Professionele Bachelor Toegepaste Informatica

Gedragsherkenning dankzij het internet of things

Dries Verbeek

Promotoren:

Christophe Vanhaeren — Ida Mediafoundry
Tom Schuyten — Hogeschool PXL Hasselt

Bachelorpaper Academiejaar 2018-2019

Professionele Bachelor Toegepaste Informatica

Gedragsherkenning dankzij het internet of things

Dries Verbeek

Promotoren:

Christophe Vanhaeren — Ida Mediafoundry
Tom Schuyten — Hogeschool PXL Hasselt

Bachelorpaper Academiejaar 2018-2019

Dankwoord

Ik heb dit eindwerk geschreven tijdens mijn laatste jaar aan de hogeschool PXL. Ik wil graag mijn Hogeschool promotor Tom Schuyten bedanken om mij te begeleiden tijdens deze opdracht. Ook wil ik de personen bedanken die mij geholpen hebben met het bewerken van materialen die nodig waren om dit project te doen slagen. Ik bedank ook mijn stagebedrijf Ida Mediafoundry die mij de kans heeft gegeven om aan zo een tof project te kunnen werken. Als laatste wil ik mijn stagebegeleider Christophe Vanhaeren bedanken die mij begeleid heeft doorheen heel het project.

Abstract

Op publieke plaatsen staan vaak borden die de bezoekers laten weten welk gedrag niet is toegelaten. Die borden hebben echter niet altijd het gewenste effect. De bezoekers negeren die borden vaak en luisteren pas als iemand hen wijst op hun ontoelaatbaar gedrag.

Om die reden wordt er tijdens dit project een systeem ontwikkeld dat het gedrag van een bezoeker kan analyseren, de bezoeker kan waarschuwen en als dat niet helpt, overschakelt naar een drastischer systeem om de bezoeker te laten gehoorzamen.

Om dit project te realiseren, worden er veel verschillende technologieën gebruikt. Om dit systeem te onderhouden wordt er gebruikgemaakt van een backendsysteem in Java. De beelden die ontvangen worden in dit backendsysteem worden geanalyseerd door Google Cloud Vision en AWS Rekognition. Voor het systeem dat de bezoeker filmt, wordt er gebruikgemaakt van OpenCV, om de locatie van de persoon te achterhalen. Voor het systeem dat de pijltjes afschiet op de roker wordt er gebruikgemaakt van een Raspberry Pi, waar *steppermotors* op aangesloten zijn.

Dit systeem maakt dus gebruik van twee verschillende services om het gedrag van de persoon te kunnen achterhalen. In dit eindwerk wordt er onderzocht hoe die twee services het beste gecombineerd kunnen worden, om zo tot een snel en accuraat resultaat te komen. Door deze twee technologieën te combineren kan er een accurater resultaat bekomen worden. Deze techniek heeft als nadeel dat het gebruik van meerdere technologieën de beslissing kan vertragen. Om deze reden wordt er onderzocht hoe deze services kunnen gecombineerd worden zonder de snelheid te veel te verminderen.

Om de accuraatheid van het systeem te testen, moeten we de testen uitvoeren op iemand die aan het roken is. Het is te omslachtig om de testen uit te voeren op iemand die aan het roken is, en daarna hem opnieuw laten roken met een andere service. Daarom worden er op voorhand verschillende foto's genomen van iemand die aan het roken is. Op deze manier kunnen we er ook voor zorgen dat beide services identiek dezelfde foto's doorgestuurd krijgen.

Aan de hand van de resultaten van de testen met de afbeeldingen kunnen we bekijken hoe het systeem moet zijn ingesteld om zo nauwkeurig mogelijk te zijn.

Om de snelheid van het systeem te bepalen gaan we de tijd meten die het systeem nodig heeft om alle stappen uit te voeren die leiden tot het bepaalde resultaat. Deze stappen zijn: het nemen van een afbeelding, het versturen van de afbeelding naar de backend, de afbeelding doorsturen naar Google Cloud Vision en AWS Rekognition, en als laatste het filteren van de resultaten.

Om de kostprijs van het systeem te bepalen, wordt de kostprijs van beide systemen vergeleken en in kaart gebracht. Wanneer alle drie de testen zijn uitgevoerd kunnen we besluiten hoe het systeem het beste kan worden ingesteld om tot een zo snel, accuraat en kostvriendelijk resultaat te komen.

Ook is het belangrijk dat dit project kan uitgebreid en aangepast worden. Daarom zal er een website ontwikkeld worden in Vue.js. Die zal gebruikt worden om het systeem te monitoren en aan te passen. Op deze manier kan een systeem administrator het systeem naar zijn wensen aanpassen en de mogelijke alarmeringssystemen, detectiesystemen en verdrijvingssystemen met elkaar combineren zoals hij dat wil.

Inhoudsopgave

Lijst van gebruikte figuren

Lijst van gebruikte tabellen

Lijst van gebruikte afkortingen

- API: Application Programming Interface
- AWS: Amazon Web Services
- HTTP: Hypertext Transfer Protocol
- HTTPS: Hypertext Transfer Protocol Secure
- IBM: International Bussiness Machines
- OpenCV: Open Computer Vision
- SDK: Software Development Kit

Inleiding

Op publieke plaatsen staan vaak borden die een bezoeker vertellen welk gedrag niet is toegelaten. Die borden hebben echter niet altijd het gewenste effect. Sommige bezoekers luisteren alleen maar als iemand hen hierop wijst. Daarom moet er een systeem ontwikkeld worden dat het gedrag van de bezoekers kan analyseren, hen op de overtreding wijzen en als dit niet helpt hen dwingen om te stoppen met dit gedrag.

Om het gedrag te bepalen van de bezoeker wordt er gebruik gemaakt van een aantal technologieën: Amazon Web Services(AWS) Rekognition en Google Cloud Vision. In dit eindwerk wordt er onderzocht hoe deze twee technologieën kunnen worden gecombineerd om tot een zo snel en accuraat mogelijk resultaat te komen.

Tijdens deze opdracht is een systeem ontworpen om te kunnen detecteren of een bezoeker aan het roken is. Hiervoor wordt er gebruik gemaakt van een Raspberry Pi, met daarop een camera aangesloten. Als alarmeringssysteem worden er pijlen opgesteld die met behulp van Philips hue lampen worden opgelicht. Als dit niet werkt om de bezoeker te doen stoppen met roken dan wordt er met zachte pijltjes op de bezoeker geschoten.

Deze opdracht wordt uitgewerkt bij Ida Mediafoundry te Hasselt. Ida Mediafoundry is een bedrijf dat deel uitmaakt van de Xplore Group. De Xplore Group is eigendom van de Cronos Group. Dit bedrijf specialiseert zich in het maken van digitale, interactieve en gebruiksvriendelijke applicaties.

I. Stageverslag

1 Bedrijfsvoorstelling

Ida Mediafoundry is een bedrijf dat deel uitmaakt van de Xplore Group. De Xplore Group is gesitueerd binnen de Cronos Group. Ida Mediafoundry heeft kantoren in zowel Hasselt, Merelbeke, Eindhoven als Kontich. Deze stageopdracht gaat door in het kantoor in Hasselt.

Dit bedrijf specialiseert zich in het maken van digitale, interactieve en gebruiksvriendelijke applicaties. Ida Mediafoundry heeft al grote projecten op poten gezet zoals het e-commerce platform van A.S. Adventure en de klantenservice van Telenet.

Ida Mediafoundry is in 2005 opgericht als *Adobe Competence Center* binnen de Cronos Group. Sinds dien hebben ze veel bereikt, zoals te zien is op figuur 1.

Figuur 1 geschiedenis van Ida Mediafoundry

In 2006 is IDA N.V. opgericht met als missie dé expert te worden in *Adobe Enterprise Solutions*. Ze werden in 2010 benoemt tot een *Adobe Business Partner*. In 2018 zijn ze uitgeroepen tot *Adobe Experience Cloud Partner Of The Year* in de Benelux.

2 Voorstelling stageopdracht

2.1 Probleemstelling

In het dagdagelijkse leven staan er op veel plaatsen borden die mensen laten weten wat niet is toegelaten. Die borden hebben echter niet altijd het gewenste effect. Mensen negeren deze borden vaak totdat iemand hen hier op wijst. Om toch tot de mensen door te dringen moet de boodschap worden overgebracht op een manier die duidelijk en origineel is. Om die reden moet er een systeem ontwikkeld worden dat het gedrag van een gebruiker kan analyseren, de gebruiker verwittigen en eventueel drastischere maatregelen kan inzetten.

2.2 Doelstellingen

Tijdens deze opdracht moet er een systeem ontwikkeld worden dat ten eerste via een camera de omgeving in het oog kan houden en het gedrag dat de mensen vertonen kan analyseren. Als het systeem merkt dat iemand gedrag vertoont dat niet gewenst is, moet het systeem die persoon hiervan op de hoogte stellen en hem een alternatief aanbieden. Het systeem moet vervolgens kijken of de persoon die de overtreding begaat effectief stopt met wat hij aan het doen was. Wanneer de persoon dit niet doet, moet het systeem overschakelen naar een mechanisme dat de gebruiker op een drastischere manier dwingt om te stoppen met de activiteit die de persoon aan het uitvoeren was.

