2nd Conference on Production Systems and Logistics

Federated Machine Learning Architecture for Energy-Efficient Industrial Applications

Can Kaymakci[1,2], Lukas Baur[1], Alexander Sauer[1,2]

[1] *Fraunhofer Institute for Manufacturing Engineering and Automation IPA, Stuttgart, Germany*
[2] *University of Stuttgart, Institute for Energy Efficiency in Production (EEP), Stuttgart, Germany*

Abstract

Due to the rise of new information and communication technologies manufacturing companies have access to huge amounts of power consumption data which are measured by sensors and processed by information systems. One of the most promising applications of extracting value out of the collected data is the detection of anomalies in process data from industrial machines and equipment. Many research and industry use cases apply machine learning (ML) techniques for anomaly detection. These techniques enable manufacturing companies to optimize their manufacturing processes but also to be more energy efficient and therefore have an impact for sustainable manufacturing. Most of the ML applications use central server infrastructures for data collection from different sources to process and analyse it for further usage. Nevertheless, privacy concerns and security risks motivate manufacturers to store the collected sensitive data from the production line locally. Therefore, suppliers of industrial machines (e.g. robots, machine tools) do not have the possibility, to store and analyse the data in the cloud, where data from all the machines of the supplier in different companies could be analysed and used for ML applications. One of the new paradigm shifts in ML is the concept of federated learning (FL) which enables local devices to use ML without sending data to a central server. This paper introduces an architecture for using the concepts of FL in manufacturing processes enabling machine suppliers to use ML for optimizing machine processes in a collaborative manner. Therefore, the more general federated learning concept is extended for industrial machinery and equipment using the industrial communication framework OPC-UA. Our architecture is tested and validated by using an industrial dataset of different compressors' power consumption.

Keywords

Energy Efficiency; Federated Machine Learning; Smart Manufacturing; Anomaly Detection

1. Introduction

Due to the ongoing digitalization of the energy and manufacturing sector, enormous amounts of data and time series from industrial machines and equipment in supply and manufacturing systems are collected each second by sensors and actuators [1]. The possibilities for increasing resource and energy efficiency by using the collected energy consumption data is enormously high. Detecting irregularities or anomalies in the power consumption of industrial machining and equipment with machine learning (ML) models is one of the keys for energy-efficient manufacturing [2]. Therefore, the use of machine learning (ML) approaches enables manufacturing companies to identify unusual or inefficient behaviour in energy consumption and to find the causes of inefficiencies in industrial machining and equipment [3]. This means that anomalies which lead to higher power consumption than necessary, such as those that occur in poorly maintained, outdated or

incorrectly controlled systems, can be detected and measures can be taken [4]. Normally, the huge amount of data is stored in a data warehouse or in data lakes and is aggregated and processed for the specific use case or service [5]. Consequently, machine and operating data must be collected decentrally but managed and processed centrally. This requires a high degree of trust, as possible security gaps in a system can lead to information leaks. In the case of industrial data, potential attackers are able to tap into information about manufacturing processes or even problems at individual companies [6]. Highly sensitive process data can lead to significant competitive disadvantages and cause considerable damage [7]. Thus, the tendency of manufacturing companies is to use only local computing capacities for the collection, management and analysis of data [8]. For machine suppliers applying industrial communication standards such as OPC-UA [9] to process data and deploy ML applications, the sensitivity and purely local management of the data can be a serious problem. Data-based services such as anomaly detection by training a ML model with data from different machines are no longer possible when data is managed locally, as local datasets are insufficient for a ML model due to the lack of generalization and the small amounts of local data available. However, the concept of federated learning has been developed that allows models to be trained without sharing data. Therefore, local devices on the field level train and test ML models locally, but exchange the implicit knowledge as model parameters for the purpose of generalization and robustness of ML models for a specific use case [10]. Recent works focus on the overall architecture of a FL system for industrial applications [11]. Nevertheless, a practice-oriented architecture for training and testing ML models for a network of industrial machinery and equipment to detect anomalies in power consumption is not considered. To close this gap the following two research questions are formulated:

- Which components are needed for training and testing ML models in FL systems in industrial applications?
- How can industrial communication standards be used for privacy-preserving anomaly detection in industrial applications?

