university of groningen

faculty of science and engineering

Nordic Wind Power Forecasting

Bogdan Sandoiu(S5549329)

Abstract

Accurate short-term wind power forecasting is essential for electricity markets with a high share of renewable energy. This paper addresses the task of forecasting hourly wind power production up to two days ahead for multiple Norwegian bidding zones, following the data availability constraints of the WindAI challenge. The dataset includes historical wind power production, observed meteorological variables, and numerical weather prediction forecasts.

An end-to-end forecasting pipeline is developed with strict temporal alignment and explicit prevention of information leakage. Several model families are evaluated, including statistical time series models (SARIMA and SARIMAX), recurrent neural networks based on gated recurrent units (GRUs), and Transformer-based architectures. While statistical and recurrent models provide reasonable baselines, Transformer models consistently achieve lower forecast error, particularly at longer prediction horizons, due to their ability to capture long-range temporal dependencies through attention mechanisms.

In addition to point forecasts, predictive uncertainty is quantified using an ensemble of Transformer models to construct calibrated prediction intervals. Overall, the results demonstrate that two-day-ahead wind power forecasting is feasible under realistic operational constraints and that attention-based models offer clear advantages over recurrent architectures when combined with appropriate preprocessing and feature engineering.

Contents

1 Introduction

Wind power plays an important role in modern electricity systems, especially in regions such as the Nordic countries where wind energy penetration is high. However, wind power production is highly variable and difficult to predict, which creates challenges for power system operation and electricity market planning. Transmission system operators therefore rely on short-term forecasts to schedule production and maintain grid stability.

This work is conducted in the context of the WindAI challenge. The objective is to forecast hourly wind power production up to two days ahead for the Norwegian bidding zones NO1 to NO4, as illustrated in Figure 1. Forecasts must be generated using only information that would be available at prediction time, including historical power production, observed weather data, and numerical weather prediction forecasts.

1.1 Motivation and challenges

Short-term wind power forecasting presents several challenges. First, wind power time series are non-stationary due to seasonal patterns, changing weather conditions, and increases in installed capacity over time. Second, forecast accuracy decreases with the prediction horizon because weather forecasts become more uncertain further into the future. Third, different bidding zones exhibit different wind regimes, which makes generalization across regions difficult.

Recent deep learning models, particularly Transformer architectures, have shown strong performance in sequence modeling tasks. These models use attention mechanisms to capture long-range dependencies. However, previous studies have shown that Transformers do not always outperform simpler models in time series forecasting, especially when strong seasonal patterns or limited data are present [1]. This raises the question of whether the added model complexity is justified for operational wind power forecasting.

1.2 Contributions

This paper investigates the effect of model complexity and attention mechanisms on two-day-ahead wind power forecasting. The main contributions are:

- An end-to-end forecasting pipeline that follows the WindAI data availability constraints.
- A clear comparison between statistical models, recurrent neural networks, and Transformer-based models.
- Experimental evidence that Transformers outperform recurrent neural network baselines.
- An uncertainty estimation approach that provides prediction intervals alongside point forecasts.

2 Data

2.1 Dataset overview

This work uses the WindAI dataset, which contains hourly wind power production and meteorological information for the Norwegian bidding zones NO1 to NO4. The dataset is designed to reflect a realistic operational forecasting setting, where only information available at the forecast issuance time can be used.

Wind power production is provided as aggregated hourly values per bidding zone. Meteorological data are available at wind park level and include both observed weather variables (now-casting) and numerical weather prediction forecasts generated by the Meteorological Ensemble Prediction System (MET MEPS) operated by the Norwegian Meteorological Institute. Metadata describing wind park locations and bidding zone mappings are also provided. All timestamps are given in UTC.

Table 1 summarizes the different data sources used in this work. Figure 1 shows the geographical location of the Norwegian bidding zones considered in the study.

Table 1: Overview of datasets used in the WindAI challenge

Dataset	Description
Wind power production	Hourly aggregated wind power generation per bidding zone (NO1–NO4), measured in megawatts.
Now-casting weather data	Observed weather variables at wind park level, including wind speed, wind direction, temperature, pressure, humidity, and precipitation, available up to forecast time.
Weather forecasts	Numerical weather prediction forecasts from the MET MEPS system, providing weather variables up to 65 hours ahead.
Wind park metadata	Information about wind park locations, bidding zone assignments, and operational properties.

