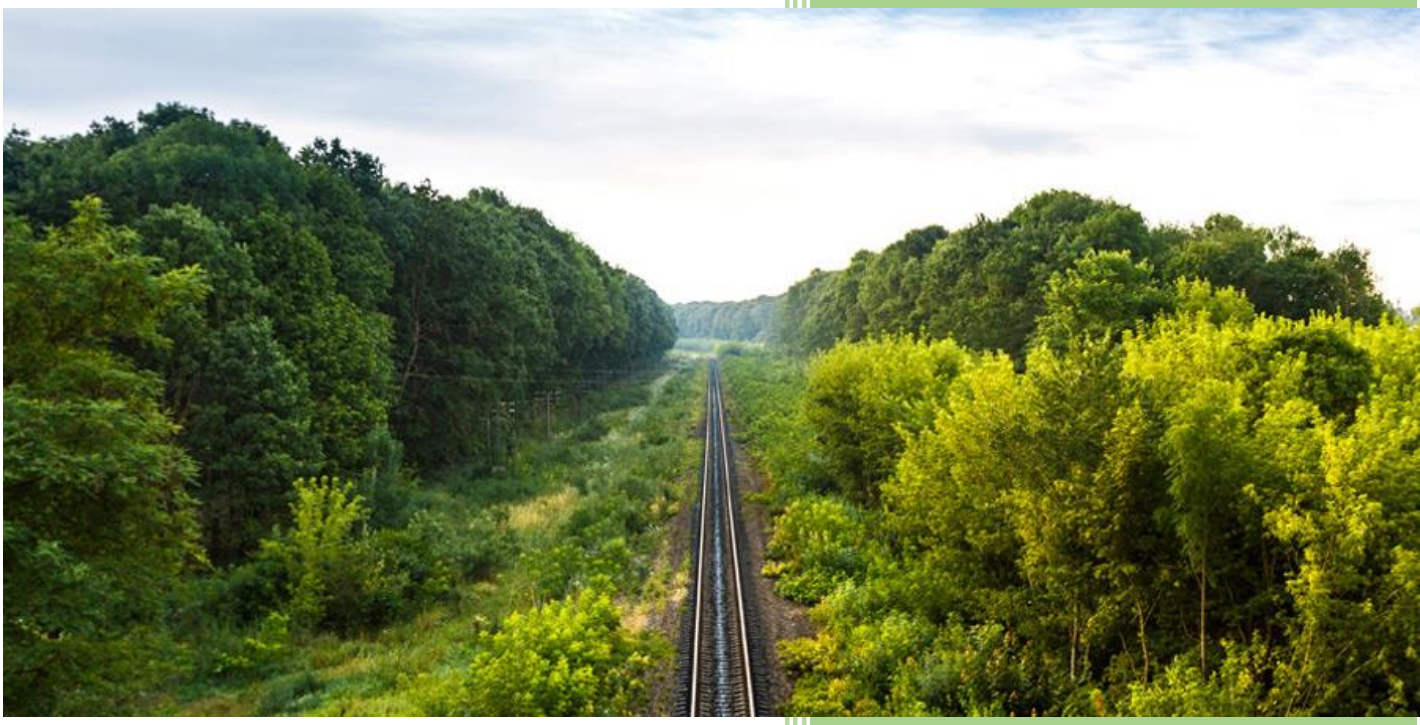
2023

Vegetation Detection



Matthew Jim, Eray Sarikaya, Rick Schellevis, Joey Welvaadt

Asset Insight

30-6-2023

1. Abstract

Voor Asset Insight zijn we op zoek gegaan naar een oplossing voor het detecteren van vegetatie op en langs het spoor. Het is belangrijk om vegetatie en volume daarvan te kunnen detecteren, omdat het schadelijk zou kunnen zijn voor het spoor. Het detecteren van vegetatie moet kunnen gebeuren in hoge snelheid beelden op een AVI-bestand dat vanuit een trein is gefilmd. Het is van belang om een efficiënte en geautomatiseerde oplossing te vinden. Op dit moment beschikt Asset Insight alleen over een groendetectiemodel waarvan de precision en recall niet hoog genoeg scoren. Ons doel is om een model te ontwikkelen dat verbetering laat zien in precision en recall voor het groendetectiemodel van Asset Insight. Daarnaast willen we ook in staat zijn om de oppervlakte van gedetecteerde vegetatie per 10m2 op het spoor te meten.

Voor ons onderzoek zijn we aan de slag gegaan met ons plan van aanpak. We hebben voornamelijk gebruik gemaakt van papers, opmerkingen op GitHub, AI-websites als bronnen. Ook hebben we waardevolle informatie van Asset Insight meegenomen. Ons product is belangrijk omdat het 2-phase model Asset Insight nauwkeuriger inzicht kan bieden op de werkelijke hoeveelheid vegetatie op het spoor in vergelijking met hun groendetectiemodel. Het nieuwe oppervlakte berekening, dat wordt toegepast op polygonen, is een volledig nieuw product voor Asset Insight. Het biedt niet alleen inzicht in de aanwezigheid van vegetatie, maar ook in de daadwerkelijke hoeveelheden in een gegeven kaart op QGIS. Op deze manier bieden wij Asset Insight meer informatie op het vinden van schadelijke vegetatie op het spoor.

2. Inhoudsopgave

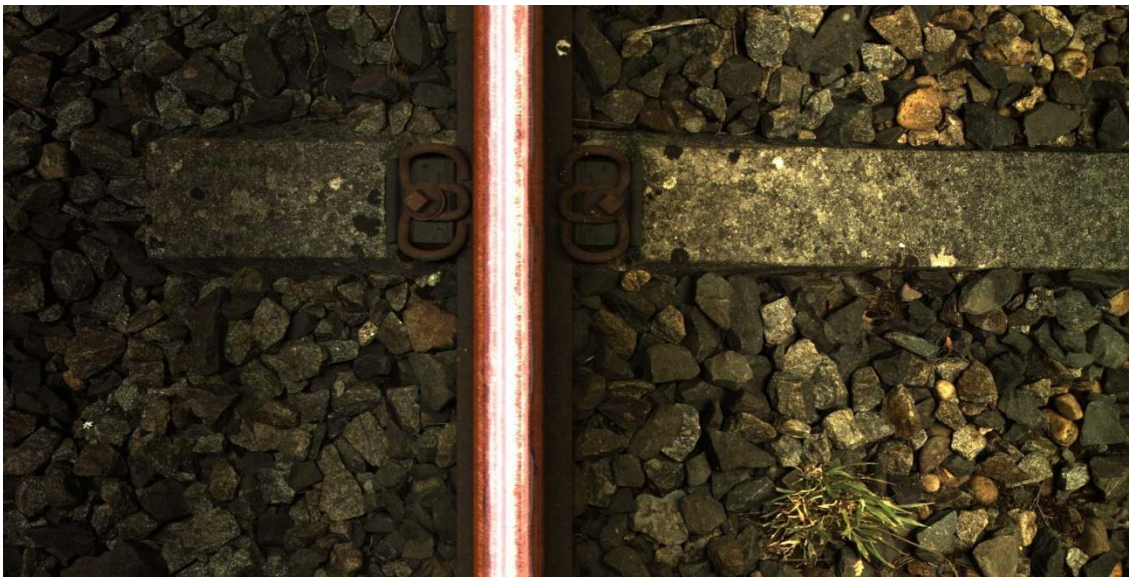
3. Inleiding

3.1. Opdrachtgever

Asset Insight is een meet- en inspectiebedrijf dat machine learning en AI gebruikt om diensten accurater en sneller uit te kunnen voeren. Ze zijn een data gedreven bedrijf dat diensten uitvoert onder andere Inspecties, Spoor- en wegmetingen, IoT-Monitoring, Data analytics & artificial intelligence. Zo doen ze metingen en monitoring op sporen of wegen voor defecten om vroegtijdig onderhoud uit te kunnen voeren. Ook maken ze gebruik van predictive modelling, zij hebben een team van data scientists dat data gebruikt om analyses en predictions te doen. Op deze manier krijgen klanten snel inzicht op huidige en toekomstige staat van hun assets.

3.2. Context

Asset Insight[1] heeft doormiddel van een voertuig genaamd de meettrein[2] die over de Nederlandse spoorwegen rijdt data kunnen verzamelen over deze spoorwegen. Eén van de databronnen die is verzameld bestaat uit hoge kwaliteit video's van boven af op het spoor. (Zie figuur 1)



Figuur 1 Enkele frame van een video van de meettrein

Nu hebben wij de opdracht gekregen om met deze verzamelde data een manier te vinden vegetatie te detecteren. Dit wilt Asset Insight gedaan hebben zodat zij de resultaten kunnen verwerken in QGis[3], zodat er een duidelijk beeld komt van de hoeveelheden vegetatie op de Nederlandse spoorwegen. Hierdoor kunnen onnodige bezoeken van aannemers aan het spoor voorkomen worden.

[1] Asset Insight B.V.
[2] Een trein met een aantal camera en andere meetapparatuur
[3] Programma voor het afbeelden van verschillende soorten landkaarten

3.3. Probleemstelling

Zoals eerder benoemd is er een hoop data verzameld, Asset Insight heeft bedacht dat met deze data er een mogelijkheid zou kunnen zijn om schadelijke begroeiing op specifieke plekken tegen te gaan.