This article is structured as follows: The second section attempts to define the concepts of anomaly detection and federated learning. Additionally, the potential for manufacturing companies of using the concepts is elaborated by analysing existing literature and comparing state-of-the-art architectures. In section 3 we introduce our architectural approach for anomaly detection with decentralized data as a federated learning system that enables the use of federated learning by integrating machine learning models with the standardized industrial communication standard OPC-UA. Section 4 demonstrates the approach with a use case of anomaly detection in the power consumption of different compressors in a manufacturing company. We discuss and validate the approach with findings from the use case before we conclude in the final section 5.

2. State of the Art

Anomaly detection in energy consumption of industrial applications is mostly used for detecting inefficiencies [2]. The main challenge is detecting increases or decreases in energy consumption not related to the operation mode of industrial applications, such as machines or the infrastructure of a manufacturing system. Connected to fault detection, it is necessary to detect anomalies in energy consumption to take measures like maintenance or turn off systems [12]. To analyze the energy consumption of industrial applications a multi-level monitoring and near real-time data processing is suggested in [13]. Himeur et al. [2] classifies the different applications of anomaly detection related to energy consumption in buildings into five different categories - abnormal behavior, faulty appliances, occupancy detection, non-technical loss detection and at-home elderly monitoring. Except the last category, these applications can also be adapted to manufacturing processes and infrastructure such as building or technical systems. Most of the current

work focuses on building new algorithms or ML models for anomaly detection but do not consider the integration into a superordinate FL system.

FL systems are classified into three categories – horizontal, vertical and federated transfer learning [14]. The horizontal FL approach is a system where the same feature space is shared but the sample space is different [10]. The typical example for horizontal FL in the manufacturing industry is a device or machine which collects the same process data (e.g. voltage or current) but is used in different companies or for different applications. The first horizontal FL solution was published by McMahan et al. [10] in which a single user of an Android phone updates model parameters locally. The updated parameters are pushed to the central server where a centralized model is trained with the parameters from several data owners. Vertical FL or feature-based FL is defined as a learning method for distributed data which is different in feature space but can have the same sample space. Considering time-series data in manufacturing and their corresponding feature spaces vertical FL can be used for manufacturing systems that have the same sample space (e.g. timestamp) but different machines collecting different data for detecting anomalies of the whole manufacturing process. One of the most promising but not widely used approaches is the combination of FL and transfer learning which is called Federated Transfer Learning (FTL) [14]. The goal of transfer learning is to transfer the knowledge, which is the model in the case of machine learning, from an existing domain to a new domain. Hence, FTL applications have neither the same sample or feature space nor multiple datasets with heterogeneous configurations. One of the first applications of FTL is FedHealth [15] enabling the recognition of activities measured by wearable devices. In [16] a FTL framework for classifying different sources of electroencephalography (EEG) data from brain-computer interfaces is proposed. FTL could also be used in machine tools which are similar such as milling machines or welding equipment from different manufacturers.

For the effective use of the described FL approach in software applications Lo et al. [17] define architectural patterns which have to be considered when building an industrial FL system. The patterns are classified into four different categories – Client Management, Model Management, Model Training and Model Aggregation. The client management category consists of patterns which define the processes and components for data management, client selection and the clustering of clients by grouping devices into user groups (e.g. using a machine tool in different industries). The model management derives from the model registry in the classic machine learning lifecycle. The main part of the model registry is to store and manage the model parameters and the different versions of the model [18]. Furthermore, a model replacement trigger and a deployment sector is necessary to change models when the performance is degrading. Additionally, the model compressor is introduced as an important step before sending model parameters to the clients. The model training and aggregation category consists of a model trainer component, a registry and different aggregation components such as secure, asynchronous and decentralized patterns.

