Universitat Pompeu Fabra Barcelona

Departament | Escola d'Enginyeria

Deep Learning

Energy Consumption Forecasting with LSTM Models

P103 - Group L

Adrià Soria - 251729

Judit Viladecans - 251437

Paula Ceprián - 252503

INTRODUCTION

On April 28th, 2025, Spain experienced one of the most severe blackouts in its history, an unexpected power outage that disrupted homes, businesses, and essential services. As reported by several newspapers, this failure was triggered by a sudden surge in electricity demand that caught grid operators unprepared. The incident exposed the limitations of reactive management strategies and showed the urgent need for proactive systems capable of anticipating and preventing such crises.

In this context, our project aims to explore the power of real-time energy consumption forecasting using deep learning models. By accurately predicting household energy use based on recent patterns in reactive power, voltage, and intensity, our model aspires to do more than just forecast, it strives to prevent. Through better visibility of future consumption, these systems can help in demand shifting, early anomaly detection, and optimized energy distribution.

STATE OF THE ART

Energy consumption forecasting has become a highly active area of research, driven by the reasons previously mentioned. Numerous studies have focused on predicting electricity demand using real-world data collected from various regions across the world, from individual households in Europe to large-scale distribution systems in Asia and North America. This diversity of datasets has enabled researchers to explore a wide range of temporal resolutions and modeling challenges.

One of the earliest and most leading deep learning approaches to this problem has been the use of **Long Short-Term Memory (LSTM)** networks. These Recurrent Neural Networks (RNNs) are particularly well-suited for time series forecasting due to their ability to capture long-term dependencies and preserve temporal context across sequences. In a study conducted using over nine years of energy data from Spain, an optimized LSTM architecture was able to predict up to four hours ahead with less than 1.5% error, outperforming traditional models such as ARIMA and SVR [1]. Similarly, other researchers demonstrated that LSTM-based models achieved a daily error reduction of 11.2% and an hourly error reduction of 16.3% compared to statistical baselines [2]. These findings consolidated LSTM as a powerful tool for modeling the nonlinear and temporally dependent behavior of energy consumption.

As research progressed, it became evident that combining LSTM with additional architectural components could further enhance predictive performance, especially when dealing with **multivariate time series** containing variables such as reactive power, voltage, and current lecture date and time. **Convolutional Neural Networks (CNNs)**, particularly 1D convolutions (Conv1D), were introduced as a preprocessing step to capture short-term patterns and local dependencies in the data. These layers act as filters that smooth out noise and highlight relevant features before passing the sequence to the LSTM. In studies such as those by Khosravi et al. [3], CNN-LSTM combination showed better performance over standalone LSTM models, due to their feature extraction capabilities. Further improvements were reported when integrating external variables like real-time energy prices, suggesting that CNN-LSTM architectures are not only powerful but also flexible in adapting to more complex input dynamics [4].

