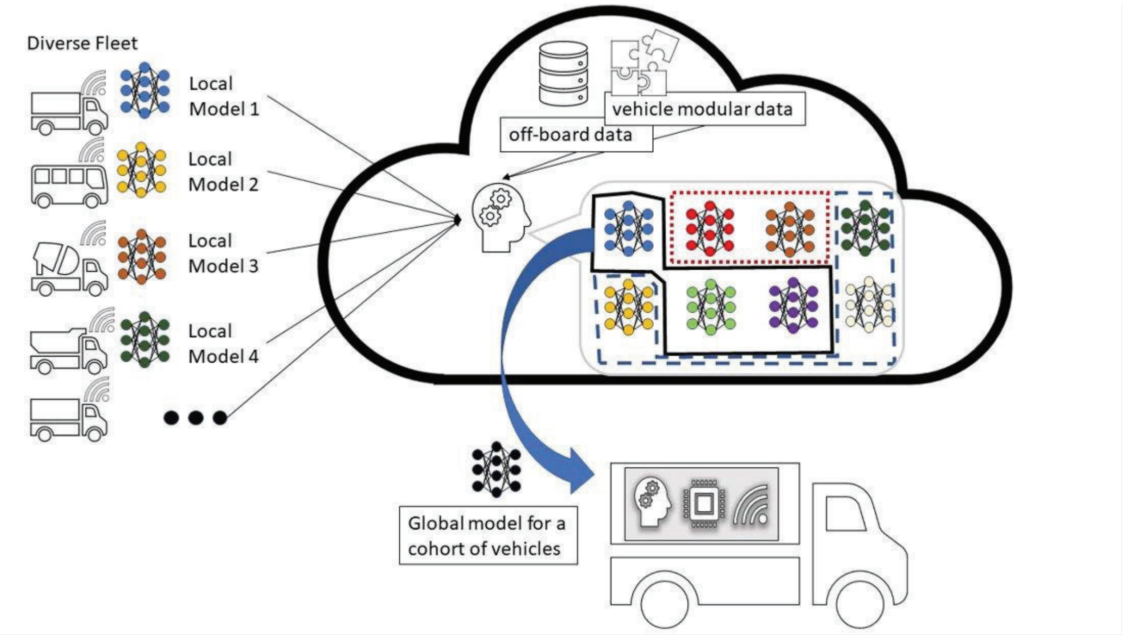
FAMOUS Final Report

Federated Anomaly Modelling and Orchestration for modular Systems

Public report



Project within Electronics, Software and Communications - FFI

Authors Juan Carlos Andresen, Erik Frisk, Andrea Magnago, Anders Vesterberg, Mattias Krysander, Göran Appelquist, Sophia Zhang Pettersson, Olov Holmer, Mahshid Helali Moghadam, Olof Steinert, Kuo-Yun Liang

Date 25-03-2024

FFI Fordonsstrategisk Forskning och Innovation
VINNOVA Energimyndigheten TRAFIKVERKET FKG VOLVO SCANIA VOLVO

Content

1. Summary

The project objective is to investigate the possibility to apply federated learning for anomaly detection using multivariate sensor data from the vehicles. This includes to propose a protocol for federated learning that deals with the intermittency of the vehicle and take the modularity of vehicles into consideration. To achieve this, we build an edge analytics prototype using the Crosser platform for orchestration and integrate a framework with federated learning support. We develop also new types of survival models for continuous distributions.

In this project we developed a way to stream flexibly CAN time-series, orchestrate and deploy federated machine learning models for anomaly detection that use CAN time-series data for training and for inference. The IoT edge analytics prototype can be installed in and used by test trucks. One of the developed federated anomaly detection models, namely the Hierarchical Gaussian Mixture model, can take advantage from a modular system. Temporal Convolutional Neural networks were implemented in a federated learning fashion to leverage the advantages and disadvantages between lightweight statistical learning and neural network-based anomaly detection models. We performed investigations on convergence for asynchronous federated learning, that show that the ratio of the buffer k and client number c is crucial to control for ensuring global model convergence. Moreover, modularity on survival models for vehicle components was investigated, showing that specialised models are not always of advantage in comparison to more general models, due mostly to the amount of data to train the models. On the other hand, we have introduced new type survival models which outperform existing models, at the cost of higher computational demand. We have made publicly available the implementation this new type of survival models.

Finally, we performed use-cases to test the advantages of an edge device and the ability to streaming in a flexible way high frequency data for fault prediction models.

2. Sammanfattning på svenska

Projektets mål är att undersöka möjligheterna att använda federerad maskininlärning för avvikelsedetektering med hjälp av sensordata från fordon. Det inkluderar att föreslå ett protokoll för federerad inlärning som hanterar asynkron och oregelbunden trådlös tillgång till internet och som tar hänsyn till fordonens modulära uppbyggnad. För att uppnå det har vi byggt en prototyp baserad på programvara från Crosser som integrerats med ett ramverk för federerad inlärning. Prototypen körs på kantnod (dvs i fordonet) och orkestreras på distans med webbgränssnitt. Prototypen kan installeras i testbilar och modellerna tränas med fordonets CAN-data. Avvikelsedetektering sker också med CAN-data. Prototypen kan också på ett flexibelt och konfigurerbart sätt strömma tidsseriedata från fordonets CAN-buss till databas i AWS.

Vi har jämfört flera algoritmer för federerad avvikelsedetektering och även utvecklat helt nya, tex Hierarchial Gaussian Mixture, som drar fördel av fordon byggda på ett modulärt sätt. Temporal Convolutional Neural Network är ett exempel på en existerande algoritm som använts i jämförelser. Hänsyn har tagits till de begränsade beräknings- och minnesresurser som är tillgängliga i fordonet.

Vi har utfört undersökningar angående konvergens för asynkront federerat lärande, som visar att förhållandet mellan bufferten k och klientnummer c är avgörande för att säkerställa global konvergens.