The concept of building a generic architecture of industrial FL systems is extended by Hiessl et al. [11] introducing “assets” that generate data on the shop floor during operation and FL “cohorts” grouping assets from the same asset type. Therefore, the different asset data characteristics influenced by operating and environmental conditions are evaluated. The different FL cohorts can be compared to manufacturing islands where the random noise in data such as in temperature or humidity is the same and is detectable by the model itself. Training rounds of the FL system is only done in a cohort. For machine suppliers the cohort could be also defined customer-centric where each customer, such as an original equipment manufacturer or specific plant, is a FL cohort.

However, Hiessl et al. [11] want to evaluate the architecture by using open source frameworks such as PySyft or TensorFlow Federated. These frameworks do not consider any industrial communication protocols between the different devices or assets. Open standards such as OPC-UA are not integrated into the frameworks for an easy-to-use FL application in manufacturing industry. Furthermore, the architectures and approaches presented in this chapter do not include the model and data tracking capabilities suggested in

[18]. An easy-to-use architecture for training and testing ML models in industrial FL applications for industrial equipment and machinery is still missing.

3. Concept

To overcome the limitations described above a novel industrial architecture for training and evaluating machine learning models in FL systems is introduced where model training can be accomplished without exchanging data from machines that share the same goal such as anomaly detection. The overall goal of the architecture is to describe and organize a FL system by defining the different components, their modules and relations. It can be considered as a plan for implementing a software system. Technological aspects such as programming languages are abstracted to generate a concept for different implementation strategies. The main goal of the proposed architecture is to enable machine suppliers for communicating trained ML models with the same feature space but machines in different locations for process optimization by detecting anomalies in energy data such as power, voltage or current. Therefore, the federated principles from [11,17] and the industrial communication protocol OPC-UA [9] is used. The automated data pipeline of collecting, pre-processing, training and testing is done by the local clients which are at the field level. The server coordinate the start and end of the data pipeline. Furthermore, the server is responsible for the aggregation of the model parameters that are approximated in each training round. It is important that the architecture only considers the subsection of training and testing ML model as a main part of a federated AI system for detecting anomalies in manufacturing [18].

The architecture consists of three main functional components – machine trainer, global model coordinator and a model database. We define the **machine trainer** as a functional component on a machine tool which is used in manufacturing processes generating data by sensors. The sensor data is ingested into a predefined data pipeline on the machine trainer. The main goal of the machine trainer is to pre-process data, train and validate a model and send the generated implicit knowledge defined in the model parameters back to the **global model coordinator**. Whereas the machine trainer is at the field level communicating directly to the machining tools the global model coordinator aggregates the model parameters generated by more than one machine trainer. The aggregation of the model parameters is done by using aggregation mechanisms such as proposed in [10]. The history and the different changes of model parameters are tracked in the **model database**. The model database is defined as a graph network showing the relations between model changes and the current use of different models on different clients. Hence, the global model coordinator and the machine trainers can be synchronized and the model database can be seen as the source of finding the best parameters for the next training iteration phase. To show the communication between the functional components and the physical assets the **dataflows** between them is presented as well.

a. Machine Trainer

The machine trainer is the functional component which is executed on a client with processing capabilities such as industrial computers or edge devices on machines. One of its three tasks is to communicate with the global model coordinator to receive events and notify about new model updates. Each incoming notification will either be a train or an evaluation request. For flexibility reasons the client automatically downloads all required model specifications and uses the included model compiler to create a model. The required model specification and the weights can be seen as a recipe that is used for building the model on the local machine without training or building it from scratch. After compiling, its weights are overwritten by the downloaded ones, this will uniquely recreate the stored model in the database. In the case of a train request, this model is used as a pre-trained model and further optimized. For the purpose of evaluation, clients private test data is used to calculate metric scores such as mean absolute error (MAE) for regression tasks or the accuracy for classification tasks. After the evaluation procedure, the optimized model weights and the metric scores are transferred to the model database.