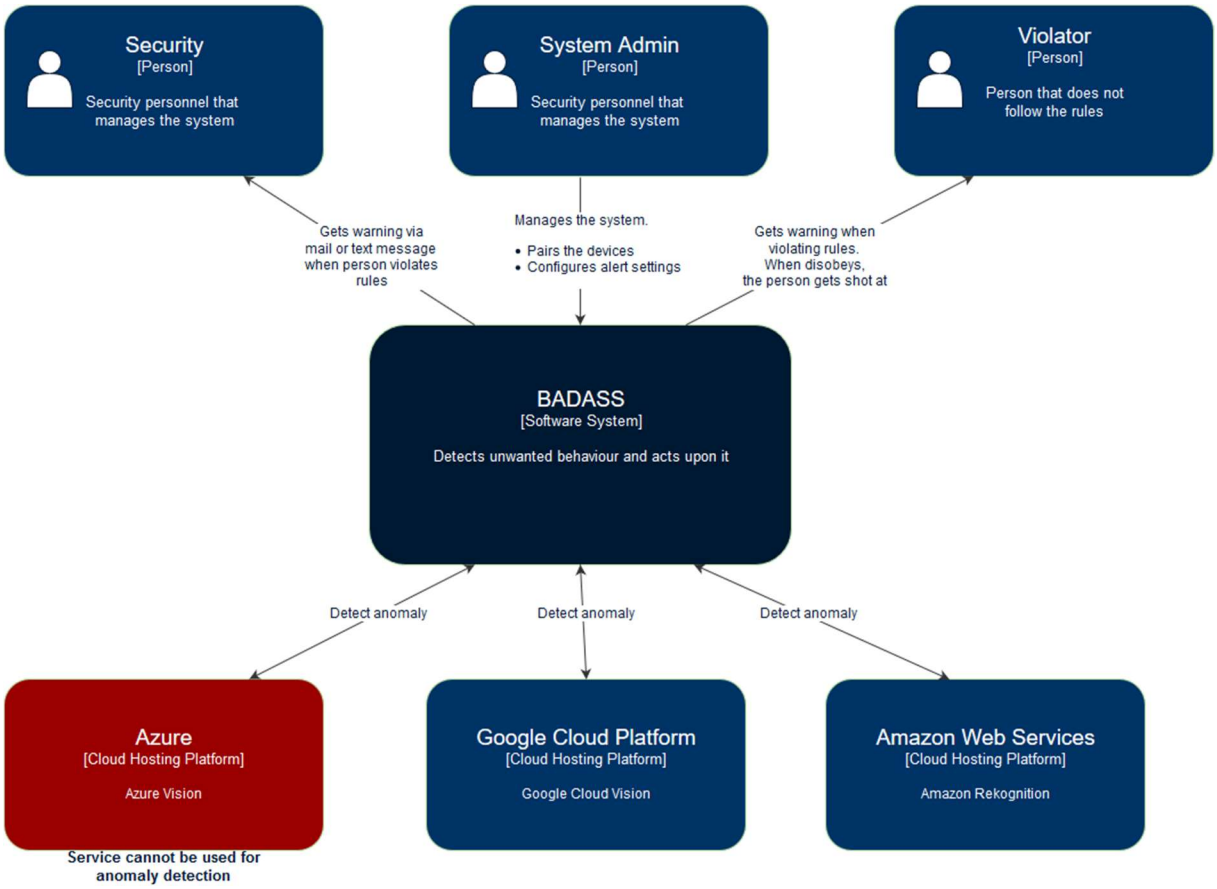
2.3 De omgeving

Om dit systeem te ontwikkelen worden er van verschillende bestaande systemen gebruikgemaakt. Om de toestellen te maken die verantwoordelijk zijn voor het detecteren van het gedrag, het melden aan de persoon en het stoppen van de activiteit wordt er gebruik gemaakt van Raspberry Pi's. Deze machines staan in verbinding met een backend systeem dat geschreven is in Java met behulp van spring boot.

3 Overzicht van het systeem

3.1 De plaats van het systeem in de context

Om de plaats van het systeem duidelijk te maken, is er een C4 diagram opgesteld, zoals te zien is op figuur 2.



Figuur 2 C4 level 0 model van het systeem

Het systeem zal door een systeemadministrator gebruikt worden om het systeem in te stellen. Het systeem biedt ook de mogelijkheid om een bericht te sturen naar iemand, wanneer het systeem iemand detecteert die de regels overtreedt, bv. Het beveiligingspersoneel. Ten slotte wordt het systeem ook nog gebruikt door de persoon die een overtreding begaat. Hij wordt verwittigd via lichtgevende pijlen, en wanneer deze pijlen geen effect hebben, zal hij beschoten worden met zachte pijltjes die naar hem worden afgevuurd.

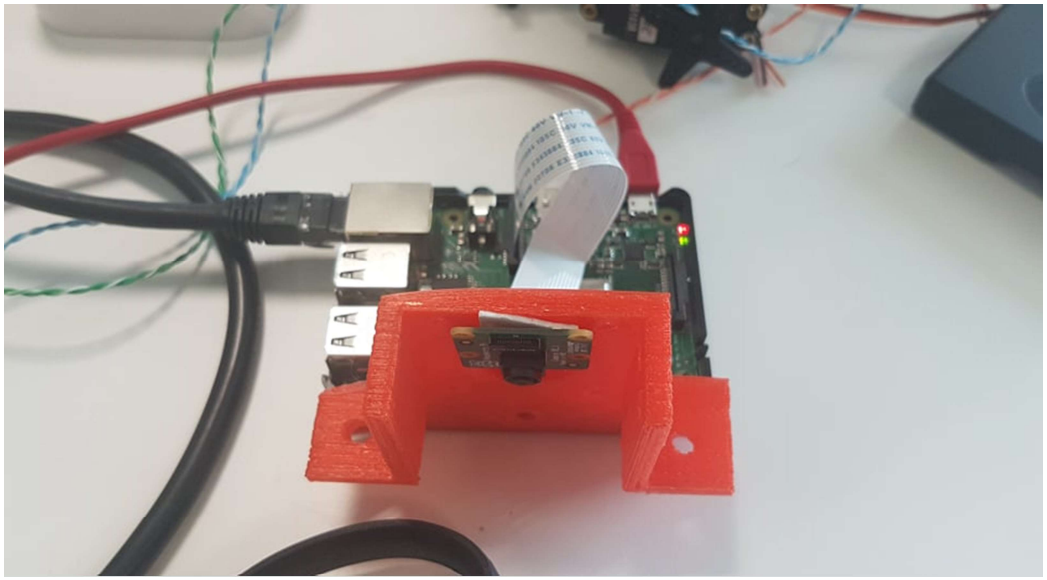
Het systeem gebruikt ook nog verschillende externe services. Deze services worden gebruikt om het gedrag dat de gebruiker vertoont te analyseren en te melden aan het systeem.

3.2 De verschillende onderdelen van het systeem

Het systeem dat wordt ontwikkeld bestaat uit verschillende onderdelen. Ten eerste is er het apparaat dat de beelden van de gebruiker filmt en doorgeeft aan het backendsysteem. Ten tweede is er het apparaat dat de gebruiker verwittigt dat het gedrag dat hij vertoont ongewenst is. Tijdens deze opdracht worden er lichtgevende pijlen gebruikt die de gebruiker doorverwijst naar een rokershokje, als alarmeringssysteem. Er is ook nog het systeem dat de gebruiker op een drastischere manier afdwingt om te stoppen met zijn gedrag. Voor deze opdracht wordt dit een geweer dat met zachte pijltjes schiet. Er is ook een frontendapplicatie aanwezig die de systeemadministrator kan gebruiken om het systeem in de gaten te houden en aan te passen. En ten slotte is er nog de backend die de communicatie tussen de apparaten voorziet en die bepaalt welk gedrag de gebruiker vertoont.

3.2.1 Het detectiesysteem

Het detectiesysteem is opgebouwd uit een Raspberry Pi met daarop een camera aangesloten, zoals te zien is op figuur 3. Wanneer het detectiesysteem opstart, zal ook de applicatie die op het apparaat staat opstarten. Wanneer de applicatie opstart zal deze zich registreren op het backendsysteem, om te laten weten dat dit apparaat online is. Wanneer dit niet lukt zal hij dit blijven proberen tot dit wel lukt.



Figuur 3 het detectiesysteem

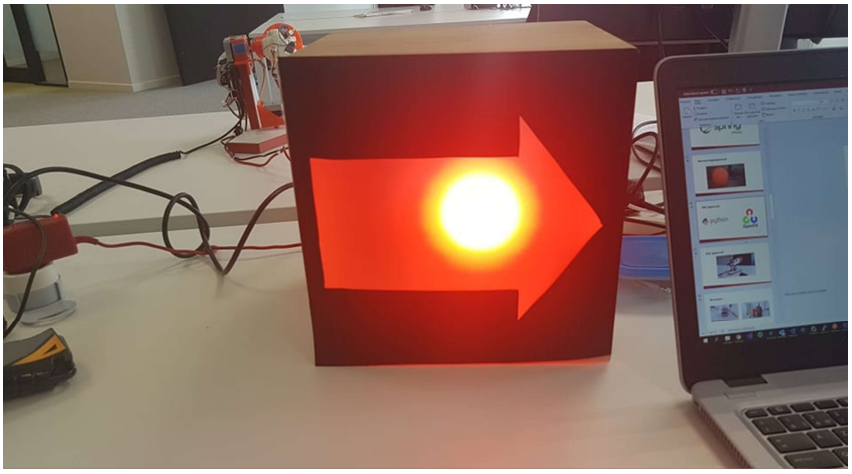
Wanneer het registreren gelukt is, zal er een spring boot applicatie opstarten. Deze is nodig zodat het backendsysteem met dit apparaat zou kunnen communiceren.

Wanneer dit allemaal gebeurd is, is het apparaat klaar om te beginnen met het nemen van foto's. Het systeem krijgt een interval van de backend . Dit interval bepaalt de tijdsspanne tussen de foto's.

Wanneer de backend ziet dat er gedrag wordt vertoond dat niet gewenst is, wordt de tijdsspanne tussen de foto's kleiner. Zo kunnen we een sneller resultaat krijgen van het gedrag van de gebruiker. Wanneer het ongewenste gedrag niet meer wordt waargenomen, zal de tijdspanne tussen de foto's weer groter worden.

3.2.2 Het alarmeringssysteem

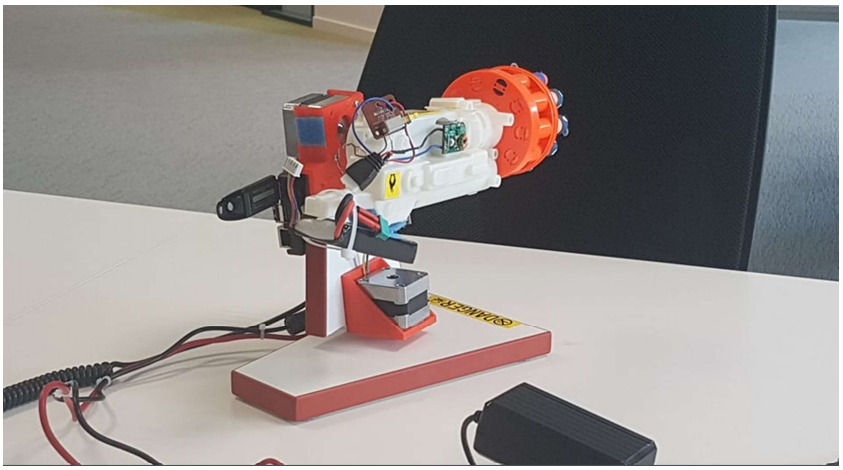
Het alarmeringssysteem bestaat uit een of meerdere Philips Hue lampen. Deze lampen zijn verbonden met een *bridge*. Aan de hand van een bestaande Application Programming Interface(API) zullen deze lampen worden opgelicht wanneer de gebruiker betrapt wordt op gedrag dat niet is toegelaten. Deze lampen zullen in een pijl worden geplaatst zodat het lijkt alsof de pijl licht geeft, zoals te zien is op figuur 4. Die pijlen zullen dan zo opgesteld worden zodat zij de richting naar het rokershokje aangeven.



