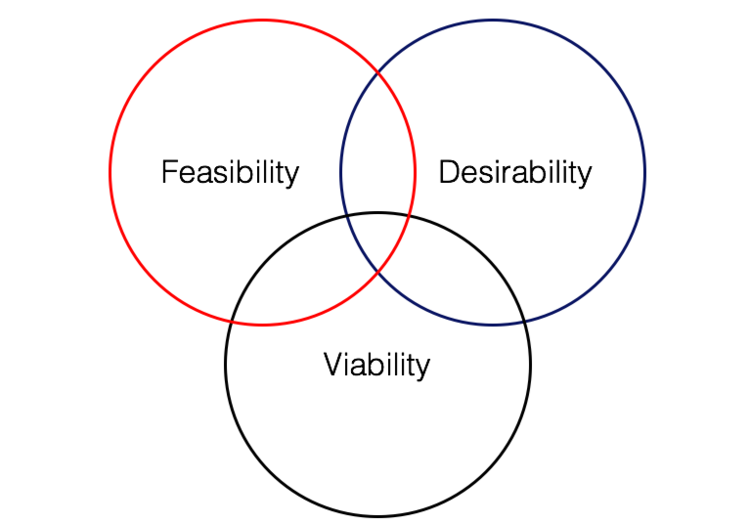
**Ontwerpproces dataset**  
1-2-2021



Docent:

Daniel Meinsma

Studenten:

Danique van Winden(19070136)

Olivia Boontjes(19053274)

Tom Dullaart(19079435)

Inhoud

[Inleiding 3](#_Toc62829678)

[Hoofdstuk 1: Ontwerp 4](#_Toc62829679)

[Values, norms en requirements met desirability, feasibility en viability 4](#_Toc62829680)

[Prioriteiten 5](#_Toc62829681)

[Hoofdstuk 2: realiseren classifi cation 6](#_Toc62829682)

[Dataset 1: foto’s van internet 6](#_Toc62829683)

[Dataset 2: 3D modellen van het internet. 6](#_Toc62829684)

[Dataset 3: onze zelfgemaakte dataset 7](#_Toc62829685)

[Hoofdstuk 3: testen classification. 9](#_Toc62829686)

[Requirements en gemeten waardes 9](#_Toc62829687)

[Meten false positives en negatives 10](#_Toc62829688)

[Methode 10](#_Toc62829689)

[Resultaten 11](#_Toc62829690)

[Conclusie 11](#_Toc62829691)

[Discussie 12](#_Toc62829692)

[Hoofdstuk 4: Overgang naar object detection 13](#_Toc62829693)

[Hoofdstuk 5: Accuraatheid 14](#_Toc62829694)

[Overfitting en Underfitting 14](#_Toc62829695)

[Accuraatheid meten 14](#_Toc62829696)

[Test 1: 15](#_Toc62829697)

[Test 2: 19](#_Toc62829698)

[Conclusie 26](#_Toc62829699)

[Hoofdstuk 6: Toekomst 27](#_Toc62829700)

## 

# Inleiding

De challenge van het LEGO-team is om een app te maken die gebruikt kan worden om een LEGO blokje te vinden in een berg van blokjes. Hiervoor is er onderzoek gedaan over hoe het beste de dataset en verwerking kunnen toepassen. Hierbij hebben is meteen een blik genomen op een inzicht van wat onze requirements zijn. Eerst is er een dataset gemaakt voor een classification model. Die is getest. Daarna is de dataset aangepast voor een object detection model door het toevoegen van bounding boxes.

In hoofdstuk 5 is er extra ingegaan op accuraatheid. Hierbij is er getest hoe accuraat het model is. Daarna is er geevalueerd hoe het de accuraatheid hoger kan. Daarna is het ontwerp van de dataset aangepast en uitgevoerd. Met de nieuwe dataset is het model opnieuw getraint en getest.

# Hoofdstuk 1: Ontwerp

## Values, norms en requirements met desirability, feasibility en viability

Het opstellen van de requirements waar aan het product dient te voldoen is gedaan aan de hand van de values en norms.[[1]](#endnote-2) Die zijn ingedeeld in drie aspecten desirability, feasibility, viability. Bij desirability gaat het erom dat het product gewenst wordt door de eindgebruiker, bij feasibility is het belangrijk dat het haalbaar is en bij viability gaat het erom dat het product winstgevend is.[[2]](#endnote-3)

|  |  |  |  |
| --- | --- | --- | --- |
| **Values:** | **Norms:** | **Requirements:** | **Desirability, Feasibility, Viability** |
| Herkenning van de legoblokjes | Snel | Kan te minste 30 fps aan | Feasible |
|  |  | Binnen 33 ms feedback geven | Feasible |
|  |  | Wanneer het blokje goed in beeld is kan het systeem het binnen 1 seconde vinden | Feasible |
|  | Accuraat | Geeft maximaal 10% false positives | Feasible |
|  |  | Mist maximaal 5% van de te zoeken blokken | Feasible |
| Gebruiksvriendelijkheid | Snel | Een nieuwe gebruiker moet gemiddeld binnen 30 seconden een blokje kunnen filteren | Desirable |
|  | Aantrekkelijk | 4 op de 5 gebruikers vinden de app er aantrekkelijk uit zien | Desirable |
|  |  | 4 op de 5 gebruikers vinden het fijn om met de app te werken | Desirable |
| Veilig | Privacy | De data wordt lokaal verwerkt | Viable |
|  |  | We vragen toestemming om de camera te openen | Viable |
|  |  | We slaan niet meer gegevens op dan nodig | Viable |
|  | Copyright | Geen dingen met de naam LEGO® publiceren zonder toestemming van LEGO® zelf | Viable |
| Aangeven van de legoblokjes | Duidelijk | 95% van de gebruikers kan zien wanneer een blokje wordt aangegeven | Desirable |
|  |  | De app maakt een geluidje wanneer hij een potentiële match heeft gevonden | Desirable |
|  |  | Er wordt weergegeven hoe groot de kans is dat de potentiële match correct is | Feasible |
|  | Snel | Er wordt binnen 0,1 seconde na het vinden een vierkant getekend | Feasible |

