Onderzoeksverslag – Mogelijkheden ontwikkelen AI met NXP tools

in opdracht van EAS Datalab HR

Yered Scheffer; Piet Poot

Abstract — In dit onderzoek is de eIQ Toolkit van NXP geëvalueerd met als doel te bepalen in hoeverre deze geschikt is voor het ontwikkelen van machine learning-toepassingen op embedded microcontrollers. De focus lag op het genereren van datasets, het optimaliseren en trainen van modellen, de integratie met de MCUXpresso IDE, en de bruikbaarheid van de bijgeleverde documentatie en voorbeeldprojecten. De resultaten bieden inzicht in de toepasbaarheid van eIQ voor embedded AI en geven een overzicht van de ontwikkelervaring binnen de NXP-omgeving.

[1] NXP Logo

I. Inleiding

In dit onderzoek is geëxperimenteerd met de **eIQ Toolkit** van NXP, ontwikkeld voor gebruik met de **FRDM-MCXN947**-ontwikkelplaat. Het doel was om te onderzoeken in hoeverre deze software geschikt is voor het ontwikkelen van een AI-toepassing op dit specifieke bord. In deze rapportage worden de verschillende onderzochte aspecten, waaronder de ontwikkeling, training en integratie van AI-modellen, stapsgewijs toegelicht.

II. Basiswerking software

De benodigde software is eenvoudig te downloaden via de website van NXP. Volgens de beschikbare documentatie biedt de eIQ Toolkit functionaliteit voor het trainen van AI-modellen, het genereren van datasets, het optimaliseren van modellen, en het direct implementeren van deze modellen op embedded microcontrollers.

De basisfunctionaliteit van de toolkit is primair gericht op beeldherkenning. Omdat dit onderzoek zich richtte op een meer geavanceerde vorm van AI-implementatie, is verder gekeken naar de mogelijkheden binnen de Time Series Studio, een geïntegreerd onderdeel van de eIQ Toolkit. Deze module stelt ontwikkelaars in staat om complexere AI-modellen te bouwen op basis van tijdreeksdata, wat relevant is voor toepassingen zoals sensorgebaseerde voorspellingen of regelstrategieën. Om die reden is het vervolg van dit onderzoek specifiek gericht op deze module.

De Time Series Studio bevat eigen documentatie die uitsluitend beschikbaar is binnen de tool zelf. Deze documentatie is niet extern toegankelijk via internet, wat de toegankelijkheid voor voorbereiding en oriëntatie enigszins beperkt..

III. Genereren dataset

Binnen de Time Series Studio is het mogelijk om datasets te genereren door een extern systeem via UART te koppelen aan de software. In dit onderzoek is gebruikgemaakt van een opstelling bestaande uit een pingpongbal in een verticale buis, waarbij de positie van de bal wordt gereguleerd met een ventilator en gemeten met een ultrasoonsensor. Deze opstelling werd ingezet om de werking van de toolkit te testen. Hoewel de verbinding met de software succesvol tot stand werd gebracht, bleek de gegenereerde data niet bruikbaar. Naar aanleiding hiervan is contact opgenomen met NXP, maar tot op heden is hierop geen reactie ontvangen.

Voor het genereren van een dataset moet eerst worden vastgesteld welke variabelen dienen als input en welke als output van het AI-model. In de Time Series Studio worden deze respectievelijk aangeduid als channels (inputs) en targets (outputs).

Hoewel het precieze gebruik van channels en targets licht kan verschillen per modeltype, geldt in grote lijnen de volgende interpretatie:

- Channels: vertegenwoordigen de verschillende soorten invoerwaarden of meerdere instanties van dezelfde invoer in verschillende omstandigheden.
- Targets: geven de verwachte outputwaarden weer, die het model tijdens de training leert te voorspellen.