Figuur 4 de lamp die de weg wijst

3.2.3 Het killsysteem

Het killSysteem bestaat uit een *Raspberry Pi* met daarop twee *steppermotors* aangesloten. Een van deze *steppermotors* zal dienen om de horizontale rotatie van het geweer in te stellen. De tweede *steppermotor* zal dienen om de verticale rotatie van het geweer in te stellen. Wanneer het backendsytseem beslist dat er een overtreding is van de regels zal het backendsystseem een verzoek versturen naar het killsysteem. In dit verzoek zit de laatste afbeelding die de camera heeft gemaakt. Het killsysteem zal de locatie van de persoon uit de afbeelding halen en zijn locatie berekenen. Het killsysteem zal vervolgens het geweer op deze persoon richten, hem een waarschuwingsboodschap afspelen en vervolgens op hem schieten. Wanneer dit voltooid is zal het killsysteem een verzoek sturen naar het backendsysteem om te melden dat de actie is voltooid. De volledige opstelling is te zien op figuur 5.



Figuur 5 de volledige opstelling van het geweer

3.2.4 De frontend

Er is ook een website beschikbaar. Deze website is enkel toegankelijk voor systeemadministrators. Op deze website kunnen ze de verschillende beschikbare apparaten zien. Ze kunnen die dan combineren hoe zij dit willen. Ze kunnen hier ook de verschillende groepen van apparaten zien en hun status. Op deze website kunnen ze dan ook het huidige beeld van de camera's zien. Deze webapplicatie is gemaakt met behulp van Vue.js.

Aan de bovenkant van het scherm is een titelbalk voorzien waarin het bedrijfslogo, de naam van de applicatie en een emailknop staat. Als de gebruiker op deze knop klikt verschijnt er een veld in een pop-up waarin hij een emailadres kan instellen. Wanneer het systeem ongepast gedrag detecteert zal het een mail sturen naar dit email-adres.

Aan de rechterkant van de applicatie staan alle beschikbare apparaten. Hier kan de gebruiker het IP-adres en het type van apparaat zien. Aan de linkerkant van het scherm is ruimte voorzien om alle apparaten die aan elkaar gelinkt zijn te zien. De gebruiker kan apparaten met elkaar verbinden door op de rode knop naast de titel van gelinkte apparaten te klikken. Als de gebruiker op deze knop klikt zal er een pop-up verschijnen waar de gebruiker de verschillende apparaten kan selecteren en deze groep een naam kan geven.

Wanneer een nieuwe groep van apparaten is opgesteld, wordt de status ingesteld op detecteren. De gebruiker kan gemakkelijk zien dat de toestellen zich in deze stand bevinden, want die zijn dan groen gekleurd. Wanneer het systeem overgaat naar het waarschuwen van de gebruiker zal het blokje geel worden en wanneer het systeem overschakelt naar het verdrijven van de gebruiker zal het blokje rood oplichten.

In het midden van het scherm is er nog ruimte voorzien voor de camera. Wanneer de gebruiker op een groep van apparaten klikt, zal er in het midden van het scherm het huidige beeld van de camera te zien zijn.

3.2.5 De backend

Centraal in dit volledige systeem staat de backend. Deze backend is ook gemaakt met behulp van spring boot. De backend is verantwoordelijk voor het ontvangen en versturen van berichten naar de verschillende apparaten. Dit onderdeel van het systeem is ook verantwoordelijk voor het zenden van de afbeeldingen naar AWS Rekognition en Google Cloud Vision, en het bepalen van het gedrag van de gebruiker. Ten slotte is het backendsysteem ook nog verantwoordelijk voor het zenden en ontvangen van data die van de frontendapplicatie komt.

4 De verschillende statussen van de applicatie

Er zijn vier verschillende statussen waarin het systeem zich kan bevinden. Om het simpel te houden voor de gebruiker van de applicatie zullen er slechts drie getoond worden, de vierde is enkel technisch bedoeld binnen in het systeem.

De eerste status is de detectiestaat. Dit wil zeggen dat de applicatie geen verkeerd gedrag heeft gedetecteerd. Het interval tussen de genomen foto's is dan ook vrij groot.

De tweede status wordt niet getoond aan de gebruiker. Het systeem bevindt zich in deze status wanneer het gedrag opmerkt dat zou kunnen worden beschouwd als overschrijdend gedrag, Maar het systeem onvoldoende zeker is van zijn analyse.

De derde status is de alarmeringsstatus. Wanneer het systeem voldoende bewijs heeft van het gedrag van de gebruiker zal hij naar deze fase overgaan. De applicatie zal dan het alarmeringssysteem in werking zetten. Deze status blijft behouden totdat de gebruiker stopt met het niet toegelaten gedrag of totdat het afdwingingssysteem in werking treed.

De vierde en laatste status is de *killstatus*. In deze fase zal het verdrijvingssysteem van de applicatie in werking treden. Deze status verdwijnt pas als de gebruiker stopt met het niet toegelaten gedrag.

5 De gebruikte technologieën

5.1 Vue.js

Het managementsysteem dat gebruikt wordt om dit systeem in te stellen en te monitoren is een website geschreven met behulp van Vue.js. Vue.js is een *progressive framework* om websites te bouwen. Vue.js is een licht *framework* dat zich focust op de *view layer* van de applicatie. Het is een *framework* dat gemakkelijk te integreren is met bestaande projecten. Langs de ander kant is Vue.js ook geschikt voor het maken van websites met maar één pagina. [1]

5.2 Spring boot

Het backend systeem dat de communicatie tussen de verschillende onderdelen van dit systeem zal verzorgen is geschreven in Java, met behulp van Spring Boot.

Spring boot is een Java *framework* dat gebruikt wordt om het bouwen van *micro services* en applicaties die op zichzelf bestaan. Spring boot zorgt voor een makkelijk configuratie van het *spring framework.* [2]

5.3 Flask

Om de motoren die op de Raspberry Pi zijn aangesloten aan te sturen, moet er gebruik gemaakt worden van de bestaande modules van Adafruit, de makers van het dochterbord voor de raspberry Pi, dat de *steppermotors* bestuurd.

Om het killapparaat en het backendsysteem met elkaar te laten communiceren draait op de Raspberry Pi een flask-applicatie. Flask is een *micro web framework* geschreven in Python. [3]

6 De doelstellingen van de applicatie

6.1 Gedragsdetectie

De belangrijkste doelstelling van de applicatie is het detecteren van het gedrag van een gebruiker. Om het gedrag van een gebruiker te analyseren wordt er gebruik gemaakt van Google Cloud Vision en AWS Rekognition. Wanneer het detectieapparaat een afbeelding doorstuurt naar de backend zal deze doorgestuurd worden naar de gedragsdetectieservice van de groep waartoe het detectieapparaat behoort. Deze zal dan via *helperclasses* de afbeelding laten analyseren door Google Cloud Vision en AWS Rekognition. Die geven Waardes terug die overeenkomen met wat ze in de afbeelding hebben gevonden en een percentage dat aangeeft hoe zeker ze zijn van hun bevindingen. Om deze acties uit te voeren bieden zowel Google als Amazon *classes* aan die deze actie verrichten. Wanneer deze acties zijn uitgevoerd worden deze waardes omgezet naar een object dat we zelf aan hebben gemaakt. Op deze manier staan de resultaten van zowel Google Cloud Vision en AWS Rekognition in hetzelfde formaat.

Daarna worden de resultaten gefilterd zodat alleen nog maar resultaten overblijven die met het ingestelde gedrag te maken hebben. Wanneer we de resultaten gefilterd worden moet ook het percentage hoog genoeg zijn, hoe hoog deze waarde is wordt bepaald in het onderzoek. Wanneer er nog resultaten overblijven na het filteren wordt dit beschouwt als genoeg overtuiging dat iemand de regels overtreed.

Wanneer er na het filteren geen resultaten meer overblijven, betekent dit dat er geen fout gedrag is gedetecteerd. Het probleem is echter dat wanneer één van de vele foto's die wordt doorgestuurd een slechte kwaliteit heeft, of er gaat iets anders mis, dat de hele cyclus van de applicatie terug van voor af aan begint. Daarom is er een soort van buffer ingebouwd, er zijn vijf foto's op rij nodig waar geen fout gedrag op wordt gedetecteerd om de cyclus te herstarten.

6.2 Het oplichten van een Philips hue lamp

Wanneer de backend beslist dat er voldoende bewijs is, dat iemand de regels overtreed moeten we de persoon die de regels overtreed verwittigen. Tijdens deze stage is er beslist om dit te doen aan de hand van een Philips hue lamp. Om dit te doen hebben we een paar parameters nodig.

De eerste parameter is het IP-adres van de *bridge.* Deze *bridge* dient als centraal punt voor de lampen.

De tweede parameter die we nodig hebben is een gebruiker. Een gebruiker is noodzakelijk om te weten dat niet zomaar iedereen de lamp kan doen branden. Deze gebruiker is een stuk tekst dat gegenereerd wordt door de *bridge.* Om deze gebruiker aan te maken moeten we een *request* sturen naar de bridge, met daarin de naam van de applicatie en een naam. De *bridge* zal dan een error terug geven dat de link knop niet is ingedrukt. Vervolgens wordt in de frontend gevraagd aan de systeemadministrator om deze knop in te drukken. Wanneer de systeemadministrator dit gedaan heeft, wordt dezelfde *request* opnieuw verstuurd. De bridge geeft nu een nieuwe gebruiker terug.

De derde parameter is het nummer van de lamp. Aangezien dat er meerder lampen op een *bridge* kunnen zijn aangesloten, moeten we specifiëren welke lamp dat we willen doen oplichten.

De vierde parameter is de tijdsspanne dat de lamp blijft branden. Wanneer de backend heeft beslist dat iemand de regels heeft overtreden laten we lamp branden, vanaf dit moment maken we een nieuwe taak aan die na de ingegeven tijdsspanne een nieuwe *request* stuurt naar de *bridge* die de lamp doet uitgaan.

De laatste parameter die nodig is, is de kleur die de lamp moet worden wanneer hij oplicht. De kleur die verstuurd moet worden in de *request* is een getal tussen nul en 65535. Dit is niet gebruiksvriendelijk dus bieden we de systeemadministrator een keuze tussen verschillende basis kleuren.

6.3 Het afvuren van het geweer

Wanneer de backend beslist dat iemand de regels overtreed, zal deze de lamp doen branden voor een bepaalde tijd. Wanneer in de tussentijd de overtreder nog niet gestopt is met zijn gedrag, dan gaan we hem beschieten met een speelgoedgeweer.

De eerste stap die wordt uitgevoerd is het kijken of we de persoon die de regels overtreed kunnen vinden, met behulp van Open Computer Vision(OpenCV). Om dit te doen beginnen we te filmen met de Raspberry Pi camera, we analyseren elke frame om te kijken of we een persoon kunnen vinden.

Wanneer we een persoon vinden dan berekenen we de hoek die het geweer moet draaien om recht op de persoon te schieten. Wanneer de persoon recht in het midden staat van de beelden, beginnen we met schieten.

