Contents lists available at ScienceDirect

Renewable Energy

journal homepage: www.elsevier.com/locate/renene

Advanced feature engineering in microgrid PV forecasting: A fast computing and data-driven hybrid modeling framework

Md. Ahasan Habib [*], M.J. Hossain

School of Electrical and Data Engineering, University of Technology Sydney, Ultimo, 2007, NSW, Australia

ARTICLE INFO

Keywords:
PV power forecasting
Hybrid deep learning model
Feature engineering
Modified EMD

ABSTRACT

This study introduces an innovative framework designed to forecast the fluctuating short-term generation of photovoltaic (PV) energy in isolated microgrids. The framework relies entirely on solar irradiance data obtained from weather stations located far from the target microgrid. It implements a sophisticated decomposition approach to effectively extract an enriched collection of features from the dataset. Subsequently, a machine learning-based clustering method further enhances the forecasting process by accurately separating the relevant data points. The approach utilizes an advanced two-stage Hybrid Data Linked Model (HDLM) architecture, integrating a Layered Recurrent Neural Framework (LRNF) for prediction and a pattern identification network unit for pattern extraction. This paper demonstrates significant improvements in both the accuracy and effectiveness of estimating PV generation, achieving a mean absolute error of 1.02, a root mean square error of 2.176, and an R-squared score of 0.991. Additionally, the method reduces computing time by 15% after finalizing the input features. A comparative analysis evaluates the superior forecasting capabilities of the HDLM in remote microgrids by benchmarking it against other advanced hybrid deep learning models. The findings highlight the HDLM's potential to greatly enhance and revolutionize the management and operation of renewable energy systems in remote microgrids.

1. Introduction

The number of customer-driven PV installations has increased dramatically in recent years, owing largely to financial incentives such as feed-in tariffs (FiT) offered for exporting excess power back to the grid [1,2]. However, this accelerated adoption of PV systems poses some obstacles, the most significant of which is its impact on grid stability [3,4]. In addition, the increasing number of photovoltaic systems adds complexity to microgrid energy management and planning, making it harder to maintain an appropriate operational balance [5]. When multiple PV sources feed power into the network simultaneously during peak solar generation periods, it can result in voltage fluctuations and potential overloads, thus compromising the stability and safety of both microgrids and the national grid [6–9]. As a result, governments and utility companies are forced to put restrictions on electricity exports, resulting in lower returns for PV owners and an extension of the payback period for their solar investments [10–12]. This circumstance emphasizes the necessity for sophisticated management strategies to mitigate the negative effects of integrating a substantial amount of solar energy into the existing electrical infrastructure, or microgrids.

Voltage stability challenges in PV-enriched microgrids and conventional electrical networks can be mitigated by accurately estimating PV generation. Utility grid operators can anticipate and prevent issues with accurate projections. With these predictive capabilities, energy supply and demand are balanced, improving power grid stability and security. Researchers have conducted a significant amount of research and created numerous models to predict PV generation as a result of the development of machine learning (ML) and deep learning (DL) algorithms. Different ML approaches, such as Random Forest (RF), Neural Networks (NN), Support Vector Machines (SVM), Recurrent Neural Networks (RNN), Convolution Neural Networks (CNN), and so on, are employed to forecast PV generation. These algorithms are widely used because they can recognize patterns in PV generation and use that information to make better predictions than statistical methods. For example, the authors of Ref. [13] forecasted solar generation at a university campus in Manchester using a variety of machine learning methods such as RF, NN, SVM, and Linear Regression (LR). The results demonstrate that RF has the lowest average error, with other algorithms having comparable but significantly greater errors of more than 32%. Ref. [14] applies similar predictive models from Ref. [13] to Alice Springs, Australia, to predict solar power generation in that area. In Ref. [15], a genetic algorithm-based support vector machine (GASVM) is used to predict the output of solar power in home PV

* Corresponding author.
E-mail address: mdahasan.habib@student.uts.edu.au (M.A. Habib).

https://doi.org/10.1016/j.renene.2024.121258
Received 19 May 2024; Received in revised form 30 July 2024; Accepted 26 August 2024
Available online 30 August 2024

systems. The model is then tested using data from Deakin University's local weather monitoring system. The analysis demonstrates that the proposed model is superior to the traditional SVM model in terms of accuracy. Ref. [16] summarizes the benefits and performance of various ML-based models in this area in a single article.

Due to the presence of a memory unit, deep learning-based models can predict better than ML-based models. Long short-term memory (LSTM), CNN, bidirectional LSTM (Bi-LSTM), and stacked LSTM models have recently been used to predict photovoltaic PV generation [17–21]. Furthermore, hybrid models that combine these units have become popular, providing more advanced and accurate predictions. The authors of the study [22] used a CNN–LSTM hybrid model to predict hourly PV generation and got an RMSE of 4.58. This was better than the LSTM model, which got an RMSE of 13 (Ref. [18]) and 7.667 (Ref. [19]). The research conducted by authors referenced in [23,24] yielded a CNN-BiLSTM-based model for PV forecasting. In their work, the authors in [23] utilized the Pearson Correlation Coefficient to capture highly correlated meteorological elements as input features. Both studies utilized the initial convolutional layer to discern patterns in PV generation. Notably, [23] demonstrated the superiority of their hybrid model over standalone CNN or Bi-LSTM models. Furthermore, in [25], a different hybrid architecture combining CNN and stacked LSTM was proposed, with the authors showcasing the effectiveness of their 5-layered network (comprising 2 CNN layers and 3 LSTM layers) over various single-stage and hybrid models. However, the specific execution details of this complex hybrid model were not provided. In recent years, researchers have advanced the field with hybrid deep-learning architectures incorporating attention mechanisms for PV forecasting, as highlighted in [26,27]. While these models exhibit efficient performance, it is crucial to note that the inclusion of attention layers escalates both computation time and model complexity. Hence, careful consideration is warranted when deploying such models in practical applications. Although hybrid models typically generate more accurate predictions with lower error rates, they necessitate significantly more computational resources and a longer execution period.

An effective deep-learning model requires a large dataset with precisely selected features. The model's prediction power is enhanced by relevant features, while execution time is impacted by dataset size. Increasing the number of relevant features aligned with target variables boosts model predictability and performance [28]. However, the majority of the previously cited articles on PV forecasting utilized numerous meteorological variables and historical data to make predictions. It might be difficult to collect temperature, humidity, and other meteorological data for remote sites far from weather stations. In such cases, feature generation becomes even more important. For instance, Ref. [29–31] uses wavelet transform (WT) as a feature engineering technique along with DL or NN to anticipate PV output. In Ref. [29,30], WT decomposes solar energy time-series data into frequency series for statistical feature extraction. In [31], discrete WT (DWT) divides PV output power into approximate and detailed components, which are fed into an adaptive neuro-fuzzy inference system (ANFIS) model for final output. WT can capture frequency-domain information but makes modeling more complicated. The authors of Ref. [32–34] used Variational Mode Decomposition (VMD) to generate features for forecasting models by decomposing the input signal into stable components with different characteristics. VMD helps to decompose highly fluctuating data into stable components with periodic features, hence boosting the models' capacity to make accurate predictions. A significant limitation of VMD is the requirement to adjust various parameters, which poses a challenging task and introduces intricacy to the modeling process.

A data-driven signal processing technique called Empirical Mode Decomposition (EMD) decomposes a complicated signal into simpler, oscillatory components that capture Intrinsic Mode Functions (IMFs). The decomposition results may be inaccurate because EMD is susceptible to mode mixing, which occurs when the IMFs are not well-separated, and because it is also subject to noise and boundary effects. To improve their predictions, the authors of Ref. [35–37] utilized EMD to create features based on the trends and insights in PV power generation data, which they then fed into a DL model. Modern feature engineering research favors hybrid EMD over standard EMD due to its accuracy and efficiency in handling complicated data structures. However, this improvement increases execution time, weighing efficiency against performance in data analysis applications. An effective short-term PV power production forecasting model in Algeria using time-varying filter EMD (TVF-EMD) and extreme learning machine (ELM) [38]. Despite using thirty newly established characteristics, which will take a lot of processing time, this analysis did not identify the most prominent IMF aspects. A combined LSTM-ARMA model is proposed in Ref. [39] augmented by ensemble EMD (EEMD) extracts related components from PV output data, lowering PV power generation intermittency and uncertainty. Compared to standard models, the technique predicts PV output fluctuations and trends better. However, the high MAPE under varied weather conditions highlights accuracy issues. The authors of Ref. [40] used Improved Complete EEMD with Adaptive Noise (ICEEMDAN) to predict the PV generation. Here, the authors put more effort into predicting the peaks and valleys of future PV production.

A few recently proposed method's decomposition and model information, including input and generated features, are shown in Table 1. One major oversight in the current research is the absence of in-depth analyses comparing various feature engineering strategies to ascertain the best approach. Numerous studies concentrate on a single approach without assessing substitutes, possibly ignoring better approaches. In addition, these studies often use all algorithm-generated features without prioritizing those most related to target variables. This strategy might result in less-than-ideal accuracy in forecasting and an increase in the complexity of the computations involved. Previous studies have often used PV power generation and weather data as inputs for prediction models. However, if any meteorological data is unavailable or contains errors, the forecasting process can yield incorrect results. To ensure accurate predictions, operators must verify the reliability of all weather data used in model training and the precision of PV power data. Without this assurance, the effectiveness of the prediction process could be compromised. This research addresses past constraints by improving feature creation with a modified decomposition technique and rigorously comparing and identifying the best feature engineering algorithm. Additionally, a significant benefit of this research is that the framework requires only PV power data. Furthermore, the proposed study presents a novel approach (to the best of our knowledge) in energy forecasting applications by estimating PV generation within an isolated microgrid with the fewest data parameters demanded. The primary contributions of this research are:

- **Advanced feature generation and evaluation:** Implemented a modified decomposition technique to improve the feature generation process. This approach greatly enhanced the quality of the input features for the computing framework. It was carefully contrasted with other feature engineering algorithms to find the most effective one.
- **Sophisticated feature selection and clustering:** Employed a machine learning algorithm to cluster and differentiate crucial characteristics associated with the target variable. Comparative analysis with various clustering methods validated the effectiveness of the used approach in pinpointing the key features for further analysis.
- **Efficient real-world data implementation for PV forecasting:** Designed and executed a hybrid deep learning model to provide quick PV generation predictions using real-world data. This method guaranteed quick calculation and showed the model's capacity to work in data-scarce contexts, making it useful for real-world applications.

Table 1
Performance comparison of decomposition algorithms and forecasting models with input features and error metrics.

Ref	FEA	Forecasting algorithm	Initial input	Generated no. of features	R^2 score	RMSE (kW)
[29]	WT	LSTM	T, CC, H, V, HPP	4	0.93817	0.0049
[30]	WT	ANN	HPP, I, T	4	–	895
[31]	DWT	ANFIS	HPP	12	–	0.002
[32]	VMD	NN	SWR, T, H, CC, t	3	0.9768	1.2381
[33]	VMD	PSO-LSTM	HPP	4	0.9578	–
[34]	VMD	LSTM-RVM	HPP	7	–	4.80
[35]	EMD	BPNN	HPP	6	–	0.07
[36]	EMD	SCA-ELM	HPP	13	–	0.04
[38]	TVF-EMD	ELM	HPP	30	0.92	23.297

Note: FEA = Feature Engineering Algorithm, RVM = Relevance Vector Machine, BPNN = Back Propagation Neural network, SCA = Sine-Cosine Algorithm, T = Temperature, I = Irradiance, CC = Cloud Coverage, t = Time, H = Humidity, and HPP = Historical PV Power.

- **Benchmark Comparisons and Technological Advances:** Extensive comparisons with state-of-the-art machine learning and deep learning models demonstrated the superior accuracy of our proposed model for solar energy forecasting, highlighting its significant advancements and potential to set new benchmarks in the field.

2. Methodology & model description

The primary objective of this study is to analyze and predict the generation of PV energy in a remote microgrid, as depicted in Fig. 1. This remote location of the microgrid from cities and weather stations creates special opportunities and challenges for sustainable energy management. The solar irradiance data for the microgrid location is first gathered from the weather station, and solar generation is then computed using the installed PV panel capacity. After that, solar power generation trends decompose to generate additional features. These extracted features explain the region's solar energy patterns. In the subsequent steps, the decomposed features are classified using a clustering mechanism, which increases the model's adaptability to a variety of solar conditions. After clustering, the HDLM inputs are prepared.

Following that, the hyperparameters of the model are optimized to ensure accurate forecasting of PV generation. This process entails fine-tuning the number of layers in each cycle, batch size, window length, and so on. Finally, the performance metrics are analyzed to ensure the prediction accuracy and superiority of the proposed model compared with the state-of-the-art models. However, the entire framework is divided into four discrete steps that are interrelated and operate sequentially. These stages are outlined below: (i) Data collection and aggregation; (ii) Feature Engineering and Data processing (iii) Model development and Optimization; and (iv) Prediction and Performance evaluation. Detailed explanations for each of these stages are provided in the subsequent sections.