Vidare har modularitetaspekten för överlevnadsmodeller undersökts, vilket visade att specialiserade modeller inte alltid är fördelaktiga i jämförelse med mer generella modeller, främst beroende på mängden data för att träna modellerna. Vi har introducerat nya typer av överlevnadsmodeller som överträffar befintliga modeller, till priset av högre beräkningsefterfrågan. Vi har gjort implementeringen av denna nya typ av överlevnadsmodell publikt tillgänglig.

Slutligen har olika användningsfall testats för att finna fördelar och nackdelar för maskininlärningsmodeller på kantnod och möjligheterna på ett flexibelt sätt strömma högfrekvent data.

3. Background

The overreaching goal of the FAMOUS project is to continue with the efforts of the previous Vinnova funded project LOBSTR (Scania CV AB, 2020) to improve Scania's capability to detect and handle vehicle faults and CODA (Scania CV AB, 2021) , which had the focus on predictive models based on off-board data and valuable insights on how to include off-board fleet data into the predictive models has been gained from this project. This project builds upon the results of the two aforementioned projects. From these, the need for a scalable vehicle edge analytics device and a framework for enabling scheduled federated learning, model deployment and on-demand CAN signal streaming was identified. Additionally, the need of combining the predictive models based on the off-board data and on-board anomaly detection models is addressed by focusing on the challenges that a connected vehicle fleet entails, such as component modularity and intermittent connectivity.

4. Purpose, research questions and method

The goals of the FAMOUS project are; the development of a federated protocol and models for fault detection with intermittent connected vehicles that guarantees convergence of the models; integrate the federated models to the underlying modular system of Scania's vehicles by developing a hierarchical clustering of vehicles based on its modular system;

and develop a scalable and flexible vehicle edge analytics solution for efficient development, testing and deployment of models as well as for streaming selected time-series sensor data signals. Figure 1 shows schematically the principle of combining federated models from a diverse fleet build upon a modular system. Key concepts in the project are federated learning and vehicle modularisation to handle the large number of variants and vehicle configurations in a learning system for fault handling and optimized maintenance. Federated and modular solutions make it possible to learn models for fault management and maintenance from the whole fleet, even though individual vehicles have unique configurations, and without sharing possibly sensitive information.

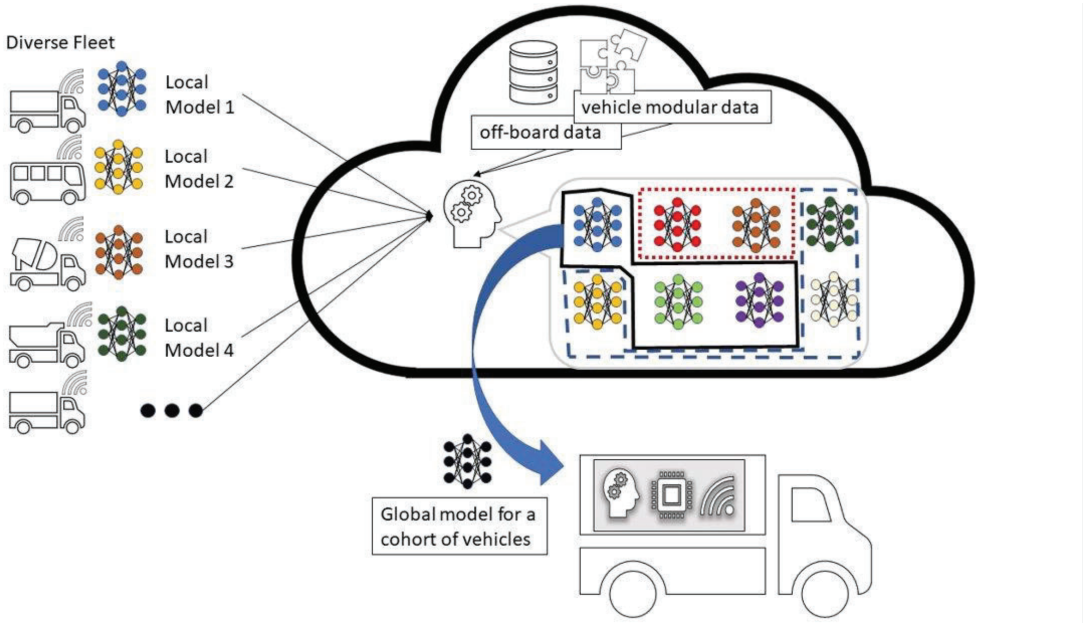


Figure 1 Schematic picture of the FAMOUS project: On the left side, different trucks represent a diverse fleet that is connected to the cloud. Within the cloud, access to the vehicle off-board data as well as information about the specific vehicle configurations are represented by the puzzle and database icons. By combining on-board data from the heterogenous fleet that is streamed through the connected systems and the off-board data of the vehicles we can find cohorts of sub-fleets that can have a common model for anomaly detection.

From the previous aforementioned projects LOBSTR (Scania CV AB, 2020) and CODA (Scania CV AB, 2021) we have gathered a set of time series data for vehicles driving in normal conditions and with injected faults and for the off-board data, we have used anonymized aggregated data. To understand how to develop models for these two kinds of data, high-resolution and low-resolution, was a key question for the project in order to take a step forward in understanding how to combine them two to get the most for developing models aimed for fault detection and predictive maintenance. To tackle these questions, especially for getting access to on-board data and deploying models on-board, new vehicle IoT infrastructure is needed. This project developed and proposed a new architecture to deal with high resolution flexible data streaming, machine learning model training on the edge, machine learning model inference on the edge and federated learning. This infrastructure was based on the Crosser Technologies platform and leveraged different approaches to federated learning.

The on-board ML models that the project focused on are the Gaussian Mixture Model (GMM) and the Temporal Convolutional Network. The GMM was developed in a fashion that modularity can be implemented in the federated learning training. The federated learning framework chosen to be integrated to the edge analytics Crosser Platform was the open source framework FEDn from Scaleout. The predictive maintenance models using off-board are network-based survival models for continuous distributions. Finally, three use-cases were done that make use of the edge analytics prototype developed in this project.

