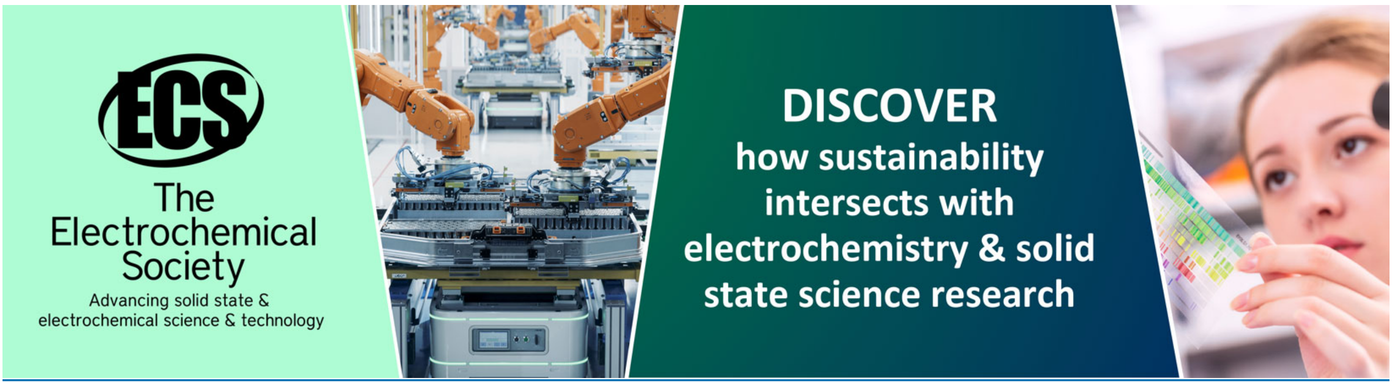
Assistance Method for the Application-Driven Design of Machine Learning Algorithms


ECS
The Electrochemical Society
Advancing solid state & electrochemical science & technology
DISCOVER
how sustainability intersects with electrochemistry & solid state science research

Assistance Method for the Application-Driven Design of Machine Learning Algorithms

Adalbert Fono[1], Gregor Thiele[1], Max Klein[1] and Jörg Krüger[2]
[1]Fraunhofer Institute for Production Systems and Design Technology, 10587 Berlin, Germany
[2] Head of the Department for Industrial Automation at the Institute for Machine Tools and Factory Management, Technical University of Berlin, 10587 Berlin, Germany

E-mail: adalbert.fono@ipk.fraunhofer.de

Abstract. Machine learning (ML) offers a lot of potential for applications in Industry 4.0. By applying ML many processes can be improved. Possible benefits in production are a higher accuracy, an early detection of failures, a better resource efficiency or improvements in quantity control. The use of ML in industrial production systems is currently not widespread. There are several reasons for this, among others the different expertise of data scientists and automation engineers. There are no specific tools to apply ML to industrial facilities neither guidelines for setting up, tuning and validating ML implementations. In this paper we present a taxonomy structure and according method which assist the design of ML architectures and the tuning of involved parameters. As this is a very huge and complex field, we concentrate on a ML algorithm for time series forecast, as this can be used in many industrial applications. There are multiple possibilities to approach this problem ranging from basic feed-forward neural networks to recurrent networks and (temporal) convolutional networks. These different approaches will be discussed and basic guidelines regarding the model selection will be presented.The introduced assistance method will be validated on a industrial dataset.

1. Introduction
Driven by faster computer hardware and more available data, ML approaches - in particular Deep Learning methods - reached a superior state of capability. Nowadays, we can see ML applied in a broad spectrum of applications, e.g. in the medical sector, in consumer electronics, and in fundamental research in natural science, technology, engineering and mathematics [1, 2, 3, 4]. In context of Industry 4.0 many processes can be improved by incorporating ML. Possible benefits in the context of production are pointed out in [5]. However, domain experts are rather unversed in concepts and techniques applied in ML. Our goal is to provide a guideline for finding the ML method suitable for given tasks in the domain of time series forecasting on industry data.