Bij het opstellen van de dataset is het belangrijk om ook de targets op te nemen in de channels-sectie, omdat deze bij de meeste modeltypen als onderdeel van de invoer worden beschouwd. Dit betekent in de praktijk:
Channels = Inputs + Targets

Naast het direct genereren van datasets via een live UART-verbinding, biedt Time Series Studio ook de mogelijkheid om datasets in CSV-formaat te importeren. In dit onderzoek is gekozen voor deze methode. De dataset werd extern opgesteld, gevalideerd en vervolgens succesvol geïmporteerd in de tool. Deze aanpak bleek wél effectief en heeft bruikbare resultaten opgeleverd. Om die reden is deze werkwijze als uitgangspunt genomen voor het verdere verloop van het onderzoek.

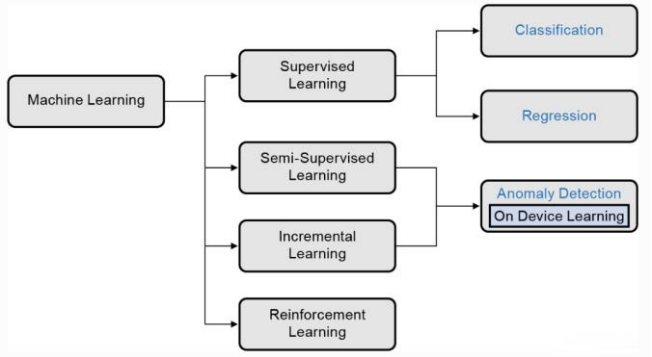
Na selectie van een dataset wordt deze automatisch gesplitst in een verhouding van 80/20%, waarbij 80% van de gegevens wordt gebruikt voor training en 20% voor validatie en testen.

Tot slot is het van essentieel belang om de juiste documentatie te raadplegen bij het opstellen van een dataset. Met name het hoofdstuk “Time Series Data” in de bijgeleverde handleiding biedt cruciale informatie over de vereiste structuur van datasets per modeltype. Hierin wordt per model uitgelegd welk formaat van de invoerdata verwacht wordt en hoe deze correct moet worden opgebouwd..

IV. Model Training

Voordat een dataset gebruikt kan worden voor het trainen van een AI-model binnen Time Series Studio, dient de gebruiker een aantal parameters in te stellen en het type model te kiezen. De tool biedt drie hoofdtypen modellen:

- **Anomaly Detection:** Gericht op het identificeren van datapunten die significant afwijken van het normale patroon. Voor het trainen van een dergelijk model is een dataset vereist die zowel normale als afwijkende gegevens bevat. In tegenstelling tot classificatie is anomaliedetectie vaak gebaseerd op unsupervised learning, waarbij volledige labeling van data niet noodzakelijk is.
- **Classification:** Hierbij worden gegevens ingedeeld in vooraf gedefinieerde categorieën. Voor classificatie zijn minstens twee klassen nodig. Het model wordt getraind op een gelabelde dataset waarin elk datapunt behoort tot een specifieke categorie.
- **Regression:** Regressiemodellen voorspellen continue waarden op basis van één of meerdere invoervariabelen. Dit modeltype is geschikt voor het modelleren van relaties en het doen van voorspellingen op basis van numerieke data.



[2] Time Series Studio Documentation

A. Projectconfiguratie

Na het kiezen van het modeltype wordt er een nieuw project aangemaakt in Time Series Studio. Hierbij moet de gebruiker:

- Het gewenste modeltype selecteren.
- De doelsysteemhardware kiezen (in dit onderzoek: FRDM-MCXN947).
- Het aantal channels (inputs + targets) en targets (outputs) opgeven.

Voor dit onderzoek is gekozen voor een project met 5 channels en 1 target. De bijbehorende dataset bevat 4 invoervariabelen (bijv. setpoint, gemeten hoogte, vorige PWM, etc.) en 1 target die het gedrag van een PID-regelaar simuleert..