Om dit punt te kunnen bereiken moeten wij een aantal stappen doorlopen. Het begint bij het doorlopen van gegeven video's verzameld uit de meetrein, uit deze video's moeten frames worden gevonden die iets van vegetatie bevat. Deze vegetatie is nog niet per definitie schadelijk, groot of klein. Na dat er frames zijn gevonden kunnen we in de volgende stap gaan kijken naar de totale volume dat de vegetatie in de hiervoor genoemde frame opvult.

Na dat de volume per frame is vast gesteld, kunnen wij deze frame met het nummer koppelen aan GPS locatie. Deze locatie kan dan worden gebruikt om door middel van Light Detection and Ranging (LiDAR) data de hoogte van de gevonden vegetatie te bepalen. Dit alles is nodig om vast te stellen of de vegetatie daadwerkelijk schadelijk is voor het spoor.

Met dit in gedachten kunnen wij nu onderzoeksvraag vaststellen die ons probleem duidelijk beschrijft.

Welke computer vision techniek(en) kunnen wij het beste toepassen bij het via foto's herkennen en meten van schadelijke begroeiing op en rondom de Nederlandse spoorwegen?

Om deze onderzoeksvraag goed te kunnen beantwoorden hebben wij deze opgedeeld in een aantal deelvragen om de weging per vraag te verlichten.

1. Hoe gaan wij de data labelen?
2. Hoeveel gelabelde data moeten wij minimaal hebben?
3. Welke computer vision technieken kunnen wij gebruiken?
4. Op welke manier bepalen wij de volume per 10m2?
5. Hoe meten wij de accuraatheid van ons gebruikte model?
6. Wat moet het model terug geven als resultaat?

De bovenstaande deelvragen zullen ons van een compleet antwoord voorzien.

Bij deelvraag 1 wordt er bepaald met wat voor programma wij de data gaan labelen en hoe de labels worden opgesteld. Na dat deze data op de juiste manier is gelabeld, moeten wij er voor zorgen dat we genoeg data hebben, maar wat is genoeg? Een antwoord wordt hierop gegeven bij het onderzoeken van deelvraag 2.

De laatste aantal deelvragen gaan over het ontwerpen en uitwerken van verschillende computer vision technieken, die ons ondersteunen bij het detecteren en identificeren van schadelijke vegetatie. Als eerst beginnen we bij deelvraag 3 met een onderzoek gericht op het vinden van de juiste computer vision techniek(en) die ons kunnen ondersteunen bij het oplossen van het gestelde probleem.

Een belangrijk punt van ons probleem is dat we moeten kunnen vast stellen hoeveel vegetatie er is per 10m2, dit onderzoeken wij in deelvraag 4.

We komen nu bij het punt dat we moeten bewijzen dat ons model een bepaalde accuraatheid heeft, dit kunnen we op verschillende manieren doen. Hoe we dit doen onderzoeken wij in deelvraag 5.

Als laatst onderzoeken wij in deelvraag 6 wat voor resultaat ons model terug moet geven aan Asset Insight en hoe wij dit resultaat gaan realiseren. Na al deze deelvragen doorlopen te hebben en antwoord gegeven op de vragen, formuleert dat samen het antwoord voor de hoofdvraag en is ons probleem verholpen.

4. Theoretisch Kader

In het theoretisch kader duiken we dieper in op verschillende begrippen die voorkomen in dit onderzoeksrapport.

4.1. Begrippen

4.1.1. Computer Vision

Computer vision is een deel onder de gehele artificial intelligence (AI) tak dat het mogelijk maakt voor computers om waardevolle informatie uit foto's en video's te halen. Dit met het doel einde om bijvoorbeeld acties te ondernemen of aanbevelingen maken bij het zien van bepaalde beelden. (*IBM*. n.d.).

In dit project zijn er twee verschillende technieken van computer vision die wij gebruiken. Dit zijn: Object Detection & Semantic Segmentation.

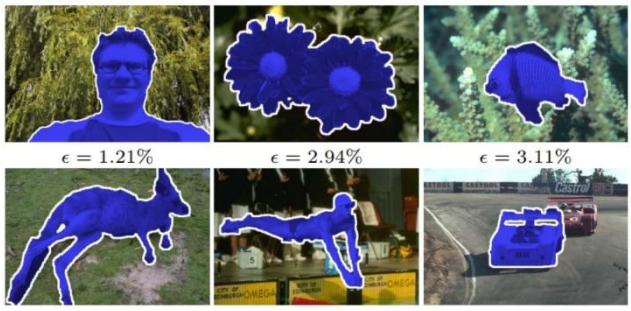
4.1.2. Object Detection

Bij de computer vision techniek genaamd Object Detection is de bedoeling om bepaalde objecten te kunnen identificeren in beelden *(What Is Object Detection?, n.d.).* De manier hoe we dit kunnen bereiken is door middel van het annoteren van bepaalde foto's. In deze foto's zijn de objecten te vinden die gedetecteerd moeten worden, nu moeten om deze objecten vierkanten worden getekend, dit noemt men annoteren. (Zie figuur 2).

Figuur 2 Geannoteerde foto (label rechts boven genaamd weed)

4.1.3. Semantic Segmentation

In deze sectie beschrijven we de computer vision techniek bekend als semantic segmentation. Met deze techniek word elke pixel in een gegeven foto of video gemarkeerd met een class (of label). *(Jordan, 2023)*. Zoals in sectie 4.1.2 is genoemd moeten foto's ook worden geannoteerd. Bij semantic segmentation is het de bedoeling dat deze annotaties een omheining maken van het object dat moet worden gesegmenteerd. (Zie figuur 3).



*Figuur 3 Omheining om object (*How to Annotate Inside Polygons(Holes) in Semantic Segmentation*, n.d.)*

4.1.4. Confusion Matrix

Een confusion matrix is een tabel waarin de voorspelde resultaten van een classificatiemodel worden vergeleken met de werkelijke resultaten. Het helpt bij het visualiseren van de prestaties van het model door het aantal juiste en foutieve voorspellingen voor elke klasse te tonen.

Een confusion matrix heeft meestal vier belangrijke elementen:

- *Waarheid positief (True Positive - TP): Het aantal voorbeelden waarbij het model correct heeft voorspeld dat de positieve klasse daadwerkelijk positief is.*
- *Waarheid negatief (True Negative - TN): Het aantal voorbeelden waarbij het model correct heeft voorspeld dat de negatieve klasse daadwerkelijk negatief is.*
- *Vals positief (False Positive - FP): Het aantal voorbeelden waarbij het model onterecht heeft voorspeld dat de positieve klasse aanwezig is, terwijl deze eigenlijk negatief is.*
- *Vals negatief (False Negative - FN): Het aantal voorbeelden waarbij het model onterecht heeft voorspeld dat de negatieve klasse aanwezig is, terwijl deze eigenlijk positief is. (Narkhede, 2021).*

Met behulp van de waarden in de confusion matrix kun je verschillende evaluatiemetingen berekenen, waaronder de recall, precision en accuracy. (zie sectie 4.1.5 voor recall & percision)

Accuracy: Dit geeft het percentage weer van alle correcte voorspellingen (zowel positief als negatief) ten opzichte van het totale aantal voorbeelden. Het wordt berekend als de som van TP en TN gedeeld door de som van TP, TN, FP en FN:

Accuracy = (TP + TN) / (TP + TN + FP + FN) (Narkhede, 2021)

4.1.5. Recall & Precision

Precision en Recall zijn 2 metrics waar naar gekeken kan worden om in te kunnen zien hoe goed een model presteert. Precision is de metric dat berekent hoe precies een model positieve labels kan detecteren. Recall is de metric dat inzicht geeft op hoe goed een model is in het vinden van alle positieve objecten. Beiden zijn nodig om een goede predictions te kunnen doen. Voor ons doeleinde is recall belangrijker dan precision omdat wij geen data willen missen die er wel zou moeten zijn en is het minder erg om iets extra's te hebben in plaats van belangrijke data te missen. Dit komt ook doordat Asset Insight handmatig gedecteerde frames controleert.

Recall equals True Positives / (True Positives + False Negatives) and precision equals True Positives / (True Positives + False Positives) (Kundu, 2023)

4.1.6. You Only Look Once

You Only Look Once (afkorting YOLO) is een populair object detection algoritme. De eerste versie kwam uit in 2016 met als meest recente versie yolov8 die uitkwam in januari 2023.

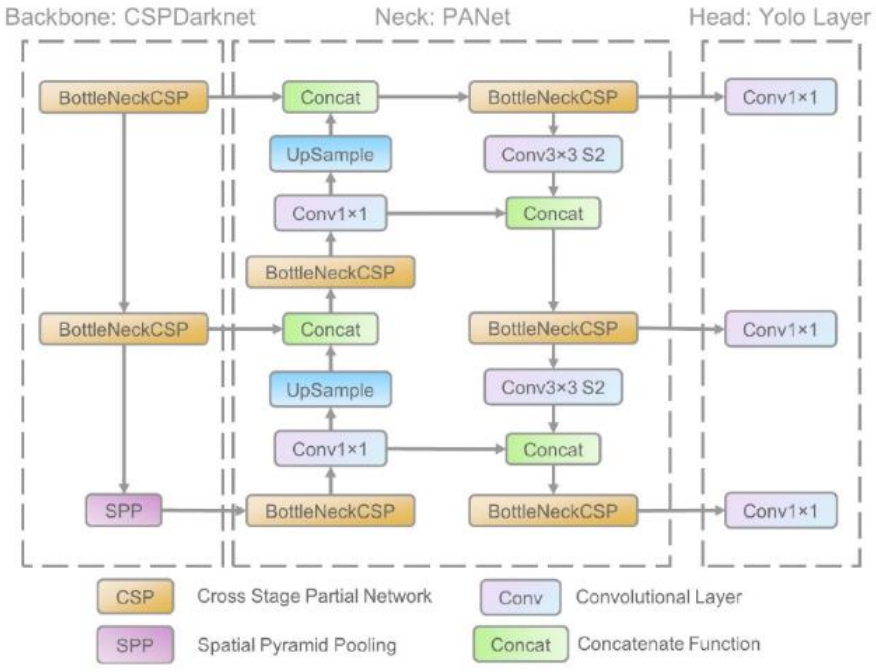
Een YOLO model werkt doormiddel van getraind worden op een training set met gelabelde data erin. In YOLO is het mogelijk om weight decay, data augmentation, learning rate en nog meer hyperparameters aan te passen en in te vullen. YOLO staat vooral bekend voor hun real-time object detection mogelijkheden. Daarnaast is YOLO sneller en accurater dan alle bekende object detectors. (Wang, 2022).

