2023

AI-Model Onderzoek

Onderzoek naar de ontwikkeling van een AI-model voor het herkennen van defecte EV-batterijen.

AI EV-BATTERIJEN
TADRAŁA,PIOTR P.P.

O-PP-CMK

I. MANAGEMENT & SAMENVATTING

Dit document omschrijgt de ontwikkeling, validatie en optimalisatie van een AI-model met als doel om nauwkeurige prediction te generen van defecte EV-batterijen. Het document beschrijft het onderzoek en beantwoordt de 4 hoofdvragen, namelijk:

- Hoe wordt een ML model ontwikkeld
- Welke algoritmes zijn het meest geschikt
- Hoeveel data is er nodig voor het trainen van een ML model
- Hoe kan de nauwkeurigheid van het AI-Model worden getest en gevalideerd?

Kern van de Hoofdstukken

- **Data en Preprocessing:** Er is gekozen voor data-augmentatie om de dataset uitgebreider te maken, dit is vooral handig met de kleine dataset de we hebben ontvangen. Technieken zoals rotatie, flipping en zoom werden toegepast om de beperkte dataset te vergrootten.
- **Modelarchitectuur en Training:** Verschillende modelconfiguraties werden getest. Diepe lagen bleken minder nuttig voor de specifieke kenmerken van de dataset, terwijl hogere resoluties een positieve invloed hadden op de nauwkeurigheid.
- **Evaluatie en Validatie:** Voor het beoordelen van de prestaties bleek *Mean Squared Error (MSE)* het meest nuttig in ons geval. Uit de onderzoek bleek Cross-validatie de meest betrouwbare teststrategie, terwijl eenvoudige data-split gevoelig bleek voor bias.
- **Optimalisatie en Overwegingen:** Maatregelen zoals dropout en early stopping werden getest om overfitting te voorkomen. Het gebrek aan data bleef echter een beperkende factor.
- **Conclusies en Aanbevelingen:** Het gebruik van MSE en cross-validatie biedt een solide basis voor modelvalidatie.

II. VERSIEBEHEER & DISTRIBUTIE

VERSIEBEHEER

Versie	**Omschrijving**	**Auteur**
0.1	Eerste opzet van het document	Piotr Tadrala
0.2	Onderzoek naar algoritmes die geschikt zijn voor ons probleem	Piotr Tadrala
0.3	Onderzoek naar het hoeveelheid data dat er nodig is voor het trainen van een model	Piotr Tadrala
0.4	Onderzoek naar het evaluatie van een model en afronding van het document	Piotr Tadrala

DISTRIBUTIEGESCHIEDENIS

Versie	**Ontvangers**
0.1	Karin Dieleman & Berry Sanders
0.2	Karin Dieleman & Berry Sanders
0.3	Karin Dieleman & Berry Sanders
0.4	Karin Dieleman & Berry Sanders

III. WOORDENLIJST

Begrip	**Omschrinving**
INNER	Bedrijf, stakeholder van het project
AI	Artificial intelligence
EV	Electric vechicle
UI	User Interface, Frontend
UX	User Experience

INHOUDSOPGAVE

1.0 INLEIDING

INNER is een innovatieve startup gevestigd in Eindhoven, gericht op het optimaliseren van het gebruik van batterijen voor elektrische voertuigen (EV) vanuit een economisch en circulair perspectief. Hun doel is om de geschiktheid van EV-batterijen voor hergebruik te toetsen door middel van geavanceerde röntgenbeeldanalyse. Met hun technologie willen ze de veiligheid en economische waarde van EV-batterijen maximaliseren en de recycling van grondstoffen uitstellen totdat de batterij niet langer herbruikbaar is.

INNER heeft ons benaderd met de vraag hoe er een systeem ontwikkeld kan worden voor het detecteren van defecte EV-batterijen met behulp van AI. In dit document richt ik me op het onderzoeken hoe wij een model kunnen ontwikkelen voor dit probleem.

2.0 CONTEXT

2.1 INLEIDING

In dit hoofdstuk wordt de context geschetst waarin het project plaatsvindt. Het doel is om inzicht te geven in de probleemstelling, de huidige situatie van INNER, en de doelstelling van dit document.

2.2 HUIDIGE SITUATIE

INNER heeft de eerste commerciële CT-machine ontwikkeld die complete batterijpakketten kan scannen. Het bedrijf is momenteel op zoek naar een efficiënte manier om met behulp van een AI-model röntgenfoto's van EV-batterijen te analyseren en defecte batterijen te detecteren.

2.3 PROBLEEMSTELLING

Om een AI-model te ontwikkelen dat foto's kan scannen op defecten, moet er eerst een AI-model worden gemaakt. Dit model moet op basis van een dataset een suggestie kunnen geven over de staat van de batterij en zijn kennis uitbreiden met elke nieuwe scan.

2.3 DOELSTELLING

Met dit document wil ik mijn onderzoek presenteren over hoe we een dergelijk AI-model kunnen ontwikkelen en welke overwegingen een rol zullen spelen bij de ontwikkeling van een AI-model voor het detecteren van defecte EV-batterijen.

2.4 CONCLUSIE

Dit document onderzoekt hoe we een AI-model voor INNER kunnen ontwikkelen voor het detecteren van defecte EV-batterijen.

3.0 ONDERZOEKSAANPAK

3.1 INLEIDING

In dit hoofdstuk laat ik zien waar dit document zich op richt. De hoofdvraag en de bijbehorende deelvragen worden in kaart gebracht, en wordt gekeken hoe deze vragen beantwoord kunnen worden.

3.2 HOOFDVRAAG

Hoe kan een AI-model ontwikkeld worden voor het onderscheiden van defecte EV-batterijen op basis van röntgenfoto's.

3.3 DEELVRAGEN

Index	**Deelvraag**
D1	Hoe wordt een AI-model ontwikkeld.
D2	Welke algoritmes zijn het meest geschikt.
D3	Hoeveel data is er nodig voor het trainen van een AI-model.
D4	Hoe kan de nauwkeurigheid van het AI-model worden getest en gevalideerd?

