Enhancing Weather Forecasting with Machine Learning and Deep Learning: A Comparative Study of Predictive Models

Gracy Markan[1*] and Shivani Kamboj[2]

Department of Mathematics, Chandigarh University, Mohali, Punjab, India
*gracymarkan@gmail.com, kambojshivani12@gmail.com

Abstract. Weather prediction plays a central role in the welfare of the society and daily economic activities. Accurate conventional NWP methods are currently widely employed but are associated with constraint in managing the non-linearity of the ambiance. This paper focuses on realizing the potential of supervised Machine Learning (ML), and Deep Learning (DL) to improve weather forecasting. A comparison of several selected models of ML and DL is made based on several factors, including accuracy of prediction, model stability, and computational complexity. The approach is validated in the context of a real scenario, showcasing on how deep learning models can outperform traditional methods achieving higher accuracy and better coping with atmospheric conditions, especially when hybrid models are employed.

Keywords: Weather Forecasting, Machine Learning, Deep Learning, Predictive Models, Atmospheric Data

1 Introduction

Forecasting weather plays an integral role in many activities including agricul- ture, disaster management, transportation, and even the control of energy and electricity supplies across regions. Weather forecasting plays an integral role in any operations functions management, planning or working with external envi- ronments dominated by adverse weather. In recent times, however, numerical weather forecasting has been the dominant form of weather prediction based on physical principles incorporating mathematical equations. These methods, how- ever pragmatic they may be, have some limitations in that they are burdensome computationally and also the non-linearities particularly in the atmospheric pro- cesses are difficult to model [1].

There has been improvement in the models used for the prediction of the weather as a result of the incorporation of the Machine Learning (ML) and the Deep Learning (DL) systems. In the case of ML and DL models, as opposed to NWP models, functional form between input and output data is not defined in advance, which is impossible for the input-output relationship in the presence of chaotic nonlinear weather systems [2]. Some of the Machine Learning mod- els that have been used in weather prediction include Decision Trees, Random

Forests, and Support Vector Machines. Random Forest (RF) is a more sophisti- cated implementation of Decision Trees, where several trees are built to improve accuracy and reduce over-fitting. Similarly, Support Vector Machine (SVM) is an advanced technique which is also adopted in predicting temperature and humid- ity since it can be incorporated in regression analysis [3]. These models, however, tend to perform poorly in spatio-temporal problems because they tend to treat time and space as separate entities and do not incorporate their interdependence. Deep Learning models have proved promising in breaking these bottlenecks.
Convolutional Neural Networks are being engaged for the extraction of spatial features based on meteorological data, while long short-term memory networks are ideal for handling temporal dependencies [4]. Shi et al. [5] demonstrated ConvLSTM, a variant of LSTM, for spatio-temporal data can perform well in precipitation now-casting, and finally, it presents a great advancement of DL- based weather prediction. An increasing volume of literature is surfacing in ML and DL models that are being touted for weather forecasting, and, quite in- tuitively, it points toward the possibility of delivering more accuracy than the traditional methods. Studies have shown that DL models outperform traditional statistical and ML approaches, mainly on large, complex datasets [6].

This paper aims at performing a thorough comparative analysis of some of the most widely applied ML and DL models to be used for weather forecasting. The main goal is to compare the models and allow an idea of which model offers the best balance between accuracy and computational efficiency. Evaluation of performance on historical weather data for Random Forest, SVM, CNN, LSTM, and hybrid CNN-LSTM also provides insight in making an appropriate selection of predictive models for different needs in weather forecasting.

2 Literature Review

The integration of Machine Learning and Deep Learning in the prediction of weather has been of interest in the last decade. Numerical models of the current type include the physical equations of motion and other physical processes, and use large computational power, like the ECMWF. [1].