Om het pistool te laten richten op de persoon die de regels overtreed, maken we gebruik van twee *steppermotors*. Eén van de ze twee *steppermotors* wordt gebruikt om het geweer rond zijn as in een horizontale beweging te laten draaien. De andere s*teppermotor* wordt gebruikt om het pistool te richten met een verticale beweging.

Wanneer het geweer zich gedraaid heeft, wordt er opnieuw gezocht naar een persoon. Op deze manier kan er gekeken worden of de persoon zicht tijdens het richten niet heeft verplaatst.

Wanneer we merken dat de persoon recht in het midden van de camera staat, dan kunnen we op hem schieten. Om het geweer te laten schieten wordt er gebruik gemaakt van een *relay*. Wanneer we een hoog signaal naar de *relay* sturen, zal het circuit verbonden zijn. Hierdoor begint het geweer te schieten. Na een aantal seconden wordt er opnieuw een signaal naar de *relay* gestuurd, maar dit keer een laag signaal, hierdoor stopt het geweer met schieten.

7 Beveiliging

Omdat het systeem met beelden van personen werkt en verschillende apparaten aanspreekt is het belangrijk dat ons systeem ook voldoende veilig is.

7.1 Philips hue lampen

De beveiliging van de Philips hue lampen laat nog veel te wensen over. De verbinding tussen de zender en de lampen gebeurt door middel van een *Hypertext transfer Protocol*(HTTP) connectie, en geen beveiligde *Hypertext Transfer Protocol Secure*(HTTPS) connectie. Dit zorgt ervoor dat iedereen die toegang heeft tot het netwerk met de juiste software alle communicatie tussen de zender en de lampen kan zien.

Bovendien staat de gebruikersnaam dat moet worden meegegeven in de *header* van de *request*. Dit zorgt ervoor dat iemand die toegang heeft tot het netwerk zeer gemakkelijk de gebruikersnaam van de zender kan stelen.

Alle beveiliging die de Philips hue lampen hebben hangt dus totaal af of dat het wachtwoord van het netwerk sterk is of niet. Dit zou een probleem kunnen vormen als dit systeem zou worden ingezet op een plaats waar het netwerk voor iedereen toegankelijk is.

7.2 Het managementsysteem

Het spreekt vanzelf dat niet zomaar iedereen toegang kan hebben tot het managementsysteem, maar enkel systeemadministrators. Daarom moet iemand die het managementsysteem wil gebruiken zich eerst inloggen. Hier is een aparte pagina op de website voor voorzien. Wanneer de persoon die wil inloggen op het managementsysteem de juiste gebruikersnaam en wachtwoord ingeeft zal hij doorgestuurd worden naar de overzichtspagina.

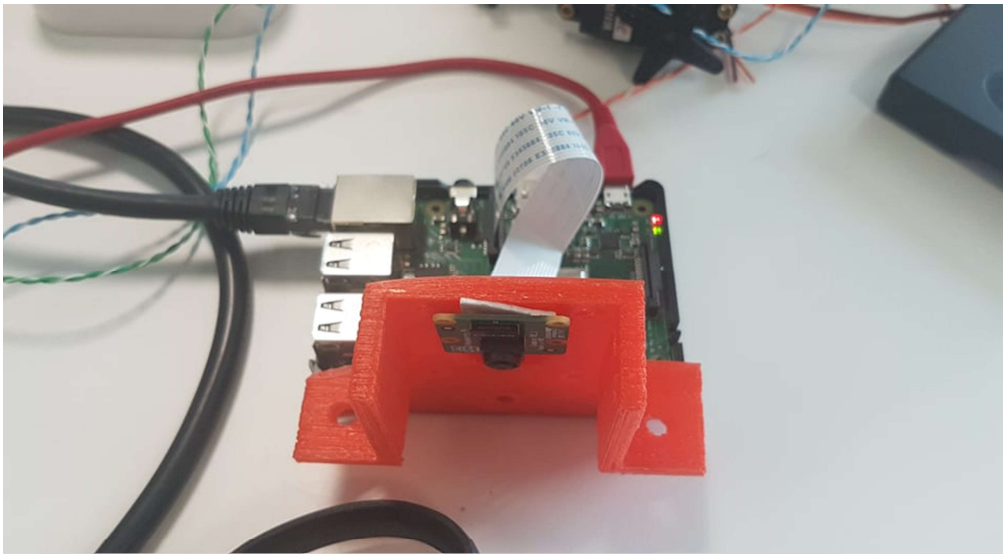
7.3 De backend

Naast de frontend moet ook de backend beveiligd worden. Om dit te verwezenlijken maken we gebruik van spring security. De wijze waarop de backend wordt beveiligd noemt *basic authentication.* Telkens wanneer het managementsysteem of een apparaat gegevens van of naar de backend wil sturen, moeten er een gebruikersnaam en een wachtwoord worden meegegeven, anders zullen zij een error terug krijgen.

8 De opstelling

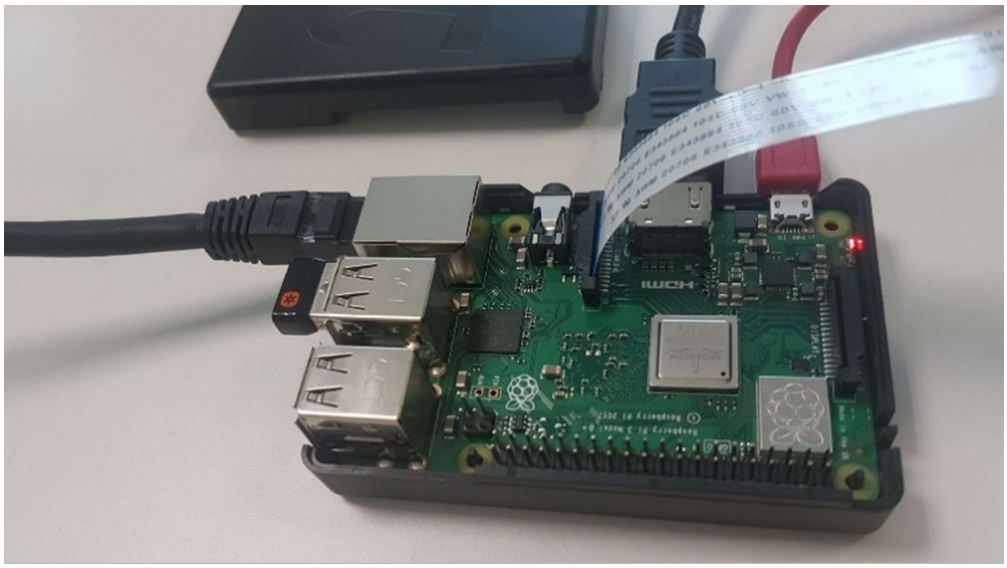
8.1 Het detectiesysteem

Het detectiesysteem is een vrij simpele opstelling. Het systeem bestaat uit een Raspberry Pi met daarop een camera aangesloten, de opstelling is te zien op figuur 6.



Figuur 6 Opstelling met camera

De Raspberry Pi is een kleine computer, ter grootte van een bankkaart. Op deze Raspberry Pi is een camera aangesloten. De kabel van deze camera sluiten we aan op de camera poort van de Raspberry Pi, zoals te zien is op figuur 7.



Figuur 7 aansluiting van de camera

Om de elektronica niet bloot te leggen is er een plastieken opzetstukje gemaakt met een 3D printer, om de camera op zijn plaats te houden.

8.2 Het killsysteem

Net zoals het detectiesysteem is de basis van het killsysteem een Raspberry Pi. Op deze raspberry Pi zijn twee *steppermotors* aangesloten.

Een *steppermotor* heeft vier kabels. De gele en de rode kabel dienen om de eerste spoel in de motor te laten draaien. De groene en grijze kabel dienen om de tweede spoel te doen draaien. Eén van deze twee *steppermotors* wordt gebruikt om de horizontale richting van het geweer in te stellen. De andere *steppermotor* zal de verticale richting van het geweer bepalen.

Op de Raspberry pi is er een dochterbord aangesloten. Dankzij dit dochterbord kunnen de *steppermotors* op een gemakkelijke manier bestuurd worden. Om het dochterbord en de *steppermotors* van voeding te voorzien is een voeding van twaalf volt aan het dochterbord gekoppeld.

Om de pijltjes af te schieten hebben we met twee kabels het schietmechanisme van het geweer omzeilt. Wanneer deze 2 kabels met elkaar in verbinding komen, begint het geweer te schieten.

Om het geweer te doen schieten, wordt er gebruik gemaakt van een *relay*. Wanneer deze *relay* een signaal binnenkrijgt, de zal deze het circuit sluiten en zal het geweer beginnen te schieten. Deze *relay* dient met 3 kabels aangesloten te worden op de Raspberry Pi. De rode kabel is aangesloten op pin twee, deze dient als voeding voor de *relay*. De bruine kabel is aangesloten op pin zes, deze dient als gronding voor de *relay*. De oranje kabel is de kabel waarover het signaal naar de *relay* wordt gestuurd. Deze is aangesloten op pin 21. Met deze opstelling moet er enkel een signaal worden gestuurd over pin 21, om het geweer te doen schieten.

9 Reflectie

Dit project was voor mij een grote uitdaging, aangezien er tijdens dit project met veel technologieën moesten gewerkt worden, waar ik nog geen ervaring mee had. Dit is de reden dat ik voor deze opdracht heb gekozen.

Tijdens de eerste twee weken van de stage heb ik mij bezig gehouden met research. Aangezien dat de doelstelling van mijn opdracht nog niet vast stonden bij aanvang van het project, heb ik tijdens deze twee weken de doelstellingen van de opdracht moeten bepalen. Doordat ik nog maar weinig kennis had over de mogelijkheden met de gekozen technologieën was het moeilijk om te weten, wat haalbaar was en wat niet.

Tijdens dit project heb ik met veel nieuwe technologieën leren werken. Dit waren technologieën zoals: Google Cloud Vision, AWS Rekognition, python, flask, OpenCV en verschillende extensies voor een Raspberry Pi. De periode voor de paasvakantie diende voor het ontwikkelen van het backendsysteem, het managementsysteem, het detectieapparaat en de communicatie met de lampen.

De periode na de paasvakantie, diende om het geweer te ontwikkelen. Tijdens deze periode zijn een aantal problemen komen opsteken. De servo's die we gekocht hadden om het pistool te richten op de roker bleken niet sterk genoeg te zijn. Om dit op te lossen hebben we sterke steppermotors gekocht.

Tot slot vind ik dat ik goed in de opzet van deze stageopdracht ben geslaagd. Met zoveel nieuwe technologieën werken was een zeer grote uitdaging, maar een zeer leerrijke uitdaging.

II. Onderzoekstopic

1 Onderzoeksvraag

Een belangrijk deel van dit project is het herkennen van mensen die roken. Met de camera's die op de Raspberry Pi's zijn aangesloten worden er foto's genomen op bepaalde intervallen. Die beelden worden vervolgens naar het backendsysteem verzonden.