## Prioriteiten

We hebben ervoor gekozen om te accuraatheid van het model de prioriteit te maken. Als het model een false positive geeft kan de gebruiker misleid worden en een blokje pakken om er dan achter te komen dat het niet het blokje is die gezocht wordt. Als er een false negative is wordt het blokje niet gevonden en de gebruiker niet geholpen. Als er te veel false negatives zijn is het product waardeloos omdat het dan niet betrouwbaar is en de gebruiker beter zelf kan gaan zoeken. Daarom zal er bij de requirements extra focus liggen op de accuraatheid. Voor uitleg van de false positives en negatives zie “Meten false positives en negatives” in hoofdstuk 3.

We hebben niet voor transparantie gekozen omdat er geen spraken is van een gevoelig onderwerp waar bijvoorbeeld sprake van discriminatie kan zijn. Bij het maken van een AI die bijvoorbeeld gezichten kan detecteren zal het veel belangrijker zijn. Maar het bij het herkennen van plastic bouwblokken hoeft er geen rekening gehouden te worden met discriminatie.

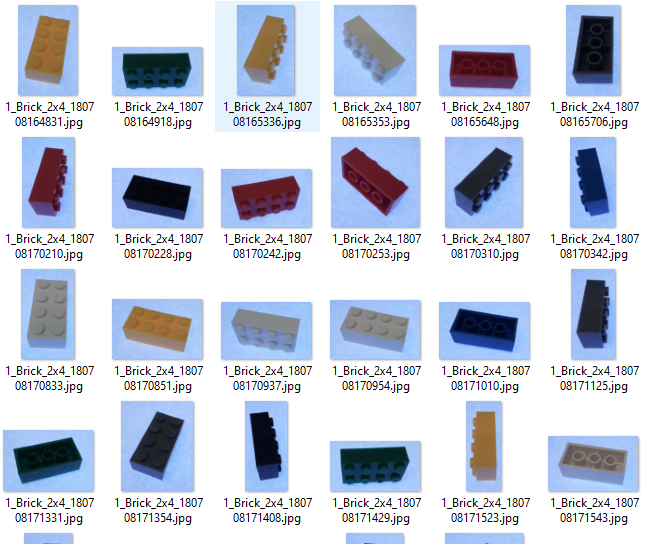
Een ander punt waar eisen voor zijn gesteld is gebruiksvriendelijkheid. Ondanks dat het zeker belangrijks is dat een app gebruiksvriendelijk is, heeft de opdrachtgever aangegeven dat het geen prioriteit is. Hij vindt het interessanter als we goede resultaten kunnen krijgen met ons model.

# Hoofdstuk 2: Realiseren classification

Na het opstellen van de requirements is de dataset gemaakt. In het proces zijn er drie datasets geweest omdat de eerste twee niet de gewenste resultaten gaven. We hebben proefondervindelijk gevonden wat wel en niet werkt voor een dataset.

## Dataset 1: Foto’s van internet

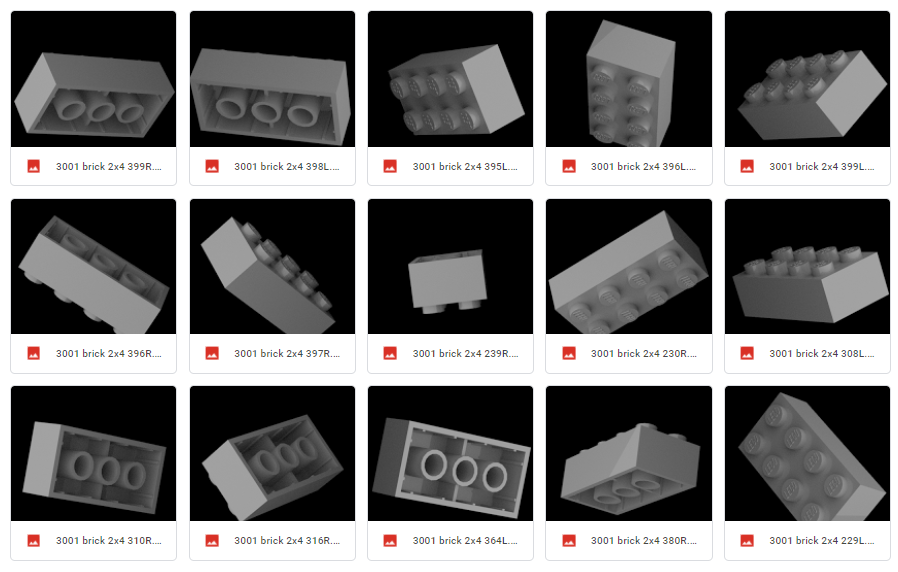
De eerste dataset bestond uit foto’s van legoblokjes die we op internet hebben gevonden. De foto’s waren alleen gesorteerd op vorm, niet op kleur. In overleg met de opdrachtgever is er besloten dat het geen probleem is als er alleen onderscheid gemaakt wordt tussen vorm en niet tussen kleur. Als er verder gegaan was met de dataset was er de optie om later unsupervised learning te gebruiken voor het sorteren van de blokjes.



