MM Science Journal | www.mmscience.eu
ISSN 1803-1269 (Print) | ISSN 1805-0476 (Online)
Special Issue | HSM 2021
16th International Conference on High Speed Machining
October 26-27, 2021, Darmstadt, Germany
DOI: 10.17973/MMSJ.2021_11_2021146

HSM2021-2021146

A DEEP LEARNING APPROACH TO ELECTRIC LOAD FORECASTING OF MACHINE TOOLS

B. Dietrich[1]*, J. Walther[1], Y. Chen[1], M. Weigold[1]

[1] Technical University of Darmstadt, Institute of Production Management, Technology and Machine Tools (PTW), Otto-Berndt-Str. 2, 64287 Darmstadt, Germany

*Corresponding author; e-mail: b.dietrich@ptw.tu-darmstadt.de

Abstract
The ongoing climate change and increasingly strict climate goals of the European Union demand decisive action in all sectors. Especially in manufacturing industry, demand response measures have a high potential to balance the industrial electricity consumption with the increasingly volatile electricity supply from renewable sources. This work aims to develop a method to forecast the electrical energy demand of metal cutting machine tools as a necessary input for implementing demand response measures in factories. Building on the results of a previous study, long short-term memory networks (LSTM) and convolutional neural networks (CNN) are examined in their performance for forecasting the electric load of a machine tool for a 100 second time horizon. The results show that especially the combination of CNN and LSTM in a deep learning approach generates accurate and robust time series forecasts with reduced feature preparation effort. To further improve the forecasting accuracy, different network architectures including an attention mechanism for the LSTMs and different hyperparameter combinations are evaluated. The results are validated on real production data obtained in the ETA Research Factory.

Keywords:
Machine learning; Machine tool; Load forecasting

1 INTRODUCTION AND MOTIVATION

The share of renewable energy sources, such as wind and solar energy, in global electricity generation has increased to almost 27% in 2020, with 6% growth in 2019, and is projected to increase further [International Energy Agency 2020]. Renewable energies are characterized by volatile electricity generation and thus reduced predictability compared to conventional electricity generation, which leads to challenges regarding the grid stability. In the past, conventional power plants matched the electricity demand and supply [Papaefthymiou 2018]. This control mechanism is no longer sufficient. Industrial demand-side management (DSM) represents a promising way to balance electricity consumption and supply, as the industrial sector is the largest electricity consumer with 41.6% global electricity consumption in 2016 [International Energy Agency 2018]. DSM leads to new saving opportunities in the industry through power procurement and demand response applications [Beier 2017]. For demand response applications, companies flexibly adapt their energy demand to the electricity supply by renewable energies. Those measures usually need an accurate electric load forecast of the respective system under consideration. The presented work focuses on deep learning for very short-term load forecasting of production machines as a basis for demand response applications on machine or factory level, for instance load shifting or peak shaving.

In the following sections, a brief literature review is conducted, the deep learning foundations are summarized, the experimental setup is presented, and the modelling results are evaluated and discussed. Eventually, a conclusion is drawn and future research fields are derived.

2 STATE OF THE ART

The term **load forecasting** refers to a systematic procedure for making statements about future energy demands [Clements 2000; Walther 2019a; Dietrich 2020]. Until today, load forecasting is mainly shaped by the supply side of the energy sector, where forecasts have been used for years to improve the information base and support the decision making process in the fields of energy purchasing, operations and maintenance [Hong 2010] or financial planning [Su 2017]. [Wang 2020] and [Ahmad 2017] forecast the load of industrial customers from energy supplier perspective using different machine learning approaches. However, these approaches did not include company-internal information like process data or production plans which can significantly improve forecasts of the electric load of industrial processes [Bracale 2017].

The application of short-term data-based electrical forecasts in the manufacturing sector is still a very young research field with few relevant scientific publications [Walther 2021]. On factory level, [Walther 2019b]

introduced a 15-minute forecasting model for the electricity consumption of a research factory based on Gradient Boosting Regression Trees. [Bracale 2017] used multiple linear regression to model the electricity consumption of an Italian factory 24 hours in advance, incorporating information about the production shifts.

On machine level, many approaches use physical models to model the electricity consumption using the main processing parameters spindle speed, feed rate, width, and depth of cut [Diaz 2010; Rajemi 2010] or material removal rate [Gutowski 2006]. However, physical modelling is usually more complex than data-based modelling and does not incorporate the stochastic nature of some signals [van Luttervelt 1998]. A data-based approach to forecast the electricity consumption of a machine tool was introduced by [Dietrich 2020]. The authors created a complex machine learning model to forecast the electricity consumption in the next 100 seconds with the aim to enable demand response applications on machine or factory level.

At present, in related forecasting tasks, there are many efforts in the field of Artificial Intelligence (AI) modelling techniques. One promising class of modelling techniques within the AI framework is deep learning. Deep learning models are particularly suitable for time series analysis, as they are capable of capturing the time varying dynamics of the underling system by considering several time steps simultaneously. [LeCun 2015; Bianchi 2017] Especially in the field of forecasting, deep learning modelling techniques seem to be promising. They show great results for related forecasting tasks such as renewable energies forecasting (see [Wang 2019] for a comprehensive review of several deep learning approaches), energy demand forecasting from the energy supply perspective [Bianchi 2017; Guo 2018; Tong 2018] as well as electrical [Cai 2019] and thermal [Fan 2017; Suryanarayana 2018] load forecasting in the building sector. Here, deep learning modelling techniques usually outperform conventional machine learning approaches in terms of accuracy, stability, and effectiveness. Especially in terms of feature engineering, deep learning models show a powerful capability to learn hidden patterns directly from raw data [LeCun 2015].