2.1. Data collection and aggregation

The first step in data collection and aggregation is to gather irradiance data for the area from a meteorological station. Outliers, missing values, and NaN (Not a number) must be removed using robust methods to improve model training and prediction.

$$Z = \frac{(X-\mu)}{\sigma} \tag{1}$$

$$x_{\text{missing}} = x_p + \frac{(x_q - x_p)}{(q-p)} \times (target - p) \tag{2}$$

$$x_n(\text{NaN}) = \text{avg}(x_{n-1}, x_{n-2}, x_{n-3}, \dots .., x_{n-10}) \tag{3}$$

The Z-score (Z), which is the normalized difference between a data point (X) and its mean (μ), divided by the standard deviation (σ), is used to identify outliers Eq. (1). An outlier is defined as any data point that has an absolute Z-score over 3. Interpolation, as described in Eq. (2), is utilized with the help of surrounding point values and indices (p, q) to append missing data points (x_{missing}). After that, the average of the 10 previous observations Eq. (3) is used in place of NaN values (x_n(NaN)). Then based on the installed PV panel information, the generated solar power is calculated and attached to the previously processed data.

2.2. Feature engineering and data processing

After collecting data, a significant number of features are created in the "Feature Engineering and Data Processing" step to improve the accuracy of forecasts. The first step is to apply three popular methods for decomposing PV generation data: VMD, EMD and CEEMDAN. Then, to further optimize the decomposition process, an advanced methodology named Modified EMD (MEMD) is applied. The subsequent paragraphs outline the methodology of the decomposition algorithms one by one.

2.2.1. VMD

VMD is a signal processing technique that decomposes a signal into separate modes. The primary objective of VMD is to break a signal down into frequency-centered modes and then use iterative updates to cut the bandwidth of each mode. Let us consider $y(t)$ as a time series signal. The VMD technique uses Eq. (4) to break down $y(t)$ into various modes v_k [32,33].

$$\min_{\{u_k\},\{w_k\}} \left\{ \sum_{k=1}^{K} \left\| \delta(t) * u_k(t) e^{-jw_k t} \right\|^2 \right\}, \text{ subject to } u_k = x(t) \tag{4}$$

Each mode is characterized by modulation around a specific frequency z_k, and $\mu(t)$ represents the modified Dirac delta function. Upon applying this equation, the signal $y(t)$ gets decomposed into these modes $v_k(t)$, with each mode precisely aligned with its respective frequency z_k [33,34].

$$Q(\{v_k\},\{z_k\},\gamma) = \sum_{k=1}^{M} \left\| \mu(t) * v_k(t) e^{-jz_k t} \right\|^2 + \beta \left(y(t) - v_k(t) \right)^2 + \gamma \left(t - y(t) v_k(t) \right) \tag{5}$$

Eq. (5) converts the constrained optimization problem into an unconstrained one by introducing a modified Lagrange multiplier (γ), and a penalty factor (β), to strike a balance between the accuracy of the signal $y(t)$ and the simplicity of the modes. The signal continues to be separated into its modes $v_k(t)$, with improved quality of decomposition as a result of γ and β. Eq. (6) utilizes the difference between the Fourier transform of $y(t)$ and other modes, regulated by β, to update each frequency domain mode v_k. The modes $v_k(t)$ are refined during each iteration to more accurately represent $y(t)$ within their respective frequency bands.

$$\hat{v}_k^{n+1}(z) = \frac{\hat{y}(z) - \sum_{i \neq k} \hat{v}_i^n(z)}{1 + 2\beta(z - z_k)^2} \tag{6}$$

The center frequency, z_k, is updated for each mode to align with its most typical frequency components. The update equation is expressed

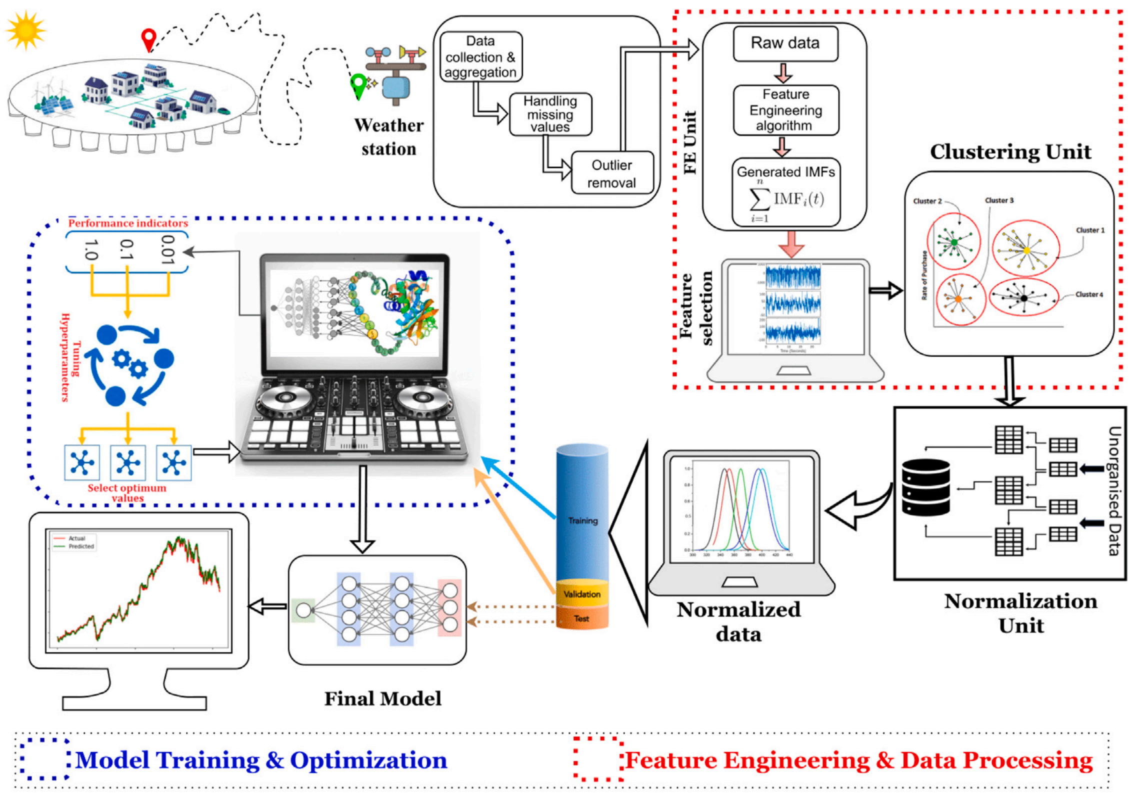


Fig. 1. Schematic depiction of the entire approach to predict PV generation of a remote location using the proposed HDLM.

as a weighted average frequency, with the magnitudes of the frequency components of the mode serving as the weights. The updated center frequency z_k^{n+1} for the kth mode in the (n+1)-th iteration is:

$$z_k^{n+1} = \frac{\int_0^\infty z \left|\hat{v}_k^{n+1}(z)\right|^2 dz}{\int_0^\infty \left|\hat{v}_k^{n+1}(z)\right|^2 dz} \tag{7}$$

This modification centers each mode (v_k) around its paramount frequency component, preserving the signal structure $y(t)$.

2.2.2. EMD

EMD is a universal signal processing method that decomposes a signal into IMFs without leaving the temporal domain. EMD can analyze non-linear and non-stationary signals better than Fourier or wavelet transforms [41]. The EMD method for a time series signal $y(t)$ involves iterative phases. First, it identifies all local extrema of $y(t)$. Then the upper envelope ($e_{\mathrm{up}}(t)$) and the lower envelope ($e_{\mathrm{low}}(t)$) are created by interpolating the local maxima and minima, respectively. The next step is to determine the mean envelope ($m(t)$) by calculating the average of $e_{\mathrm{up}}(t)$ and $e_{\mathrm{low}}(t)$. and $m(t)$ is subtracted from $y(t)$ to produce the detail, $d(t)$. If $d(t)$ meets IMF conditions, it is labeled IMF_1. The IMF is subtracted from $y(t)$, generating a residue $r(t) = y(t) - \mathrm{IMF}_1$. This process is repeated on $r(t)$, extracting subsequent IMFs until no more can be derived. The completion of the EMD process results in the accumulation of IMFs, denoted as $\mathrm{IMF}_1, \mathrm{IMF}_2, \dots, \mathrm{IMF}_n$, in addition to a final residue, denoted as $r_n(t)$, which symbolizes the typical trend of the signal. Each IMF recognizes unique oscillatory modes in the original signal $y(t)$. The algorithm of this decomposition process is represented in Algorithm 1.

2.2.3. CEEMDAN

The Complete Ensemble Empirical Mode Decomposition with Adaptive Noise (CEEMDAN) is a modern signal processing method that breaks down non-stationary and non-linear data into IMFs. CEEMDAN enhances the robustness of the classical EMD by systematically adding white noise to the data. In CEEMDAN, instead of adding the same level of white noise to the signal, the amount of added noise is adaptively adjusted depending on the residue of the decomposition process. Another common problem in EMD is mode mixing, which happens when different scales are mixed into one mode. The CEEMDAN method reduces this problem as much as possible. CEEMDAN enhances the stability and precision of the decomposition by incorporating noise and facilitating the extraction of actually signed IMFs through the averaging of multiple ensembles, each modified with a distinct realization of noise. The step-by-step working principle of CEEMDAN over a time series signal $y(t)$ is presented in Algorithm 2.

Algorithm 1 Extraction of IMFs from Time Series Data using EMD

```
Require: Time series signal $y(t)$
Ensure: Set of Intrinsic Mode Functions (IMFs)
 1: Initialize residue $r(t) = y(t)$, $i = 1$
 2: while residue $r(t)$ has at least two extrema do
 3:     Identify all local extrema of $r(t)$
 4:     Interpolate between maxima to form envelope $e_{\mathrm{up}}(t)$
 5:     Interpolate between minima to form envelope $e_{\mathrm{low}}(t)$
 6:     Compute the mean envelope $m(t) = \frac{e_{\mathrm{up}}(t)+e_{\mathrm{low}}(t)}{2}$
 7:     Extract detail $d(t) = r(t) - m(t)$
 8:     Check if $d(t)$ is an IMF (satisfies IMF criteria)
 9:     if $d(t)$ is an IMF then
10:         IMFs[$i$] $= d(t)$ and $i = i + 1$
11:         Update residue $r(t) = r(t) - d(t)$
12:     end if
13: end while
14: return Set of Intrinsic Mode Functions IMFs
```

Algorithm 2 Extraction of IMFs from Time Series Data using CEEMDAN

```
Require: Time series signal $y(t)$, number of ensembles $N_e$, initial noise amplitude $A$
Ensure: Set of IMFs.
 1: Initialize residue $r_0(t) = y(t)$, $i = 1$
 2: while residue $r_{i-1}(t)$ has at least two extrema do
 3:     Initialize ensemble IMF storage IMF_ens
 4:     for $j = 1$ to $N_e$ do
 5:         Add white noise $w_j(t)$ of amplitude $A$ to $r_{i-1}(t)$: $y_j(t) = r_{i-1}(t) + w_j(t)$
 6:         Apply EMD to $y_j(t)$ to extract the first IMF $c_{i,j}(t)$
 7:         Store $c_{i,j}(t)$ in IMF_ens
 8:         Subtract $c_{i,j}(t)$ from $y_j(t)$ for next iteration: $r_j(t) = y_j(t) - c_{i,j}(t)$
 9:     end for
10:     Calculate ensemble mean of extracted IMFs: $c_i(t) = \frac{1}{N_e}\sum_{j=1}^{N_e} c_{i,j}(t)$
11:     Store $c_i(t)$ as an IMF and update $i = i + 1$
12:     Update residue $r_i(t) = r_{i-1}(t) - c_i(t)$
13:     Reduce noise amplitude $A$ for the next level
14: end while
15: return Set of IMFs
```

2.2.4. MEMD

In the proposed framework, MEMD is used which is the extended version of EMD algorithm to create more features and make the decomposition process more robust. The step-by-step process of the MEMD algorithm will be discussed below.

