



Predict-IT - Forecasting District Heating Loads: An Open-Source and User-Friendly Neural Network-Powered Platform

Léo Bonal¹, Marnoch Hamilton-Jones^{2,3}, Zahra Nasrollahinayeri¹, Katharina Dimovski¹,
Doris Entner¹ , Philip Ohnewein² , and Harald Trinkl⁴

¹V-Research GmbH, Dornbirn, Austria

²AEE INTEC, Gleisdorf, Austria

³Institute of Software Technology, TU Graz, Austria

⁴GET, Güssing, Austria

*Correspondence: Léo Bonal, leo.bonal@v-research.at

Abstract: In the realm of many thermal energy systems, and particularly within district heating networks, heat load forecasts play a pivotal role in optimizing system operation and efficient infrastructure usage. While district heating operators routinely log measurement data, its potential remains underutilized. One essential application of such data is forecasting a network's heat load based on historical data records. Such forecasts can improve the efficient usage of plant infrastructure and facilitate predictive operational strategies. This paper introduces "Predict-IT", a web-based platform designed to standardize the entire forecasting pipeline, making the generation of predictions largely independent of expert knowledge. The Predict-IT platform is powered by a state-of-the-art long short-term memory (LSTM) based neural network algorithm which only requires very little inputs (measured heat load and ambient temperature) to deliver satisfying forecasting accuracy, even a couple of days ahead. The prediction algorithm is validated on two data sets from local Austrian district heating networks, showing the general applicability of the LSTM-based neural network, given an appropriate set of hyperparameters. The Predict-IT platform simplifies the process of forecasting heat loads into a few discrete steps: data upload, algorithm training, heat load forecast generation, and visualization of forecasts. The source code will be open-source, and deployment and installation will be facilitated by an easily installable Docker solution.

Keywords: Heat Load Forecast, LSTM-Based Neural Network, Web-Based Platform, Open-Source Software

1 Introduction

Forecasting the heat load of district heating plants remains an active research area, as evidenced by recent works such as [1], [2]. While existing literature primarily focuses on creating and refining prediction algorithms, this paper takes a different approach. Here, we introduce "Predict-IT", a user-friendly web-based application designed for

heat load forecasts of district heating networks. We describe the state-of-the-art algorithm that powers these forecasts, and how Predict-IT wraps this core algorithm, enabling straightforward application at specific district heating plants. Leveraging docker containers, deployment and installation are streamlined, and platform-independent usage of Predict-IT is ensured. The Predict-IT software and its core algorithms will be licensed under an open-source software license which explicitly allows for commercial use and distribution.

To enhance broad applicability, the Predict-IT platform is independent of the plant control system and requires minimal inputs, namely historic measurement data of a network's total heat load and ambient temperature as time series. The software automatically retrieves past and future ambient temperature values from an online weather service, using the given plant location. The long short-term memory (LSTM) based neural network algorithm uses the weather forecasts to generate predictions of the future heat load, whereas the forecast horizon (e.g., 36 hours ahead, or three days ahead) and forecast frequency (e.g., forecasts for every 20 minutes, or for every hour over the forecast horizon) can be chosen by the user. Overall, the Predict-IT platform for heat load forecasts reduces reliance on expert knowledge and the impact of staff fluctuations, ensuring a more stable and reliable operational framework for district heating networks.

This paper is organized as follows. In Section 2, the underlying approach is discussed, describing the required data and necessary data preparation steps, and explaining the LSTM-based neural network used in the forecasting algorithm and the details of training and optimizing the neural network. Section 3 presents the data of the two local Austrian heating networks, and contains the results and validation of the algorithm tuning and prediction performance. In Section 4, the Predict-IT software platform is introduced. Finally, Section 5 discusses the results and outlines future work, and Section 6 concludes the paper.

2 Methodology and Technology

This section outlines the procedure of the Predict-IT software for generating forecasts. Predict-IT utilizes historical time series data representing measurements of a district heating network's heat load and ambient temperature. Figure 1 illustrates the data flow diagram of Predict-IT, involving the following steps: Data extraction from control systems or data loggers and data upload into the Predict-IT platform are followed by the data pre-processing stages (see Section 2.1): Data format standardization (time stamps, units, etc.), algorithm-specific data preparation (normalization, handling missing values, feature engineering, etc.), and the generation of sub-samples. Following this, the data is partitioned into training and test sets to facilitate algorithm training and evaluation (see Section 2.2). During the training phase, the LSTM-based neural network algorithm is trained and evaluated on the test set, and the algorithm hyperparameters are optimized and stored for subsequent use (see

