2023

ML-Model Advies

Advies over de ontwikkeling en implementatie van een ML-Model

AI EV-BATTERIJEN
TADRAŁA,PIOTR P.P.

O-PP-CMK

INHOUDSOPGAVE

1.0 INLEIDING

Dit document zal advies geven over het ontwikkelen van een ML-model(len) op basis van de research uit het document 'ML Model Onderzoek'. Hierin zal ik toelichten wat ik heb geleerd tijdens het onderzoek en test en advies geven over specifieke aspecten van de ontwikkeling van een ML-model voor het herkennen van defecte EV-batterijen.

2.0 CONTEXT

2.1 INLEIDING

INNER wilt door middel van een AI tool defecte ev-batterijen herkennen aan de hand van CT scans. Om een AI-model te ontwikkelen dat foto's kan scannen op defecten, moet er eerst een AI-model worden ontwikkeld en getraind. 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.

2.2 SAMENVATTING VAN HUIDIGE SITUATIE

INNER gebruikt een CT-machine voor batterijen en wil deze data inzetten in een systeem die door middel van ML suggesties geeft over de staat van de batterij.

2.3 REQUIREMENTS

- Het model moet defecte batterijen kunnen herkennen.
- De dataset van het model moet uitbreidbaar zijn met nieuw scans.
- Het model moet zijn kennis uitbreiden met elke nieuw dataset die toegevoegd wordt.
- Het model moet ingebouwd worden in een API anroepbaar layer.

3.0 ONTWIKKELING

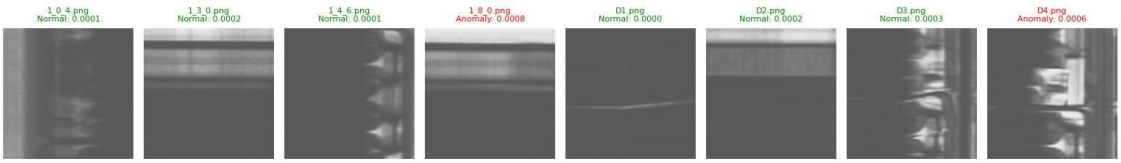
3.1 DATA

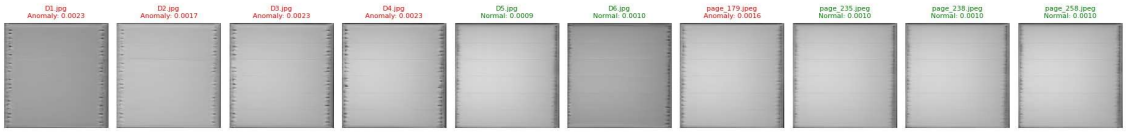
- Pas data-augmentatie toe om de variatie in de trainingsdata te vergroten.

- Een hogere resolutie van de trainingsdata leidt tot een hogere accuracy. Door een gebrek aan trainingsdata kon ik echter niet vaststellen op welk punt de resolutie-accuracy-ratio stopt met stijgen.
- Als het model defecte connectors, case, cellen, etc moet herkennen, is kan het nuttig zijn om eerst individuele componenten te classificeren met een classificatiemodel. Vervolgens kunnen defecten specifiek per component worden gedetecteerd. Door de beperkte hoeveelheid data leverde deze aanpak echter geen goede resultaten op.

3.2 ARCHITECTUUR

- Gebruik Convolutionele neurale netwerken (CNN) voor beeldherkenning.
- Gebruik Anomaly detection voor asymetrische datasets waarbij reconstructie errors worden gebruikt als indicator voor defecte.
- In verband met relatief weining details in de datasets, hebben diepere CNN layers weining toegevoegd waarde.
- Dropout layers kunnen handig zijn om overfitting te voorkomen.





3.3 TRAINING EN VALIDATIE

- Monitor overfitting en gebruik technieken zoals dropout en early stopping om die te voorkomen

4.0 INTERGRATIE

4.1 STACK

De applicatie zal worden ontwikkeld in Python. Deze keuze is gemaakt omdat Python momenteel de meest gebruikte programmeertaal is voor machine learning. Voor het machine learning-gedeelte zal de TensorFlow library worden gebruikt. Ik heb gekozen voor TensorFlow in plaats van bijvoorbeeld PyTorch, omdat TensorFlow een stuk meer gebruiksvriendelijker is voor beginners.

4.2 FUNCTIONALITEITEN

PREDICTION

Het model moet kunnen predicten of een afbeelding een anomalie is in vergelijking met de dataset waarop het is getraind.

CONTINUOUS IMPROVEMENT

Het model moet zijn nauwkeurigheid kunnen verbeteren op basis van nieuwe data die vanuit de backend wordt aangeleverd. Na elke retraining moet het model nieuwe threshold berekenen.

MODEL CHECKPOINTS

Na elke retraining moet het model een checkpoint maken. Als het model plotseling nauwkeurigheid verliest, moet er een mogelijkheid zijn om terug te keren naar een eerder checkpoint.

DOCKER

De volledige applicatie moet worden dockerized, zodat deze in een Docker Compose-omgeving kan samenwerken met andere applicaties.

TESTS

De applicatie moet unit-testbaar zijn.

4.3 ENDPOINTS

Voor de applicatie worden in eerste instantie twee API-endpoints aangemaakt. Indien er in de toekomst behoefte ontstaat aan extra functionaliteiten zoals het terugdraaien van checkpoints, kan de applicatie hiermee worden uitgebreid.

Path	Data	Response	Omschrijving

/predict	{
	“image_url”: “/”
	}
		{
		“reconstruction_error”: 0.001,
		“is_anomaly”: true
		}
			Neemt een afbeelding URL als parameter en laat het model voorspellen of de afbeelding een anomalie bevat.

/retrain	{
	“image_url”: “/”
	}
		{
		“new_validation_error”: 0.001,
		“new_anomaly_error”: 0.001,
		}
			Neemt een afbeelding URL als parameter en traint het model met de nieuwe data.

/anomaly	{
	“image_url”: “/”
	}
		{
		“new_validation_error”: 0.001,
		“new_anomaly_error”: 0.001,
		}
			Neem een afbeeld URL als parameter en evalueert de nieuwe reconstruction error.

5.0 CONCLUSIE

De voorgestelde aanpak biedt een potentiële oplossing voor het detecteren van defecten in EV-batterijen door middel van machine learning. Het model maakt gebruik van anomaly detection en CNN’s, en kan zijn prestaties verbeteren door het toevoegen van nieuwe data en het hertrainen van het model. Modelcheckpoints zorgen voor terugvalmogelijkheden bij prestatieverlies, terwijl de toepassing van data-augmentatie en dropout-layers de effecten van beperkte datasets minimaliseert. Door de integratie van API’s en Docker wordt de oplossing schaalbaar en onderhoudbaar, wat bijdraagt aan de continue optimalisatie van het model.