Let us examine a time series signal $u(t)$. For each ensemble j, a noise series $w_j(t)$ is added to the input signal which serves multiple purposes shown in Eq. (8). The first benefit of adding noise is that it clarifies scales in IMFs, reducing mode mixing. Secondly, the noise provides a continuous background to find important patterns despite smaller oscillations. Finally, this increases decomposition resilience and reduces overfitting.

$$u_j(t) = u(t) + w_j(t), \text{ where } w_j(t) \sim \mathcal{N}(0, \sigma^2) \tag{8}$$

Now, at any point t_a in this noise-assisted signal $u_j(t)$ is a local maximum if it satisfies Eqs. (9) and (11) otherwise it will be local minima (As Eq. (10)).

$$u_j(t_a) > u_j(t_a - \delta t) \quad \text{and} \quad u_j(t_a) > u_j(t_a + \delta t) \tag{9}$$

$$u_j(t_a) < u_j(t_a - \delta t) \quad \text{and} \quad u_j(t_a) < u_j(t_a + \delta t) \tag{10}$$

$$\forall t \in (t_a - \epsilon, t_a + \epsilon)\backslash\{t_a\}, \quad u_j(t_a) > u_j(t) + \frac{1}{2}\frac{d^2u_j}{dt^2}\Big|_{t=t_a}(t - t_a)^2 + o((t - t_a)^2) \tag{11}$$

where δt and ϵ are small increments. The MEMD algorithm uses cubic spline functions to interpolate local extrema in the noise-assisted signal $u_j(t)$ to establish upper and lower envelopes for each ensemble iteration j. Eq. (12) defines the cubic spline interpolation, $S_j(t)$, for each ensemble using cubic polynomials.

$$S_j(t) = a_{ij}(t - t_{ij})^3 + b_{ij}(t - t_{ij})^2 + c_{ij}(t - t_{ij}) + d_{ij}, \quad t_{ij} \le t \le t_{i+1,j} \tag{12}$$

where a_{ij}, b_{ij}, c_{ij}, and d_{ij} are the spline coefficients for each interval in the j-th ensemble. The next stage in the MEMD is to compute the mean of these envelopes ($M_j(t)$), which provides a local trend of the signal. This is used to detrend the original data and extract the IMFs. $M_j(t)$ is calculated by averaging upper ($U_j(t)$) and lower ($L_j(t)$) envelopes at each point using the following expression [35,36]:

$$M_j(t) = \frac{U_j(t) + L_j(t)}{2} \tag{13}$$

$$D_j(t) = u_j(t) - M_j(t) \tag{14}$$

Whenever the detail ($D_j(t)$) meets the criteria for an IMF, it is regarded as an IMF. For instance, IMF_{ij} for that particular ensemble is considered. This approach is repeated until each ensemble iteration

Algorithm 3 MEMD algorithm to decompose solar generation data.

```
 1: Input: High-dimensional solar generation data series $u(t)$
 2: Output: Refined Intrinsic Mode Functions (IMFs): $\sum_{i=0}^{N-1} \hat{e}_i(t) + \hat{R}_N(t)$
 3: Parameter Initialization:
 4: Ensemble size $N_e$, noise standard deviation $\sigma$, total IMFs $N$
 5: Tolerance $\epsilon$, Sifting iterations limit $L$, Adaptive noise factor $\alpha$
 6: for $j = 1$ to $N_e$ do
 7:     Generate white noise series $w_j(t)$ with $\sigma$
 8:     Synthesize noise-assisted signal $u_j(t) \leftarrow u(t) + \alpha_j \cdot w_j(t)$
 9:     Initialize $h(t) \leftarrow u_j(t)$, $n \leftarrow 0$, IMF set $\mathcal{E}_j \leftarrow \emptyset$
10:     while Stopping criterion not met for $h(t)$ do
11:         Extract local extrema of $h(t)$
12:         Interpolate extrema to form upper $\mathcal{U}_j(t)$ and lower $\mathcal{L}_j(t)$ envelopes
13:         Compute the local mean $M_j(t) \leftarrow \frac{\mathcal{U}_j(t)+\mathcal{L}_j(t)}{2}$
14:         Extract detailed component $D_j(t) \leftarrow h(t) - M_j(t)$
15:         if Standard deviation of change in $h(t)$ is below $\epsilon$ then
16:             $\mathcal{E}_j \leftarrow \mathcal{E}_j \cup \{D_j(t)\}$
17:             Subtract the extracted IMF from $h(t)$ and update $h(t)$
18:         end if
19:         Update $\alpha_j$ based on the direct error control strategy
20:         Increment $n$
21:         if $n \ge L$ then
22:             Break while loop
23:         end if
24:     end while
25:     Store the residue $R_j(t) \leftarrow h(t)$
26: end for
27: Compute ensemble mean of corresponding IMFs: $\hat{e}_i(t) \leftarrow \frac{1}{N_e}\sum_{j=1}^{N_e} E_{ij}(t)$
28: Calculate the final residue: $\hat{R}_N(t) \leftarrow \frac{1}{N_e}\sum_{j=1}^{N_e} R_j(t)$
29: Combine all refined IMFs and residue to reconstruct the signal: $u'(t) \leftarrow \sum_{i=1}^{N} \hat{e}_i(t) + \hat{R}_N(t)$
30: Return the refined IMFs $\{\hat{e}_i(t)\}$ and the final residue $\hat{R}_N(t)$
```

j residue $R_j(t)$ has only one extremum, signifying no further IMF extraction. To obtain the final IMFs for the original signal $u(t)$, the corresponding IMFs from all ensembles are averaged, and the results are displayed in Eq. (15).

$$IMF_i(t) = \frac{1}{N_e}\sum_{j=1}^{N_e} IMF_{ij}(t) \tag{15}$$

where N_e is the number of ensembles. The original signal can be reconstructed from the IMFs and the final residue:

$$u(t) = \sum_{i=1}^{N} IMF_i(t) + R(t) \tag{16}$$

where N is the total number of IMFs extracted. The proposed MEMD offers several technical improvements over the traditional EMD. Primarily, MEMD addresses the mode mixing problem inherent in EMD by adding noise to the data, facilitating a more efficient separation of the IMFs. Additionally, MEMD enhances the reliability and steadiness of the decomposition procedure, leading to more uniform results across various iterations. Moreover, MEMD improves the accuracy of the decomposition, ensuring that the IMFs align more closely with the actual signals. The algorithm of the proposed MEMD is shown in Algorithm 3.

Following feature engineering, MEMD algorithm-generated IMFs are included in the dataset. This updated dataset is then sent to the clustering unit. This framework uses different clustering algorithms, such as K-means clustering, DBScan, Spectral clustering, and Gaussian Mixture Models (GMM), to select the most suitable clustering method. KMeans clustering is a well-known machine learning-based method that divides data into segments. This algorithm is an iterative method that is used to divide a dataset into a predetermined number of clusters, which is denoted by the symbol Q. The procedure begins with the selection of Q initial centroids at random, denoted by $\theta_1, \theta_2, \dots, \theta_Q$. After that, the Euclidean distance is used to map each data point P_i in the dataset to the centroid that is closest to it. The mathematical expression for this assignment is as follows [42]:

$$\text{GroupLabel}(i) = \underset{q\in\{1,2,\ldots,Q\}}{\text{argmin}} \|P_i - \theta_q\|^2 \tag{17}$$

The centroids are recalculated as the mean of each cluster's data points after assignment. The revised centroid for the q-th cluster is calculated using the equation:

$$\theta_q = \frac{1}{|\{i|\text{GroupLabel}(i) = q\}|} \sum_{\{i|\text{GroupLabel}(i)=q\}} P_i \tag{18}$$

The data point assignment and centroid updating process is repeated until the centroids stabilize, indicating convergence. The technique generates a set of Q clusters, each with minimal within-cluster variances and different centroids.

DBScan, which stands for Density-Based Spatial Clustering of Applications with Noise, is another famous clustering method. It groups data points based on their density, which enables it to find clusters of any shape and adapt to noise effectively. The algorithm requires two parameters: ϵ, which denotes the utmost distance permissible between two points to be regarded as neighbors, and *MinPts*, which signifies the smallest number of points necessary to constitute a dense region. A point P_i is considered a core point when at least *MinPts* points are within a radius of ϵ from it. The neighborhood of P_i is made up of the collection of points inside this radius, which is mathematically defined as [43]:

$$N_\epsilon(P_i) = \left\{P_j \in D \mid \text{dist}(P_i, P_j) \le \epsilon\right\} \tag{19}$$

From these central points, DBScan repeatedly expands clusters by combining adjacent dense areas and labels points outside of any cluster as noise.

Spectral Clustering is a clustering method that uses a similarity matrix derived from the data to perform dimensionality reduction and then clusters in a reduced space. It is especially effective for identifying clusters in data that is not well-separated in the original feature space. The method starts with creating a similarity matrix S, where each element S_{ij} indicates the similarity between data points i and j. A common choice for the similarity measure is the Gaussian similarity function [44]:

$$S_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right) \tag{20}$$

where σ is a scaling parameter. From this similarity matrix, the Laplacian matrix L is constructed as:

$$L = D - S \tag{21}$$

where D is the degree matrix with $D_{ii} = \sum_j S_{ij}$. Spectral Clustering then computes the Laplacian matrix's eigenvalues and eigenvectors to distribute the data onto a lower-dimensional space.

Gaussian Mixture Models (GMM) is a probabilistic clustering technique that assumes a combination of various Gaussian distributions with unknown parameters creates data. A Gaussian distribution represents each cluster. The Expectation–Maximization (EM) algorithm is used to predict the model parameters. The estimates of the parameters are improved over and over again. The probability density function for a data point $\mathbf{x}$ in GMM is:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{22}$$

where K is the number of clusters, π_k is the mixing coefficient for the kth Gaussian component, $\boldsymbol{\mu}_k$ is the mean vector, and $\boldsymbol{\Sigma}_k$ is the covariance matrix of the kth Gaussian distribution. The EM algorithm alternates between the Expectation phase, which computes each Gaussian component's responsibility for each data point, and the Maximization step, which adjusts the parameters to maximize data likelihood. This process is repeated until convergence.

To identify the optimal clustering technique, the following performance indicators were utilized: The Silhouette Score (SS) quantifies the similarity of an object to its cluster relative to other clusters. Higher values indicate more distinct clusters. The Davies–Bouldin (DB) Index measures the average similarity ratio between each cluster and its most similar cluster. Lower values indicate better clustering. On the other hand, the Calinski–Harabasz (CH) Index evaluates the ratio of the sum of between-cluster dispersion to within-cluster dispersion. Higher values indicate better-defined clusters with more separation. The driving equations of the metrics are given below [45,46]:

$$SS = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \tag{23}$$

$$DB = \frac{1}{N}\sum_{i=1}^{N} \max_{j\neq i}\left(\frac{\sigma_i + \sigma_j}{d(c_i, c_j)}\right) \tag{24}$$

$$CH = \frac{\text{tr}(B_k)}{\text{tr}(W_k)} \times \frac{N-k}{k-1} \tag{25}$$

where, $a(i)$ is the average distance from the i-th point to the other points in the same cluster, and $b(i)$ is the minimum average distance from the i-th point to points in a different cluster. In Equation. (24), σ_i is the average distance between each point in cluster i and the centroid of cluster i, and $d(c_i, c_j)$ is the distance between the centroids of clusters i and j. In Equation. (25), $\text{tr}(B_k)$ is the trace of the between-cluster dispersion matrix, $\text{tr}(W_k)$ is the trace of the within-cluster dispersion matrix, N is the total number of points, and k is the number of clusters.

Following the implementation of the best clustering algorithm, it is important to perform data normalization to improve the accuracy and reliability of the model. This process commonly involves utilizing techniques such as Standard Scaler, Min-Max Scaler, and Max Absolute Scaler [47,48]. In this study, the Min-Max Scaler is utilized, which rescales each feature to a [0, 1] range while preserving the original data's distribution. The first step is to determine the minimum and maximum values for each feature. After that, Eqs. (26) and (27) are used to change each data point (x_i), putting the feature on a scale from 0 to 1, where 0 is the lowest number in the raw data and 1 is the highest.

$$x_i' = x_i - \min(X) \tag{26}$$

$$x_i'' = \frac{x_i'}{\max(X) - \min(X)} \tag{27}$$

2.3. Model development and optimization

A two-staged neural network architecture predicts future PV generation using normalized data from the previous level shown in Fig. 2. A complex convolutional structure is used in the model's first stage, named the feature synthesis module. This module is quite good at extracting and refining relevant characteristics from complicated datasets. Afterward, in the second stage, the layered recurrent neural network variant predicts PV generation. The next section explains the underlying principle in sequential order using mathematical formulas.

2.3.1. Convolutional layer

In the first stage of the proposed HDLM, the convolutional layer processes grid-like data. With trainable filters and biases, it can learn spatial groups of features through forward and backward propagation [22].