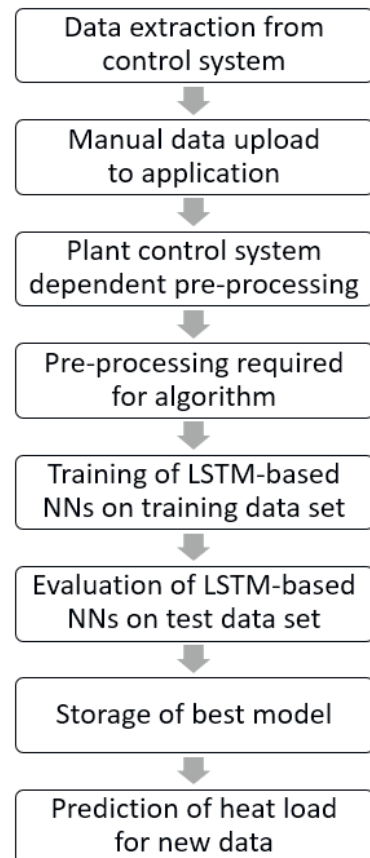


Figure 1. Data flow diagram

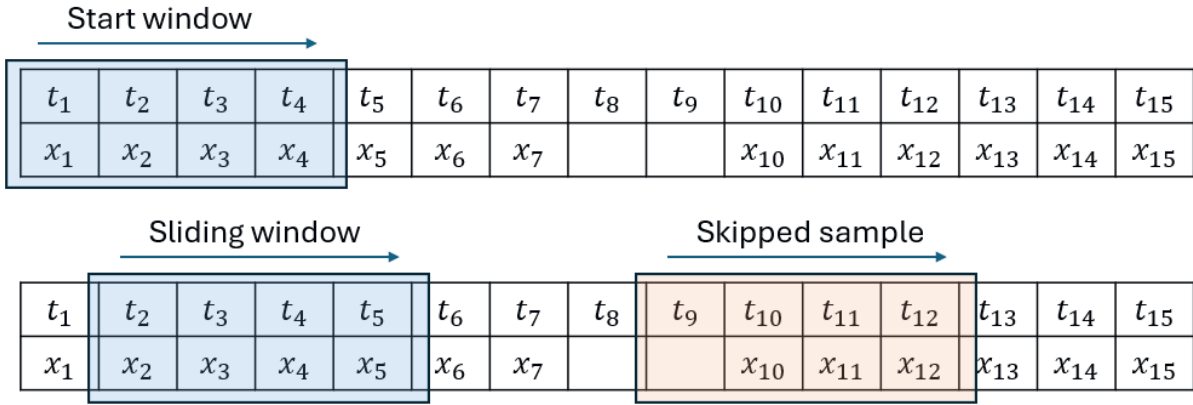


Figure 2. Illustration of the sliding window approach with a window length of 4. The top image shows the first sample comprising the first 4 data measurements; the bottom image demonstrates the shift to the second sample, comprising the 2nd to 5th measurements. If any observations are missing within the window, the entire sample is disregarded.

Section 2.3). Utilizing the stored model, heat load forecasts can be generated with minimal computational effort (see Section 3).

In terms of the Predict-IT modeling technology, data processing as well as model learning and hyperparameter tuning are implemented in Python using Tensorflow [3], [4]. Predict-IT employs the state-of-the-art Django web framework [5] to facilitate user interaction (data upload, training, etc.) and to make forecast results available for visualization and download. The deployment strategy for Predict-IT, including all necessary components, is envisaged to be based on Docker [6], facilitating the packaging, deployment and installation processes.

2.1 Data pre-processing

In a first step, the historical heat load and ambient temperature time series undergo a preparation step to ensure they adhere to the required formatting in terms of time stamps and physical units. Subsequently, these pre-processed data are resampled to achieve uniform time intervals, such as 20-minutes or hourly intervals, depending on the original data's sampling rate. This resampling step is necessary because measurement data are typically not available at constant sampling rates. Notably, if there are gaps between consecutive measurements that exceed a user-defined threshold, no resampling is done, and such data points are marked as missing values (compare Figure 2).

In preparation for the algorithm, the resulting time series is segmented into equally-sized samples using a sliding window approach, illustrated in Figure 2. The length of the sliding window varies based on the resampling rate and the forecast horizon. For example, with data resampled every 20 minutes (3 values per hour) and the forecast horizon of 3 days (24·3 hours), the window length would be $3 \cdot 24 \cdot 3 = 216$, meaning each sample comprises 216 individual measurements. Alternatively, if data were resampled every hour, a sample would comprise $24 \cdot 3 = 72$ measurements.

To ensure accurate forecasting, distinctive patterns within the heat load time series are exploited. The heat load of a district heating network represents the overall thermal power required by all connected customers. As such, the data encapsulate patterns that represent daily, weekly and seasonal fluctuations: Daily patterns (e.g. different hot water usage in the morning and at lunch, different building heat losses day and night),

weekday patterns (e.g. different hot water or industry loads, depending on workday or weekend), and seasonal patterns (reduced thermal loads in spring or summer compared to autumn and winter). To effectively encode these patterns for algorithm training, feature engineering is employed: For instance, the "weekday" information could be encoded by integer numbers (1 to 7, assigned to each weekday from Monday to Sunday), depicted as green star marks in Figure 3. However, this would result in a discontinuity at the end of the week, even though weekdays are cyclic. Therefore, a sinusoidal encoding approach is adopted, where each weekday is projected onto sine (blue circle marks) and cosine (orange cross marks), providing smooth transitions. Similarly, encoding for time of day utilizes sine and cosine functions, and encoding for seasonal effects involves projecting the distance from each day to the 21st of June to sine and cosine functions. Additionally, the presence of public holidays is encoded using a boolean variable.

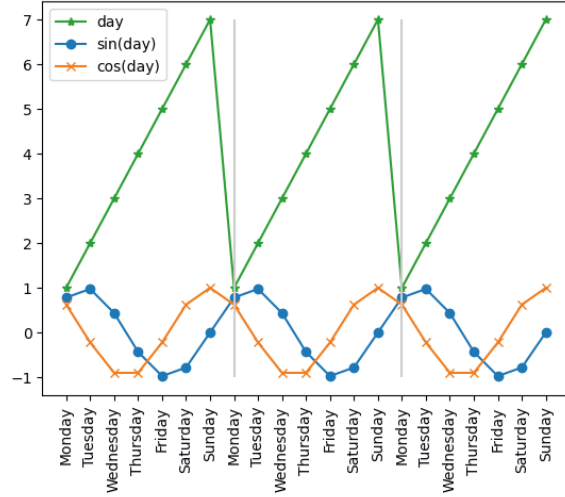


Figure 3. Feature engineering.