Since deep learning modelling techniques show superior results for these related forecasting tasks, the forecasting task presented in the previous work [Dietrich 2020] has been enhanced to a deep learning based forecasting model in this work. By that, we anticipate a reduced manual effort for feature engineering and an improved forecasting accuracy and stability.

A forecasting horizon of 100 seconds is selected for this work to be able to compare the results to the previous work [Dietrich 2020]. This forecasting horizon may be sufficient for very short term peak shaving on machine level, but should be prolonged to at least 15 minutes to be able to participate on the intraday market [Dietrich 2020].

3 DEEP LEARNING CONCEPTS

In general, deep learning can be classified as a sub-area of machine learning which in turn is a sub-area of AI. While AI can be generally defined as the study of intelligent agents, machine learning is rather a collection of data-driven algorithms that can learn from data without being explicitly programmed. Deep learning, in turn, refers to the study of Artificial Neural Networks (ANNs) and related machine learning algorithms that contain more than one hidden layer, also known as deep neural networks. [Ongsulee 2017]

NOMENCLATURE

AI	Artificial Intelligence
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
DSM	Demand-side management
LSTM	Long Short-Term Memory
MT	Machine Tool
R^2	Coefficient of Determination
RNN	Recurrent Neural Network
RMSE	Root Mean Squared Error

Two main concepts of deep learning are utilized in this work. Recurrent Neural Networks (RNNs) and especially its variant Long Short-Term Memory (LSTM) networks are well-suited for time series forecasting [Bianchi 2017], while Convolutional Neural Networks (CNNs) are a promising method for extracting features and patterns from data sequences like time series [Brownlee 2020a]. These concepts are introduced in more detail below with focus on the peculiarities and hyperparameters important for this work.

RNNs are a form of ANNs but additionally take into account the sequence or time domain of input and output by recursively self-connecting their neurons [Bianchi 2017]. A known drawback of simple RNNs is that they often fail to store information of long sequences due to so-called vanishing or exploding gradients. That means that especially for many time steps, a simple RNN may not be able to capture the long-term patterns. LSTMs, however, are able to store long-term information better. [Géron 2017] Like ANN, RNN and LSTM are trained using a back propagation algorithm, the so-called back propagation in time, for which the network is unfolded in the time domain.

The input of LSTM networks consists of fixed-length sequences of the input features. One input sequence is called a **sample**, and a certain number of samples that is passed to the network in one training step is called a **batch**.

Related to the internal state (or memory) transfer between batches and samples, there are two kinds of LSTM models: **stateless** and **stateful** LSTMs. In stateless LSTMs, the samples in a batch may be shuffled before training and the state is not passed on to the next batch, which means that the network learns from disconnected samples and does only consider the current input sequence for the current prediction. In contrast, batches are not shuffled and their state is passed on to the next batch in stateful LSTMs. They thus base their prediction on the current as well as past input sequences. The internal state of stateful LSTM networks may be reset manually, for example after a training epoch [Brownlee 2020b]. These two concepts are examined and compared in terms of their forecasting performance in this work.

According to [Brownlee 2020a], 1-dimensional CNNs are a suitable method for time series forecasting since they automatically extract the most important features from the given input sequences. CNNs consist of convolutional and pooling layers designed to aggregate and densify the original information. Convolutional layers apply filters (or **kernels**) to sub-sequences of the input, thereby aggregating the information into so-called feature maps. Hyperparameters of convolutional layers are the **number of filters** and the **kernel size**. Pooling layers apply aggregation functions like the maximum or mean to the input sub-sequences. They are used to prevent overfitting

