THOMAS MORE

BACHELORPROEF

Realtime detectie van tanks op basis van infraroodbeelden

PROFESSIONELE BACHELOR IN HET INFORMATIEMANAGEMENT EN DE MULTIMEDIA

Opleiding Informatie, Management & Security (IMS)

Keuzetraject: **Data Analytics**

Begeleider: **Dr. Charlie Beirnaert**

Lukas Van Den Bosch

Campus De Vest, Zandpoortvest 60, BE-2800 Mechelen

2024 - 2025

Inhoudsopgave

Inleiding

Het doel van deze bachelorproef is om de mogelijkheden van kunstmatige intelligentie te onderzoeken voor de detectie en classificatie van militaire voertuigen, meer bepaald tanks, op basis van hun warmtesignaturen. De detectie van militaire voertuigen via warmtebeelden vormt een unieke uitdaging binnen het bredere domein van objectdetectie, waarbij zowel de nauwkeurigheid als de snelheid van de detectie belangrijke factoren zijn. In deze paper ligt de focus op de ontwikkeling van een proof-of-concept detectiesysteem dat zowel in een statische als een realtime omgeving ingezet kan worden, gebruikmakend van deep learning technieken en specifiek het YOLOv11 framework.

De toepassing van warmtebeelddetectie voor militaire voertuigen is een belangrijk onderzoeksgebied met zowel militaire als civiele toepassingen. Warmtebeeldtechnologie biedt aanzienlijke voordelen ten opzichte van conventionele visuele detectiemethoden, aangezien deze in een omgeving met weinig licht kan functioneren en het potentieel heeft om door verschillende vormen van visuele camouflage heen te kijken. Een grote uitdaging binnen dit domein is echter het verkrijgen van geschikte trainingsdata, aangezien authentieke warmtebeelden van militaire voertuigen beperkt beschikbaar zijn.

Om deze dataschaarste aan te pakken, stelt deze paper een andere benadering voor waarbij gebruik wordt gemaakt van de militaire simulatiegame "War Thunder" als databron. Hoewel deze aanpak geen perfecte replicatie van authentieke warmtesignaturen biedt, creëert het wel een gecontroleerde omgeving voor de ontwikkeling en het testen van detectiealgoritmen. Dit vormt een proof-of-concept dat later kan worden aangepast naar praktijktoepassingen. De nachtzichtmodus van het spel biedt een redelijke benadering van warmtebeeldkarakteristieken, wat het mogelijk maakt om een diverse dataset op te bouwen met verschillende tankmodellen vanuit verschillende hoeken, afstanden en omgevingscondities.

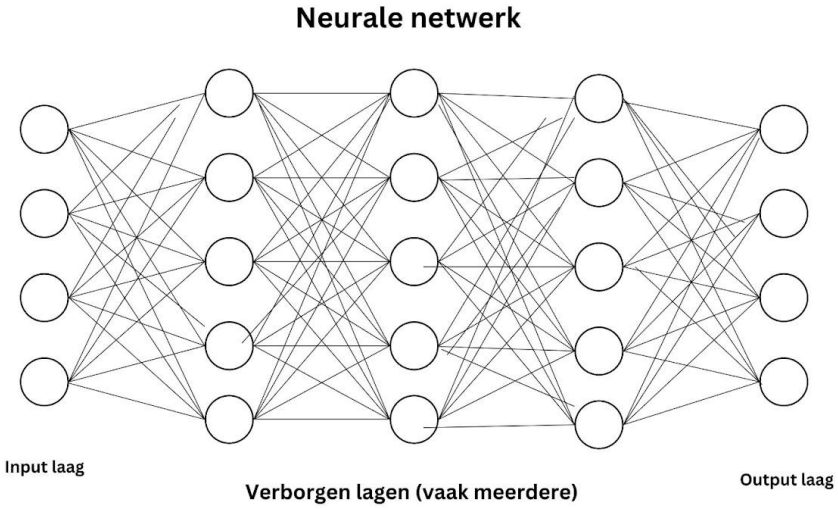
Voor de implementatie wordt gebruik gemaakt van Ultralytics YOLOv11, een state-of-the-art object detectie framework dat bekend staat om zijn balans tussen snelheid en nauwkeurigheid. YOLOv11 bouwt voort op het succes van eerdere versies en introduceert nieuwe functionaliteiten en verbeteringen die zowel de prestaties als de flexibiliteit verder verbeteren. Dit framework is specifiek gekozen vanwege zijn vermogen om realtime detectie uit te voeren, wat essentieel is voor de beoogde praktische toepassingen van dit onderzoek.

1. Achtergrond

1.1. Deep Learning

Artificiële neurale netwerken (vaak neurale netwerken of ANN genoemd) vormen de basis van moderne beeldherkenningssystemen. Ze zijn geïnspireerd door de werking van biologische neuronen in het menselijk brein. Een neuraal netwerk bestaat uit verschillende lagen van onderling verbonden kunstmatige neuronen die gegevens verwerken. Elke neuron ontvangt een input, past gewichten toe, en produceert een output via een activatiefunctie.

De structuur van een neuraal netwerk bestaat uit verschillende lagen, geïllustreerd in Figuur 1.1. Een invoerlaag die ruwe gegevens ontvangt, waarbij in dit geval elke pixel van de thermische beelden wordt voorgesteld door één node in deze invoerlaag. Na de invoerlaag volgen meerdere verborgen lagen die de patronen en complexe structuren verwerken. Het netwerk eindigt met een uitvoerlaag die classificaties of voorspellingen produceert, waarbij voor het tankdetectiesysteem elke outputnode een specifiek tanktype voorstelt. De node die de hoogste activatiewaarde heeft (lees: het meest 'oplicht') bepaalt uiteindelijk welke tank wordt voorspeld door het model. [2]



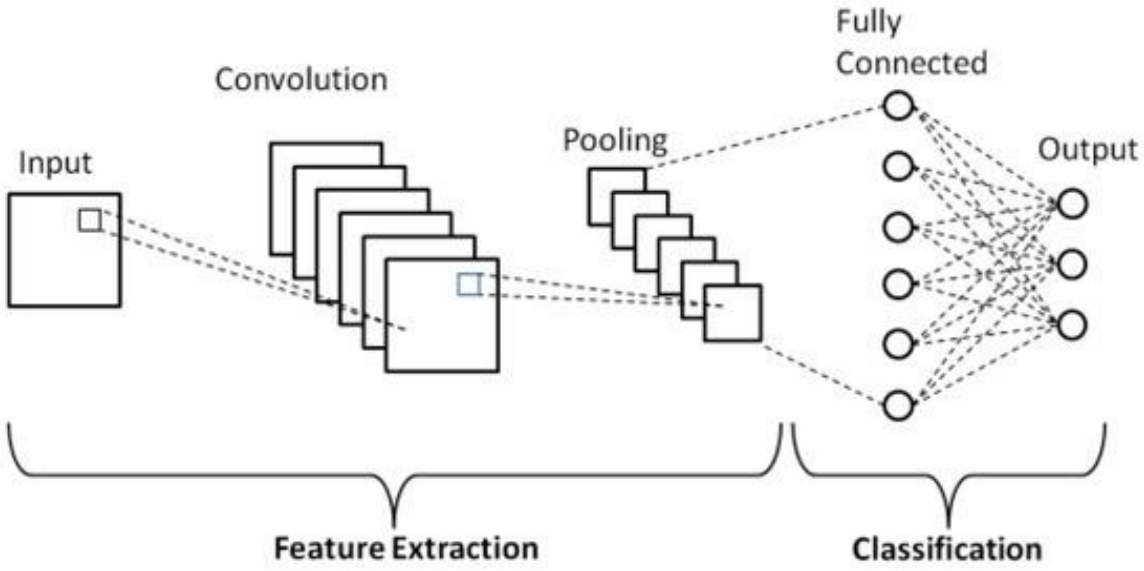
Figuur 1.1: Neuraal netwerk

Het leerproces van deze netwerken verloopt via 'backpropagation', waarbij de verschillen tussen voorspelde en werkelijke uitkomsten (de 'loss') worden gebruikt om de gewichten in het netwerk aan te passen. Dit proces gebeurt iteratief, waarbij elke trainingscyclus het netwerk verfijnt om betere voorspellingen te maken Dit leerproces kan worden onderverdeeld in drie fasen. In de eerste fase wordt het netwerk geïnitialiseerd, waarbij de gewichten (die de sterkte van verbindingen tussen neuronen bepalen) en biases (de drempelwaarden die bepalen wanneer een neuron activeert) meestal met kleine, willekeurige waarden worden aangemaakt. De tweede fase omvat het definiëren van een verliesfunctie (ook wel cost function), zoals

cross-entropy voor classificatietaken, die bepaalt hoe ver de voorspellingen afwijken van de werkelijke waarden. In de derde fase vindt de eigenlijke backpropagation plaats, waarbij gradiënten worden berekend om te bepalen hoe elk gewicht moet worden aangepast om de totale fout te verminderen, waarna deze aanpassingen worden doorgevoerd volgens een optimalisatie-algoritme zoals gradient descent.

1.2. Convolutionele Neurale Netwerken (CNN)

Een Convolutioneel Neuraal Netwerk (CNN) is een type neuraal netwerk dat vooral gebruikt wordt voor beeldherkenning [2]. Het bestaat uit opeenvolgende convolutionele lagen, die met behulp van kleine filters patronen in een afbeelding leren herkennen. Anders dan bij traditionele beeldverwerkingsmethoden worden deze patronen niet voorafgaand bepaald. Het netwerk leert zelf tijdens de training welke visuele kenmerken (zoals randen, hoeken of vormen) relevant zijn voor de taak. Deze filters schuiven over de afbeelding — dit is de eigenlijke convolutie-operatie — en produceren feature-maps die aangeven waar bepaalde kenmerken voorkomen. Omdat dezelfde filter op meerdere plaatsen wordt toegepast, zijn er veel minder parameters dan bij een volledig verbonden laag, wat het netwerk efficiënter en sneller maakt. Bovendien zorgt dit ervoor dat het netwerk een object kan herkennen, ongeacht waar het zich in de afbeelding bevindt, een eigenschap bekend als translatievariantie.



Figuur 1.2: Convolutioneel neuraal netwerk

Aan het einde van het netwerk worden meestal één of meerdere fully-connected lagen (verder FC-lagen genoemd) toegevoegd. In een FC-laag is elke neuron verbonden met alle neuronen uit de vorige laag, wat het krachtig maakt om complexe combinaties van kenmerken te leren. Het nadeel is echter dat deze lagen veel geheugen en rekenkracht vereisen, omdat ze een groot aantal parameters bevatten. Daarom worden ze enkel op het einde gebruikt om de verzamelde features om te zetten naar een eindresultaat, zoals een classificatie.

1.3. Objectdetectie vs. Classificatie

Bij beeldverwerking is het belangrijk om het verschil tussen classificatie en detectie te begrijpen. Classificatie beantwoordt de vraag "Wat is er in dit beeld?" en kent een klasse toe aan een volledig beeld. Een classificatiemodel kan bepalen of een beeld een tank bevat, maar niet waar die tank zich precies bevindt. Objectdetectie gaat verder en beantwoordt zowel "Wat is er in dit beeld?" als "Waar bevindt het zich?". Een detectiemodel localiseert objecten met zogenaamde 'bounding boxes' en classificeert elk gedetecteerd object. In onze toepassing betekent dit dat we niet alleen kunnen identificeren dat er een tank aanwezig is, maar ook waar deze zich precies in het warmtebeeld bevindt en welk type tank het is.

Traditionele objectdetectiemethoden gebruikten vaak een 'sliding window'-benadering, waarbij een classificatiemodel wordt toegepast op verschillende delen van het beeld. Deze methode is echter computationeel intensief en onpraktisch voor realtime toepassingen zoals deze beoogt in dit project. Moderne algoritmes hebben deze beperking overwonnen, waaronder ook het YOLO-framework (You Only Look Once), dat later in sectie 3.2 uitgebreid wordt besproken. YOLO transformeert objectdetectie naar een regressieprobleem waarbij het hele beeld in één keer wordt geanalyseerd, wat een drastische verbetering in verwerkingssnelheid oplevert zonder grote compromissen op het gebied van nauwkeurigheid.

1.4. Warmtebeelden en Infrarooddetectie

Warmtebeeldtechnologie, ook bekend als thermische beeldvorming, detecteert infraroodstraling die door objecten wordt uitgezonden en zet deze om in visuele beelden [13]. In tegenstelling tot conventionele detectiemethoden die afhankelijk zijn van zichtbaar licht, baseert warmtebeeldtechnologie zich op temperatuurverschillen tussen objecten en hun omgeving. Deze technologie biedt voor militaire toepassingen grote voordelen ten opzichte van andere nachtzichttechnologieën. Het belangrijkste voordeel is de mogelijkheid om te functioneren in omgevingen met weinig of geen licht. Thermische sensoren hebben geen externe lichtbron nodig en kunnen volledig opereren in het donker, wat hen ideaal maakt voor nachtelijke operaties. Daarnaast kunnen warmtebeelden tot op zekere hoogte doorheen rook, lichte mist, stof en lichte vegetatie kijken, waardoor operationele zichtbaarheid behouden blijft in omstandigheden waar conventionele systemen falen.

Het is belangrijk om het onderscheid te maken tussen conventionele 'Night Vision' (nachtzicht) en 'Thermal Imaging' (warmtebeeldvorming), aangezien deze technologieën op fundamenteel verschillende principes werken. Conventionele nachtzichttechnologie, ook bekend als beeldversterking, werkt door het versterken van beschikbaar omgevingslicht (zoals maanlicht of sterrenlicht) en vereist minimaal enige vorm van licht om te functioneren. Deze technologie versterkt reflecterend licht en toont beelden in de karakteristieke groene tint. Thermal Imaging daarentegen detecteert uitgezonden infraroodstraling (warmte) en werkt volledig onafhankelijk van beschikbaar licht, waardoor het kan functioneren in absolute duisternis en door bepaalde obstructies heen kan 'zien'. Hoewel we in deze paper regelmatig de term 'nachtzichtmodus' gebruiken wanneer we refereren naar de functionaliteit in War Thunder, betreft het hier specifiek Thermal Imaging technologie. Deze simulatie geeft de warmtesignaturen van tanks weer in het kenmerkende 'wit-heet' kleurenschema, waarbij warmere objecten in lichtere tinten worden getoond tegen een donkere achtergrond, wat karakteristiek is voor thermische beeldvorming in militaire toepassingen en niet voor de standaard groene nachtzichtbeelden.