2.2 Algorithm

For time series forecasting, various data-driven algorithm approaches are available, broadly categorized into two groups: Statistical models of the ARIMA/SARIMA class, which model time correlations among input variables using statistical methods with a well-established theoretical foundation, and machine learning algorithms such as boosting and support vector regression (see e.g. [2], [7]). In Predict-IT, a state-of-the-art LSTM-based neural network is adopted due to its capability to represent time series data with complex system dependencies [1], [8]. As a downside, LSTM-based neural networks typically require longer training times compared to statistical algorithms.

The forecast horizon, representing the time span into the future for which Predict-IT generates heat load forecasts, is a crucial parameter in forecasting models, specifying granularity and accuracy of the predictions. In Predict-IT, the forecast horizon is a parameter that can be adjusted by users in the web-based interface (see Section 4), allowing users to tailor the Predict-IT forecasts to their specific needs.

The model structure utilized in Predict-IT is illustrated in Figure 4: It consists of two LSTM models and a dense output layer. The first LSTM processes heat load data from previous days as inputs x_i , with the number of inputs defined by a sliding window size; it outputs an internal state, s_0 . This state s_0 and the weather forecasts w_i (ambient temperature at the given location) are fed into the second LSTM, producing internal states s_i . Finally, a dense layer is applied to obtain the outputs, i.e. the forecasts of future heat loads, \hat{x}_i .

2.3 Model training

In this section, we provide insights into the training of the LSTM-based neural networks, focusing particularly on hyperparameter tuning, i.e., the selection of the optimal values

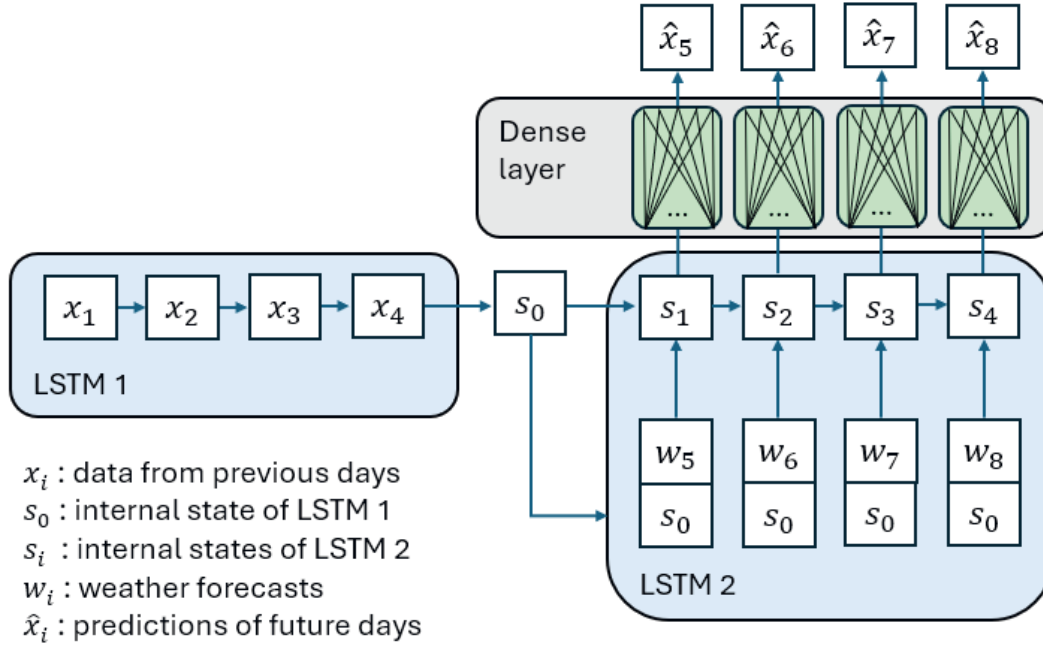


Figure 4. Model diagram of the LSTM-based neural network

for the model hyperparameters. The objective is to find good candidate sets of hyperparameters which can be used by plant operators as a starting point for model training without requiring the complete, computationally expensive hyperparameter optimization. Speeding up the hyperparameter optimization is possible using parallel computing techniques, yet this requires a CUDA compatible graphics card, which is not typically available at the hardware of local district heating plant operators.

Towards this end, the neural networks were trained on 70 % of the available time series data. The input consists of three days of historical measurement data of the total heat load, resampled to 20-minutes intervals (i.e. a sliding window size of 216), and the output delivers predictions for the heat loads of the coming three days, in 20-minutes intervals, using future ambient temperature as input. Presently, measured ambient temperatures are used instead of weather forecasts (see Section 5 for further discussion). We use the ready-implemented grid search approach of KerasTuner [9] to obtain a set of hyperparameters that yield satisfying forecasts. Model training was conducted on a NVIDIA Quadro P2000 graphics card.

Hyperparameter tuning of the Predict-IT forecasting model - the LSTM-based neural network - involves the optimal choice of three parameters: the learning rate (determining the speed at which the neural network learns by incorporating new knowledge), the number of latent dimensions in the network (reflecting model complexity), and the batch size (indicating the number of samples processed before model parameters are updated). During hyperparameter optimization, the number of epochs (representing the number of times the model iterates through the entire dataset) is kept constant at 40, with an early stopping strategy implemented to prevent overfitting: The training is stopped once the validation loss starts to increase again after reaching a local minimum. After these first optimization steps, the number of epochs is tuned by training this model 20 times and averaging the result. The used hyperparameter values are listed in Table 1. In total, this grid search approach leads to $3 \cdot 10 \cdot 2 = 60$ combinations of hyperparameters.