2.1 Machine Learning Approaches

Weather prediction has been done by Machine Learning models since they have the ability to take data and find out the pattern. From the many Machine Learn- ing models used for weather predicting, one could name the Random Forest (RF) or Support Vector Machine (SVM). The RF, or Random Forest, is a machine learning technique that constructs a plethora of decision trees on the training, then merge the prediction of these trees into one final, output decision [6]. The method has been applied for the prediction of rainfall, temperature and other meteorological variables with some degree of reliability [7]. Nevertheless, the ML models do have some restrictions regarding the spatial as well as the tempo- ral aspects of the weather data.Weather by its nature is spatial and temporal;

that is, observations at different places at different time are related. RF and SVM algorithms, for example, do not contain a mechanism for learning such dependencies, which can be unbeneficial in weather prediction [8].

2.2 Deep Learning Techniques

Deep Learning models have demonstrated to represent complex, nonlinear relationships well in large datasets and are therefore excellent for weather fore- casting applications. CNNs have been successfully applied to extract spatial features from meteorological data such as temperature and precipitation grids. The models can be very efficient in identifying spatial patterns that make them a good choice for a weather condition prediction system across different regions [9]. Other well-accepted DL models for weather forecasting include LSTM net- works. LSTMs are a type of RNN which learn the long-term dependencies in addition to time-series forecasting. It can learn long-term dependencies but ef- fective, Hochreiter and Schmidhuber [10] came up with LSTMs as a solution to the vanishing gradient problem encountered by traditional RNNs. LSTMs have been applied for the prediction of different meteorological parameters including temperature, wind speed, and precipitation with promising results [11].

2.3 Comparative Studies

The research indicates that when the models of deep learning and machine check- ing are compared, the model used in one of the descriptions above based on deep learning proved to be the most effective in handling large data sets and weather-related forecasts. In addition, DL models can learn features from raw data automatically, allowing for a significantly reduced need for a large amount of feature engineering as is typically needed in ML models [12]. It has its own set of challenges. Training deep networks is quite resource intensive and in resource constrained settings, not exactly an easy option. DL models are also much less interpretable than the more traditional ML models, and so they may not find a lot of favor with meteorologists and other domain experts [13].

3 Methodology

3.1 Data collection

Perhaps the most essential of all the practices in building up a strong weather forecasting model is data gathering. In this study, historical data on weather such as temperature, humidity, wind speed and precipitation were obtained from the publicly available meteorological datasets, GHCN [7]. The observations were made for several years at different sites, and in different climatic conditions, which provided a diverse set of training and testing samples for the models.

3.2 Data Preprocessing

Therefore, the quality of data is a determinant factor of the performance of the predictive models. In the data preparation phase, several steps were done before feeding a dataset to the modeling process. The missing values were, therefore, appropriately dealt with by applying the right imputation techniques so that the dataset was made complete. To make all features of the same scale, data normalization scales all features to the same range which is very important for most of the machine learning models that are sensitive to the features magnitude. Feature selection was also done in order to decrease the number of features in the dataset as well as to increase the efficiency of the model.

3.3 Model Implementation

3.3.1 Machine Learning Models:

- **Random Forest (RF):** RF is a supervised learning approach in which a large number of decision trees are created, and their predictions are combined in order to enhance accuracy and reduce variance. This was necessary as the model was over-fitting. Hyperparameter tuning was used to determine the optimum number of trees and also the optimum depth of the trees. [6]
- **Support Vector Machine (SVM):**is also an algorithm that performs op- erations of Classification and Regression on the data which is not linear. For the present analysis, the RBF kernel was implemented. So there arose a hy- perparameter which is the regularization parameter and the kernel coefficient also has to be adjusted. [3].