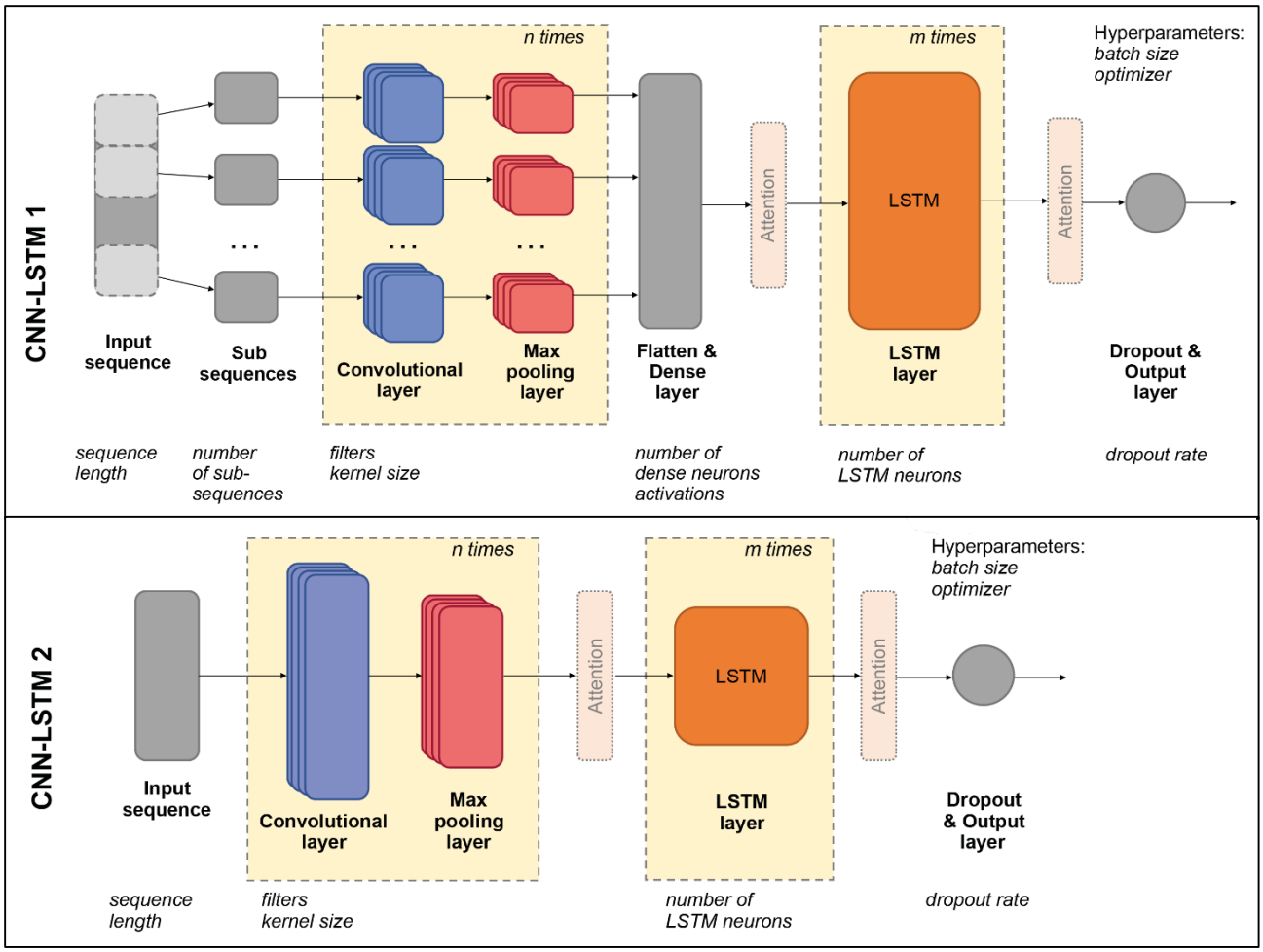

Figure 1: Neural network architecture of the CNN-LSTM models in this work, based on an illustration in [Livieris 2020].

and reduce computational requirements in CNNs. [Géron 2017] Following the procedure in [Livieris 2020], pooling layers have kernel size 2 and output the maximum in this work.

A further deep learning concept important for the presented approach is the **attention mechanism**. It is for example successfully applied in the field of language translation [Luong 2015], but also widely used for time series forecasting tasks, for example by [Kim 2019]. The attention mechanism is used to make hidden information in earlier layers available to the following layers [Kim 2019]. In LSTM-networks, it can be applied at different locations: **before** or **after** the LSTM-layers.

If the attention mechanism is used **after** the LSTM-layers, it makes the internal memory (hidden states) of the last LSTM-layer available to the following layers [Luong 2015]. It can be further distinguished into attention2D (2-dimensional output) or attention3D (3-dimensional output). Attention2D computes a weighted sum of the hidden states, while attention3D assigns weights to the hidden states without performing a summation. Attention3D needs to be followed by a flatten and dense layer to provide the expected 2D-output to the output layer. On the other hand, attention layers **before** the LSTM-layers are a kind of feature selection [Kim 2019]. They learn which inputs are most relevant for the following layers. In order to evaluate the added value of the attention mechanism to the presented architectures, it is applied at all the different described locations in this work.

The neural network architecture in this work combines the benefits of both CNN and LSTM. This architecture was for instance proposed by [Livieris 2020] for forecasting the gold price. The authors state that by using a CNN before an LSTM network, important features can be extracted by the CNN and time-dependencies in the data can be learned by the LSTM.

The two main architectures of this work, CNN-LSTM 1 and CNN-LSTM 2, are visualized in Figure 1. The tuned hyperparameters are added in italics beneath the respective layers and the locations of the attention layers are indicated. The main difference between the two architectures is the input shape. For CNN-LSTM 1, the input sequence is split uniformly into a number of sub-sequences which are passed into individual CNNs for feature extraction and then combined again for the LSTM in a flatten layer. This was done to first capture possible time dependencies within the subsequences with the CNNs and subsequently capture the long-term time dependencies with the LSTM. Conversely, for CNN-LSTM 2, the input sequence was fed directly into the CNN-layers without building sub-sequences, thus extracting features from the whole sequence. Both approaches are compared according to their forecasting performance in the results section.

The attention mechanism was tested at the indicated positions in Figure 1 for both models. The hyperparameter tuning, data preparation and experimental setup is explained in detail in the next chapter.

4 EXPERIMENTAL SETUP

This work represents a continued development of the work presented in [Dietrich 2020] with the objective to forecast the electric load of production machines with a forecasting

horizon of 100 seconds. Compared to the previous study, the experimental setup has not been modified, but the authors focused on the data from one machine tool and one production process (MT1-OP10) for model development. In particular, the examined machine is a vertical CNC grinding center for fine machining (EMAG VLC 100 GT) with automatic pickup system. It was run in automatic mode to ensure a constant cycle time. A reduced feature set containing the following input features was used:

- Electric load of machine tool and components (suction, hydraulic aggregate, cooling lubricant supply unit) (continuous)
- Operating mode 'working' of machine tool (discrete)

The data was collected directly from the machine PLC and an electricity meter with a sampling rate of 1 second. The data set includes production and non-production times.

The generalization capability of the model was evaluated on the data from another machine tool, a vertical CNC turning center (EMAG VLC 100 Y) (MT2-OP10). The data set size for each machine tool is listed in Table 1.