2. State of the art
The most popular classes of ML methods for time-series processing are based on artificial neural networks. We present a short overview with advantages and disadvantages with regard to time series forecasting.

Feedforward Neural Networks (NNs) can be applied in many different data domains - such as tabular data, image data, etc. However, they do not offer any intrinsic capability to deal with sequential data like time series. A possible workaround is to apply a sliding window and consider only the corresponding interval. In case of data with long-ranging time dependencies this approach becomes unfeasible as it requires large window sizes which lead to an intractable NN model. Hence, it can be preferable to solve the time dependencies by a customized feature vector which contains useful information like gradient, mean, variance and other properties for every involved time series; in order to use this vector as additional input data. However, recurrent neural networks (see below) seem like a more natural and better choice. A comparison on medical data was conducted in [6].

Recurrent Neural Networks (RNN) are inherently designed to handle sequential data and, thus, seem like the obvious choice to deal with time series data [7]. In practice, two slightly adapted specifications are very popular - the Long Short Term Memory (LSTM) [8] and the Gated Recurrent Unit (GRU) [9] architecture. Due to their construction the learning procedure runs more smoothly and yields superior results in comparison with the classical RNNs. There are only subtle differences in their design and in most cases they reach similar performance [10]. Therefore, in the remainder we focus on LSTM networks but the conclusions also hold for GRU nets. LSTMs can be used in multiple network types such as bidirectional [11] (where the input sequence is processed forward and backwards), stacked [12] (comprised of multiple LSTM layers) and encoder-decoder networks [9] (where the input sequence is encoded into a fixed length vector representation by the encoder part, and the decoder part decodes the representation into another sequence). In recent years LSTMs are often combined with an attention mechanism [13] which helps the network - as the name suggests - to focus its attention on the 'relevant' parts of the input sequence. Moreover, there are networks which solely rely on the attention mechanism and dispense with recurrence entirely [14]. The main limitation of LSTM nets is that they can merely process regular time series, i.e. sequences of evenly spaced (uniform sampled) data points. There have been made efforts to adapt LSTMs to this irregular case [15], however, to the authors' knowledge no straightforward and efficient solution exists. Instead of adapting the LSTM unit directly, many approaches introduce (complex) architectures which try to handle the irregularity by variable mechanisms like signal splitting, incorporating the time differences and learning dense representations for each time step [16, 17].

Convolutional Neural Networks (CNNs) were originally designed to cope with image data [18] but its strengths also transfer to sequential data [19]. Moreover, CNNs can also be combined with LSTMs [20] or even adapted to handle time series data naturally with so-called Temporal Convolutional Networks (TCNs) [21, 22]. The advantages of those networks are their ability to deal with long-ranging time dependencies in the input sequence and thereby allowing the processing of long sequences. Similar to RNNs, these architectures are ill-suited for irregular time series and additional effort is required for adaptation [23, 24].

3. Concept

Applying an ML method to a given task requires corresponding knowledge. Nowadays many solutions and frameworks exist to support the implementation. Nevertheless, many decisions - regarding model selection and architecture, preprocessing, etc. - have to be made and various parameter need to be tuned. We aim to assist the process of finding a suitable structure and parameters accordingly. Here, we propose a guideline, in particular for model selection, which facilitates this process by emphasizing critical steps and their underlying motivation.
With a look on industrial demands, we can shrink the scope of scenarios. So, we focus on data acquired/recorded from sensors and control devices. Generally, the data form a multivariate

irregular time series. Our assessment is based upon expert interviews and profound knowledge acquired from multiple projects in this domain.