Table 1. Possible values of the hyperparameters for the grid search.

Hyperparameter	Values
Learning rate	{0.003, 0.005, 0.009}
Latent dimensions	{4, 6, 8, 10, 12, 16, 32, 64, 128, 256}
Batch size	{256, 512}
Epochs	First constant with early stopping, then {1, ..., 40}

Table 2. Details of the data sets

	Plant A	Plant B
Location	Austria	Austria
Characteristics	Private households only	Private households, two larger buildings
Observation period	10 years	5 years
Missing data	330 days 10 hours	324 days 21 hours
Data samples	200.888 samples	73.328 samples

For all experiments, the Mean Squared Error (MSE) is used as the loss function on the test set, measuring the averaged squared differences between forecasted and measured heat loads. Each hyperparameter combination undergoes five trials using a grid search approach, with the final MSE computed as the average across these five trials.

3 Results and Validation

In Section 3.1, we describe the data sets of two local Austrian district heating networks used as use cases for validating the Predict-IT algorithm. For the two use cases, we present the best found model hyperparameters (Section 3.2), the heat load forecasts and discuss the applicability of a general hyperparameter set (Section 3.3).

3.1 Test Plants and Datasets

In order to validate the Predict-IT algorithm, the development team had access to two datasets from local Austrian district heating networks. The LSTM-based neural network was trained and evaluated on both datasets. Table 2 summarizes the two datasets.

- The first district heating network, referred to as Plant A, consists exclusively of private households as customers, with measurement data spanning a period of ten years, albeit with approximately one year of data missing. These gaps resulted from several periods, lasting between 6 and 26 consecutive days. Employing a sliding window of length 216, as defined for the algorithm training (see Section 2.3), a total of 200.888 data samples were derived for Plant A.
- The second district heating network, referred to as Plant B, predominantly consists of households, but also of two larger buildings, one of which is mostly used on work days. Data observations span five years, with nearly one year of data missing. Most periods of missing data span between 6 and 17 consecutive days, with one longer data gap of approximately half a year. From these data, 73.328 samples are generated using a sliding window of length 216.

3.2 Validation of hyperparameter tuning

The results of the hyperparameter tuning are illustrated in Figures 5 (grid search) and 6 (epochs). During the application of the algorithm on Plant A, for 256 latent dimensions only a batch size of 256 (but not of 512) was utilized due to hardware constraints on the graphics card. Consequently, 57 combinations of hyperparameters were explored for Plant A and 60 combinations for Plant B. For Plant A, the best performing combination of hyperparameters comprises a learning rate of 0.005, a batch size of 512, and 6 latent dimensions, indicated by the white-coloured, large circle in Figure 5 (a) at position 6 at the x-axis and position 0.005 at the y-axis. Figure 6 (a) depicts the tuning process for the number of epochs for this specific hyperparameter combination. Averaging over all the results (orange line) indicates an optimal value of 12 epochs. Conversely, for Plant B, the hyperparameter tuning yields slightly different results, as evidenced by Figure 6 (b): Here, the optimum learning rate is the same at 0.005, but the batch size is 256 and the number of latent dimensions is 16; Figure 5 (b) reveals an optimal number of epochs of 18 for Plant B.

A note on model complexity: While for Plant A the best results are reached with a small latent dimension, indicating a less complex model, Plant B required more latent dimensions, indicating a more complex model. This discrepancy could be attributed to the structure of the underlying district heating network: Plant A has a rather homogeneous customer set (only private households), whereas Plant B includes a mix of private households and larger buildings, resulting in increased variability of the measured heat load data.

A note on generalizability: Overall, as can be seen from Figure 5, the hyperparameter combinations that are optimal for Plant A yield a larger MSE when applied on the data of Plant B, and vice versa. This implies that utilizing a single set of hyperparameters across multiple district heating plants is not sufficient, in practice. To further investigate this, forecasts based on each hyperparameter set are evaluated using the datasets of both plants (see Section 3.3).

A note on training time: Finally, we want to point out that both optimal model hyperparameters do not require the largest number of latent dimensions, nor the smallest learning rate. In practice, this implies that the models can be trained on standard computers without dedicated CUDA-compatible graphics cards.

3.3 Validation on Test Plants

After training the models and optimizing hyperparameters for Plant A and Plant B, the Predict-IT algorithm was ready for generating heat load forecasts for the plants. To assess the quality of the two inferred hyperparameter sets and the forecasts, a total of four models were trained, as outlined in Table 3: For each plant, two models are trained, based on the model hyperparameters derived in Section 2.3. For instance, the model labeled "*Model_hypB_datA*" in Table 3 utilizes data from Plant A to generate forecasts for Plant A, but employing hyperparameters inferred for Plant B.

All models were trained to forecast three days ahead in 20-minute intervals, based on measured ambient temperature (see Section 5 for further discussion). Prediction results were smoothed by summing three consecutive values, yielding hourly forecasts with reduced noise.