De resultaten met de dataset zijn helaas niet zoals gewenst. Het probleem kan eraan liggen dat de kwaliteit laag is, de achtergrond blauw is of dat de verhoudingen van de afbeeldingen verschillen. Om het model te trainen worden alle afbeeldingen vervormd naar een vaste resolutie.

## Dataset 2: 3D modellen van het internet.

Nadat de eerste dataset geen succes was is er doorgezocht naar een tweede. Deze bestaat uit afbeeldingen van grijze 3D modellen van verschillende soorten LEGO blokjes. Ze zijn duidelijk en veelbelovend.



Helaas geeft deze dataset nog slechtere resultaten. Elementen zoals belichting zijn bij echte legoblokjes heel anders. Het is mogelijk om synthetische data te laten werken, maar dan moet sowieso een gedeelte niet synthetisch zijn.[[3]](#endnote-4)

## Dataset 3: Onze zelfgemaakte dataset

Het is duidelijk geworden dat foto’s van 3D modellen niet bruikbaar zijn omdat deze niet genoeg overeenkomen met echte foto’s. Verder gaan de foto’s van internet niet werken omdat die te verschillend zijn met de omgeving waarin het model gebruikt gaat worden. Omdat er geen geschikte dataset te vinden was voor de gestelde eisen is er besloten om zelf een dataset te maken.

Voor het maken van de foto’s zijn er in totaal negen verschillende legoblokjes gekozen. Hierbij is ervoor gezorgd dat de soorten het interessantst zijn voor het testen. Er zitten meer unieke blokjes tussen, zoals een bloem en een minifigure die het model zonder problemen moet kunnen herkennen, als een soort van nulmeting. Om het model beter te testen hebben we er ook voor gezorgd dat er vergelijkbare blokjes tussen zitten, zoals een groen en een blauw 2x2 blokje. Van de rode blokjes hebben we 1x2 en 2x4 blokjes. Die hebben dezelfde verhoudingen wat voor extra complexiteit zorgt.

Voor het maken van de foto’s zijn de blokjes op verschillende achtergronden zoals een tafel en een schrift neergelegd. Er is voor gezorgd dat de blokjes van alle kanten zijn gefotografeerd zodat het model ze ook van alle kanten kan herkennen. De foto’s zijn gemaakt met mobiele telefoons. De camera komt overeen met de camera die de eindgebruiker gaat gebruiken omdat het model in een app verwerkt gaat worden. Om snel veel foto’s te maken is gebruik gemaakt van continu opnames. De mobiel maakt binnen een paar seconden tot honderd foto’s snel achter elkaar. Door de mobiel langzaam rond het blokje te bewegen zijn er zo vrij snel veel foto’s met andere hoeken genomen. Een nadeel van deze methode is dat er veel foto’s zijn die veel op elkaar lijken. Verder zitten er wazige foto’s tussen omdat de mobiel bewoog tijdens het fotograferen.



Deze dataset gaf wel veelbelovende resultaten. Zowel bij de snelle testen in de notebook zelf als in de app werkt het model correct. Daarom zijn we verdergegaan met het testen van het model met deze dataset aan de hand van de voorheen opgestelde requirements.

# Hoofdstuk 3: Testen classification.

Nadat een geschikte dataset was gemaakt is er aan de hand van de requirements gekeken hoe goed het model uiteindelijk is voor onze situatie.

## Requirements en gemeten waardes

|  |  |
| --- | --- |
| **Requirement** | Gemeten waarde |
| Kan te minste 30 fps aan | Rond de 30 fps |
| Binnen 33 ms feedback geven | De interence time op een samsung S8 zit rond de app is 100-150 ms, wat te hoog is. Deze meting is op CPU. Op de GPU is de inference time tussen 30-50 ms. |
| Wanneer het blokje goed in beeld is kan het systeem het binnen 1 seconde vinden | Nog niet gemeten |
| Geeft maximaal 10% false positives | 0% (zie hoofdstuk Meten false positives en negatives) |
| Mist maximaal 5% van de te zoeken blokken  (false positive) | 10% (zie hoofdstuk Meten false positives en negatives) |
| Een nieuwe gebruiker moet gemiddeld binnen 30 seconden een blokje kunnen filteren | Om dit aspect te meten hebben we een Human computer interaction onderzoek nodig. |
| 4 op de 5 gebruikers vinden de app er aantrekkelijk uit zien | Om dit aspect te meten hebben we een Human computer interaction onderzoek nodig. |
| 4 op de 5 gebruikers vinden het fijn om met de app te werken | Om dit aspect te meten hebben we een Human computer interaction onderzoek nodig. |
| De data worden lokaal verwerkt | Ja, we maken geen gebruik van een server |
| We vragen toestemming om de camera te openen | Ja, er wordt toestemming gevraagd voor de camera |
| We slaan niet meer gegevens op dan nodig | Ja, we hebben geen database dus we slaan niets op (want het is niet nodig) |
| Geen dingen met de naam LEGO® publiceren zonder toestemming van LEGO® zelf | Ja, want we hebben niets gepubliceerd |
| 95% van de gebruikers kan zien wanneer een blokje wordt aangegeven | Om dit aspect te meten hebben we een Human computer interaction onderzoek nodig. |
| De app maakt een geluidje wanneer hij een potentiële match heeft gevonden | Nee, we hebben nog geen geluid toegevoegd aan onze app |
| Er wordt weergegeven hoe groot de kans is dat de potentiële match correct is | Ja, er is een percentage met de zekerheid die weergegeven wordt op het scherm |
| Er wordt binnen 0,1 seconde na het vinden een vierkant getekend op de plek waar het blokje ligt. | Nee, we hebben classification, geen object detection, dus er wordt geen vierkant getekend. |