3.1. Criteria for model selection

In Sec. 2 we have described multiple different models with their strengths and weaknesses. In general, it can be differentiated between two classes. The first category covers basic models like NNs and plain LSTMs that are straightforward to implement as beginner-friendly frameworks exist, e.g. Keras for Tensorflow [25]. The drawback is that these models either have limited capacities in relation to sequential data (NNs) or require specific input types (regular time series for LSTMs). Hence, in many cases additional effort is required in the data preparation process. The second category includes more complex models like evolved LSTMs or TCNs for irregular time series which offer an end-to-end solutions that require minimal data processing and enable superior precision. Unfortunately, implementation and learning processes are more complex and require in-depth ML knowledge. Following, we present relevant criteria for the model selection.

The nature of available data forms the starting point for the model selection. Is it favorable to use the original data or a re-sampled version as model input? Using the original data is mainly motivated by a straightforward data preparation step without any obstacles. However, this simpler pipeline comes at the cost of a more evolved ML model. In particular, for irregular time series the applied model must be capable of handling this data. Unfortunately, there is no straightforward model architecture in this case and it is still an open research challenge to design a suitable model (see Sec. 2). Design, implementation and training of these models requires advanced ML knowledge. Due to the non-uniform sampling, non-specialized models cannot grasp the underlying system dynamics, i.e. the training does not yield a useful model.
In contrast, inserting a resampling step in the data pipeline yields a straightforward model selection, even for data of uneven structure. In this case, most methods are applicable, i.e. even less complex models are sufficient which eases the training step as well as the implementation. The drawback is that a thorough analysis of the data is required. The main task is to find a suitable sampling rate. Inapt sampling rates may introduce different characteristics or neglect important information of the time series. The problem of non-uniform sampling is discussed in the signal processing community for a long time [26]. Especially in online applications, resampling needs to tackle all relevant phenomena in the data stream despite of the limited accessible interval.
If both the generic approach and the resampling fail, a customized feature engineering can offer a transformation of the temporal information. The data needs to be analyzed by experts in order to set-up a feature vector which incorporates characteristics of the time series, e.g. statistical parameters, time difference between points, and causalities. Plenty of transformations are explained in [27] including useful code snippets in python. These customized features may help to predict the system behavior based on non-uniform signals even though not performing a complete resampling.

The prediction horizon is the second important parameter to be set. Does the task require multi-step predictions or is a simple one-step prediction sufficient? Models and data are not equally appropriate for each scenario. Some models are not suited for multi-step predictions, e.g. basic NNs are less applicable in this scenario compared to LSTMs.
The data itself can be used as guidance to a suitable model. Are there long-ranging time-dependencies or rather immediate causalities? In the first case, rather complex models are required which can cope with long sequences because past events may still have important impact far later. For phenomena with short-term effects, a concise horizon can already adequately describe the state of the underlying system and, thereby, less complex models suffice.