The two models trained with the same plant data and hyperparameters (referred to as *Model_hypA_datA* and *Model_hypB_datB*) are utilized to assess forecast quality,

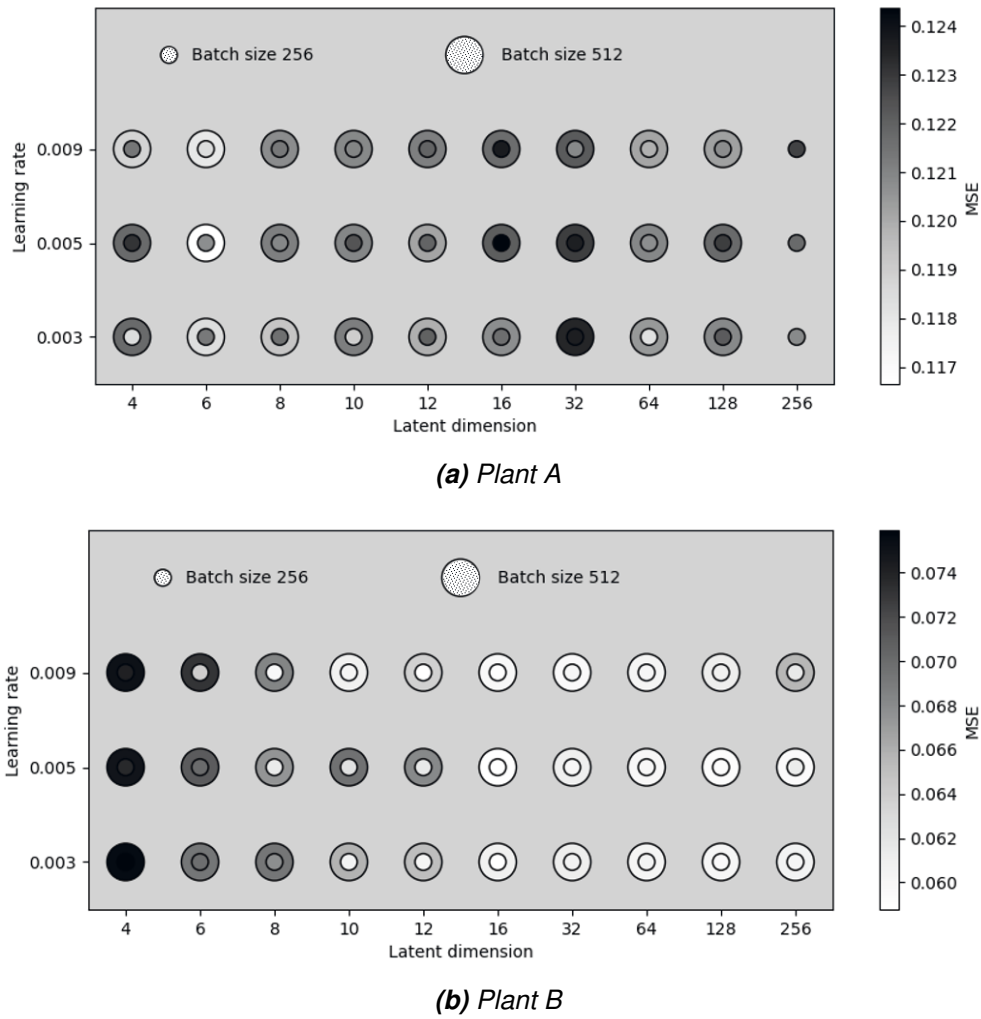


Figure 5. Results of the grid search for (a) Plant A and (b) Plant B. Along the x-axis are the latent dimensions, along the y-axis the learning rates. Each combination of latent dimension and learning rate was run with two batch sizes: 256 (represented by the smaller circle), and 512 (larger circle), except for Plant A where only the smaller batch size was used with 256 latent dimensions. The colour of the circles indicate the MSE - the lighter the color, the lower the MSE (i.e., the better the result), as indicated by the colormap at the right side.

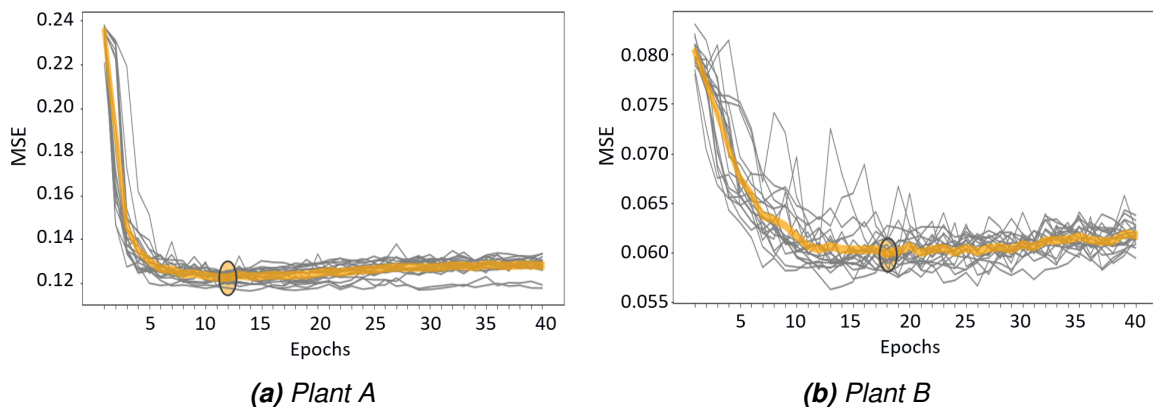


Figure 6. MSE on the validation set for (a) Plant A and (b) Plant B using the hyperparameters found in the grid search for the respective plants. Along the x-axis are the epochs, along the y-axis the MSE. The grey lines show the results of 20 single runs, which are averaged to the final result, shown in the thick orange line. The eclipse represents the optimal number of epochs.