- **Forward Propagation:** During the forward propagation phase, every neuron in the network gets information from its receptive field in the previous layer or input sequence. In Eq. (28), the input time series X is convolved with the filter w_{lm} and bias b_{li} to derive the intermediate values x_{li}. Following this, Eq. (29) applies the

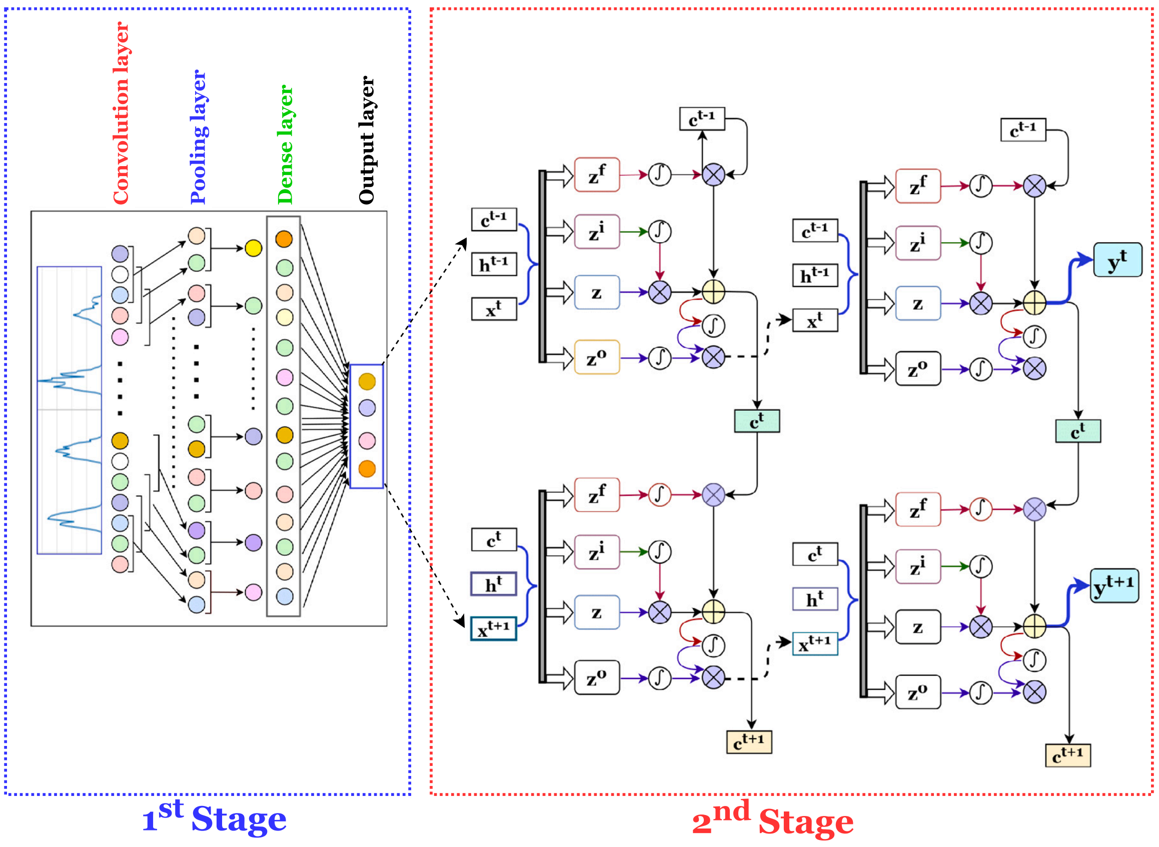


Fig. 2. Two-stage HDLM schematic for remote PV generation prediction.

activation function $f(\cdot)$ to these intermediate values to create the output feature map o_{li}.

$$x_{li} = \sum_{m=0}^{2} w_{lm} * o_{l-1,i+m} + b_{li} \tag{28}$$

$$o_{li} = f(x_{li}) \tag{29}$$

- **Backward Propagation:** The model is updated by estimating the error function gradients concerning various parameters during backward propagation. Eq. (30) calculates the gradient of the error concerning the input x_{li} by incorporating the activation function's derivative and the reversed filter weights. Eq. (31) estimates the gradient with respect to the filter weights w_{lm}, while Eq. (32) computes the gradient with respect to the bias b_{li}.

$$\frac{\partial E}{\partial x_{li}} = \delta_{li} * \text{rot180}\{w_{l+1,m}\} f'(x_{li}) \tag{30}$$

$$\frac{\partial E}{\partial w_{lm}} = \sum_{i=0}^{H-3} \delta_{li} o_{l-1,i+m} \tag{31}$$

$$\frac{\partial E}{\partial b_{li}} = \sum_{i=0}^{H-3} \delta_{li} \tag{32}$$

2.3.2. Pooling layer

After the convolutional layer, the pooling layer reduces feature map spatial dimensions in the proposed HDLM. This decrease makes the computations easier and lowers the risk of overfitting. The pooling layer functions independently on each depth slice of the input, resizing it spatially.

- **Forward Propagation:** Forward propagation of the pooling layer simplifies convolutional layer outputs. As shown in Eq. (33), max-pooling reduces data spatial dimensions by selecting the highest value from a given location of the previous layer.

$$\text{Max-pooling:} \quad o_{l,i} = \max(o_{l-1,2i}, o_{l-1,2i+1}) \tag{33}$$

- **Backward Propagation:** The gradients are retransmitted throughout the network during backward propagation. Eqs. (34) and (35) show that max-pooling provides the gradient of the error based on the output of neurons in the preceding layer that contributed to the maximum value during the forward pass.

$$\frac{\partial E}{\partial o_{l-1,2i}} = \begin{cases} \frac{\partial E}{\partial o_{l,i}} & \text{if } o_{l-1,2i} = \max(o_{l-1,2i}, o_{l-1,2i+1}) \\ 0 & \text{otherwise} \end{cases} \tag{34}$$

$$\frac{\partial E}{\partial o_{l-1,2i+1}} = \begin{cases} \frac{\partial E}{\partial o_{l,i}} & \text{if } o_{l-1,2i+1} = \max(o_{l-1,2i}, o_{l-1,2i+1}) \\ 0 & \text{otherwise} \end{cases} \tag{35}$$

2.3.3. Weight update

By modifying the model's parameters specifically, the weights and biases during the weight update phase of the first stage of the proposed HDLM, the error can be minimized. The network's weights and biases are modified following backward propagation gradient computation. Eqs. (36) and (37) determine the update step magnitude and direction based on the learning rate (η) and the computed gradients of the error for weights and biases.

$$w'_{lm} = w_{lm} - \eta \frac{\partial E}{\partial w_{lm}} \tag{36}$$

$$b'_{li} = b_{li} - \eta \frac{\partial E}{\partial b_{li}} \tag{37}$$

After finishing the first stage's data propagation, the data is passed into the second stage, which is dedicated to predicting future values. This level uses consecutive layers of a recurrent neural framework

Table 2
Symbols and interpretation of symbols for the proposed RNF architecture.

Symbol	Interpretation
x_t	Input at time t
$h_{t-1}^{(l)}, c_{t-1}^{(l)}$	Previous HS, CS of layer l
$W_{x*}^{(l)}, W_{h*}^{(l)}, b_{x*}^{(l)}, b_{h*}^{(l)}$	Weights, biases for IG, FG, CS, OG of layer l
$i_t^{(l)}, f_t^{(l)}, g_t^{(l)}, o_t^{(l)}$	Gate activations of layer l
$c_t^{(l)}, h_t^{(l)}$	CS, HS of layer l at t
$E_{\delta t}^{(l)}$	Gradient of loss at t wrt $h_t^{(l)}$

(RNF) with memory units to memorize patterns for better prediction [49]. In this framework, data propagation takes place in both forward and reverse directions. The system uses dynamically updated weights and biases to forecast values after training, and the stage's operation is discussed in the following discussion. Fig. 2 displays four RNF cells, and it is assumed that the information proceeds from left to right. The operation of every cell is identical, and the following table (Table 2) provides symbols that will be utilized to clarify the operation.

The bidirectional data transfer of the proposed RNF is dependent on three important gates: the forget gate (FG), the input gate (IG), and the output gate (OG). The FG specifies whether to keep or delete old information, the OG creates the final hidden state (HS), and the IG adds new information to the cell state (CS). In forward propagation, layer l's IG handles x_t, $h_{t-1}^{(l)}$, and biases, creating gate activation $i_t^{(l)}$ Eq. (38). Similarly, the FG provides activation $f_t^{(l)}$ Eq. (39), influencing the previous cell state retention. Eq. (40) is used to compute the cell state update $g_t^{(l)}$, which involves activating the candidate state using LeakyReLU to ensure continuous gradient flow and faster learning. The updated cell state $c_t^{(l)}$ Eq. (41) combines scaled previous and candidate states.

$$i_t^{(l)} = \sigma(W_{xi}^{(l)} \cdot x_t + W_{hi}^{(l)} \cdot h_{t-1}^{(l)} + b_{xi}^{(l)} + b_{hi}^{(l)}) \tag{38}$$

$$f_t^{(l)} = \sigma(W_{xf}^{(l)} \cdot x_t + W_{hf}^{(l)} \cdot h_{t-1}^{(l)} + b_{xf}^{(l)} + b_{hf}^{(l)}) \tag{39}$$

$$g_t^{(l)} = \text{LeakyReLU}(W_{xc}^{(l)} \cdot x_t + W_{hc}^{(l)} \cdot h_{t-1}^{(l)} + b_{xc}^{(l)} + b_{hc}^{(l)}) \tag{40}$$

$$c_t^{(l)} = f_t^{(l)} \cdot c_{t-1}^{(l)} + i_t^{(l)} \cdot g_t^{(l)} \tag{41}$$

Finally, the OG determines the current cell state's exposure to the next hidden state $h_t^{(l)}$ Eq. (43), using output activation $o_t^{(l)}$ Eq. (42).

$$o_t^{(l)} = \sigma(W_{xo}^{(l)} \cdot x_t + W_{ho}^{(l)} \cdot h_{t-1}^{(l)} + b_{xo}^{(l)} + b_{ho}^{(l)}) \tag{42}$$

$$h_t^{(l)} = o_t^{(l)} \cdot \text{LeakyReLU}(c_t^{(l)}) \tag{43}$$

The backward pass commences after the completion of the forward pass, starting from the output gate terminal. The derivative $E_{\delta t}^{(l)} = \frac{\partial E_t}{\partial h_t^{(l)}}$ measures how the loss function changes as the hidden state $h_t^{(l)}$ is adjusted, and it is used to guide parameter updates in the back-propagation process. The loss gradients with respect to the activation of the output gate $o_t^{(l)}$ and the cell state $c_t^{(l)}$ are calculated in the following manner:

$$\frac{\partial E_t}{\partial o_t^{(l)}} = E_{\delta t}^{(l)} \cdot \text{LeakyReLU}(c_t^{(l)}) \tag{44}$$

$$\frac{\partial E_t}{\partial c_t^{(l)}} = E_{\delta t}^{(l)} \cdot o_t^{(l)} \cdot (1 - \text{LeakyReLU}^2(c_t^{(l)})) \tag{45}$$

The gradients related to the activation of the input gate $i_t^{(l)}$ and the forget gate $f_t^{(l)}$ are derived into the following equation:

$$\frac{\partial E_t}{\partial i_t^{(l)}} = \frac{\partial E_t}{\partial c_t^{(l)}} \cdot g_t^{(l)} \cdot \sigma'(i_t^{(l)}) \tag{46}$$

$$\frac{\partial E_t}{\partial f_t^{(l)}} = \frac{\partial E_t}{\partial c_t^{(l)}} \cdot c_{t-1}^{(l)} \cdot \sigma'(f_t^{(l)}) \tag{47}$$

Next, the loss gradient with respect to candidate cell state $g_t^{(l)}$ is determined:

$$\frac{\partial E_t}{\partial g_t^{(l)}} = \frac{\partial E_t}{\partial c_t^{(l)}} \cdot i_t^{(l)} \cdot \text{LeakyReLU}'(g_t^{(l)}) \tag{48}$$

Gradients of weights W and biases b are systematically calculated, pivotal in the training's optimization phase:

$$\begin{aligned}\frac{\partial E_t}{\partial W^{(l)}} = \frac{\partial E_t}{\partial o_t^{(l)}} &\cdot \text{LeakyReLU}(c_t^{(l)}) \cdot \text{concatenate}(h_{t-1}^{(l)}, x_t^{(l)})^T \cdot \\ &\text{diag}(o_t^{(l)} \odot (1 - \text{LeakyReLU}^2(c_t^{(l)}))) \cdot W_{\text{output}}^T\end{aligned} \tag{49}$$

$$\begin{aligned}\frac{\partial E_t}{\partial b^{(l)}} = \frac{\partial E_t}{\partial o_t^{(l)}} &\cdot \text{LeakyReLU}(c_t^{(l)}) \cdot \text{concatenate}(h_{t-1}^{(l)}, x_t^{(l)})^T \cdot \\ &\text{diag}(o_t^{(l)} \odot (1 - \text{LeakyReLU}^2(c_t^{(l)})))\end{aligned} \tag{50}$$

Finally, the derivatives of loss for each weight and bias type are:

$$\frac{\partial E_t}{\partial W_{x*}^{(l)}} = \sum_{*\in\{i,o,c,f\}} \left(\frac{\partial E_t}{\partial *_t^{(l)}} \cdot x_t \right) \tag{51}$$

$$\frac{\partial E_t}{\partial b_{x*}^{(l)}} = \sum_{*\in\{i,f,c,o\}} \left(\frac{\partial E_t}{\partial *_t^{(l)}} \right) \tag{52}$$

This sequence of computations forms the backbone of the backward pass, guiding parameter updates in every RNF. The proposed deep learning model uses exhaustive parameter searches in a predetermined hyperparameter space. A range is specified for each hyperparameter in this procedure, including the length of the input sequence, the number of RNF layers, the units per layer, the batch size, and the window length of every training epoch. The algorithm goes through all the possible combinations of these factors, training a new model instance for each one and checking how well it does on a validation dataset. The combination with the best validation performance is chosen as the optimal set.

Eqs. (53)–(56) evaluate the performance of the proposed model using Mean Absolute Error (MAE), Mean Square Error (MSE), Root Mean Squared Error (RMSE), and the R-squared (R^2) score [32,38,50–52]. The MAE provides a clear understanding of prediction accuracy by measuring the average magnitude of errors between expected and actual values. MSE measures the average squared differences between expected and actual values, emphasizing higher prediction accuracy errors. The RMSE signifies the magnitude of errors, is greater when values between predictions and actuals are increased. The R^2 score represents the goodness-of-fit of the model and shows how much of the variance in the dependent variable is determined by the independent variables. A closer value to 1 indicates a better fit, which emphasizes the model's ability to explain the data.