Figure 1: Norwegian power market bidding zones considered in the WindAI challenge.

2.2 Temporal splitting and leakage prevention

All experiments use a strict chronological split into training, validation, and test sets. The split is based only on time and does not involve random shuffling. For a forecast issued at time t, only data available at or before time t are used as model inputs.
Any statistics used for feature scaling are computed using the training set only and then applied unchanged to the validation and test sets. This prevents information from the future from leaking into the models and ensures a realistic evaluation.

2.3 Feature engineering for deep learning

Feature engineering plays an important role in the deep learning pipeline. The features described in this subsection are used only by deep learning models. Statistical models rely on raw or minimally processed wind power time series.

Wind direction encoding. Wind direction is a circular variable, meaning that values near 0 degrees and 360 degrees represent similar physical directions. To avoid discontinuities, wind speed and wind direction are converted into two horizontal wind components.
For a given wind speed and wind direction, the components are computed as:

$$W_x = \text{speed} \cdot \cos(\text{direction}),$$

$$W_y = \text{speed} \cdot \sin(\text{direction}),$$

where the direction is expressed in radians. This transformation is applied to both observed and forecast wind variables. The original wind speed and direction values are then replaced by the corresponding components.

Cyclic time features. Daily and yearly seasonal patterns are represented using cyclic time encodings. Four time-based features are added:

$$\text{Day}_{\text{sin}} = \sin\left(\frac{2\pi \cdot t}{\text{one day}}\right), \quad \text{Day}_{\text{cos}} = \cos\left(\frac{2\pi \cdot t}{\text{one day}}\right),$$

$$\text{Year}_{\text{sin}} = \sin\left(\frac{2\pi \cdot t}{\text{one year}}\right), \quad \text{Year}_{\text{cos}} = \cos\left(\frac{2\pi \cdot t}{\text{one year}}\right).$$

These features allow the models to learn smooth daily and yearly seasonality without introducing artificial jumps between consecutive time steps.

Lagged wind power features. To provide short-term historical context, lagged values of wind power production are included. For each time step t, lag features are created as:

$$y(t-1),\ y(t-2),\ \ldots,\ y(t-7),$$

where $y(t)$ denotes the wind power production at time t. These lag features help the models capture recent trends and autocorrelation in the data.

Weather ensemble summary features. Weather forecasts are provided by the Meteorological Ensemble Prediction System (MET MEPS), which generates multiple forecast realizations for each weather variable at every forecast issuance time. Each ensemble member predicts future values of wind speed, wind direction, temperature, pressure, humidity, and wind gusts for multiple lead times.
To transform these ensemble forecasts into fixed-size inputs suitable for deep learning models, simple statistical summaries are computed across ensemble members. For each weather variable and forecast lead time, the ensemble mean and standard deviation are calculated. For example, the mean wind speed at time t is computed as

$$\text{mean}(t) = \frac{1}{N}\sum_{i=1}^{N} v_i(t),$$

where $v_i(t)$ denotes the wind speed predicted by ensemble member i, and N is the total number of ensemble members. The corresponding standard deviation is computed as

$$\text{std}(t) = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (v_i(t) - \text{mean}(t))^2}.$$

For wind direction, the same encoding strategy described earlier is applied before computing ensemble statistics. Specifically, wind speed and wind direction from each ensemble member are first converted into horizontal wind components using sine and cosine transformations. The ensemble mean and standard deviation are then computed separately for these components. This ensures consistency between observed and forecast wind representations and avoids discontinuities at the 0°/360° boundary.
The resulting ensemble summary features capture both the expected future weather conditions through the mean and the

forecast uncertainty through the ensemble spread. All ensemble statistics are computed using only information available at the forecast issuance time and are used exclusively as input features for the deep learning models.

2.4 Model-specific preprocessing

Different preprocessing pipelines are used for statistical models and deep learning models due to their different modeling assumptions.