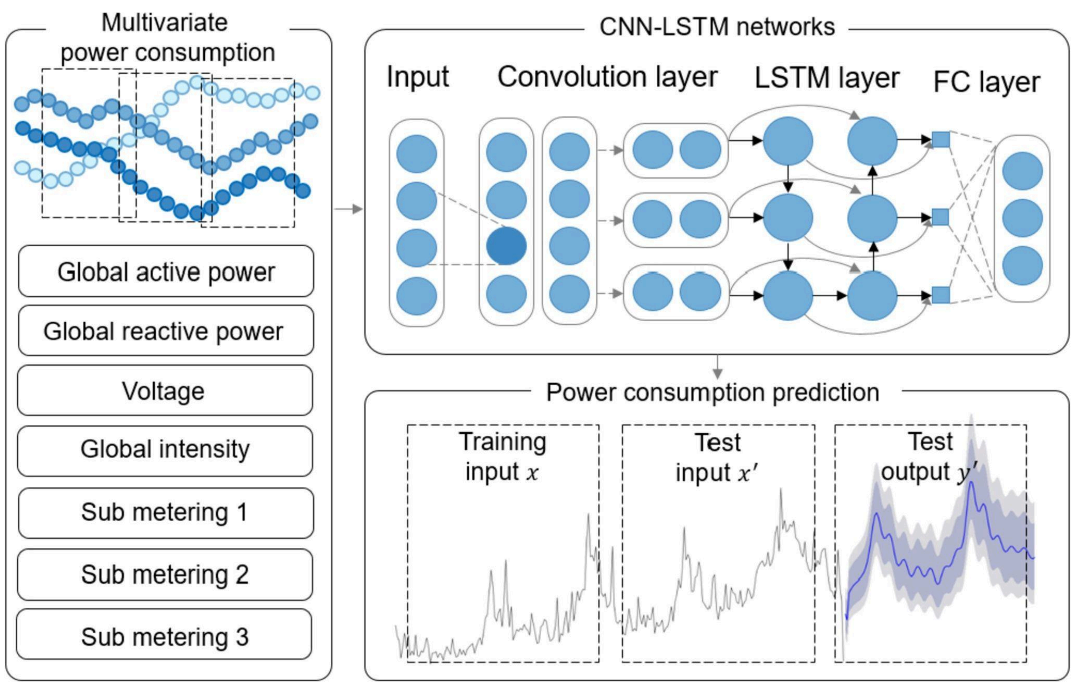

Figure 1. Proposed architecture for predicting residential power consumption using CNN-LSTM model [3]

In addition to convolutional layers, researchers have adopted **fully connected (dense) layers** following LSTM outputs. These layers serve as a mapping from the sequence-level feature space into a final predictive space. Combined with activation functions such as **ReLU or LeakyReLU**, and regularization techniques like **Dropout**, they help reduce overfitting and improve generalization, particularly when working with noisy datasets [5].

Recent reviews emphasize the implementation of hybrid architectures that combine Conv1D, LSTM, and Dense layers. These designs support the strengths of each component: Conv1D for local pattern detection, LSTM for temporal memory, and Dense for final prediction refinement. For instance, Wang et al. [6] conducted a comparative study of architectures including CNN-LSTM, concluding that such hybrids consistently exceed their ‘equivalents' in terms of robustness, precision, and computational efficiency.

In conclusion, the current state of the art reflects a clear progression from pure LSTM models to deep hybrid networks that effectively combine local pattern filtering, long-term memory modeling, and high-level abstraction.

METHODOLOGY

Data Analysis

The dataset used in this project has been provided by the UCI machine learning repository, and it includes over 2 million energy measurements collected from a single household in Sceaux, France, across nearly four years (December 2006 to November 2010) [7]. This multivariate time-series dataset offers a view of real-world household energy usage, considering different features such as reactive power and voltage.

Before starting our implementation, we analyzed our data and addressed issues. Firstly, we found missing values in a 23-day gap (August 2008), which we filled by smoothing data from temporally similar periods (August 2007). Then, unnecessary columns such as sub-metering variables were dropped due to unreliable values (large clusters of zeros or inconsistent readings). Finally, outliers and consumption peaks that did not reflect typical patterns were removed to avoid noise in training and improve model robustness.

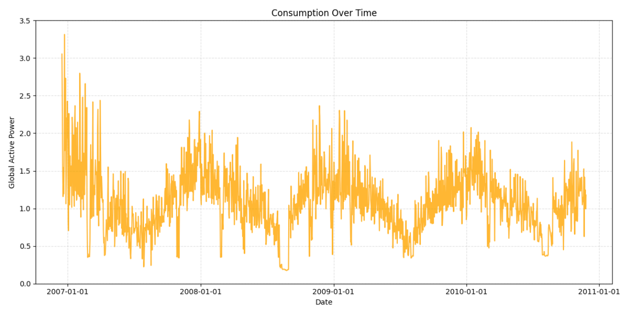

Figure 2. Raw Dataset

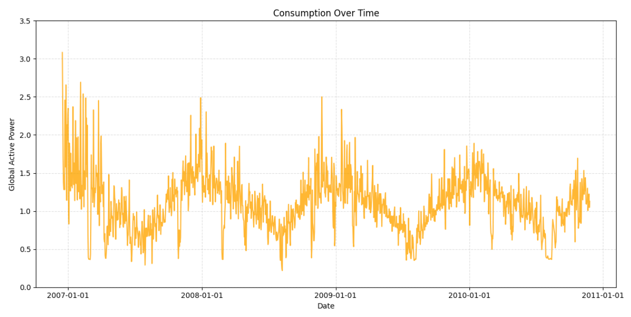

Figure 3. Cleaned Dataset

The cleaned dataset allowed us to observe meaningful consumption trends over time, helping us identify seasonal patterns and anomalies that informed both feature engineering and model selection. The preprocessing phase was as critical as model design, directly impacting the performance and reliability of our forecasting.