Deze intervallen moeten op voorhand worden vastgelegd. Wanneer deze intervallen kleiner worden, worden er meer beelden verzonden, waardoor er een sneller resultaat kan worden vastgelegd. Dit heeft als nadeel dat de kosten van het systeem zullen stijgen. Daarom wordt er onderzocht welk interval moet worden genomen om een snel resultaat te krijgen dat toch budgetvriendelijk is.

Om de beelden te analyseren bestaan er verschillende systemen. Voorbeelden van deze systemen zijn: Amazon Rekognition en Google Cloud Vision. Door deze twee systemen te combineren kan er een accurater resultaat bekomen worden. Deze techniek heeft als nadeel dat het gebruik van meerdere systemen de beslissing kan vertragen. Om die reden wordt er onderzocht hoe deze services kunnen gecombineerd worden zonder de snelheid te veel te verminderen.

Tijdens dit onderzoek wordt er dus onderzocht hoe we deze criteria het beste kunnen instellen zodat er een snelle, accurate en budgetvriendelijke beslissing kan worden genomen.

2 Methode van onderzoek

Om de accuraatheid van het systeem te testen, moeten er testen uitgevoerd worden op iemand die aan het roken is. Het is te omslachtig om de testen uit te voeren op iemand die aan het roken is, en daarna hem opnieuw laten roken met een andere service. Daarom worden er op voorhand verschillende foto's genomen van iemand die aan het roken is. Op deze manier is het zeker dat beide services identiek dezelfde foto's doorgestuurd krijgen.

Wanneer Google Cloud Vision of AWS Rekognition een afbeelding binnenkrijgt, geven zij beide een collectie terug van resultaten. Deze resultaten staan voor de elementen die de service in de afbeelding heeft gevonden. In de resultaten zijn 2 soorten waardes te vinden. De eerste waarde is een beschrijving wat er in een beschrijving van een element dat de service in de afbeelding heeft terug gevonden. De tweede waarde is een percentage. Dit percentage is de zekerheid waarvan de service is dat het gevonden element wel degelijk in de afbeelding te vinden is.

De afbeeldingen die worden genomen moeten van elkaar verschillen om te de services met elkaar te kunnen vergelijken. Op voorhand wordt er gedefinieerd welke afbeeldingen worden beschouwd als overschrijdend gedrag. Het is ook belangrijk dat er getest wordt waarbij de camera in verschillende posities staat. Hierdoor wordt er getest of het systeem nog steeds werkt uit verschillende perspectieven. Als laatste is het belangrijk dat een aantal foto's bij zijn waarbij de services zouden kunnen denken waarbij het om roken gaat, maar waarbij dit niet het geval is.

Aan de hand van de resultaten van de testen met de afbeeldingen kan er bekeken worden hoe het systeem moet zijn ingesteld om zo nauwkeurig mogelijk te zijn.

Om de snelheid van het systeem te bepalen word de tijd die het systeem nodig heeft om alle stappen uit te voeren die leiden tot het bepaalde resultaat gemeten. Deze stappen zijn: het nemen van een afbeelding, het versturen van de afbeelding naar de backend, de afbeelding doorsturen naar Google Cloud Vision en AWS Rekognition, en als laatste het filteren van de resultaten.

Om de kostprijs van het systeem te bepalen, wordt de kostprijs van beide systemen vergeleken en in kaart gebracht.

Wanneer alle drie de testen zijn uitgevoerd kan er besloten worden hoe het systeem het beste kan worden ingesteld om tot een zo snel, accuraat en kostvriendelijk resultaat te komen.

3 Vergelijking van de verschillende services

3.1 AWS Rekognition

3.1.1 Wat is AWS Rekognition

AWS Rekognition is een service van Amazon die ontwikkelaars van software toelaat om afbeeldingen te laten analyseren in hun applicaties. Met AWS Rekognition kunnen ontwikkelaars objecten, scenes en gezichten herkennen in afbeeldingen. Bovendien kan een developer met deze service gezichten vergelijken met elkaar. [4]

3.1.2 Eigenschappen van AWS Rekognition

Met AWS Rekognition kan een ontwikkelaar objecten zoals, dieren, voertuigen, gebouwen herkennen. Dat is de belangrijkste eigenschap voor dit project. [5]

AWS Rekognition biedt ook de mogelijkheid om gezichten te herkennen in meerdere afbeeldingen, zo kan een ontwikkelaar foto's filteren waarin dezelfde personen voorkomen. [5]

Daarnaast heeft AWS Rekognition ook nog andere eigenschappen zoals het herkennen van beroemdheden, het herkennen van een tekst in een afbeelding en het detecteren van ongepaste afbeeldingen. [5]

3.1.3 Het gebruik van AWS Rekogntion

Amazon bied ontwikkelaars een gemakkelijk te gebruiken *API* aan. Met simpele code kan een ontwikkelaar zijn afbeelding laten analyseren.

Ten eerste moet er een *client* worden aangemaakt. Hier worden dingen zoals de *credentials*, de regio, etc. ingesteld. Dit is te zien op figuur 8.

```
public static AmazonRekognition createClient(){
    ClientConfiguration clientConfig = new ClientConfiguration();
    clientConfig.setConnectionTimeout(30000);
    clientConfig.setRequestTimeout(60000);
    clientConfig.setProtocol(Protocol.HTTPS);

    AWSCredentialsProvider credentialsProvider = new ProfileCredentialsProvider();

    return AmazonRekognitionClientBuilder
            .standard()
            .withClientConfiguration(clientConfig)
            .withCredentials(credentialsProvider)
            .withRegion("eu-west-1")
            .build();
}
```

Figuur 8 Code om een client aan te maken

Vervolgens moet er een *request* worden aangemaakt. Hierin wordt er duidelijk gemaakt welke afbeeldingmoet geanalyseerd worden, hoeveel labels er maximaal worden teruggegeven etc. Dit is te zien op figuur 9.

```java
public void run() {

    String imgPath = "C:\\Users\\11600838\\Downloads\\man-smoking.jpg";
    byte[] bytes;
    try{
        bytes = Files.readAllBytes(Paths.get(imgPath));
    } catch (IOException e){
        System.err.println("failed to load image: " + e.getMessage());
        return;
    }
    ByteBuffer byteBuffer = ByteBuffer.wrap(bytes);

    AmazonRekognition rekognition = ClientFactory.createClient();

    DetectLabelsRequest request = new DetectLabelsRequest()
            .withImage(new Image().withBytes(byteBuffer))
            .withMaxLabels(15);
    DetectLabelsResult result = rekognition.detectLabels(request);

    List<Label> labels = result.getLabels();
    for(Label label : labels){
        System.out.println(label.getName() + ": " + label.getConfidence());
    }
}
```

Figuur 9 het sturen van de request

Deze code zou dan de resultaten in tabel 1 geven voor de foto op figuur 10.

Figuur 10 de voorbeeldfoto

Tabel 1 de resultaten van het voorbeeld

Omschrijving	zekerheid
Smoke	98.65%
Smoking	98.65%
Human	98.65%
Person	98.65%
Home Decor	83.48%
People	66.87%

3.2 Google Cloud Vision

3.2.1 Wat is Google Cloud Vision

Google Cloud Vision is een *API* dat ontwikkelaars toelaat om afbeeldingen te laten analyseren. Dankzij de *API* van Google Cloud Vision kunnen ontwikkelaars te weten komen wat er allemaal in een afbeelding aanwezig is. [6]

3.2.2 Eigenschappen van Google Cloud Vision

Met behulp van Google Cloud Vision kunnen ontwikkelaars verschillende objecten herkennen in afbeeldingen. Verder kan Google Cloud Vision net als AWS Rekognition gezichten en tekst in afbeeldingen herkennen. In tegenstelling tot AWS Rekognition biedt Google Cloud Vision geen ondersteuning om video's te analyseren. [6]

3.2.3 Het gebruik van Google Cloud Vision

Om de Google Cloud Vision *API* te gebruiken in een Java applicatie moet er eerst de juiste *dependency* worden toegevoegd, dit is te zien op figuur 11.

```
<dependency>
    <groupId>com.google.cloud</groupId>
    <artifactId>google-cloud-vision</artifactId>
    <version>1.64.0</version>
</dependency>
```

Figuur 11 dependency voor Google Cloud Vision

Vervolgens moet de afbeelding worden omgezet naar *bytes* en moet een *request* worden opgebouwd. Deze resultaten komen terug in de vorm van *AnnotateImageResponses*. Die dienen dan nog handmatig worden omgezet naar het formaat dat gebruikt word in de rest van de applicatie.

4 Literatuurstudie

Om te weten te komen hoe de technologieën het best kunnen worden gecombineerd is het belangrijk om te weten hoe de 2 afbeelding analyse *API's* verschillen van elkaar. Daarom worden er drie artikels onderzocht waarin Google Cloud Vision en AWS Rekognition met elkaar vergeleken worden.

4.1 Vergelijking van de beste API's voor computervisie [7]

Computervisie is een van de bekendste toepassing van *machine learning*. Een service zoals deze maken is moeilijk en kost veel tijd. Gelukkig bieden verschillende bedrijven deze service aan als een *API*. In een artikel van data sience central worden de meest bekende *API's* voor afbeeldinganalyse vergeleken.

Google is één van de beste bedrijven die *machine learning* gebruikt. De Google Cloud Vision *API* laat ontwikkelaars toe om afbeeldingen te laten analyseren. Deze *API* kan de elementen in een afbeelding analyseren, monumenten en landschappen herkennen. Deze *API* laat ontwikkelaars ook toe om soort gelijke afbeeldingen online op te zoeken.

Azure is een andere *cloud service*. Azure bied verschillende *API's* aan. Zo is er de *Computer Vision API* voor algemene object detectie. Er is de *Face API* die aan gezichtsherkenning kan doen. De *Optical Character Recognition* wordt gebruikt om tekst uit een afbeelding te kunnen halen. Azure werkt ook aan services die hetzelfde kunnen doen met video's, maar deze is nog maar enkel beschikbaar als een voorsmaakje.

Amazon Rekognition is een *API* die gebouwd is door Amazon. Dit is een service die goed samenwerkt met de andere services van Amazon. Net als de andere services kan Amazon elementen uit een afbeelding herkennen. Een van de sterkste punten van Amazon Rekognition is de mogelijkheid om mensen in een afbeelding of video te kunnen identificeren.

Clarifai is een jong bedrijf dat computervisie als een service aanbied. Elke taak van de visie werkt met een specifiek model. Er zijn bestaande modellen beschikbaar, maar sommige zijn nog maar in de bèta fase. Clarifai biedt ook de mogelijkheid om zelf een model op te bouwen. Clarifai bied ook de mogelijkheid om ongepaste inhoud te detecteren in een afbeelding.

International Bussiness Machines(IBM) Watson Visual Recognition is ook een service die met modellen werkt, maar heeft niet zo veel ingebouwde modellen als Clarifai.