Een requirement die we duidelijk nog niet gehaald hebben is het tekenen van een vierkant. We zijn begonnen met het trainen van een classification model. Dat heeft als voordeel dat het simpeler is en de training afbeeldingen niet een voor een gelabeld hoeven te worden. Het nadeel is dat het niet geschikt is voor het vinden van een blokje.

Voor de inference time hebben we gemerkt dat het afhankelijk is van het apparaat. Een Samsung Galaxy s8 zat rond de 100-150 ms op CPU en heeft rond de 30 ms gehaald op GPU. Dan wil de GPU om nog onbekende redenen niet altijd werken. Op andere apparaten hebben we weer andere resultaten, waardoor het moeilijk is om een duidelijke conclusie te trekken. Verder is het zo dat de waardes onaangenaam snel voorbij flitsten als de inference time rond de 30 ms zit. Daarom is het verstandig om de requirements nog een keer te overwegen.

De waarde van de verversingssnelheid is helaas niet betrouwbaar. Het is nog niet gelukt om een manier te vinden om in de app de verversingsnelheid te meten. We hebben wel een app gevonden die de verversingssnelheid van een mobiel kan meten, maar het is onduidelijk hoe betrouwbaar die resultaten zijn. Verder is het niet gedaan met behulp van een accurate meting maar door te kijken naar de nummers en daar ongeveer het gemiddelde van te pakken.

Helaas is het nog niet gelukt om te meten hoelang het duurt totdat een blokje gevonden is.

We hebben onderzocht wat onze false positives en negatives zijn en daarvan zijn de resultaten positief, maar helaas niet betrouwbaar doordat er sprake is van een menselijke factor.

Er is een keuze is gemaakt om de requirements die met desirability te maken hebben niet te testen omdat daar niet de prioriteit op licht, zie hoofdstuk Prioriteiten. Omdat de app zonder server en database werkt voldoet het aan alle viablity aspecten.

## Meten false positives en negatives

Zoals al eerder vermeld bij de prioriteiten wordt de focus gelegd op de accuraatheid. Daarvoor gaat er gebruik gemaakt worden van een confusion matrix. Een confusion matrix bestaat uit een overzicht van de true en false positives en negatives.

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Actual class | |
| Positive | Negative |
| Predicted class | Positive | True positive | False negative |
| Negative | False positive | True negative |

Voor object detection betekent dit:  
True positive: object is aanwezig en gedetecteerd  
False positive: object is niet aanwezig en gedetecteerd  
True negative: object is niet aanwezig en niet gedetecteerd  
False negative: object is aanwezig en niet gedetecteerd[[4]](#endnote-5)   
Hier is het dus belangrijk om te zorgen zo veel mogelijk ”true” results te krijgen terwijl de ”false" results minimaal blijven

### Methode

De metingen hebben plaatsgevonden door middel van het selecteren van een blokje in de app. Als het model het blokje denkt te zien, wordt dat in het beeld weergegeven. Voor elk blokje is het te zoeken blokje van twee hoeken weergegeven en zijn er drie andere hoeken van niet te zoeken blokjes weergegeven. Aan de hand van uitkomst is er genoteerd of de uitkomst correct is en of er dus sprake is van een true of false en een positive of negative.