Als het YOLO model voldoende data heeft en is getraind is het instaat om zelf de objecten te detecteren in een frame of foto. In ons project hebben wij gewerkt met YOLOv5, YOLOv7 en YOLOv8.

4.1.7. YOLOv5 Architectuur

Tijdens ons onderzoek hebben wij kunnen concluderen dat YOLO modellen regelmatig worden gebruikt in computer vision projecten waaronder ook projecten gericht op het detecteren van vegetatie.

Dit word als oplossing gebruikt, maar ook om met andere computer vision modellen te vergelijken. *Yolo is a state-of-the-art, real-time object detector, and Yolov5 is based on Yolov1-Yolov4. Continuous improvements have made it achieve top performances on two official object detection datasets: Pascal VOC (visual object classes) [32] and Microsoft COCO (common objects in context) [33]* (Xu et al., 2021). Wij hebben de keuze gemaakt om met verschillende YOLO versie aan de gang te gaan, die wij hier zullen toelichten.



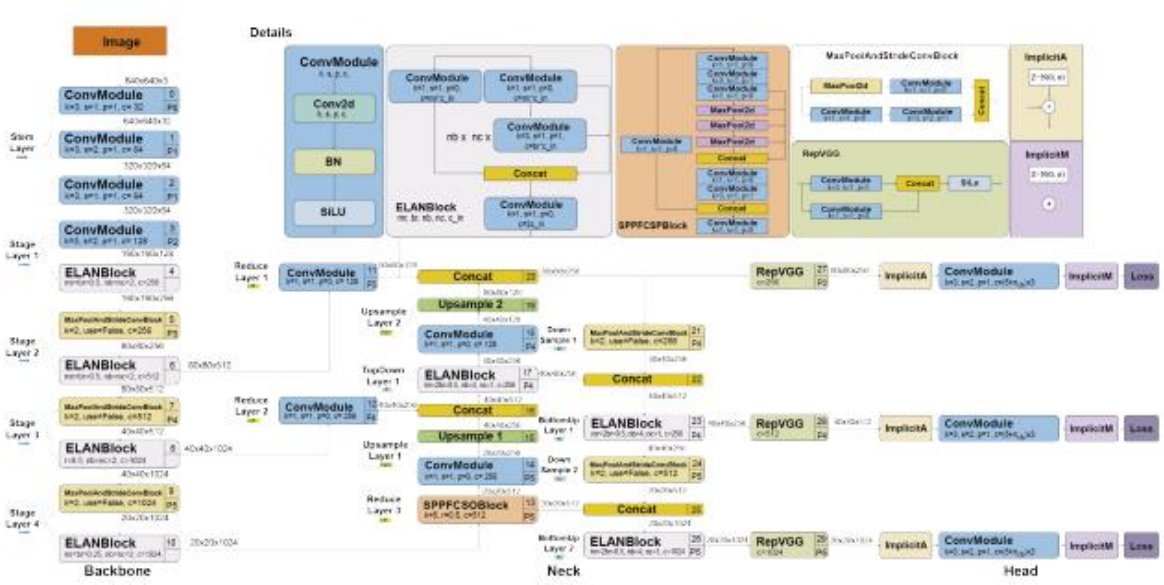
Figuur 4 Architectuur van YOLOv5 netwerk (Xu et al., 2021)

The network architecture of Yolov5. It consists of three parts: (1) Backbone: CSPDarknet, (2) Neck: PANet, and (3) Head: Yolo Layer. The data are first input to CSPDarknet for feature extraction, and then

fed to PANet for feature fusion. Finally, Yolo Layer outputs detection results (class, score, location, size). (Xu et al., 2021)

4.1.8. YOLOv7 Architectuur

Bij het onderbouwen van YOLOv7 zijn prestaties noemt Terven and Cordova-Esparza (2023) dat bij de uitkomst van het model, deze de snelste en het meest accuraat was in een range van 5 tot 160FPS. Dit zou moeten komen door de architectuur aanpassingen die zijn gemaakt in de YOLOv7 structuur zegt Terven and Cordova-Esparza (2023).



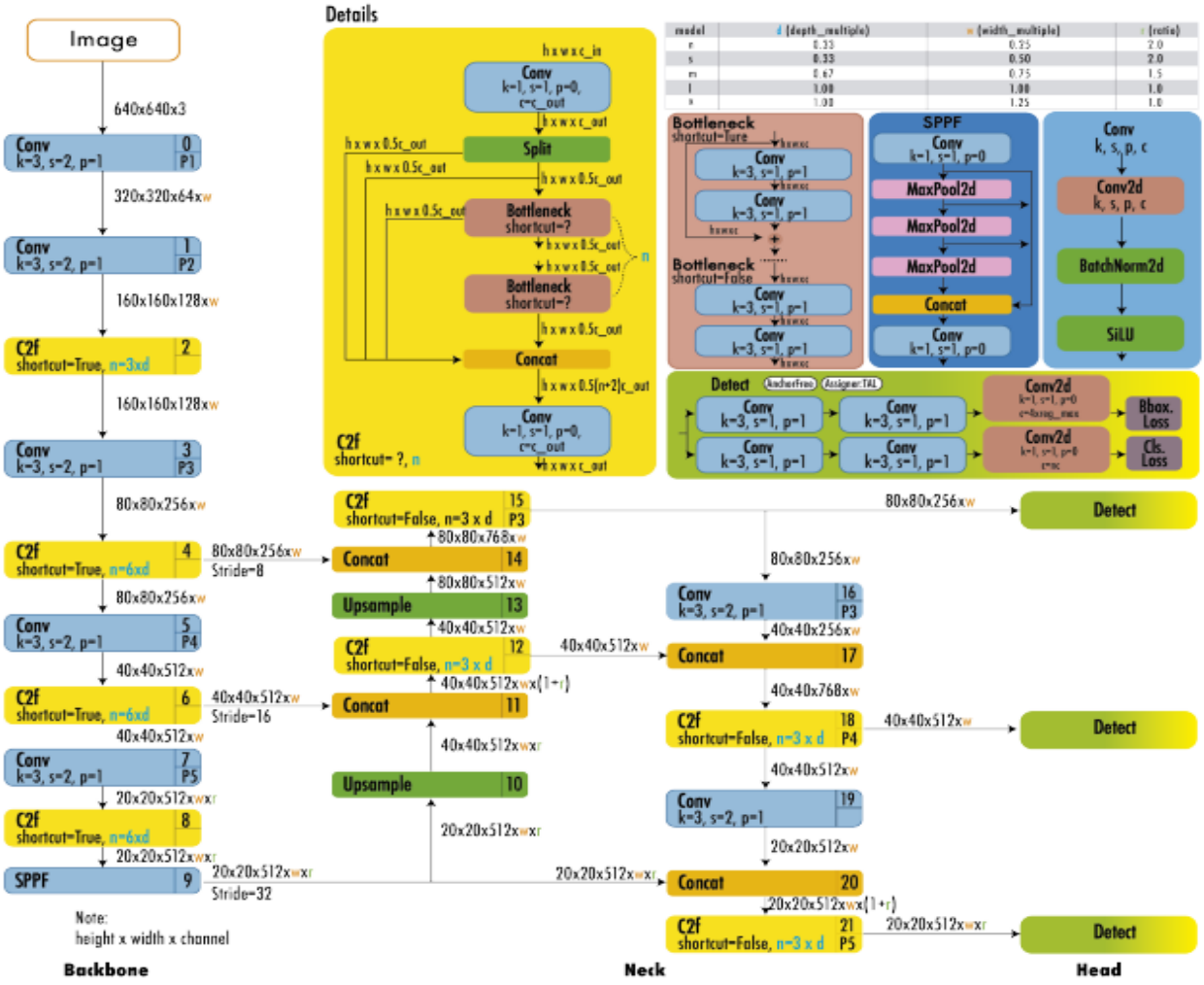
Figuur 5 YOLOv7 architectuur (Terven & Cordova-Esparza, 2023)

Changes in this architecture include the ELAN blocks that combine features of different groups by shuffling and merging cardinality to enhance the model learning and modified RepVGG without identity connection. (Terven & Cordova-Esparza, 2023)

4.1.9. YOLOv8 Architectuur

Vanwege het succes van de voorgangers van YOLOv8 willen wij er in dit onderzoek gebruik van maken.