Table 1 : Amount of used data for model training and evaluation including the cycle time and number of cycles recorded for each machine tool.

Machine	Cycle Time (s)	Cycles	Data set size (s)
MT1- OP10	66	685	96,960
MT2- OP10	354	102	33,300

To prepare the forecasting task, the target needs to be shifted into the future according to the procedure in [Walther 2019b]. For simplicity reasons, a one-point-forecast was developed, meaning that the model only outputs the value in t+100 seconds. For data preprocessing, missing values were dropped, outliers were not treated and the MinMaxScaler was used for scaling the numerical features. These preprocessing steps were adjusted in preliminary studies [Chen 2020]. The data was divided into 70% training and 30% test data.

The main experiments were divided into four stages. In stage 1, two different kinds of simple stacked LSTM-networks without CNN-layers were compared: stateless and stateful LSTM. The slightly better performing stateless LSTM was then used in stage 2 to compare the more complex architectures including CNNs presented in section 3, CNN-LSTM 1 and CNN-LSTM 2.

In stage 3, the attention mechanism was introduced at the different locations within CNN-LSTM 1 and CNN-LSTM 2:

att-1: attention2D after LSTM
att-2: attention3D before LSTM
att-3: attention3D after LSTM
att-4: attention3D before **and** after LSTM

In stage 4, the best performing model architecture (CNN-LSTM 2 att-3) was transferred to MT2, examining its generalization capability by tuning and training it on a different data set. The stages in overview:

Stage 1: Stateless vs. stateful LSTM

Stage 2: CNN-LSTM 1 vs. CNN-LSTM 2

Stage 3: Attention mechanism

Stage 4: Transfer to different machine

Table 2 : Tuned hyperparameters and ranges for stage 1: stateful vs. stateless LSTM

Hyperparameter	Range [2]
Sequence length	100 (fixed)
Batch size [1]	choice(200,500,1000)
Optimizer	choice('adam','AdaGrad','RMSProp', 'AdaDelta')
LSTM layers	choice(2,3,4,5)
LSTM neurons	choice(30,40,50,60,70,80,90,100)
Dropout rate	choice(0,0.1,0.2,0.3,0.4,0.5,0.6,0.7)
Output activation	choice('sigmoid','linear','tanh')

[1] *Only for stateless LSTM*
[2] *choice(): random choice between the given options*

Table 3 : Tuned hyperparameters and ranges for stage 2, 3 and 4 : CNN-LSTM including attention

Hyperparameter	Range [4, 5]
Sequence length	quniform(100, 1000, 100)
Number of sub-sequences [1]	choice(factors(sequence length))
Batch size	quniform(100, 1000, 50)
Optimizer	choice('adam','RMSProp', 'AdaDelta')
CNN layers (*n*)	choice(2,3,4,5,6)
Kernel size	choice(2, 3, 4, 5)
Filters	choice(16, 32, 64)
Dense neurons (1) [1]	quniform(50,500,50)
LSTM layers (*m*)	choice(1,2,3)
LSTM neurons	choice(30,40,50,60,70,80,90,100)
Dense layers (2) [2]	choice(1,2,3)
Dense neurons (2) [3]	quniform(50,500,50)
Dropout rate	choice(0,0.1,0.2,0.3,0.4,0.5,0.6,0.7)
Output activation	choice('sigmoid','linear','tanh')

[1] *Only for CNN-LSTM 1*
[2] *Only for att-3*
[3] *Only for att-1 and att-3*
[4] *quniform(low, high, q): discrete uniform distribution between low and high value with step size q*
[5] *choice(): random choice between the given options*

For statistical evaluation, each model in each stage was trained five times for each tested hyperparameter set during optimization. Six evaluation metrics are used to compare the model performance: The mean of the Root-Mean-Squared-Error (RMSE) and the coefficient of determination (R^2), each on train and test data, and the standard deviation and maximum value of the R^2 on the test data. Additionally, the mean of the computation time for model training is evaluated. Since it depends on the hardware and parallel processes, it is here only used as a qualitative assessment and weighted lower than the accuracy and error terms. A naïve baseline model is used as a benchmark. The naïve model uses the current value as the forecast [Hyndman 2018]. In addition, the model performance is compared to the results of the previous work [Dietrich 2020].

The tuned hyperparameters and their respective ranges are listed in Table 2 and Table 3. Since the neural network architectures differ between stage 1 and 2 to 4, the hyperparameters are listed separately for stacked LSTMs

(stage 1) in Table 2 and CNN-LSTM (stage 2, 3, 4) in Table 3. The hyperparameter tuning was performed using the open source toolkit for automated machine learning Microsoft NNI [Microsoft 2021], with a Bayesian hyperparameter optimization approach using tree-structured Parzen estimator [Bergstra 2011]. The neural networks were implemented using the Python-library Keras and an early stopping mechanism was used to prevent overfitting.

Some peculiarities in the hyperparameter selection are discussed below. The input sequence length was fixed to 100 time steps for the stacked LSTMs in stage 1 but later treated as a hyperparameter and tuned between 100 and 1000 time steps in order to provide more time related information to the models. Since stateful LSTMs transport their internal state over multiple samples, the sequence length was not expected to impact their performance. The sequence length may however have an impact on stateless LSTMs since they do not propagate their internal states between samples. For CNN-LSTM 1, the number of sub-sequences and consequently number of CNN-cells per layer was tuned as random choice between whole factors of the selected sequence length. For the convolutional layers, the number of filters was tuned only for the first layer. In subsequent convolutional layers, the number of filters was doubled in each layer following the procedures in [Swapna 2018; Livieris 2020]. The architectures presented in section 3 were adjusted if attention layers were introduced after the LSTM layers. For attention2D, a dense layer was added after the attention layer and the number of neurons of this dense layer was tuned. For attention3D, a flatten layer to reduce the dimensionality was added followed by multiple dense layers of which the number of neurons and number of layers were tuned.