### Resultaten

|  |  |  |  |
| --- | --- | --- | --- |
| Gegeven blokje | Gevraagde blokje | Output |  |
| Minifig staand | Minifig | Correct | True positive |
| Minifig liggend op buik | Minifig | Correct | True positive |
| Rood 1x2 bovenkant | Minifig | Correct | True negative |
| Blauw 1x1 onderkant | Minifig | Correct | True negative |
| Groen 1x4 lange zijkant | Minifig | Correct | True negative |
|  |  |  |  |
| Rood 1x2 onderkant | Rood 1x2 | Correct | True positive |
| Rood 1x2 korte zijkant | Rood 1x2 | Incorrect | False negative |
| Blauw 2x2 bovenkant | Rood 1x2 | Correct | True negative |
| Groen 2x2 zijkant | Rood 1x2 | Correct | True negative |
| Minifigure liggend op rug | Rood 1x2 | Correct | True negative |
|  |  |  |  |
| Rood 2x4 onderkant | Rood 2x4 | Correct | True positive |
| Rood 2x4 bovenkant | Rood 2x4 | Correct | True positive |
| Blauw 1x1 zijkant | Rood 2x4 | Correct | True negative |
| Groen 2x2 onderkant | Rood 2x4 | Correct | True negative |
| Rood 1x2 bovenkant | Rood 2x4 | Correct | True negative |
|  |  |  |  |
| Blauw 1x1 zijkant | Blauw 1x1 | Correct | True positive |
| Blauw 1x1 bovenkant | Blauw 1x1 | Correct | True positive |
| Minifig staand | Blauw 1x1 | Correct | True negative |
| Blauw 2x2 onderkant | Blauw 1x1 | Correct | True negative |
| Groen 1x4 bovenkant | Blauw 1x1 | Correct | True negative |
|  |  |  |  |
| Blauw 2x2 bovenkant | Blauw 2x2 | Correct | True positive |
| Blauw 2x2 onderkant | Blauw 2x2 | Correct | True positive |
| Blauw 2x4 zijkant | Blauw 2x2 | Correct | True negative |
| Groen 2x2 bovenkant | Blauw 2x2 | Correct | True negative |
| Bloem zijkant | Blauw 2x2 | Correct | True negative |
|  |  |  |  |
| Bloem onderkant | Bloem | Correct | True positive |
| Bloem bovenkant | Bloem | Correct | True positive |
| Minifig | Bloem | Correct | True negative |
| Rood 2x4 bovenkant | Bloem | Correct | True negative |
| Groen 2x2 onderkant | Bloem | Correct | True negative |

Omdat de betrouwbaarheid van de onderzoeksmethode niet hoog is, is het besluit gemaakt om het af te breken en niet verder te meten.

### Conclusie

Er zijn geen true positive aanwezig bij de resultaten. Daarom hebben we maar 0% true positive gemeten.

Eén van de verwachte positives is als false negative uitgekomen. Omdat we tien metingen hebben gedaan die positive zouden moeten zijn hebben we een false negative rate van 1/10 \* 100% = 10 %

### Discussie

Deze testmethode is niet betrouwbaar doordat er te veel invloed is van de menselijke factor. De te controleren blokjes horen willekeurig gekozen te worden, maar een mens zal altijd erbij stil staan welke blokjes er gekozen worden, helemaal als het al bekend is met het model. Dit kan zowel gedaan worden om het model het makkelijker te maken, maar ook om het model op de proef te stellen. Verder werd er gewerkt met een video. Het nadeel daarvan is dat het constant nieuwe beelden geeft. De resultaten veranderen daardoor en staan niet vast zoals bij een foto. Tot slot werd de mobiel in de hand gehouden. Als het model niet de gewenste resultaten geeft kan onderzoeker de mobiel een klein beetje bewegen tot het model dat wel doet.

Om het betrouwbaarder te testen kunnen foto’s gemaakt worden. Die worden dan gebruikt om het model te testen. De computer verdeeld de foto’s willekeurig over de te testen blokjes. Aan de hand van de resultaten van het model kunnen er veel betrouwbaardere conclusies getrokken worden omdat het menselijke aspect een veel kleinere rol speelt.

# Hoofdstuk 4: Overgang naar object detection

Op dit moment is er een getraind model die lijkt te voldoen aan het grootste deel van onze requirements en dus zeer positieve resultaten geeft. Er is echter een probleem, namelijk dat ons model alleen gebruikt kan worden voor image classification en niet object detectie. Voor object detectie is behalve een ander model ook een andere dataset nodig.

Er moet bij elke foto aangegeven worden welke blokjes er te zien zijn en op welk veld van pixels ze zitten. Dit noem je bounding boxes. Omdat bij elke foto bounding boxes aangegeven moeten worden is er besloten om de schaal van de dataset wat kleiner te maken. Een voordeel is wel dat een object detectie model niet vereist dat de foto’s per type in aparte folders verdeeld zijn. De blokjes hun type die in de foto’s voorkomen worden immers in xml files samen met de bounding boxes vermeldt.



Doordat de overgang gemaakt is van image classification naar object detection is gemaakt is voldoet het nu wel aan de requirement dat er een vierkant wordt getekend. Helaas is het nog niet gelukt om het model werkend te krijgen in een app, met als gevolg dat het model niet meer aan andere requirements voldoet. Eerder al is er besloten dat de focus op de accuracy gelegd gaat worden. In overleg met de opdrachtgever is besloten om die focus niet te veranderen. De andere requirement zullen achterwegen gelaten worden.

# Hoofdstuk 5: Accuraatheid

Zoals al eerder vermeld bij prioriteiten (hoofdstuk 1) is de keuze gemaakt om te focussen op de accuraatheid. Na het veranderen van het model naar object detection is er besloten om zie focus nog steeds te behouden (zie eind hoofdstuk 4). Om dat te bereiken gaat er gekeken worden naar hoe goed het model en de dataset bij elkaar passen. Dit is om te voorkomen dat er sprake is van under- of overfitting. Een ander punt is ervoor zorgen de foto’s goed overeenkomen met de achtergrond van de omgeving waar het model in gebruikt gaat worden, dus met als achtergrond achtergronden zoals tafels, maar ook andere legoblokjes.

## Overfitting en Underfitting

Bij overfitting is sprake van een model dat zich te veel aanpast aan de data. Het model ziet vergelijkbare afbeeldingen en focust daarop, maar herkend daarna niet een ander foto van hetzelfde object. Een teken van overfitting is als het model goede resultaten geeft op de training set maar niet op de test set.