Models and Optimization

One key point on constructing a model is how well it performs. For that, it is important to choose appropriate metrics that will help us evaluate performance, errors and accuracy:

- RMSE (Root Mean Squared Error): How far our predictions are (on average) from the true values. Lower is better.
- R2 (Coefficient of Determination): How well the model captures the variability in the data. Higher is better.
- MAPE (Mean Absolute Percentage Error): It measures the mean relative error as a percentage. It is not an "accuracy" measure, but it is interpretive. Lower is better.
- Accuracy with relative tolerance: standalone "Accuracy" is typically used for classification, not regression. That is why here we need to use a variant of it, which is to define an acceptable tolerance and measure how many predictions fall within that range. Higher is better.

Our initial **baseline model** follows a traditional LSTM architecture, inspired by the work of Chousiadas [8], and includes two stacked LSTM layers separated by ReLU activation functions. It was selected for its capacity to capture long-term temporal dependencies in energy consumption sequences. Key hyperparameters included sequence length, future period prediction, number of epochs and batch size. Sequence length indicates how long of a preceding sequence to collect (in hours), whereas future period prediction shows how far into the future we are trying to predict (in hours).

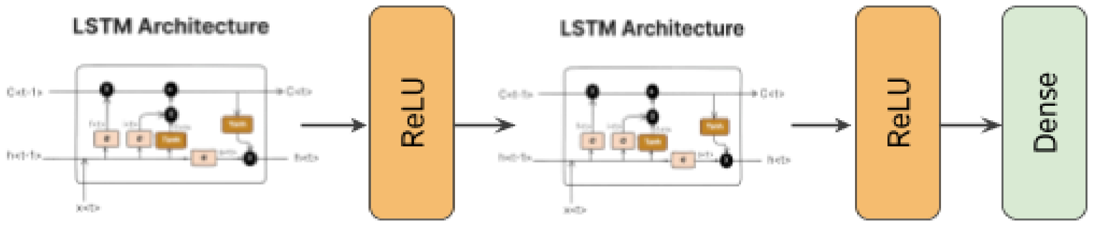

Figure 4. Baseline model architecture.

While functional, this model exhibited limitations such as potential overfitting at higher epochs and less flexibility in feature extraction. However, it served as a useful benchmark.

Our **second approach** consisted of adding a fully connected layer that improved the model's ability to extract meaningful features from the temporal representation. This simplified yet more stable architecture included one single LSTM layer, rather than two LSTMs, followed by a dense (fully connected) layer to extract features from the LSTM output. This setup was combined with a ReLU activation function and a Dropout layer to prevent overfitting.

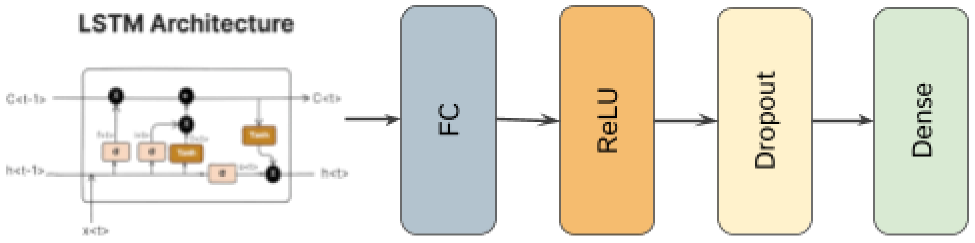

Figure 5. Second architecture approach.

This model proved easier to train and more generalizable, offering better performance by turning temporal dependencies into predictive features with lower risk of overfitting.

Our **final approach** and most successful model combined two causal Conv1D layers with LeakyReLU activations to extract short-term patterns and reduce noise, followed by an LSTM layer to capture long-term dependencies, and ended with a dense layer with LeakyReLU to transform features into the prediction space.

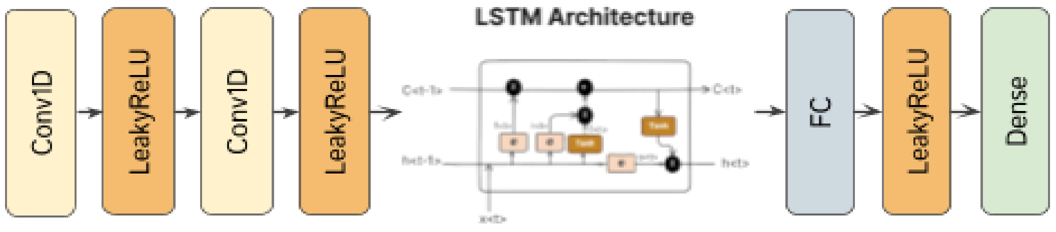

Figure 6. Third and final model architecture approach.