3.3.2 Deep Learning Models:

- **Convolutional Neural Networks (CNN):** This method was also imple- mented to capture the spatial domain on the meteorological pictures. This particular design first has a convolutional layer on the top and this is fol- lowed by a pooling layer which reduces the output in size thus minimizing the computational costs. [9]
- **Long Short-Term Memory (LSTM):** Although LSTMs allow for predic- tion of trends in time series data; in this case, they were used to anticipate the duration's of normal weather. The LSTM networks also have several more layers to ensure that the best learnt model is capable of managing time series data where the determining dependencies are quite distant from one another.
- **CN-LSTM** Similarly, a Hybrid model of this type was also constructed to address the spatial and temporal issues. In the CNN layers, the spatial data was included while in the LSTM layers the tim Series data was included. Additionally, the model became spatial and temporal adaptive to the weather data. [5]

3.4 Testing of the Models

The models were ranked based on the two primary metrics, predictability and computing efficiency. For every model, the performance was defined using the following measures: MSE adn RMSE. They are regression measures which measure similarity between two numerical datasets in a prediction problem with a fixed number of target values. [14]. In the case of ML models, such as RF and SVM, the grid search was performed in their hyperparameters to guarantee that they generate the best performance. In addition, various optimizers such as Adam, RMSprop were used to train different DL models with early stopping to prevent over-fitting.

4 Model Designing

The main objective of this paper is to use LSTM network model which is a form of Recurrent Neural Network designed in order to avoid vanishing gradients and take advantage of the sequential data's long term dependencies.

4.1 Model Architecture and Design of LSTM

Various layers have been incorporated in the architectures of the LSTM model, particularly for the time series component and the non-linearity in the weather data. This includes:

– **Input layer:** This layer then passes the preprocessed weather data where each of the input vectors contain a sequence of time steps say the last 24 hours weather data. Each of the input features has a direct corresponding meteorological variable like temperature, humidity, wind speed, and atmo- spheric pressure.
– **LSTM layers:** The model core is made of two LSTM layers with 128 mem- ory cells in each of them, connected one below the other. All of these learn temporal dependencies through maintaining a hidden state over consecutive time steps. In LSTM cells, there are input gate, forget gate, and output gate that enables it to decide what to remember or forget in the pattern. Employing these two LSTM layers enables complex temporal relations to be described since the first layer LSTM outputs are used as inputs to the second layer LSTM. In fact, even when via the use of cross-layer architecture one gives the model long inputs, it still manages to learn thanks to structure stacking.
– **Dropout regularization:** The problem of overfitting was dealt with by us- ing Dropout, which is simply inserting dropout layers in between the LSTM layers with a dropout of 0.2. Using Dropout whenever we choose to ignore some fraction of the input units by setting them to zero even in testing im- proves the model generalization by lessening the model's reliance on certain neurons.

– **Dense layer:** This section was added because LSTM layers output should be converted to yield the final forecast. Given that this is a regression problem and the target outcome is an ever changing value such as temperature or rainfall [16]
– **Output Layer:** The forecasted weather parameter (e.g., temperature at the next time instant) is delivered by the last hidden layer. The model focuses in order to achieve this is set to close the gap between the predicted and the true values as much as possible.

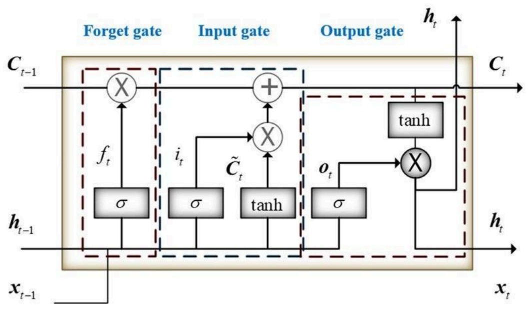


Fig. 1: Architecture of LSTM

Figure 1 shows the full structure of the proposed LSTM architecture[22], where the direction of input sequences being processed through LSTM cells, dropout layers and a dense output layer is also provided.

4.2 Input and Feature Engineering

The LSTM model was fed with the input data in the form of time []. The input data can be in the time measure of hours, days or even years depending on the variables for instance, temperature, humidity, wind speed, pressure, etc. That is why, Feature engineering was used so that only the needed data is fed into the model, without any irrelevant information. The input data was normalized to a range from 0 to 1 based on Min-Max Scale which is known to improve the convergence of models during training [17]. A group of Domain specialists and correlation analysis techniques worked together to Wreak Feature selection. Variables whose correlation with a temperature variable, for example, the target temperature variable, was high were kept while features that did not add value were filtered out to avoid overfitting of the model [18].