2.4.1 Preprocessing for statistical models

SARIMA, and SARIMAX models operate on univariate or low-dimensional time series and therefore use a deliberately simple preprocessing pipeline. Hourly wind power data are reshaped from wide to long format and aggregated per bidding zone by summing production across all wind parks, yielding a single continuous time series for each zone.
To capture deterministic seasonality, cyclic time features are constructed from the timestamp. The same sine and cosine encoding technique used for feature engineering in the deep learning models is reused here to represent daily and yearly seasonal patterns in a smooth and continuous manner. These cyclic features are included as optional exogenous regressors for the SARIMAX model, while SARIMA relies solely on the wind power series.
After aggregation, the data are split by bidding zone and ordered chronologically. Missing values are removed, and only numerical inputs required by each model are retained. Stationarity is handled internally through differencing and seasonal differencing when required, and no normalization or scaling is applied. All inputs are strictly aligned in time to prevent information leakage.

2.4.2 Preprocessing for deep learning models

Deep learning models operate on multivariate inputs and use the engineered features described above. Weather variables are first aggregated from wind park level to bidding zone level by computing the mean across all wind parks within each zone.
Feature construction is performed separately for the training, validation, and test splits. To preserve causality at split boundaries, lagged features for the validation and test sets are computed using a short history from the preceding split. After feature construction, numerical features are standardized using statistics computed on the training set only. For a feature x, normalization is performed as

$$x_{\text{norm}} = \frac{x - \mu_{\text{train}}}{\sigma_{\text{train}}},$$

where μ_{train} and σ_{train} denote the mean and standard deviation computed on the training data. The same transformation is then applied unchanged to the validation and test sets.
Cyclic time features based on sine and cosine encodings are not standardized, as their values are already bounded within the interval $[-1, 1]$.

2.5 Window construction

The processed time series is converted into supervised learning samples using sliding windows. A window represents a fixed-length snapshot of the recent past that the model observes in order to make a forecast.
For a bidding zone z and time t, the input window of length L is defined as:

$$X(z,t) = \{f(z,t-L+1), \dots, f(z,t)\},$$

where $f(z,t)$ contains the standardized weather variables and engineered features available at time t. In practice, this window summarizes the evolution of weather conditions and power production over the previous L hours and serves as the model input.
The corresponding target is a multi-step forecast horizon:

$$Y(z,t) = \{y(z,t+1), \dots, y(z,t+H)\},$$

which contains the true wind power production for the next H hours following time t.
Windows that overlap dataset boundaries or contain missing values are discarded to ensure consistency and causality. The same windowing strategy is used for all deep learning models so that differences in performance are attributable to model architecture rather than data preparation.

3 Methods

This section describes the forecasting models used in this work and explains how each architecture operates. All models are evaluated using identical temporal splits, input windows, and forecast horizons to ensure a fair comparison.

3.1 Problem formulation

The forecasting task is a multi-step prediction problem. Given all data available up to time t, the objective is to predict hourly wind power production for the next H hours:

$$\hat{y}(t+1), \hat{y}(t+2), \dots, \hat{y}(t+H).$$

Forecasts are generated independently for each Norwegian bidding zone. Statistical models operate directly on the wind power time series, while deep learning models use multivariate input windows containing wind power, weather variables, and engineered features described in the Data section.

3.2 Statistical baseline models

Statistical models are included as reference baselines. They rely on linear assumptions and fixed temporal structures.

SARIMA. SARIMA extends the classical Autoregressive Integrated Moving Average (ARIMA) model, which represents the current value of a time series as a linear combination of past observations and past forecast errors after differencing to ensure stationarity. While ARIMA captures short-term temporal dependencies, it does not explicitly model seasonal patterns. SARIMA addresses this limitation by incorporating a seasonal autoregressive and moving-average structure with a fixed seasonal period of 24 hours, corresponding to daily cycles in wind power production. In this work, SARIMA is implemented using the SARIMAX framework without exogenous variables. The selected configuration combines a non-seasonal order $(2,1,2)$ with a seasonal order $(1,1,1,24)$, allowing the model to capture both intra-day dynamics and daily repetition. As with ARIMA, forecasts are generated directly for the next H hours and evaluated using RMSE.