b. **Global Model Coordinator**

The global model coordinator plays the major role in the communication between the machine trainers and the model database. The global model coordinator hosts an OPC-UA server and implements an interactive terminal for human-interaction with the connected devices and the network. All requests and events are send or received from or to this functional component. It is responsible for creating a base model which serves as the starting model for the first training round. A training round initiated by the global model coordinator is defined as the sum of one training and communication process on each active machine trainer. From the mentioned base model all other models are derived during the iterative training rounds. Also, the coordinator instructs all clients to train on a certain model by updating the respective model OPC-UA node and it collects all client responses to keep track on who is working on which model. Moreover, it receives the model training termination messages from each client after each training round. As soon as all termination notifications for a model are received, the merging phase is started: All respective new model weights are downloaded from the model database and a combined model is calculated by using FederatedAveraging [10]. Finally, the coordinator forwards the evaluation request from a user by triggering an evaluating event. This event starts the evaluation procedure on available machine trainers that test the local data on the merged model. Especially in the case of manufacturing systems the used machines depend on the process. Therefore, the evaluation on different clients is necessary to overcome problems of overfitting and reduce generalization error.

c. **Model Database**

The database manages four types of components which can also be seen as tables with references to each other: models, structures, model evaluation results and learning graph entities. The former two encode a model uniquely by splitting it into its model structure and model weights. Storing a precompiled model in the database might lead to problems using different client devices and operating systems installed. Therefore only the relevant information such as weight and structure is stored to compile a model from scratch. Whenever a model is needed from the model database the serialized plain description of the model is downloaded and compiled by the machine trainer as a compatible instance. The structure of the model is saved as a reference to look up its structure in the structure table. The model evaluation metrics are uploaded by the clients directly and stored for each model-client pair, since a model performs differently on different local client data. The centralized storage allows the evaluation process to work fully asynchronously. Since in each federated learning training round a model for each client and a combined model resulting from the merge process is created, the number of models grow fast. To keep track of the model evolution, a directed acyclic graph inspired by [19] for model tracking is built up and updated whenever a new model is created.

d. **Dataflow**

In this section the dataflow (Figure 1) is conceptually explained by drafting a typical learning iteration in more detail. The OPC-UA server on the model coordinator side is stated and at least one client is online. The clients register on the server's model node in their booting code. An initial model separated into its model structure, its initial model weights and its parameters of the optimizer is uniquely defined in the database by storing a foreign structure ID a foreign optimizer-entry ID and its weights. The server starts an iteration round by overwriting the model node by a database model ID. The global model coordinator automatically triggers an event to all subscripted clients. Each available client will immediately respond to the server to await for signals to train on the new global model in this round.

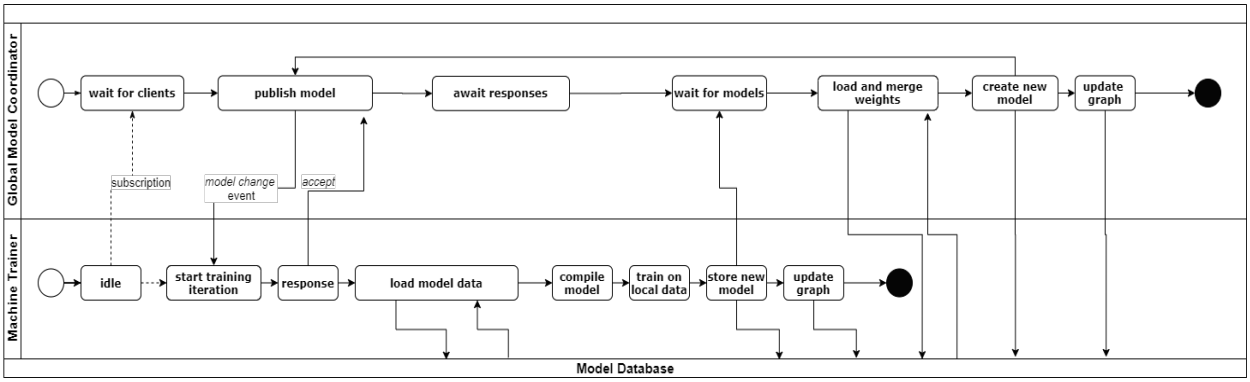

Figure 1: Typical sequence of model training and the interaction between the components