Table 3. Trained LSTM-based neural networks: For both data sets, models were trained using the two optimal sets of hyperparameters from Section 3.2 (learning rate, latent dimensions, batch size, epochs)

		Hyperparameter set	
		Tuned on Plant A (0.005, 6, 512, 12)	Tuned on Plant B (0.005, 16, 265, 18)
Data	Plant A	Model_hypA_datA	Model_hypB_datA
	Plant B	Model_hypA_datB	Model_hypB_datB

based on the MSE. These represent the best available forecasts from our LSTM-based neural network. However, these results have to be interpreted with caution, as the forecasts are made on the test set which was also used to obtain the hyperparameters. Conversely, the other two models (*Model_hypB_datA* and *Model_hypA_datB*) serve to validate the generalizability of a single hyperparameter set to district heating plants for which they were not trained. Visual inspection of the results allows comprehension of model generalization on unseen data, demonstrating the model capabilities and facilitating comparison across different models. Further evaluation approaches, such as residual analysis, will be considered in subsequent investigations.

Figure 7 illustrate two representative examples of predictions for each of the four models. The top two rows show predictions for Plant A, once based on the same plant’s hyperparameters (a), and once based on the other plant’s hyperparameters (b). Similarly, the results for Plant B are illustrated in the bottom two rows, first based on the same plant’s hyperparameters (c), and once based on the other plant’s hyperparameters (d). In each plot, the blue line represents the heat load of the past 3 days, serving as input to the model. The solid green line represents the true heat loads of the next 3 days, while the dashed purple line represents our model predictions.

For (a) *Model_hypA_datA* and (c) *Model_hypB_datB*, the results are satisfactory: The predictions (dashed purple line) closely align with the true heat load values (green solid line). However, as mentioned above, it’s important to interpret these outcomes with caution, as the hyperparameters are inferred from the same dataset used for the forecasts, rather than from a separate validation set. For further validation, we examine the other two models: For (b) *Model_hypB_datA* the results remain satisfactory. The used model (based on the hyperparameters of Plant B) is more complex than the one based on the grid search on Plant A data, since it uses 16 instead of 6 latent dimensions. Overly complex models bear the risk of overfitting, but this concern does not appear significant in this case. Conversely, for (d) *Model_hypA_datB*, the results are less satisfactory, although the model still captures the overall trend. This discrepancy arises from the model’s lower complexity, compared to the model obtained from hyperparameter optimization on the Plant B data.

With respect to the initial objective in Section 2.3 of identifying robust candidate hyperparameter sets, these findings suggest a preference for hyperparameters that overfit the data. Given the longer training times associated with more complex models, a trade-off between the number of models to train (i.e., the number of hyperparameter candidate sets) and their complexity is expected. However, further investigation is required to explore optimal model configurations.