3.4 ONDERZOEKSMETHODEN

Index	**Strategie**	**Methode**	**Verwachte uitkomst**
D1	Internet research	Literatuuronderzoek	Toelichting over de ontwikkeling van een AI-model.
D2	Internet research	Literatuuronderzoek	Lijst met de meest geschikte algoritmes.
D3	Internet research & Experiments	Literatuuronderzoek & Tests uitvoeren	Toelichting over de hoeveelheid data / nauwkeurigheid ratio.
D4	Internet research & Experiments	Literatuuronderzoek & Tests uitvoeren	Manier om de output van een model te valideren.

3.5 SCOPE

Dit document richt zich voornamelijk op de ontwikkeling van een AI-model en de validatie van de output. De UX/UI en de implementatie van het model in de uiteindelijke applicatie zullen hier niet worden behandeld.

3.6 CONCLUSIE

In dit hoofdstuk is de onderzoekaanpak toegelicht, gericht op de ontwikkeling van een AI-model voor het onderscheiden van defecte EV-batterijen op basis van röntgenfoto's. De hoofdvraag en deelvragen zijn in kaart gebracht, en de gebruikte onderzoeksmethoden zijn omschreven. Dit document richt zich voornamelijk op de technische aspecten en de validatie van de output, zonder in te gaan op UX/UI en implementatie.

4.0 HOE WORDT EEN AI-MODEL ONTWIKKELD?

4.1 INLEIDING

De eerste stap in de ontwikkeling van een model is het onderzoeken hoe een model überhaupt ontwikkeld dient te worden. In dit hoofdstuk zal ik ingaan op de requirements voor het ontwikkelen van een AI-model, het ontwikkelproces toelichten, de mogelijke algoritmes bespreken waaruit we kunnen kiezen, en uitleggen hoe het trainingsproces van het model zal verlopen.

4.2 REQUIREMENTS VOOR AI-MODEL ONTWIKKELING

PROBLEEM DEFINIEREN

- **Probleem Definieren:** Het probleem dat het model moet oplossen, moet duidelijk en meetbaar worden geformuleerd.
- **Datasets:** Het is cruciaal om voldoende data waarop het model getraind kan worden te verzamelen.
- **Algoritme kiezen:** Het juiste algoritme kiezen dat het meest geschikt is voor het probleem.
- **Framework kiezen:** Het meest geschikte framework kiezen waarin het model ontwikkeld zal worden, bijvoorbeeld TenserFlow of PyTorch.

- **Hardware:** Voor het trainen van een complex AI-model is voldoende rekenkracht cruciaal.

- Software Mind, (2024) How to create an AI Model?
- Yash Jain, (2023) How to create your own AI Mode from scratch.

4.3 ONTWIKKELPROCES

STAP 1: PROBLEEM DEFINIEREN

De eerste stap bij het ontwikkelen van een model is het definiëren van het probleem. Wat willen we met het model bereiken? In ons geval willen we onderscheid kunnen maken tussen goede en defecte EV-batterijen op basis van röntgenfoto's van de batterijen.

STAP 2: VERZAMEL VOLDOENDE DATA

De quality en quantity van de data hebben een grote invloed op de performance en nauwkeurigheid van het model. Het is daarom cruciaal om voldoende data te verzamelen. Daarnaast moet de data worden opgesplitst in trainings- en testsets, zodat het model op een betrouwbare manier gevalideerd kan worden.

STAP 3: FRAMEWORK SELECTEREN

Vervolgens moet het juiste framework worden gekozen. Een framework biedt de tools en libraries die nodig zijn voor het bouwen, trainen en optimaliseren van een model. De meest populaire frameworks zijn TensorFlow en PyTorch, die verschillende algoritmes ondersteunen. Bij de keuze van het framework moet je rekening houden met de complexiteit van het model. TensorFlow is bijvoorbeeld meer geschikt voor grootschalige projecten, waarbij PyTorch beter aansluit bij kleinere applicaties.

STAP 4: ONTWERP DE ARCHITECTUUR

Neurale netwerken vormen de kern van de meeste AI-modellen. Het ontwerpen van de architectuur houdt in dat je beslist over het aantal lagen, het type neuronen, de verbindingspatronen, en de activatie functies. Hierbij moet rekening gehouden worden met de complexiteit van het probleem.

STAP 5: TRAINING

Tijdens de trainingsfase leert het AI-model patronen herkennen in de data door de parameters van het model aan te passen op basis van de inputdata. Het model wordt getraind op de trainingsset, waarbij het gebruikmaakt van een geselecteerd algoritme om de voorspellingen te verbeteren.

STAP 6: EVALUATIE

Zodra het model getraind is, is het belangrijk om de performance te beoordelen. Afhankelijk van het type probleem kun je verschillende metrics gebruiken, zoals nauwkeurigheid of precisie. Op basis van de resultaten kunnen aanpassingen worden gedaan in de architectuur of het trainingsproces om de prestaties te verbeteren.

- Yash Jain, (2023) How to create your own AI Mode from scratch.
- Vihar Kurama (2024) PyTorch vs. TensorFlow: Key Differences to Know for Deep Learning

4.4 ALGORITMES

WAT IS EEN ALGORITME

Een algoritme is in in principe een set aan instructies die in de berekeningen gevolgd moeten worden. In het geval van AI is het de code die aangeeft hoe de AI moet leren en uiteindelijk beslissingen moet nemen. Hieronder worden de drie grootste algoritmes toegelicht.

SUPERVISED LEARNING ALGORITME

Supervised learning is een algoritme waarbij het model wordt getraind op een gelabelde dataset. Dit betekent dat de input data gekoppeld is aan de juiste output, waardoor het model leert om patronen te herkennen en voorspellingen te doen. Tijdens het trainingsproces probeert het model de juiste relaties te vinden tussen de input en de output, zodat het deze kennis kan toepassen op nieuwe, ongeziene data.