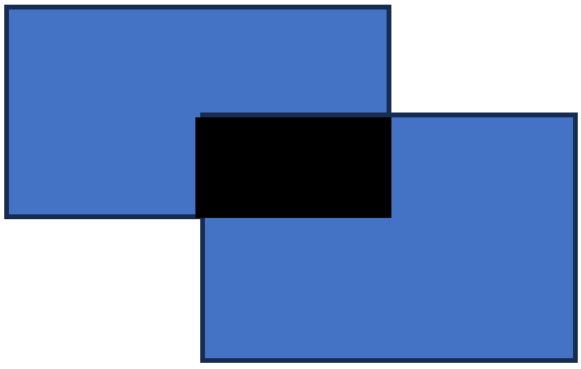
In de context van thermische beeldvorming zijn verschillende kleurenweergavesystemen beschikbaar, elk met specifieke toepassingsgebieden. Het meest gebruikte in militaire context is het wit-heet schema, waarbij warmere objecten in wittere tinten worden weergegeven tegen een donkerdere achtergrond. Dit biedt optimaal contrast en intuïtieve interpretatie. Het alternatieve 'zwart-heet' schema toont warmere objecten in donkerdere tinten. Daarnaast bestaan er polychrome schema's (regenboog, ijzer, arctisch) die een breder kleurenspectrum gebruiken om temperatuurverschillen weer te geven, maar deze worden in tactische militaire situaties minder frequent toegepast vanwege hun complexere visuele interpretatie.

Warmtebeeldtechnologie is bijzonder effectief bij het detecteren van visueel gecamoufleerde objecten. Terwijl conventionele camouflage ontworpen is om objecten te verbergen in het zichtbare spectrum, heeft dit doorgaans weinig invloed op de warmtesignatuur. Militaire voertuigen zoals tanks blijven duidelijk zichtbaar voor thermische camera's, ongeacht hoe goed ze visueel gecamoufleerd zijn. Een uitzondering hier is thermische camouflage aan de hand van isolerend materiaal maar dit valt buiten de scope van deze paper.

Tanks vertonen karakteristieke warmtepatronen die ze identificeerbaar maken via thermische beeldvorming. De warmtebronnen in tanks omvatten voornamelijk de motorwarmte en uitlaatsystemen, die veel hitte genereren tijdens operatie. Verschillende materiaaltypes in de constructie van tanks (metaal, rubber, etc.) absorberen en stralen warmte verschillend uit, wat zorgt voor een herkenbaar thermisch mozaïek. Het contrast tussen het doorgaans warmere voertuig en de koelere natuurlijke omgeving creëert bovendien een duidelijk herkenbare thermische handtekening.

1.5. Evaluatie van Detecties

Om de effectiviteit van ons detectiesysteem te meten, gebruiken we verschillende evaluatiemetrieken. Intersection over Union (IoU) meet de nauwkeurigheid van bounding boxes door de verhouding tussen het overlappende gebied en het totale gebied van de voorspelde en werkelijke boxes te berekenen. Dit principe wordt geïllustreerd in Figuur 1.3. Een IoU-waarde van 0.5 of hoger wordt vaak beschouwd als een succesvolle detectie.



Figuur 1.3: De IoU is de oppervlakte van de doorsnee (zwart) gedeeld door de oppervlakte van het geheel (blauw + zwart)

Precision meet het percentage correcte detecties ten opzichte van alle detecties die het model maakt. Het beantwoordt de vraag: "Van alle zaken die het model identificeerde als tank, hoeveel waren daadwerkelijk tanks?"

Recall, ook bekend als sensitivity, meet het percentage correcte detecties ten opzichte van alle werkelijke objecten. Het beantwoordt de vraag: "Van alle tanks die daadwerkelijk in de beelden aanwezig waren, hoeveel heeft het model er gevonden?"

Precision-Recall curve toont de balans tussen precision en recall bij verschillende drempelwaarden. De oppervlakte onder deze curve, bekend als Average Precision (AP), geeft een enkele waarde die de algehele prestatie van het detectiesysteem samenvat.

Mean Average Precision (mAP) is het gemiddelde van de AP-waarden over verschillende klassen of categorieën (in ons geval verschillende tankmodellen), en biedt een allesomvattende maatstaf voor het vergelijken van detectiesystemen.

Voor ons project zijn deze metrieken [1] belangrijk om de prestaties van ons YOLO-model te evalueren en te verbeteren, zowel voor de statische als de realtime implementatie.

2. Probleemstelling

Het hoofddoel van deze bachelorproef is om een systeem te ontwikkelen dat militaire tanks kan detecteren en classificeren op basis van warmtebeelden, waarbij zowel statische als realtime toepassingen worden onderzocht. De focus ligt op het ontwikkelen van een proof-of-concept met behulp van gesimuleerde warmtebeelden uit War Thunder, waarbij gebruik wordt gemaakt van het YOLOv11 framework voor snelle en accurate detectie.

De mogelijkheden en voordelen van dit project zijn veelzijdig:

- Het ontwikkelde systeem kan bijdragen aan verbeterde militaire verkenning en identificatie;
- De realtime detectiecapaciteit biedt potentieel voor directe tactische toepassingen;
- De methode kan worden uitgebreid naar andere militaire voertuigen en objecten;
- De innovatieve aanpak voor het verzamelen van trainingsdata via gaming kan een blauwdruk vormen voor vergelijkbare projecten.

In dit onderzoek zullen de volgende kernvragen worden beantwoord:

- Hoe effectief is YOLOv11 in het detecteren en classificeren van tanks in warmtebeelden?
    - Wat is de nauwkeurigheid van de detectie?
    - Hoe snel kan het systeem tanks identificeren?
    - Welke factoren beïnvloeden de prestaties van het systeem?
- Hoe betrouwbaar is de classificatie van verschillende tankmodellen?
    - Welke kenmerken zijn doorslaggevend voor accurate classificatie?
    - Hoe goed presteert het systeem bij verschillende kijkhoeken en afstanden?
    - Wat is de impact van omgevingsfactoren op de classificatie?
- In hoeverre is de War Thunder dataset representatief?
    - Welke beperkingen heeft het gebruik van gesimuleerde warmtebeelden?
    - Hoe vertaalbaar zijn de resultaten naar real-world toepassingen?
    - Welke aanpassingen zijn nodig voor implementatie met echte warmtebeelden?
- Wat zijn de prestaties van het realtime systeem?
    - Kan het systeem effectief functioneren tijdens het gamen?
    - Welke hardware-vereisten zijn er voor optimale prestaties?
    - Hoe kan de realtime implementatie worden geoptimaliseerd?
- Welke verbeteringen zijn mogelijk voor toekomstige ontwikkeling?

- Kan het systeem worden uitgebreid naar andere militaire voertuigen?
- Welke aanvullende features zouden de prestaties kunnen verbeteren?
- Hoe kan het systeem worden aangepast voor specifieke use-cases?

De evaluatie van het systeem zal worden uitgevoerd door:

- Systematische tests met verschillende tankmodellen in War Thunder
- Analyse van de detectie- en classificatienauwkeurigheid
- Meting van de verwerkingssnelheid en systeemprestaties
- Beoordeling van de realtime functionaliteit tijdens het gamen
- Vergelijking van prestaties onder verschillende omstandigheden (afstand, hoek, belichting)

3. Methode

3.1. Modelevaluatie

Bij de ontwikkeling van een systeem voor tankdetectie op basis van warmtebeelden is de keuze van het juiste objectdetectiemodel essentieel. Het model moet een optimale balans bieden tussen nauwkeurigheid en verwerkingssnelheid, vooral gezien onze doelstelling om zowel statische als realtime detectie te realiseren.

Objectdetectiemodellen worden traditioneel ingedeeld in twee categorieën: two-stage detectors en single-stage detectors. Two-stage detectors, zoals de R-CNN familie, gebruiken eerst een apart algoritme om regio's van belang te identificeren en daarna een classificatiemodel om die regio's te evalueren. Single-stage detectors, zoals YOLO en SSD, evalueren daarentegen het hele beeld in één keer. Dit is typisch sneller maar mogelijk minder nauwkeurig is. De prestaties van deze modellen worden gemeten en vergeleken aan de hand van twee hoofdcriteria. De eerste is nauwkeurigheid, meestal uitgedrukt als mean Average Precision (mAP), waarbij indices zoals mAP_{50} en mAP_{75} verwijzen naar IoU-drempelwaarden van respectievelijk 0.5 en 0.75. De tweede is verwerkingssnelheid, uitgedrukt in Frames Per Second (FPS), wat aangeeft hoeveel beelden het model per seconde kan verwerken. Voor commerciële realtime toepassingen wordt meestal 30 FPS gewenst, maar in dit geval is het acceptabel om te streven naar minimaal 10 FPS. Deze lagere drempelwaarde is voldoende voor onze proof-of-concept met beperkte rekenkracht.

In praktische toepassingen kan de beeldstroom bovendien worden gesubsampled door verschillende methoden. Ten eerste kan de resolutie van beelden worden verlaagd door slechts een deel van de originele pixels te behouden, een proces ook bekend als decimatie of downsampling, wat de hoeveelheid te verwerken data vermindert terwijl het gezichtsveld behouden blijft. Ten tweede kan de beeldenstroom temporeel worden gesubsampled door slechts één op de zoveel beelden door het model te laten verwerken, bijvoorbeeld elke derde frame analyseren in plaats van elke frame. Beide technieken leiden tot snellere verwerkingstijden ten koste van enig detailverlies. Tabel 3.1 toont een vergelijking van verschillende 2D-objectdetectiemodellen en hun performantie op benchmark datasets bestaande uit algemene objecten (PASCAL VOC 2012 en MS COCO). Deze datasets, hoewel niet specifiek gericht op tanks noch warmtebeelden, bieden een gestandaardiseerde maatstaf voor het vergelijken van modelprestaties.

Model	**Year**	**Backbone**	**Size**	**AP[0.5:0.95]**	**AP0.5**	**FPS**
R-CNN*	2014	AlexNet	224	-	58.50%	~0.02
SPP-Net*	2015	ZF-5	Variabel	-	59.20%	~0.23
Fast R-CNN*	2015	VGG-16	Variabel	-	65.70%	~0.43
Faster R-CNN*	2015	VGG-16	600	-	67.00%	5
R-FCN	2016	ResNet-101	600	31.50%	53.20%	~3
FPN	2017	ResNet-101	800	36.20%	59.10%	5

Mask R-CNN	2017	ResNeXt-101-FPN	800	39.80%	62.30%	5
DetectorRS	2020	ResNeXt-101	1333	53.30%	71.60%	~4
YOLO*	2016	(Modified) GoogLeNet	448	-	57.90%	45
SSD	2016	VGG-16	300	23.20%	41.20%	46
YOLOv2	2017	DarkNet-19	352	21.60%	44.00%	81
RetinaNet	2018	ResNet-101-FPN	400	31.90%	49.50%	12
YOLOv3	2018	DarkNet-53	320	28.20%	51.50%	30
CenterNet	2019	Hourglass-104	512	42.10%	61.10%	7.8
EfficientDet-D2	2020	Efficient-B2	768	43.00%	62.30%	41.7
YOLOv4	2020	CSPDarkNet-53	512	43.00%	64.90%	31
Swin-L	2021	HTC++	-	-	57.70%	-

Modellen gemarkeerd met een * zijn getest met de PASCAL VOC 2012 benchmark, anderen met MS COCO. De grijs gekleurde rijen zijn realtime detectoren (>30 FPS)

Tabel 3.1: Benchmarks van 2D-objectdetectiemodellen [15]

Uit deze benchmarks kunnen we verschillende belangrijke observaties maken over de prestaties van deze modellen. De YOLO-familie toont een uitstekende balans tussen nauwkeurigheid en snelheid. Het oorspronkelijke YOLO-model bereikt een mAP_{50} van 57,9% met een FPS van 45. YOLOv2 heeft een lagere mAP_{50} van 44,0% maar een aanzienlijk hogere verwerkingssnelheid van 81 FPS. YOLOv3 verbetert de nauwkeurigheid naar een mAP_{50} van 51,5% met een nog steeds zeer acceptabele 30 FPS, terwijl YOLOv4 dit verder optimaliseert naar een mAP_{50} van 64,9% bij 31 FPS. Het is belangrijk op te merken dat deze vergelijking niet alle varianten in de YOLO-familie meeneemt, aangezien er sinds de publicatie van deze benchmarks nieuwere versies zijn uitgebracht, waaronder YOLOv5, YOLOv8 en de meest recente YOLOv11.

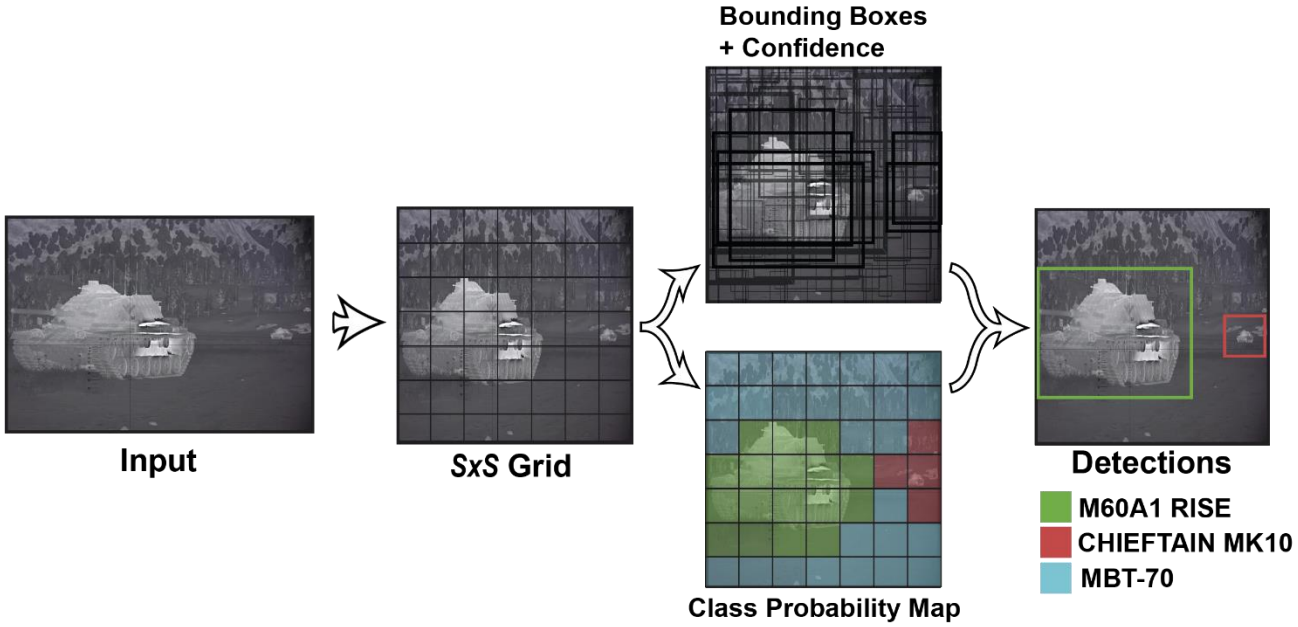
Aan de andere kant zien we dat two-stage detectors zoals Faster R-CNN een respectabele mAP_{50} van 67,0% behalen, maar met een verwerkingssnelheid van slechts 5 FPS, wat ze ongeschikt maakt voor onze realtime toepassingen. Het meest nauwkeurige model in de tabel is DetectorRS met een indrukwekkende mAP_{50} van 71,6%, maar met slechts ongeveer 4 FPS is het eveneens ongeschikt voor realtime gebruik. SSD bereikt een mAP_{50} van 41,2% bij 46 FPS, maar deze nauwkeurigheid is aanzienlijk lager dan die van YOLOv4. EfficientDet-D2 is een sterke concurrent met een mAP_{50} van 62,3% bij 41,7 FPS, wat het tot een van de weinige modellen maakt die zowel nauwkeurig als snel is. Wegens de hoge performantie, uitgebreide documentatie en constante vernieuwing heb ik besloten om te kiezen voor het YOLO-framework.