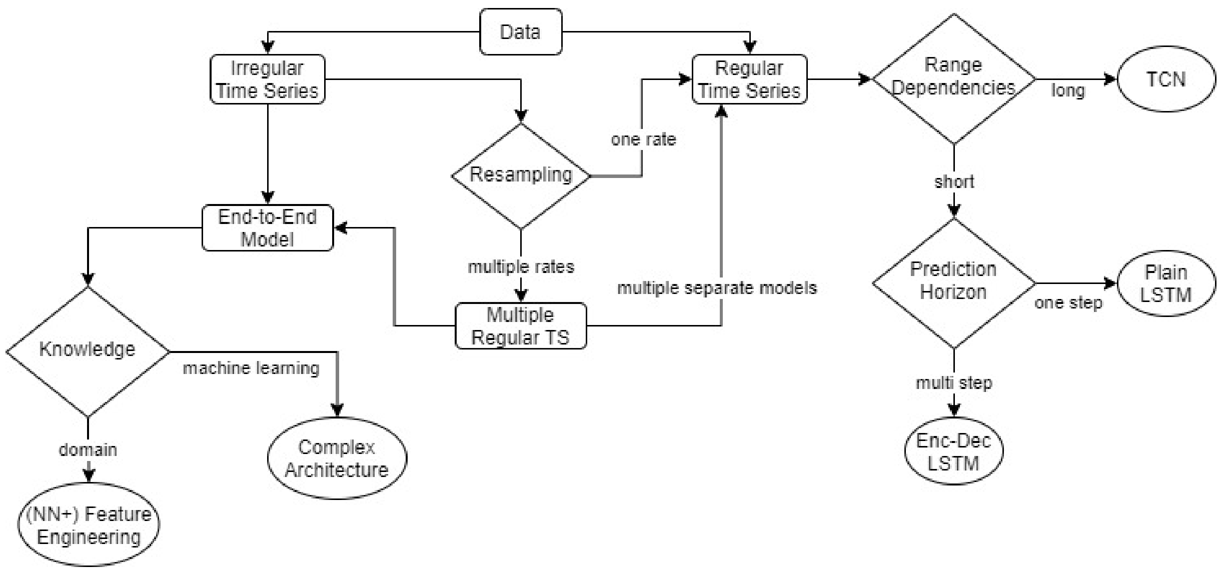


Figure 1. Taxonomy of the proposed concept

Model performance is a decision criterion that also needs to be considered. In an online environment with low latency requirements, computation time of the complete pipeline can be the bottleneck. In case of a large prediction horizon - compared with the latency - this problem diminishes. Similarly, if very high prediction accuracy is demanded, low complexity models may fail. In contrast, it is often easier to reach an acceptable floor using simpler models. Hence, it is worth to clearly weigh the options and demands beforehand.

3.2. Finding a suitable structure

How can the aimed ML pipeline be derived from the collected criteria? First, we can identify the following three main notions: 1) time series characteristics such as regular or irregular, long or short ranging dependencies, slow or fast dynamics; 2) addressed functionality of the model such as prediction horizon, one-step or multi-step prediction, required precision; 3) capabilities and effort of the developer such as domain knowledge, ML knowledge, development time. Then these notions and their evaluation for a specific task and data can serve as input arguments for a decision method which guides us to a proper structure of an ML model. The goal is to end up with the most reasonable starting point concerning performance as well as required effort. The taxonomy is presented in Fig. 1.

4. Practical application

We want to demonstrate the capabilities of our taxonomy on a practical example. Therefore, the process data from a combined heat and power plant was considered, in particular multiple heat sensors. Every sensor sends a base signal to the control unit in fixed time intervals, in our case 60 seconds. Additionally, the flexible send-on-delta sampling triggers the sensors in case of changes in the underlying signals exceeding a individual customized threshold. Hence, the data can be consider as a multivariate irregular time series. For checking the plausibility of process data, the 60 seconds base measurements are satisfying. Thus, focusing only on these measurements yields a regular time series with approximately the same information content which grasps the underlying process dynamics quite well. Therefore, an additional resampling step appears unnecessary. This leaves us with two options, either considering the irregular time