UNSUPERVISED LEARNING ALGORITME

Unsupervised learning is een algoritme waarbij in tegenstelling tot supervised learning het model leert zonder vooraf gelabelde data. Het model probeert zelf patronen te ontdekken in de data. Dit wordt vaak gebruikt om groepen of categorieën te vinden zonder dat van tevoren bekend is wat deze groepen zijn.

REINFORCED LEARNING ALGORITME

Reinforcement learning is een type machine learning waarbij een agent leert door middel van interactie met een omgeving. De agent ontvangt beloningen of straffen op basis van zijn acties, waardoor hij leert welke acties de beste resultaten opleveren. Het doel is om een strategie te ontwikkelen die de totale beloning maximaliseert.

- Tableau, (onbekend) Artificial intelligence (AI) algorithms: a complete overview
- Raj Dasgupta, (2023) Reinforced learning: AI Algorithms, Types & Examples

4.5 TRAINING

Training is het proces waarbij een AI-model leert van een dataset. Tijdens dit proces past het model zelfstandig zijn parameters aan om patronen en relaties in de data te herkennen. Het model ontvangt inputgegevens en voorspelt een output. Deze voorspellingen worden vervolgens afhankelijk van de algoritme vergeleken met de werkelijke resultaten, en op basis van deze feedback worden aanpassingen gemaakt om de nauwkeurigheid te verbeteren.

DATASETS

Voor het trainingsproces worden meestal twee soorten datasets gebruikt. Het model wordt getraind op een trainingsset, waarna de nauwkeurigheid gemeten wordt op een trainingsset. Dit maakt het mogelijk om de nauwkeurigheid op een betrouwbare manier te meten.

WAT IS HET DOEL?

Het doel van de training is om het model in staat te stellen om nauwkeurige voorspellingen te doen op basis van nieuwe data. Door het model te trainen met een dataset kan het leren welke kenmerken belangrijk zijn voor het maken van beslissingen.

EPOCHS

Een epoch is een iteratie van een dataset waarop het model wordt getraind. Tijdens het trainen van een model wordt dezelfde dataset meerdere keren gebruikt, waarbij elke iteratie een epoch genoemd wordt. Door het model meerdere keren te trainen met een dataset kan het zijn parameters blijven optimaliseren voor betere performance. Echter, als er te veel epochs worden gebruikt, kan het overtraind raken, wat leidt tot lagere performance.

- Michael Chen, (2023) What Is AI Model Training & Why Is It Important?
- Simplilearn, (2023) What is Epoch in Machine Learning?

4.6 VORBEELD UIT PAKTIJK

Om te laten zien hoe een eenvoudig CNN-supervised model in de praktijk kan worden gebouwd en getraind, heb ik zelf een model ontwikkeld. Zie hiervoor het document "Supervised Learning Model."

4.7 CONCLUSIE

In dit hoofdstuk heb ik onderzocht welke requirements er zijn voor het ontwikkelen van een model, hoe een model vanaf scratch wordt ontwikkeld en het proces in een stappenlijst weergegeven. Daarnaast zijn de algoritmes besproken die gebruikt kunnen worden, en is het trainingsproces toegelicht. In het volgende hoofdstuk wil ik onderzoeken welk specifieke algoritme het meest geschikt is voor ons probleem.

5.0 WELKE ALGORITMES ZIJN HET MEEST GESCHIKT?

5.1 INLEIDING

In onze situatie is het niet effectief om een supervised learning model te gebruiken dat op duizenden foto's wordt getraind, voornamelijk omdat onze dataset zeer beperkt is. Een nog groter probleem is dat we een asymmetrische dataset zullen ontvangen, waarbij we geen of vrijwel geen foto's van defecte batterijen zullen hebben. In dit document zal ik onderzoeken hoe we dit specifieke scenario kunnen aanpakken.

5.2 COMPONENTHERKENNING DOOR MIDDEL VAN GRID-AUGMENTATIE

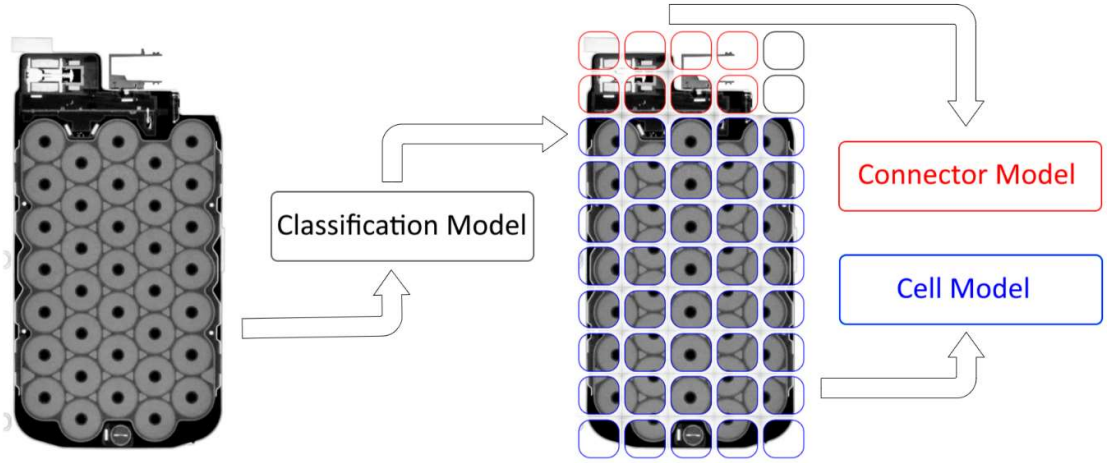
5.2.1 CONTEXT

Om het probleem van het gebrek aan data om ons model op te trainen aan te pakken, heb ik een mogelijke oplossing bedacht: component-specifieke modellen. Hierbij wordt een foto van een EV-batterij opgedeeld in een grid, waarbij elk blokje een nieuwe afbeelding is. Deze afbeeldingen worden vervolgens door een model geclassificeerd, bijvoorbeeld als een batterijcel, een connector, of gewoon niks.