4.3 Loss Function and Optimization

The LSTM model was trained on the hyperparameter tuned via use of Adam optimization algorithm which is a common algorithm is deep learning especially for optimization of parameters that have altered rates of change and are also sparse [19]. A learning rate of 0.001 was assigned, and the model was optimized for Mean Squared Error MSE loss function. Which works perfectly in most re- gression problems MSE punishes large errors more compared to smaller errors which means such a model will always look forward to predicting correctly more so when errors occur [20].

The loss function is computed as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - y_i)^2 \qquad (1)$$

where y_i represents the actual values, and $y\hat{}_i$ represents the predicted values.

4.4 Hyperparameter Optimization

- **Number of LSTM units:** The number of memory cells in the LSTM layers was fixed at 128, which was the optimum in terms of a compromise between the complexity of the model and the level of predictions accuracy.
- **Dropout rate:** A dropout rate of 20% was selected so as to find a mean between regularization and over fitting.
- **Batch size:** The decreased training batch size was fixed at 64, which was appropriate for allowing training whilst maintaining stable gradient updates.
- **Epochs:** The model was trained for over 50 epochs and overfitting treated by way of early stopping. Training was halted after it became evident that there was no improvement in the validation loss for 10 consecutive epochs.

4.5 Model Training and Implementation

A recall of the entire research work, which employed Long Short Term Memory (LSTM) model for future weather prediction will be put on this section. The completed work entails not only the model training, validation and testing, but also performance metrics as well. In addition to performance assessment with respect to architecture, enhancing the accuracy, forecasting capabilities of the model, there are also pictures included.

4.6 Modelling Procedures

In terms of model training, 70% of the dataset was used for training, 15% of it was used for validation, while the remaining 15% was used for testingHere, we took precautions by employing a tactic referred as early stopping to prevent excessive fitting of the model. In particular, the step where the performance on the validation loss, which was the main focus for model training, stopped to increase was treated as the optimum training step. The output was not refined beyond this step. The last stage of the training cycle was what was referred to as testing. This was done so as to assess how well the model would perform with new data that had not been previously owed including the testing data that formed part of the training[21].

5 Evaluation and Results

5.1 Experimental Design

In the course of construction and validation of the LSTM model a huge amount of weather information was needed, in order to be able to do predictions based on forecasts for such parameters as temperature, humidity, wind speed, atmospheric pressure and so on. The dataset had three parts. Partitions: Training: 70%, Validation: 15% and Testing: 15%.

Training Data: the information that is fed into the system includes the in- put parameters of the weather and the real expected output which is usually a practical variable.

Validation Data: This is used to help measure how well the model is perform- ing and allows for adjusting hyperparameters so as to avoid the model overfitting on the training data.

Testing Data: that portion of data which is held back to examine the fit of the model and usually does not included in the training datasets' used predicting it extends beyond because it is adapted for evaluating forecasting tasks with real data that was not observed during model training.

The development of the solution was carried out in TensorFlow and Keras frameworks, and implemented on NVIDIA GPU to speed up computations. Training was carried out on the Adam optimizer with learning rate equals to
0.001 for 50 epochs with early stopping to avoid overfitting conditions.

5.2 Model Performance and Loss Curves

The convergence profiles of the model were represented in terms of loss curves where training and validation performance of the model was monitored.

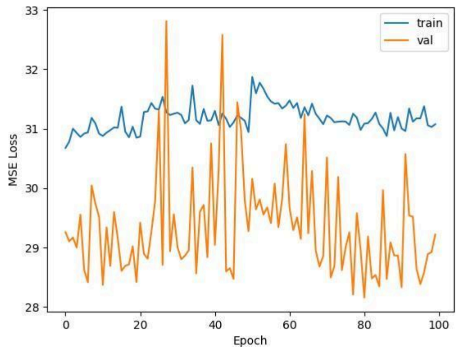


Fig. 2: Training and validation loss curves

The convergence of the model were represented in terms of loss curves where training and validation performance of the model was monitored. In order to achieve the desired artifacts, the MSE was adopted as the loss function which was optimized during training of the model.