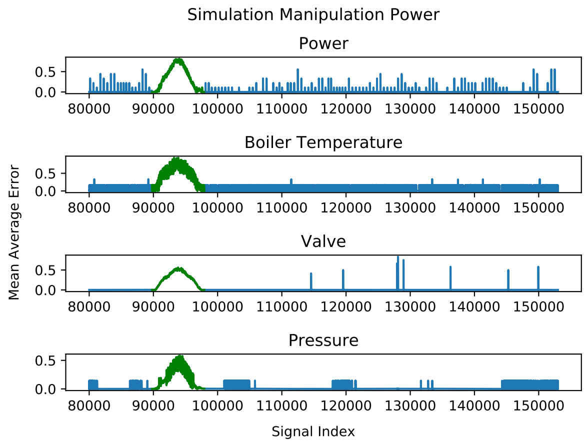


Figure 2. Simulation Manipulation Power, NN predictor

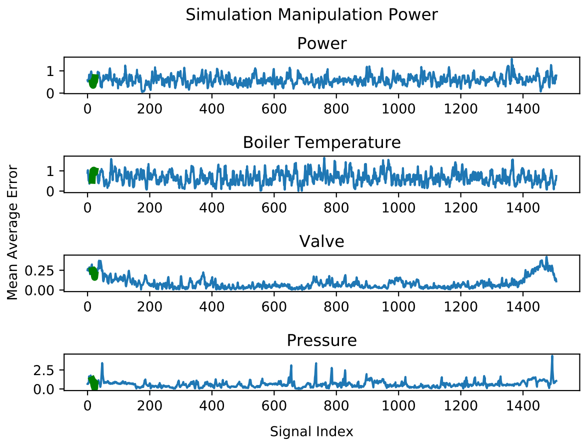


Figure 3. Simulation Manipulation Power, LSTM predictor

series with a complex model or involving only the regular time series using a simple model accordingly. Now, our decision method enables us to quickly determine a fitting model based on the desired use case.

Case-Study: Anomaly detection in power generation In [28], we compared two different ML algorithms developed for anomaly detection in distributed power generation. Both aim to handle time signals; the first one combines regular NN with customized features, the second one utilizes a plain LSTM. From a real power plant, we acquired data from multiple sensors. As explained above, they were sampled with both a uniform 60 seconds clock and triggered by relative changes which results which results in a non-uniform distribution.
We examined two artificial anomalies: scaling of a rotation speed measurement and overriding a power sensor in a calm operation point with a constant, i.e. signals are only generated uniformly every 60 seconds.. The first one leads to a contradiction in the underlying physics, e.g. every single instant is invalid. The second one results in unreasonable sampling which is only visible in the time information. While the LSTM approach performs superior in the first case, it fails to detect the odd sampling in the second case. In contrast, the combination of NN and a customized feature engineering detects the first manipulation less robust but achieves respectable results in the second case. Results of the second manipulation are shown in Fig. 2 and Fig. 3. This little experiments illustrates how design decision in machine learning can affect the performance.

5. Discussion and outlook

Confronted with plenty of possible ML configuration, we suggest a taxonomy to derive a suitable algorithm based on a methodical analysis of the use-case. The advantage of our decision method is that it enables the (inexperienced) user to choose a fitting model based on a quick analysis without in-depth knowledge of the whole field.
This approach poses the question if and how the covered design decisions can have decisive impact on the performance respectively. An input-output-schema could show how each architecture decision and parameter effects on the desired functionality, especially under consideration of the industrial context.

Acknowledgment
This research was proportionate supported in the project EnerSec (project number 16KIS0847) by the German Federal Ministry of Education and Research and in the project *Reinforcement Learning for complex automation technology* by Investitionsbank Berlin (IBB), co-financed by the European Regional Development Fund.

References

[1] Krizhevsky A, Sutskever I and Hinton G E 2012 Imagenet classification with deep convolutional neural networks *Proc. of the 25th Int. Conf. on Neural Information Processing Systems - Volume 1* NIPS'12 (Red Hook, NY, USA: Curran Associates Inc.) p 1097–1105

[2] Hinton G, Deng L, Yu D, Dahl G E, Mohamed A, Jaitly N, Senior A, Vanhoucke V, Nguyen P, Sainath T N and Kingsbury B 2012 *IEEE Signal Processing Magazine* **29** 82–97

[3] AlQuraishi M 2019 *Bioinformatics* **35** 4862–4865

[4] Silver D *et al.* 2016 *Nature* **529** 484–503

[5] Fahle S, Prinz C and Kuhlenkötter B 2020 *Procedia CIRP* **93** 413 – 418 ISSN 2212-8271 53rd CIRP Conference on Manufacturing Systems 2020