3.2. You-Only-Look-Once

De kern van YOLO's efficiëntie ligt in het vermogen om objectdetectie te behandelen als een regressieprobleem, waarbij het gehele beeld slechts één keer wordt geanalyseerd om alle objecten te detecteren [6]. Dit principe verklaart ook de naamgeving 'You Only Look Once'. Dit is in tegenstelling tot eerdere methoden zoals R-CNN varianten die eerst regio's moeten identificeren en vervolgens classificeren, of sliding window-technieken die het beeld meerdere keren moeten scannen. In de volgende secties wordt eerst de algemene werking besproken van het YOLO-algoritme, gevolgd door een analyse van de verliesfunctie die het leerproces stuurt. Vervolgens worden verschillende technieken toegelicht die YOLO tot een bijzonder krachtig algoritme maken, waaronder non-max suppression voor het elimineren van overlappende detecties, spiegeltechnieken voor het verbeteren van de betrouwbaarheid, en segmentatiemethoden voor het omgaan met verschillende beeldformaten.

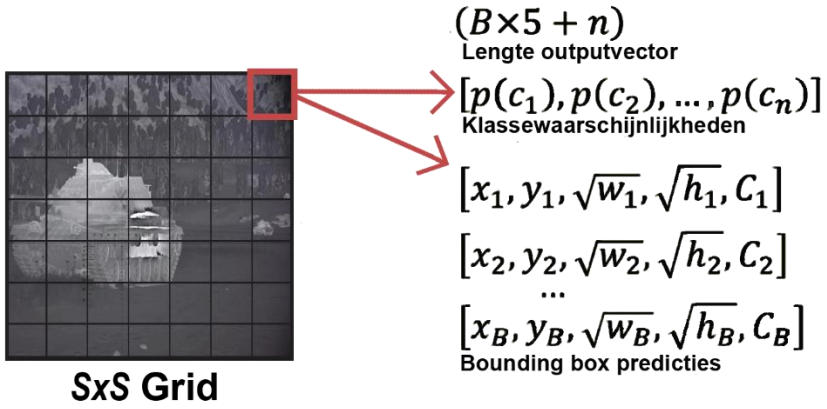
3.2.1. Werking

Het YOLO-algoritme deelt de inputafbeelding op in een S×S grid van cellen [5]. Elke gridcel is verantwoordelijk voor het voorspellen van objecten waarvan het middelpunt binnen die cel valt. Voor elke gridcel produceert het model een reeks begrenzingskaders (bounding boxes) met bijbehorende betrouwbaarheidsscores (confidence scores). Dit zijn scores tussen 0 en 1 die aangeven hoe zeker het model is dat er een object in het kader aanwezig is, evenals een waarschijnlijkheidskaart voor de verschillende objectklassen, die aangeeft wat voor soort object zich waarschijnlijk in de cel bevindt. Deze informatie wordt vervolgens gecombineerd om de uiteindelijke detecties te produceren. Het is belangrijk te begrijpen dat er voor elke gridcel altijd een predictie gemaakt wordt over welke klasse het meest waarschijnlijk is. Een klasse met de hoogste waarschijnlijk zal dus niet noodzakelijk leiden tot een detectie als de gekozen grenswaarde niet overschreden wordt. Dit concept wordt geïllustreerd in Figuur 3.1 waarbij de MBT-70 tank als meest waarschijnlijke klasse wordt voorspeld voor de gridcellen die geen tank bevatten.



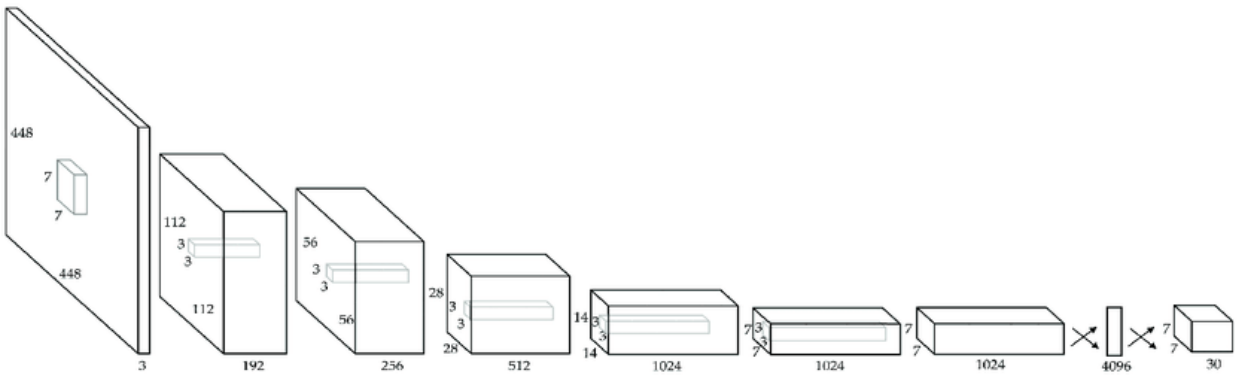
Figuur 3.1: Werking YOLO-model

Concreet betekent dit dat voor elke gridcel een reeks outputvectoren wordt berekend. Als eerste een vector van klassenwaarschijnlijkheden. De tweede is een reeks parameters voor de bounding boxes. Per gridcel kunnen meerdere bounding boxes berekend worden, hierdoor kan YOLO meerdere objecten in dezelfde cel detecteren. Hierbij is B het aantal bounding boxes per gridcel. X en y zijn respectievelijk het x-as en het y-as coördinaat van het middelpunt van de gridcel. De breedte wordt aangetoond met w en de hoogte met h. C is de betrouwbaarheidsscore voor elk kader. In het voorbeeld uit Figuur 3.1 waarbij een 7x7 grid gehanteerd wordt met drie bounding boxes per cel en drie verschillende klassen zou dus voor elk van de 49 gridcellen een outputvector krijgen met lengte (Bx5+n) = (3x5+3) = 18 berekend worden.



Figuur 3.2: Outputvectoren van de gridcellen

De transformatie van een inputafbeelding naar de verschillende outputvectoren gebeurt aan de hand van een Convolutioneel Neuraal Netwerk. Dit is een type neuraal netwerk die de verschillende features uit visuele data kan halen. De architectuur hiervan bestaat uit een reeks convolutielagen die het beeld omzetten in een abstractere representatie, gevolgd door Fully Connected Layers die deze representatie verder transformeren tot outputvectoren voor elke gridcel.



Figuur 3.3: Convolutioneel Neuraal Netwerk

3.2.2. Verliesfunctie

De verliesfunctie (Loss function) is een belangrijke component die het leerproces van het model stuurt. Het optimaliseren van deze functie gebeurt via een aangepaste vorm van de som van kwadraten van fouten. Het proces begint door te bepalen welke bounding box voorspellers verantwoordelijk zijn voor een ground truth. Deze toewijzing gebeurt door voor elke ground truth de voorspeller te zoeken met de grootste overlap, gemeten in IoU. Als er geen enkele voorspeller overlapt, wordt de voorspeller gekozen die het dichtste bij ligt, gemeten via de kleinste som van kwadraten. Een belangrijke voorwaarde is dat een bounding box voorspeller alleen verantwoordelijk kan zijn voor een ground truth als het middelpunt van die ground truth zich in hetzelfde segment bevindt. Bovendien kan elke voorspeller maar voor één ground truth verantwoordelijk zijn, en omgekeerd.

De loss functie voor bounding box-voorspellers die verantwoordelijk zijn voor een ground truth is als volgt [7]:

$$\lambda_r \times [(x - x')^2 + (y - y')^2 + (\sqrt{b} - \sqrt{b'})^2 + (\sqrt{h} - \sqrt{h'})^2] + (Conf - IoU)^2$$

Hierin zijn b, h en Conf respectievelijk de breedte, hoogte en zekerheidsscore van de bounding box. De factor λ_r is typisch 5, wat ervoor zorgt dat de lokalisatiefouten zwaarder doorwegen dan andere fouten. Dit is nodig om het aantal lokalisatiefouten te verminderen. Het gebruik van de vierkantswortel bij breedte en hoogte zorgt ervoor dat fouten bij kleine objecten relatief zwaarder doorwegen dan bij grote objecten. De IoU term in de formule is de daadwerkelijke overlap tussen de voorspelde bounding box en de ground truth.

Voor voorspellers die niet verantwoordelijk zijn voor een ground truth object, is de loss functie eenvoudiger:

$$\lambda_{no} \times \text{Conf}^2$$

Deze term drijft de zekerheidswaarde naar nul voor voorspellers die geen object detecteren. De factor λ_{no} is meestal 0.5, wat lager is dan λ_r omdat de meeste voorspellers geen object moeten detecteren en we niet willen dat hun zekerheidsscore te hard naar nul wordt gedreven.

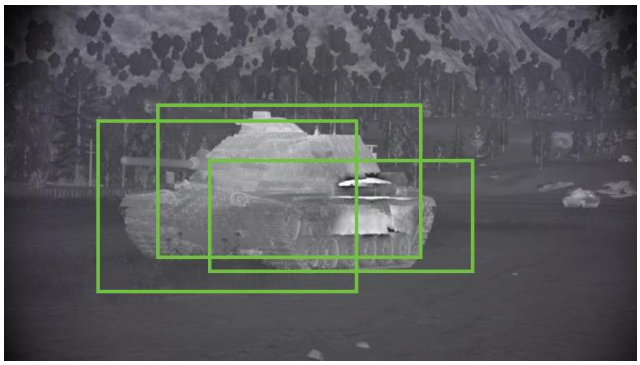
De loss functie voor de klassevoorspellingen wordt per segment berekend met de formule:

$$\sum_{c \in klassen} (\mathrm{p}(\mathrm{c}) - \mathrm{p}'(\mathrm{c}))^2$$

Hierbij is p'(c) gelijk aan 1 als er een object van klasse c in het segment aanwezig is, en anders 0. Deze term wordt alleen toegepast op segmenten waarin daadwerkelijk een object aanwezig is. Voor segmenten zonder objecten is deze loss component 0.

3.2.3. Non-max suppression

Wanneer er in één bounding box meerdere klassen voorspeld worden en de IoU van de mogelijke detecties groter is dan een bepaalde grenswaarde, dan wordt enkel de predictie met de hoogste confidence score gebruikt. Standaard zit deze grenswaarde op 0.5. Dit process heet “non max suppression” en zorgt ervoor dat sterk overlappende detecties worden samengenomen en dat objecten niet dubbel gedetecteerd worden [7]. Verder kan non-max suppression gebruikt worden als een vorm van post-processing die de accuraatheid van de finale bounding box verhoogt. Hierbij neemt men de confidence scores van de verschillende detecties op als gewogen gemiddelde om uiteindelijk een nieuwe bounding box te creëren die rekening houdt met alle verschillende detecties in plaats van enkel de meest zekere.



Figuur 3.4: Geen non-max suppression

Figuur 3.5: Non-max suppression

3.2.4. Spiegelen

Een andere methode die gebruikt wordt om de kwaliteit van het model te verhogen is door de inputafbeelding te spiegelen om vervolgens de detectie van de gespiegelde en de niet-gespiegelde afbeelding samen te nemen [7]. Als een detectie in beide versies voorkomt, dan is de kans groter dat het om een true positive gaat dan wanneer de detectie slechts in één van de versies voorkomt. Een detectie die slechts in één versie voorkomt zal bijgevolg een lagere waarschijnlijkheid toegewezen krijgen.

Het samenvoegen van detecties uit de originele en gespiegelde beelden verloopt via een aangepaste vorm van non-max suppression. In plaats van simpelweg de hoogste betrouwbaarheidsscore te behouden bij overlappende detecties, worden de probability scores opgeteld en vervolgens gehalveerd. Dit creëert een gewogen gemiddelde waarbij detecties die in beide varianten voorkomen hun gemiddelde betrouwbaarheidsscore behouden, terwijl detecties die slechts in één versie verschijnen effectief met 50% worden gereduceerd. Deze

techniek wordt toegepast nadat eerst reguliere non-max suppression afzonderlijk op beide beeldsets is uitgevoerd.

3.2.5. Segmentatie

Wanneer een afbeelding, zoals een nachtzichtopname van een tank in War Thunder getoond in Figuur 3.6, een aspect ratio heeft die sterk afwijkt van het vierkante invoerformaat dat het YOLO-netwerk vereist, ontstaat er een nieuw probleem. Direct schalen naar een vierkant formaat zou leiden tot een vervorming van de tanks die we proberen te detecteren, zoals te zien is in Figuur 3.7. Segmentatie biedt hier een elegante oplossing voor. De originele afbeelding wordt opgedeeld in meerdere vierkante secties – in dit geval zichtbaar als een groene en rode zone op Figuur 3.8 – die elk afzonderlijk door het netwerk worden geanalyseerd [7]. In beide secties wordt de tank gedetecteerd, alleen wordt in de groene sectie de tank volledig gedetecteerd en in de rode slechts gedeeltelijk. In dit geval waar een tank deels in meerdere segmenten voorkomt, zou dit mogelijks kunnen leiden tot dubbele of onvolledige detecties. Om dit tegen te gaan, implementeert het model een soort grensfilter. Detecties die zeer dicht bij de rand van een segment liggen worden verwijderd, uitgezonderd aan de werkelijke randen van de oorspronkelijke afbeelding. Zonder segmentatie zou het model moeite hebben om de tank correct te identificeren vanwege de vervorming in de geschaalde foto. Met segmentatie blijft de oorspronkelijke aspect ratio binnen elk segment behouden, waardoor de tank in zijn originele proportie wordt geanalyseerd en betrouwbaar zonder dubbele detectie wordt gedetecteerd in het groene segment, zoals in Figuur 3.9.