YOLOv8 uses an anchor-free model with a decoupled head to independently process objectness, classification, and regression tasks. This design allows each branch to focus on its task and improves the model's overall accuracy. (Terven & Cordova-Esparza, 2023)



Figuur 6 YOLOv8 architectuur (Terven & Cordova-Esparza, 2023)

4.1.10. Pipeline

In ons onderzoeksrapport verwijst het woord "pipeline" naar een opeenvolging van stappen die worden genomen om gegevens vanuit verschillende bronnen te verwerken en uiteindelijk te visualiseren in QGIS. Deze stappen omvatten het gebruik van YOLOv7 en YOLOv8, het berekenen van oppervlakten, het koppelen van coördinaten, het opslaan van gegevens in een CSV-bestand en het importeren van het CSV-bestand in QGIS. De pipeline vertegenwoordigt de gestructureerde en sequentiële aanpak die we volgen om onze onderzoeksgegevens te analyseren en te presenteren.

4.1.11. mAP Score

De mAP-score (Mean Average Precision) bij objectdetectie verwijst naar de gemiddelde precisie over meerdere klassen en is een veelgebruikte evaluatiemeting om de prestaties van objectdetectiemodellen te beoordelen.

Om de mAP score te berekenen (Zie figuur 7 voor formule):

- *Bereken de precision en recall voor elk geretourneerd document in de resultaten. (zie sectie 4.1.5 voor precision & recall)*
- *Bereken de Average Precision (AP) voor die query door de precisiewaarden te berekenen op verschillende recall-niveaus en van deze het gemiddelde te pakken.*
- *Neem het gemiddelde van de AP-waarden over alle objectklassen om de mAP-score te verkrijgen. (Shah, 2023).*

$$\text{mAP} = \frac{1}{N} \sum_{i=1}^{N} \text{AP}_i$$

Figuur 7 Mean average precision formula (Shah, 2023).

De MAP-score is een enkele maatstaf om de prestaties van een objectdetectiemodel te beoordelen. Het maakt vergelijkingen tussen modellen mogelijk en geeft inzicht in hun effectiviteit bij het detecteren en lokaliseren van objecten in afbeeldingen.

Een hogere MAP-score geeft betere prestaties aan, waarbij een score van 1 perfecte detectie betekent.

4.1.12. Epoch

Een epoch in machine learning is een term die wordt gebruikt om één volledige doorloop van de trainingsdata te beschrijven tijdens het trainingsproces van een model. Het houdt in dat alle trainingsvoorbeelden één voor één worden doorgevoerd naar het model, waarbij het model de voorspellingen berekent, deze vergelijkt met de werkelijke uitvoer en de interne parameters (gewichten en biases) aanpast op basis van de fout. Dit proces van optimalisatie en aanpassing heeft als doel om het model te laten leren en betere voorspellingen te genereren.

Het aantal epochs dat wordt gebruikt bij het trainen van een model is een belangrijke factor. Het kiezen van het juiste aantal epochs is erg belangrijk om ervoor te zorgen dat het model voldoende leert van de trainingsdata, maar tegelijkertijd voorkomt dat het model te veel afhankelijk wordt van de specifieke voorbeelden in de trainingsset (overfitting) of niet genoeg leert om goede voorspellingen te kunnen doen (underfitting). Door het model meerdere epochs te laten doorlopen, krijgt het de kans om zijn prestaties te verbeteren door geleidelijk aan meer informatie uit de trainingsdata te halen.

5. Resultaten deelvragen

Dit hoofdstuk is bedoeld om het onderzoek naar onze hoofdvraag toe te lichten. Wij zullen om dit te doen door elke deelvraag heen lopen. De deelvragen worden opgedeeld in kleinere secties om het doelgericht te kunnen toelichten.

5.1. Hoe gaan wij de data labelen?

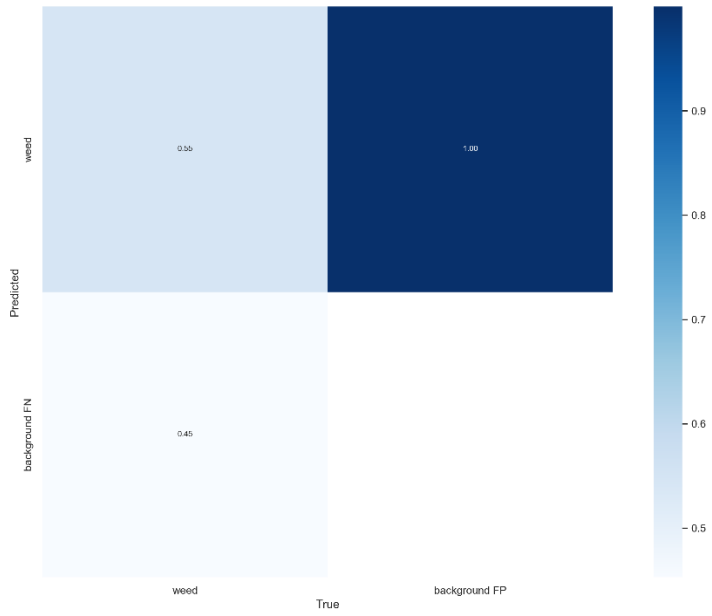
In deze sectie gaan we onderzoeken op wat voor manier we onze data moeten gaan labelen, we nemen een betere kijk op hoe onze dataset in elkaar steekt met betrekking tot bijvoorbeeld kwantiteit van het aantal gelabelde foto's of de kwaliteit van die gegeven labels.

> *When only using the Creeping thistle class, the best performance achieved in this thesis was a map of 0.33. When using four classes the best performance achieved in this thesis was a map of 0.07. (Ahlqvist, 2022)*

Wij zullen dieper in gaan op de boven aangegeven citaat, dit met de bedoeling om te achterhalen of dit voor onze dataset ook het geval zal zijn.

5.1.1. Hoe zijn de labels gekozen?

Wij zijn begonnen met de data annoteren met 1 label genaamd: weed. Om te kunnen achterhalen hoe goed dit zou presteren hebben wij een simpel model opgezet in YOLO voor object detection. Dit model geeft ons dan een confusion matrix. De bedoeling van het lezen van zo een confusion matrix is dat je kan zien welke classes het wel goed doen en welke niet (zie figuur 8).



Figuur 8 Confusion matrix 1 label

Uit de confusion matrix te zien in figuur 7 is nog niet heel veel op te maken. We hebben kunnen aantonen dat het voor over 50 procent lukt om van onze geannoteerde data de labels terug te vinden. Wat het moeilijk maakt is dat de vegetatie die zich bevind in onze dataset vrij divers is. Met dit en de eerder benoemde citaat in gedachten hebben wij er voor gekozen om de data op te delen in

verschillende labels. Deze labels zijn in overleg met de opdrachtgever tot stand gekomen dit zijn: bushlike, weed en vines.

- Bushlike: dit zijn stukken vegetatie die lijken op bosjes. (Zie figuur 9.1)
- Weed: dit zijn alle stukken vegetatie die bladeren of sprieten bevaten. (Zie figuur 9.2 & 9.3)
- Vines: dit zijn alle stukken vegetatie die meer takken hebben dan bladeren en uitgebreid groeien. (Zie figuur 9.4)

Figuur 9.1 Bushlike voorbeeld

Figuur 9.2 Weed voorbeeld (sprieten)

Figuur 9.3 Weed voorbeeld (blaadjes)

Figuur 9.4 Vine voorbeeld

De reden dat wij deze labels op deze manier gebruiken is ontstaan vanuit het argument dat hoewel bepaalde soorten vegetatie op elkaar kunnen lijken er tussen een aantal grote verschillen zijn (zie figuren 9.1-9.4). We als team hebben samen met de opdracht gekeken naar 1000+ verschillende foto's van het spoor waar vegetatie op de vinden was en een generalisatie gemaakt van de soorten vegetatie die wij zijn tegengekomen. Dit zodat ons model bij het trainen niet hoeft te differentiëren tussen soorten vegetatie die weinig tot niets verschillen in hun uiterlijk.

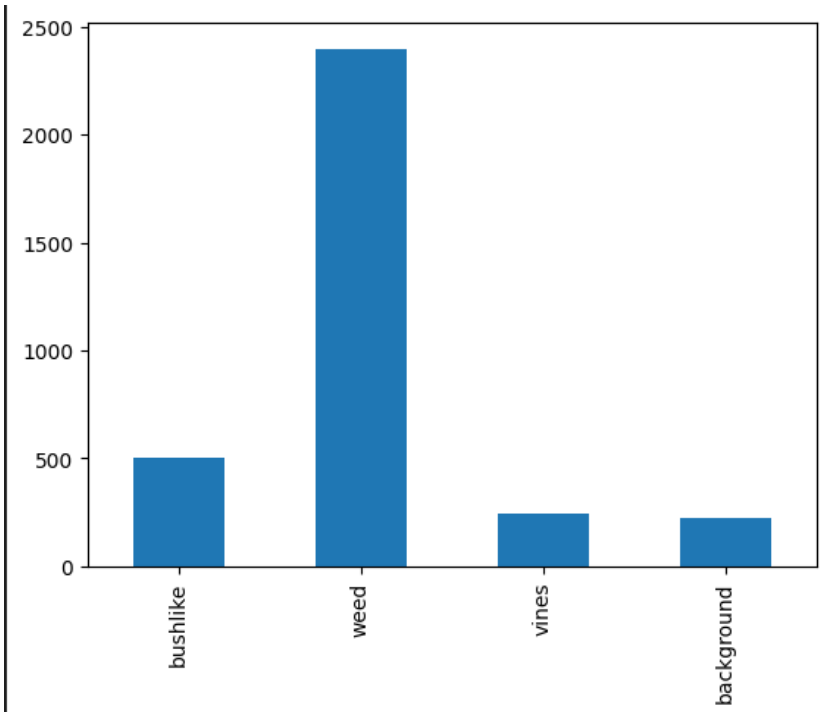
Nu dat wij twee datasets hebben gerealiseerd, één geannoteerd met 1 class en de ander geannoteerd met 3 verschillende classes. We hebben voor beide datasets weer een simpel YOLO model laten trainen, allebei met dezelfde parameters. De resultaten worden weergeven in tabel 1.