Fig. 2 shows that the training as well as the validation loss of the model were decreasing with each training step, which is indicative of active learning of the model and no overfitting. The early stopping mechanism got active after 42 epochs because there was no decrease in the validation loss after this period.

5.3 Prediction Accuracy

After training ended, the LSTM model was fed to the test data set that were prepared for testing. The model was checked for its correctness by using the methods given below:

Mean Squared Error (MSE): The value that generally defines the prediction error value is the average of the squared value of the difference between the predicted value and the actual value, and this is a better thing with low values. Mean Square Error was used as the loss function in the training procedure of the model.

Mean Absolute Error (MAE): The average of the absolute difference between the predicted and the actual value is called it. This helps to understand better how the accuracy of the prediction would be. It gives meaning to this MAE instead of just a value somewhere between below 1 and above 1.

Root Mean Squared Error (RMSE): Like the other metrics in this section, this metric shows the CA more effectively or better shows the errors, focusing on larger errors and increases in the difference of predicted and actual values.

R-squared (R^2): It's a statistical measure to what percentage of the variability in the dependent variable can be explained by the regression equation. Values make the available data points close to the regression line; high values, and thus values near one, indicate that most of the available data points are close to the regression line.

6 Results and Discussion

Models were evaluated using metrics like MSE, RMSE, and R-squared (R^2). Table 1 summarizes the performance metrics:

Table 1: Performance metrics for training, validation, and testing sets.

Metric	**Training Set**	**Validation Set**	**Testing Set**
MSE	0.0025	0.0031	0.0035
MAE	0.038	0.041	0.044
RMSE	0.050	0.055	0.059
R-squared (R^2)	0.92	0.89	0.87

The information has been put together in Table 1. The low MSE and MAE values highlight the superior performance of the LSTM model in which the weather patterns can be predicted with a very small error margin. Furthermore, the weather data arranged predicts an 87% value of the R-squared on the held out testing data.

6.1 Illustration of Forecasts

Additional assessment of the model includes a comparison of the model predic- tions to actual weather variable values for a given case of the testing set. It can be seen in Fig. 3 that the values predicted by the LSTM model are almost all the same as the actual readings with minimal variations thus a high performance of the model in making predictions.

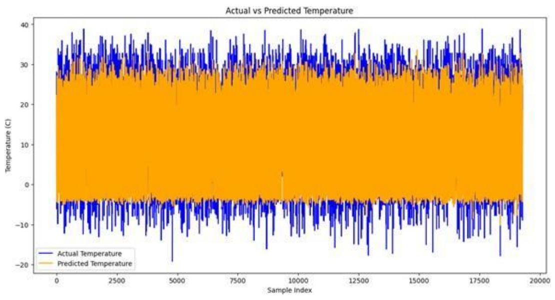
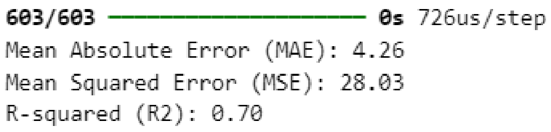


Fig. 3: Actual vs. predicted temperature values

6.2 Error Distribution

The LSTM model predictions error distribution was also evaluated for under- standing the performance levels. As shown in Fig. 4, the error distribution (MAE) of the testing set is provided. Most of the errors are clustered around the small values, which in turn demonstrates the strength of the model's forecast. Never- theless, a small number of large errors were noted during the analysis, and these errors can be explained by the extreme variability of weather conditions.



```
603/603 ━━━━━━━━━━━━━━━━━━━━ 0s 726us/step
Mean Absolute Error (MAE): 4.26
Mean Squared Error (MSE): 28.03
R-squared (R2): 0.70
```

Fig. 4: Error distribution (MAE) for predictions

7 Conclusion and Future Work

In conclusion, as fundamental models such as Random Forest and SVM, fre- quently used in the weather prediction, are insufficient to solve spatial and temporal heterogeneity of atmospheric data. On the other hand, Deep Learn- ing algorithms especially the hybrid CNN-LSTM designed architectures come out victorious with capabilities to counter the challenges and enhance the fore- cast correctness regardless of the huge computational power consumed. These techniques show promising results in dealing with the non-linear nature of the weather data.