After the registration thread has finished his server response, a model loading thread is started. Given the server's global model ID, the model structure, the model optimizer parameters and the weights are loaded. A parser will compile the model based on the low-level representation. After being successfully instantiated the model, is passed on to a train worker thread performing the actual training on the local data. Once the training has finished, a new model is uploaded to the database uniquely defined by the old global model structure ID, the old model optimizer parameter ID and the new weight vectors. As a final step, the model clients update the training graph database accordingly. It completes its work by sending a response containing the new model ID to the server.

Once all active clients returned their updated model, the server will download the respective weights implicitly linked by the model IDs. A new weighting is calculated by the server using a standard weight averaging approach, similar to [5]. The new global model is stored to the database by uploading the new merged weights and referencing the structure and optimizer. Lastly, the server updates the graph database to keep track of the merging phase. Now the process will start all over again.

OPC-UA intuitively motivates the use of asynchronous working threads. We therefore decoupled the evaluation from the learning phase. We are interested in how the evaluation error evolves over time i.e., how the global model performed on unseen data on various devices and how the learning weights have been distributed during the iterations. Starting from the initial model ID, we scan the graph network for the resulting global models. Having collected all IDs, we trigger the clients asynchronously to start downloading the respective models and evaluate them. After finishing the evaluation, the clients upload their scores to the evaluation database.

4. Use Case

For the purpose of validating the functionality and the convergence of the training phase from machine learning models in a distributed way, a use case from a manufacturing company for anomaly detection is considered. In a manufacturing system different oil-injected screw compressors for further applications such as injection moulding machines are used. The compressors are procured from the same supplier. Furthermore, the compressor types are different, which can be seen in the product data sheets and the overall performance (e.g. working pressure, rotational shaft speed or nominal shaft power). The compressors are installed in a manufacturing plant and integrated into the infrastructure for other machines and systems. For simulating and validating our approach historic data of the power consumption of five compressors is divided into five separated datasets that are transferred on five different simulated edge devices which are considered as virtual compressors. In our experiments, the virtual compressors are simulated on laptop computers with different operating systems and hardware specifications but also single-board computers such as a RaspberryPi. The data is sampled in 15 minutes intervals and pre-processed locally on the edge device. Each

client splits its data into 80 percent training and 20 percent testing data und scales the data in a pre-processing step.

By integrating the developed machine trainer into the virtual compressors it is possible to collect and pre-process data but also train and test models without sending the power consumption data. Furthermore, the communication between the virtual clients and the global model coordinator which is deployed on a personal computer is tested and validated. The machine trainer is implemented on all devices by using Python 3.8 and the Python package “freeopcua” to integrate OPC-UA into the environment. The global model coordinator was hosted on one of the virtual compressors. In principal, the model database and the server could have been set up on any other virtual computer accessible in the network. By using different hardware, data of power consumption from oil-injected screw compressors, a closed machine environment and the communication between different compressors for detecting anomalies in their power consumption can be simulated.

The model for detecting anomalies is a LSTM-CNN-based forecast model which uses the predicted power consumption for detecting anomalies. Therefore, the difference between the predicted and the actual consumption is calculated [20]. If it reaches a defined threshold, the power consumption at a given timestamp can be classified as an anomaly. The training and optimization of the Huber loss function is done by the Adam optimizer. For implementing our model we used the Python packages “Keras” and “TensorFlow”. Our case study is divided into two different scenarios. Training a ML model on data from one virtual client is considered as the baseline. For each federated learning experiment, we always performed five *training iterations*. A training iteration is defined as single training on local data of the virtual compressor with a batch of size 32 and 20 learning epochs.

To compare the performances of the federated learning data privacy-preserving approach, we set up two main experiments seen in Figure 2 and 3: In the *reference experiment* we trained a ML model based on the data of a single virtual compressor which can be seen as baseline.

