https://cdn.discordapp.com/attachments/1001852267403685978/1002150855665008681/1-s2.0-S221313882100878X-main.pdf

Sliding window:

* variables converted into 12 steps lag datasets to train model
* outputs transformed by first-order differencing

evaluation parameters (maximum absolute error, confidence interval (CI), linear regression plot, mean absolute error (MAE), root mean square error (RMSE), mean absolute percentage error (MAPE), and R squared) showed that the deep hybrid model consisting of convolutional neural network-long short term memory (CNN-LSTM) outperforms in multistep forecasting (reliable+high accuracy)

normal solar irradiance forecasting statistical models show poor robustness when correlation between parameters is not strong (not able to capture non-linearity accurately -> use ML)

ML models used for forecasting in the last years: artificial neural network (ANN) [26], support vector machine (SVM) [27], random forest (RF) [28], multi-layer perceptron (MLP) [29], feedforward neural network (FNN) [30], back-propagation neural network (BPNN) [31], extreme learning machine (ELM)  
3 techniques: neural network (NN), Gaussian process, and SVM

ML – mostly only one or no hidden layer to predict SI – drawbacks:

* need appropriate input data to make model reliable
* less generalization -> overfitting, gradient vanishing, network training explosion
* /w huge datasets unstable and slow models

To overcome: deep learning network (multiple processing levels)

Superior performance for time series, classification and regression-based forecasting

Most powerful DNN (deep neural network) tools: long short-term memory (LSTM), 1 dimensional convolutional neural network (CNN1D), gated recurrent unit (GRU), and hybrid models

Deep CNN (part of FNN) good for image, audio, and video processing  
Deep LSTM (advanced RRN = recurrent NN) for text, speech and time series where data is in subsequential form

DL – more hidden layers = more parameters to optimize = training requires more time  
but DL is more efficient for forecasting

LSTM introduced for first term prediction + 1dimensional CNN model to reduce time series forecasting problems

LSTM to handle time dependence  
CNN can extract spatial features from meteorological parameters

Hybrid models: LSTM + CNN -> extract temporal and special characteristics for time series prediction analysis – better than single models (for generalization and performance)

Wavelet packet decomposition (WPD) is used with hybrid deep models for 1-hr ahead SI prediction with multivariable inputs.  
Best DL model: WPD based CNN-LSTM-MLP model  
CNN can extract input characteristics from predictive variables very nicely whereas LSTM absorbs them for prediction