Na het aanmaken van het project zijn vier hoofdonderdelen beschikbaar:

1. **Dataset**
2. **Training**
3. **Emulation**
4. **Deployment**

In dit hoofdstuk wordt de focus gelegd op de eerste twee onderdelen.

B. Datasetbeheer

De eerste stap is het uploaden van de dataset. Na het uploaden moet de structuur van de dataset worden aangegeven:

- Welke kolommen behoren tot de channels.
- Welke kolom fungeert als target.
- Hoe de data verdeeld is over de rijen (tijdsvolgorde).

De tool biedt daarnaast een visuele validatie van de dataset, waarmee gecontroleerd kan worden of de data correct is ingelezen en logisch verdeeld is.

[2] Time Series Studio Documentation

Task Type	Model Name	Full Name	Python Code Source	Python Code License	Deployment C Code Source
Anomaly Detection (On-Device Learn)	ZSM	Z-Score Model	NXP	NXP	NXP
	IPCA	Incremental Principal Component Analysis	sickit-learn	BSD-3-Clause	NXP
	IMHL	Incremental Mahalanobis Distance Based Model	NXP	NXP	NXP
	IKM	Incremental KMeans	NXP	NXP	NXP
Anomaly Detection (No On-Device Learn)	ZSM	Z-Score Model	NXP	NXP	NXP
	OCSVM	One Class Support Vector Machine	sickit-learn	BSD-3-Clause	NXP
	GMM	Gaussian Mixture Model	sickit-learn	BSD-3-Clause	NXP
	PCA	Principal Component Analysis	sickit-learn	BSD-3-Clause	NXP
	MHL	Mahalanobis Distance Based Model	NXP	NXP	NXP
Classification	MLP	Multi Layer Perceptron	sickit-learn	BSD-3-Clause	NXP
	SVM	Support Vector Machine	sickit-learn	BSD-3-Clause	NXP
	RF	Random Forest	sickit-learn	BSD-3-Clause	NXP
	XGBoost	Extreme Gradient Boosting	XGBoost	Apache-2.0	NXP
	Catboost	Categorical Boosting	CatBoost	Apache-2.0	NXP
	LGBM	Light Gradient Boosting Machine	LightGBM	MIT	NXP
Regression	RFR	Random Forest Regressor	sickit-learn	BSD-3-Clause	NXP
	MLP	Multi Layer Perceptron	sickit-learn	BSD-3-Clause	NXP
	XGBoost	Extreme Gradient Boosting	XGBoost	Apache-2.0	NXP
	LGBM	Light Gradient Boosting Machine	LightGBM	MIT	NXP
	CatBoost	Categorical Boosting	CatBoost	Apache-2.0	NXP
	RIDGE	Ridge regression	sickit-learn	BSD-3-Clause	NXP

[2] Time Series Studio Documentation

C. *Modeltraining*

Zodra de dataset correct is geconfigureerd, kan het model getraind worden. Bij het trainen van een regressiemodel worden de volgende algoritmen standaard ingezet:

- RFR (Random Forest Regressor)
- MLP (Multilayer Perceptron)
- XGBoost (Extreme Gradient Boosting)
- LGBM (Light Gradient Boosting Machine)
- Ridge Regression

Tijdens de training wordt de dataset automatisch gesplitst in een 80/20-verhouding: 80% van de data wordt gebruikt voor training, en 20% voor validatie. In het Training-scherm van de tool zijn vervolgens de resultaten zichtbaar van elk model op basis van deze 20%.