It demonstrated a strong ability to balance local and long-range temporal patterns, resulting in the highest accuracy (up to 95%). It also maintained robust gradient flow during training and effectively handled the variability present in energy consumption data.

Experiment 1: Multiple LSTMs with Hyperparameter Tuning

The first experiment served as our baseline implementation, which was intended to act as a benchmark for performance before introducing any architectural or methodological enhancements. To train the model, we used **Mean Absolute Error (MAE)** as the loss function, as it provides a direct measure of the average magnitude of errors without being overly sensitive to outliers. For splitting the data, we applied **TimeSeriesSplit**, a method specifically designed for sequential data that preserves the temporal order between training and validation sets.

To evaluate the initial effectiveness of the model, we trained it with the provided configuration and obtained the following results:

Seq. len	Future predict	Epochs	Batch	RMSE	R^2	MAPE	Accuracy
100	6	30	64	0.3349	0.4748	23.754%	29.28%%

These values indicate that the model struggled significantly with long-term forecasting. An R^2 value below 0.5 shows that the model captured less than half of the variability in the data, and a MAPE of nearly 24% reflects a high average error relative to actual consumption. Additionally, with less than 30% accuracy under a tolerance metric of 10%, the model's predictions were often not within acceptable bounds.

Given this poor performance, we decided to tune the model's hyperparameters to explore whether better configurations could achieve substantial improvements. We focused on the following hyperparameters:

- Sequence Length: We tested 50 and 100 to see if shorter memory windows could improve generalization.
- Prediction Horizon: We compared 1-hour vs. 3-hour ahead forecasting to evaluate the model's short- and mid-term forecasting ability.
- Epochs: We compared 20 and 50 to examine how training duration impacts convergence and overfitting.
- Batch Size: Fixed at 64 as it is a stable choice for time series LSTM models.

These choices allowed us to systematically study the model's behavior. By focusing on shorter prediction intervals and tuning the input sequence length, we aimed to optimize the baseline structure without altering the model's overall architecture. The results of these variations are discussed in the next section.

RESULTS AND DISCUSSION

As shown in Table 1, the model performed best when the prediction horizon was short (1 hour) and the sequence length was moderate (50–100 timesteps). For example, the configuration with sequence length 50, 50 epochs, and prediction horizon of 1 hour achieved an RMSE of 0.1048 and R^2 of 0.9483.

However, the performance decreased significantly for longer prediction horizons (3 hours), where the RMSE nearly doubled, and accuracy dropped below 45%. This degradation suggests that while LSTM networks are capable of short-term forecasting, stacked configurations may lead to overfitting and poor generalization, especially when forecasting further into the future.

Seq. len	Future predict	Epochs	Batch	RMSE	R^2	MAPE	Accuracy
50	**1**	**20**	**64**	**0.1068**	**0.9463**	**7.04%**	**75.18%**
50	**1**	**50**	**64**	**0.1048**	**0.9483**	**6.66%**	**76.28%**
50	3	20	64	0.2306	0.7500	16.93%	40.27%
50	3	50	64	0.2366	0.7367	15.81%	42.96%
100	**1**	**20**	**64**	**0.1080**	**0.9454**	**6.88%**	**76.14%**
100	**1**	**50**	**64**	**0.1111**	**0.9422**	**7.39%**	**75.52%**
100	3	20	64	0.2363	0.7387	16.03%	42.03%
100	3	50	64	0.2339	0.7438	15.99%	42.67%

Table 1. Performance evaluation of the baseline model with hyperparameter tuning

In conclusion, this baseline model served as a reference point for further improvements, highlighting the need for more stable architectures and better regularization strategies.

Experiment 2: One LSTM with Fully Connected Layers

After analyzing the results of the previous experiment, we observed that the baseline model struggled with generalization, especially when forecasting several hours into the future. Its stacked LSTM architecture did not consistently have a good performance and often showed high error rates and unstable training. In response, we decided to simplify the model.