5. Objective

The main objectives of this project, as stated in the research application are:

- build a scalable vehicle edge device prototype for orchestration of federated learning, model deployment and model testing
- develop one or more federated anomaly detection methods optimized for edge computing and evaluated on injected faults and on a small test fleet with an anomaly classification method for mapping anomaly classes to known faults or to undiscovered faults
- module-based vehicle clustering methodology for federated learning models

The project was very successful, however, some minor adjustment on the main objectives had to be done. We successfully developed a scalable vehicle edge device prototype for orchestration of federated learning, model deployment and model testing. We used the Crosser platform as the backbone and adapted the open source FEDn framework. Moreover, we developed two types of lightweight anomaly detection models in a federated and modular fashion and one federated learning anomaly detection model based on neural networks. When researching on the modular-based vehicle using predictive models, with the motivation to the bridge between on-board data and off-board data and leverage the modular data stored in the cloud to include this information into the on-board federated and modular models for anomaly detection, we were confronted with some barriers. Unfortunately, the module-based vehicle clustering methodology for predictive showed to be more difficult than expected, probably due to the quality and quantity of the available data. With the studied datasets our results did not show any significant improvement on the predictions when taking into consideration the modularity compared to generic. Therefore, the focused moved from modular cohorts to the development neural network-based survival models for continuous distributions. This new focus was very successful and proposed a new family of models, namely, piecewise survival models and energy-based survival models.

6. Results and deliverables

Edge ML Prototype

For the project we developed an edge ML prototype device by combining the Welotec Edge Gateway, the Crosser Platform and FEDn framework. This prototype was tested in a truck and ML models for anomaly detection were developed tailored to this prototype.

Edge analytics framework

The initial aim of the project was to integrate the Crosser IoT node to the new telematics unit model from Scania. Unfortunately, the development of the new telematics unit was delayed. Therefore, it was decided to use an industrial IoT device that will act as the telematic unit: we used three Welotec Edge Gateways devices with 2 CAN interfaces, 4LTE cellular interface running GNU/Linux (Debian). Figure 2 shows a picture of one of the actual Welotec Gateways. The Crosser Node was configured to access the CAN interfaces of the device and had connectivity enabled to AWS via MQTT. Additionally, we used the Crosser Cloud to orchestrate and monitor the deployment of the Crosser Flows. This Edge Gateway with the Crosser Node installed could be added to a test truck and can connect to two different CAN buses.

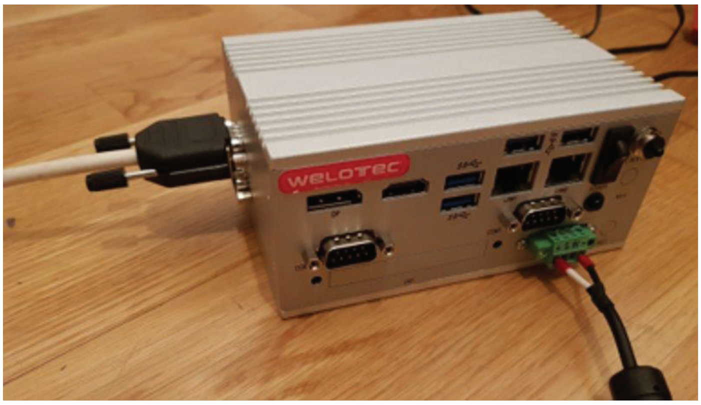


Figure 2 Picture of one Welotec Gateway with one Serial CAN connector.

Crosser

The *Crosser Platform* consists of two components, the *Crosser Node* and the *Crosser Cloud.* The Crosser Node is the real-time engine of the platform. It is usually deployed as a Docker Container. It supports edge computing, MLOps, integration with a large variety of data sources, low-code development, and version control. The Crosser Cloud is the hub where design and orchestration occur. It allows users to create, manage and monitor data flows, automation, and integrations. The Crosser Cloud can be deployed within an internal firewall. For this project, the Crosser Cloud was hosted on Crosser's infrastructure.

The way to process data within the Crosser Platform is via a Data Flow, which is a sequence of interconnected modules designed within the Crosser Flow Studio to process and manipulate data in real-time. These modules are built based upon specific requirements. During the project, Crosser developed new modules and extended existing ones tailored to the project's needs. The crosser data flows enable users to create sophisticated data processing pipelines in a modular way. If the existing modules do not fit the needs, custom python code can be used to develop user defined modules.

FEDn Framework

Scaleout Systems develop the open source framework FEDn for Federated Machine Learning. It is a framework that can be integrated into existing systems. Its focus is to enable federated learning.
During the project we adapted the open source FEDn client for our needs and deployed it using the Crosser Platform. The combination of the Crosser Platform with the FEDn framework has proven to be an easy way to create data flows in the Edge Gateway *and* federated learning of different machine learning models.

Federated Anomaly Detection Methods

Federated GMM

In this project we propose two novel solutions for federated GMM. The algorithm of federated single GMM (FSGMM), which is based on incremental learning and includes a novel approach for aggregating the local models into a single global GMM at the server level. And federated hierarchical GMM (FHGMM), which utilizes clustering of local models to form multiple global GMMs. During inference, the clients choose the global GMM that is deemed to be most suitable, given the current data distribution of each client.

Federated GMM Discussion

Overall, the results indicate that both the FSGMM and the FHGMM produce similar results as the non-federated model. More experiments with larger sets of non-IID empirical vehicle data would need to be conducted to determine whether FHGMM has a performance advantage in real scenarios. FSGMM has similar performance as the non-federated model in all tested scenarios.
The federated models use incremental learning, which means that new data is presented on every local round. The models still seem to be able to learn the patterns in the input even when each input window is quite small (e.g., 100 datapoints).
Hyperparameter optimization was not carried out in any systematic way for the federated solutions. However, the somewhat arbitrary choices of hyperparameters still resulted in acceptable performance.

Federated TCN

Several simulation experiments were conducted, with different model architectures and hyperparameters. Similar architectures were tested for both vehicle data and SMD dataset (NetManAIOps, OmniAnomaly, 2023) (besides input and output matching the number of features in respectively datasets), both prediction and autoencoder approaches were tested. For the prediction method, we predicted the next time step for all features. The autoencoder maintained the dimensions in the output. Sequence length is the length of the moving window that we used across the data to create data samples (as we do not want to feed the model the whole log file as just one data sample). Figure 3 illustrates how the process steps are for our federated setup.