Kairos is een service die zich specialiseert in het herkennen van gezichten. Deze service kan leeftijd, emoties en geslacht herkennen. Kairos is beschikbaar als *API* en als *Software development kit(SDK).*

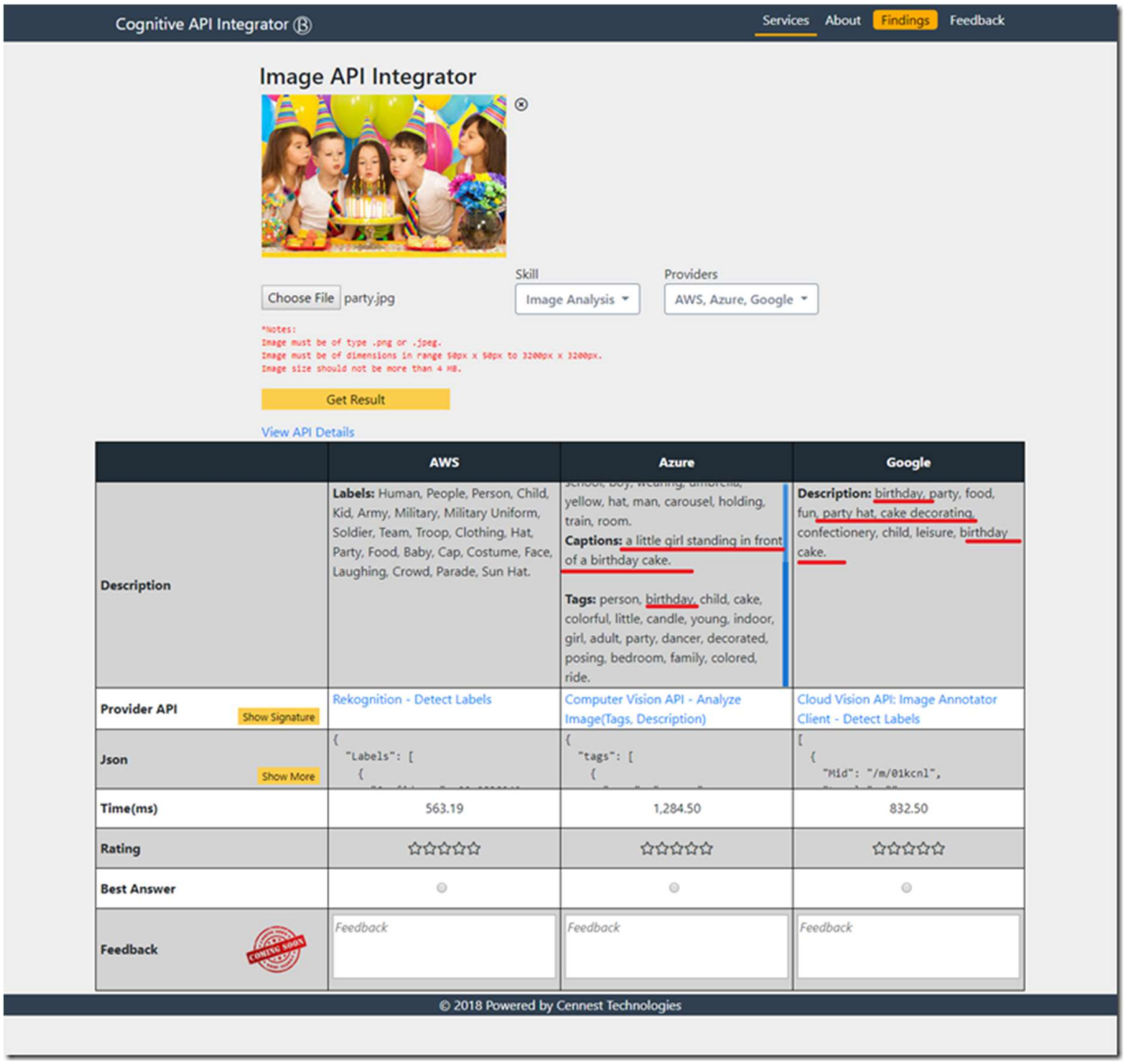
Er zijn veel verschillende *API's* beschikbaar. Veel van hen lijken op het eerste zicht hetzelfde, maar sommige leggen de focus meer op gezichtsherkenning zoals Kairos, en sommige leggen meer de focus op het bouwen van modellen zoals IBM en Azure.

4.2 Videoanalyse: Azure en Google komen boven als winnaars [8]

Om een betere vergelijking van de verschillende API's te verkrijgen, moeten deze getest worden. Anshulee Asthana van Cloud Zone heeft daarom Google Cloud Vision, Azure Vision en AWS Rekognition tegen elkaar getest, en er een artikel over geschreven.

Hij gebruikte *Cognitive API Integrator* om de antwoorden van deze 3 services te vergelijken met elkaar.

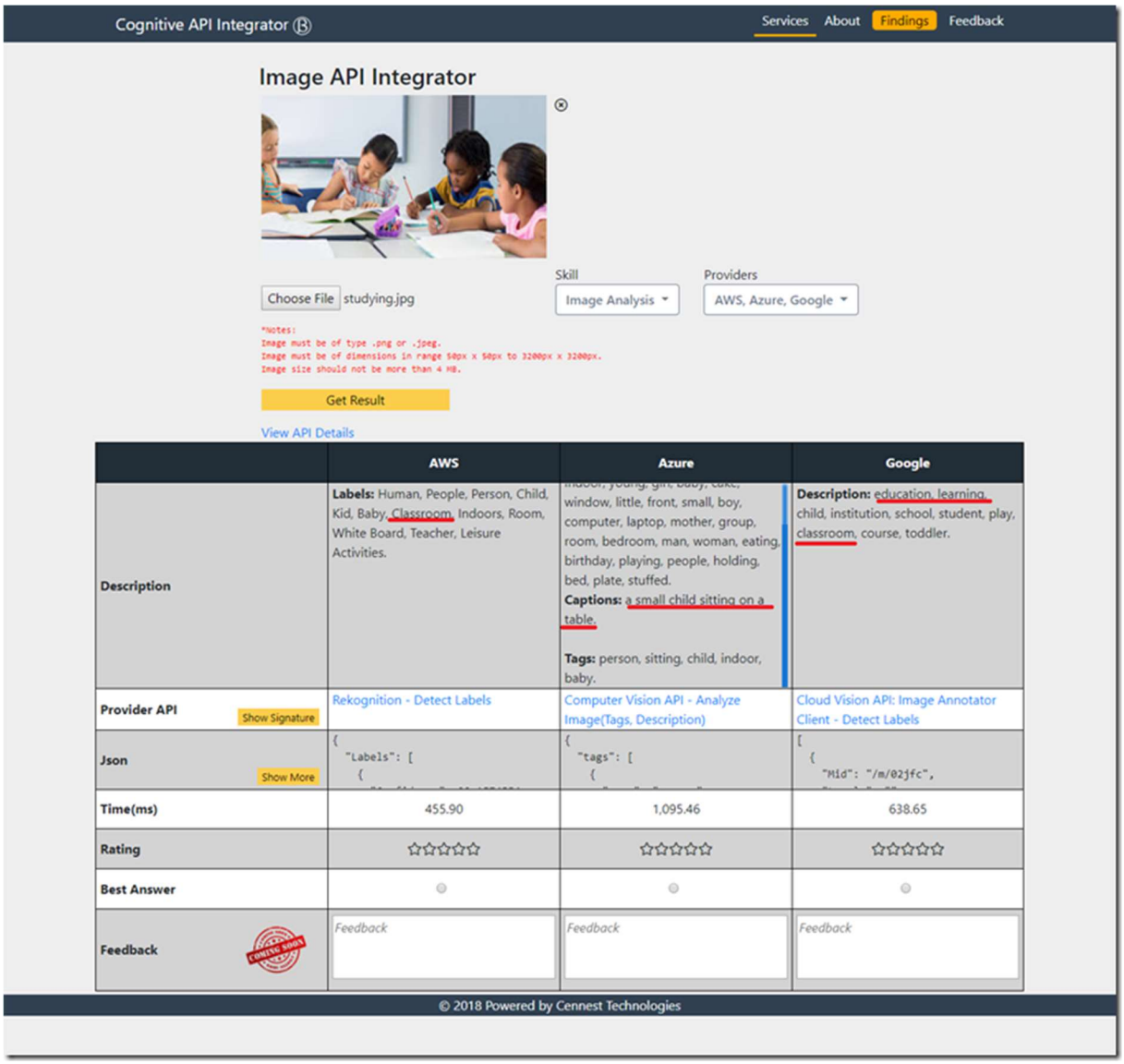
Tijdens het eerste voorbeeld werd aan alle drie de services gevraagd om een foto van een verjaardagsfeestje voor kinderen te analyseren, deze is te zien op figuur 12.



Figuur 12 de eerste test

Amazon gaf veel resultaten terug die relevant zijn, maar slaagt er niet in om de algemene context van een verjaardagsfeestje te vinden. Azure geeft een zin terug waarin beschreven staat wat er gebeurt en is zeer accuraat. Ook Google wist te achterhalen dat het om een verjaardag gaat.

Tijdens de tweede test lieten ze de drie services een afbeelding van een groep kinderen die aan het studeren zijn analyseren, deze afbeelding is te zien op figuur 13.



Figuur 13 de tweede test

Google wist veel relevante resultaten te vinden zoals kind, klaslokaal, leerkracht etc. Azure heeft een zin opgesteld die goed omschrijft wat er in de afbeelding te vinden is, maar gaf ook een heleboel irrelevante resultaten terug. AWS gaf ook relevante elementen terug, maar slaagde er weer niet in, om de context van de afbeelding te vinden.

Dit artikel noemt de proef op de som en doet een test om de verschillen te zoeken tussen Google Cloud Vision, AWS Rekognition en Azure Vision. Tijdens beide testen gaven alle drie de services resultaten terug die redelijk accuraat zijn. Tijdens deze testen bleek dat Azure en Google beter de context konden inschatten dan Amazon.

4.3 Vergelijking van Afbeeldingherkenning API's [9]

Uit het vorige artikel blijkt dat alle *API's* de meest voorkomende elementen uit een afbeelding kunnen vinden. Om de *API's* te testen moet er getest worden met afbeeldingen die iets unieker zijn dan de afbeeldingen uit het vorige artikel.

In dit onderzoek worden de volgende opties onderzocht: Google Cloud Vision, IBM Watson Visual Recognition, Amazon Rekognition, Azure Vision, Clarifai en Cloudsight. Er worden een aantal gevarieerde foto's geanalyseerd en de resultaten vergeleken.

Tijdens de eerste test wordt er een foto getoond waar in het midden een grote witte auto staat geparkeerd langs de rijbaan, deze is te zien op figuur 14. Elke geteste *API* heeft wel kunnen achterhalen dat er een auto in de afbeelding staat. Google Cloud Vision en Cloudsight zagen allebei dat het een luxewagen was, Azure en Clarifai konden allebei nog wat extra context toevoegen.

