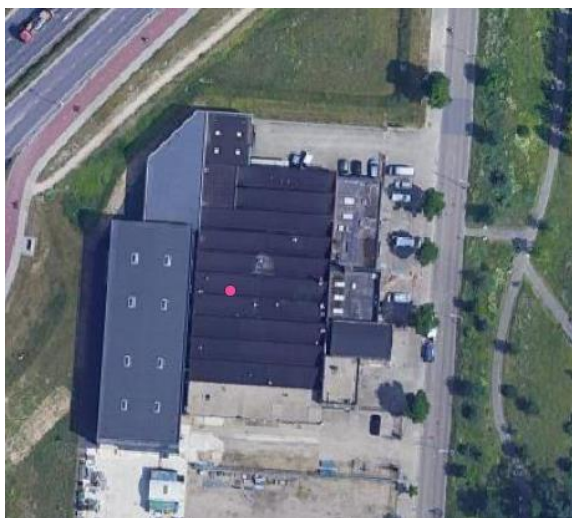


1.2 Niet kenmerkbare EVActiviteiten

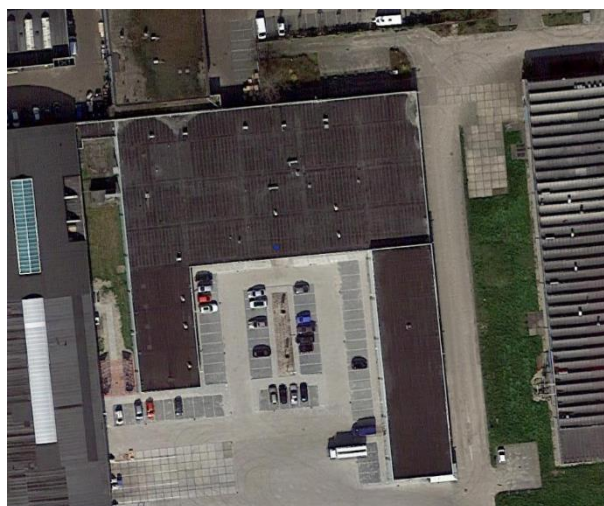
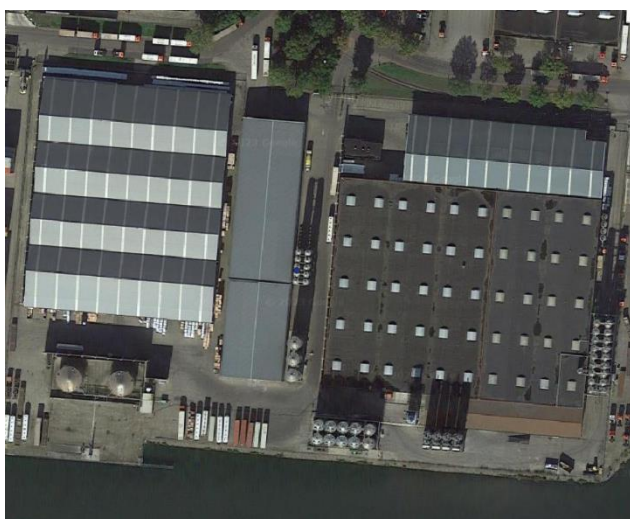
1.2.1 Ammoniak-Koelinstallatie