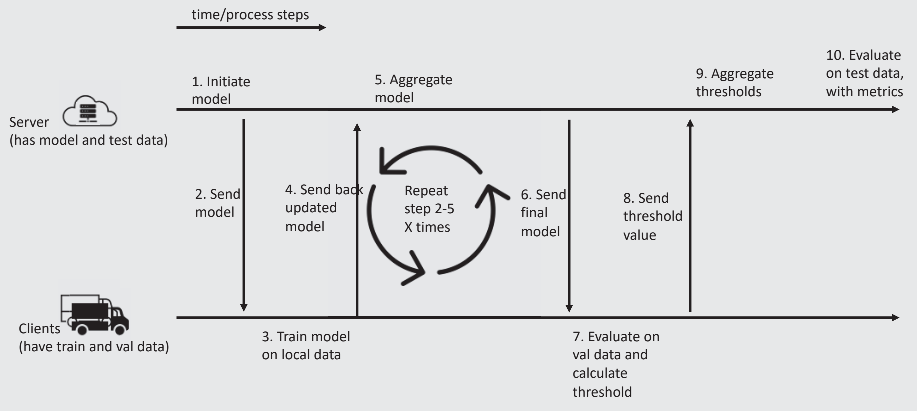

time/process steps
Server
(has model and test data)
1. Initiate model
5. Aggregate model
9. Aggregate thresholds
10. Evaluate on test data, with metrics
2. Send model
4. Send back updated model
Repeat step 2-5 X times
6. Send final model
8. Send threshold value
Clients
(have train and val data)
3. Train model on local data
7. Evaluate on val data and calculate threshold

Figure 3: Steps for federated NN learning in our setup.

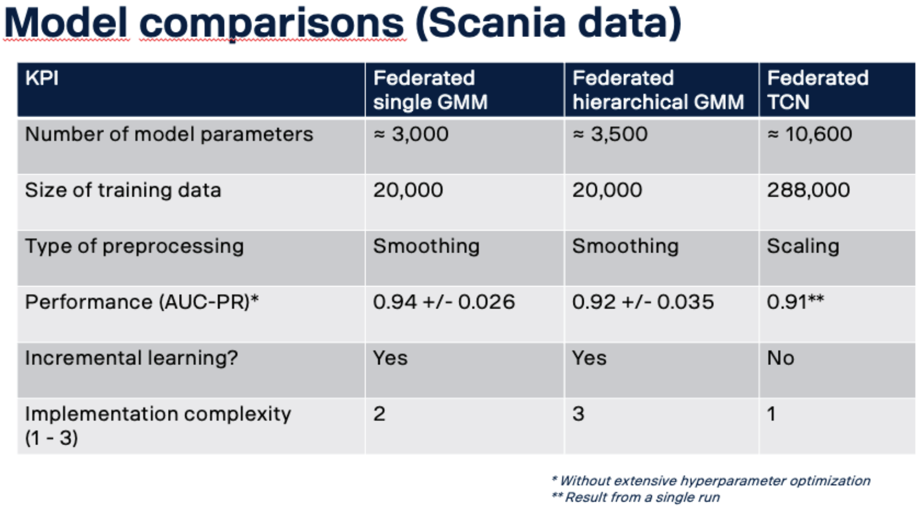
Federated TCN Discussion

We have tried anomaly detection on Scania's on-board vehicle dataset and public SMD dataset, using two different methods (prediction and autoencoder). We used temporal convolutional network (TCN) as the main building block for designing different prediction and autoencoder models. Autoencoder performed better on the on-board vehicle dataset while prediction performed better on the SMD dataset. We selected the 2-3 best performing models and did federated learning with them. As expected, they perform slightly worse (than the single vehicle case) depending on how frequent the global communications are but that is to be expected.

Comparison between Federated GMM and Federated TCN

In general, it is a difficult task to develop models for time series anomaly detection with high precision and recall. The fact that we want to avoid sending the time series data out of the vehicles adds extra complication to the task. However, we have explored and developed two alternative approaches for this, a Gaussian Mixture Model and Temporal Convolutional Networks. We did comparison on their performance to handle highly variable time series, where sequences of data points must be considered. Further, we have developed a protocol for federated learning of GMMs. For the TCN models, Fed-Avg methods were used, i.e., model aggregation was done by averaging of the weights in the neural network.

A summary of the findings can be found in Table 1. However, since the nature of the two solutions Federated GMMs and Federated TCN-AE is so different, it is hard to make a "fair" comparison, for instance, for pre-processing with Federated GMM solutions the smoothing technique is crucial important while scaling operation is more important for Federated TCN solution. The comparison is conducted based on the experiments on the Scania empirical data.



Model comparisons (Scania data)

KPI	Federated single GMM	Federated hierarchical GMM	Federated TCN
Number of model parameters	≈ 3,000	≈ 3,500	≈ 10,600
Size of training data	20,000	20,000	288,000
Type of preprocessing	Smoothing	Smoothing	Scaling
Performance (AUC-PR)*	0.94 +/- 0.026	0.92 +/- 0.035	0.91**
Incremental learning?	Yes	Yes	No
Implementation complexity (1 - 3)	2	3	1

* Without extensive hyperparameter optimization
** Result from a single run

Table 1: Comparison between three federated solutions (FSGMM, FHGMM, Federated TCN).

Asynchronous Federated Learning

Federated learning is a machine learning technique in which models are trained on the edge nodes and the trained models are later federated on the server at regular intervals. This federated model is then sent back to clients. In this way the knowledge gained on individual clients is shared across the rest of clients.

In a fully synchronous federated learning, all clients send their model updates and server waits until all the clients have submitted their updates. In this setting, the server must wait for the slowest client to submit its update, and thus for each global iteration slowest client (straggler) is a bottle neck. This slows down the training process as a result and clients cannot get updated model until all the clients have sent in their updates. In practical settings, this is usually a limitation since all the clients or vehicles in Scania's context are not available at a given time. At a given time there will always be a part of population available for synchronization. Therefore, we need an asynchronous way of federation, where we could aggregate updates from available vehicles at a given time.