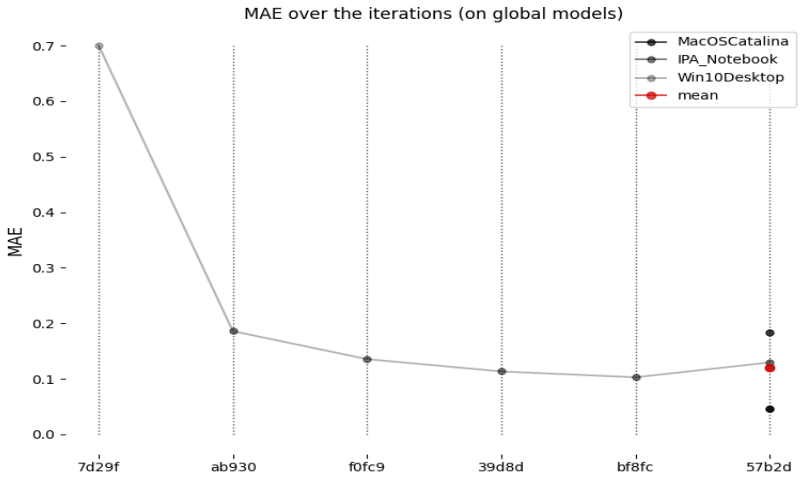

Figure 2: Progress of training in the reference experiment

This is related to a model which is only accessible within a compressor with computing capacity and does not perform any parameter exchange. To argue about its general performance, we evaluated it on both its own test data and the test data of other clients by calculating the mean absolute error (MAE). Consequently, the trained ML model is evaluated by using not only data from the same virtual compressor that the model did not see in training, but also data from different compressor types. Secondly, we performed the *federated learning experiment* using multiple clients to combine their weights after each iteration resulting in a new

global model which is used as a starting model for each client in the next iteration. In Figure 2 the decrease of the MAE of the reference experiment can be seen. The x-axis shows the ID of the model which is used for training and testing. The ID of the virtual compressors seen in Figure 2 are the names of the computing devices. fter the complete training phase, the last model is evaluated on two other virtual compressors.

In Figure 3 the federated learning experiment is shown by plotting the MAE after every training iteration and model update. For a better visualization the MAE on the y-axis is plotted in a logarithmic scale. The red line shows the average of the MAE validated on each test data and iteration. The comparison of the two resulting experiment plots show two key insights. Obviously, both experiments converge to some final value, but the convergence speed differs: The evaluation error on the models after the first iteration performs on average better that the reference training. This effect can be measured even better when the number of clients is increased further. After the first two model exchange iterations, the training error remains stable, convergence is approached. Secondly, the weight-averaging pays off in terms of variance reduction: While the MAE reduced in both experiments, the federated learning experiment reduced the overall evaluation error significantly better than the local training (final average MAE of 0.12 for reference, 0.06 for FL experiment). This is because more heterogenic power consumption data from different compressors have been implicitly taken into account, due to the model exchange. Assuming equally powerful computing capacity, a negligible small time overhead for communication and small merging times, due to the parallel training the FL approach results a variance reducing global model in a similar training time. Hence, more power consumption data can be processed and integrated into a combined model for detecting anomalies in power consumption of each compressor. Furthermore, the experiment shows that different machine types of industrial oil-injected screw compressors with the same feature space and target values (detecting anomalies in power consumption) can be integrated into the proposed architecture. The effort for machine suppliers and manufacturers to train and test ML models on different machines decreases significantly.