Dankzij deze aanpak creëren we kleine samples van specifieke componenten, die door een component-specifiek model kunnen worden geanalyseerd om te bepalen of ze defect zijn. Hiermee genereren we meer data en kunnen we gespecialiseerde modellen ontwikkelen die zich richten op kleine samples om te beoordelen of een bepaald onderdeel defect is. Bovendien kan het model specifiek aangeven waar in een foto een defect wordt gedetecteerd, in plaats van alleen een binair antwoord te geven per batterij.

5.2.2 VISUALISATIE

Visualisatie waarbij een CT-scan van een batterij wordt opgesplitst in een grid van losse afbeeldingen. Deze losse afbeeldingen worden vervolgens door een ML-model geclassificeerd in specifieke componenten, waarna de geclassificeerde componenten worden geanalyseerd door een in dat specifiek component gespecialiseerd model.



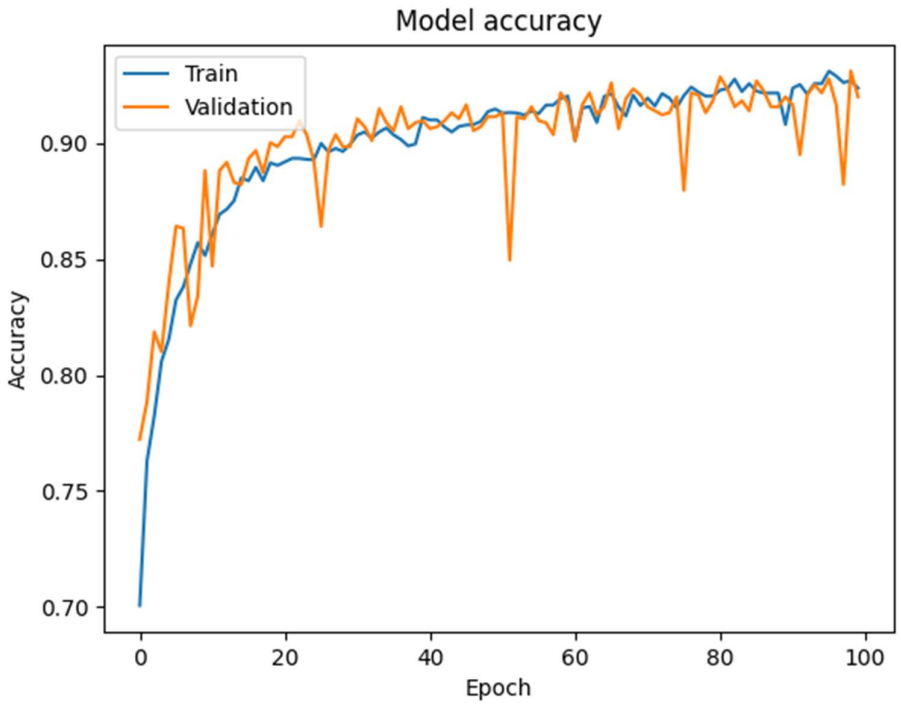
5.2.3 CLASSIFICATIE MODEL

Voor het ontwikkelen van een classificatiemodel is een supervised model het meest geschikt. Een unsupervised model maakt geen gebruik van gelabelde data, waardoor het moeilijk is om zowel een goede nauwkeurigheid te behalen als deze nauwkeurigheid te verifiëren.

- Geeksforgeeks, (2024) Supervised and Unsupervised Learning

5.2.4 VOORBEELD UIT PRAKTIJK

Om dit te demonstreren, heb ik een kleine demo gemaakt. Al met maar een afbeelding van een CT-scan heb ik een nauwkeurigheid van 93% kunnen bereiken door deze methode toe te passen. Zie de documentatie 'Supervised Classificatie Model' voor een uitgebreide toelichting.



5.3 DEFECTDETECTIE MET ASYMETRISCHE DATASETS

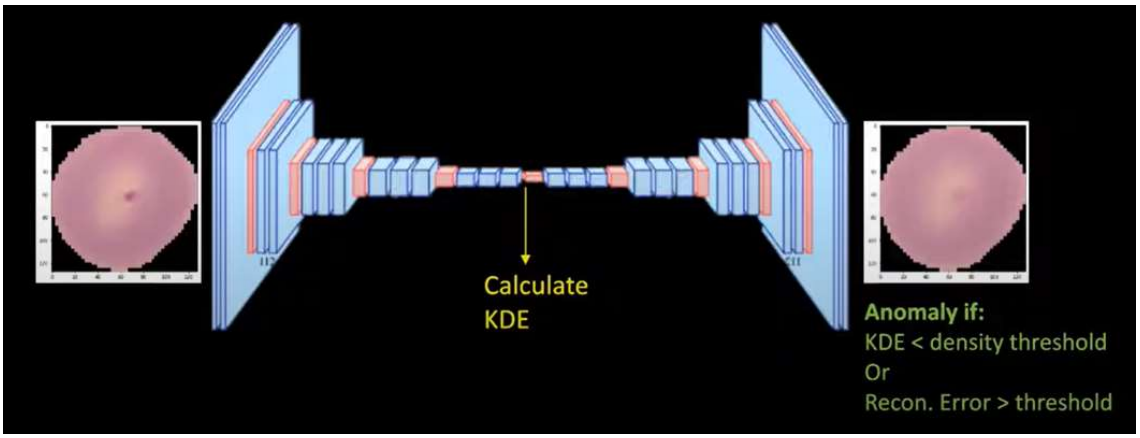
5.3.1 CONTEXT

Los van het probleem dat we zeer weinig data zullen ontvangen, is het ook nog zo dat we uitsluitend data van goede batterijen zullen krijgen. Dit maakt het niet mogelijk om het voorbeeldmodel uit hoofdstuk 4, waarbij we een model op twee datasets trainen, toe te passen. Om dit probleem aan te pakken, kunnen we een concept genaamd 'Anomaly Detection' toepassen.