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{53}$$

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2 \tag{54}$$

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \tag{55}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} \tag{56}$$

where y_i stands for the real values, $\hat{y}_i$ for the predicted values, and $\bar{y}_i$ for the mean of the actual values. n indicates the total number of observations. To maintain accuracy and clarity in the formulation and computation of the error metrics, these symbols are applied uniformly across every metric.

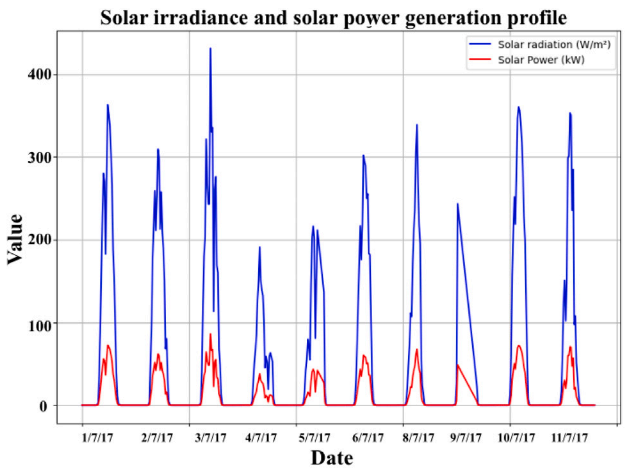


Fig. 3. Daily solar irradiance and solar power output of the microgrid located in NSW, Australia.

3. Results and discussion

The data for this study comes from a small microgrid of 41 different types of residential and commercial buildings in New South Wales, Australia. In this microgrid, rooftop PV and small solar firms installed PV panels spanning 1000 square meters with an estimated overall system efficiency of 20%. A weather station situated approximately 40 kilometers from the microgrid site contributed solar irradiance readings to complement the on-site data. Despite the distance, this station effectively captures solar irradiance conditions pertinent to the geographical location of the microgrid. Fig. 3 shows daily solar irradiance data and computed solar power to show this microgrid's solar energy potential. The VMD method, as explained in Section 2.2.1, is employed to analyze PV generation data to discover the inherent patterns in solar power generation. Fig. 4 shows a series of figures that visually reflect the breakdown results after decomposition. Fig. 4(a) illustrates the four distinct modes obtained from the VMD, visually representing the inherent oscillatory components contained in the input data. Following this, a heatmap is illustrated in Fig. 4(b) that demonstrates the correlation between generated modes and PV generation. It is clear from 4(b) that, the first three modes are closely related to the PV generation data. Fig. 4(c) illustrates the distance correlation among the remaining variables with solar power.

Now, this part depicts and analyzes solar power decomposition using EMD, CEEMDAN and MEMD. This study used an NVIDIA T4 GPU with 16 GB of VRAM to speed up model training and evaluation. The decomposed IMFs of solar power signals using EMD and MEMD, respectively, are shown in Figs. 5(a) and 5(b). Based on this figure, MEMD generates more IMFs than EMD. Due to its robust ensemble methodology, MEMD is capable of extracting more intricate features from the signal. Moreover, there is a noticeable difference in the amplitude variation range of IMFs in MEMD and EMD. MEMD reduces noise and captures the solar power signal's fundamental dynamics more precisely and stably. Consequently, the reduced amplitude variations in MEMD's IMFs provide a smoother and more consistent representation of the basic trends and oscillations in solar power data. Again, the correlation coefficients of the EMD, CEEMDAN, and MEMD-generated IMFs are graphically represented in Figs. 6(a), 6(b), and 6(c) where it is also evident that the MEMD-generated IMFs are better linked with solar power. Nevertheless, to improve computational efficiency, the subsequent analysis focuses on identifying IMFs that demonstrate notable correlations with PV-generated data, particularly those with a correlation coefficient exceeding 0.15. Pearson's correlation coefficient is the metric for assessing the correlation between the generated IMFs and PV generation. Consequently, eleven of the seventeen IMFs and the residual derived from the MEMD decomposition approach are selected for subsequent input processing.

Table 3
Performance Metrics for different clustering algorithms.

Clustering technique	SS	DB Index	CH Index
DBScan	−0.3418	**1.3171**	13.8499
Spectral Clustering	0.1138	2.1207	5179.4616
Gaussian Mixture	0.0176	3.0626	5398.6232
KMeans	**0.1162**	1.4127	**8838.0562**

The following figures (Figs. 7 and 8) show that MEMD performs better in solar power forecasting than VMD, EMD, and CEEMDAN techniques. The evaluation metrics MAE, RMSE, and R-squared score for each of the decomposition methods are shown in Figs. 7(a), 7(b), and 7(c). In each of these metrics, MEMD consistently outperforms its counterparts, underscoring its robustness and precision in extracting relevant features from solar power data. The next graph (Fig. 8) compares actual and forecast PV generation for four days using the strongly correlated modes, or IMFs, derived by each decomposition technique. Additionally, an enlarged view of the most critical points is shown in the boxes for a better understanding of the readers. The model exhibits enhanced performance and possesses a greater capacity to accurately forecast critical or peak points in comparison to the models employing the MEMD decomposition technique. As a result, the subsequent analysis will use the input parameters created by the MEMD decomposition algorithm for model tuning and optimization.

Table 3 compares the performance measures (SS, DB Index, and CH Index) of four clustering techniques: DBScan, Spectral Clustering, Gaussian Mixture, and KMeans. KMeans had the greatest SS (0.1162) and CH Index (8838.0562), suggesting the most distinct and well-defined clusters. Although DBScan has the lowest DB Index (1.3171), its lower SS and CH Indexes indicate less effective clustering overall. Therefore, KMeans is identified as the best clustering algorithm for this dataset due to its superior performance in key metrics.

The following discussion will look at the hyperparameters of the suggested three-layer hybrid data-connected model. First, the number of cells in each layer is adjusted to determine the best number of units per layer. In this experimental approach, the number of units is incrementally increased by 16, spanning from 16 to 96 units. The error metrics for each three-layer cell tuning are shown in Figs. 9(a), 9(b), and 10. Fig. 9(a) demonstrates that the error is smaller for 16 units, but it rapidly increases for 32 units, and it steadily decreases as the number of units increases. However, the result with 96 units is much closer to the result of 16 units, but the higher number of cells necessitates more time for training and prediction stages. For this reason, the ideal number of CNN units is 16. Fig. 9(b) shows similar results, and due to the aforementioned reason, the best number of cells for the second layer is considered as 32. Finally, according to Fig. 10, the errors are lower for 16, 48, and 96 units; however, to guarantee an immediate response, the ideal number of units is considered to be 16. According to this analysis, the 16-32-16 layout will be deemed as the optimal number of units for the proposed three-layer HDLM.

The next step in achieving optimal results during training and testing is to adjust the batch size and window length to the optimal numbers. The performance metrics of the proposed three-layer hybrid model are depicted in Figs. 11(a) and 11(b) for different batch sizes and window lengths. In model training, the batch size is the number of training samples used in each iteration. The small batch size can cause noisy gradient updates, but it can also assist the model in escaping local minima, which could result in improved generalization. Furthermore, larger batch sizes necessitate additional memory. Fig. 11(a) demonstrates that optimal performance is attained with a batch size of 24. So the proposed model's optimal batch size is 24. Following that, the window length indicates the length of the input sequence that the

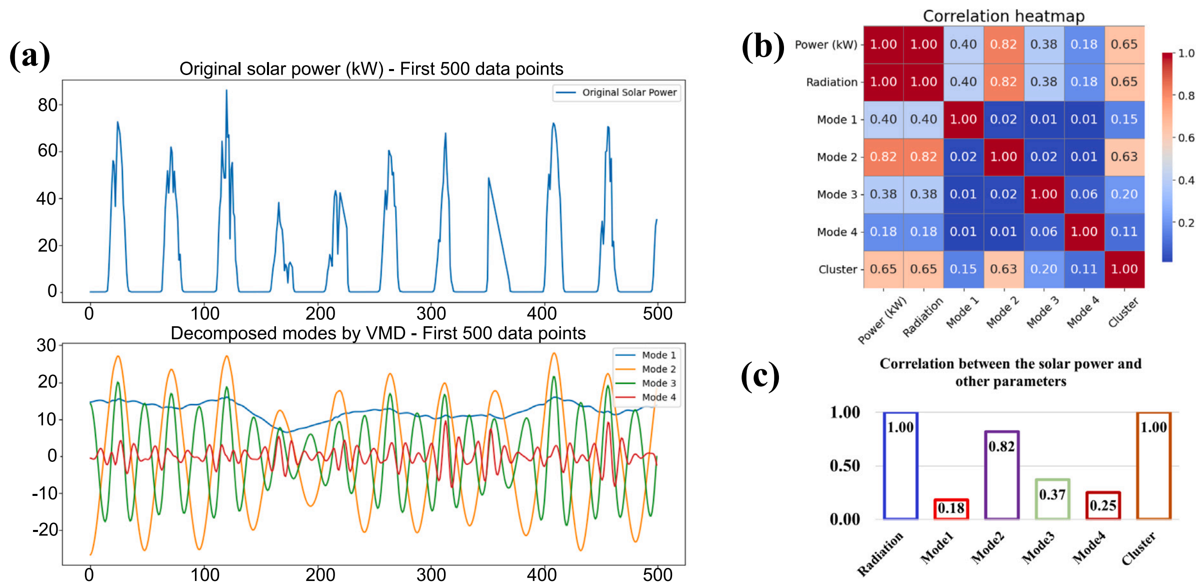
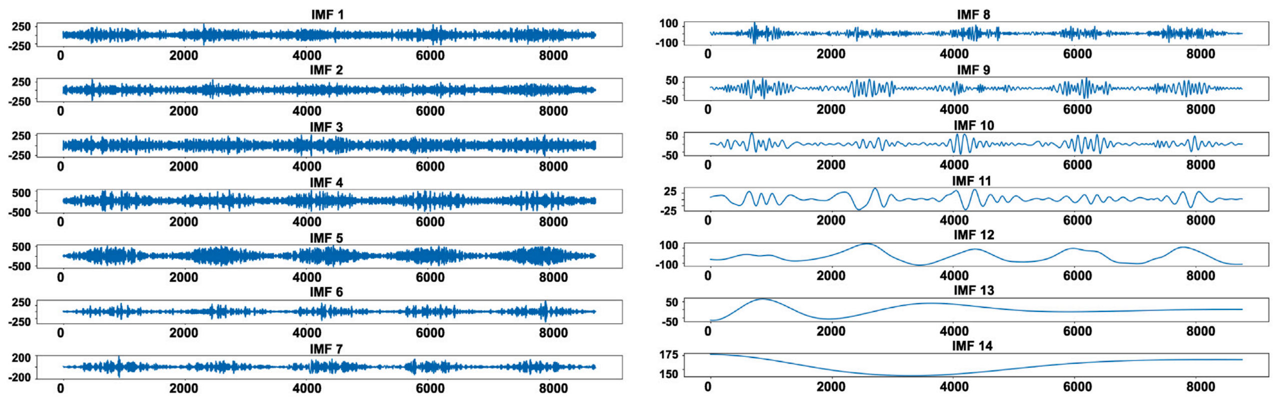
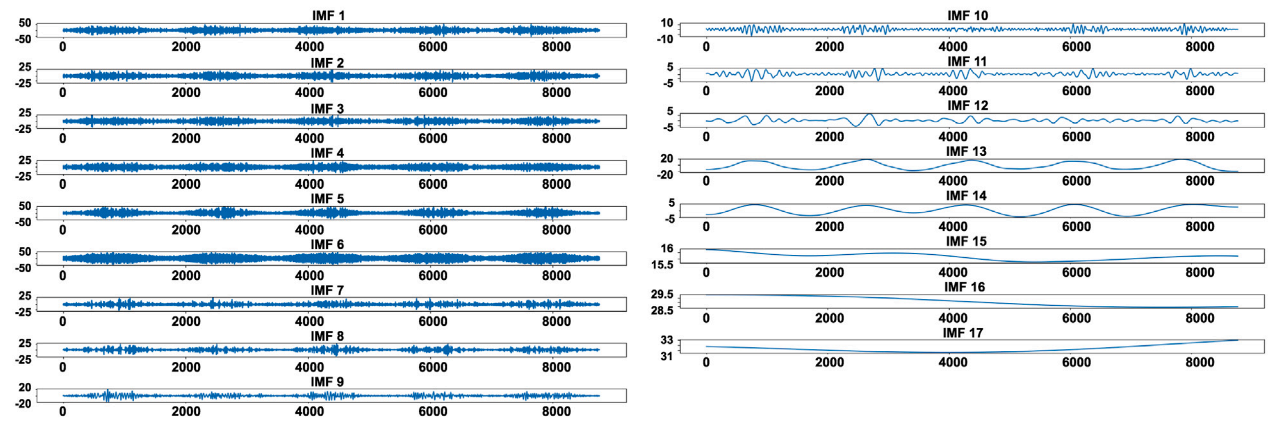


Fig. 4. Graphical representation of (a) mode decomposition after VMD algorithm, (b) correlation heatmap of the variables, and (c) distance correlation map of the variables with solar power.



(a)



(b)

Fig. 5. IMFs of solar power signal as decomposed by (a) EMD and (b) MEMD algorithm for capturing the essential frequency components.