Bij underfitting is er sprake van data dat te complex is voor het model. Het model zal dan niet goed kunnen aanpassen aan de data. Het model zal slechte resultaten geven op zowel de test als de trainingset. [[5]](#endnote-6)



## Accuraatheid meten

De accuracy van een model geeft aan hoeveel van de metingen van een model correct heeft geraden. Daarbij is het een getal tussen 0 en 1, waarbij 1 een honderdprocent succesrate is en 0 een nul procent succesrate. De accuracy wordt berekend door middel van een het delen van de hoeveelheid van de correct voorspellingen door de totale hoeveelheid voorspellingen.[[6]](#endnote-7)

Behalve alleen de accuracy zelf willen we zelf graag weten hoeveel false positives en negatives we hebben(zie hoofdstuk 3: meten false positives en negatives) . Voor het meten van de accuraatheid hebben we besloten om een set aan foto’s te maken die overeenkomt met scenario’s die het model in gebruikt kan tegenkomen. De foto’s hebben verschillende achtergronden, van een lege tafel tot in een legobak.

Om het voor te zorgen dat er geen onduidelijkheid is over wat wel en niet geld als correct hebben we de volgende definities opgesteld:

**False positive:** Er is een vierkant getekend zonder dat het desbetreffend blokje in het vierkant licht.

**True positive:** Er is een (of meerdere) vierkanten gerekend waarin het desbetreffende blokje grotendeels in het vierkant licht. Ondanks dat het niet wenselijk is gaan ligt de focus niet op dubbele of half gevonden blokjes. Dit is omdat het belangrijk is dat het blokje gevonden wordt, het is niet relevant of het blokje maar half gevonden wordt of dubbel, het belangrijkste de locatie van het blokje duidelijk wordt. De gebruiker zal ook geholpen worden als het niet helemaal correct wordt aangegeven.

**True negative:** Een object die niet in de dataset zit wordt niet getekend. Er is een geel blokje toegevoegd aan de testfoto’s die niet in de dataset zit. Verder liggen er ook andere voorwerpen die op een bureau te vinden zijn in de foto’s, een computermuis, een pen en een calculator.

**False negative**: Er wordt geen vierkant getekend om een blokje die in de dataset zit.

Na het testen gaan we de testresultaten analyseren en gaan we bedenken hoe het komt dat de resultaten op een bepaalde manier zo zijn. Dan gaan we de dataset aanpassen en nog een keer testen.

Om te voorkomen dat er false positives komen waar er hoogstwaarschijnlijk niets staat hebben is er een treshhold. Het is een waarde en voorspellingen die onder die waarde zitten worden niet weergegeven op het scherm. Als de treshhold laag is de kans om false positives hoog maar als hij te hoog is de kans om true negatives hoog. Tijdens het meten is er gewerkt met een treshold van 0.3.

### Test 1:

Tijdens het bekijken van de resultaten van het model zijn we erachter gekomen dan de twee blokjes waar we mee begonnen zijn als enige getekend werden. Nadat de eerste twee blokjes verwijderd werden hebben is het model nog een keer getraind hebben zijn we tot de conclusie gekomen dat het model de blokjes wel herkent, maar de accuracy heel laag is. Met een treshold van 0.3 worden er nog maar een paar blokjes getekend.

|  |  |
| --- | --- |
| Foto 1 | Foto 2 |
| Foto 3 | Foto 4 |
| Foto 5 |  |
|  |  |
|  |  |
|  |  |

Opvallend aan de resultaten is dat blokjes die wat verder weg staan niet herkend worden. Daarom willen we ook meer blokjes die op een afstand liggen toevoegen aan de dataset. De dataset heeft niet veel foto’s, daarom willen we het uitbreiden. Ons doel is om van elke class honderd blokjes toe te voegen. Tot slot zitten er redelijk wat foto’s tussen die wat wazig zijn. Die kunnen mogelijk ook de resultaten beïnvloeden. Dat gaat ook aangepast worden.

|  |  |
| --- | --- |
| True positives | 1 |
| False positives | 12 |
| True negatives | 33 |
| False negatives | 46 |

Aan de hand van deze testfoto’s is er een accuracy van 36%.

### Test 2:

Na het verbeteren van de dataset is het model opnieuw getraind en zijn dezelfde afbeeldingen getest. Daaruit zijn veel betere resultaten gekomen. Het model had nu veel zekerdere resultaten en was ook vaker correct. Er zijn helaas nog een hoeveelheid false positives, waaronder het gele blokje die niet in de dataset zit en voorwerpen die voor het testen in de afbeeldingen zijn geplaatst.

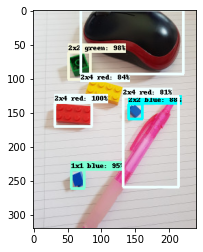
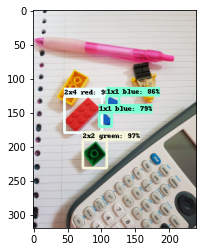
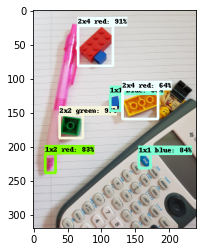
|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

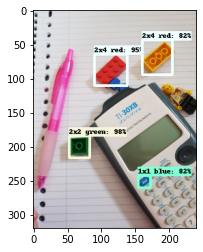
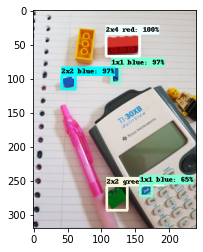
|  |  |
| --- | --- |
| True positive | 46 |
| False positive | 29 |
| True negative | 8 |
| False negative | 0 |