Class	Precision	Recall	mAP@0.5
Aantal classes: 1			
Weed	0.493	0.45	0.423
Average	**0.493**	**0.45**	**0.423**
Aantal classes: 3			
Bushlike	0.176	0.514	0.147
Weed	0.188	0.694	0.366

Vines	0.092	0.341	0.101
Gemiddeld	**0.152**	**0.516**	**0.204**

Tabel 1 resultaten YOLO model met 75 epochs

Wat opvalt is dat met het ophogen van de classes de mAP niet omhoog maar omlaag is gegaan, we kunnen hieruit concluderen dat het model als score overall zeker slechter scoort. Echter de recall van het model met de classes weed en bushlike is hoger dan het gemiddelde van een model met 1 class. Echter de class vine scoort enorm slecht. *Vegetatie dat klimop representeert zoals bijvoorbeeld de Japanse Duizendknoop, zijn moeilijk te detecteren door de enorme variatie in groei. (Jesse Asset Insight, 2023).* Uit onderzoek vanuit de opdrachtgever is in een voorgaand project voor het goed detecteren van vines een dataset van 7000+ foto's opgezet. Bij het creëren van onze dataset hebben wij niet de mogelijkheid gehad om een grote hoeveelheid foto's van deze class te verzamelen. (Zie figuur 10)



Figuur 10 Y-as: aantal instances per class, X-as: de class

5.1.2. Deelconlcusie

Met het onderzoek dat wij hebben uitgevoerd kunnen wij concluderen dat in het algemeen een model om vegetatie te detecteren het gemiddelde beter doet, echter in ons geval willen wij een hoge recall hebben, dit word verder toegelicht in sectie 5.5. Met dit in gedachten kiezen wij er voor om meerdere classes te gebruiken als eerst voor de hogere recall, maar ook om door het project heen beter duidelijkheid te hebben in welke soorten vegetatie classen het minder goed doen dan andere.

5.2. Hoeveel gelabelde data moeten wij minimaal hebben?

Aan de grond van elk computer vision programma of model ligt altijd data. In deze sectie gaan we bespreken hoeveel data wij nodig hebben op het model optimaal te kunnen laten werken. Het is namelijk tot op zekere aantallen het geval dat hoe meer data je hebt, hoe beter het model kan worden.

> *Most of the time, if your dataset is sufficiently large and well labelled, good results can be obtained with no changes to the models or training settings (Ultralytics, n.d.).*

Er is gelukkig documentatie van onze modellen die precies beschrijft wat een aan te raden hoeveelheid data is. In de gids over het model wordt aangeraden om per klasse minstens 1500 afbeeldingen te hebben. Dit zou in ons geval betekenen rond de 6000 afbeeldingen met de klasses “bushlike”, “weed”, “vines” en “background”. Ook wordt er aangeraden om per klasse minstens 10000 instanties te hebben. Dit zou met het minimale aantal afbeeldingen van 1500 per klasse betekenen dat elke afbeelding gemiddeld tussen de 6 en 7 instanties van die klasse heeft. In de praktijk is dit zelden het geval, en ligt het gemiddelde meer rond de 2 instanties van een klasse per afbeelding. Dit zou betekenen dat het aangeraden aantal afbeelding voor ons neer komt op 5000 per klasse, oftewel 20000 afbeeldingen in totaal.

> *Images per class. ≥ 1500 images per class recommended. Instances per class. ≥ 10000 instances (labeled objects) per class recommended (Ultralytics, n.d.)*

5.2.1. Hoe zien andere datasets eruit?

Om een goed beeld te krijgen van hoe onze dataset er uit zo kunnen zien, en hoe een goede dataset er in het algemeen uit ziet bij een image recognition project, kunnen we kijken naar vergelijkbare projecten en hun data. Dit hoeven niet per sé projecten te zijn die ook over vegetatiedetectie gaan, maar dit mag natuurlijk wel. Voor zelfs vergelijkbaar simpele modellen worden enorme hoeveelheden data gebruikt. Bijvoorbeeld bij een model wat katten en honden moet onderscheiden. *We selected total of nearly 24,000 images of cats and dogs and also having rotations and scaling of many images as each scaled image is a new image for the model as input. (Patil, 2021).* Zoals beschreven wordt er bij dat model ook aangepaste versies van een oorspronkelijke foto gegeven aan het model, om zo te helpen met trainen. Op die manier wordt het aantal afbeeldingen en dus ook de bruikbare data, enorm vergroot. Dit kan echter niet bij elk project worden toegepast.

5.2.2. Deelconclusie

Uit onderzoek vallen meerdere richtlijnen te trekken over wat een gewenste hoeveelheid data is, en uit wat voor bestanden deze data het beste kan bestaan. Volgens de uitvoerder van de YOLO modellen zijn er in ons geval minstens rond de 20000 afbeeldingen vereist. Hierbij gaan we er van uit dat op elke afbeelding gemiddeld 2 instanties van een klasse voor komen. Dit is echter zo bij gelijke verdeling van de klasses. In figuur 7 uit sectie 5.1.1 blijkt echter dat de minst voorkomende klasse, de background klasse, slechts in ongeveer 5% van de afbeeldingen voorkomt. Ook kan een background klasse maar 1x per afbeelding voorkomen. Omdat je dit van te voren niet kan bepalen is op die manier een ideaal aantal afbeeldingen te bepalen. Als voor elke klasse 10000 afbeeldingen aanwezig moet zijn, en

background images op ongeveer 5% van de afbeeldingen voorkomt, is een aan te raden hoeveelheid afbeeldingen 10000 / 0.05 = 200.000 afbeeldingen. Dit is echter niet reëel en blijkt ook niet nodig te zijn voor ons model. In een vergelijkbaar project van over vegetation detection wordt gewerkt met in totaal 5500 gelabelde instanties: *Roughly 5 500 bounding box annotations were used for training, validation and evaluation in this thesis. (Ahlqvist, 2022).* Deze hoeveelheid komt ook veel meer in de buurt van het aantal instanties dat wij hebben gelabeld. Voor ons model is het type van de data niet van groot belang, zolang de afbeeldingen maar een redelijke resolutie hebben. Als het namelijk met het blote oog niet goed te zien is, dan zal ons model niet veel beter presteren.

5.3. Welke computer vision technieken kunnen wij gebruiken?

Bij het krijgen van de opdracht zijn wij na gegaan wat er precies moet gebeuren om een resultaat in stand te brengen. Wij zijn al snel tot de conclusie gekomen dat YOLO object detection, dit omdat het er al eerdere project succesvol zijn afgerond met als onderwerp weed detection in voornamelijk YOLO. Zie citaten hieronder.

> *In addition, YOLOv3, YOLOv5, and Faster R-CNN were employed to train weed identification models using the Weed25 dataset. The average precision was 91.8%, 92.4%, and 92.15% respectively (P. Wang et al., 2022b)*

> *An improved YOLOv5 network named the TIA-YOLOv5 was developed for weed and crop objection detection. In the backbone of the TIA-YOLOv5, a transformer encoder block was used to improve the sensitivity to small weed objects. (A. Wang et al., 2022)*

5.3.1. Welke YOLO(s) hebben wij gekozen?

We zijn begonnen met het onderzoeken van de modellen YOLOv5 en Faster R-CNN. Al snel kwamen we erachter dat YOLOv5 een betere keuze voor ons is dan Faster R-CNN. Dit komt doordat YOLOv5 een lichter programma is en minder belastend is voor onze CPU en GPU tijdens het trainingsproces. Hoewel dit als een goede reden werd beschouwd om Faster R-CNN te schrappen, besloten we verder onderzoek te doen om te kijken of het misschien toch de moeite waard is om met Faster R-CNN door te gaan. Toch kwamen we al snel op meerdere bronnen die ons het tegendeel bewees. *"The final comparison between the two models reveals that the small YOLO v5 model runs about 2.5 times faster while managing better performance in detecting smaller objects. Ultralytics has done a fantastic job on their YOLO v5 open sourcing a model that is easy to train and run inference on"* (Dwivedi, 2021). Bovenop het feit dat onze computers Faster R-CNN niet goed aankon, presteert het slechter dan het lichtere model YOLOv5.

Na YOLOv5 boven Faster R-CNN gekozen te hebben, zijn we doorgegaan met ons onderzoek om te kijken of er nog nieuwere en betere modellen zijn. Na een tijd onderzoek te doen zijn we uitgekomen op YOLOv7. We zijn uiteindelijk overtuigd om YOLOv7 te gebruiken door een bron die we tijdens ons onderzoek hebben gevonden. *YOLOv5, YOLO-X, YOLO-R and YOLOv7 models were applied to a soccer game play video to track the moving ball and the players in the playground. All the models performed well however YOLOv7 proved to be state of the art in terms of the accuracy showing 58% AP whereas YOLOR-p6 which comes right after the former model and shows 56% AP.* (Gillani et al., 2022). Wij hebben geconcludeerd dat in ons project accuraatheid belangrijk is, vooral voor de eerste filter, object detection. Overigens is door Gillani et al. (2022) benoemd dat hoewel YOLOv7 beter scoort op accuraatheid, hij minder goed scoort op snelheid. Wij hebben er voor gekozen dat dit in ons geval geen probleem moet zijn.