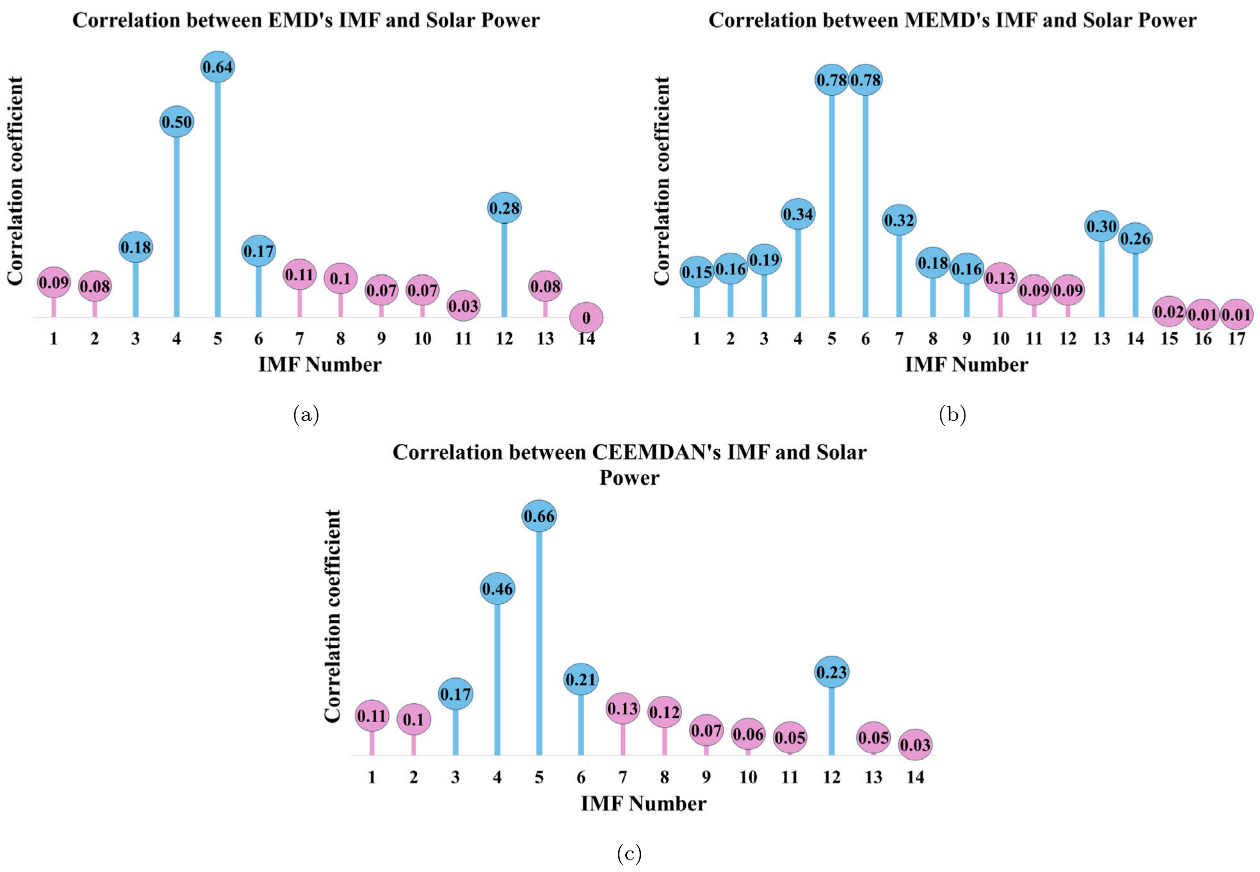


Fig. 6. Correlation of IMFs with solar power signal as decomposed by (a) EMD, (b) MEMD, and (c) CEEMDAN algorithms.

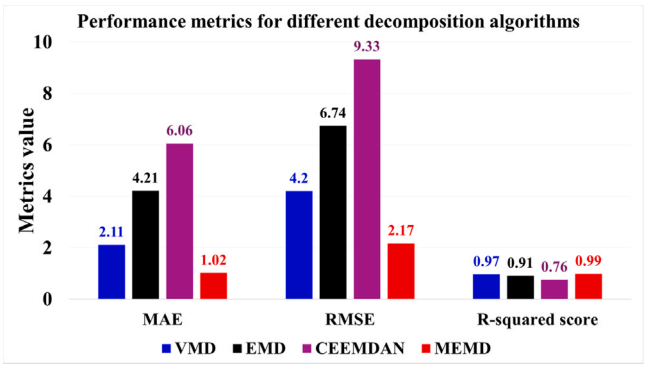


Fig. 7. (a) MAE, (b) RMSE, and (c) R^2 score of the proposed model using the decomposed signals as input.

model considers when making a prediction. Small windows may not supply enough information for the model to produce accurate predictions, while big windows may bring noise or cause overfitting problems. Additionally, larger windows increase the computation required and the intricacy of the model, despite providing more context. The window length varies from 64 to 192, and Fig. 11(b) shows that the smallest error is at 128 and that the error increases for lower and higher values than 128. Therefore, the proposed model is deemed to be optimal with 128 samples per iteration.

The impact of the number of MEMD-generated IMFs on computation time is illustrated in Table 4, which emphasizes the trade-off between training duration and accuracy. It is important to mention that the solar power, irradiance, and cluster information are designated as inputs for each time, with the IMFs number specified in Table 4 as the input. After

Table 4
Performance metrics for different input configurations.

Sl No.	Inputs	Time per epoch (s)	Accuracy		
			MAE	RMSE	R^2 Score
1	All IMF	39	2.5161	3.7238	0.9728
2	IMF 1–14	37	2.0760	3.0870	0.9813
3	IMF 1-10, 13, 14	38	1.4793	2.7297	0.9827
4	**IMF 1-9, 13, 14**	33	**1.0231**	**2.1762**	**0.9907**
5	IMF 4-7, 13, 14	**30**	1.4399	3.1009	0.9670
6	IMF 4–7	**30**	2.3877	4.3729	0.9625
7	IMF 5, 6	31	2.7063	4.9923	0.9511

analyzing the performance metrics for several input configurations, it is clear that the input configuration with serial number 4 (IMF 1-9, 13, 14) produces the most favorable results. The configuration attains the lowest error with an MAE of 1.0231 and an RMSE of 2.1762, accompanied with the highest R-squared score of 0.9907. Although the times per epoch are shorter in configurations 5 to 7, their model performance is significantly inferior. This means that the inputs from serial number 4 (IMF 1-9, 13, 14) offer the most excellent combination of accuracy and efficiency, making them ideal for optimal IMF utilization, even though the training period is slightly longer.

Following the preceding discussion and thorough hyperparameter tuning, the best hyperparameters are appropriately set up, which improves the performance of the reported HDLM. After carefully implementing these ideal parameters, the microgrid's actual and predicted PV generation output is presented in Fig. 12(a). From the aforementioned figure, it is clear that the model can predict future PV generation accurately. Fig. 12(b) displays a regression plot, demonstrating a clear correlation between predicted and actual values, highlighting the model's effectiveness. According to this strong evidence, the model can

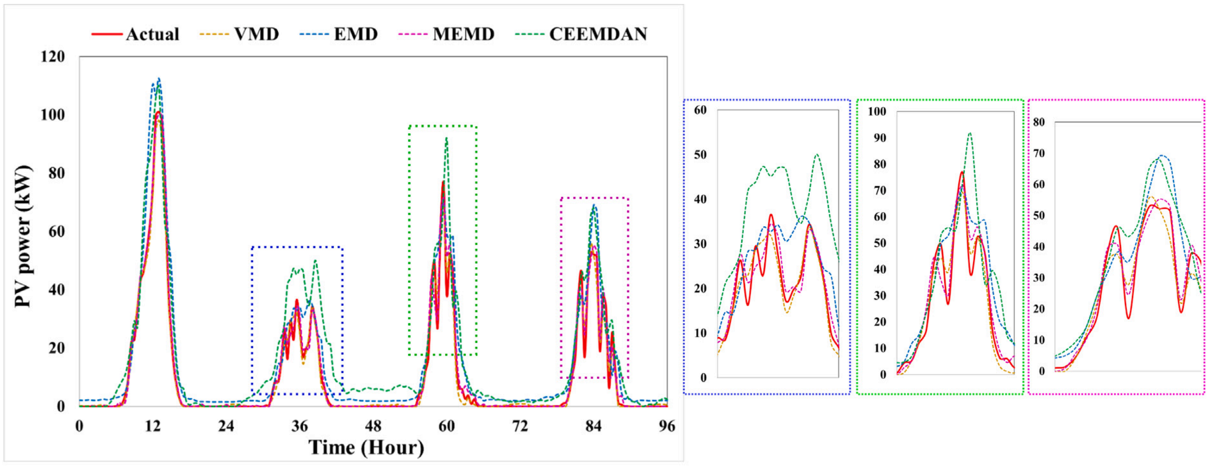


Fig. 8. Comparison of actual and predicted solar power by using different decomposition algorithms.

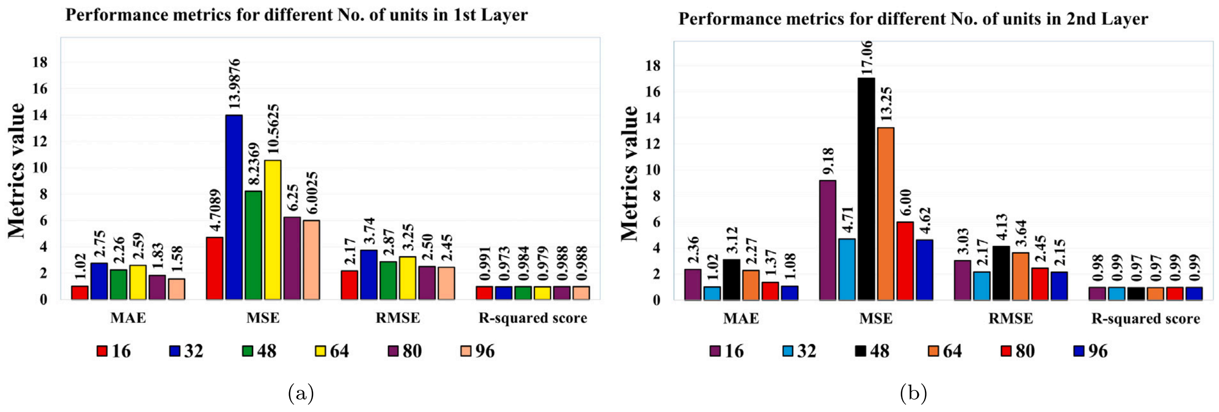


Fig. 9. Evaluation of the performance of the proposed model for determining the optimal number of units of first and second layer.

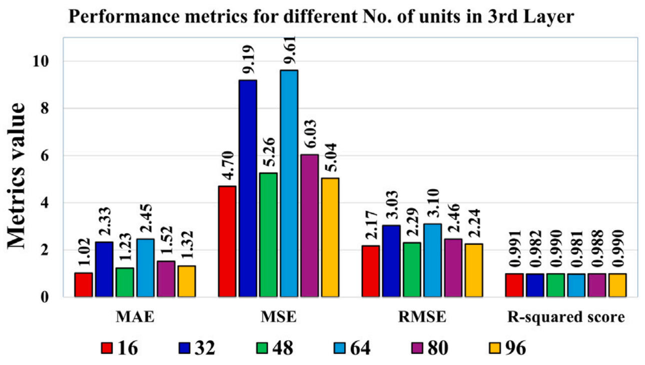


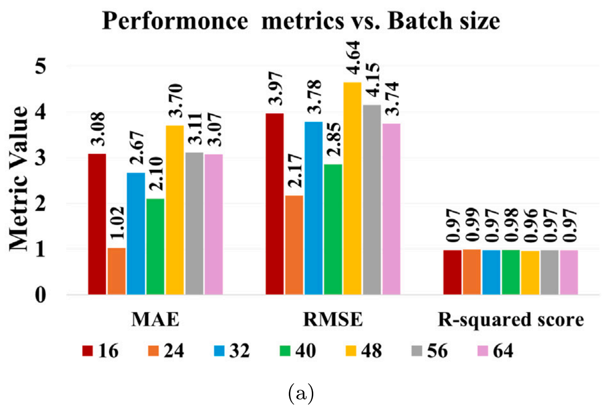
Fig. 10. Performance analysis of the proposed model for different numbers of units in the third layer.

accurately forecast PV generation, which has a significant impact on managing and planning for renewable energy.

Comparing the proposed approach against several advanced deep learning and hybrid models is important for confirming the method's effectiveness. In Fig. 13, the comparison of performance metrics (MAE, MSE, RMSE, and the coefficient of determination (R^2)) shows that the suggested model is better than any other models named CNN, CNN–LSTM, LSTM, Bi-LSTM, Stacked-LSTM, TCN (Temporal Convolutional Network), CNN-BiLSTM, CNN-S-BiLSTM (CNN Stacked BiLSTM). The main model configuration and key hyperparameters are added in Table 5. Here, the figure also indicates that the CNN, CNN-BiLSTM and stacked LSTM models exhibit the closest results of the proposed HDLM. CNN is well-known for its ability to extract trends, whereas Bi-LSTM and stacked LSTM memory cells excel at memorizing and reproducing trend data. A significant distinction between them is that Bi-LSTM utilizes bidirectional processing, whereby the input sequence is processed in both the forward and backward orientations. This distinctive characteristic offers superior forecasting capabilities compared to stacked LSTM. In PV generation data, the mentioned models perform better than others because of the existence of a clear trend or pattern. However, as the proposed model is a hybrid of the two, the outcome is significantly superior to either model alone.