SARIMAX. SARIMAX further extends SARIMA by incorporating external explanatory variables that are known at forecast issuance time. In this project, sinusoidal time features representing daily and yearly cycles are included as exogenous regressors. These variables provide the model with explicit information about periodic structure that may not be fully captured by autoregressive terms alone. The autoregressive and seasonal orders are kept identical to those used in the SARIMA model to ensure a fair comparison. By combining past wind power observations with time-based external information, SARIMAX aims to improve forecast accuracy while maintaining a relatively simple and interpretable structure.

3.3 Deep learning models

Deep learning models use an input window of length $L = 336$ hours and predict an output horizon of length $H = 48$. The input features include weather variables and engineered features (time encodings, wind components, and lag features) as described in Section 2.3.
To keep the comparison fair, the same loss function is used across all deep learning models, namely mean squared error:

$$\mathcal{L} = \frac{1}{H}\sum_{h=1}^{H}\left(y(t+h)-\hat{y}(t+h)\right)^2.$$

Even if some early code versions used a custom loss, the final experiments reported in this paper use Mean Squared Error(MSE) for all deep learning models.

3.3.1 Recurrent baselines and transition to Transformers

The recurrent modeling approach was explored in several stages using GRU-based architectures. We initially trained a high-capacity GRU model (GRU_Deep) with the goal of capturing complex temporal dependencies. Although this model achieved very low training error, it exhibited strong overfitting, with validation performance degrading quickly. This indicates that the model capacity was too large relative to the effective information content of the data, leading to poor generalization. The architectural details of this model, as well as all other neural architectures used in this work, are summarized in Table 2.

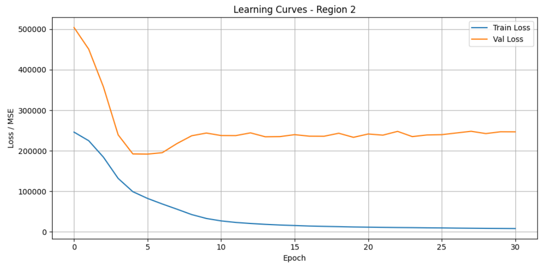

Figure 2: Training and validation loss curves for the GRU_Deep model, illustrating rapid overfitting despite strong optimization of the training objective.

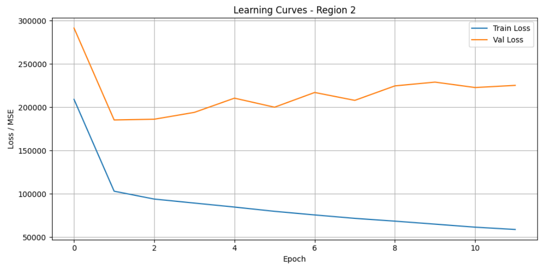

Figure 3: Training and validation loss curves for the GRU_Weak model, showing more stable convergence and reduced overfitting compared to the higher-capacity GRU_Deep architecture.

To mitigate this issue, a lower-capacity recurrent model (GRU_Weak) was introduced. By reducing the number of parameters and applying stronger regularization, this model showed more stable training dynamics and improved validation performance.
For this reason, GRU_Weak is retained as the primary recurrent baseline in this study (see Table 2 for the corresponding architecture).
Despite these improvements, both GRU-based models consistently tended to converge toward predicting the mean or median level of the wind power signal. In particular, the recurrent models struggled to learn meaningful temporal structure such as sharp ramps, peaks, and seasonal patterns, instead favoring smooth average forecasts. This behavior suggests that, under the given data regime and training setup, recurrent architectures were unable to fully exploit long-range dependencies and complex interactions in the input features.
The observed limitations of recurrent models motivated the exploration of Transformer-based architectures. Transformers were evaluated using two decoder configurations: a masked (causal) decoder and an unmasked decoder, both described in Table 2. This comparison allows us to assess the effect of enforcing strict temporal causality in the decoder. As shown in the Results section, Transformer models substantially outperform the recurrent baselines, with the unmasked variant achieving the best overall forecasting performance.