Om te voorkomen dat segmentatie de verwerkingssnelheid van het model aanzienlijk vertraagt, worden alle segmenten parallel verwerkt als één batch. Deze aanpak maakt efficiënt gebruik van de GPU, zodat de meerdere beeldevaluaties gelijktijdig plaatsvinden in plaats van sequentieel.

Figuur 3.6: Originele foto

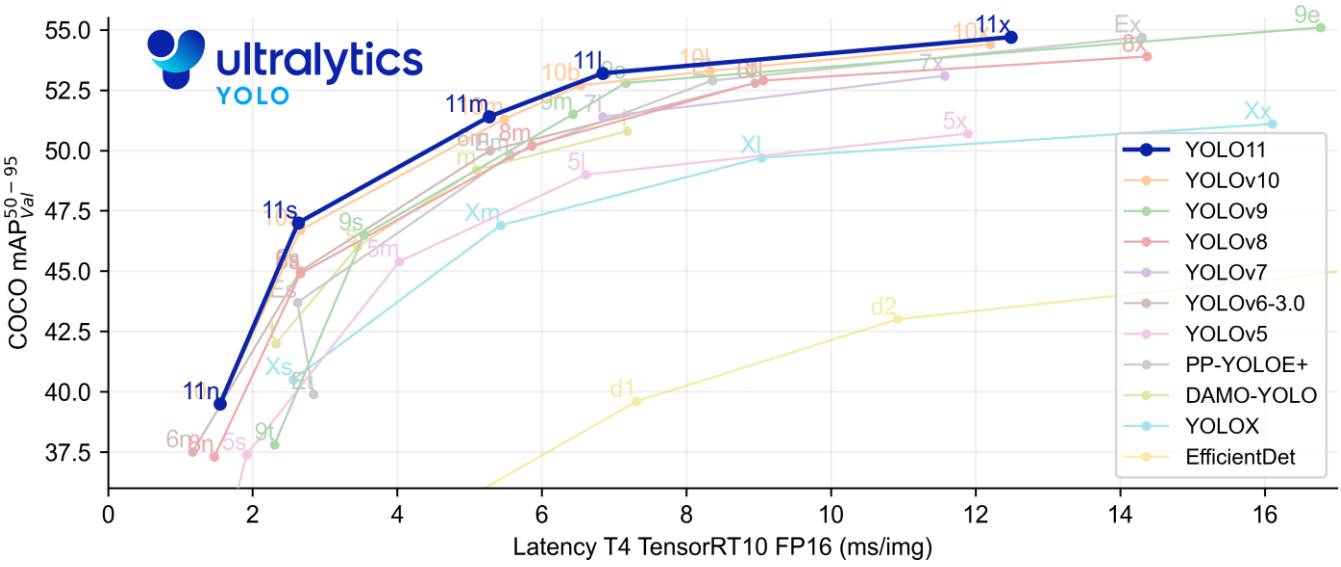
Figuur 3.7: Herschaalde vierkante foto, tank is misvormd

Figuur 3.8: Detectie op gesegmenteerde foto

Figuur 3.9: Gefilterde detectie op gesegmenteerde foto

3.3. Mogelijkheden binnen YOLO

Sinds de introductie van het originele YOLO-model in 2015-2016 heeft het algoritme aanzienlijke verbeteringen ondergaan. YOLOv1 introduceerde het basisprincipe, maar had moeite met detectie van kleine objecten en objecten die dicht bij elkaar liggen. YOLOv2/YOLO9000 verbeterde de nauwkeurigheid en introduceerde de mogelijkheid om duizenden objectcategorieën te detecteren. YOLOv3 verbeterde de detectie van objecten op verschillende schalen door gebruik te maken van feature pyramids, wat belangrijk is voor het detecteren van objecten van verschillende groottes. YOLOv4 integreerde nieuwe technieken zoals Cross-Stage Partial Networks (CSPNet), Path Aggregation Networks (PANet), en Mish activatie, wat leed tot een grote verbetering in zowel nauwkeurigheid als snelheid. Latere versies zoals YOLOv5, YOLOv8 en de meest recente YOLOv11 hebben verdere verbeteringen aangebracht in de architectuur en prestaties.



Figuur 3.10: Benchmarks verschillende YOLO-modellen [6]

Voor dit onderzoeksproject is er besloten te werken met YOLOv11, de nieuwste iteratie in de YOLO-familie. YOLOv11 bouwt voort op de sterke punten van zijn voorgangers en introduceert nieuwe functies die bijzonder relevant zijn voor onze toepassing. Het is uitermate geschikt voor onze tankdetectietaak vanwege zijn superioriteit voor realtime detectie, effectieve multi-scale detectie, robuustheid tegen variaties in warmtebeelden, efficiënt gebruik van computationele middelen, en de ondersteuning van een actieve gemeenschap en uitgebreide documentatie.

3.3.1. Modelvariant

YOLOv11 biedt vervolgens ook verschillende modelgroottes die elk een unieke balans tussen nauwkeurigheid en snelheid vertegenwoordigen. De prestaties van deze varianten worden vergeleken in Tabel 3.2 en zijn geëvalueerd op de COCO val2017 dataset.

De mAPval 50-95 (Mean Average Precision) is een maatstaaf die aangeeft hoe goed het model objecten kan detecteren en lokaliseren, waarbij de waarden 50-95 verwijzen naar de IoU

drempelwaarden die meten hoe goed de voorspelde begrenzingsvakken overlappen met de werkelijke locaties; een hogere mAP betekent betere detectienauwkeurigheid. Speed CPU ONNX (ms) en Speed T4 TensorRT10 (ms) meten de latentie of verwerkingstijd in milliseconden - hoelang het duurt om één afbeelding te analyseren, waarbij de CPU-meting prestaties op reguliere processors toont en de T4 TensorRT10-meting de snelheid op een NVIDIA T4 GPU met TensorRT-optimalisatie weergeeft; lagere getallen betekenen snellere verwerking. Het aantal trainbare parameters (params M) in miljoenen geeft de modelcomplexiteit aan, waarbij meer parameters kunnen leiden tot hogere nauwkeurigheid maar meer opslag en verwerkingskracht vereisen. FLOPs (B), ofwel Floating-point Operations in miljarden, is een maat voor de rekenkundige complexiteit van het model en geeft aan hoeveel berekeningen nodig zijn om een afbeelding te verwerken; lagere FLOPs betekenen doorgaans snellere verwerking, mogelijk ten koste van nauwkeurigheid.

Model	**size** (pixels)	**mAP^{val}** 50-95	**Speed** CPU ONNX (ms)	**Speed** T4 TensorRT10 (ms)	**params** (M)	**FLOPs** (B)
YOLO11n	640	39.5	56.1 ± 0.8	1.5 ± 0.0	2.6	6.5
YOLO11s	640	47.0	90.0 ± 1.2	2.5 ± 0.0	9.4	21.5
YOLO11m	640	51.5	183.2 ± 2.0	4.7 ± 0.1	20.1	68.0
YOLO11l	640	53.4	238.6 ± 1.4	6.2 ± 0.1	25.3	86.9
YOLO11x	640	54.7	462.8 ± 6.7	11.3 ± 0.2	56.9	194.9

Tabel 3.2: Benchmarks verschillende modelgroottes YOLOv11 [6]

De YOLO11n-variant is het lichtste model met slechts 2,6 miljoen parameters en 6,5 miljard FLOPS, en bereikt een mAP-waarde van 39,5 op de COCO-dataset. Dit model is ideaal voor omgevingen met beperkte rekenkracht, maar biedt niet de hoogste nauwkeurigheid. Aan de andere kant staat YOLO11x, het meest uitgebreide model met 56,9 miljoen parameters en 194,9 miljard FLOPS, dat een mAP van 54,7 bereikt. Dit model biedt de hoogste nauwkeurigheid, maar vereist aanzienlijk meer rekenkracht en is langzamer in de uitvoering.

Voor dit project is er specifiek gekozen voor YOLO11l, wat een optimale middenweg biedt tussen nauwkeurigheid en snelheid. Met zijn 25,3 miljoen parameters en 86,9 miljard FLOPS bereikt het een mAP van 53,4, wat slechts marginaal lager is dan het veel grotere YOLO11x-model. De inferentiesnelheid van YOLO11l op CPU (ONNX) is ongeveer 238,6 milliseconden, terwijl deze op een NVIDIA T4 GPU met TensorRT-optimalisatie daalt naar slechts 6,2 milliseconden per beeld. Deze prestaties maken het model bijzonder geschikt voor onze realtime overlay toepassing, waar we een hoge doorvoersnelheid nodig hebben zonder te veel in te leveren op nauwkeurigheid.

Naast de modelgrootte biedt YOLOv11 ook flexibiliteit in de inferentie-instellingen. We kunnen parameters zoals het confidence-drempelwaarde en de IoU-drempelwaarde voor non-maximum suppression aanpassen om de balans tussen precisie en recall te optimaliseren. Voor onze tankdetectie-toepassing is een hogere recall wenselijk om ervoor te zorgen dat we geen tanks missen, zelfs als dit betekent dat we hierbij mogelijks meer false positives krijgen.

4. Model trainen

4.1. Databron

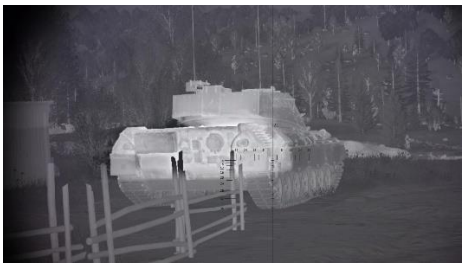
Een van de grootste uitdagingen is om voldoende trainingsdata te verkrijgen om het detectiemodel te trainen. Thermische beelden van militaire voertuigen zijn schaars en zeer moeilijk te verkrijgen. Ze worden vaak beschermd door militaire geheimhouding, zijn kostbaar om te produceren en vereisen gespecialiseerde en dure apparatuur om te produceren. Om deze uitdaging te overwinnen hebben we gekozen voor een andere benadering waarbij we gebruik maken van de populaire militaire simulatiegame "War Thunder" als databron. War Thunder is een free-to-play multiplayer oorlogssimulatiespel ontwikkeld door Gaijin Entertainment, dat bekend staat om zijn relatief accurate representatie van militaire voertuigen, inclusief tanks uit verschillende tijdperken. Het spel biedt ook een nachtzichtmodus aan die we kunnen gebruiken om zelf warmtebeelden te genereren, vergelijkbaar met echte warmtebeeldtechnologie.

War Thunder biedt honderden verschillende type tanks aan, zowel bestaande als volledig fictieve modellen [14]. De meeste ervan zijn lichte varianten van echte modellen. Om een proof-of-concept model van hoge kwaliteit te bouwen is het belangrijk dat we slechts een beperkt aantal verschillende tanks gebruiken, en dat de gekozen tanks voldoende verschillen van elkaar zodat er een duidelijk zichtbaar onderscheid gemaakt kan worden. Een ander probleem dat zich voordoet bij de dataverzameling is het vooruitgangsysteem van War Thunder. In het spel behoort elke tank tot een bepaalde rang. Een rang gaat van I tot VIII en geeft een ruwe indicatie over hoe sterk een tank is. Rang I staat voor oudere, zwakkere modellen die voornamelijk tijdens Wereldoorlog 2 gebruikt werden terwijl rang VIII dan weer bestaat uit de sterkste en nieuwste modellen die vandaag de dag worden ingezet. Als nieuwe speler heb je enkel toegang tot tanks uit rang I en moet je progressief per rang meerdere tanks vrijspelen om toegang te krijgen tot een hogere rang. Dit kan echter zeer lang duren en het probleem hier is dat de vroegste tank die toelaat om het spel in nachtzichtmodus te spelen een tank uit rang V is. Je zou het spel dus honderden uren moeten spelen opdat je nog maar zou kunnen beginnen met het verzamelen van data. Sinds enkele maanden geleden hebben de makers van het spel echter een nieuwe functionaliteit toegevoegd die dit probleem omzeilt. Door de mogelijkheid toe te voegen om een ‘testrit’ te maken kunnen nieuwe spelers nu tanks uit eender welke rang uitproberen, waaronder ook tanks die uitgerust zijn met een nachtzichtmodus. Het enige nadeel bij deze oplossing is dat de omgeving waar de testrit plaatsvindt een vastgelegde omgeving waar verschillende ‘dummy’ tanks opgesteld zijn als soort schietbaan. We hebben zelf echter geen controle over welke tanks beschikbaar zijn en hoe deze gepositioneerd zijn. Tabel 4.1 toont de verschillende tanks die zich in deze testomgeving bevinden en waarvan we op deze methode data kunnen verzamelen.

Afbeelding	Type	Oorsprong	Introductiejaar	Actief
	Leopard A1A1	Duitsland	1965	Ja
	Centurion MK 10	Verenigd Koninkrijk	1946	Ja (varianten)
	M60A1 RISE	Verenigde Staten	1959	Ja
	Chieftain MK 10	Verenigd Koninkrijk	1966	Nee (tot jaren 90)
	MBT-70	Verenigde Staten, West-Duitsland	1960 (Ontwikkeling)	Nee (prototype)

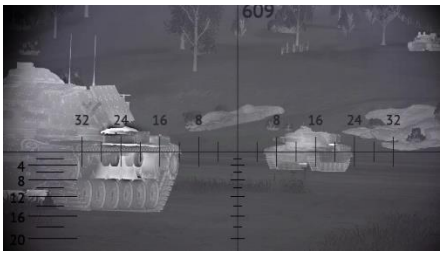
Tabel 4.1: Beschikbare tanks in de testritomgeving [14]

Het verzamelen van de data gebeurt door een schermopname te maken terwijl het spel gespeeld wordt in nachtzichtmodus en de speler een traject volgt dat 360° loopt rond elke tank die we willen classificeren, zowel van een korte als een lange afstand. Tijdens dit traject komen diverse omgevingselementen in beeld die bedoeld zijn om een realistisch scenario na te bootsen. Ook zorgen we ervoor dat we voldoende variatie hebben in onze trainingsdata. We observeren de tanks vanuit verschillende hoeken en zorgen voor voldoende ruis. Uit de schermopname worden vervolgens individuele frames geëxtraheerd, wat resulteert in duizenden thermische beelden van alle verschillende type tanks vanuit diverse perspectieven. Figuur 4.1 toont enkele voorbeelden van deze frames.