5.3.2 ANOMALY DETECTION

Anomaly Detection model maakt het mogelijk om een model te trainen met asymmetrische data. Het model wordt getraind op een dataset, en vervolgens wordt alles wat daar niet op lijkt, binnen een bepaalde treshold, als een afwijking (anomaly) herkend. Heel simpel gezegd, wat er gebeurt is het volgende: een afbeelding wordt

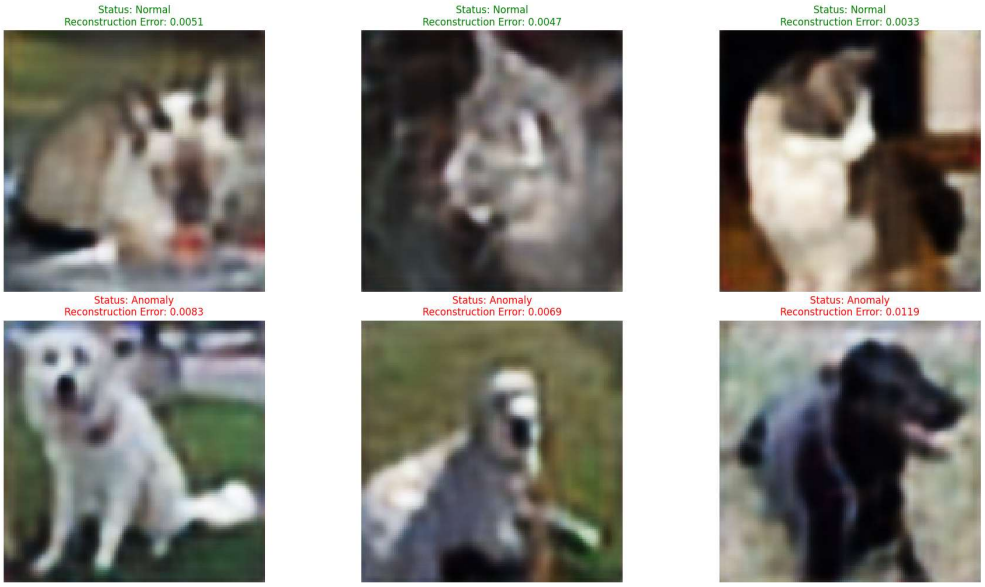
eerst gedownscaled en daarna weer geupscaled, waarna de reconstruction error wordt berekend. Omdat het model op een specifieke dataset is getraind, zal het tijdens de reconstruction altijd proberen te reconstructen naar de patronen waarop het getraind is. Wanneer je als input iets gebruikt waar het model niet op is getraind, zal de reconstructiefout veel hoger zijn.



- DigitalSreeni, (2022) Identifying anomaly images using convolutional autoencoders.

5.2.3 VOORBEELD UIT PRAKTIJK

Om te onderzoeken of de besproken oplossing ook in de praktijk werkt, heb ik een anomaly detection-model gebouwd. Voor de tests heb ik het model getraind op 1.000 afbeeldingen van katten en heb ik vervolgens gekeken of het in staat was om katten van honden te onderscheiden. Voorbeeld van de resultaten:



5.4 CONCLUSIE

Vanwege het gebrek aan data en de asymmetrische dataset is het onmogelijk om een eenvoudig model te ontwikkelen dat op basis van een CT-scan direct aangeeft of de batterij defect is of niet. De voorgestelde oplossing, waarbij de CT-scan eerst in componenten wordt opgesplitst met behulp van een classificatie model, en vervolgens component-specifieke anomaly detection-modellen worden gebruikt, lijkt op de eerste ogenblik positieve resultaten op te leveren. Echter, zal deze methode uiteindelijk gevalideerd moeten worden met werkelijke data, waaronder scans van een defecte batterij.

6.0 HOEVEEL DATA IS ER NODIG VOOR HET TRAINEN VAN EEN AI-MODEL?

6.1 INLEIDING

Voor het trainen van een model is het nuttig om vooraf te bepalen hoeveel data nodig is om een inschatting te maken van de verwachte accuracy. Daarnaast is het ook belangrijk om te weten hoe je de beschikbare data optimaal kunt gebruiken. In dit hoofdstuk onderzoek ik hoe we de data zo effectief mogelijk kunnen inzetten voor onze modellen.

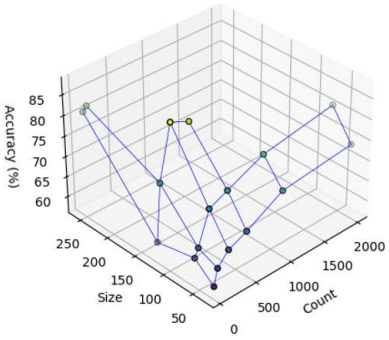
Voor alle tests in dit onderzoek heb ik gebruik gemaakt van data-augmentatie, zoals zoom, width & height shift, rotation en image flip.

6.2 INVLOED VAN RESOLUTIE EN DATAGROOTTE

Om de invloed van de resolutie van de afbeeldingen en de datasetgrootte op de accuracy te onderzoeken, heb ik in totaal 36 modellen getraind. Elk model is getest met een unieke combinatie van het aantal afbeeldingen per categorie en de resolutie. De resultaten kunnen kort worden samengevat: hoe meer, hoe beter. Bij een toename van zowel de resolutie als het aantal afbeeldingen was een duidelijke stijging in de accuracy zichtbaar.

size / count	50	100	250	500	1000	2000
32	58%	62%	65%	67%	72%	74%
64	62%	64%	74%	78%	79%	81%
128	60%	74%	77%	85%		
256	81%	82%				