An alternate way of federated learning is asynchronous federated learning, where the updates are processed when needed. Several approaches have been published to achieve asynchronous federation of clients' models. In this project, we implement an algorithm inspired from (Nguyen, 2021). In this algorithm the server waits for *k* clients before it federates the clients' updates. Weights of the federated model are then shared with the clients whose updates have been federated in the last round. Clients that are not synchronized for a long time are likely to have deviated from the federated model. Such stragglers are penalized accordingly to reduce their negative impact on the federated. The algorithm is illustrated in Figure 4.

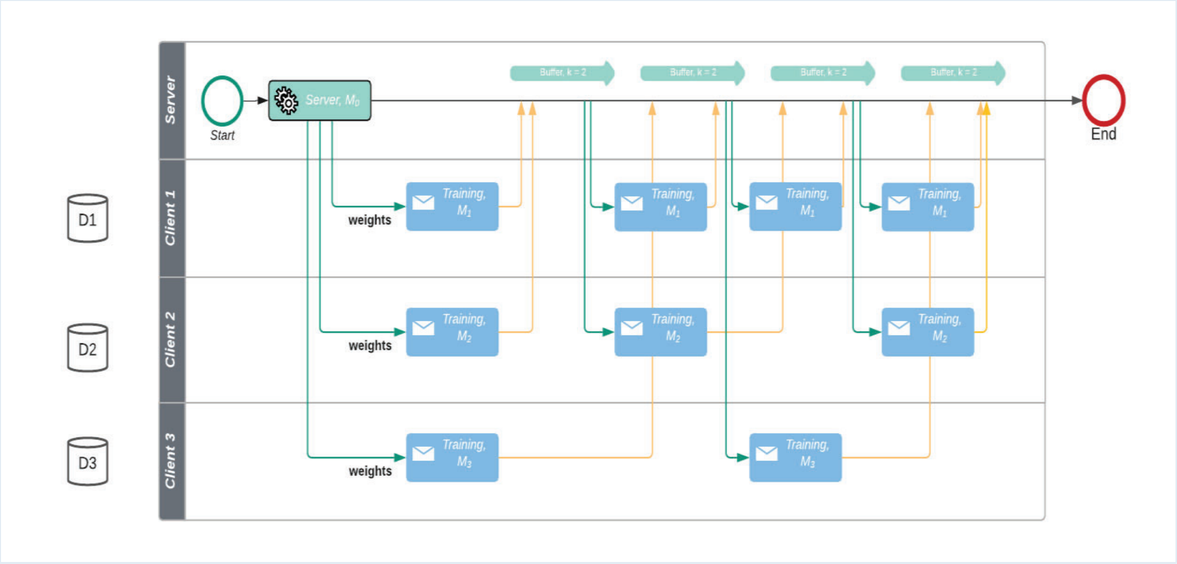


Figure 4 Illustration of the Asynchronous Algorithm

Asynchronous Federated Learning Discussion

In general, we see reasonably good convergence for asynchronous way of federated learning, if the ratio between buffer size and count of clients (k/c) is not too low. We got reasonably good convergence as long as the k/c ratio is at least 0.2. One important factor to note is that in the low k/c ratio settings, updates start queuing up and gets penalized as a result. A future improvement can be to adjust the penalizing scheme such that late arrivals due to k/c ratio do not get penalized. This will enable the algorithm to federate all the incoming updates that arrive on time (but are only delayed due to the buffer size). Another future improvement can be to replace the buffer size by a time interval and consider all the updates that arrive in each time interval. This way only those updates will be penalized which are delayed due to unavailability. The number of global iterations in every simulation also needs to be adjusted according to the k/c ratio, i.e., with very small k/c ratios, we need large number of global iterations. One way to get quick convergence is also to train the model locally for some time before federation.

Survival models for modular vehicles

The main topic of this work package has been to develop data-driven survival models for modular vehicles based on accumulative usage. That is, to predict the distribution of the remaining life of a component based on how it is used. These types of models have many uses, especially when it comes to predictive maintenance where accurate predictions regarding a components useful life are essential.

The data used in this work package consists of battery failures from a fleet of modular vehicles. The data both contains vehicles where the actual time of failure have been recorded, but also censored failure times, and is therefore classified as survival data. Due to the high complexity of the data, data-driven approaches are attractive and Neural network-based survival models have shown to be particularly effective at describing this type of data; however, research on these types of models is still in an early phase and the number of models available in the literature are still low, in particular there is a lack of expressive continuous time models. One of the main topics of this work package has therefore been to develop neural network-based survival models for continuous distributions. Two contributions have been made on this topic: In [(Holmer O. F., 2024)] a family of models called piecewise models survival models are presented; in these models a survival function is parameterized based on piecewise polynomials which has shown to give a good trade-off between computational demand and accuracy. In (Olov Holmer, 2023) Energy based survival models are presented; in this model a neural network directly specifies the failure density leading to a high expressible model which has been shown to outperform existing methods, but at the cost of slightly higher computational demand. A python package called *PySaRe – a Python package for Survival analysis and Reliability engineering* that implements these models, as well as some other useful tools, has also been developed and is currently being used in the Vinnova project (Scania CV AB, 2021), both at Linköping University and Scania CV. Parts of the package have also already been made publicly available (Holmer O. , 2023), and a full release is being prepared.

The explanatory variables in the dataset consist of accumulative usage gathered during the vehicle's lifetimes at specific times, called snapshots. The snapshots are sporadically and unevenly spaced over time and the number of snapshots from each vehicles varies greatly. The fact that the dataset contains more than one snapshot from each vehicle means that multiple predictions can be done, one for each snapshot, and since predictions from the same vehicle clearly are not independent maximum likelihood training is not directly applicable on this type of data. Maximum likelihood training is the standard way to train this type of models and therefore the main topic of [(Holmer O. K., 2024)] is to investigate how maximum likelihood-based training can be applied to this type of data. The results show that if the data is on a specific format, where the snapshot times for all vehicles are the same, maximum likelihood training can be applied and yield good results. For datasets that are not on this format, which is the case for the dataset used in this work package, it is proposed to resample the data. How this resampling is done is shown to be an important parameter; it should be chosen large enough to produce good results, but this also increases the size of the dataset which makes training slow. Therefore, to reduce the sensitivity on this parameter it is also proposed to, rather than resampling the data once before the training starts, resample the data at each epoch during training so that over time many different resampled versions of the dataset are used in the training.