Voorbeeldframe 1

Voorbeeldframe 2



Voorbeeldframe 3 Voorbeeldframe 4

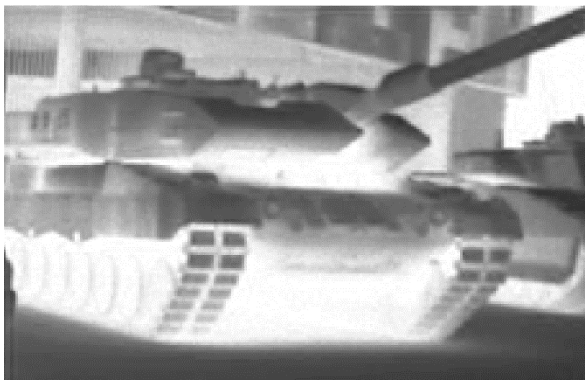
Figuur 4.1: Voorbeeldframes van de trainingsdata

Deze frames vormen de basis van onze trainingsdata en worden zorgvuldig gesorteerd in mappen per tankmodel. Voor elke afbeelding worden manueel bounding box-annotaties opgesteld die de exacte locatie van de tank(s) in het beeld markeren. Dit werd gedaan aan de hand van de open-source tool “Label Studio” en resulteerde uiteindelijk in 2874 geannoteerde frames. Deze beelden worden vervolgens omgezet in een formaat die geschikt is voor het trainen van YOLOv11.

4.2. Beperkingen van de data

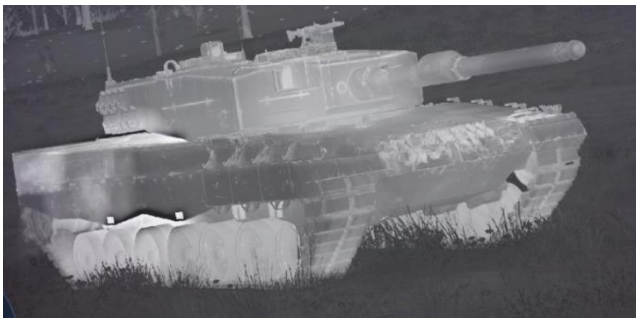
Hoewel War Thunder ervoor bekend staat om realistische tanks aan te bieden, zijn de warmtebeelden van deze tanks minder realistisch. Het is dus belangrijk om de inherente beperkingen van deze benadering te erkennen en te begrijpen hoe deze de resultaten kunnen beïnvloeden. De thermische weergave in War Thunder is een simulatie die, hoewel gebaseerd op realistische principes, niet volledig overeenkomt met echte warmtebeeldtechnologie. De game modelleert warmtesignaturen op een vereenvoudigde manier, waarbij niet alle subtiele temperatuurvariaties worden weergegeven die in echte warmtebeelden zichtbaar zouden zijn. De temperatuurverschillen tussen verschillende onderdelen van de tank zoals motor, loopwerk en geschutskoepel zijn in werkelijkheid veel genuanceerder dan in de simulatie, waar slechts een algemene gradiënt de vorm van de tank omvat en minder onderscheid gemaakt wordt tussen warme en koude onderdelen. In echte warmtebeelden beïnvloedt de omgeving bovendien sterk hoe objecten worden weergegeven. Factoren zoals zonnestraling die bepaalde delen opwarmt, recente activiteit van de tank resulterend in een warmere motor en de algemene omgevingstemperatuur hebben een complexe invloed die in de game slechts beperkt wordt gesimuleerd. De figuur hieronder toont enkele verschillen tussen realistische warmtebeelden en de beelden uit War Thunder.

(a) Leopard 2A4 [10]



(b) Warmtebeeld Leopard 2A5

(c) Leopard 2A4 (War Thunder)



(d) Warmtebeeld Leopard 2A4 (War Thunder)

Figuur 4.2: Verschillen War Thunder en realiteit

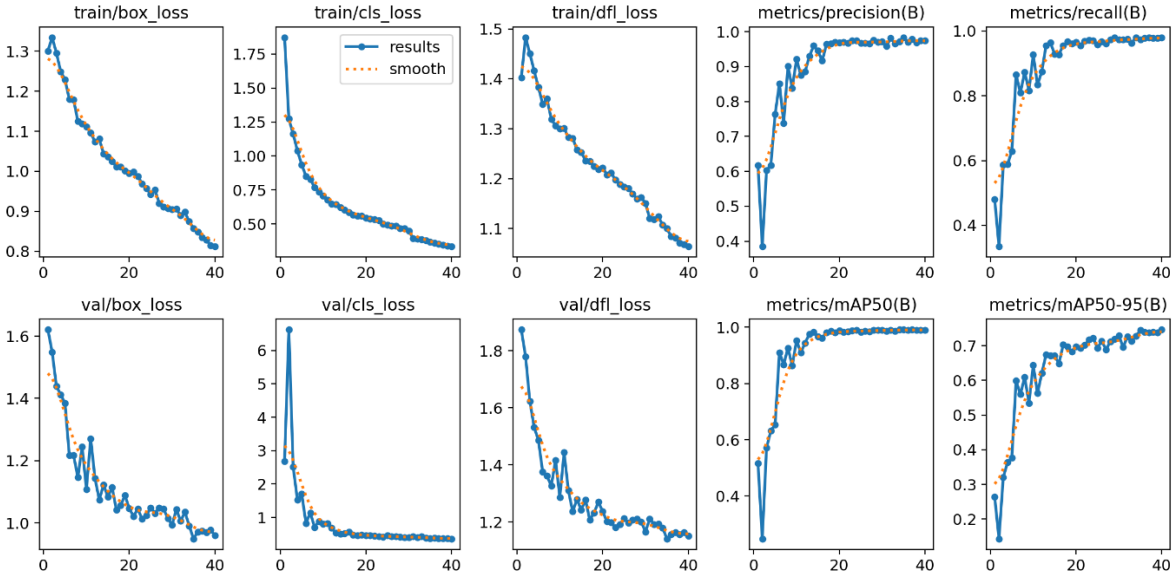
Ook bevinden er zich op het scherm heel wat HUD-elementen (*heads-up display*) zoals een mini-map in de hoek, een gezondheidsbalk, spelersnamen en andere items om de speler te helpen. De meeste hiervan die impact kunnen hebben op onze data kunnen echter uitgeschakeld worden, op het dradenkruis na.

Deze beperkingen benadrukken het proof-of-concept karakter van dit onderzoek. Het primaire doel is niet om een direct inzetbaar militair detectiesysteem te ontwikkelen, maar om de haalbaarheid van de benadering te demonstreren en een methodologisch kader te bieden voor toekomstig onderzoek wanneer reële warmtebeelden beschikbaar worden. Ondanks deze beperkingen biedt de War Thunder-dataset een waardevolle bron die een sterke basis vormt voor een proof-of-concept demonstratie.

4.3. Trainen

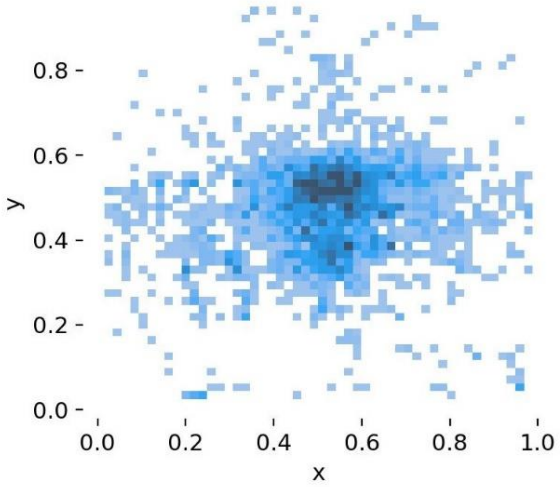
Het finale YOLO-model is getraind op een Nvidia RTX 3060TI over 40 epochs met afbeeldinggrootte 640x640 pixels. Hierbij is een train-test split van 80-20 gehanteerd met 2299 afbeeldingen als testset en 575 afbeeldingen als validatieset. Figuur 4.3 toont het gedrag van het model over de verschillende trainingsintervallen. De bovenste rij behandelt de trainingsset en de onderste de validatieset. Met box_loss (Box Loss) wordt de accuraatheid van de opgestelde bounding boxes in vergelijking met de actuele waardes gemeten. Hoe lager de box loss hoe meer deze twee overeenkomen. De cls_loss (Class Loss) is een maatstaf voor hoe accuraat de classificatie van de objecten is. Opnieuw streven we hier naar een lagere loss, aangezien de loss aantoont hoevaak het model foutief was. De dlf_loss (Distribution Focal

Loss) wordt ook gebruikt om de Bounding Boxes te verfijnen, alleen ligt hier de focus op onderscheid maken tussen objecten die hard op elkaar lijken. Een lagere loss is hier ook wenselijk.



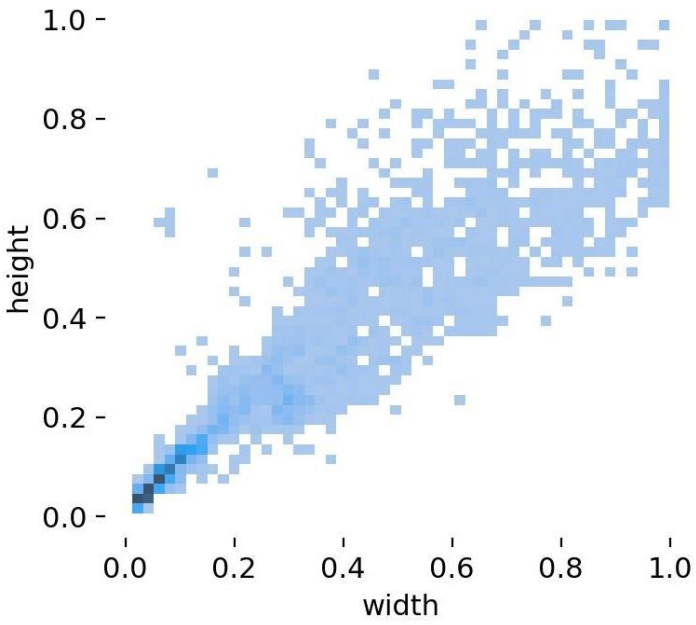
Figuur 4.3: Grafieken over het gedrag doorheen de trainingsintervallen

Figuur 4.4 toont een scatterplot die de distributie van de centrale posities (x,y-coördinaten) van alle gedetecteerde tanks in de dataset weergeeft. Elke punt vertegenwoordigt het middelpunt van een voorspelde bounding box. De hogere dichtheid (donkerblauw) in het midden geeft aan dat tanks meestal in het centrum van de afbeeldingen voorkomen. Op eerste zicht lijkt het dat dit invloed zou kunnen hebben op hoe goed het model tanks detecteert die aan de randen van het scherm verschijnen en dat je idealiter een meer gelijkmatige verdeling hebt over het hele coördinatenstelsel. Dit is echter niet het geval. Doordat YOLO de inputafbeeldingen in verschillende gridcellen opdeelt en elke cel apart behandelt is dit geen probleem voor de detectie van randgevallen.



Figuur 4.4: Distributie van centrale bounding box posities

Figuur 4.5 toont een grafiek over de relatie tussen breedte en hoogte van de bounding boxes. Er is een duidelijke correlatie zichtbaar, wat aangeeft dat de verhoudingen van tanks in de dataset consistent zijn - grotere tanks zijn zowel breder als hoger, wat logisch is. Er zijn ook enkele outliers zichtbaar waar een tank met zeer kleine breedte en grote hoogte is vastgesteld. Dit kan verklaard worden door tanks die slechts deels zichtbaar waren omwille van één of meer omgevingselementen dat de tank bedekte, zoals een schuur of boom. De donkerblauwe concentratie nabij de oorsprong suggereert dat er veel kleine tanks in de dataset zitten (tanks die in de verte te zien zijn), terwijl de verspreiding naar rechtsboven duidt op tanks van verschillende groottes. Deze informatie helpt het model om beter te worden in het voorspellen van nauwkeurige bounding boxes voor tanks van verschillende afmetingen.