5 RESULTS

The results of the first three stages are listed in Table 5. First, an analysis regarding the LSTM network type was carried out. There is only a marginal difference between the two types. The stateless LSTM performed slightly better (Δ Test RMSE ≈ 1 W (0,1%), Δ Test R^2 ≈ 0,01) than the stateful LSTM. Its computation time, however, was three times lower. Therefore, stateless LSTM was used for the further analyses. For the model architecture comparison, CNN-LSTM 1 performed slightly better (Δ Test RMSE ≈ 25 W (2,1%), Δ Test R^2 ≈ 0,02), but its computation time was higher than that of CNN-LSTM 2.

Regarding the attention mechanism, the attention3D after LSTM (att-3) outperforms the other combinations for both architecture types with a slightly better test RMSE for CNN-LSTM 2. However, Table 5 shows that the difference between all attention mechanisms is not high. The difference between the best and least performing model in the test set is ΔRMSE 130 W (11,4%) and a ΔR^2 of 0.1. All attention mechanisms except attention2D deliver slightly better results than without attention. However, the difference between the test results for the best model with and without attention mechanism is only 44 W (3,8%). Introducing the attention mechanism does not lead to significant changes in the computation time.

To statistically validate the differences in the forecasting accuracy between the models, a two-step nonparametric statistical analysis was conducted. First, the Friedman's test was used for the comparison among all algorithms. The test allows to detect differences considering the global set of algorithms.

Table 4 : Hyperparameter selection of the best two models

Hyperparameter	CNN-LSTM 1 att-3	CNN-LSTM 2 att-3
Sequence length	100	100
Number of sub-sequences [1]	1	-
Batch size	950	950
Optimizer	adam	adam
CNN layers (*n*)	2	4
Kernel size	3	5
Filters	32	32
Dense neurons (1) [1]	150	-
LSTM layers (*m*)	1	1
LSTM neurons	90	90
Dense layers (2)	2	1
Dense neurons (2)	400	150
Dropout rate	0.0	0.0
Output activation	tanh	sigmoid

[1] *Only for CNN-LSTM 1*

With the ranked test results of the Friedman's test, the Holm test was conducted in the second step as a post-hoc procedure to find the concrete pairwise comparisons. The two-step procedure was conducted on the RMSE and R^2 values of the twelve models. The results show whether one of the proposed new methods offers a significant performance improvement. The null hypothesis of the tests is that there is no difference between the performance of the different algorithms. To reject this hypothesis, a significance level α of 0.05 was used. If the adjusted p-value of the Holm test is smaller than 0.05, there is a significant difference between the performance of the compared models. The Holm test for the RMSE values resulted in no statistically significant differences between the twelve models. Conversely, the Holm Test for the R^2 values resulted in five pairs showing a significant difference (see Table 6). When considering the results of the Friedman's test, the best performing algorithm can be identified (the higher the ranking the better). The three best performing models in order are therefore the CNN-LSTM 2 att-3, CNN-LSTM 1 att-3 and CNN-LSTM 1 att-2. The three worst performing models in order are stateful LSTM, stateless LSTM and CNN-LSTM 2 att-1.

Interestingly, the number of sub-sequences in CNN-LSTM 1 was selected to 1, making its architecture nearly identical to the CNN-LSTM 2. While batch size and optimizer were selected identically in both models, CNN-LSTM 2 had twice as many CNN-layers and a larger kernel size than CNN-LSTM 1. The LSTM-layers were selected identically, whereas the dense layers following the attention3D layer were different. Dropout was selected to 0.0 in both cases, while the activation of the output layer differed again.

Table 4 shows the selected hyperparameters of the best two models (CNN-LSTM 1 att-3 and CNN-LSTM 2 att-3).

The input sequence length was 100 in both cases. The predictions of the best model (CNN-LSTM 2 att-3) are visualized in Figure 2. As in the previous work, the cyclic parts and valleys are predicted well, but especially short-term peaks and the dressing operation (starting at about 5350 s) are not captured well by the model. This operation is performed every ten cycles by the machine tool.

The best two model architectures of the previous stages were then transferred to another machine tool (MT2) in stage four. The results are listed in Table 7. Although the R^2-score of the models of MT2 is significantly lower than

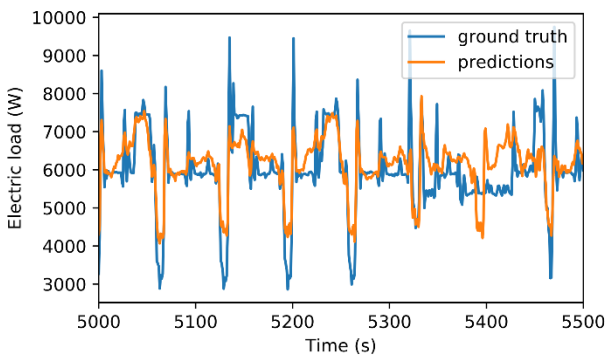

Figure 2 : Ground truth and predictions of CNN-LSTM 2 att-3 for a time period including four production cycles and the following dressing operation in the test set.

that of the models of MT1, it is also significantly higher than that of the previous work [Dietrich 2020]. Between the two tested model architectures, CNN-LSTM 1 att-3 performs slightly better. Different from the results of MT1, the train scores of MT2 are significantly better than the test scores, indicating a possible overfitting problem. However, a visual inspection of the train and test predictions yields that the predictions on train and test data are comparable.