The dataset also contains information about the configuration of the vehicles in the form of categorical variables specifying the vehicle type. The amount of data from each vehicle type, i.e. the number of vehicles with a specific configuration, varies from thousands of vehicles for the types with the most data all the way down to vehicle types only represented by single vehicles. How to handle this type of information during the modelling has been investigated, in particular if multiple specialized models should be developed representing specific vehicle types, or if a general model describing all vehicle types gives better performance. The results indicate that building specialized models can be beneficial for vehicle types where there is a lot of data, however, for most of the vehicle types a general model tend to perform better. It was also observed that the improvements on the model that this type of investigations could give probably is smaller than those that the other topics explored in the work package could give, and therefore those topics were prioritized.

Use Cases

Lithium-Ion Battery Capacity Estimation Use Case

We developed a machine learning model to predict the battery capacity and deployed the model in the edge analytics prototype on a bench.

Lithium-ion (Li-ion) batteries are a critical component in battery electric vehicles (BEVs) as they provide the primary source of energy for propulsion. However, Li-ion battery capacity degradation over time is a significant challenge that affects the performance and driving range of BEVs. Accurate prediction and estimation of battery capacity are of utmost importance in the context of BEVs to ensure optimal energy management, reliable operation, and enhanced user experience. To run battery capacity estimation models on-board can be beneficial as it can give direct feedback to the driver, moreover, to be able to train and deploy ML models on-board in a flexible and fast way will pave the way for fast model development and increase model accuracy. Training on-board neural networks using time series data that is not possible to stream due to the high frequency and large amount of signals opens up new possibilities for model development.

We chose to train an LSTM (long short-term memory) neural network model for this use-case. LSTM models can capture complex temporal dependencies within time series data. They maintain an internal memory cell and gating mechanisms to control information flow, which helps capture both short-term and long-term patterns. LSTMs can handle time series data with variable sequence lengths, making them adaptable to real-world scenarios where data might not be evenly spaced or may contain missing values.

LSTM Model results

The proposed LSTM model consists of an LSTM layer containing 50 units, followed by a Dense layer with 1 unit in the output layer—w.r.t using Adam (Adaptive Moment Estimation) optimizer. Table 2 presents the LSTM model's performance in the case of training separate models for battery cells as well as its performance when a unified model is trained sequentially on battery cells #1 and #2 and then tested on battery cell #3.

Table 2. LSTM model's performance

	Distinct models trained for battery cells and tested on their respective cells			**Unified model trained on battery cells #1 and #2 and tested on cell #3**
Test Battery Cell.	Cell #1	Cell #2	Cell #3	Cell #3
MSE	0.119	0.208	0.039	0.002

The LSTM model as an example model was deployed on a Welotec industrial PC and a summarized view of the performance of the model at the deployment is as follows:
Training on Welotec (with batch size 72, epoch 50)

- Handling each dataset ≈ 35 Sec.
- Training on each dataset ≈ 16 Sec.

Inference on Welotec

- Loading saved model and inference < 1 Sec.

The model was tested on the edge analytics prototype, training and inference worked as expected.

Flexible data collection using operational data use-case

To investigate the benefits of flexible data collection we extracted operational data from Scania trucks with a certain specification from the Scania Data Lake. This data was then matched with data containing failure of one specific vehicle component critical for the vehicle's operation, hence this meant that the vehicles with failure were labelled as the positive class and all other vehicles as the negative class. To test the hypothesis that flexible data collection could improve predictions, the vehicles were divided into three categories, high frequency, medium frequency and low frequency.

This data sets were then put into different machine learning algorithms that could handle multivariate timeseries data. The best method for handling this data got the following results, shown in the table below, showing the average balanced accuracy over 5 training rounds together with the variance. We choose to evaluate the models on many performance metrics but balanced accuracy which we present here is most important to give a good overview of the total performance, as the data sets are so skewed when it comes to class proportions (around 25:1, i.e., the negative class is 25 times more common than the positive class).

Data set	Balanced accuracy
High frequency	0.748 +/- 0.123
Medium freq.	–0.706 +/- 0.099
Low frequency	–0.581 +/- 0.055

Given the results we can conclude that it seems that higher frequency data can lead to an improvement in classification performance. In this study we simulated the ability of flexible data collection, and the conclusion is that having the ability to adjust data collection frequencies for example when a vehicle is not behaving normally, could lead to the creation of higher quality prediction models, than is possible with current data. Part of this data set and initial experiment was later expanded to a study[1] of how to best generate synthetic data to improve model performance when the data set has severe class imbalance.

Flexible data collection using operational data and streaming data use-case

The purpose of this work is to present a framework combining commonly used low-frequency data in the form of aggregated read-outs, with these streaming signals. To test such approach, experiments were conducted on actual operational data linked to the NOx sensor, a critical component in the vehicle's exhaust system.

Fault prediction based on snapshot data

For low-frequency aggregated data, both binary classification and survival analysis were considered for this purpose. A framework for fault prediction over time windows of variable lengths was developed. The time window is selected so we can answer the question "Will a NOx sensor fault happen in the next few days within the time window?". Snapshots within the selected time window will be labelled as the final results (repaired or non-repaired), while snapshots outside the time window will be labelled as non-repaired regardless of the final results. First, the binary classification methods incl. Random Forest and XGBoost have been applied to the preprocessed data. The evaluation metric is the precision-recall AUC. The result show that the performance improves with growing sizes of the time window when all snapshots are considered in the training data. Then, survival analysis approaches incl. Random Survival Forest and GBDT-SA have been tried and the same evaluation metric has been used to be comparable to the binary classification approaches. Results from the experiments show that performances of GBDT-SA and XGBoost are very similar. This, together with the computational limitations, justified the decision of not experimenting further with SA, which was here discarded in favor of traditional ML classification.