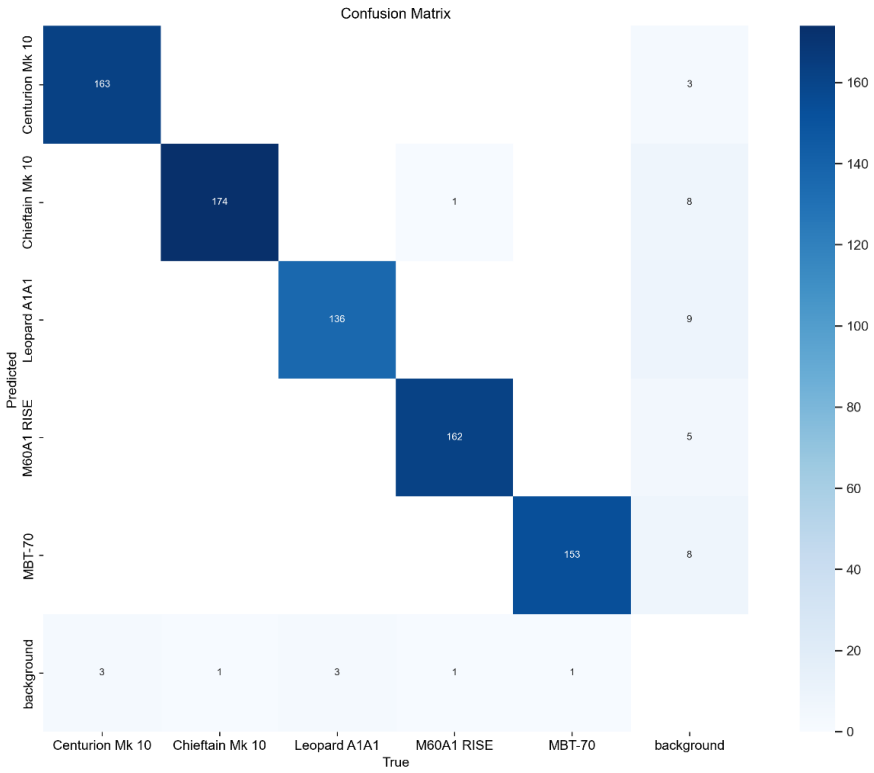
Figuur 4.5: Verhouding breedte en hoogte van bounding boxes

5. Evaluatie

5.1. Evaluatie van model

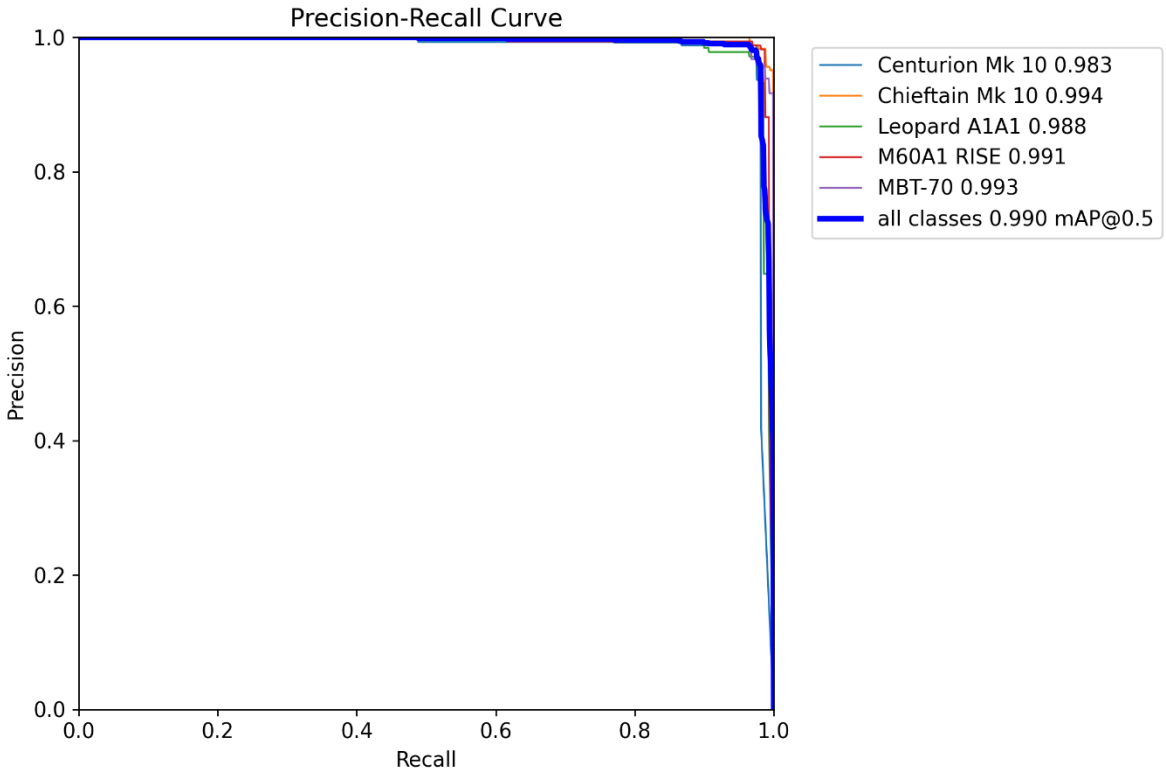
De prestaties van het YOLOv11l-model zijn uitvoerig geëvalueerd met behulp van standaard metrieken voor computervisie. Wanneer het model als realtime overlay wordt ingezet zonder CUDA-drivers detecteert het aan een snelheid van 2 à 3 FPS. Wanneer CUDA-drivers ingeschakeld worden stijgt deze snelheid naar een acceptabele 11 à 13 FPS.

De confusion matrix in Figuur 5.1 toont de classificatieprestaties over de vijf gedefinieerde tanktypes geëvalueerd op de validatieset. De primaire diagonaal van de matrix vertoont hoge waarden, met Centurion Mk 10: 163, Chieftain Mk 10: 174, Leopard A1A1: 136, M60A1 RISE: 162 en MBT-70: 153, wat duidt op uitstekende classificatienauwkeurigheid voor alle tankklassen. Het totaal aantal voorspelde tanks volgens de confusion matrix is enigszins groter dan het aantal frames in onze validatieset, wat aantoont dat het model correct in staat is om meerdere tanks in één frame te interpreteren. De belangrijkste misclassificaties doen zich voor bij Leopard A1A1, met 9 gevallen waarbij deze als "background" werd geclassificeerd, Chieftain Mk 10, met 8 gevallen van verkeerde classificatie als "background", en MBT-70, eveneens met 8 misclassificaties als "background". Opvallend is dat misclassificaties voornamelijk optreden tussen de tanktypes en de achtergrondklasse, met minimale verwisseling tussen verschillende tanktypes onderling. Dit suggereert dat het model uitstekend onderscheid maakt tussen de verschillende tankmodellen, maar soms moeite heeft met het detecteren van tanks tegen complexe achtergronden.



Figuur 5.1: Confusion matrix

De precision-recall curves in Figuur 5.2 bieden aanvullend inzicht in de performantie. Alle tanktypes vertonen uitzonderlijk hoge Area Under Curve (AUC) waarden, variërend van 0.983 (Centurion Mk 10) tot 0.994 (Chieftain Mk 10). De gemiddelde $mAP_{0.5}$ over alle klassen bedraagt een indrukwekkende 0.990. De curves tonen een bijna perfecte precisie (boven 0.98) over het gehele bereik van recall-waarden tot bijna 0.9, wat duidt op een model dat zeer weinig false positives genereert. De abrupte daling in precisie bij hoge recall-waarden (>0.9) wijst op een duidelijke beslissingsgrens in het model, wat gunstig is voor praktische toepassingen omdat het een natuurlijke drempelwaarde suggereert voor betrouwbare detecties.



Figuur 5.2: Precision-Recall curves

Op basis van individuele klassen presteert het model het best bij het identificeren van de Chieftain Mk 10 (0.994) en MBT-70 (0.993), mogelijk vanwege hun meer onderscheidende thermische profielen. De Centurion Mk 10 (0.983) heeft de relatief laagste, maar nog steeds uitstekende, precisiewaarde.

De resultaten tonen uitzonderlijk niveau van nauwkeurigheid, wat de geschiktheid van het model voor deze specifieke taak bevestigt. De bijna perfecte precisie- en recall-waarden, samen met de minimale verwarring tussen klassen, demonstreren de effectiviteit van de gekozen modelarchitectuur en trainingsmethodologie voor deze specifieke toepassing. Wanneer het model als realtime overlay wordt ingezet zonder CUDA-drivers detecteert het aan een snelheid van 2 à 3 FPS. Wanneer CUDA-drivers ingeschakeld worden stijgt deze snelheid naar een acceptabele 11 à 13 FPS.

5.2. Overfittingsrisico's

De uitzonderlijk hoge nauwkeurigheidsmetrieken van het model, met precisie- en recall-waarden boven de 0,98 voor alle tanktypen, kunnen ook een signaal zijn van potentiële overfitting. Dit risico wordt versterkt door de specifieke kenmerken van onze dataverzamelings- en splitsingmethodologie.

Onze dataset bestaat uit frames die zijn geëxtraheerd uit schermopnamen van War Thunder-gameplay. Deze sequentiële aard van de gegevensverzameling creëert een fundamenteel probleem: opeenvolgende frames uit dezelfde opname vertonen zeer hoge visuele correlatie. Wanneer een tank gedurende enkele seconden wordt gefilmd, kunnen tientallen opeenvolgende frames worden geëxtraheerd die vrijwel identiek zijn, met slechts minieme verschillen in positie, belichting of achtergrond. Voor de evaluatie van ons model hanteren we een traditionele willekeurige splitsing van datasets in trainings- en validatiesets. Bij een typische 80/20-verdeling is het zeer waarschijnlijk dat van een reeks van 10 bijna identieke frames er 8 in de trainingsset en 2 in de validatieset terechtkomen. In essentie betekent dit dat het model wordt gevalideerd op beelden die vrijwel identiek zijn aan de trainingsbeelden, wat mogelijks leidt tot kunstmatig opgeblazen evaluatiemetrieken. Dit concept wordt (overdreven) geïllustreerd in de onderstaande figuur.

Figuur 5.3: Batch van validatieframes

Dit scenario verklaart mogelijk waarom ons model zo'n indrukwekkende prestaties toont. Het model heeft mogelijk niet de onderliggende kenmerken geleerd die tanks onderscheiden, maar eerder de specifieke pixelpatronen van de trainingsbeelden gememoriseerd. Dit zou betekenen dat het model uitstekend presteert op onze validatieset omdat deze effectief "gezien" is tijdens de training, maar mogelijk veel slechter zou presteren op werkelijk nieuwe beelden.

De bijna perfecte scheiding tussen tankklassen in de confusion matrix ondersteunt deze hypothese verder. In echte detectiesystemen is een bepaalde mate van verwarring tussen visueel vergelijkbare klassen normaal, zelfs bij goed presterende modellen. De afwezigheid hiervan suggereert dat het model mogelijk geen generaliserende kenmerken heeft geleerd, maar in plaats daarvan specifieke voorbeelden heeft gememoriseerd. Om na te gaan in welke mate dit het geval is, is het model ook getest op een aparte schermopname waarvan de tanks uit andere perspectieven geobserveerd zijn aldus de frames licht verschillen, maar ook hier toont het model quasi perfecte performantie.

Voor een betrouwbaardere evaluatie zou een alternatieve strategie voor datasplitsing noodzakelijk zijn. In plaats van willekeurige frame-gebaseerde splitsing zou een opname-gebaseerde splitsing meer geschikt zijn, waarbij volledige opnamesessies exclusief aan de trainings- of de validatieset worden toegewezen. Dit zou garanderen dat de validatieset werkelijk nieuwe beelden bevat vanuit andere hoeken, afstanden en omgevingscondities dan de trainingsset. Maar doordat we ons realtime model vervolgens getest hebben op een nieuwe, unieke validatieset lijkt de kans zeer groot dat evaluatiemetrieken via deze methode quasi hetzelfde zouden zijn aangezien we ook hier quasi perfecte performantie ondervonden.

5.3. Limitaties van het model

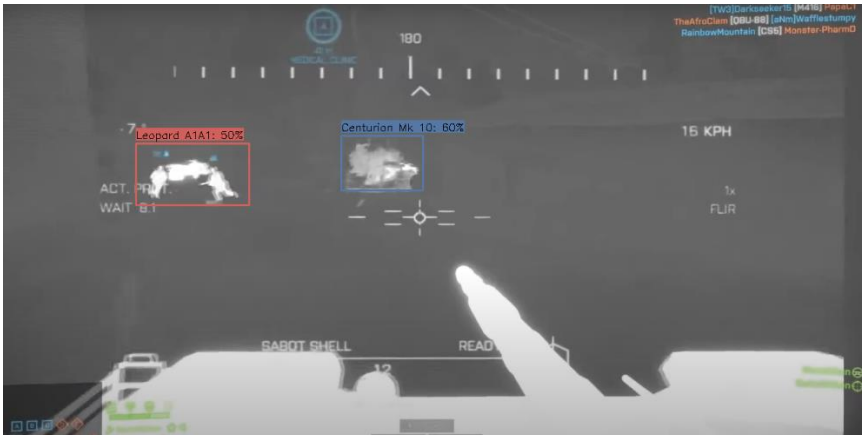
Ondanks dat het model als proof-of-concept dient en wegens de aard van de data niet rechtstreeks ingezet kan worden in echte scenario's is het alsnog interessant om op zoek te gaan naar de limieten van het model. Vermits men een lagere grenswaarde toepast over wanneer een object gedetecteerd wordt als tank zou het nog steeds mogelijk moeten zijn om tanks die zich buiten de War Thunder omgeving bevinden al dan niet in beperkte mate te detecteren. Om dit te verkennen heb ik een collectie opgesteld van warmtebeelden van tanks uit zowel andere videospellen als echte beelden die openbaar gemaakt zijn.

Het is belangrijk te begrijpen dat er zeer weinig externe warmtebeelden beschikbaar zijn over de verschillende tanks waar ons model specifiek op getraind is. Daarom ligt de focus hierbij naar onderzoek in welke mate het model in staat is om nieuwe tanks juist te detecteren in plaats van het effectief correct labelen van deze tanks. Ook hebben we bij de verzamelde beelden over andere videospellen weinig controle over welke HUD-elementen getoond worden en kunnen deze de werking van ons model verstoren.

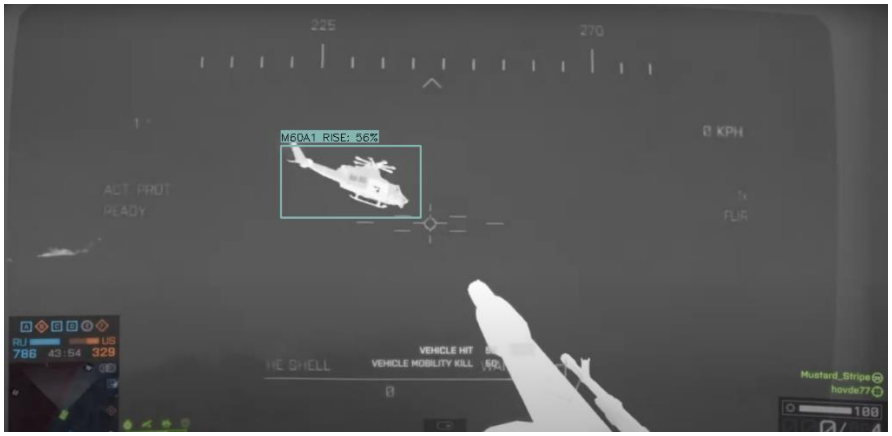
5.3.1. Toepassing op andere videospellen

Een ander videospel dat ons toelaat om in een nachtzichtmodus militaire tanks te observeren is de first-person shooter 'Battlefield 4' uit 2013. Het aanbod is hier wel beperkter aangezien Battlefield 4 slechts vier verschillende tanks aanbiedt. Ook de warmtebeelden van deze tanks zijn minder gedetailleerd dan deze uit War Thunder. In tegenstelling tot de data waar het model op getraind is, wordt er bij deze tanks geen enkele rekening gehouden met de interne warmteverdeling. Wanneer de speler nachtzichtmodus inschakelt wordt er simpelweg een eentonige lichtfilter toegepast over de hele omtrek van de tank die de speler observeert. Desondanks de verschillen in kwaliteit is het model instaat om tanks te detecteren. Hierbij accepteren we echter een hoger aantal false positives. De onderstaande figuren tonen enkele bevindingen van het realtime model toegepast op beelden uit Battlefield 4.