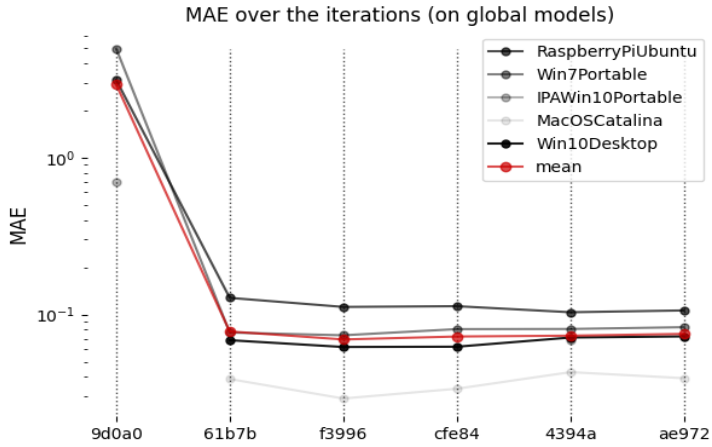

Figure 3: Progress of training in the FL experiment

5. Conclusion

Our presented use case allows the simulation and validation of the training and testing phase of the architecture enabling machine learning for industrial machinery and equipment. There is no need to collect and aggregate sensitive manufacturing data such as power consumption allowing potential attackers to gain information about production time or the usage of machine tools. In our study, we tackled the research questions how to combine the research fields of FL and use cases in manufacturing for anomaly detection into an industrial federated system. Therefore, we developed a generic architecture of communicating model

parameters of a machine learning model during the training and testing phase. With our architecture we depict the central components, structure them, and reveal their relationships. Based on the architecture the three important components for training and evaluating ML models in FL for machines used in manufacturing are the machine trainer, the global model coordinator and the model database. Consequently, the architecture and the functional components as research artifacts were prototypically implemented and validated.

Our results have several technical and organizational implications. First, we showed the possibility to combine the actual trends in industrial communication standards such as OPC-UA and privacy-sensitive machine learning environments in Python. Typical libraries (e.g. Keras and TensorFlow) can be used for training local data and aggregating it in a global model. Second, by applying FL manufacturing companies can use machine learning for anomaly detection without losing data ownership and let service providers get access to their data. Last, machine suppliers (e.g. compressor manufacturer) are able to sell machine learning services with guaranteed data ownership to their customers. In addition, our architecture is independent of machine type or operating system and acts as a system for training a model in which different machines participate. This is especially applicable for industrial machinery and equipment with computing capacity (e.g. edge devices) where machine learning can enable optimized processes. To extend the work in this paper the whole machine learning lifecycle and its AI system from data collecting to model tracking as mentioned in [18] will be developed. Therefore, the components and the information flows will be further validated and analyzed to guarantee a continuous improvement process.

References

[1] Chasin, F., Paukstadt, U., Ullmeyer, P., Becker, J., 2020. Creating Value From Energy Data: A Practitioner's Perspective on Data-Driven Smart Energy Business Models. Schmalenbach Bus Rev.

[2] Himeur, Y., Alsalemi, A., Bensaali, F., Amira, A., 2020. Anomaly detection of energy consumption in buildings: A review, current trends and new perspectives, 35 pp. http://arxiv.org/pdf/2010.04560v1.

[3] Chen, H., Fei, X., Wang, S., Lu, X., Jin, G., Li, W., Wu, X., 2014 - 2014. Energy Consumption Data Based Machine Anomaly Detection, in: 2014 Second International Conference on Advanced Cloud and Big Data. 2014 Second International Conference on Advanced Cloud and Big Data (CBD), Huangshan, China. 20.11.2014 - 22.11.2014. IEEE, pp. 136–142.

[4] M. De Nadai, M. van Someren, 2015. Short-term anomaly detection in gas consumption through ARIMA and Artificial Neural Network forecast, in: 2015 IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems (EESMS) Proceedings. 2015 IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems (EESMS) Proceedings, pp. 250–255.

[5] Patel, P., Ali, M.I., Sheth, A., 2018. From Raw Data to Smart Manufacturing: AI and Semantic Web of Things for Industry 4.0. IEEE Intell. Syst. 33 (4), 79–86.

[6] Mohammed M. Alani, Mohamed Alloghani, 2019. Security Challenges in the Industry 4.0 Era, in: , Industry 4.0 and Engineering for a Sustainable Future. Springer, Cham, pp. 117–136.