6 DISCUSSION

The results show that the introduced deep learning models all perform similarly to the machine learning models introduced in the previous work. In terms of statistical tests, significant differences were only detected between the worst and best performing models when considering the test R^2-score. Therefore, we can conclude that the presented deep learning models are all similarly suited for industrial load forecasting regardless of the specific architecture.

However, some differences were detected in the computation time. Stateless LSTMs seem to require less computation time than stateful LSTMs with similar forecasting accuracy. A possible reason is the higher complexity of stateful LSTMs since they retain their internal state over longer time periods. In the presented use case, the stateful LSTM could have been expected to outperform the stateless LSTM regarding the dressing operation that has a long-term time dependency. This pattern, however, was not learned by the stateful LSTM. Therefore, stateless LSTMs seem sufficient and more efficient for the task.

Compared to the pure LSTM networks, the CNN-LSTM networks show a slightly better forecasting accuracy and significantly shorter computation time. A likely reason for the shorter computation time is the reduction of the time dimensionality performed by the CNN max pooling layers, greatly reducing the number of input time steps to the LSTM layers and thus reducing the time unfolding operations needed for training. Therefore, the combination of CNNs and stateless LSTMs seems to be more efficient for the task than pure LSTMs.

Regarding the attention mechanism, no statistically significant improvement of the forecasting accuracy or computation time can be noted. Attention3D after LSTM slightly improves the accuracy and is the best-performing model architecture in this work. However, further research is needed if it reliably improves the forecasting accuracy. Attention3D before LSTM and attention2D seem to have no or even a negative effect on the forecasting accuracy.

A closer look at the hyperparameter tuning results reveals that both best performing models have similar hyperparameters. In particular, an input sequence of 100 time steps seems sufficient for the task, combined with a large batch size of 950. The splitting of input sequences into different CNNs was neglected even in CNN-LSTM 1 by setting the number of sub-sequences to 1. That means that CNN-LSTM 2 is likely better suited for the task. The LSTM-layers were set to 1 in each case, which means that a small LSTM is sufficient following the CNN-operations. That has likely additional positive effects on the computational efficiency. The introduced dropout-layer after the LSTMs was apparently not needed since the dropout rate was set to 0 in both models. Possibly, the max pooling layers within the CNN and the early stopping mechanism already provide enough regularization, since no significant overfitting could be observed especially when comparing the train and test RMSE of the best models. Finally, the activation function of the output was not selected to be linear, which could be further examined in future research. Intuitively, in regression, the output activation should be linear since the target is continuous. However, an activation like sigmoid could lead to a more stable prediction since extreme prediction values are scaled down by the sigmoid transformation.

The transfer of the best model architecture to MT2 yields that the proposed deep learning architecture may be more robust and able to cope with less data than the models proposed in the previous work, since significantly less data was available for MT2. However, more tests regarding the amount of data and the transferability to other data sets are needed to verify this hypothesis.

Regarding limitations of the study and methodology, it should be noted that no cross-validation was performed during the experiments, which could possibly result in overfitting the test data sets. When looking at the train and test RMSEs, however, this seems unlikely. Furthermore, the hyperparameter space was set mostly by using the choice-function in NNI. Using probability distributions, e.g. quniform or lognormal, could further improve the hyperparameter optimization by providing more context to the Bayesian optimization algorithm. Additionally, the effect of computation time should be studied more carefully in future work by recognizing and eliminating external influencing factors like different hardware or parallel processes.

The presented models still need improvement to enable demand response measures. Especially the forecasting horizon should be increased to at least 15 minutes to allow for sufficient reaction time for the intraday electricity market [Buhl 2019]. The effect of the forecasting accuracy on the demand response measures needs to be examined further to allow for a well-founded assessment.

7 CONCLUSION

This paper presents a deep learning approach based on CNN and LSTM to forecast the very short term electric load of machine tools. Novelties of the presented work compared to previous studies are the application of deep learning methods to industrial load forecasting applications and especially the use of CNNs instead of a traditional feature engineering and selection process. We can conclude that CNNs are a good alternative to the complex feature preprocessing process, since the results are at least as good or better than those in the previous work. However, the attention mechanism improved the results just marginally. The deep learning architectures including CNNs

and stateless LSTMs prove computationally efficient and may be more robust and transferable to different data sets than previously proposed machine learning models.

Future research is still needed to verify the results especially concerning the computation time, to examine the transferability of the models, and to test the influence of the data set size.

Table 5 : Results of the main experiments with five trials each with the best result highlighted in bold. The computation time depends on hardware and parallel processes and therefore needs further investigation.