Figuur 5.4: Een tank wordt successvol gedetecteerd [8]



Figuur 5.5: Een brandende tank wordt successvol gedetecteerd (rechts) terwijl een quad met twee spelers errond foutief gedetecteerd is (links) [8]



Figuur 5.6: Een helikopter wordt foutief gedetecteerd [9]

In Figuur 5.4 zien we een succesvolle detectie van een tank die door het model wordt geïdentificeerd als een Leopard A1A1 met een betrouwbaarheidsscore van 62%. Ondanks de vereenvoudigde thermische weergave blijkt het model in staat te zijn om de algemene vormkenmerken te herkennen die kenmerkend zijn voor tanks. Deze succesvolle detectie, ondanks de lagere betrouwbaarheidsscore vergeleken met de War Thunder-resultaten, toont aan dat het model niet uitsluitend afhankelijk is van gedetailleerde interne thermische patronen maar ook vorm- en contourinformatie effectief benut.

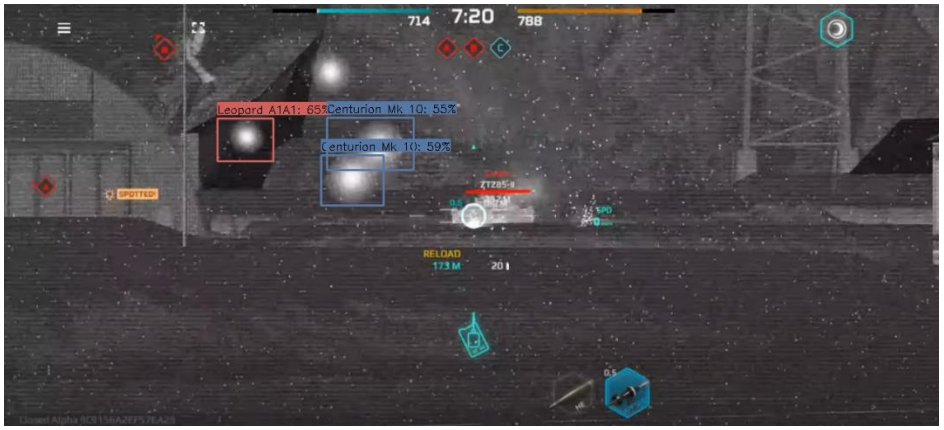
Figuur 5.5 toont een complexer scenario met zowel een correcte als een incorrecte detectie. Rechts wordt een brandende tank correct geïdentificeerd als een Centurion Mk 10 met 50% betrouwbaarheid. Tegelijkertijd wordt links een quad met twee spelers foutief herkend als een Leopard A1A1 met 50% betrouwbaarheid. Deze situatie is bijzonder informatief omdat het laat zien hoe het model reageert op een atypische thermische signatuur (een brandende tank) terwijl het ook een niet-tank object met een vagelijk vergelijkbare vorm en helderheid misclassificeert. Dit toont aan dat het model, wanneer geconfronteerd met beelden die sterker afwijken van de trainingsdata, terugvalt op oppervlakkige vormovereenkomsten, wat leidt tot een verhoogd risico op false positives.

Figuur 5.6 illustreert een ander probleem: een helikopter wordt foutief geïdentificeerd als een M60A1 RISE met 58% betrouwbaarheid. Deze misclassificatie bevestigt opnieuw dat het model moeite heeft met het onderscheiden tussen verschillende types militaire voertuigen. De relatief hoge betrouwbaarheidsscore bij deze foutieve detectie wijst erop dat bepaalde kenmerken van de helikopter, zoals mogelijk de langwerpige romp en uitstekende componenten, voldoende overeenkomsten vertonen met tankprofielen om het model substantieel te misleiden.

Een ander videospel met nachtzichtmodus en tanks waar we ons model op kunnen tussen is de mobiele game “MWT: Tank Battles”. Aangezien de focus van dit spel ligt op het simuleren van tankoorlog hebben we hier een ruimer aanbod van tanks. Ook zijn de warmtebeelden hier genuanceerder dan bij Battlefield, waardoor de beelden nauwer aansluiten aan onze trainingsdata. Het nadeel hier is dat dit spel gemaakt is om op een Android-GSM te spelen waardoor de schermresolutie lager is dan bij Battlefield of War Thunder. Hierdoor zijn HUD-elementen op het scherm mogelijks meer zichtbaar, wat de detecties van het model kan verstoren. De onderstaande figuren tonen enkele prestaties van het model op data uit MWT: Tank Battles.

Figuur 5.7: Een tank wordt successvol gedetecteerd [4]

Figuur 5.8: Nachtzichtmodus uitgeschakeld, een tank wordt niet gedetecteerd terwijl de voet van een vlagpaal kortstondig foutief gedetecteerd is [4]



Figuur 5.9: Afgevuurde kogels worden gedetecteerd als tank terwijl de tank zelf niet gedetecteerd wordt [4]

Figuur 5.7 toont een succesvolle detectie van een MBT-70 tank met een hoge betrouwbaarheidsscore van 73%. Dit positieve resultaat bevestigt opnieuw dat het model effectief kan generaliseren naar thermische beelden van tanks uit andere spelomgevingen, mits deze visueel voldoende overeenkomsten vertonen met de War Thunder-trainingsdata.

Figuur 5.8 toont een ook een interessante beperking van het model: wanneer de nachtzichtmodus is uitgeschakeld, faalt het model volledig in het detecteren van de duidelijk zichtbare tank in het beeld. Dit is enigszins logisch aangezien het model niet getraind is op deze data. In plaats daarvan wordt alleen de voet van een vlagpaal kortstondig en foutief geïdentificeerd als een Leopard A1A1, mogelijks door de grijze kleuren en vorm die licht overeenkomt met een tank.

In Figuur 5.9 zien we een interessante fout: afgevuurde kogels of projectielen worden foutief geïdentificeerd als tanks, terwijl de daadwerkelijke tank die de kogels afvuurt niet wordt gedetecteerd. De felle thermische signatuur van de projectielen vertoont blijkbaar genoeg

overeenkomsten met bepaalde kenmerken van tanks om het model te misleiden. Vermoedelijk ziet het model de kogels als tanks die zeer ver weg zijn en waar dus weinig details geobserveerd worden, terwijl de effectieve tank verstoord wordt door de HUD-elementen. Dit duidt erop dat het model mogelijk te sterk focust op algemene helderheidspatronen in plaats van op de specifieke morfologische eigenschappen die tanks onderscheiden.

Deze observatie brengt ons tot een nieuwe theorie rond mogelijkse overfitting die los staat van de eerder besproken overfitting door sequentiële frame-correlatie. Het model heeft mogelijk een simplistische detectiestrategie ontwikkeld waarbij elke licht gekleurde witte vlek automatisch wordt geclassificeerd als een tank, simpelweg omdat het tijdens de training uitsluitend tanks heeft waargenomen tegen donkere achtergronden. Deze benadering werkt uitstekend binnen de gecontroleerde War Thunder-omgeving waar tanks inderdaad de enige heldere objecten zijn, maar faalt zodra het model wordt geconfronteerd met andere warmtesignaturen zoals projectielen, helikopters, of andere thermische anomalieën die eveneens contrast vertonen tegen de achtergrond.

Om dit fundamentele probleem aan te pakken zou een tweestapsarchitectuur kunnen worden geïmplementeerd, bestaande uit twee opeenvolgende modellen. Een primaire detector zou eerst onderscheid maken tussen tanks en andere warmtesignaturen en daarmee een antwoord geven op de vraag "is er een tank aanwezig?". Dit eerste model moet getraind worden op een diverse dataset met zowel tanks als andere objecten met warmtesignaturen zoals helikopters, civiele voertuigen, afgevuurde projectielen en gebouwen. Dit zou dan fungeren als een filter om te bepalen of een warmtesignatuur daadwerkelijk afkomstig is van een tank. Vervolgens zou ons huidige YOLOv11-model dienen als het tweede model dat alleen wordt toegepast op objecten die door de primaire detector als tanks zijn geïdentificeerd, waarbij het zich uitsluitend kan concentreren op de classificatie tussen verschillende tankmodellen.

Deze aanpak zou de specificiteit van het systeem aanzienlijk kunnen verbeteren door het aantal false positives te reduceren, terwijl de hoge classificatienauwkeurigheid tussen tanktypen behouden blijft. De observatie van projectielen die als tanks worden geclassificeerd bevestigt dat deze tweestapsarchitectuur niet alleen theoretisch interessant is, maar praktisch noodzakelijk voor robuuste implementatie in real-world scenario's waar diverse warmtebronnen aanwezig zijn.

5.3.2. Toepassing op echte tanks

De onderstaande figuren tonen enkele merkwaardige bevindingen wanneer het realtime model losgelaten werd op echte warmtebeelden van tanks.

Figuur 5.10: True Positive [12]

Figuur 5.11: False Negative (links) en True Positve (rechts) [11]

Figuur 5.12: Twee False Positives [11]

Figuur 5.13: False Negative [12]

Het model biedt waardevolle inzichten in de praktische beperkingen en generaliseerbaarheid van het model. De resultaten tonen aan dat het model een zekere mate van transfervermogen bezit, maar ook enkele tekortkomingen vertoont die verdere ontwikkeling vereisen.

In Figuur 5.10 zien we een correcte detectie van een tank, wat aantoont dat het model in staat is om fundamentele tanksilhouetten te herkennen over verschillende databronnen heen, net zoals bij de andere videospellen.

Figuur 5.11 illustreert echter een inconsistentie in de detectiecapaciteit. Hoewel de tank rechts gedetecteerd is, wordt een visueel zeer vergelijkbaar voertuig links niet gedetecteerd. Deze inconsistentie kan worden toegeschreven aan subtiele verschillen in warmtesignatuur, oriëntatie, of beeldkwaliteit die de beslissingsgrens van het model beïnvloeden. Dit toont aan dat het model gevoelig is voor kleine variaties die in een gameomgeving mogelijk minder uitgesproken zijn dan in reële thermische beelden.

De false positives in Figuur 5.12 illustreren een gelijkaardig probleem als bij Battlefield 4, waarbij helikopters worden geclassificeerd als M60A1 RISE tanks. Dit duidt erop dat het model sterk vertrouwt op algemenere vormkenmerken van militaire voertuigen, en wellicht niet voldoende heeft geleerd om de specifieke kenmerken te onderscheiden die tanks differentiëren van andere militaire voertuigen met vergelijkbare warmtesignaturen. De

afwezigheid van helikopters in de trainingsdata, alsook de eerder besproken theorieën rond overfitting kunnen een verklaring zijn waarom het model deze misclassificaties maakt.

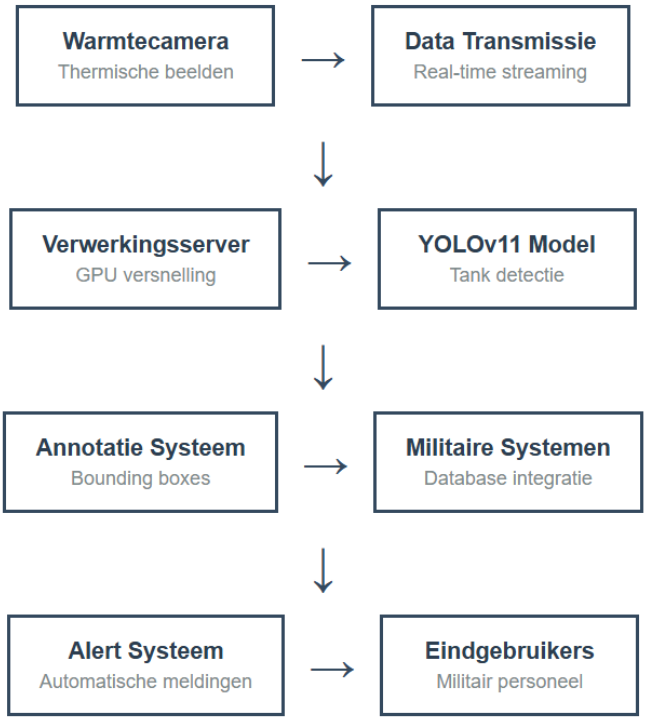
Figuur 5.13 illustreert een van de meest uitdagende het model, namelijk omgevingsinterferentie. De tank op de afbeelding rijdt weg van de camera, waardoor achter de tank heel wat stof en uitstootgassen verschijnen. Dit creëert een thermische 'wolk' die het kenmerkende profiel verstoort, wat resulteert in een false negative. Dit type dynamische camouflage, waarbij de thermische signatuur van het voertuig versmelt met omgevingsfactoren, komt zelden voor in statische spelbeelden maar is cruciaal in reële operationele omstandigheden.

Het model mist dus contextbegrip voor verschillende militaire voertuigtypes, wat leidt tot verwarring tussen tanks en helikopters. De thermische karakteristieken van echte voertuigen zijn complexer en variabeler dan in gesimuleerde omgevingen, waarbij factoren zoals motorwarmte, uitlaatgassen en omgevingsinteractie een grotere rol spelen. Hoewel de trainingsdata heel wat variatie in voertuigoriëntaties bevat, ontbreekt er nog steeds een grote variatie in omgevingscondities en gedeeltelijke occlusies die in reële scenario's veel prominenter voorkomen.

Ondanks deze beperkingen toont het model veelbelovende capaciteiten om als basis te dienen voor verdere ontwikkeling. De succesvolle detecties suggereren dat de fundamentele principes van thermische tankdetectie zijn vastgelegd, maar dat het model verfijning nodig heeft met echte thermische beelden. Een hybride benadering, waarbij het model initieel wordt getraind op ruim beschikbare gesimuleerde data en vervolgens wordt fijngestemd met een kleinere set echte thermische beelden, zou de prestatiekloof kunnen overbruggen.

5.4. Realtime inzetting

Figuur 5.14 illustreert de operationele architectuur van het ontwikkelde model in een mogelijkse real-world militaire omgeving. Het systeem volgt een lineaire workflow die begint met het vangen van de ruwe thermische beelden via warmtecamera’s in het betreffende operatiegebied. Deze beelden worden via beveiligde datatransmissie doorgestuurd naar een centrale verwerkingsserver die uitgerust is met GPU-versnelling voor optimale prestaties. Hier wordt het getrainde YOLOv11-model ingezet om tanks te detecteren en classificeren binnen de ontvangen thermische beelden. De gedetecteerde objecten worden vervolgens door het annotatiesysteem voorzien van bounding boxes en classificatielabels, waarna deze geannoteerde data wordt geïntegreerd in bestaande militaire systemen en databases. Het systeem is idealiter uitgebouwd met een alertsysteem dat prioritaire meldingen genereert en deze via push-notificaties doorzendt naar het relevante militaire personeel wanneer een tank gedetecteerd wordt.