Data from a Test Vehicle

This subsection is concluded with a presentation of the results for the whole snapshot history of a test vehicle for different time windows with XGBoost trained on the whole available snapshot data. At the same time, real-time streaming signals from the same test

[1] https://ieeexplore.ieee.org/document/9815660

vehicle were treated using Gaussian Hidden Markov Models (GHMM) and Long Short-Term Memory Recurrent Neural Networks (LSTM) for task of anomaly detection. In this case, the relation between fault and anomaly also had to be characterised, and this was done on the base of observed results. Based on the snapshot results for the test vehicle, we set up the experiments to test GHMM and LSTM for anomaly detection.

From our results we can conclude that while it is true that LSTM showed better results than GHMM and it could, for some signals, detect the fault at a time that is comparable to the DTCs, this cannot be said for other signals that present more "noisy" patterns. More generally, the obtained results face several limitations and should be regarded as an indication that LSTM could be a good fit for anomaly detection for fault prediction.

Proposed combined approach

Experiments on the test vehicle suggested that low frequency data showed signs of a failure up to 6 months in advance, while anomalous patterns in the signals were clearly visible within weeks from the breakdown event. Based on this, the combined approach featured running multiple time-window models on a same low frequency data read-out while an anomaly detection model runs in the background in an on-line fashion. Figure 5 shows the suggested approach.

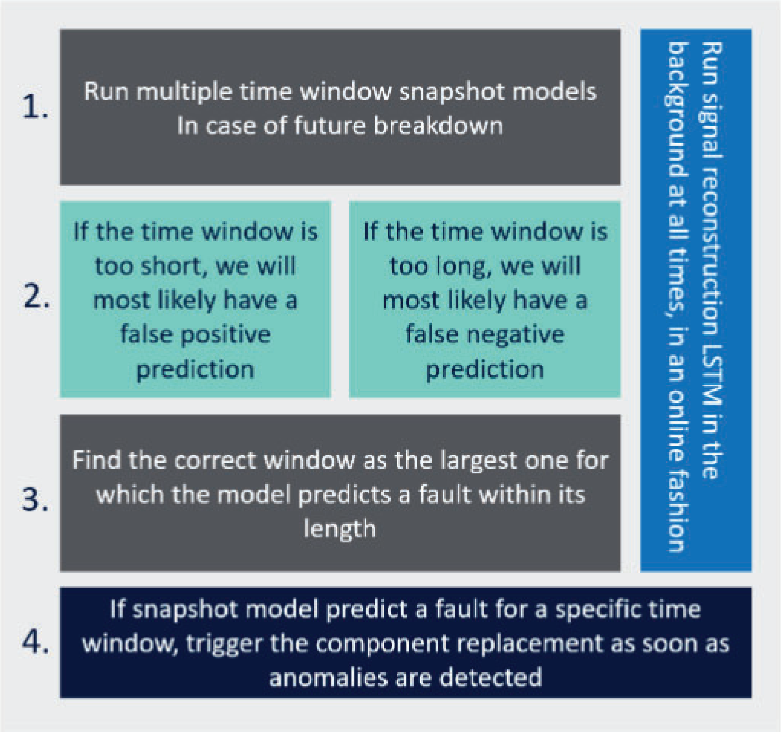

1.
Run multiple time window snapshot models In case of future breakdown
2.
If the time window is too short, we will most likely have a false positive prediction
If the time window is too long, we will most likely have a false negative prediction
3.
Find the correct window as the largest one for which the model predicts a fault within its length
Run signal reconstruction LSTM in the background at all times, in an online fashion
4.
If snapshot model predict a fault for a specific time window, trigger the component replacement as soon as anomalies are detected

Figure 5 Illustration of the suggested approach

At the present state, anomaly detection could be used as a high-resolution model for fault prediction or when the time horizon is too small for time window models. Of course, to make LSTM for anomaly detection usable, one would have to characterise anomalies in detail. For this reason, a viable alternative within the limits of this thesis could be to replace anomaly detection with the first activation of relevant DTCs.

7. Deliveries

Deliveries stated in the application

- D1: Report of federation protocol handling intermittent data sources and tested with federated models
- D2: Deploy a model on a vehicle via Crosser engine.
- D3: Demonstration of updating edge algorithms at scale (minimum 1000 simulated nodes (vehicles))
- D4: Show potential of flexible data collection.
- D5: Demonstration of re-training of ML models at the edge on the Crosser platform and exchanging updated models with a central cloud service
- D6: Published paper or report on methods for modular federated learning, applied to fleet data from Scania.
- D7: Show effect of intermittent connection on federated neural network model.
- D8: Pipeline of anomaly classification method and classification mapping to existing faults

Most of the deliveries were satisfactory fulfilled, D3 was done with less simulated nodes due to the complexity and time needed to set up such a large infrastructure. Instead, we simulated up to 20 nodes with the developed architecture. Nevertheless, Crosser Technologies showed their results for thousands of nodes but not with the algorithms developed in this project. We assume that the combination of both results, indirectly show the wanted scalability. D8 was unfortunately not feasible due to the difficulty to couple different data sources in an automated way to create a pipeline. We take the learnings on how to approach this problem for future projects.