This second experiment implemented a simpler architecture with only a **single LSTM layer**, followed by a **fully connected (dense) layer**, a **ReLU activation**, and a **Dropout layer**. The goal was to maintain the model's ability to learn temporal dependencies with LSTM while enhancing its generalization capacity and reducing the risk of overfitting.

Given that the previous results indicated that shorter forecast horizons (1 hour) performed much better than longer ones (6 hours), we selected **prediction intervals of 1, 3, and 6 hours** to analyze this effect. Additionally, we retained the **sequence lengths of 50 and 100 timesteps** to continue exploring how much historical data improves the predictions.

To ensure consistency and comparability, we fixed other training parameters like the number of epochs (30 as it is a good balance between convergence and overfitting) and batch size (64 based on common definition in time series forecasting). This experimental setup allowed us to evaluate whether reducing architectural depth and introducing regularization could improve performance while maintaining lower complexity.

RESULTS AND DISCUSSION

The results (Table 2) show a marked improvement across all metrics, especially in the short-term horizon (1 hour). The best performance was achieved with a sequence length of 100 and a 1-hour forecast window, with an RMSE of **0.0547**, an R² of **0.9789**, and an accuracy of **93.48%**. These values represent a significant improvement over the initial model.

Seq. len	Future predict	Epochs	Batch	RMSE	R^2	MAPE	Accuracy
50	**1**	**30**	**64**	**0.0551**	**0.9784**	**3.94%**	**91.86%**
50	3	30	64	0.1266	0.8858	8.69%	65.35%
50	6	30	64	0.1982	0.7211	14.35%	48.06%
100	**1**	**30**	**64**	**0.0547**	**0.9789**	**3.91%**	**93.48%**
100	3	30	64	0.1328	0.8754	9.37%	64.47%
100	6	30	64	0.1911	0.7420	13.83%	49.21%

Table 2. Performance evaluation of the second model with different values for sequence length and prediction horizon

Although the accuracy decreased for longer prediction horizons, this model still outperformed the baseline under every setting. The use of Dropout also helped reduce overfitting, making the model more robust and easier to train. Overall, this architecture proved that reducing complexity and applying proper regularization can lead to better performance, especially when forecasting in the short term.

Experiment 3: LSTM with CNN and Fully Connected Layers

Following the results of the second experiment, we observed a significant improvement in forecasting accuracy and generalization compared to the baseline. However, the model still fell slightly short of our **target accuracy of 95%** for short-term forecasting. In particular, although we achieved high accuracy (~93%), we identified room for improvement in the model's ability to extract short-term temporal patterns from noisy input data.

To address this, we introduced a **hybrid architecture** that could handle both **short-term variations** and **long-term dependencies** in energy consumption. Convolutional layers are particularly effective at identifying localized patterns and filtering noise, making them ideal as a preprocessing stage before the LSTM.

The final model architecture consisted of **two Conv1D layers** to detect local temporal features and smooth the input signal, **LeakyReLU activation functions** after each Conv1D, a single **LSTM layer** to model sequential dependencies in the data and **a dense output layer with LeakyReLU**. This structure was designed to complement the strengths of the previous model by adding a local pattern extraction phase.

We kept the same evaluation framework: constant number of epochs of 30, batch size of 64, sequence length of 50 and 100 timesteps and prediction horizons between 1 and 6 hours.

RESULTS AND DISCUSSION

This hybrid model demonstrated the **best overall performance**, particularly for short-term forecasting. The best configuration (sequence length 50, prediction horizon 1 hour) achieved an RMSE of **0.0525**, an R^2 of **0.9804**, and an accuracy of **95.00%** (see Table 3). These results confirm that the Conv1D layers significantly improved the model's capacity to extract meaningful short-term patterns, which were then effectively processed by the LSTM layer.

Performance remained competitive even for longer horizons, although, as expected, accuracy decreased as the prediction window widened. Still, compared to previous models, this architecture consistently outperformed in both precision and robustness.

Seq. len	Future predict	Epochs	Batch	RMSE	R^2	MAPE	Accuracy
50	**1**	**30**	**64**	**0.0525**	**0.9804**	**3.61%**	**95.00%**
50	3	30	64	0.1341	0.8722	8.91%	65.44%
50	6	30	64	0.2180	0.6626	14.45%	46.79%
100	**1**	**30**	**64**	**0.0533**	**0.9799**	**3.71%**	**94.69%**
100	3	30	64	0.1278	0.8846	8.88%	66.35%
100	6	30	64	0.2193	0.6603	14.54%	47.61%

Table 3. Performance evaluation of the third model with different values for sequence length and prediction horizon

In summary, this experiment supports the hybrid approach combining convolutional and recurrent layers, which exploits the strengths of both architectures.