Aan de hand van deze testfoto’s is er een accuracy van 65%

Het model is al een stuk verbeterd. Vooral de aantal false negatives is drastisch afgenomen. Verder is het aantal true positives toegenomen. Het enige waar het model op kan verbeteren is de hoeveelheid false positives en true negatives. Het model is nu te gevoelig en pakt andere blokjes en voorwerpen. Mogelijk ligt het aan de achtergrond. De achtergrond van de testfoto’s zit in ook in veel foto’s van de dataset. Alle foto’s waar meerdere blokjes opstaan hebben de achtergrond. Mogelijk is het model overfitted op die achtergrond.

Om het te testen hebben is het model ook toegepast op andere foto’s die een andere achtergrond hebben:



Na het testen met deze foto’s is de aantal false positives percentueel een stuk lager

|  |  |
| --- | --- |
| True positive | 21 |
| False positive | 12 |
| True negative | 17 |
| False negative | 3 |

Aan de hand van deze testfoto’s is er een accuracy van 71%

Aan de hand van de andere testfoto’s kan geconcludeerd worden dat de achtergrond belangrijk is voor het accuracy van het model. Ondanks dat er in de andere foto’s twee keer een 1x1 blauw zit, een blokje waar het model nog wat moeite mee heeft is de accuracy hoger geworden. Het is niet te bedoeling dat het de testfoto’s aangepast worden op het model om de accuracy hoger te krijgen. Daarom dient de dataset aangepast te worden met meer diversiteit van achtergrond voor een hogere accuracy.

# Conclusie

De dataset is hetgeen waar de meeste aanpassingen aan zijn gedaan om voortgang te maken. Door bijvoorbeeld te zorgen dat er van elk type blokje een even grote hoeveelheid aan foto's in de dataset zit en te sleutelen aan de verhouding tussen train en test foto's is de accuracy steeds hoger geworden. De laatste tests kwamen zelfs uit op een accuracy van 71%. Ondanks alles zijn we hier heel tevreden mee.

De conclusie is dus dat het een redelijk goed ontwerp is. Het model is niet optimaal, niet alle requirements worden tegelijkertijd gehaald. De hoofdoorzaak daarvan is niet de dataset, maar het model zelf. Doordat het object detection model niet wil converteren naar een tflite model is het niet gelukt om de video specifieke requirements te halen.

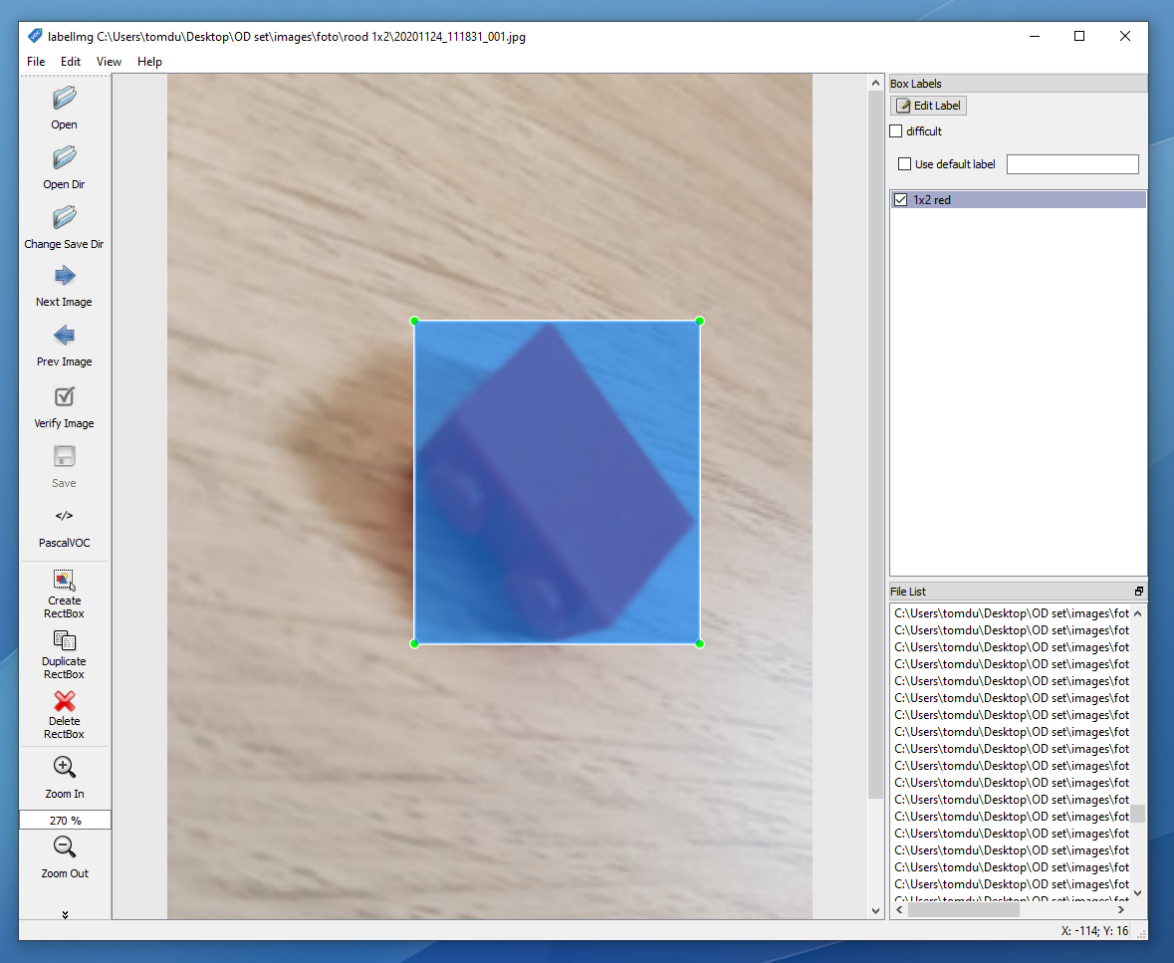
Nog een verbeterpunt zou kunnen zijn dat er voor bepaalde achtergronden in onze dataset sprake is van overfitting. Dit is in de resultaten te zien omdat foto's met bepaalde achtergronden accuratere resultaten behalen dan andere foto's. Het is duidelijk geworden dat het meeste werk bij het trainen van een object detectie model zit in het opbouwen van een bruikbare dataset. Ondanks dat het project voor dit semester klaar is, is de kennis die hierover opgedaan is heel relevant voor een eventuele volgende LEGO Challenge.