[7] Maschler, B., Jazdi, N., Weyrich, M., 2019. Maschinelles Lernen für intelligente Automatisierungssysteme mit dezentraler Datenhaltung am Anwendungsfall Predictive Maintenance. 0083-5560.

[8] Chen, B., Wan, J., Celesti, A., Di Li, Abbas, H., Zhang, Q., 2018. Edge Computing in IoT-Based Manufacturing. IEEE Commun. Mag. 56 (9), 103–109.

[9] Leitner, S.-H., Mahnke, W., 2006. OPC UA - Service-Oriented Architecture for Industrial Applications. ABB Corporate Research Center. https://pi.informatik.uni-siegen.de/gi/stt/26_4/01_Fachgruppenberichte/ORA2006/07_leitner-final.pdf.

[10] McMahan, H.B., Konečný, J., Yu, F.X., Richtárik, P., Suresh, A.T., Bacon, D., 2016. Federated Learning: Strategies for Improving Communication Efficiency. http://arxiv.org/pdf/1610.05492v2.

[11] Hiessl, T., Schall, D., Kemnitz, J., Schulte, S., 2020. Industrial Federated Learning – Requirements and System Design, in: de La Prieta (Ed.), Highlights in Practical Applications of Agents, Multi-Agent Systems, and Trustworthiness. The PAAMS Collection, vol. 1233. Springer International Publishing, Cham, pp. 42–53.

[12] Henning, S., Hasselbring, W., Burmester, H., Möbius, A., Wojcieszak, M., 2020. Goals and Measures for Analyzing Power Consumption Data in Manufacturing Enterprises, 24 pp. http://arxiv.org/pdf/2009.10369v1.

[13] Henning, S., Hasselbring, W., 2020. Scalable and Reliable Multi-dimensional Sensor Data Aggregation in Data Streaming Architectures. Data-Enabled Discov. Appl. 4 (1).

[14] Aledhari, M., Razzak, R., Parizi, R.M., Saeed, F., 2020. Federated Learning: A Survey on Enabling Technologies, Protocols, and Applications. IEEE Access 8, 140699–140725.

[15] Chen, Y., Wang, J., Yu, C., Gao, W., Qin, X., 2019. FedHealth: A Federated Transfer Learning Framework for Wearable Healthcare. https://arxiv.org/pdf/1907.09173.

[16] Ju, C., Gao, D., Mane, R., Tan, B., Liu, Y., Guan, C., 2020. Federated Transfer Learning for EEG Signal Classification. Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual International Conference 2020, 3040–3045.

[17] Lo Kit, S., Lu, Q., Wang, C., Paik, H.-Y., Zhu, L., 2020. A Systematic Literature Review on Federated Machine Learning: From A Software Engineering Perspective. http://arxiv.org/pdf/2007.11354v7.

[18] Kaymakci, C., Wenninger, S., Sauer, A., 2021. A Holistic Framework for AI Systems in Industrial Applications, in: Ahlemann, F., Schütte, R., Stieglitz Stefan (Eds.), Wirtschaftsinformatik 2021 Proceedings.

[19] Thulasiraman, K., Swamy, M.N.S., 1992. Graphs: Theory and algorithms. http://onlinelibrary.wiley.com/book/10.1002/9781118033104.

[20] Ullah, W., Ullah, A., Haq, I.U., Muhammad, K., Sajjad, M., Baik, S.W., 2020. CNN features with bi-directional LSTM for real-time anomaly detection in surveillance networks. Multimed Tools Appl, 1–17.

Biography

Can Kaymakci has been working as a research associate in the Industrial Energy Systems Department at Fraunhofer IPA. His current research topic is the concept and development of distributed machine learning applications for manufacturing with federated learning.

Lukas Baur joined the Industrial Energy Systems Department at Fraunhofer IPA as a research assistant in August 2020. Since then, he has been supporting the development of internal machine learning framework components.

Prof. Dr. Alexander Sauer is the Executive Director of the Institute for Energy Efficiency in Production (EEP) at the University of Stuttgart as well as the director of the Fraunhofer Institute for Manufacturing Engineering and Automation IPA.