[2] Time Series Studio Documentation

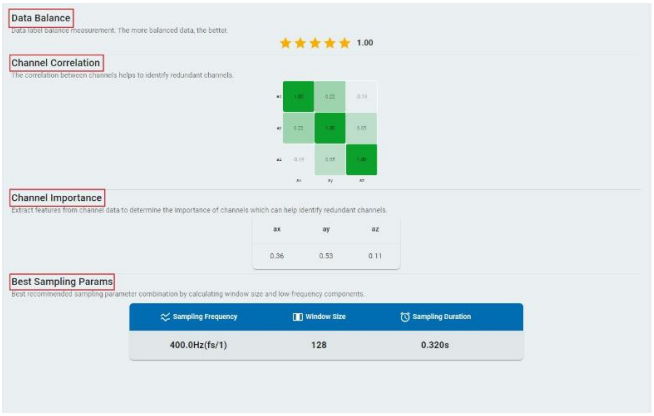
De belangrijkste onderdelen van het trainingsproces zijn:

- Training: Toont de prestaties van elk model, inclusief verbruik van RAM en Flash-geheugen, en een grafische vergelijking van de nauwkeurigheid.
- Benchmark: Geeft inzicht in de benodigde tijd voor training, evenals het geheugengebruik op de doelsysteemhardware.
- Validation Results: Bevat de resultaten van de validatiedata; idealiter ligt de voorspelling zo dicht mogelijk bij de werkelijke (target)waarden.
- Evaluation Metrics: Geeft aanvullende prestatie-indicatoren (zoals RMSE, MAE, R^2-score), afhankelijk van het gekozen modeltype.

Op basis van deze resultaten wordt het best presterende model geselecteerd voor verdere stappen binnen de tool, zoals emulatie en deployment.

V. Model optimalisatie

Om het model goed te kunnen trainen, is het belangrijk dat de data van goede kwaliteit is. In dit project is hiervoor de *Data Intelligence*-functie van Time Series Studio gebruikt. Dit hulpmiddel controleert automatisch of de dataset geschikt is voor training en geeft suggesties voor verbeteringen.



[2] Time Series Studio Documentation

Tijdens deze analyse zijn de volgende onderdelen bekeken:

A. *Balans van de data*

Er is gecontroleerd of alle klassen ongeveer evenveel trainingsdata hebben. Een goede balans is belangrijk, zodat het model niet te veel leert van één bepaalde klasse. In dit geval was de data goed verdeeld over alle klassen.

B. *Samenhang tussen kanalen:*

Er is gekeken of de verschillende datakanalen veel op elkaar lijken (correlatie). Als kanalen sterk op elkaar lijken, kan dat onnodige herhaling zijn. In deze dataset waren de kanalen voldoende verschillend.

C. *Belang van de kanalen*

Voor elke meetwaarde is bepaald hoe belangrijk deze is voor het herkennen van de klassen. Minder belangrijke kanalen zouden eventueel weggelaten kunnen worden om het model eenvoudiger te maken.

D. *Advies voor de meetinstellingen*

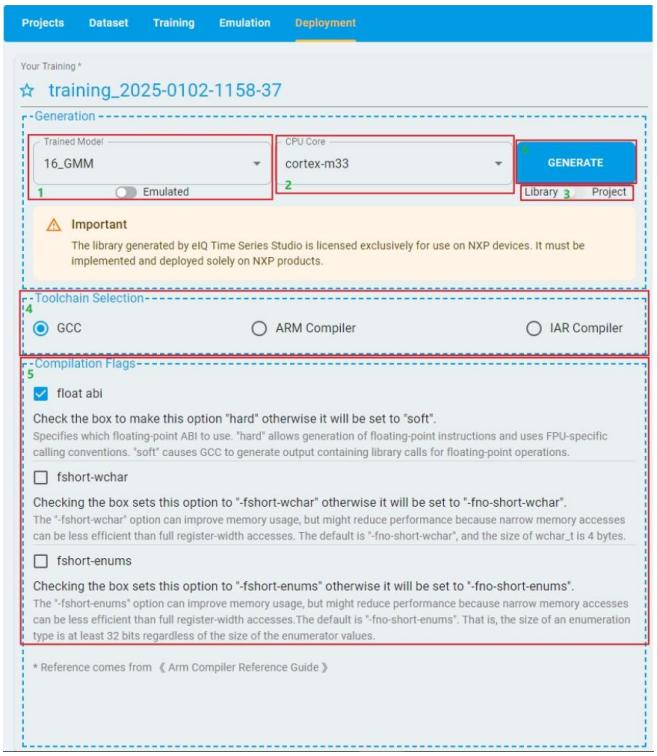
Er is een advies gegeven voor de beste meetfrequentie en venstergrootte (window size). Door bijvoorbeeld de meetfrequentie iets te verlagen, worden onnodige details (zoals ruis) weggefilterd. Hierdoor wordt het trainen sneller en blijft de belangrijke informatie behouden.