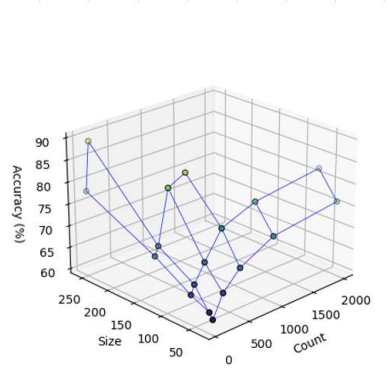
image_size	image_count_per_category	epochs	accuracy
32	50	25	58%
32	100	25	62%
32	250	25	65%
32	500	25	67%
32	1000	25	72%
32	2000	25	74%
image_size	image_count_per_category	epochs	accuracy
64	50	25	62%
64	100	25	64%
64	250	25	74%
64	500	25	78%
64	1000	25	79%
64	2000	25	81%
image_size	image_count_per_category	epochs	accuracy
128	50	25	60%
128	100	25	74%
128	250	25	77%
128	500	25	85%
128	1000	25	
128	2000	25	
image_size	image_count_per_category	epochs	accuracy
256	50	25	81%
256	100	25	82%
256	250	25	
256	500	25	
256	1000	25	
256	2000	25	



Om te bepalen of een combinatie op korte termijn een 'headstart' heeft, maar op lange termijn minder effectief blijkt te zijn dan een andere, heb ik dezelfde test opnieuw uitgevoerd, maar over 50 epochs in plaats van 25. Hieruit bleek dat wanneer een combinatie efficiënter is dan een andere, het ook zo blijft op de langere termijn.

size / count	50	100	250	500	1000	2000
32	63%	61%	66%	70%	74%	76%
64	65%	67%	71%	77%	80%	82%
128	70%	72%	84%	86%		
256	78%	89%				

image_size	image_count_per_category	epochs	accuracy
32	50	50	63%
32	100	50	61%
32	250	50	66%
32	500	50	70%
32	1000	50	74%
32	2000	50	76%
image_size	image_count_per_category	epochs	accuracy
64	50	50	65%
64	100	50	67%
64	250	50	71%
64	500	50	77%
64	1000	50	80%
64	2000	50	82%
image_size	image_count_per_category	epochs	accuracy
128	50	50	70%
128	100	50	72%
128	250	50	84%
128	500	50	86%
128	1000	50	
128	2000	50	
image_size	image_count_per_category	epochs	accuracy
256	50	50	78%
256	100	50	89%
256	250	50	
256	500	50	
256	1000	50	
256	2000	50	



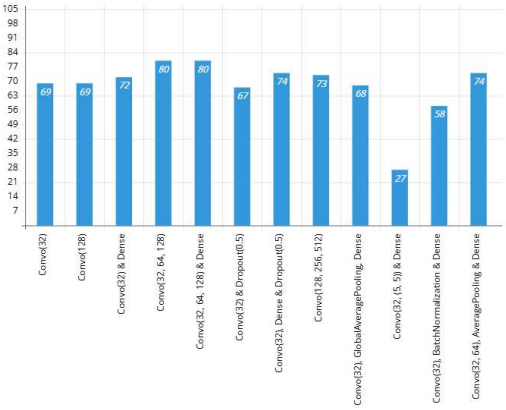
6.3 INVLOED VAN DE ARCHITECTUUR

Vervolgens heb ik onderzocht welke architectuurkeuzes in ons geval het meest effectief zijn. Hiervoor heb ik 12 modellen getraind op de batterijen data die we hebben ontvangen. Alle modellen zijn getraind op 1.000 afbeeldingen met een resolutie van 128 pixels gedurende 50 epochs. De nauwkeurigheid is vervolgens gemeten op een set van 250 afbeeldingen die de modellen vooraf niet hadden gezien. De resultaten hiervan zijn weergegeven in de onderstaande grafiek.

Uit het onderzoek blijkt dat het toevoegen van meer convolution lagen een meerwaarde heeft, aangezien deze beter presteren dan een enkele Conv(32) layer. Waarbij diepere convolution lagen (zoals 128, 256, 512) geen betere prestaties opleveren, wat mogelijk te verklaren is door het beperkte detailniveau in de afbeeldingen.

Dropout lijkt een goede practice te zijn, wat blijkt uit de hogere nauwkeurigheid van Conv(32), Dense & Dropout ten opzichte van Conv(32) & Dense. Dropout zorgt ervoor dat een percentage van de neuronen tijdens het trainen wordt genegeerd, waardoor het model minder afhankelijk wordt van specifieke neuronen. Op de lange termijn kan dit nuttig zijn om overfitting te voorkomen.

architecture	image_count	image_size	epochs	overfitted	accuracy
Convo(32)	1000	128	50	no	69%
Convo(128)	1000	128	50	no	69%
Convo(32) & Dense	1000	128	50	no	72%
Convo(32, 64, 128)	1000	128	50	no	80%
Convo(32, 64, 128) & Dense	1000	128	50	no	81%
Convo(32) & Dropout(0.5)	1000	128	50	no	67%
Convo(32), Dense & Dropout(0.5)	1000	128	50	no	74%
Convo(128, 256, 512)	1000	128	50	no	73%
Convo(32), GlobalAveragePooling, Dense	1000	128	50	no	68%
Convo(32, (5, 5)) & Dense	1000	128	50	no	27%
Convo(32), BatchNormalization & Dense	1000	128	50	yes	58%
Convo(32, 64), AveragePooling & Dense	1000	128	50	no	74%