Figuur 14 een auto die geparkeerd staat

Tijdens de tweede test werd er een foto gebruikt van twee katten die op een bed liggen te rusten, deze foto is te vinden op figuur 15. De foto is ook genomen vanaf de bovenkant, wat het lastiger kan maken. Ook bij deze test gaven alle services aan dat het om katten ging. Bij deze test gaf Microsoft Computer Vision wel aan dat een van de katten een knuffeldier is. Tijdens deze test gaf Clarifai wel wat meer context aan dan de andere services.

Figuur 15 twee katten die op een bed liggen

Tijdens de derde test is er een foto van de grand canyon gebruikt, om te zien hoe de services een natuurlijk landschap behandelen. Dit is te zien op figuur 16. Dit was een moeilijkere opdracht voor de services, aangezien dat de informatie die terug gegeven werd veel vager is dan anders, enkel AWS Rekognition was er zeker van dat het om een ravijn ging.

Figuur 16 de grand canyon

Tijdens de vierde test, werd er getest hoe goed de services mensen konden herkennen, daarom werd er een foto gebruikt van mensen die poseren voor een foto op straat, zoals te zien is op figuur 17. Net zoals bij de andere foto's konden ze allemaal besluiten dat er een straat en personen aanwezig zijn, met uitzondering van Cloudsight. Maar toch kon geen van hen exact achterhalen wat ze precies aan het doen zijn. Cloudsight had wel de kleding herkent, maar kon geen personen vinden in de foto.

Figuur 17 mensen die poseren voor een foto

Tijdens de vijfde test werd er een schilderij gebruikt van een man met een paard, om te kijken of de services het verschil konden zien tussen een schilderij en een echte situatie. Dit schilderij is te vinden op figuur 18. Tijdens deze testen zijn er grotere verschillen tussen de *API's*. Het enigste wat alle *API's* het over eens waren, was dat er een persoon aanwezig is. Er waren grote verschillen tussen de *API's* over welk dier er op de afbeelding stond, enkel Clarifai kon zien dat het om een paard ging. Enkel IBM Watson Virtual Recognition en Amazon Rekognition konden niet achterhalen dat het om een schilderij ging.

Figuur 18 schilderij van een cowboy

Tijdens de laatste test werd er een foto gebruikt waar een fles wijn op staat, zoals te zien is op figuur 19. Op het etiket van de fles staan verschillende woorden met verschillende lettertypes. Ook tijdens deze laatste test konden alle *API's* achterhalen dat het een fles met alcoholische drank was, enkel Clarifai kon achterhalen dat het ging om een fles wijn. Enkel Google Cloud Vision en Azure Cloud Vision konden de tekst op het etiket lezen.

Figuur 19 een fles wijn

Ook uit dit artikel blijkt dat alle *API's* er in slagen om de meeste elementen uit een afbeelding te kunnen herkennen. Er is wel verschil in de omliggende elementen waar de *API's* verschillen, maar de zaken die het meeste in de afbeeldingen voorkomen kunnen ze allemaal wel achterhalen.

4.4 Besluit

In alle drie de artikels wordt vermeld dat alle services in staat zijn om resultaten uit de geteste afbeeldingen te halen die relatief goed bij de afbeelding passen. Er is wel echter een verschil, services zoals Google Cloud Vision en Azure Vision zijn beter in staat de context van de afbeelding te bepalen dan AWS Rekognition. Het is ook duidelijk dat sommige services zoals Azure en Kairos beter zijn in het achterhalen van gezichtskenmerken dan labels in een afbeelding.

5 Uitvoering van het onderzoek

5.1 De accuraatheid van het systeem

5.1.1 Overzicht van de geteste afbeeldingen

Een overzicht van de geteste afbeeldingen is te vinden in bijlage A. Afbeelding 1 is een afbeelding waar een persoon opstaat die niet aan het roken is. Aan de hand van deze foto kan er bepaald worden wat de services vinden als er niks aan de hand is. Afbeelding 2 is een afbeelding waarop een persoon te zien is die een pakje sigaretten vasthoud. De services zouden dit kunnen vinden, maar dit mag niet als overtredend gedrag vastgelegd worden. Op afbeelding 3 is er een persoon te zien die een sigaret vasthoud, maar hij houd deze nog ver weg van zijn mond. Aan de hand van deze afbeelding kan er bepaald worden of dat de services een sigaret kunnen vinden zonder dat de persoon aan het roken is. Veel van deze foto's hebben te maken met roken, toch overtreed de persoon op deze foto de regels niet. Het systeem mag dus dit gedrag niet classificeren als een overtreding op de regels.

Op afbeelding 4 is een persoon te zien die wel aan het roken is, hij houd de sigaret dan ook aan zijn mond. Op afbeelding 5 is dezelfde persoon nog steeds aan het roken, op deze foto is de sigaret niet meer zichtbaar, maar blaast de persoon net een klein wolkje rook uit. Op afbeelding 6 is een persoon te zien die aan het roken is, het verschil is dat deze keer de afbeelding genomen is langs de zijkant van de persoon. Aan de hand van deze afbeelding kan er getest worden of dat de services andere resultaten terug geeft als het perspectief van de afbeelding veranderd. Afbeelding 7 is ook een afbeelding die genomen is langs de zijkant van de roker, dit keer houd hij de sigaret niet in zijn mond, maar is de sigaret onderaan de afbeelding.

Op afbeelding 8 is een persoon te zien die aan het roken is, maar de afbeelding is van dichtbij genomen. Op afbeelding 9 staat de roker in dezelfde positie als op afbeelding 8, maar dit keer is de afbeelding genomen ver van de roker af. Door de resultaten van deze twee afbeeldingen te vergelijken kunnen bepalen of dat de afstand tussen de camera en de roker een verschil maakt.

5.1.2 De resultaten van de afbeeldingen

Als eerste wordt afbeelding 1 geanalyseerd. De resultaten van deze test zijn te zien in tabel 2. Zoals verwacht geeft geen enkele van de services aan dat er gerookt wordt, enkel AWS Rekognition geeft aan dat er een persoon op de afbeelding staat.

AWS Rekognition		Google Cloud Vision	
Beschrijving	Zekerheid	Beschrijving	Zekerheid
Person	99.4%	T-shirt	78.7%
Human	s99.4%	Room	65.7%
Clothing	99.1%	Singage	62.5%
Apparel	99.1%	Vacation	54.6%
Sleeve	95.8%	Sign	53.3%
Long Sleeve	86.5%		
Sweater	81.7%		
Sweatshirt	81.7%		

Tabel 2 resultaten van de eerste test

Op afbeelding 2 staat de persoon terwijl hij een pakje sigaretten in zijn hand heeft. Zowel AWS Rekognition als Google Cloud Vision konden niet herkennen dat het om een pakje sigaretten ging.

Google Cloud Vision dacht dat het om een elektronisch apparaat ging. De resultaten van deze tweede test staan in tabel 3.

AWS Rekognition		Google Cloud Vision	
Beschrijving	Zekerheid	Beschrijving	Zekerheid
Person	99.4%	Electronic Device	61%
Human	99.4%	T-shirt	50.2%
Clothing	99.1%		
Apparel	99.1%		
Pants	85.2%		
Man	80.9%		
Door	62.9%		
Denim	59.4%		
Jeans	59.4%		
Symbol	57.6%		
Face	56.9%		

Tabel 3resultaten van de tweede test

Op afbeelding 3 staat een persoon die een sigaret vast houd. Deze is wel nog ver van zijn mond verwijdert. Geen van beide services kan de sigaret identificeren, Google Cloud Vision vind wel dat de persoon aan het roken is, maar is er slechts 56.7% zeker van het resultaat. De resultaten van deze test zijn te zien in tabel 4.

AWS Rekognition		Google Cloud Vision	
Beschrijving	Zekerheid	Beschrijving	Zekerheid
Person	99.6%	T-shirt	71.6%
Human	99.6%	Room	65.7%
Finger	85.4%	Sign	61.6%
Clothing	67.2%	Smoking	56.7%
Apparel	67.2%	Signage	56.2%
Face	62%		
Man	61.6%		
Sleeve	58%		

Tabel 4 resultaten van de derde test

Op afbeelding 4 is een persoon te zien die wel degelijk aan het roken is. Zowel Google Cloud Vision als AWS Rekognition zijn bij deze test er heel zeker van dat er iemand aan het roken is. De resultaten van deze test zijn te zien in tabel 5.

AWS Rekognition		Google Cloud Vision	
Beschrijving	Zekerheid	Beschrijving	Zekerheid
Human	99.5%	Smoking	93.4%
Person	99.5%	Darts	57.9%
Smoking	99.3%	Tobacco products	53.7%
Smoke	99.3%	Gesture	51.7%
People	62.5%		
Teen	60.6%		

Tabel 5 resultaten van de vierde test

Op de vijfde afbeelding staat een persoon die aan het roken is, de sigaret is niet meer in beeld en hij blaast net een klein beetje rook uit. Zowel Google Cloud Vision en AWS Rekognition konden de rook in de afbeelding niet vinden. De resultaten van deze vijfde test zijn te vinden in tabel 6.

AWS Rekognition		Google Cloud Vision	
Beschrijving	Zekerheid	Beschrijving	Zekerheid
Human	99.4%	Hair	98.1%
Person	99.4%	Face	97.1%
Gate	94.9%	Chin	91.4%
Face	79.9%	Nose	907%
Railing	72.1%	Head	89%
Head	64.2%	Ear	87.9%
Outdoors	62.5%	Hairstyle	85.6%
Hair	60.1%	Neck	85.4%
Fence	56.5%	Male	84.9%
		Forehead	82.2%

Tabel 6 resultaten van de vijfde test

Op afbeelding 6 staat een persoon die aan het roken is. Het verschil met afbeelding 5 is dat deze afbeelding van de zijkant van de roker is genomen. Tijdens deze test is gebleken dat beide services nog steeds het roken kunnen herkennen. AWS Rekognition was er nog steeds heel zeker van het resultaat, Google Cloud Vision daarin tegen was veel minder zeker over het roken dan over het roken op afbeelding 6.

AWS Rekognition		Google Cloud Vision	
Beschrijving	Zekerheid	Beschrijving	Zekerheid
Person	99.6%	Nose	90.7%
Human	99.6%	Chin	86.5%
Smoking	93.9%	Male	81.9%
Smoke	93.9%	Ear	81.8%
Apparel	80.4%	Jaw	79.8%
Clothing	80.4%	Neck	69%
Teen	80.3%	Mouth	67.9%
Face	67.9%	Smoking	67.4%
Railing	65.6%	T-shirt	50.2%
Girl	58.3%		
Female	58.3%		

Tabel 7 resultaten van de zesde test

Afbeelding 7 is ook een afbeelding die van de zijkant van de roker is genomen. Het verschil is hier dat de sigaret niet aan de mond van de roker bevind, maar aan de onderkant van de afbeelding. Zowel Google Cloud Vision als AWS Rekognition vonden in deze foto geen elementen terug die met roken te maken hebben. Google Cloud Vision kon verrassend weinig elementen uit deze afbeelding halen, zoals te zien is in tabel 8.