De aangepaste dataset is daarna opgeslagen en gebruikt om het model verder te trainen. Door deze automatische controle is de kans op fouten kleiner en verloopt het optimaliseren van het model sneller.

Deze functie binnen de toolkit was zeer bruikbaar en werkte probleemloos.

VI. Implementatie

Na het trainen en testen van het model is het model geïmplementeerd met behulp van de *Deployment*-functionaliteit binnen de Time Series Studio. Deze functie maakt het mogelijk om het getrainde model om te zetten naar een kant-en-klare algoritme-bibliotheek die geschikt is voor embedded toepassingen.



[2] Time Series Studio Documentation

Binnen de deployment-module hebben we eerst ons beste model geselecteerd dat eerder tijdens de emulatie was gevalideerd. Vervolgens is gekozen voor het genereren van een library file (libtss), in plaats van een volledig project. Deze library bevat alle benodigde algoritmes in gecompileerde vorm, inclusief:

- de bibliotheek zelf (*libtss.a*),
- het headerbestand (*TimeSeries.h*) voor integratie in de software,
- metadata met informatie over het model en de hardware,
- en een licentiebestand.

Het grote voordeel van deze aanpak is dat de gegenereerde library eenvoudig geïntegreerd kan worden in de bestaande AI-opstelling. Door de standaard gegenereerde interface kunnen ontwikkelaars de bibliotheek eenvoudig koppelen aan hun eigen applicatiecode.
Voor dit onderzoek is alleen het proces tot en met het genereren van de library file van belang. De verdere implementatie op de daadwerkelijke hardware-opstelling valt buiten de scope van dit onderzoek.
Deze deployment-functionaliteit binnen Time Series Studio werkte erg goed en heeft het implementatieproces sterk vereenvoudigd.

VII. Conclusie

In dit onderzoek is de eIQ Time Series Studio van NXP succesvol onderzocht en toegepast voor het ontwikkelen van een embedded AI-toepassing. Ondanks enkele beperkingen bij het direct verzamelen van data via UART, bood de mogelijkheid om externe datasets te importeren een effectieve oplossing om het onderzoek voort te zetten.

De tools binnen Time Series Studio, zoals de *Data Intelligence*-functie, bleken zeer bruikbaar voor het analyseren en optimaliseren van de dataset. Hierdoor kon het model goed getraind worden op kwalitatieve data. Daarnaast heeft de ingebouwde *Deployment*-functionaliteit het mogelijk gemaakt om zonder veel handmatige stappen een kant-en-klare library file (libtss) te genereren. Deze library is direct inzetbaar voor verdere implementatie in embedded hardware.

Al met al biedt de eIQ-toolkit, en in het bijzonder de Time Series Studio, een gebruiksvriendelijke en goed gedocumenteerde omgeving voor het ontwikkelen van AI-modellen op embedded systemen. De combinatie van data-analyse, modeltraining en eenvoudige deployment maakt het een geschikte oplossing voor vergelijkbare AI-projecten binnen embedded toepassingen.

Referentielijst

[1] "NXP Semiconductors | Automotive, Security, IoT," *Nxp.com*, 2019. https://www.nxp.com/

[2] NXP, "Time Series Studio Documentatie