8. Dissemination and publications

Dissemination

How are the project results planned to be used and disseminated?	Mark with X	Comment
Increase knowledge in the field	x	All partners have profited and advanced their knowledge in the field of federated learning and modularity and architecture for IoT devices for the vehicle industry.
Be passed on to other advanced technological development projects	x	The results from FAMOUS such as PySaRe and the survival models are used in the RAPIDS (Scania CV AB, 2021) project , the edge analytics prototype will be used in DELPHI (Scania CV AB, 2021) and developed anomaly detection models in IICOM

		(Scania CV AB, 2021),FAIM (RISE, 2023) and in an internal project for self-awareness for autonomous driving.
Be passed on to product development projects	x	Lessons on how to run models on edge devices, the architecture needed as well as the device requirements have been passed to pre-development projects.
Introduced on the market		
Used in investigations / regulatory / licensing / political decisions		

Publications

- Kuo-Yun Liang, et. al. Modular Federated Learning International Joint Conference on Neural Networks (IJCNN), 2022, pp. 1-8
- Olov Holmer, et al. "Energy-Based Survival Models for Predictive Maintenance", 22nd IFAC World Congress: Yokohama, Japan, July 9-14, 2023.
- Sophia Zhang Pettersson, et al, Aggregation of Generative models in Federated Learning, to be published.
- Sophia Zhang Pettersson, et al. Hierarchical Federated Gaussian Mixture Model, to be published.
- Olov Holmer, et al. "Neural Network-Based Piecewise Survival Models", to be published.
- Olov Holmer, et al. "Usage-Specific Survival Modeling Based on Operational Data and Neural Networks", to be published.
- Olov Holmer, et al. "PySaRe – a Python package for survival analysis and reliability engineering", available on GitHub and to be published on the Python Package Index

9. Conclusions and future research

During this project we have developed an Edge Device Prototype that enables Federated Machine Learning for anomaly detection using time series data collected from the CAN buses. Using the Crosser Data Flows this prototype can flexibly collect different CAN signals and stream them in the desired frequency in Snowflake using Amazon Web Services. For machine learning model deployment, the FEDn client from the FEDn framework was adapted to the Crosser Data Flows, enabling different machine learning models for anomaly detection using CAN sensor data.

Within the project we implemented different federated machine learning anomaly detection; the Hierarchical Gaussian Mixture Model and Temporal Convolutional Networks. These two models perform well for the datasets tested; depending on the specific needs one can be more advantageous than the other. Notorious differences are the size of training data needed, where the Gaussian Mixture Models are much less data hungry for training than the temporal convolutional networks. Similarly, for the number of parameters. The implementation complexity, on the contrary, is higher for the Federated Gaussian

Mixture Models than for the Temporal Convolutional Networks. In general, neural networks can be federated in an easier and more straight forward way than general statistical learning methods.

Here we also implemented and analysed different asynchronous federated learning recipes for a Neural Network, where clients that send late model updates are penalized. Important to control is the fraction of the models arriving in synchronous time (called buffer size *k*) and the total number of local models (called clients *c*), that is the *k/c* ratio is a key to control the convergency of the global federated model. We find that a ration of *k/c*>0.2 gives good convergence, independently of the number of local epochs. Of course, convergence might be slower but still, if sufficiently long trained, the global model will converge.

Using aggregated data, it was investigated if by exploiting the modular system of Scania CV AB multiple specialized survival models for components could be developed to improve accuracy against generic models. The results indicate that only in the case that the specialized models have large amount of data this approach can be beneficial. This is in general not the case, therefore, generic survival models without clustering based on the modular system is the preferred way. Therefore, the research focus shifted towards neural network-based survival models for continuous distributions. This type of neural network directly specifies the failure density leading to a high expressible model which has been shown to outperform existing methods, but at the cost of slightly higher computational demand. A python package that implements these models has been developed and made publicly available.

One use-case made use of the edge analytics prototype for testing training and deployment. The built infrastructure was scalable and robust, suitable for edge ML models for anomaly detection in a federated and modular way. Further use-cases were done during the project to explore the advantages of having access to high frequency data for fault prediction.

10. Participating parties and contact persons

Scania CV AB: Juan Carlos Andresen, juan-carlos.andresen@scania.com
Crosser Technologies AB, Andrea Magnago, andrea.magnago@crosser.io
Linköping University, Erik Frisk, erik.frisk@liu.se

11. References

Holmer, O. (2023). *PySaRe – a Python package for survival analysis and reliability engineering*. Hämtat från https://github.com/oholmer/PySaRe

Holmer, O. F. (2024). Neural Network-Based Piecewise Survival Models. *arXiv preprint arXiv:2403.18664*.

Holmer, O. K. (2024). Usage-Specific Survival Modeling Based on Operational Data and Neural Networks. *arXiv preprint arXiv:2403.18739*.

Krizhevsky, A. &. (2009). Learning multiple layers of features from tiny images.

Lässig, F. (den 26 July 2023). *Temporal Convolutional Networks and Forecasting*. Hämtat från https://unit8.com/resources/temporal-convolutional-networks-and-forecasting/

NetManAIOps, OmniAnomaly. (den 27 July 2023). Hämtat från https://github.com/NetManAIOps/OmniAnomaly/tree/master/ServerMachineDataset

Nguyen, J. M. (2021). Federated Learning with Buffered Asynchronous Aggregation. *arXiv preprint arXiv:2106.06639*. Hämtat från http://arxiv.org/abs/2106.06639

Olov Holmer, E. F. (2023). *Energy-Based Survival Models for Predictive Maintenance.* IFAC-PapersOnLine. doi:https://doi.org/10.1016/j.ifacol.2023.10.762

RISE. (2023). *Future AI-based maintenance.* Vinnova ref. No. 2023-01917.

S. Bai, J. K. (2018). An Empirical Evaluation of Generic Convolutional and Recurrent Neural Networks for Sequence Modeling. *arXiv preprint arXiv:1803.01271*. Hämtat från https://arxiv.org/pdf/1803.01271.pdf

Scania CV AB. (2020). *Learning On-Board Signals for Timely Reaction.* Vinnova ref. No. 2017-03046.

Scania CV AB. (2021). *Diagnosis by Exploiting Physical Insights in Neural Network Models.* Vinnova ref. No. 2021-05036.

Scania CV AB. (2021). *Interpretable Artificial Intelligence for Condition Monitoring.* Vinnova ref. No. 2020-05138.

Scania CV AB. (2021). *Predictive models with interpretability and analysis of drift data.* Vinnova ref. No. 2017-03046.

Scania CV AB. (2021). *RAPIDS - Pålitligt adaptivt prediktivt underhåll och intelligent beslutsstöd.* Vinnova ref. No. 2017-03046.

Tan, A. Y. (2021). Towards Personalized Federated Learning. *arXiv preprint arXiv:2103.00710*. Hämtat från https://arxiv.org/pdf/2103.00710.pdf

Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv preprint arXiv:1708.07747*.