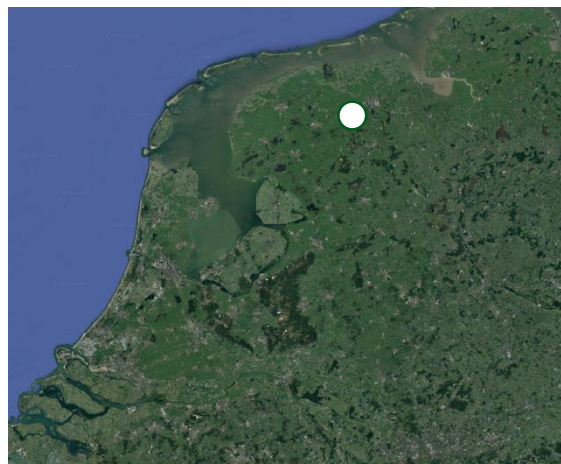
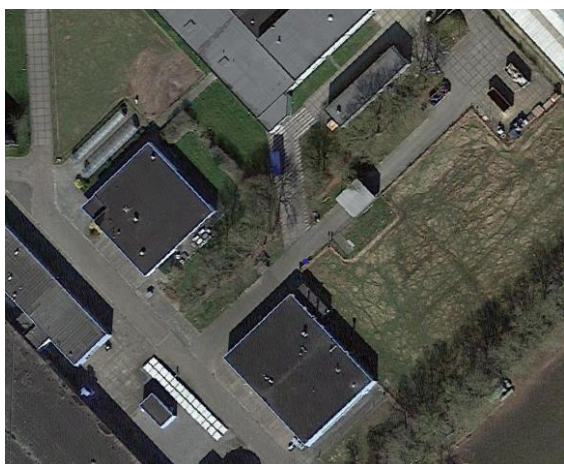
1.2.2 Bad-Giftig



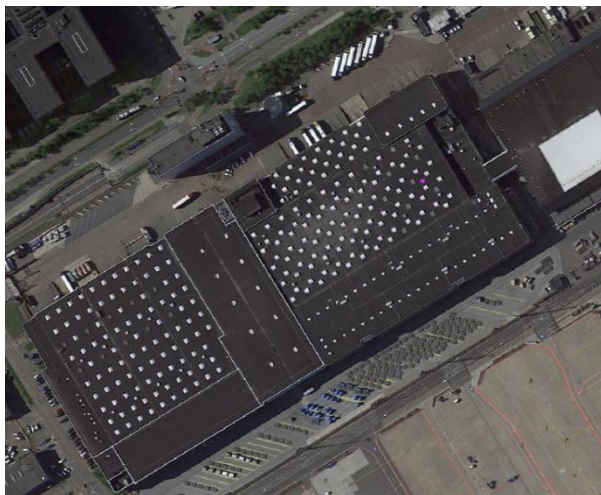
1.2.3 Opslag brandbare vloeistof geen diesel bovengronds



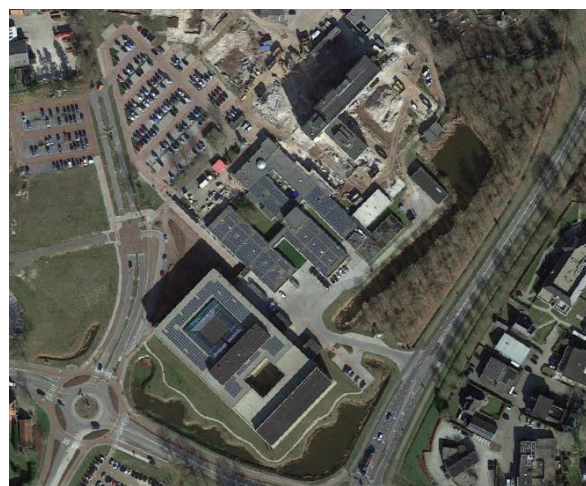
1.2.4 Opslag brandbare vloeistof geen diesel ondergronds



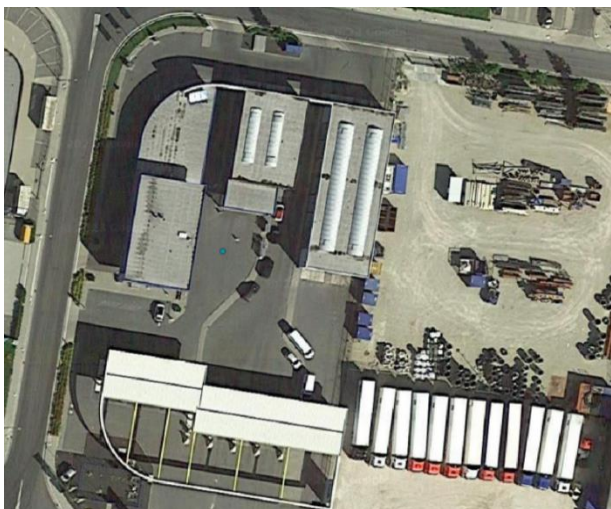
1.2.5 opslag ontplofbaar ADR (1.1, 1.3 en 1.4)