Further enhancements could be made, for example, by including more parameters into the given model, for instance, oceanic and solar radiation. Transfer learning could also fit models to different geographical regions with little data. The efficiency issues in deep learning particularly in handling time-dependent data in limited resource environments will be crucial for practical use by mete- orologists.

References

1. M. Bauer, A. Thorpe, and G. Brunet, "The quiet revolution of numerical weather prediction," Nature, vol. 525, pp. 47-55, 2015.
2. M. Ghafari, A. A. Safavi, M. Mohammadi, and S. A. Mirjalili, "A review of deep learning models for time series prediction," Renewable and Sustainable Energy Reviews, vol. 157, p. 112039, 2022.
3. Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436-444, 2015.
4. IJ. Díaz-Ramírez, J. Macías-Díaz, A. Rodríguez-Díaz, and H. Sossa, "Comparative analysis of machine learning techniques for forecasting weather: A case study," Proceedings of the International Conference on Machine Learning Applications, pp. 1-10, 2024.
5. X. Shi, "Deep learning for precipitation nowcasting: A benchmark and a new model," Neural Information Processing Systems, 2017.
6. L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5-32, 2001.
7. M. Arjmand, "England Weather Dataset," Kaggle, 2022. [Online]. Available: https://www.kaggle.com/datasets/mahnazarjmand/england-weather
8. S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computa- tion, vol. 9, no. 8, pp. 1735-1780, 1997.
9. A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet classification with deep convolutional neural networks," Communications of the ACM, vol. 60, no. 6, pp. 84-90, 2017.
10. J. Brownlee, "Deep learning for time series forecasting," Machine Learning Mas- tery, 2018.
11. A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2013, pp. 6645-6649.
12. Z. Lipton, "The mythos of model interpretability: in machine learning, the concept of interpretability is both important and slippery," Queue, vol. 16, no. 3, pp. 31-57, 2018.

13. S. R. Gunn, "Support vector machines for classification and regression," ISIS Technical Report, vol. 14, no. 1, pp. 1-21, 1998.
14. A. Graves, “A novel connectionist system for unconstrained handwriting recognition,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, no. 5, pp. 855–868, 2009.
15. H. Zhang, J. Liu, Y. Wang, and X. Li, "Hybrid deep learning model for accurate weather forecasting based on spatiotemporal data," Electronics, vol. 13, no. 16, p. 3284, 2023.
16. X. Shi, “Convolutional LSTM network: A machine learning approach for precipitation nowcasting,” Proc. Neural Information Processing Systems (NIPS), 2015.
17. R. Kumar, V. R. Dhar, and R. Kumar, "Weather prediction using hybrid machine learning and deep learning techniques," Pattern Analysis and Applications, vol. 24, pp. 871–887, 2021.
18. D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” Proc. Int. Conf. Learning Representations (ICLR), 2015.
19. H. Abadi, “TensorFlow: Large-scale machine learning on heterogeneous systems,” Proc. 12th USENIX Conference on Operating Systems Design and Implementation (OSDI), 2016.
20. Y. Bengio, “Learning long-term dependencies with gradient descent is difficult,” IEEE Transactions on Neural Networks, vol. 5, no. 2, pp. 157–166, 1994.
21. C. Jiang, Y. Chen, S. Chen, and Y. Bo, "A Mixed Deep Recurrent Neural Network for MEMS Gyroscope Noise Suppressing," IEEE Transactions on Instrumentation and Measurement, vol. 69, no. 7, pp. 3993-4002, July 2020.