Figuur 5.14: Realtime inzet van het model

Deze operationele architectuur verschilt aanzienlijk van onze huidige proof-of-concept demonstratie, waarbij we lokaal op één computer het War Thunder-spel uitvoeren terwijl we het scherm live capteren en rechtstreeks via het YOLOv11-model annotaties toevoegen zonder externe datatransmissie of systeem-integratie. In de real-world implementatie vereist het systeem robuuste netwerkinfrastructuur, beveiligde communicatieprotocollen, en integratie met complexe militaire command-and-control systemen, terwijl ons proof-of-concept zich beperkt tot lokale verwerking van gesimuleerde warmtebeelden.

Dit systeem kan worden ingezet in verschillende militaire scenario's, zoals grensbewaking, perimeterbescherming van militaire installaties, of ondersteuning van verkenningsoperaties. De realtime detectiecapaciteit maakt het mogelijk om onmiddellijk te reageren op gedetecteerde tankbewegingen, terwijl de geautomatiseerde classificatie helpt bij het onderscheiden tussen verschillende dreigingsniveaus. Door de integratie met bestaande militaire command-and-control systemen kan deze informatie naadloos worden opgenomen in de operationele besluitvorming.

5.5. Technology Readiness Level (TRL)

Technology Readiness Level (TRL) is een systematische meetmethode die oorspronkelijk door NASA is ontwikkeld en nu breed wordt toegepast om de maturiteit van opkomende technologieën te evalueren. Het bestaat uit negen niveaus, waarbij niveau 1 staat voor het basisonderzoeksstadium waar fundamentele principes worden geobserveerd en niveau 9 voor een volledig operationeel systeem dat zijn waarde en betrouwbaarheid in de praktijk heeft bewezen. Elk TRL-niveau vertegenwoordigt een mijlpaal in het ontwikkelingsproces van een

technologie, waarbij hogere niveaus duiden op toenemende technische en commerciële gereedheid. Deze classificatie helpt organisaties om de ontwikkelingsstatus van nieuwe technologieën te beoordelen, risico's in te schatten en gefundeerde beslissingen te nemen over verdere investeringen en implementatie.

In de context van ons tankdetectiesysteem op basis van YOLOv11 hebben we het huidige TRL-niveau geëvalueerd op 3. Dit betekent dat we binnen de context van softwareontwikkeling werken met een werkend proof-of-concept die is gevalideerd in een developmentomgeving maar nog niet in een relevante operationele omgeving ingezet kan worden. Deze beoordeling is gebaseerd op meerdere factoren: we hebben een functionerend proof-of-concept systeem dat succesvol tanks kan detecteren en classificeren op basis van warmtebeeldsignaturen, maar dit is gerealiseerd met gesimuleerde data uit de War Thunder-game in plaats van met echte warmtebeelden. Het model toont goede prestaties binnen de grenzen van onze testomgeving, met een acceptabele nauwkeurigheid en verwerkingssnelheid voor zowel statische als realtime analyse, maar het model faalt om met hoge performantie en betrouwbaarheid echte tanks te detecteren en classificeren.

Om deze technologie naar hogere TRL-niveaus te brengen en een volledig operationeel systeem te realiseren, zijn verschillende kritieke stappen noodzakelijk. Ten eerste moet het model opnieuw worden getraind met een uitgebreide dataset van authentieke militaire warmtebeelden van diverse tankmodellen, vastgelegd onder uiteenlopende operationele omstandigheden, terreintypen, weersomstandigheden, afstanden en hoeken. Vervolgens moet er een softwaresysteem gebouwd worden die het getrainde model kan integreren met bestaande militaire systemen, wat optimalisaties vereist voor de specifieke hardware waarop het uiteindelijk zal draaien. De prestaties van het model moeten worden gevalideerd via een reeks rigoureuze tests, beginnend met gecontroleerde omgeving met echte tanks en warmtecamera’s en geleidelijk uitbreidend naar veldtests in realistische scenario's. Daarnaast moeten robuustheidsmaatregelen worden geïmplementeerd om de betrouwbaarheid onder wisselende omstandigheden te waarborgen, en moet er een uitgebreid evaluatieprotocol worden ontwikkeld om de nauwkeurigheid en betrouwbaarheid in verschillende operationele contexten te meten. Ten slotte zijn uitgebreide documentatie, trainingsprotocollen en onderhoudsprocedures essentieel om de technologie effectief te kunnen implementeren in real-world defensietoepassingen.

6. Conclusie

Deze bachelorproef had als doel om de mogelijkheden van kunstmatige intelligentie te onderzoeken voor de detectie en classificatie van tanks op basis van hun warmtesignaturen. Door gebruik te maken van het YOLOv11 framework en gesimuleerde warmtebeelden uit de militaire simulatiegame "War Thunder" is een proof-of-concept systeem ontwikkeld dat zowel in statische als realtime omgevingen kan worden ingezet.

Het YOLOv11l-model heeft uitstekende prestaties geleverd binnen de context van de gebruikte dataset. Met een gemiddelde mAP0.5 van 0.990 over alle tanktypes en precision-recall curves die bijna perfecte precisie (boven 0.98) tonen over een breed bereik van recall-waarden, is het model zeer effectief gebleken in het detecteren van tanks in warmtebeelden. De confusion matrix toonde minimale verwarring tussen verschillende tanktypes, wat aangeeft dat het model goed in staat is om onderscheid te maken tussen de vijf getrainde tankmodellen.

Qua snelheid heeft het model acceptabele resultaten geleverd. Zonder CUDA-drivers detecteert het model tanks aan een snelheid van 2-3 FPS, terwijl met CUDA-ondersteuning dit stijgt naar 11-13 FPS, wat voldoende is voor realtime toepassingen. Deze balans tussen nauwkeurigheid en snelheid bevestigt de geschiktheid van het YOLOv11 framework voor deze specifieke toepassing.

Verschillende factoren hebben invloed op de prestaties van het systeem. De belangrijkste misclassificaties deden zich voor tussen tanktypes en de achtergrondklasse, wat suggereert dat het model soms moeite heeft met het detecteren van tanks tegen complexe achtergronden. De detailniveau en realisme van de warmtebeelden blijken van groot belang, zoals aangetoond door de verminderde prestaties bij toepassing op andere databronnen (Battlefield 4, MWT: Tank Battles, echte warmtebeelden).

De trainingsmethode en de wijze van datasplitsing hebben mogelijk geleid tot een risico op overfitting, waarbij het model mogelijk specifieke pixelpatronen heeft gememoriseerd in plaats van generaliserende kenmerken te leren. Dit wordt gesuggereerd door de bijna perfecte evaluatiemetrieken op de validatieset die bestaat uit frames die sterk correleren met de trainingsframes.

Het model presteert het best bij het identificeren van de Chieftain Mk 10 (0.994) en MBT-70 (0.993), mogelijk vanwege hun meer onderscheidende thermische profielen. De Centurion Mk 10 (0.983) toonde de relatief laagste, maar nog steeds uitstekende, precisiewaarde.

Bij toepassing op beelden buiten de trainingsomgeving bleek het model in staat om basisvormen van tanks te herkennen, maar toonde het inconsistenties in de detectiecapaciteit. Factoren zoals oriëntatie, afstand, omgevingsinterferentie (zoals stof en uitlaatgassen) en de aanwezigheid van andere militaire voertuigen (zoals helikopters) beïnvloedden de betrouwbaarheid significant.

Hoewel de War Thunder dataset een waardevolle bron vormt voor het experimenteren met en verfijnen van AI-detectietechnieken voor warmtebeelden, kent deze belangrijke beperkingen. De thermische weergave in het spel is een vereenvoudigde simulatie die niet alle subtiele temperatuurvariaties weergeeft die in echte warmtebeelden zichtbaar zouden zijn. Factoren zoals zonnestraling, recente activiteit van de tank, en de algemene omgevingstemperatuur hebben een complexe invloed die in de game slechts beperkt wordt gesimuleerd.

De tests op echte warmtebeelden toonden aan dat het model enige transfercapaciteit bezit, maar ook enkele tekortkomingen vertoont. Het model mist contextbegrip voor verschillende militaire voertuigtypes, wat leidt tot verwarring tussen tanks en helikopters. De thermische karakteristieken van echte voertuigen zijn complexer en variabeler dan in gesimuleerde omgevingen.

Het realtime systeem functioneert effectief tijdens het gamen met een verwerkingssnelheid van 11-13 FPS wanneer CUDA-drivers ingeschakeld zijn. Dit is voldoende voor praktische toepassingen, maar suggereert dat verdere optimalisatie nodig is voor hoogwaardige realtime prestaties. Voor optimale prestaties is grafische verwerkingshardware (GPU) met CUDA-ondersteuning sterk aangeraden.

Het huidige TRL-niveau van het ontwikkelde systeem is beoordeeld op niveau 3, wat aangeeft dat we een werkend proof-of-concept hebben dat is gevalideerd in een developmentomgeving maar nog niet in een relevante operationele omgeving ingezet kan worden.

Het systeem zou in de toekomst kunnen worden uitgebreid naar andere militaire voertuigen en objecten. Een hybride benadering, waarbij het model initieel wordt getraind op ruim beschikbare gesimuleerde data en vervolgens wordt fijngestemd met een kleinere set echte thermische beelden, zou de prestatiekloof kunnen overbruggen.

Verbeteringen in de dataverzamelings- en evaluatiemethodologie, zoals opname-gebaseerde datasplitsing in plaats van frame-gebaseerde splitsing, zouden leiden tot betrouwbaardere evaluatiemetrieken en mogelijk robuustere modellen. Daarnaast zou de integratie van temporele informatie uit videostromen de detectienauwkeurigheid verder kunnen verbeteren.

Dit onderzoek heeft aangetoond dat kunstmatige intelligentie, specifiek het YOLOv11 framework, veelbelovend potentieel biedt voor de detectie en classificatie van tanks op basis van warmtebeelden. Hoewel het ontwikkelde systeem nog op het niveau van een proof-of-concept staat (TRL 3), laten de resultaten zien dat de fundamentele principes van thermische tankdetectie zijn vastgelegd. De methode van het gebruik van gesimuleerde data uit gaming voor de initiële ontwikkeling en training biedt een innovatieve benadering die kan dienen als blauwdruk voor vergelijkbare projecten.

De uitdagingen en beperkingen die zijn geïdentificeerd, met name rond de generaliseerbaarheid naar echte warmtebeelden en de gevoeligheid voor omgevingsfactoren, wijzen op belangrijke aandachtsgebieden voor toekomstig onderzoek. Met verdere verfijning, training op authentieke beelden, en optimalisatie voor specifieke use-cases, heeft deze technologie het potentieel om bij te dragen aan verbeterde militaire verkenning en identificatie in zowel defensieve als civiele toepassingen.

Bibliografie

[1] Beirnaert, C. (z.d.). *Les 5. Performance Evaluation* [Presentatieslides; PDF]. Machine Learning & Forecasting (B-TM-YP0572), Mechelen, België. Canvas.

[2] Beirnaert, C. (z.d.*). Les 11. Neural Networks* [Presentatieslides; PDF]. Machine Learning & Forecasting (B-TM-YP0572), Mechelen, België. Canvas.

[3] Both, M., Demeester, E., & UHasselt. Faculteit Industriële Ingenieurswetenschappen. Opleiding master in de industriële wetenschappen. Elektronica-ICT degree granting institution. (2021). *Hand Localization Using YOLO on Depth Data*. UHasselt. Faculteit Industriële Ingenieurswetenschappen.

[4] Caydan. (2024, 15 oktober). M1A1 Abrams Gameplay | Testing Thermal Sight | MWT: Tank Battles [Video]. YouTube. https://www.youtube.com/watch?v=GRTzn-hjkKM

[5] DeepBean. (2023, 12 maart). How YOLO Object Detection Works [Video]. YouTube. https://www.youtube.com/watch?v=svn9-xV7wjk

[6] Jocher, G., Qiu, J., & Chaurasia, A. (2023). Ultralytics YOLO (Version 8.0.0) [Computer software]. https://github.com/ultralytics/ultralytics

[7] Manhaeve, R., De Raedt, L., De Grave, K., & KU Leuven. Faculteit Ingenieurswetenschappen. Opleiding Master in de ingenieurswetenschappen. Computerwetenschappen degree granting institution. (2017). *Realtime detectie van weggebruikers met één evaluatie van een diep neuraal netwerk*. KU Leuven. Faculteit Ingenieurswetenschappen.

[8] McCallihan. (2015, 8 november). Thermal Optic! Battlefield 4 Tank Thermal Gameplay. [Video]. YouTube. https://www.youtube.com/watch?v=saTJjf3QKng

[9] Mustard Stripe. (2016, 11 februari). Battlefield 4TM Tank nightvision heli kill [Video]. YouTube. https://www.youtube.com/watch?v=jydculLUlGs

[10] Pivoňka, M. (2024, 20 november). Leopard 2A4 tank. Czdefence. https://www.czdefence.cz/clanek/armada-porizuje-dalsich-14-tanku-leopard-2a4-73-tankovy-prapor-jich-tak-bude-mit-celkem-42

[11] Thermoteknix. (2017, 25 oktober). Fuji Firepower 2017 - TiCAM 600+ Thermal Imager [Video]. YouTube. https://www.youtube.com/watch?v=0-Nige5sayE

[12] Thermoteknix. (2019, 11 november). TiCAM 600 Main Battle Tank Live Firing [Video]. YouTube. https://www.youtube.com/watch?v=R-WTgdiP74w

[13] Tok, O. (2023, 10 juli). Infrared vs. Thermal Cameras: What are the Differences? a1securitycameras. https://www.a1securitycameras.com/blog/infrared-vs-thermal-cameras/

[14] War Thunder Wiki. (2011). Gaijin Games Kft. https://wiki.warthunder.com/

[15] Zaidi, S. S. A., Ansari, M. S., Aslam, A., Kanwal, N., Asghar, M., & Lee, B. (2022). A survey of modern deep learning based object detection models. Digital Signal Processing, 126, 103514. https://doi.org/10.1016/j.dsp.2022.103514