1.2.6 Opslag oxiderend verstikkend gas



1.2.7 Opslagtank gas te berekenen afstand vergunningsplicht



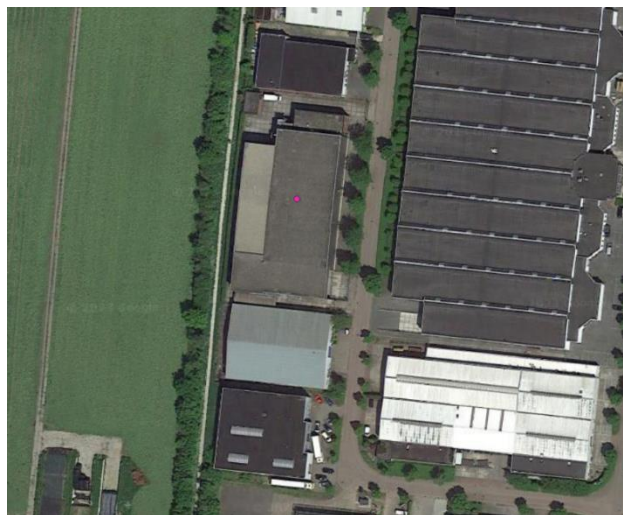
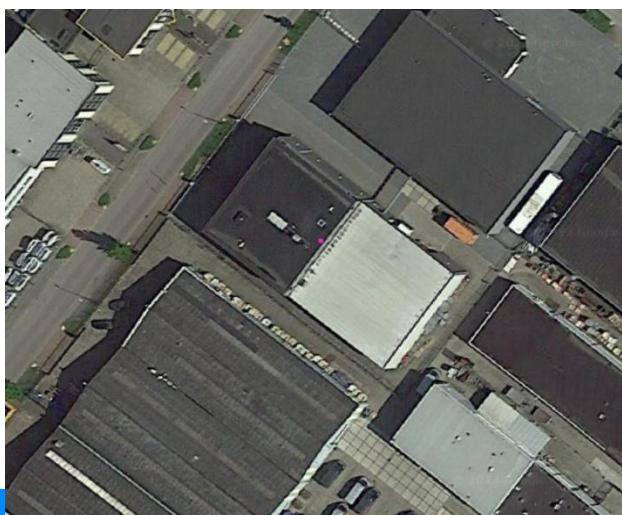
1.2.8 Opslagtank vloeistof



1.2.9 Opslagtank verpakt



1.2.10 Opslag vuurwerk



1.3 Geen onderdeel WFS

De volgende onderdelen zijn vastgelegd in het REV maar zijn niet verwerkt in de vrijgegeven WFS.

- 6.2.1.1 Opslag-Meststof
- 6.2.1.4 Opslag-Ontplofbaar-Militair_Te-Berekenen-Afstand
- 6.2.1.31 Opslag-Vuurwerk-F4
- 6.2.1.25 Opslag-Ontplofbaar-Complexen
- 6.2.1.38 Opslag-Ontplofbaar-Militair_Vaste-Afstand
- 6.2.1.41 Tanken-Grootschalig-Vaartuig
- 6.2.1.14 Tanken-Kleinschalig-Vaartuig
- 6.2.1.13 Kernenergie-Inrichting
- 6.2.1.19 Mijnbouwwerk-Verplaatsbaar
- 6.2.1.37 Mijnbouwwerk-Exploiteren