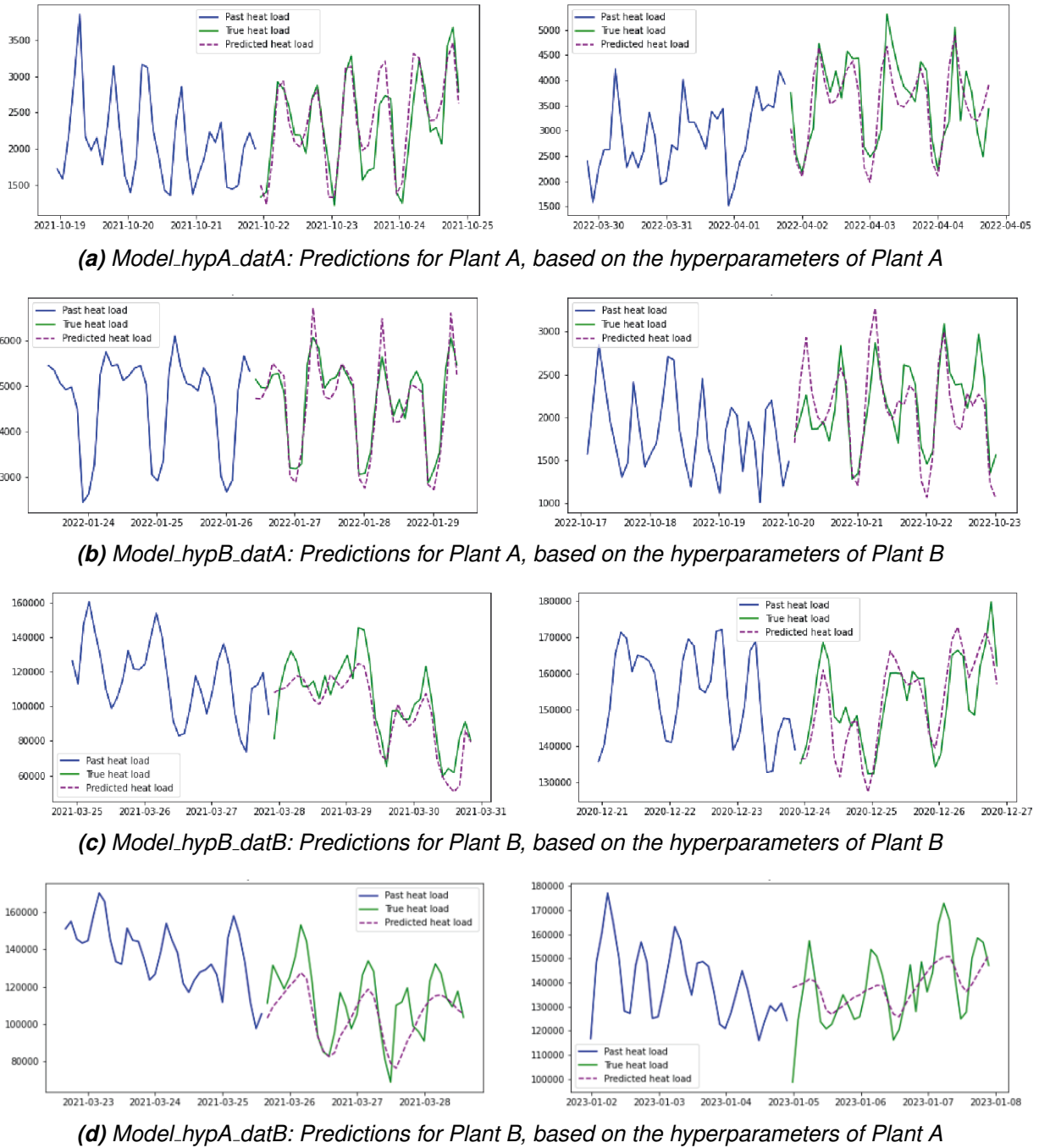


Figure 7. Results of the heat load prediction. In blue the network input (past heat loads), in green the true future heat loads and in dashed purple the predictions.

4 Predict-IT Platform

The Predict-IT platform is a web-based user interface to the LSTM-based neural network model, leveraging training the forecast model and producing heat load forecasts. The platform guides the user through the required steps, as illustrated in Figure 8:

- To set up a specific district heating plant, a few settings have to be entered by the user, e.g. plant location, and forecast horizon and frequency (e.g. forecasts for 12 hours or 3 days ahead, in 20-minutes or hourly intervals).
- Next, a measurement data file must be uploaded, with specified column separator or format. The measurement data are pre-processed in the background. In case

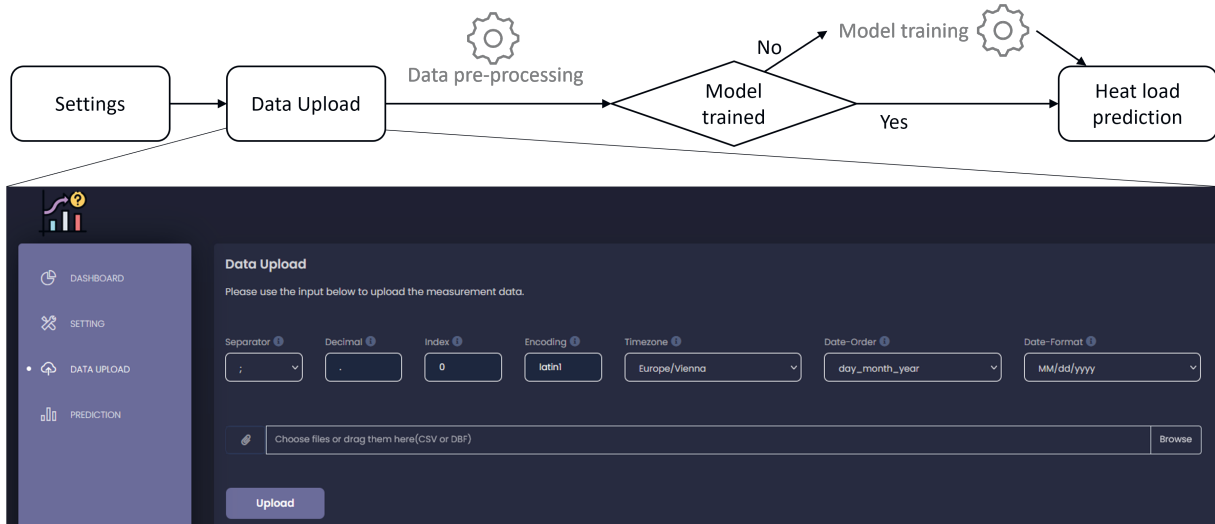


Figure 8. Usage diagram of the Predict-IT platform, showing the Data Upload page. The steps in grey font run in the background.

a pre-trained model for the plant is available, the heat load forecast is made and visualized in real-time.

- Finally, if the model has not been trained yet, model training is triggered in the background.

In the authors' opinion, there are three crucial points for widespread and intuitive applicability of the Predict-IT platform by plant operators:

1. **Data input:** Since the algorithm only requires historic time series of the total heat load and the ambient temperature, data handling for the plant operator is limited to providing these inputs which are typically available for heating networks. The ambient temperature can also be retrieved automatically by the algorithm, given the plant location. Specifically, for the data formats of two leading Austrian providers of plant control systems, automated data pre-processing has been implemented.
2. **Model training:** The training time of our algorithm depends on the model complexity (i.e., the hyperparameter values) and can range from minutes to several hours, such that the initial training could be done, e.g., overnight. Within the platform, two measures are taken to limit the training time: Pre-defined sets of hyperparameters can be used, based on previously executed experiments, and re-training strategies will be tested to evaluate when a complete retraining is needed (time-intensive) and when a fine-tune is enough (time-efficient). To improve training performance, model training can be performed on the computer's graphics card, leading to a significant speed-up in training time.
3. **Platform usage:** The Predict-IT application will be made available under an open-source software license (details to be determined), and the deployment and installation will be realised using Docker, leveraging an installation process suitable for various platforms and operating systems.

5 Discussion and future work

We presented "Predict-IT", an open-source web-based platform to generate forecasts of heat loads of district heating systems. The forecasts are powered by a state-of-the-art data-driven model. This model, a LSTM-based neural network, has been validated

against data of two Austrian local district heating networks. The model has shown to perform satisfactorily on the test data, delivering accurate heat load forecasts up to three days ahead, based on weather forecasts. The web-based Predict-IT platform builds the basis for providing straightforward heat load forecasts to plant operators.