# Hoofdstuk 6: Toekomst

Om de dataset te verbeteren zou er meer diversiteit moeten zijn in de achtergrond. Bij de laatste versie van de dataset zijn er te veel foto’s gemaakt met dezelfde achtergrond. Daardoor zijn er veel false positives met foto’s waar dezelfde achtergrond wordt gebruikt. Er is sprake van overfitting.

Verder kan het aantal foto’s per blokje omhoog. Toen er nog gewerkt werkt met een classification model waren er veel foto’s in de dataset. Dat had als resultaat dat het accuraat was. Daarom zal het model accurater worden als er meer foto’s toegevoegd worden.

Na het behalen van acceptabele resultaten met de object detectie dataset is het nu de vraag wat de beste vervolg stap zou zijn. Aan het eind van de rit wordt als het mogelijk is een relatief uitgebreide dataset opgeleverd omdat daar beter mee te testen is op de accuraatheid van het model. Ook is een grotere dataset natuurlijk handiger voor de groep die verder gaat met het project. De dataset die gebruikt werd voor het image recognition model was al vrij uitgebreid en daarom worden de foto's uit die dataset hergebruikt om de object detectie dataset uit te breiden. Dit is echter niet een kwestie van simpelweg overzetten. Bij een object detectie dataset moet namelijk bij elke foto worden aangegeven welke blokjes erop te zien zijn en op welk pixel veld ze staan. Dit wordt gedaan via XML-bestanden die gegenereerd worden via een programma, genaamd labelImg.



Foto's labelen via dit programma kost heel veel tijd omdat het allemaal handmatig moet gebeuren. Om deze reden hebben is er overwogen of het mogelijk was om het proces te stroomlijnen. Een suggestie was om bijvoorbeeld klasgenoten erbij te betrekken. Dit zou echter betekenen dat er een manier gevonden zou moeten worden om de oude dataset per student te verdelen en hier instructies bij te leveren over hoe het labelen werkt. Dan zou er nog een ruimte nodig zijn waar alle studenten hun gelabelde data kunnen uploaden. Wegens een gebrek aan tijd is er besloten om dit niet te doen.

De groep gaat dus zelf nog tijd besteden aan het labelen van foto’s zodat de dataset wat uitgebreider wordt en er een beter beeld kan komen van de accuraatheid van het model als er meer blokjes in de dataset zitten.

Literatuurlijst

1. van den Hoven, J. (2017). *Ethics for the Digital Age: Where Are the Moral Specs?* Geraadpleegd van https://www.delftdesignforvalues.nl/wp-content/uploads/2019/03/Van-den-Hoven\_Ethics-For-The-Digital-Age-Where-Are-the-Specs.pdf [↑](#endnote-ref-2)
2. Orton, K. (2019, 27 maart). Desirability, Feasibility, Viability: The Sweet Spot for Innovation. Geraadpleegd op 18 december 2020, van https://medium.com/innovation-sweet-spot/desirability-feasibility-viability-the-sweet-spot-for-innovation-d7946de2183c [↑](#endnote-ref-3)
3. Daniel West. (2019, 3 december). LEGO Sorter AI: How Does It Work? [Videobestand]. Geraadpleegd van https://www.youtube.com/watch?v=-UGl0ZOCgwQ [↑](#endnote-ref-4)
4. Wikipedia contributors. (2020, 27 september). *Confusion matrix*. Geraadpleegd op 18 december 2020, van https://en.wikipedia.org/wiki/Confusion\_matrix [↑](#endnote-ref-5)
5. Tripathi, M. (2020, 13 juni). Underfitting and Overfitting in Machine Learning. Geraadpleegd op 11 januari 2021, van https://datascience.foundation/sciencewhitepaper/underfitting-and-overfitting-in-machine-learning [↑](#endnote-ref-6)
6. Classification: Accuracy | Machine Learning Crash Course. (2020, 10 februari). Geraadpleegd op 14 januari 2021, van https://developers.google.com/machine-learning/crash-course/classification/accuracy [↑](#endnote-ref-7)