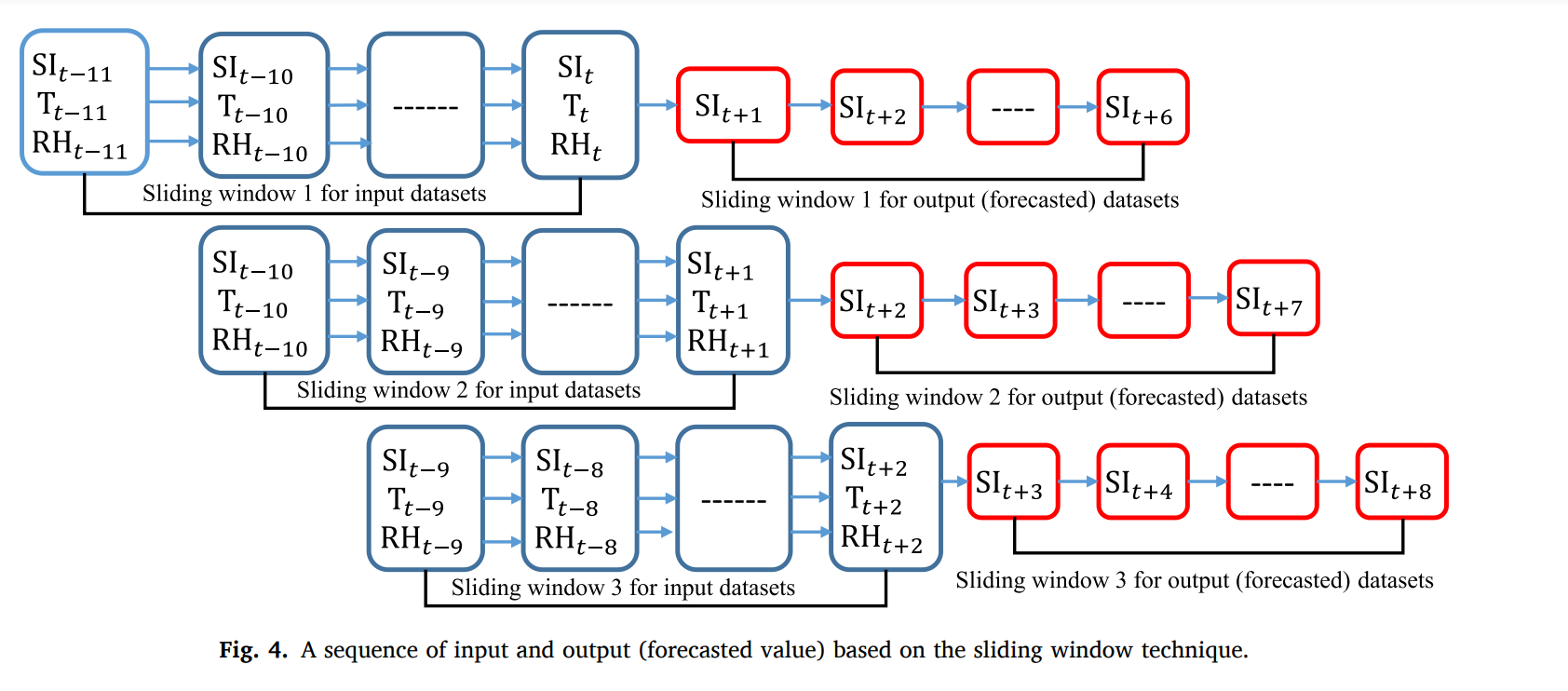
Hybrid models have been applied frequently for short term prediction (30 min – 1h) but not for very short-term (15m)

aims of the researcher who wrote the paper:

* address very short-term (15min) forecasting
* convert a time-series problem into a supervised learning problem for ML-based models (sliding window technique with a window size of 12 hours)
* make time-series data stationary for better prediction results (first-order differencing during the data processing stage)
* evaluate deep CNN-LSTM model

3 parameters – average in 15-minute interval, split in 80% training & 20% testing

Sliding window technique – SI forecasting = time series analysis in supervised learning problem for ML models  
datasets can be framed into supervised learning for multivariate data and multi-step forecasting

3 input variables, window size 12 (12 step lag), want predict 6 steps (1 step = 15min) ahead  
each window used to train and update mode  
when it completes each computation it shift to a new position by one step

He used 12 step lag input and 6 step ahead output

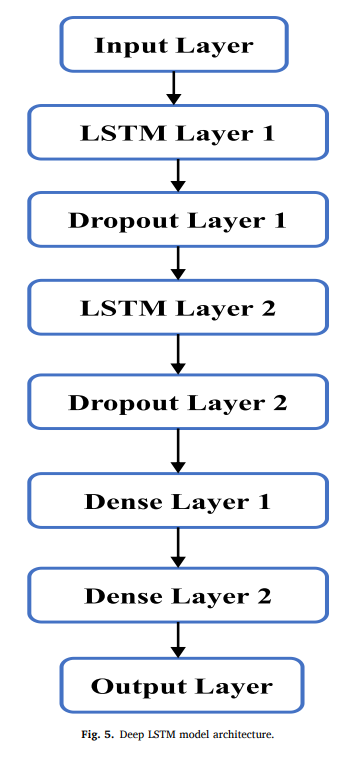
time series datasets usually contain trends that are responsible for a varying mean and seasonality which is responsible for a changing variance over time, therefore time-series datasets are called nonstationary

differencing for data transformation -> make datasets stationary

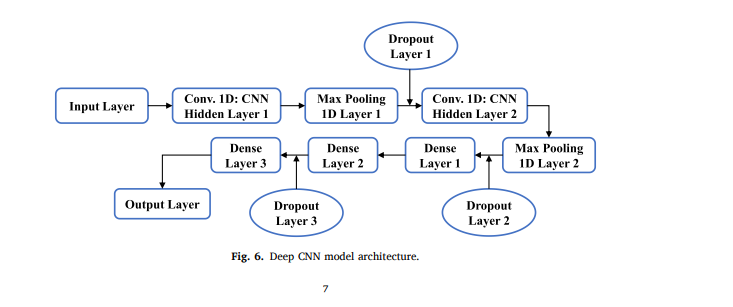
by making dataset stationary mean and variance are stable so model makes better predictions