Nu YOLOv7 is gekozen voor objectdetectie, werd er gekeken naar extra modellen voor semantische segmentatie. Hier kwam het recent uitgebrachte YOLOv8. Dit is een nauwkeuriger model dan YOLOv7, maar een groot nadeel is dat het 2 keer zo sloom traint.

The OP took about 5 hours to complete 300 epochs, whereas my custom YOLOv8 model took 3.4 hours to finish 100 epochs (Jain, 2023).

Uit ons bron bleek dat YOLOv7 twee keer zo snel kan trainen als YOLOv8. Hoewel 2 keer zo langzaam een groot nadeel is hebben we een slimme toepassing kunnen vinden voor YOLOv8 en een 2-phase model kunnen maken. Verdere uitleg hierover in ons deelconclusie.

5.3.2. Deelconclusie

Simpelweg hebben we voor elk model gekeken naar de voor- en nadelen en zijn we uitgekomen op een '2-phase model'. Het uitgevoerde idee is om YOLOv7, het licht minder nauwkeurig maar snellere model, te gebruiken om onze dataset te 'filteren' en zoveel mogelijk frames met vegetatie door te geven aan YOLOv8. Hierna wordt YOLOv8 gebruikt voor het belangrijke gedeelte van semantische segmentatie. Door het filteren krijgt het model een aanzienlijk minder aantal frames en heeft het langzamer eigenschap van het model minder consequenties. Op deze word er zo efficiënt mogelijk gebruik gemaakt van de sterke punten van de twee gekozen modellen.

Hier onder hebben we nog een SFA-matrix van afwegingen die wij hebben meegenomen in ons proces.

Model	**Architectuur**	**Trainingssnelheid**	**Nauwkeurigheid**	**Bruikbaarheid**	**TOTAAL**
Faster R-CNN	Region-based	1	2	2	5
Yolov5	One-stage	4	3	3	10
Yolov7	One-stage	4	4	3	11
Yolov8	One-stage	2	5	4	11

5.4. Op welke manier bepalen wij de volume per 10m2?

We hebben één van de vereisten van Asset Insight gekregen om het volume van vegetatie op het spoor te berekenen. Deze eis is ontstaan met de bedoeling om het zo goed te kunnen visualiseren in het QGis. Deze keuze is ook gemaakt met in gedachte dat niet overal op het spoor vegetatie te vinden is, zo moet er een generalisatie gemaakt worden binnen een bepaalde omgeving van de hoeveelheid vegetatie die zich daar bevindt.

Om een volume van 10m² te bepalen, maken we gebruik van twee verschillende camera's. Deze camera's bevinden zich aan de linker- en rechterzijde van de trein en bieden een bovenaanzicht. We gebruiken de term 'camera 0' om te verwijzen naar het weergegeven perspectief van links, en 'camera 1' voor het perspectief van rechts. (Zie figuur 11.1 & 11.2)

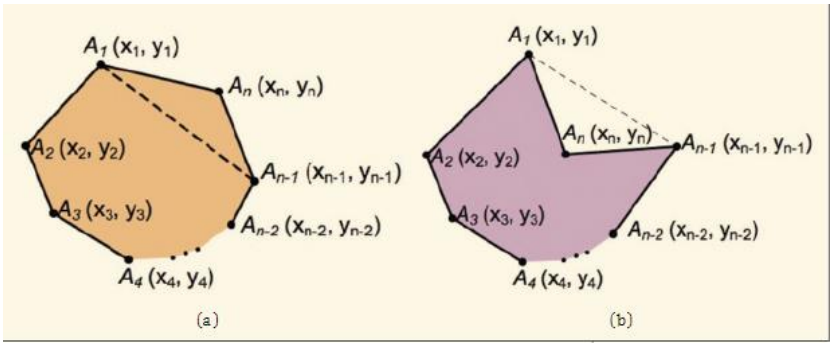
Figuur 11.1 Camera 0

Figuur 11.2 Camera 1

5.4.1. Berekening oppervlakte van gevonden vegetatie?

We hebben besloten om gebruik te maken van semantic segmentation om het volume van vegetatie te vinden. Voor dit doel hebben we het YOLOv8-model gekozen, dat semantic segmentation uitvoert en een polygoon creëert rondom de gedetecteerde vegetatie. Om de oppervlakte van de polygoon te berekenen, maken we gebruik van de shoelace-formule (zie figuur 9), zoals hieronder weergegeven. In de formule vertegenwoordigen de X- en Y-coördinaten de hoekpunten van de polygoon op een frame.

*Shoelace-formule: Area = 1/2 * |(x1 * y2 + x2 * y3 + ... + xn-1 * yn + xn * y1) - (y1 * x2 + y2 * x3 + ... + yn-1 * xn + yn * x1)|*



Figuur 9 Shoelace formula

(Lee & Lim, 2017)

5.4.2. Deelconclusie

Om een volledig beeld te krijgen, hebben we twee frames nodig van elk perspectief. Het is essentieel dat deze frames zich op dezelfde locatie bevinden. Een van de vereisten voor het berekenen van het volume van vegetatie op het spoor is het kennen van de oppervlakte van een enkel frame. Om dit te benaderen, hebben we gekeken naar de breedte van een spoor in Nederland, die volgens onze opdrachtgever 1.435 meter is (Asset Insight, 2023). Door twee frames naast elkaar te plaatsen, krijgen we dus een volledige foto van het spoor met een geschatte oppervlakte van ongeveer 1,5 vierkante meter.

Om een volume van 10m² te bereiken, hebben we zeven paar frames nodig en moeten we de oppervlakte van de resulterende polygonen onthouden. Dit wordt bereikt door 7 keer 1,5m² op te tellen, wat resulteert in 10m². We hebben het volume van het object berekend door de oppervlakte van deze zeven duo-frames te gebruiken. Hiervoor hebben we de shoelace-formule toegepast om de polygonen te berekenen. Het uiteindelijke resultaat geeft het volume van vegetatie per 10m² weer.

5.5. Hoe meten wij de accuraatheid van ons gebruikte model?

In deze sectie gaan wij het hebben over het meten en/of scoren van onze modellen. Zoals in sectie 5.3 is uitgelicht, maken wij in dit project gebruik van twee verschillende YOLO modellen. Bij het trainen van deze modellen komen een aantal zogeheten `metrics` naar boven, deze metrics helpen ons bij het begrijpen en onderbouwen waarom een model wel of niet goed werkt.

In ons theoretisch kader hebben wij een aantal metrics toegelicht, deze gaan wij in dit hoofdstuk gebruiken om te achterhalen hoe accuraat onze modellen zijn. De metrics zijn: precision, recall & confusion matrix.

5.5.1. YoloV7

Wij beginnen onze pipeline met object detection, met een Yolo V7 model. In dit model is de belangrijkste eis dat het zoveel mogelijk kan detecteren in een gegeven video. Dit betekent dat recall in dit model voor ons zwaar mee telt, precision telt minder mee in het afwegen of het model goed presteert.

Als laatst hebben wij elke frame in één gegeven video bekeken om zo te kunnen achterhalen welke en hoeveel frames er terug gegeven moeten worden door het model. We bespreken de uiteindelijke resultaten van het model in sectie 5.5.2

5.5.2. Deelconclusie

Wij hebben in dit hoofdstuk in kaart gebracht welke metrics wij opmeten, hoe we dit doen en waarom. In de deelconclusie willen wij de resultaten laten zien die zijn gekomen uit het trainen van de modellen. Zoals eerder is aangekaart hebben wij een video frame voor frame bekeken zodat wij konden bepalen hoeveel frames ons model terug zou moeten krijgen (zie tabel 2).

Type	Aantal	Vegetatie	Background
Met de hand	124	124	0
Yolo V7	146	114	32

Tabel 2 resultaat yolo v7

Ondanks dat er nog een aantal background foto's naar voren komen, lijkt het erop dat ons model het goed doet, we kunnen hiermee aantonen door middel van een rekensom (100% / 124 * 114) aantonen wat onze accuraatheid is in percentage. Het komt voor deze video terecht op +91%. Als laatst willen wij ook aantonen wat onze confusion matrix terug geeft op een test set waar het model niet op getraind of gevalideerd is (zie figuur 10). In deze confusion matrix kunnen we aan de tinten van blauw zien hoe goed een bepaalde class het doet. Omdat de classes bushlike, weed en vines als 1 class gezien mogen worden gaan we deze samen voegen tot één geheel (zie tabel 3).

Class	TP	FP	FN
Weed	0.80666666666	0.19666666666	0.33

Tabel 3 confusion matrix samengevoegd

Na het trainen van het model hebben we nu aan kunnen tonen dat het werkt zoals het zou moeten werken en grotendeels van wat hij zou moeten vinden ook kan terug vinden.