AWS Rekognition		Google Cloud Vision	
Beschrijving	Zekerheid	Beschrijving	Zekerheid
Person	99.8%	Tree	68.7%
Human	99.8%	T-shirt	50.2%
Automobile	98.9%		
Car	98.9%		
Vehicle	98.9%		
Transportation	98.9%		
Outdoors	77.7%		
Parking	66.8%		
Parking lot	66.8%		
Garden	60%		

Tabel 8 resultaten van de zevende test

Afbeelding 8 is een close-up van iemand die aan het roken is. Zowel Google Cloud Vision en AWS Rekognition vinden hierbij allebei iemand die aan het roken is terug in de afbeelding. Dit is de enigste afbeelding waarin één van de services, namelijk Google Cloud Vision, de sigaret in de afbeelding kan herkennen. De resultaten van deze voorlaatste test zijn terug te vinden in tabel 9.

AWS Rekognition		Google Cloud Vision	
Beschrijving	Zekerheid	Beschrijving	Zekerheid
Person	99.8%	Smoking	95.6%
Human	99.8%	Nose	84.2%
Smoke	97.5%	Tobacco products	81.4%
Smoking	97.5%	Cigarette	75.6%
Face	88.4%	Cool	70.3%
People	67.6%	Mouth	67.9%
Teen	59.6%	Jaw	63%
		Ear	63%
		T-shirt	62.4%
		Gesture	51.7%

Tabel 9 resultaten van de achtste test

Tijdens de laatste test is afbeelding 9 getest. Afbeelding 9 is dezelfde afbeelding als afbeelding 8, maar in plaats van dichtbij, is de afbeelding van veraf genomen. Hier vonden Google Cloud Vision en AWS Rekognition beide geen elementen die met roken te maken hebben. De resultaten van deze laatste test zijn terug te vinden in tabel 10.

AWS Rekognition		Google Cloud Vision	
Beschrijving	Zekerheid	Beschrijving	Zekerheid
Person	99.7%	Standing	89.9%
Human	99.7%	Snapshot	84.3%
Apparel	97.9%	Photography	67.8%
Clothing	97.7%	Door	61.3%
Sleeve	95.8%	Sign	50.2%
Pants	94%		
Door	94%		
Long Sleeve	78.7%		
Man	66.4%		
Sitting	61.5%		
Sliding door	59.2%		
Jeans	56.5%		
Denim	56.5%		

Tabel 10 resultaten van de negende test

5.1.3 Besluit

Uit de uitgevoerde testen is gebleken dat Google Cloud Vision en AWS Rekognition over de grote lijn dezelfde antwoorden geven. Uit de testen is ook gebleken dat om het rookgedrag te moeten detecteren, de sigaret zich aan de mond van de roker moet bevinden. Enkel Google Cloud Vision gaf roken aan waar dit niet zou mogen, bij afbeelding 4. Bij deze test is gebleken dat Google Cloud Vision 56.7% zeker is van dat resultaat. Om dit resultaat te elimineren in het systeem moet dus de zekerheid minstens 60% bedragen. Ten slotte is nog uit deze testen gebleken dat AWS Rekognition in de meeste situaties beter het roken kon herkennen dan Google Cloud Vision. Voor de toepassing op het systeem om te kunnen detecteren of iemand aan het roken is, is er dus alleen maar nood voor AWS Rekognition. Het staat echter niet vast of dit ook het geval is voor andere soorten gedrag.

5.2 De prijs van het systeem

Zowel AWS Rekognition als Google Cloud Vision berekenen hun prijs per maand, afhankelijk van het aantal afbeeldingen die worden geanalyseerd.

De eerste 1000 afbeeldingen per maand die geanalyseerd worden door Google Cloud Vision zijn gratis. De volgende 4000 afbeeldingen die het systeem doorstuurt kosten $1.50 per 1000 stuks. Na 5000 afbeeldingen daalt de prijs tot $1.00 per 1000 afbeeldingen. Als het systeem dus 7500 afbeeldingen zou doorsturen naar Google Cloud Vision, zou de prijs voor deze service 7.5 dollar per maand kosten. [10]

Net als Google Cloud Vision bied AWS Rekognition ook een portie van de afbeeldingen gratis aan. Bij AWS Rekognition zijn de eerste 5000 afbeeldingen gratis. Dit is enkel voor nieuwe klanten voor de eerste 12 maanden. Daarna vallen deze gratis afbeeldingen weg. De eerste 1 miljoen afbeeldingen die geanalyseerd worden door de service kosten één dollar per 1000 afbeeldingen. De volgende negen miljoen kosten $0.80 per 1000 afbeeldingen, de volgende 90 miljoen afbeeldingen kosten $0.60 per 1000afbeeldingen. Na deze eerste 100 miljoen afbeeldingen blijft de prijs vast op $0.40 per 1000 afbeeldingen die geanalyseerd worden. Als we bij deze service dezelfde 7500 afbeeldingen lieten analyseren, zou ons dat twee en een halve dollar per maand kosten tijdens de eerste 12 maanden en zeven en een halve dollar voor de volgende maanden. [11]

In conclusie blijkt dat AWS Rekognition een goedkoper service is dan Google Cloud Vision. Mocht het systeem echter minder dan 1000 afbeeldingen per maand versturen naar de services, dan is Google Cloud Vision echter gratis. Als het systeem minder dan 5000 afbeeldingen verstuurd per maand, en het systeem niet langer dan een jaar gebruikt word, dan is AWS Rekognition de goedkopere oplossing.

5.3 De snelheid van het systeem

Aangezien dat we werken met twee systemen, zou dit een invloed kunnen hebben op de snelheid van het systeem. Door *multithreaded* te werken kunnen we echter deze twee acties tegelijkertijd uitvoeren. Het Systeem moet echter wachten tot beide acties zijn voltooid. Hierdoor is de snelheid van de applicatie afhankelijk van de traagste service. Natuurlijk is de snelheid van de verzoeken naar de service niet alleen afhankelijk van de snelheid van de analyse door de service, maar ook van de snelheid van het netwerk waarmee het verzoek wordt gestuurd.

Een verzoek om een afbeelding te laten analyseren door AWS Rekognition duurt 2.8 seconden. Een verzoek naar Google Cloud Vision is iets sneller en duurt 2.6 seconden.

In conclusie blijkt dus dat beide services ongeveer even snel zijn. De snelheid van het netwerk, De tijdsspanne tussen het nemen van de foto en de snelheid van de apparaten heeft een grotere invloed op de snelheid van de applicatie dan de snelheid van de services.

Conclusie

Uit het onderzoek is gebleken dat, zowel Google Cloud Vision als AWS Rekognition beide rookgedrag kunnen detecteren. Er is wel gebleken dat AWS Rekognition het gedrag van de persoon op de foto accurater kon inschatten dan Google Cloud Vision. Al was het in de meeste gevallen een gelijkspel.

Uit verder onderzoek is gebleken dat AWS Rekognition goedkoper is bij meerdere afbeeldingen per maand. Mocht het systeem echter minder dan 1000 afbeeldingen per maand versturen naar de services, dan is Google Cloud Vision gratis. Als het systeem minder dan 5000 afbeeldingen verstuurd per maand, en het systeem niet langer dan een jaar gebruikt word, dan is AWS Rekognition de goedkopere oplossing.

Tijdens de derde test, werd de snelheid van de services getest. Uit dit onderzoek is gebleken dat beide services ongeveer even snel zijn, al was Google Cloud Vision met een zeer kleine marge iets sneller. Dit verschil in snelheid is echter verwaarloosbaar, aangezien dat er factoren zijn zoals de snelheid van het netwerk die een groter verschil hebben op de snelheid dan de snelheid van de services.

Als we de resultaten uit alle drie de onderzoeken naast elkaar leggen dan kan daaruit geconcludeerd worden dat AWS Rekognition over de grote lijn een betere oplossing is dan Google Cloud Vision om roken te detecteren. Echter zou het systeem in de toekomst makkelijk kunnen worden uitgebreid om verschillende gedragen te detecteren, gedragen die Google Cloud Vision misschien beter kan herkennen dan AWS Rekognition.

Bibliografie

[1] „Vue.js," [Online]. Available: https://vuejs.org/v2/guide/. [Geopend 5 April 2019].

[2] „tutorialspoint," [Online]. Available: https://www.tutorialspoint.com/spring_boot/spring_boot_introduction.htm. [Geopend 2 Mei 2019].

[3] „Wikipedia," 27 April 2019. [Online]. Available: https://en.wikipedia.org/wiki/Flask_(web_framework). [Geopend 08 06 2019].

[4] Editorial Team, "Linkeit," Linke, 3 Juli 2017. [Online]. Available: https://www.linkeit.com/blog/what-is-aws-rekognition. [Accessed 15 Maart 2019].

[5] Amazon, „aws," Amazon, [Online]. Available: https://aws.amazon.com/rekognition/image-features/. [Geopend 15 Maart 2019].

[6] Bartosz, „Smashingmagazine," 9 Januari 2019. [Online]. Available: https://www.smashingmagazine.com/2019/01/powerful-image-analysis-google-cloud-vision-python/. [Geopend 19 Maart 2019].

[7] I. Bobriakov, „Data Science Central," 16 Augustus 2018. [Online]. Available: https://www.datasciencecentral.com/profiles/blogs/comparison-of-the-top-cloud-apis-for-computer-vision. [Geopend 3 April 2019].

[8] A. Asthana, „Cloud Zone," 8 augustus 2018. [Online]. Available: https://dzone.com/articles/cognitive-image-analysis-azure-and-google-come-out-1. [Geopend 4 April 2019].

[9] y. Keenan, „upwork," [Online]. Available: https://www.upwork.com/hiring/data/comparing-image-recognition-apis/. [Geopend 3 april 2019].

[10] „Google Cloud," Google, [Online]. Available: https://cloud.google.com/vision/pricing?hl=nl. [Geopend 03 05 2019].

[11] „AWS," Amazon, [Online]. Available: https://aws.amazon.com/rekognition/pricing/. [Geopend 03 05 2019].

Bijlagen

A. Foto's van het onderzoek

A. Foto's van het onderzoek

afbeelding 1 persoon die niet aan het roken is

afbeelding 2 persoon met een pakje sigaretten

afbeelding 3 persoon met een sigaret

afbeelding 4persoon die aan het roken is

afbeelding 5 persoon die een wolkje rook uitblaast

afbeelding 6 persoon die aan het roken is vanaf de zijkant

afbeelding 7 persoon met een sigaret onder in beeld

afbeelding 8 persoon die aan het roken is van dichtbij

afbeelding 9 persoon die aan het roken is van veraf