Therefore, in the present work, we are introducing first-order differencing for output datasets during the training phase where the previous observation will be subtracted from the current observation while during the testing (prediction) phase we are using the inverting process for converting back these difference datasets into its original scale.

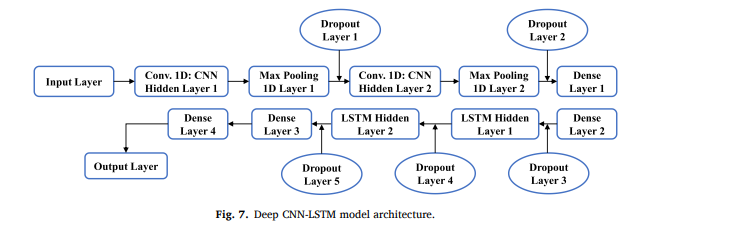
**Deep LSTM model**: 8 total layers including input and output (2 hidden layer+dropouts+2dense layers)  
LSTM layers 1&2 are fully connected layers that help to learn data sequence  
outputs layers help overcome over-fitting   
dense layers help linearize output  
At the initial stage, the three variables (average SI, average T, and average RH) are directly fed to the input layers with a lagged sequence of the previous 12 steps (3 h) and directly fed to the first hidden LSTM Layer 1. Before feeding the input data to the hidden layers, the shape of the data must be of 3 dimensional (sample, time steps, and features) format. With the dropout of 20% for both of the hidden LSTM layers, some random neurons are dropped during the training phase which helps to overcome the overfitting issue of the data. The dense layer performs linear operation before it finally approaches the output layer. For hidden LSTM layers we are using default activation functions of LSTM architecture which are tanh and sigmoid functions whereas for output we are using linear activation functions. Here, “MSE” is considered for the error evaluation while the number of epochs and batch size is selected as 500 and 128 respectively during the entire training process based on trial and error procedure. Fig. 5 shows the architecture of the proposed deep LSTM model for the present study



**Deep CNN model**A deep CNN model that has been introduced for the experiment in the present study, consists of two hidden layers (Conv. 1D layer) along with two max-pooling layers, three dropout layers, and three dense layers. To reduce the possibility of over-fitting, here also we are using dropout with a value of 10% for both of the hidden layers (Conv. 1D layer) and after 2 dense layers. For all hidden and dense layers, we are using rectified linear unit (ReLU) as an activation function whereas, for output, a linear activation function is used. Based on “MSE” we are measuring the error during the training while the number of epochs and batch size is selected as 500 and 512 respectively during the entire training process based on trial and error procedure. Fig. 6 shows the architecture of the proposed deep CNN model.



**Deep CNN-LSTM hybrid model**This model consists of CNN and LSTM architecture together. The CNN model works with the same principle as described in the previous section 2.3.2. However, for the LSTM model, the inputs are provided from previous dense layers which are getting inputs from CNN pooling layers. Here, the 1D convolutional layer of CNN is utilized for classification and pattern extraction from the input datasets while the LSTM helps to predict the input sequence. Dropout values for LSTM layers are selected as 20% whereas for the rest of the hidden layers it is 50% based on trial and error procedure. For Conv. 1D layer the selected activation function is ReLU, whereas for LSTM the default activation functions are used. For output, we are again using the linear activation function. “MSE” is the error measurement criterion while the number of epochs and batch size is selected as 500 and 128 respectively during the entire training process based on trial and error procedure. Fig. 7 displays the architecture of the proposed deep CNN-LSTM model.

****

Use Adam optimizer /w learning rate of 0.001  
evaluation: RMSE, MAE, MAPE, R-squared