Table 2: Summary of deep learning architectures (input length $L = 336$, forecast horizon $H = 48$).

Design choice	**GRU_Weak**	**GRU_Deep**	**Transformer**
Input	(L,F) window	(L,F) window	Encoder: (L,F), Decoder: $(H,1)$
Input regularization	GaussianNoise(0.1)	GaussianNoise(0.1)	None
Core layers	GRU 256 $\rightarrow$ 128 $\rightarrow$ 64 (seq)	GRU 512 $\rightarrow$ 512 $\rightarrow$ 512 $\rightarrow$ 256 $\rightarrow$ 128 (seq)	2 encoder blocks + 2 decoder blocks
Normalization	LayerNorm after each GRU	LayerNorm after each GRU	LayerNorm inside each block
Dropout	0.4, 0.3, 0.2 (+ head 0.3)	0.2 after each GRU	0.1 inside attention blocks
Regularization	L2 on GRU (10^{-3})	L2 on GRU (10^{-4})	AdamW weight decay (10^{-5}) if available
Horizon handling	Slice last H steps, then dense head	Slice last H steps, then dense head	Teacher forcing during training
Output head	TD Dense(64, ReLU) $\rightarrow$ Dense(32, ReLU) $\rightarrow$ Dense(1)	TD Dense(16, ReLU) $\rightarrow$ Dense(1)	TD Dense(1)
Attention masking	Not applicable	Not applicable	Decoder self-attention masked or unmasked

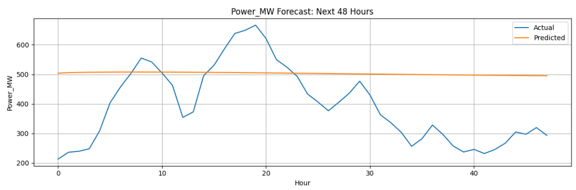

Figure 4: Ground truth versus predicted wind power for the GRU_Weak model over the forecast horizon. The predictions exhibit a strong bias toward average production levels, with limited responsiveness to abrupt changes in the true signal.

3.4 Training procedure

All deep learning models are trained on the training split and validated on the validation split using mini-batches of size 64. Early stopping is applied based on validation loss to prevent overfitting, and ReduceLROnPlateau is used to adapt the learning rate when validation performance stops improving. The best-performing model is saved using validation-based checkpointing and subsequently evaluated on the test split.
GRU-based models are optimized using the Adam optimizer with gradient clipping (clip norm = 1.0). The learning rate is set to 2×10^{-4} for the GRU_Weak model and 1×10^{-3} for the GRU_Deep model. L2 regularization is applied to all GRU layers, with stronger regularization used for the lower-capacity GRU_Weak architecture.
Transformer models are trained using the AdamW optimizer when available, with a learning rate of 1×10^{-3} and weight decay of 10^{-5}. Training employs teacher forcing, where the decoder input is constructed by shifting the ground-truth output sequence by one time step. Learning-rate scheduling and early stopping are applied consistently across all Transformer variants.
All models are trained for a maximum of 100 epochs, with training terminated early if validation loss does not improve for a predefined number of epochs.

3.5 Uncertainty estimation using an ensemble of Transformers

Uncertainty is estimated only for the Transformer model because it achieved the best forecasting performance in this project. Uncertainty is computed using an ensemble of $N = 10$ Transformer models trained with different random seeds. The ensemble mean is:

$$\hat{y}_{\text{mean}}(t) = \frac{1}{N}\sum_{i=1}^{N}\hat{y}_i(t),$$

and the ensemble standard deviation is:

$$\text{std}(t) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\hat{y}_i(t) - \hat{y}_{\text{mean}}(t)\right)^2}.$$

Calibration. A single scaling factor is tuned on the validation set by grid search to improve calibration. The standard deviation is multiplied by this factor.

Prediction intervals. Two uncertainty bands are reported: parametric bands using $\pm 1 \cdot \text{std}(t)$ and $\pm 2 \cdot \text{std}(t)$ around the ensemble mean, and non-parametric bands using ensemble quantiles (for example 90% and 95%).