Figuur 10 confusion matrix yolo v7

5.6. Wat moet het model terug geven als resultaat?

Aan het eind van ons model krijgen wij een dataframe terug van alle plekken met begroeiing met daarin de volgende data: De file locatie (oftewel het frame in kwestie), het framenummer, de camera van het frame en het percentage van het frame waar begroeiing aanwezig is. Verder hebben we de bestand met de fotolocaties van elk frame gegeven in RD coördinaten. Deze coördinaten kunnen in QGIS worden ingevoerd om zo een visualisatie te geven van waar in Nederland wel of geen begroeiing aanwezig is. QGIS is een programma waar allerlei soorten kaarten kunnen worden ingeladen, en met hoge nauwkeurigheid een plek aangegeven kan worden aan de hand van coördinaten. Wij hebben RD coördinaten, maar in QGIS kunnen ook bijvoorbeeld latitude - longitude coördinaten worden gebruikt, als dat soort data aanwezig was geweest.

6. Discussie

In de discussie gaan we het hebben over de tekortkomingen van het project. Dit kunnen tekortkomingen zijn op het gebied van de hardware, de data enz. Door terug te kijken op deze tekortkomingen kunnen we bij een volgend vergelijkbaar project hier op tijd rekening mee houden. Ook kunnen we enkele van deze tekortkomingen verwerken in het advies aan de opdrachtgever, om hen te helpen met het verder gebruiken van ons model.

6.1.1. Data Labelen

Hoewel we achteraf tevreden zijn met onze hoeveelheid data, was het wel erg veel werk om allemaal te labelen. Vooral het uitlijnen van de planten was duurde erg lang. Wij hebben het labelen van de data verdeeld onder alle teamleden. Dit is goed gelukt, maar het is erg belangrijk om goed af te spreken hoe alles gelabeld moet worden. Er mag geen verschil zitten in hoe verschillende leden een plant uitlijnen, en ook niet in de classificatie labels. Er moet dus goed afgesproken worden hoe elke plant gelabeld wordt, zodat dezelfde plantsoort niet door een iemand gelabeld wordt als “weed” en door iemand anders als “vines”. Als dit goed geregeld wordt is het verdelen van de data bij het labelen geen probleem. Ook is het een optie om het labelen te outsourcen. Ook dan moet erg goed worden afgesproken wat je verwacht.

6.1.2. Grafische Kaart

Voor het trainen van het model was een sterke grafische kaart nodig die tevens van Nvidia moest zijn. Omdat wij werkten met onze eigen laptops en desktop pc’s kon niet iedereen de modellen even makkelijk trainen. Als de grafische kaart niet goed genoeg was dan ging het trainen van het model een stuk langzamer om soms helemaal niet. Hierdoor was het soms lastig om ergens op locatie te werken, vooral omdat de apparaten waar uiteindelijk op getraind zijn allemaal desktop pc’s zijn. We hadden er wel eerder achter kunnen komen dat er een sterke Nvidia kaart nodig bij een beter vooronderzoek over de modellen. Dit had echter de modelkeuze niet beïnvloed. Wel zijn we er achter gekomen dat dit een goede reden is voor vroegtijdig model onderzoek, om zo eventueel tijd te hebben om benodigde hardware te regelen via bijvoorbeeld de opdrachtgever.

6.1.3. Dataset vergroten

Bij het bespreken van de verschillende labels is benoemd dat enkele labels slechter scoren dan anderen. Er werd genoemd dat dit zo was omdat deze types planten moeilijker te detecteren zijn. Ook was het zo dat deze types planten een stuk minder voorkwamen dan de grootste klasse “weed”. Wat toegepast had kunnen worden is het spiegelen van data om zo extra frames te krijgen met de labels “bushlike” en “vines”. Hoewel vanuit onze opdrachtgever al was teruggekomen dat het simpelweg een stuk moeilijker is om deze planten te indentificeren, hadden we kunnen controleren of deze toename aan data wellicht voor een beter model had kunnen zorgen.

7. Conclusie

In deze sectie nemen wij een terugblik op hoe ons project is verlopen en uiteraard wat voor ons de uiteindelijke conclusie was die we hebben kunnen nemen op onze onderzoeksvraag. Wij hebben in dit onderzoeksrapport een aantal problemen in kaart gebracht om de onderzoeksvragen te kunnen beantwoorden.

Dit zijn de deelvragen die wij hebben uitgelicht in hoofdstuk 5. We gaan in dit hoofdstuk door de deelconclusies van elke deelvraag heen lopen om zo een beeld te kunnen vormen voor het antwoord op onze onderzoeksvraag.

1. Hoe gaan wij de data labelen?
2. Hoeveel gelabelde data moeten wij minimaal hebben?
3. Welke computer vision technieken kunnen wij gebruiken?
4. Op welke manier bepalen wij de volume per 10m2?
5. Hoe meten wij de accuraatheid van ons gebruikte model?
6. Wat moet het model terug geven als resultaat?

Hierboven staan de deelvragen nog een keer uitgelicht, we zullen de deelvragen 1, 2, 3, 4 en 6 bespreken in onze conclusie.

7.1. Data

We hebben in dit project 2 deelvragen die ons helpen bij het opzetten van de dataset. We hebben in dit onderzoek kunnen achterhalen wat voor ons doeleinde een complete dataset is. Het is belangrijk voor een computer vision project zoals deze om vooral voldoende data te kunnen vergaren, voor ons voldoen wij aan de hoeveelheid als er per class een minimum van 500 instanties aanwezig zijn. Met dit aantal zijn wij namelijk in staat om computer vision modellen zodanig te trainen data zij resultaat terug kunnen geven.

Naast de aantallen is het ook belangrijk om ervoor te zorgen dat de geannoteerde data consistent is met elkaar, dat een label niet meer dan 50% background heeft en dat de label het object goed ingesloten. Als laatst is het voor een model handig om te weten wat hij niet moet detecteren, hier komen background foto’s voor in het spel, deze moeten in aantallen zijn van 1-10% van de grote van de uiteindelijke dataset.

Dit alles helpt ons bij het realiseren van een dataset, die bruikbaar is voor ons doeleinde.

7.2. Computer Vision

In dit stuk brengen we de conclusie van deelvragen 3 & 4 in kaart, deze gaan specifiek over de technische oplossing van ons probleem.

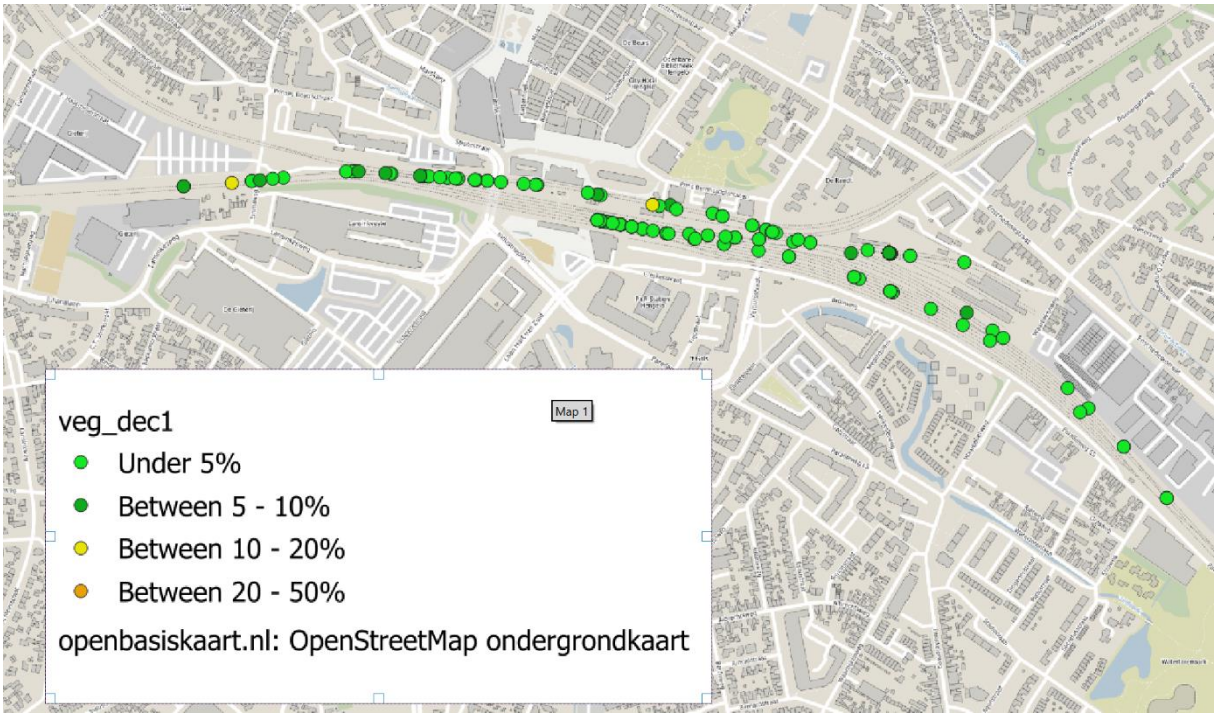
Wij hebben in dit onderzoek kunnen achterhalen dat door middel van het gebruiken van de computer vision technieken: object detection en semantic segmentation, je zodanig vegetatie kan herkennen dat je met deze resultaten ook weer nieuwe informatie kan aanmaken. Dit is exact wat wij ook doen, wij gebruiken Yolo V7 zijn object detectie om zo een verzameling van frames die een potentie van vegetatie heeft te maken, deze word dan doorgevoerd aan Yolo V8 zijn semantic segmentation.

Yolo V8 heeft het daarbij voor ons mogelijk gemaakt om terug te halen hoeveel procent een frame word bedekt door vegetatie, dit noemen wij de volume.