Experiment	Ø Test RMSE [W]	Ø Train RMSE [W]	Ø Test R^2 [-]	Ø Train R^2 [-]	σ Test R^2 [-]	max Test R^2 [-]	Ø Computation time [h:mm:ss]
Naïve baseline	1,704.66	1,607.60	0.17	0.64	-	-	-
Previous work *	1,226.13	-	0.62	-	0.01	0.63	-
Stateless LSTM	1,238.59	1,126.08	0.56	0.82	0.03	0.58	0:23:42
Stateful LSTM	1,239.67	1,191.45	0.55	0.80	0.02	0.57	1:07:04
CNN-LSTM 1	1,188.69	1,166.33	0.59	0.81	0.03	0.63	0:02:07
CNN-LSTM 2	1,213.47	1,057.12	0.57	0.84	0.05	0.61	0:00:29
CNN-LSTM 1 att-1	1,275.21	1,189.86	0.55	0.80	0.07	0.63	0:01:20
CNN-LSTM 2 att-1	1,216.82	953.88	0.57	0.87	0.03	0.60	0:02:57
CNN-LSTM 1 att-2	1,158.82	1,127.54	0.64	0.82	0.01	0.66	0:00:53
CNN-LSTM 2 att-2	1,185.35	1,167.74	0.59	0.81	0.02	0.62	0:03:29
CNN-LSTM 1 att-3	1,155.49	1,099.22	0.65	0.83	0.02	0.68	0:00:30
CNN-LSTM 2 att-3	**1,144.66**	**1,130.53**	**0.65**	**0.82**	**0.02**	**0.68**	**0:01:02**
CNN-LSTM 1 att-4	1,172.73	1,163.93	0.60	0.81	0.01	0.61	0:01:58
CNN-LSTM 2 att-4	1,196.64	1,168.40	0.58	0.81	0.02	0.61	0:18:28

** Train-test split was 75%/25% in previous work, 70%/30% in this work*

Table 6 : Significant results of the post-hoc Holm test of R^2-values including the rankings found by the previous Friedman test. An adjusted p-value of <0.05 indicates a statistical difference between two samples. A higher ranking indicates a higher R^2-score.

Comparison	Holm Test			Friedman Test	
	z-value	unadjusted p-value	adjusted p-value	ranking	
Stateful LSTM vs. CNN-LSTM 2 att-3	3.98	0.00007	0.005	Stateful LSTM CNN-LSTM 2 att-3	2.8 12.6
Stateful LSTM vs. CNN-LSTM 1 att-3	3.57	0.00035	0.027	Stateful LSTM CNN-LSTM 1 att-3	2.8 11.6
Stateful LSTM vs. CNN-LSTM 1 att-2	3.57	0.00035	0.027	Stateful LSTM CNN-LSTM 1 att-2	2.8 11.6
Stateless LSTM vs. CNN-LSTM 2 att-3	3.49	0.00048	0.036	Stateless LSTM CNN-LSTM 2 att-3	4 12.6
CNN-LSTM 2 att-1 vs. CNN-LSTM 2 att-3	3.41	0.00065	0.048	CNN-LSTM 2 att-1 CNN-LSTM 2 att-3	4.2 12.6

Table 7 : Results of the transfer of the best models to a different machine tool, five trials each.

Experiment	Ø Test RMSE [W]	Ø Train RMSE [W]	Ø Test R^2 [-]	Ø Train R^2 [-]	σ Test R^2 [-]	max Test R^2 [-]	Ø Computation time [h:mm:ss]
Naïve baseline	1,008.19	1,026.78	-0.03	0.16	-	-	-
Previous work *	1,052.05	-	0.07	-	-	-	-
CNN-LSTM 1 att-3	767.84	477.53	0.32	0.82	0.02	0.35	0:01:57
CNN-LSTM 2 att-3	770.36	600.65	0.30	0.71	0.08	0.40	0:01:23

** Train-test split was 75%/25% in previous work, 70%/30% in this work*

8 REFERENCES

[Ahmad 2017] Ahmad, A., et al. An Accurate and Fast Converging Short-Term Load Forecasting Model for Industrial Applications in a Smart Grid. IEEE Trans. Ind. Inf., 2017, 13, 5, 2587–2596, 10.1109/TII.2016.2638322.

[Beier 2017] Beier, J. Simulation Approach Towards Energy Flexible Manufacturing Systems. Sustainable Production, Life Cycle Engineering and Management, Cham, s.l.: Springer International Publishing, 2017, 978-3-319-46638-5.

[Bergstra 2011] Bergstra, J., et al. Algorithms for Hyper-Parameter Optimization. In: 25th Annual Conference on Neural Information Processing Systems (NIPS 2011): Neural Information Processing Systems Foundation.