To test the Predict-IT system in real life and to obtain feedback from an operator, we are planning to deploy the platform at a district heating plant. Furthermore, there are several directions of future work:

- Currently, manual data upload is required for the historic heat load data used in model training. The data upload could be automated and be done in real time, if plant control and plant setup allow to do so.
- In the forecasting algorithm, training and testing steps currently utilize historic ambient temperature instead of forecast data. Using forecasts introduces additional model uncertainty, necessitating further investigation into its impact on model performance. Ideally, historic ambient temperature measurements would be replaced with historic forecasts, an ongoing effort that involves identifying potential data providers.
- The MSE is currently used as model loss function during training, albeit known to be sensitive to residual peaks and not energy conservative. Exploring alternative energy-conserving loss measures, such as the mean absolute error (MAE), could be investigated.
- Limiting training time for the LSTM-based neural network is critical for the practical utility of the Predict-IT platform. This can be achieved in two ways: (1) Investigating suitable generalized hyperparameter sets in greater detail, considering cross-validation strategies and applying the model to diverse datasets, such as the Danish residential buildings dataset [10]. (2) Evaluating fine-tuning strategies to enable retraining the model solely on new data, thereby accelerating overall performance and providing usage guidelines for plant operators.
- We aim to conduct a more thorough model evaluation using residual analysis to gain deeper insights into model performance and potential areas for improvement.

6 Conclusion

This paper describes Predict-IT, a software platform designed to produce forecasts of the heat load of district heating networks, based on historic data and weather forecasts. The algorithm used to power the Predict-IT forecasts is a state-of-the-art LSTM-based neural network; the model and its training process are described in detail in this paper. With very few input data, Predict-IT produced promising heat load forecasts when tested with real data from two Austrian local district heating networks. Predict-IT is independent of the specific plant control system, and the software boasts user-friendly web-based utilization and installation (via Docker). The software will be accessible and usable also for commercial purposes under an open-source software license, fostering widespread accessibility and collaboration in the field.

Data availability statement

The data sets of the two local Austrian district heating networks (referred to as Plant A and Plant B in the article) are not publicly available to preserve the anonymity of the plant operators.

Table 4. Authors' contributions according to the *CreDIT* guidelines.

Author	Contribution
L. Bonal	Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft
M. Hamilton-Jones	Formal analysis, Methodology, Software
Z. Nasrollahinayeri	Methodology, Software
K. Dimovski	Formal analysis, Methodology, Project administration, Visualization
D. Entner	Conceptualization, Funding acquisition, Methodology, Project administration, Visualization, Writing – review & editing
P. Ohnewein	Conceptualization, Funding acquisition, Methodology, Writing – review & editing
H. Trinkl	Data curation, Funding acquisition, Investigation, Software

Underlying and related material

There is currently no underlying and related material available. However, it is planned to release the code of the Predict-IT platform under an open source software license in the near future.

Author contributions

Table 4 shows the authors' contributions according to the *CreDIT* guidelines.

Competing interests

The authors declare that they have no competing interests.

Funding

We thank the Austrian Cooperative Research (ACR) for funding this work within the project "PredictIT 2.0" (ACR project number SP-2022-05).

Acknowledgements

We thank the two local Austrian district heating networks for providing the data for our experiments.

References

- [1] M. Cui, "District heating load prediction algorithm based on bidirectional long short-term memory network model," *Energy*, vol. 254, p. 124 283, 2022. DOI: [10.1016/j.energy.2022.124283](https://doi.org/10.1016/j.energy.2022.124283).
- [2] L. M. Dang, J. Shin, Y. Li, *et al.*, "Toward explainable heat load patterns prediction for district heating," *Scientific Reports*, vol. 13, no. 1, p. 7434, 2023. DOI: [10.1038/s41598-023-34146-3](https://doi.org/10.1038/s41598-023-34146-3).
- [3] M. Abadi, P. Barham, J. Chen, *et al.*, "TensorFlow: A system for large-scale machine learning," in *Proceedings of the 12th USENIX conference on Operating Systems Design*

- and Implementation*, ser. OSDI'16, USA: USENIX Association, 2016, pp. 265–283, ISBN: 978-1-931971-33-1.
- [4] Python Software Foundation, "Python language reference," [Online]. Available: <http://www.python.org> (visited on 01/11/2024).
- [5] Django Software Foundation, "Django docs," [Online]. Available: <https://docs.djangoproject.com/> (visited on 01/11/2024).
- [6] Docker Inc., "Docker docs," [Online]. Available: <https://docs.docker.com/> (visited on 01/11/2024).
- [7] S. Grosswindhager, A. Voigt, and M. Kozek, "Online short-term forecast of system heat load in district heating networks," in *Proceedings of the 31st International Symposium on Forecasting*, Prague, Czech Republic, 2011.
- [8] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: LSTM cells and network architectures," *Neural Computation*, vol. 31, no. 7, pp. 1235–1270, 2019. DOI: [10.1162/neco_a_01199](https://doi.org/10.1162/neco_a_01199).
- [9] Keras, "KerasTuner," [Online]. Available: https://keras.io/keras_tuner/ (visited on 02/27/2024).
- [10] M. Schaffer, T. Tvedebrink, and A. Marszal-Pomianowska, "Three years of hourly data from 3021 smart heat meters installed in Danish residential buildings," *Scientific Data*, vol. 9, no. 1, p. 420, 2022, ISSN: 2052-4463. DOI: [10.1038/s41597-022-01502-3](https://doi.org/10.1038/s41597-022-01502-3).