Fig. 14(a) depicts a full comparison, showing both the actual and projected values of the proposed model as well as the compared models. In this graph, the proposed framework predicts sharp or critical points more accurately than the others. Furthermore, Fig. 14(b) digs deeper by displaying the difference curves for each model. This visual representation shows the discrete differences between real and anticipated values and emphasizes the proposed model's superiority in forecasting applications. Table 5 shows isolated microgrid PV generation forecasting model information and performance metrics.

The performance of the proposed model is also evaluated by comparing it with other advanced models, including the Transformer model, Stacked LSTM with Attention Mechanism (S-LSTM-AM), LSTM with Attention Mechanism (LSTM-AM), and Temporal Fusion Transformer (TFT) [53–55]. The comparison results are detailed in Table 6. Notably, the proposed model has a very short training time per cycle of just 33 s, making it faster than the Transformer (49 s), S-LSTM-AM



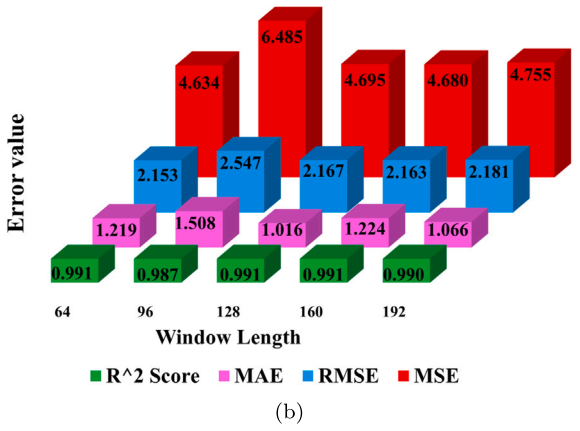
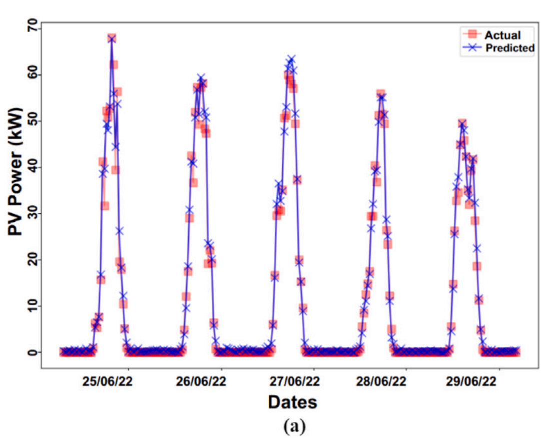


Fig. 11. Performance metrics for various (a) batch sizes and (b) window lengths of the proposed HDLM.



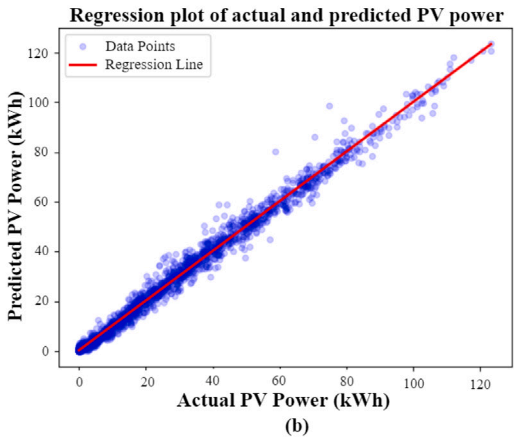


Fig. 12. Graphical representation of (a) actual and predicted values of the proposed HDML, (b) regression plot of the actual and predicted PV generation.

Table 5
Comparison of Different Models based on Model Attributes and Performance Metrics.

Section	Attributes	CNN	C-LSTM	CNN-BL	S-LS.	Bi-LS.	TCN	Prop.
Model	Units	16	16–16	16–32	32–16	32	32	16-32-16
	Learn. rate	0.0008	0.0008	0.0008	0.0008	0.0008	0.0008	0.0008
	Epochs	50	100	50	75	100	50	50
	Window len.	128	128	128	128	128	128	128
	Batch size	24	24	24	24	24	24	24
	Dropout	0.2	0.2	0.2	0.2	0.2	0.2	0.2
	Activation	ReLU	ReLU	Sigm	Sigm	ReLU	ReLU	LReLU
	Optimizer	Adam	Adam	Adam	Adam	Adam	Adam	Adam
	Loss Function	MSE	MSE	MSE	MSE	MSE	MSE	MSE
Error	MAE	1.26	2.11	1.09	1.58	2.96	1.31	**1.02**
	MSE	8.99	7.99	4.84	5.11	12.89	9.36	**4.70**
	RMSE	2.99	2.82	2.20	2.26	3.59	3.06	**2.16**
	R^2 score	0.990	0.984	0.990	0.990	0.975	0.982	**0.991**

Note: C-LSTM = CNN–LSTM, CNN-BL = CNN-BiLSTM, S-LS = stacked LSTM, Bi-LS = Bi-LSTM, Sigm= Sigmoid, and LReLU= LeakyReLU.

(40 s), and TFT (89 s), and only slightly longer than the LSTM-AM (32 s). The proposed model excels in both efficiency and forecasting accuracy. It achieves remarkably low MAE, MSE, and RMSE values of 1.02, 4.70, and 2.16, respectively. Additionally, it boasts the highest R-squared score of 0.991, indicating near-perfect predictive performance. These results underscore the effectiveness of the proposed model in delivering superior performance with reduced computing time.

This study provides solid evidence that the proposed framework significantly improves feature generation by outperforming existing decomposition techniques, resulting in more accurate predictive models. The next points confirm the claimed solar energy forecasting advances and contributions.

Table 6
Performance Comparison of the proposed LRNF with advanced hybrid DL models.

Name	Time	MAE	RMSE	MSE	R^2 score
Transformer	49 s	14.68	18.54	343.69	0.19
S-LSTM-AM	40 s	3.07	3.84	14.74	0.97
LSTM-AM	**32 s**	6.96	9.50	90.31	0.83
TFT	89 s	21.34	23.45	550.02	0.08
Proposed	33 s	**1.02**	**2.16**	**4.70**	**0.991**

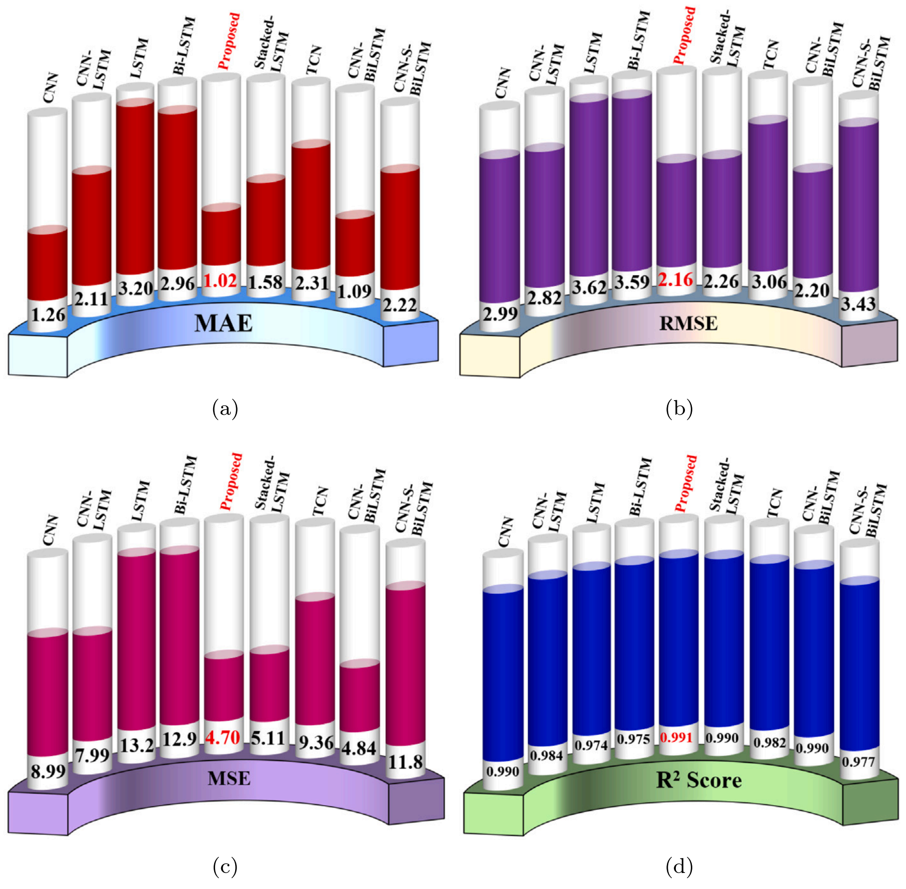


Fig. 13. Comparison of performance metrics for different deep learning models and the proposed HDLM.

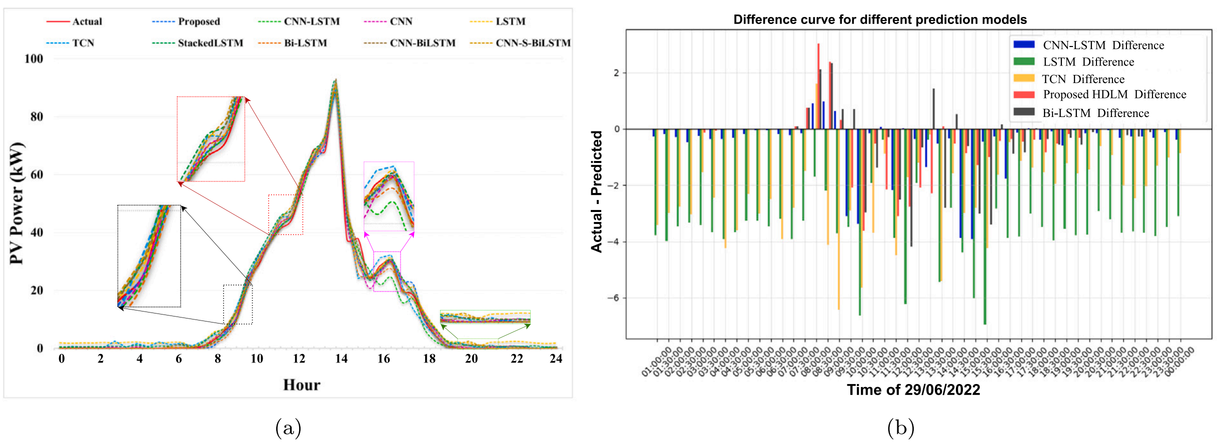


Fig. 14. Graphical representation of (a) actual and predicted PV generation of one day using different models, (b) difference curve between the actual and prediction by using different models.

- The first contribution of my research focuses on advanced feature generation and evaluation which is validated through a comprehensive comparison of various decomposition algorithms. Figs. 6 and 7 demonstrate that the proposed MEMD algorithm significantly outperforms the other techniques, such as VMD, EMD, and CEEMDAN. Additionally, compared to the other algorithms, the IMFs produced by the MEMD algorithm are more closely related to solar power.
- The comparative study presented in Table 3 validates the second contribution of this research, which involves complex feature clustering. After achieving the highest CH Index (8838.06), Silhouette Score (0.12), and competitive DB Index (1.41), the KMeans clustering algorithm was found to be the most effective approach. According to these statistics, the KMeans algorithm clusters better than DBScan, Spectral Clustering, and Gaussian Mixture.
- The next contribution of this research is the effective use of a hybrid deep learning model to predict PV generation quickly using real-world data. This study uses authentic microgrid data, making the model more practical than others that use simulated or machine-generated data. The effectiveness of the proposed framework is thoroughly validated by the analysis of Tables 4, 5, and 6. Table 4 emphasizes the model's capacity to make

quick predictions, while Tables 5 and 6 show the minimal errors achieved compared to other advanced deep learning models.

- The final contribution of this study is benchmark comparisons and technological advancements, confirming the superior accuracy of the proposed model for predicting solar energy. Based on Figs. 13 and 14, Tables 5 and 6, and thorough assessments, this framework significantly outperforms cutting-edge machine learning and deep learning models. Compared to sophisticated models, the proposed framework exhibited lower MAE (1.02), RMSE (2.16), and the highest R-squared score (0.991). Therefore, our model not only meets but surpasses current benchmarks, affirming its significant contribution and emphasizing its influence on future solar energy prediction research.

4. Conclusion & future scope

The present investigation highlights the outstanding HDLM as a reliable approach for predicting solar power generation in remote microgrids using solar irradiance data. Its novel two-stage three-layer architecture, which combines a spatial pattern recognition network with a layered modified recurrent neural framework, improves feature extraction and prediction accuracy. After creating the extensive features, highly correlated feature selection and clustering ensure fast computations of the proposed framework. The proposed HDLM outperforms other single-stage and hybrid deep learning models in terms of MAE, MSE, and RMSE, as well as R^2 score. These results confirm the HDLM's outstanding predictive capability and its potential to transform renewable energy management systems.