4 Results

This section presents the forecasting results on the held-out test set. To keep the presentation concise, results are shown only for the first Norwegian bidding zone (NO2). All models are evaluated using identical temporal splits, input windows, and a fixed 48-hour forecast horizon. Forecast quality

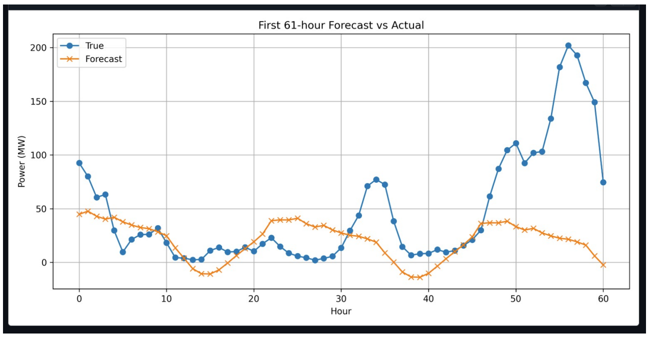

Figure 5: Forecast produced by the SARIMA model for bidding zone NO2.

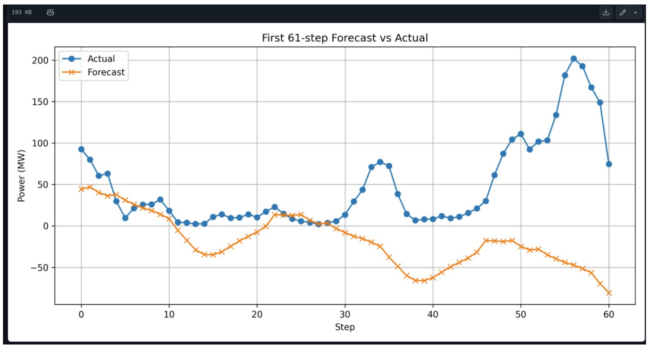

Figure 6: Forecast produced by the SARIMAX model for bidding zone NO2.

is assessed quantitatively using the root mean squared error (RMSE) and qualitatively through visual inspection of forecast trajectories.
It should be noted that the ground-truth trajectories shown across different figures are not always identical. This is due to the use of shuffled mini-batches during inference and visualization for the deep learning models, meaning that individual plots correspond to different test windows within the same bidding zone. All quantitative metrics reported in this section are computed over the full test set and are therefore directly comparable across models.

4.1 Statistical baseline models

SARIMA, and SARIMAX serve as reference statistical baselines. Figure 5 and Figure 6 illustrate representative forecasts produced by these models for bidding zone NO2. The predictions are generally smooth and struggle to capture rapid changes in wind power production. In particular, sharp ramps and sudden drops are underestimated, and the forecasts tend to revert toward a mean level.
The test-set RMSE values for the statistical models are:
- SARIMA: RMSE = **138.6** MW
- SARIMAX: RMSE = **131.9** MW

4.2 Deep learning models

Deep learning models substantially improve performance over the statistical baselines. Both GRU-based architectures and

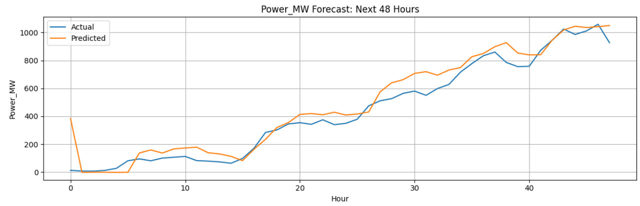

Figure 7: Forecast produced by the Transformer model with an unmasked decoder for bidding zone NO2.

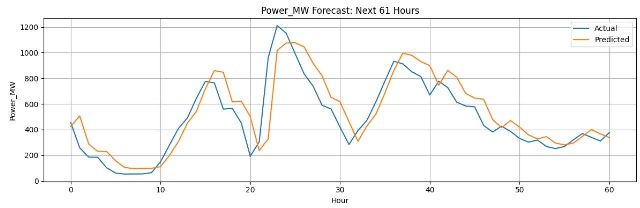

Figure 8: Forecast produced by the Transformer model with a masked (causal) decoder for bidding zone NO2.