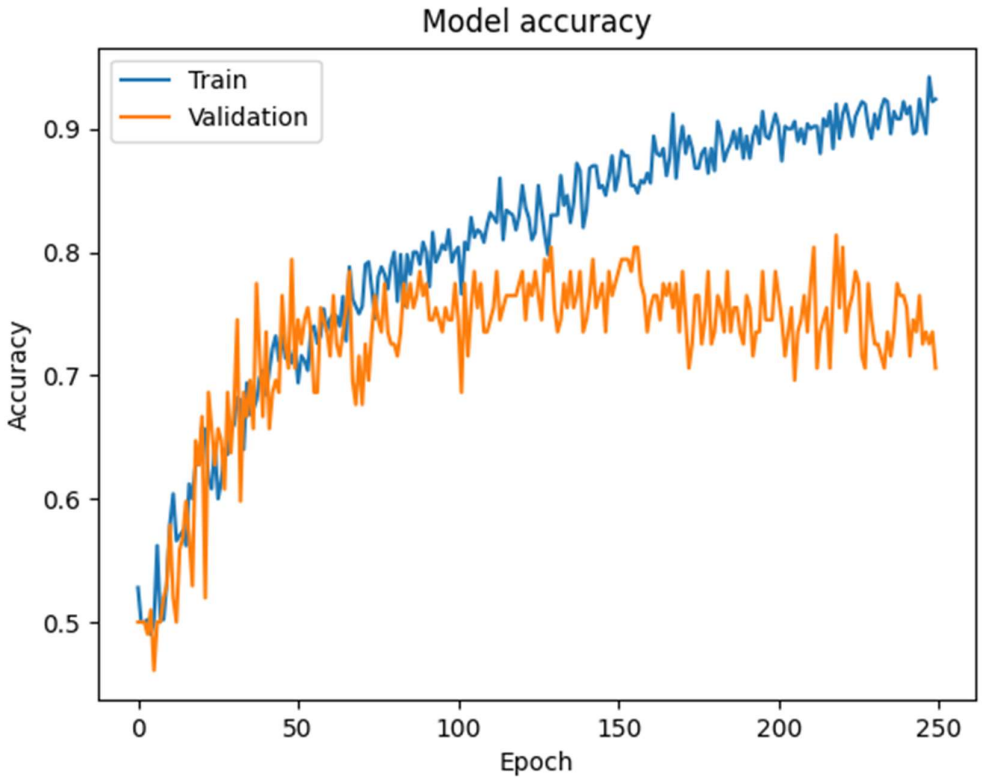
6.4 OVERFITTING

6.4.1 WAT IS OVERFITTING

Overfitting treedt op wanneer een model begint te focussen op irrelevante patronen in de data. Hierdoor lijkt het model goed te presteren op de trainingsdata, maar wanneer het wordt getest op nieuwe, ongeziene data, daalt de nauwkeurigheid.

6.4.2 OVERFITTING IDENTIFICEREN

Overfitting van een model is eenvoudig te herkennen. Het is te herkennen aan een stijgende accuracy op de trainingsdata, terwijl de nauwkeurigheid op de validatiedata na een bepaald punt begint af te nemen.



6.4.3 MAATREGELEN TEGEN OVERFITTING

• **Grotere dataset** Door de dataset te vergroten, krijgt het model meer data om op te trainen, waardoor die zicht minder snel op irrelevante patronen begint te richten.

• **Dropout**: Met Dropout worden bij elke iteratie willekeurig neuronen genegeerd. Dit voorkomt dat het model afhankelijk wordt van specifieke neuronen en helpt overfitting te verminderen.

• **Data Augmentatie:** Door data-augmentatie toe te passen, kan de trainingsdataset synthetisch worden vergroot, wat zorgt voor meer variatie van het model.

• **Early Stopping:** Door het model te monitoren en het trainingsproces te stoppen zodra tekenen van overfitting zichtbaar worden, kan verdere overfitting worden voorkomen.

•Taurun, (2024) How to handle overfitting in TensorFlow models?

6.2 CONCLUSIE

Dankzij de uitgevoerde tests kwam ik tot de volgende conclusies: hoe hoger de resolutie, hoe beter de nauwkeurigheid. Het was echter, vanwege een gebrek aan data, niet mogelijk te onderzoeken op welk punt een verdere toename in resolutie geen effect meer heeft op de nauwkeurigheid.

Daarnaast bleken in ons geval "diepe" lagen (conv 128, 256, 512) geen toegevoegde waarde te hebben. Dit is mogelijk te verklaren door het gebrek aan details in de gebruikte afbeeldingen.

Om het model met een groter aantal epochs te trainen zonder overfitting, kunnen maatregelen zoals een grotere dataset, dropout, data-augmentatie en early stopping worden toegepast.

7.0 HOE KAN DE NAUWKEURIGHEID VAN HET AI-MODEL WORDEN GETEST EN GEVALIDEERD?

7.1 INLEIDING

In dit hoofdstuk wordt onderzocht hoe de nauwkeurigheid van een ML-model effectief kan worden getest en gevalideerd. Het doel is om te beoordelen hoe goed het model presteert op nieuwe data. Hierbij worden testmethoden, evaluatiemetrics en validatieresultaten besproken.

7.2 WAT IS DE DOEL VAN MODEL EVALUATIE?

Model evaluatie heeft als doel ervoor te zorgen dat een ML model goed en betrouwbaar in praktijk gebruikt kan worden. De belangrijkste doelen zijn:

- **Prestatie meten:** Het model beoordelen op nieuwe data met metrics, afhankelijk van het type model kunnen er verschillende metrics gemeten worden.
- **Generalization:** Evalueren of het model goed werkt met nieuwe data en niet alleen goed presteert op de trainingsdata.

- **Overfitting & Underfitting:** Controleren of het model niet te complex is (Overfitting) of te simpel is (Underfitting).
- **Modellen vergelijken**: Verschillende modellen of iteraties vergelijken om te bepalen welke het beste presteer met behulp van metrics en teststrategien.

- Daniella, (2024) How do you evaluate a Machine Learning Model? https://en.innovatiana.com/post/how-to-evaluate-ai-models

7.3 VALIDATIE METRICS

Validatie metrics worden gebruikt om de prestaties van AI-modellen te evalueren. Het kiezen van de juiste metrics is cruciaal om de nauwkeurigheid en effectiviteit van een model te beoordelen. Hierbij een paar van de veel gebruikte metrics:

- **Accuracy:** Accuracy meet het percentage correcte voorspellingen van het totale aantal voorspellingen. Geschikt voor classificatie modellen met gelabelde datasets.
- **MSE:** Mean Squared Error (MSE) is geschikt voor regressie taken en meet het gemiddelde kwadraat van de verschillen tussen voorspelde waarden en werkelijke doelwaarden.
- **F-1:** De F1-score combineert precisie (hoeveel van de voorspellingen correct zijn) en recall (hoeveel van de werkelijke positieve gevallen worden gevonden) in één getal. Het is vooral handig bij asynchronische gelabelde datasets om een beter beeld te krijgen van de prestaties van een model.