FINAL CONCLUSIONS

This project set out with a clear and ambitious goal: to leverage deep learning for real-time residential energy consumption forecasting, motivated by the critical vulnerability exposed by Spain's blackout on April 28th, 2025. Through a combination of robust data preprocessing, careful model architecture design, and rigorous experimentation, we successfully demonstrated the viability and value of predictive systems in anticipating household power usage patterns.

Our work reaffirmed the strengths of LSTM-based models in capturing temporal dependencies in energy consumption data. However, we found that stacked LSTM architectures, while powerful, also were prone to overfitting and performed inconsistently, particularly when forecasting multiple hours into the future. By simplifying the architecture to a single LSTM layer and incorporating dense layers with Dropout regularization, we significantly improved both model stability and generalization, achieving much lower error rates and higher forecast accuracy across various time horizons.

The final and most effective architecture: hybrid CNN-LSTM model with dense layers, capitalized on the individual strengths of its components. The causal Conv1D layers extracted local temporal patterns and filtered noise, the LSTM layer modeled long-term dependencies, and the dense layers mapped learned features to accurate consumption predictions. This model achieved the highest performance, with up to **95% accuracy**, a **MAPE as low as 6.66%**, and an **R^2 score above 0.94** for short-term (1-hour) forecasts, demonstrating its potential for real-world application in energy monitoring systems.

From a broader perspective, our results support the growing consensus in the literature that hybrid architectures, especially those combining convolutional and recurrent layers, offer a powerful, flexible foundation for multivariate time series forecasting.

Ultimately, this project contributes to the ongoing shift from reactive to proactive energy management. By providing more accurate short-term forecasts, these deep learning systems can serve as the foundation for smarter grids, helping to prevent crises like the one that motivates us to work on that project. While challenges remain, particularly in scaling models to handle real-time data from diverse households and integrating external variables like weather or pricing signals, the path forward is clear: deep learning holds strong promise in enabling a more resilient, efficient, and intelligent energy future.

REFERENCES

The following references were consulted to better understand some theoretical differences, address methodological doubts, and explore the functionalities of several architectures and forecasting techniques that have been implemented and evaluated throughout this project:

1. Liu, Y., Xie, J., Wu, J., Chen, Z., & Zhang, B. (2023). Hybrid deep learning framework for short-term energy forecasting in Spain using LSTM optimized by coronavirus algorithm. *Big Data Mining and Analytics*. https://www.sciopen.com/article/10.26599/BDMA.2023.9020030

2. Deb, C., Zhang, F., Yang, J., Lee, S. E., & Shah, K. W. (2017). A review on time series forecasting techniques for building energy consumption. *Energy and Buildings, 134*, 33–45. https://www.sciencedirect.com/science/article/pii/S0360544216315006

3. Khosravi, A., Nahavandi, S., Creighton, D., & Atiya, A. F. (2019). Multivariate residential electricity consumption prediction using CNN-LSTM neural networks. *Energy, 169*, 1046–1060. https://www.sciencedirect.com/science/article/pii/S0360544219311223

4. Wang, X., Chen, H., & Yang, J. (2019). Short-term load forecasting with multi-scale CNN-LSTM architecture considering electricity price signals. *Sustainable Cities and Society, 49*, 101612. https://www.sciencedirect.com/science/article/pii/S2210670719335413

5. Che, J., Xu, C., Wang, Z., & Wang, H. (2021). Improved short-term load forecasting based on LSTM and attention mechanism. *Applied Mathematical Modelling, 95*, 345–361. https://www.sciencedirect.com/science/article/pii/S0378475421001774

6. Heidari, M., Ghasemi, M., Ghadimi, N., & Guerrero, J. M. (2021). Electric load forecasting using hybrid deep learning model based on CNN, LSTM, and TCN. *Neural Computing and Applications*. https://link.springer.com/article/10.1007/s00521-021-06773-2

7. UCI Machine Learning Repository. (n.d.). *Individual household electric power consumption dataset*. https://archive.ics.uci.edu/dataset/235/individual+household+electric+power+consumption

8. Chousiadas, C. (n.d.). *Forecasting energy consumption using LSTM*. GitHub. https://github.com/Housiadas/forecasting-energy-consumption-LSTM