7.3. Resultaat

Deze sectie geeft licht aan wat voor uiteindelijk resultaat ons is dat zich vormt uit onze modellen. Het gaat hier om een tabel dat word gegenereerd na de semantic segmentation van Yolo V8. In deze tabel bevinden zich een aantal kolommen, waarvan de belangrijkste zijn: frame nummer, camera nummer, xy coordinaten & percentage vegetatie coverage. Deze tabel kunnen wij exporteren en inladen in QGis, deze tabel is daarbij ook het antwoord op deel vraag 6.

Nu dat we dit hebben kunnen achterhalen, krijgen wij in QGis een resultaat te zien zoals word weergegeven in figuur 11. Door het aflezen van de kleur waardes die elk zo een puntje bevat kan er een uiteindelijk conclusie worden getrokken door Asset Insight of de vegetatie op dat punt in het spoor zodanig aanwezig is, dat het schadelijk is voor het spoor.



Figuur 11 Qgis punten met gevonden vegetatie

7.4. Hoofdvraag

Als laatst willen wij terug komen op onze hoofdvraag. Door het bespreken en beantwoorden van alle deelvragen kunnen wij hier nu een definitief antwoord op geven.

Welke computer vision techniek(en) kunnen wij het beste toepassen bij het via foto's herkennen en meten van schadelijke begroeiing op en rondom de Nederlandse spoorwegen?

Met alle informatie dat wij hebben vergaard in ons onderzoek, kunnen wij concluderen dat met behulp van de computer vision technieken object detection en semantic segmentation het mogelijk is om aan te kunnen tonen waar op het nederlandse spoor zich schadelijke begroeiing bevindt.

8. Advies

8.1. Lidar-data gebruiken voor hoogtebepaling

Een uitdaging bij het gebruik van alleen beeldgebaseerde detectie is het accuraat bepalen van de hoogte van de vegetatie. Door lidar-data te integreren in het detectieproces en deze te clippen, kan je een beter begrip krijgen van de driedimensionale structuur van de vegetatie en de hoogte hiervan bepalen.

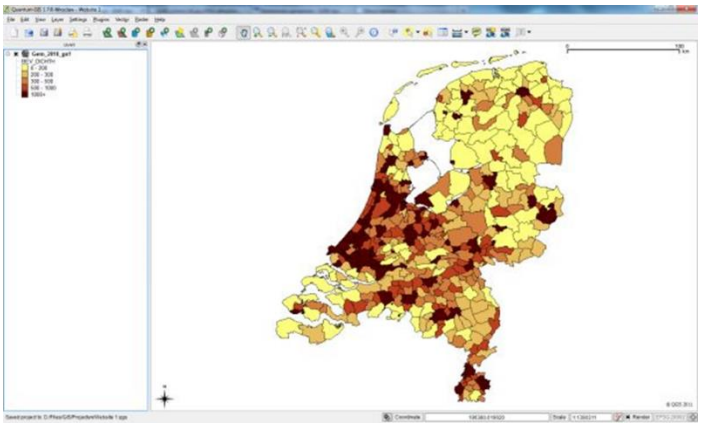
8.2. Meerdere classes aanhouden voor een beter model

Het handhaven van de verschillende klassen, zoals "weed", "vines", "bushlike" en "background", in het model is essentieel voor het verkrijgen van betere resultaten. Door deze specifieke klassen te behouden, kan het model zich richten op het herkennen en classificeren van de verschillende soorten vegetatie op de treinrails. Dit zorgt voor een meer gerichte detectie en analyse, waardoor het model in staat zijn om specifieke problematische vegetatietypes te identificeren en passende maatregelen te nemen.

Door te focussen op specifieke klassen wordt het model ook minder breed in het herkennen van objecten. Het kan zich concentreren op de specifieke kenmerken en eigenschappen van de vegetatieklassen die van belang zijn. Dit resulteert in een nauwkeurigere detectie en maakt het mogelijk om de vegetatie op een meer gedetailleerd niveau te analyseren.

8.3. Visualiseren locaties

Zoals eerder benoemd is één van de outputs in ons model de RD-coördinaten van de frames. Hieronder volgt een uitleg over hoe deze coördinaten te visualiseren zijn in het programma QGIS. Allereerst: waarom QGIS. QGIS is bij lange na niet het enige programma dat kan worden gebruikt om RD-coördinaten te vinden en te visualiseren. Ook bijvoorbeeld Google Earth en Maps zijn hier prima toe in staat. Het is ook een prima oplossing om deze programma's te gebruiken om snel een positie te vinden op de kaart. Een van de eerste en sommige situaties belangrijkste voordelen van QGIS is dat het veel simpeler is om een reeks getallen tegelijkertijd weer te geven. Zo krijg je een veel beter inzicht op bijvoorbeeld de afstand tussen twee aangrenzende frames of de grootte van een gebied aan frames. Verder is het mogelijk om verschillende soorten kaarten weer te geven in QGIS, denk hierbij aan bijvoorbeeld de gemeentes of waar de spoorwegen duidelijk op zichtbaar zijn. Al deze kaarten zijn gratis beschikbaar, en er zijn zelfs extra kaarten te downloaden door te zoeken naar GIS kaarten.



Figuur 12 QGIS met kaar

9. Bronnen

1. Ultralytics. (n.d.). *how to use Background images in training? · Issue #2844 · ultralytics/yolov5*. GitHub. https://github.com/ultralytics/yolov5/issues/2844#issuecomment-851338384
2. Yella, S., Nyberg, R. G., Payvar, B., Dougherty, M., & Gupta, N. M. (2013). Machine Vision Approach for Automating Vegetation Detection on Railway Tracks. *Journal of Intelligent Systems*, *22*(2), 179–196. https://doi.org/10.1515/jisys-2013-0017
3. *The 10 Best Public Datasets for Object Detection in 2022 | Blog | Scale AI*. (n.d.). ScaleAI. https://scale.com/blog/best-10-public-datasets-object-detection
4. *What is Computer Vision? | IBM*. (n.d.). What Is Computer Vision? | IBM. Retrieved June 3, 2023, from https://www.ibm.com/topics/computer-vision
5. *What Is Object Detection?* (n.d.). MATLAB & Simulink. Retrieved June 3, 2023, from https://nl.mathworks.com/discovery/object-detection.html
6. Jordan, J. (2023). An overview of semantic image segmentation. *Jeremy Jordan*. https://www.jeremyjordan.me/semantic-segmentation/
7. Wang, P., Tang, Y., Luo, F., Wang, L., Li, C., Niu, Q., & Li, H. (2022). Weed25: A deep learning dataset for weed identification. *Frontiers in Plant Science*, *13*. https://doi.org/10.3389/fpls.2022.1053329
8. Ahlqvist, A. A. (2022). Examining Difficulties in Weed Detection. *Master of Science Thesis in Electrical Engineering*.
9. Wang, A., Peng, T., Cao, H., Xu, Y., Wei, X., & Cui, B. (2022). TIA-YOLOv5: An improved YOLOv5 network for real-time detection of crop and weed in the field. *Frontiers in Plant Science*, *13*. https://doi.org/10.3389/fpls.2022.1091655
10. Narkhede, S. (2021, June 15). Understanding Confusion Matrix - Towards Data Science. Medium. https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62

11. Shah, D. (2023, April 24). Mean Average Precision (mAP) Explained: Everything You Need to Know. V7. https://www.v7labs.com/blog/mean-average-precision#h1
12. Patil, A. N. (2021). IMAGE RECOGNITION USING MACHINE LEARNING. International Journal of Engineering Applied Science and Technology, 6(1). https://doi.org/10.33564/ijeast.2021.v06i01.027
13. Kundu, R. (2023, May 11). Precision vs. Recall: Differences, Use Cases & Evaluation. *V7*. https://www.v7labs.com/blog/precision-vs-recall-guide
14. Lee, Y., & Lim, W. (2017). Shoelace Formula: Connecting the Area of a Polygon and the Vector Cross Product. *Mathematics Teacher: Learning and Teaching PK–12*, *110*(8), 631–636. https://doi.org/10.5951/mathteacher.110.8.0631
15. Dwivedi, P. (2021, December 15). YOLOv5 compared to Faster RCNN. Who wins? - Towards Data Science. *Medium*. https://towardsdatascience.com/yolov5-compared-to-faster-rcnn-who-wins-a771cd6c9fb4
16. Gillani, I. S., Munawar, M., Talha, M., Azhar, S., Mashkoor, Y., Uddin, M. S., & Zafar, U. (2022). Yolov5, Yolo-x, Yolo-r, Yolov7 Performance Comparison: A Survey. *Computer Science & Information Technology (CS & IT)*. https://doi.org/10.5121/csit.2022.121602
17. *How to annotate inside polygons(holes) in semantic segmentation*. (n.d.). https://discuss.pytorch.org/t/how-to-annotate-inside-polygons-holes-in-semantic-segmentation/90953. https://discuss.pytorch.org/uploads/default/original/3X/4/8/487c7e3be814e4e8beeb7106808ea9e110b31a5b.png
18. Xu, R., Lin, H., Lu, K., Cao, L., & Liu, Y. (2021). A forest fire detection system based on ensemble learning. *Forests*, *12*(2), 217. https://doi.org/10.3390/f12020217

19. Terven, J. R., & Cordova-Esparza, D. M. (2023). A COMPREHENSIVE REVIEW OF YOLO: FROM YOLOV1 AND BEYOND. In *UNDER REVIEW IN ACM COMPUTING SURVEYS*.