Although the study shows promise, there are several limitations. First, solar irradiance data from distant weather stations may cause latency or inaccuracies in real-time applications, affecting model performance. Additionally, the three-layer structure of the HDLM may necessitate the use of costly computers and hardware due to its advanced architecture.

Looking ahead, the study paves the way for more investigations. By using a hybrid decomposition approach, the model's predicted accuracy may be improved, and deeper insights from the solar irradiance data may be discovered. Furthermore, the computation time may be reduced by using sophisticated optimization approaches for feature selection.

CRediT authorship contribution statement

Md. Ahasan Habib: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **M.J. Hossain:** Writing – review & editing, Supervision, Software, Resources, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work, the author(s) used Quillbot to paraphrase the sentences for less similarity indexing. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

References

[1] A.C. Duman, Ö. Güler, Economic analysis of grid-connected residential rooftop PV systems in Turkey, Renew. Energy 148 (2020) 697–711.
[2] A.Z. Gabr, A.A. Helal, N.H. Abbasy, Economic evaluation of rooftop grid-connected photovoltaic systems for residential building in Egypt, Int. Trans. Electr. Energy Syst. 30 (6) (2020) e12379.
[3] F.A. Pramadya, K.N. Kim, Promoting residential rooftop solar photovoltaics in Indonesia: Net-metering or installation incentives? Renew. Energy 222 (2024) 119901.
[4] F. Damayra, T. Khatib, Assessment of innovation policy standards' impact on local development of renewable energy in Palestinian local government units, Renew. Energy 187 (2022) 177–192.
[5] H. Fontenot, B. Dong, Modeling and control of building-integrated microgrids for optimal energy management–a review, Appl. Energy 254 (2019) 113689.
[6] M. Shafiullah, S.D. Ahmed, F.A. Al-Sulaiman, Grid integration challenges and solution strategies for solar PV systems: A review, IEEE Access 10 (2022) 52233–52257.
[7] L. Meegahapola, A. Sguarezi, J.S. Bryant, M. Gu, E.R. Conde D, R.B. Cunha, Power system stability with power-electronic converter interfaced renewable power generation: Present issues and future trends, Energies 13 (13) (2020) 3441.
[8] R. Nandi, M. Tripathy, C.P. Gupta, Coordination of BESS and PV system with bidirectional power control strategy in AC microgrid, Sustain. Energy Grids Netw. 34 (2023) 101029.
[9] S. Mishra, K. Anderson, B. Miller, K. Boyer, A. Warren, Microgrid resilience: A holistic approach for assessing threats, identifying vulnerabilities, and designing corresponding mitigation strategies, Appl. Energy 264 (2020) 114726.
[10] N. Stringer, N. Haghdadi, A. Bruce, I. MacGill, Fair consumer outcomes in the balance: Data driven analysis of distributed PV curtailment, Renew. Energy 173 (2021) 972–986.
[11] N. Tang, Y. Zhang, Y. Niu, X. Du, Solar energy curtailment in China: Status quo, reasons and solutions, Renew. Sustain. Energy Rev. 97 (2018) 509–528.
[12] M. Shafiullah, S.D. Ahmed, F.A. Al-Sulaiman, Grid integration challenges and solution strategies for solar pv systems: A review, IEEE Access 10 (2022) 52233–52257.
[13] C. Scott, M. Ahsan, A. Albarbar, Machine learning for forecasting a photovoltaic (PV) generation system, Energy 278 (2023) 127807.
[14] K. Mahmud, S. Azam, A. Karim, S. Zobaed, B. Shanmugam, D. Mathur, Machine learning based PV power generation forecasting in alice springs, IEEE Access 9 (2021) 46117–46128.
[15] W. VanDeventer, E. Jamei, G.S. Thirunavukkarasu, M. Seyedmahmoudian, T.K. Soon, B. Horan, S. Mekhilef, A. Stojcevski, Short-term PV power forecasting using hybrid GASVM technique, Renew. Energy 140 (2019) 367–379.
[16] P.N.L. Mohamad Radzi, M.N. Akhter, S. Mekhilef, N. Mohamed Shah, Review on the application of photovoltaic forecasting using machine learning for very short-to long-term forecasting, Sustainability 15 (4) (2023) 2942.
[17] D. Kothona, I.P. Panapakidis, G.C. Christoforidis, A novel hybrid ensemble LSTM-FFNN forecasting model for very short-term and short-term PV generation forecasting, IET Renew. Power Gener. 16 (1) (2022) 3–18.
[18] D. Kim, D. Kwon, L. Park, J. Kim, S. Cho, Multiscale LSTM-based deep learning for very-short-term photovoltaic power generation forecasting in smart city energy management, IEEE Syst. J. 15 (1) (2020) 346–354.
[19] B. Liu, C. Song, Q. Wang, Y. Wang, Forecasting of China's solar PV industry installed capacity and analyzing of employment effect: based on GRA-BiLSTM model, Environ. Sci. Pollut. Res. 29 (3) (2022) 4557–4573.
[20] W. Lin, B. Zhang, H. Li, R. Lu, Multi-step prediction of photovoltaic power based on two-stage decomposition and BILSTM, Neurocomputing 504 (2022) 56–67.
[21] S. Ghimire, R.C. Deo, H. Wang, M.S. Al-Musaylh, D. Casillas-Pérez, S. Salcedo-Sanz, Stacked LSTM sequence-to-sequence autoencoder with feature selection for daily solar radiation prediction: a review and new modeling results, Energies 15 (3) (2022) 1061.
[22] S.-C. Lim, J.-H. Huh, S.-H. Hong, C.-Y. Park, J.-C. Kim, Solar power forecasting using CNN-LSTM hybrid model, Energies 15 (21) (2022) 8233.
[23] Y. He, Q. Gao, Y. Jin, F. Liu, Short-term photovoltaic power forecasting method based on convolutional neural network, Energy Rep. 8 (2022) 54–62.
[24] A. Rai, A. Shrivastava, K.C. Jana, A CNN-BiLSTM based deep learning model for mid-term solar radiation prediction, Int. Trans. Electr. Energy Syst. 31 (9) (2021) e12664.
[25] N. Elizabeth Michael, M. Mishra, S. Hasan, A. Al-Durra, Short-term solar power predicting model based on multi-step CNN stacked LSTM technique, Energies 15 (6) (2022) 2150.
[26] Z.A. Khan, T. Hussain, S.W. Baik, Dual stream network with attention mechanism for photovoltaic power forecasting, Appl. Energy 338 (2023) 120916.
[27] Z. Zhang, J. Wang, D. Wei, Y. Xia, An improved temporal convolutional network with attention mechanism for photovoltaic generation forecasting, Eng. Appl. Artif. Intell. 123 (2023) 106273.
[28] J. Chen, L.Q.R. Ooi, T.W.K. Tan, S. Zhang, J. Li, C.L. Asplund, S.B. Eickhoff, D. Bzdok, A.J. Holmes, B.T. Yeo, Relationship between prediction accuracy and feature importance reliability: An empirical and theoretical study, NeuroImage 274 (2023) 120115.

[29] M. Mishra, P.B. Dash, J. Nayak, B. Naik, S.K. Swain, Deep learning and wavelet transform integrated approach for short-term solar PV power prediction, Measurement 166 (2020) 108250.
[30] T. Zhang, C. Lv, F. Ma, K. Zhao, H. Wang, G.M. O'Hare, A photovoltaic power forecasting model based on dendritic neuron networks with the aid of wavelet transform, Neurocomputing 397 (2020) 438–446.
[31] C.-R. Chen, F.B. Ouedraogo, Y.-M. Chang, D.A. Larasati, S.-W. Tan, Hour-ahead photovoltaic output forecasting using wavelet-ANFIS, Mathematics 9 (19) (2021) 2438.
[32] S. Netsanet, D. Zheng, W. Zhang, G. Teshager, Short-term PV power forecasting using variational mode decomposition integrated with Ant colony optimization and neural network, Energy Rep. 8 (2022).
[33] L. Wang, Y. Liu, T. Li, X. Xie, C. Chang, Short-term PV power prediction based on optimized VMD and LSTM, IEEE Access 8 (2020) 165849–165862.
[34] S. Wang, L. Wei, L. Zeng, Ultra-short-term photovoltaic power prediction based on VMD-LSTM-RVM model, in: IOP Conference Series: Earth and Environmental Science, vol. 781, IOP Publishing, 2021, 042020.
[35] H.K. Yadav, Y. Pal, M.M. Tripathi, Short-term PV power forecasting using empirical mode decomposition in integration with back-propagation neural network, J. Inform. Optim. Sci. 41 (1) (2020) 25–37.
[36] M.K. Behera, N. Nayak, A comparative study on short-term PV power forecasting using decomposition based optimized extreme learning machine algorithm, Eng. Sci. Technol. Int. J. 23 (1) (2020) 156–167.
[37] Y. Bao, W. Guo, Photovoltaic power prediction based on EMD-BLS model, in: J. Phys. Conf. Ser., 2427, (1) IOP Publishing, 2023, 012016.
[38] R. Khelifi, M. Guermoui, A. Rabehi, A. Taallah, A. Zoukel, S.S. Ghoneim, M. Bajaj, K.M. AboRas, I. Zaitsev, et al., Short-term PV power forecasting using a hybrid TVF-emd-ELM strategy, Int. Trans. Electr. Energy Syst. 2023 (2023).
[39] Y. Jiang, L. Zheng, X. Ding, Ultra-short-term prediction of photovoltaic output based on an LSTM-ARMA combined model driven by EEMD, J. Renew. Sustain. Energy 13 (4) (2021).
[40] T. Zhang, X. Zhang, T.K. Chau, Y. Chow, T. Fernando, H.H.-C. Iu, et al., Highly accurate peak and valley prediction short-term net load forecasting approach based on decomposition for power systems with high PV penetration, Appl. Energy 333 (2023) 120641.
[41] P. Singh, S.D. Joshi, R.K. Patney, K. Saha, The Fourier decomposition method for nonlinear and non-stationary time series analysis, Proc. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci. 473 (2199) (2017) 20160871.
[42] G. Sideratos, A. Ikonomopoulos, N.D. Hatziargyriou, A novel fuzzy-based ensemble model for load forecasting using hybrid deep neural networks, Electr. Power Syst. Res. 178 (2020) 106025.
[43] J. Zhang, Y. Zhang, Forecast of photovoltaic power generation based on DBSCAN, in: E3S Web of Conferences, vol. 236, EDP Sciences, 2021, p. 02016.
[44] N.G. Trillos, D. Slepčev, A variational approach to the consistency of spectral clustering, Appl. Comput. Harmon. Anal. 45 (2) (2018) 239–281.
[45] L.E. Ekemeyong Awong, T. Zielinska, Comparative analysis of the clustering quality in self-organizing maps for human posture classification, Sensors 23 (18) (2023) 7925.
[46] A. Ullah, M.I. Mohmand, H. Hussain, S. Johar, I. Khan, S. Ahmad, H.A. Mahmoud, S. Huda, Customer analysis using machine learning-based classification algorithms for effective segmentation using recency, frequency, monetary, and time, sensors 23 (6) (2023) 3180.
[47] M.M. Ahsan, M.P. Mahmud, P.K. Saha, K.D. Gupta, Z. Siddique, Effect of data scaling methods on machine learning algorithms and model performance, Technologies 9 (3) (2021) 52.
[48] V. Gómez-Escalonilla, P. Martínez-Santos, M. Martín-Loeches, Preprocessing approaches in machine-learning-based groundwater potential mapping: an application to the Koulikoro and Bamako regions, Mali, Hydrol. Earth Syst. Sci. 26 (2) (2022) 221–243.
[49] J.A. Franklin, Jazz melody generation from recurrent network learning of several human melodies, in: FLAIRS Conference, 2005, pp. 57–62.
[50] M.Y. Junior, R.Z. Freire, L.O. Seman, S.F. Stefenon, V.C. Mariani, L. dos Santos Coelho, Optimized hybrid ensemble learning approaches applied to very short-term load forecasting, Int. J. Electr. Power Energy Syst. 155 (2024) 109579.
[51] L. de Azevedo Takara, A.C. Teixeira, H. Yazdanpanah, V.C. Mariani, L. dos Santos Coelho, Optimizing multi-step wind power forecasting: Integrating advanced deep neural networks with stacking-based probabilistic learning, Appl. Energy 369 (2024) 123487.
[52] N. Kaneko, K. Okazawa, D. Zhao, H. Nishikawa, I. Taniguchi, H. Murayama, Y. Yura, M. Okamoto, F. Catthoor, T. Onoye, Non-intrusive thermal load disaggregation and forecasting for effective HVAC systems, Appl. Energy 367 (2024) 123379.
[53] J. Pospíchal, M. Kubovčík, I. Dirgová Luptáková, Solar irradiance forecasting with transformer model, Appl. Sci. 12 (17) (2022) 8852.
[54] T. Yang, B. Li, Q. Xun, LSTM-attention-embedding model-based day-ahead prediction of photovoltaic power output using Bayesian optimization, IEEE Access 7 (2019) 171471–171484.
[55] M. López Santos, X. García-Santiago, F. Echevarría Camarero, G. Blázquez Gil, P. Carrasco Ortega, Application of temporal fusion transformer for day-ahead PV power forecasting, Energies 15 (14) (2022) 5232.