[Bianchi 2017] Bianchi, F.M., et al. An overview and comparative analysis of Recurrent Neural Networks for Short Term Load Forecasting. 2191-5768, 2017, 10.1007/978-3-319-70338-1.
[Bracale 2017] Bracale, A., et al. Short-term industrial load forecasting: A case study in an Italian factory. In: 2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe): IEEE, 1–6.
[Brownlee 2020a] Brownlee, J. Deep Learning for Time Series Forecasting. Predict the Future with MLPs, CNNs and LSTMs in Python, 2020.
[Brownlee 2020b] Brownlee, J. Long Short-Term Memory Networks with Python. Develop Sequence Prediction Models with Deep Learning, 2020.
[Buhl 2019] Buhl, H.U., et al. Ausgangsbedingungen für die Vermarktung von Nachfrageflexibilität : Status-Quo-Analyse und Metastudie. 2 Fassung, 2019, 10.15495/EPub_UBT_00004455.
[Cai 2019] Cai, M., et al. Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques. Applied Energy, 2019, 236, 3, 1078–1088, 10.1016/j.apenergy.2018.12.042.
[Chen 2020] Chen, Y. Development of Load Forecasting Models for Machine Tools Based on Recurrent Neural Networks. Master Thesis, Technical University of Darmstadt, 2020.
[Clements 2000] Clements, M.P., and Hendry, D.F. Forecasting economic time series. The Marshall lectures on economic forecasting, Cambridge: Cambridge Univ. Press, 2000, 0521632420.
[Diaz 2010] Diaz, N., et al. Machine tool design and operation strategies for green manufacturing. UC Berkeley: Laboratory for Manufacturing and Sustainability, 2010.
[Dietrich 2020] Dietrich, B., et al. Machine learning based very short term load forecasting of machine tools. Applied Energy, 2020, 276, 115440.
[Fan 2017] Fan, C., et al. A short-term building cooling load prediction method using deep learning algorithms. Applied Energy, 2017, 195, 222–233, 10.1016/j.apenergy.2017.03.064.
[Géron 2017] Géron, A. Hands-on machine learning with Scikit-Learn and TensorFlow. Concepts, tools, and techniques for building intelligent systems, Beijing, Boston, Farnham, Sebastopol, Tokyo: O'Reilly, 2017, 978-1-491-96229-9.
[Guo 2018] Guo, Z., et al. A deep learning model for short-term power load and probability density forecasting. Energy, 2018, 160, 9, 1186–1200, 10.1016/j.energy.2018.07.090.
[Gutowski 2006] Gutowski, T., et al. Electrical energy requirements for manufacturing processes. Proceedings of LCE2006, 2006.
[Hong 2010] Hong, T. Short Term Electric Load Forecasting. Dissertation, North Carolina State University, 2010.
[Hyndman 2018] Hyndman, R.J., and Athanasopoulos, G. Forecasting. Principles and practice, [Melbourne]: OTexts, 2018, 9780987507112.
[International Energy Agency 2018] International Energy Agency. Key World Energy Statistics 2018, 2018.
[International Energy Agency 2020] International Energy Agency. Tracking Power, 2020. https://www.iea.org/reports/tracking-power-2020. Accessed 30 December 2020.
[Kim 2019] Kim, S., and Kang, M. Financial series prediction using Attention LSTM, 2019.
[LeCun 2015] LeCun, Y., et al. Deep learning. Nature, 2015, 521, 7553, 436–444, 10.1038/nature14539.
[Livieris 2020] Livieris, I.E., et al. A CNN–LSTM model for gold price time-series forecasting. Neural Comput & Applic, 2020, 32, 23, 17351–17360, 10.1007/s00521-020-04867-x.
[Luong 2015] Luong, M.-T., et al. Effective Approaches to Attention-based Neural Machine Translation, 2015.
[Microsoft 2021] Microsoft. NNI. Neural Network Intelligence, 2021. https://nni.readthedocs.io/en/stable/. Accessed 27 April 2021.
[Ongsulee 2017] Ongsulee, P. Artificial intelligence, machine learning and deep learning. In: Proceedings 2017 Fifteenth International Conference on ICT and Knowledge Engineering. November 22-24, 2017, Bangkok, Thailand, Piscataway, NJ: IEEE, 1–6.
[Papaefthymiou 2018] Papaefthymiou, G., et al. Power System Flexibility Tracker: Indicators to track flexibility progress towards high-RES systems. Renewable Energy, 2018, 127, 6, 1026–1035, 10.1016/j.renene.2018.04.094.
[Rajemi 2010] Rajemi, M.F., et al. Sustainable machining: selection of optimum turning conditions based on minimum energy considerations. Journal of Cleaner Production, 2010, 18, 10-11, 1059–1065, 10.1016/j.jclepro.2010.01.025.
[Su 2017] Su, P., et al. Recent Trends in Load Forecasting Technology for the Operation Optimization of Distributed Energy System. Energies, 2017, 10, 9, 1303.
[Suryanarayana 2018] Suryanarayana, G., et al. Thermal load forecasting in district heating networks using deep learning and advanced feature selection methods. Energy, 2018, 157, 141–149, 10.1016/j.energy.2018.05.111.
[Swapna 2018] Swapna, G., et al. Automated detection of diabetes using CNN and CNN-LSTM network and heart rate signals. Procedia Computer Science, 2018, 132, 1253–1262, 10.1016/j.procs.2018.05.041.
[Tong 2018] Tong, C., et al. An efficient deep model for day-ahead electricity load forecasting with stacked denoising auto-encoders. Journal of Parallel and Distributed Computing, 2018, 117, April, 267–273, 10.1016/j.jpdc.2017.06.007.
[van Luttervelt 1998] van Luttervelt, C.A., et al. Present Situation and Future Trends in Modelling of Machining Operations Progress Report of the CIRP Working Group 'Modelling of Machining Operations'. CIRP Annals - Manufacturing Technology, 1998, 47, 2, 587–626, 10.1016/S0007-8506(07)63244-2.
[Walther 2019a] Walther, J., et al. Generic Machine Learning Approach for very short term load forecasting of production machines. Proceedings of the International Conference on Applied Energy 2019, 2019.
[Walther 2019b] Walther, J., et al. Very short-term load forecasting on factory level – A machine learning approach. Procedia CIRP, 2019, 80, 705–710, 10.1016/j.procir.2019.01.060.
[Walther 2021] Walther, J., and Weigold, M. A Systematic Review on Predicting and Forecasting the Electrical Energy Consumption in the Manufacturing Industry. Energies, 2021, 14, 4, 968, 10.3390/en14040968.
[Wang 2019] Wang, H., et al. A review of deep learning for renewable energy forecasting. Energy Conversion and Management, 2019, 198, 111799, 10.1016/j.enconman.2019.111799.
[Wang 2020] Wang, Y., et al. Short-Term Load Forecasting for Industrial Customers Based on TCN-LightGBM. IEEE Trans. Power Syst., 2020, 1, 10.1109/TPWRS.2020.3028133.