[6] Lipton Z C, Kale D C, Elkan C and Wetzel R 2015 *arXiv e-prints* arXiv:1511.03677 (*Preprint* `1511.03677`)

[7] Rumelhart D E, Hinton G E and Williams R J 1986 Learning internal representations by error propagation *Parallel Distributed Processing – Explorations in the Microstructure of Cognition* (MIT Press) chap 8, pp 318–362

[8] Hochreiter S and Schmidhuber J 1997 *Neural computation* **9** 1735–80

[9] Cho K, van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H and Bengio Y 2014 Learning phrase representations using RNN encoder–decoder for statistical machine translation *Proc. of the 2014 Conf. on Empirical Methods in Natural Language Processing (EMNLP)* (Doha, Qatar: Association for Computational Linguistics) pp 1724–1734

[10] Chung J, Gulcehre C, Cho K and Bengio Y 2014 Empirical evaluation of gated recurrent neural networks on sequence modeling (*Preprint* `1412.3555`)

[11] Schuster M and Paliwal K 1997 *Signal Processing, IEEE Transactions on* **45** 2673 – 2681

[12] Malhotra P, Vig L, Shroff G and Agarwal P 2015 Long short term memory networks for anomaly detection in time series *Proc. of 23rd European Symp. on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN 2015* pp 89–94

[13] Graves A, Wayne G and Danihelka I 2014 Neural turing machines (*Preprint* `1410.5401`)

[14] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser L and Polosukhin I 2017 Attention is all you need *Proc. of the 31st Int. Conf. on Neural Information Processing Systems* NIPS'17 (Red Hook, NY, USA: Curran Associates Inc.) p 6000–6010

[15] Neil D, Pfeiffer M and Liu S C 2016 Phased lstm: Accelerating recurrent network training for long or event-based sequences *Proc. of the 30th Int. Conf. on Neural Information Processing Systems* vol 29 pp 3882–3890

[16] Srivastava S, Sen P and Reinwald B 2020 *arXiv e-prints* arXiv:2004.03398 (*Preprint* `2004.03398`)

[17] Singh B P, Deznabi I, Narasimhan B, Kucharski B, Uppaal R, Josyula A and Fiterau M 2019 *arXiv e-prints* (*Preprint* `1905.00125`)

[18] LeCun Y and Bengio Y 1998 *Convolutional Networks for Images, Speech, and Time Series* (Cambridge, MA, USA: MIT Press) p 255–258

[19] Zhao B, Lu H, Chen S, Liu J and Wu D 2017 *J. of Systems Engineering and Electronics* **28** 162–169

[20] Livieris I, Pintelas E and Pintelas P 2020 *Neural Computing and Applications* **32** 1–10

[21] Lea C, Flynn M, Vidal R, Reiter A and Hager G 2017 Temporal convolutional networks for action segmentation and detection *2017 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* pp 1003–1012

[22] Bai S, Zico Kolter J and Koltun V 2018 *arXiv e-prints* (*Preprint* `1803.01271`)

[23] He Y and Zhao J 2019 *J. of Physics: Conf. Series* **1213** 042050

[24] Wan R, Mei S, Wang J, Liu M and Yang F 2019 *Electronics* **8** 876

[25] Chollet F *et al.* 2015 Keras `https://keras.io`

[26] Marvasti F A 2001 *Nonuniform sampling : theory and practice* (New York (N.Y.) : Kluwer academic/Plenum)

[27] Subasi A 2020 Chapter 2 - data preprocessing *Practical Machine Learning for Data Analysis Using Python* ed Subasi A (Academic Press) pp 27 – 89 ISBN 978-0-12-821379-7

[28] Klein M, Thiele G, Fono A, Khorsandi N, Schade D and Krüger J 2020 Process data based anomaly detection in distributed energy generation using neural networks *2020 International Conference on Control, Automation and Diagnosis (ICCAD)* pp 1–5