Transformer-based models are better able to capture daily seasonality and respond to changes in wind conditions.
The test-set RMSE values for zone NO2 are:
- GRU_Weak: RMSE = **158.6** MW
- GRU_Deep: RMSE = **112.4** MW

While the GRU_Deep model improves upon the weaker recurrent baseline, the GRU_Weak architecture frequently collapses toward near-constant predictions close to the empirical mean of the signal. This behavior is visible in ground-truth versus prediction plots (Figure 4) and results in large systematic error. Such behavior is consistent with recurrent models minimizing loss by regressing toward the central tendency of the data under limited effective capacity.

4.3 Transformer models: masked vs. unmasked decoder

Two Transformer variants are evaluated:
- **Unmasked decoder**: decoder self-attention is unrestricted.
- **Masked (causal) decoder**: decoder self-attention is constrained to past time steps only.

Both models share the same encoder architecture, input features, and training procedure. The only difference lies in the decoder self-attention mechanism.
The quantitative performance for zone NO2 is:
- Transformer (unmasked): RMSE = **72.4** MW
- Transformer (masked): RMSE = **81.9** MW

Figure 7 shows the forecast produced by the unmasked Transformer. The model closely tracks observed wind power production, capturing diurnal patterns, sharp ramps, and recovery phases across the full 48-hour forecast horizon.
Figure 8 presents the forecast from the masked Transformer. While the overall structure is similar, the causal constraint leads to slightly smoother predictions and reduced responsiveness during periods of rapid change.

Overall, the unmasked Transformer achieves the best forecasting performance in this study and is therefore selected for uncertainty estimation. Ground-truth versus prediction visualizations are included in the Methods section for the GRU_Weak model to illustrate its tendency to converge toward the mean or median of the wind power signal (Figure 4). Additional forecast plots for the GRU architectures are not included here, as their predictive performance is substantially lower than that of the Transformer and their behavior is already sufficiently characterized by the diagnostic analyses presented earlier.

4.4 Uncertainty estimation

Uncertainty is estimated only for the best-performing model, namely the unmasked Transformer. An ensemble of $N = 10$ independently trained Transformer models is used to quantify predictive uncertainty.
For each forecast step, the ensemble mean is computed as:

$$\hat{y}_{\text{mean}}(t) = \frac{1}{N}\sum_{i=1}^{N}\hat{y}_i(t),$$

and uncertainty is quantified using the ensemble standard deviation:

$$\text{std}(t) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\hat{y}_i(t) - \hat{y}_{\text{mean}}(t)\right)^2}.$$

Prediction intervals are constructed using both parametric bands (± 1 and ± 2 standard deviations) and non-parametric ensemble quantiles. The uncertainty bands widen during volatile periods and contract during stable conditions, reflecting the model's confidence in its predictions.
Representative uncertainty metrics for zone NO2 are:

- 90% interval coverage: **88.7**%
- 95% interval coverage: **94.1**%
- Average interval width (95%): **182.6** MW

5 Discussion

The results show that Transformer-based models outperform both statistical models and recurrent neural networks for two-day-ahead wind power forecasting. This improvement is most noticeable at longer horizons.
Statistical models rely on strong assumptions about stationarity and linearity, which do not hold for wind power data. Recurrent models handle non-linearity better but struggle with long-range dependencies due to limited memory.
The Transformer performs better because attention allows the model to focus on relevant past time steps, even if they are far in the past. This matches the nature of wind dynamics, where delayed weather effects are common.
Uncertainty estimation further improves the usefulness of the forecasts by providing information about confidence. This is important for operational decision-making.

A limitation of this work is that only one preprocessing strategy is used. Future work could explore signal decomposition methods such as STL, wavelet transforms, or empirical mode decomposition to separate trend and seasonal components before modeling.
Overall, this study shows that attention-based models are well suited for realistic wind power forecasting tasks when combined with careful preprocessing and feature engineering.

Bibliography

[1] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, pp. 11121–11128, AAAI Press, 2023.