- Ajiboye Abayomi Adewole, (2023) Understanding the class of losses in TensorFlow for Deep Learning https://medium.com/@abayomiajiboye46111/understanding-the-class-of-losses-in-tensorflow-for-deep-learning-4bd71c8e52a7
- Rohit kindu, (2022) F1 Score in Machine Learning: Intro & Calculation https://www.v7labs.com/blog/f1-score-guide

7.4 TESTSTRATEGIEËN

Om een model te testen, moet er een passende teststrategie worden toegepast. Er zijn verschillende methodes en tools beschikbaar om ML-modellen te evalueren. Hier zijn enkele voorbeelden:

7.3.1 DATA SPLIT (TRAIN/TEST)

Hierbij wordt de dataset opgedeel in een traingsset en een testset. Het model wordt getraind met de trainingsdata en getest met de testdata.

- **Voordelen:** Snel een eenvoudig toe te passen. Geeft een eerste indruk van de prestaties van het model.
- **Nadelen:** Kan bias veroorzaken als de data asymetrisch verdeeld is. Hierdoor kan het model minder goed werken op nieuwe data

7.3.2 CROSS-VALIDATIE

Cross-Validatie verdeelt de data in folds (subsets). Het model wordt vervolgens getraind, waarbij telkens een andere subset wordt gebruik als validatieset, en de rest als trainingsset

- **Voordelen:** Betrouwbaarder dan een simplere data split omdat alle data wordt gebruikt voor zowel trainings als validatie.
- **Nadelen:** is resource heavy, vooral bij grotere datasets.

7.3.3 STRATIFIED CROSS-VALIDATION

Een speciale vorm van cross-validatie waarbij elke subset dezelfde verhouding van klassen bevat als de volledige dataset.

- **Voordelen:** Ideaal voor datasets met een asymetrische dataset verdeling, war sommige klassen minder vaak voorkomen
- **Nadelen:** Ingewikkelder om te implementeren

- Daniella, (2024) How do you evaluate a Machine Learning Model? https://en.innovatiana.com/post/how-to-evaluate-ai-models

7.4 CONCLUSIE

In ons geval lijkt het gebruik van MSE-metrics het meest zinvol. Hiermee kunnen we de reconstruction error van het model analyseren, wat ons helpt om op basis daarvan anomalies te identificeren. Qua teststrategien lijkt cross-validation het meest geschikt, omdat deze betrouwbaarder en omdat alle data gebruikt wordt voor zowel training als validatie wat nuttig kan zijn bij kleinere datasets.

8.0 CONCLUSIE

Op basis van de uitgevoerde analyses, experimenten en evaluaties kan geconcludeerd worden dat zowel de gekozen architectuur als de toegepaste validatiestrategieën een cruciale rol spelen in de prestaties van het ML model. Het gebruik van MSE lijkt nuttig te zijn om reconstruction errors te analyseren en anomalieën te identificeren. Daarnaast bood cross-validatie een betrouwbaardere evaluatie van het model in vergelijking met een eenvoudige datasplitsing, omdat het een uitgebreider gebruik van de beschikbare data mogelijk maakte.

De resolutie van de inputdata bleek significant bij het verbeteren van de nauwkeurigheid, hoewel het gebrek aan data een beperking vormde om de optimale grens te bepalen. Verder bleken diepe lagen in het model weinig toegevoegde waarde te hebben, waarschijnlijk door het beperkte detailniveau van de gebruikte afbeeldingen.

Verder kwam ik tot de conslusie dat het gebruik maken van data-augmentatie, dropout en early stopping om overfitting tegen te gaan en het model robuuster te maken enorm nuttig kan zijn.

9.0 BRONEN

- Software Mind, (2024) How to create an AI Model? https://softwaremind.com/blog/how-to-create-an-ai-model/
- Yash Jain, (2023) How to create your own AI Mode from scratch. https://techgranth.medium.com/how-to-create-your-own-ai-model-from-scratch-10-easy-steps-with-code-e17b1c94377c
- Vihar Kurama (2024) PyTorch vs. TensorFlow: Key Differences to Know for Deep Learning https://builtin.com/data-science/pytorch-vs-tensorflow

- Tableau, (onbekend) Artificial intelligence (AI) algorithms: a complete overview https://www.tableau.com/data-insights/ai/algorithms
- Raj Dasgupta, (2023) Reinforced learning: AI Algorithms, Types & Examples https://www.opit.com/magazine/reinforcement-learning-2/
- Michael Chen, (2023) What Is AI Model Training & Why Is It Important? https://www.oracle.com/nl/artificial-intelligence/ai-model-training/
- Geeksforgeeks, (2024) Supervised and Unsupervised Learning https://www.geeksforgeeks.org/supervised-unsupervised-learning/
- DigitalSreeni, (2022) Identifying anomaly images using convolutional autoencoders. https://www.youtube.com/watch?v=q_tpFGHiRgg&t=1785s&ab_channel=DigitalSreeni
- Taurun, (2024) How to handle overfitting in TensorFlow models? https://www.geeksforgeeks.org/how-to-handle-overfitting-in-tensorflow-models/
- Ajiboye Abayomi Adewole, (2023) Understanding the class of losses in TensorFlow for Deep Learning https://medium.com/@abayomiajiboye46111/understanding-the-class-of-losses-in-tensorflow-for-deep-learning-4bd71c8e52a7
- Rohit kindu, (2022) F1 Score in Machine Learning: Intro & Calculation https://www.v7labs.com/blog/f1-score-guide
- Daniella, (2024) How do you evaluate a Machine Learning Model? https://en.innovatiana.com/post/how-to-evaluate-ai-models

10.0 BIJLAGEN

- Supervised Learning Model voorbeeld: https://github.com/Piotr-InforDB/SupervisedLearningModel