International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Enhanced Rainfall Prediction with Weighted Linear Units using Advanced Recurrent Neural Network

Dr. Yashwant Dongare[1], Ashvini Shende[2], Amol Dhumane[3], Mubin Tamboli[4], Dr. Satpalsing Devising Rajput[5], Dr. Vinod S. Wadne[6*]

Submitted: 26/06/2023 **Revised**: 06/08/2023 **Accepted**: 27/08/2023

Abstract: A precise rainfall forecast is essential for successful decision-making and catastrophe prevention in the field of meteorology. This article suggests a more effective technique for predicting rainfall that uses a sophisticated recurrent neural network (RNN) and weighted linear units (WLUs). The proposed model seeks to increase the precision and efficacy of rainfall forecasts as compared to existing approaches. The main architecture of the prediction model is an RNN based on Intensified Long Short-Term Memory (Intensified LSTM). The network is trained and assessed using a sizable dataset of rainfall observational data. Indicators of projected rainfall, such as intensity and duration, are generated by the trained model. The performance of the suggested model is assessed using a number of evaluation criteria, such as Root Mean Square Error (RMSE), accuracy, number of epochs, loss, and learning rate. Extreme Learning Machine (ELM), Autoregressive Integrated Moving Average (ARIMA), Holt-Winters, and other innovative RNN and LSTM models are compared to more established approaches. The objective is to show the superior prediction skills of the proposed model. The addition of WLUs enhances the network's capacity to identify intricate linkages and patterns in rainfall data, leading to more precise predictions. The results highlight how the proposed approach may help meteorologists make better decisions and take precautions against disasters.

***Keywords**: Rainfall prediction, Long Short-Term Memory, deep learning, Recurrent Neural Network*

1. Introduction

Rainfall, as a major natural event, has far-reaching consequences for ecosystems, the management of water resources, agriculture, and the daily lives of those who rely on it. It is the principal supply of water for a variety of activities, influencing the lives of countless people. Because atmospheric systems are complex and dynamic, predicting rainfall is a difficult undertaking. Traditional statistical models frequently fail to represent the complex linkages and nonlinear patterns found in rainfall data. RNNs have gained popularity in the field of time series forecasting in recent years due to their ability to efficiently handle sequential data. RNNs are well-suited for rainfall prediction problems because they can capture temporal dependencies by exploiting hidden states that store information from prior time steps.

Heavy rainfall ranging from 3000 mm to 1500 mm per year assures that farmers in certain regions of India can cultivate their crops without fear of water scarcity. In contrast, significant rainfall can cause devastating floods in some area, inflicting extensive property damage and harming the local economy. The healing process for afflicted persons is lengthy, and it takes a long time for them to return to their normal life and resume their jobs. Floods in various parts of India killed roughly 1400 people in 2018, underscoring the human tragedy connected with such catastrophes.

Rainfall forecasting has become a major concern for businesses, governments, risk management organizations, and the scientific community. Because of its random character and the complexities surrounding its occurrence, the intricacy of this meteorological event has drawn attention. Rainfall can vary in timing and intensity even under comparable meteorological conditions, making accurate forecast critical for understanding atmospheric conditions. Predictive analytics, which makes use of previous datasets, has arisen as a novel technique to forecasting future events and making predictions. It is critical in understanding and predicting the climatic conditions that affect many social events such as energy production, building and agriculture in the context of weather prediction.

[1]Assistant Professor, Department of Computer Engineering, Vishwakarma Institute of Information Technology Pune, Maharashtra, India
[2]Adjunct Faculty, Symbiosis school of economics, S. B. Road, Pune, Maharashtra, India
[3]Associate Professor, Department of Computer Science and Engineering, Symbiosis Institute of Technology, Lavale, Pune, Maharashtra, India
[4]Associate Professor, Department of Computer Engineering, Pimpri Chinchwad College of Engineering Nigdi, Pune, Maharashtra, India
[5]Assistant Professor, Department of Computer Engineering, Pimpri Chinchwad College of Engineering Nigdi, Pune, Maharashtra, India
[6]Department of Computer Engineering, JSPM's ICOER wagholi, Pune, Maharashtra, India*
yashwant.dongare@viit.ac.in[1], ashvini.shende@sse.ac.in[2], amol.dhumane@sitpune.edu.in[3], mubin.tamboli@pccoepune.org[4], rajputsatpal@gmail.com[5], vinods1111@gmail.com[6]*
Corresponding Author: Dr. Vinod S. Wadne (vinods1111@gmail.com)

Traditional weather forecasting methods frequently relied on meteorologists' experience and interpretation of weather charts. Years of observation and scientific understanding of weather patterns informed this method. However, as technology advances and large amounts of data become available, there has been a trend toward numerical weather predictions and algorithms using machine learning for weather forecasting. The significance of rainfall prediction stems from its ability to foresee and alleviate the negative effects of catastrophic events. It becomes possible to take preventive measures and efficiently control the hazards connected with landslides, floods, and other weather-related calamities by precisely predicting rainfall patterns. This insight has fuelled the development of mathematical models for weather prediction and the use of machine learning techniques in weather forecasting. Existing prediction approaches frequently face various obstacles, such as limited memory capacity, vanishing gradients in the network, higher prediction errors, and the necessity for accurate rainfall predictions. As the prediction network, this study provides a unique method based on an Intensified LSTM-based RNN.

By employing LSTM networks, the improved LSTM based RNN tackles the problem of limited memory capacity. The ability of LSTMs can store and use information from prior time steps allows for effective modelling of temporal interdependence. The suggested model, by adding LSTM units, can overcome the constraints of standard prediction approaches and identify long-term patterns in rainfall data. To overcome the limitations of existing prediction algorithms, this research offers an Intensified LSTM-based RNN. The model addresses issues such as limited memory capacity, vanishing gradients, increased prediction errors, and the necessity for precise rainfall forecasts. The suggested model intends to increase the efficiency and accuracy of rainfall prediction by adding LSTM units, multiplying the input sequence, and applying the Adam optimizer, thus contributing to the improvement of predictive analysis in this domain.

2. Review of Literature

The Holt-Winters approach was used in [11] to forecast the highest and lowest temperatures time series for the Junagadh region. An Excel spreadsheet was used to make the forecasts. To forecast the temperature data, the methodology used triple exponential smoothing. It should be noted, however, that processing huge datasets in an Excel spreadsheet can be difficult, and this method is better suited for smaller datasets. The additive Holt-Winters approach was used in [12] to analyse rainfall series from river catchment basins. The model was used to forecast rainfall, and both anticipated and actual rainfall data were entered into an evaluation model to assess the forecasting approach's performance.

The ARIMA (Autoregressive Integrated Moving Average) model, especially notably suitable for non-stationary time series applications, is another statistical technique commonly employed for prediction algorithms. Several papers, including [13-15], have examined and explored the ARIMA model in context prediction. In reference [13], the emphasis was on obtaining and predicting time series values related to rain attenuation. In contrast, reference [14] discussed the application of the Box Jenkins series of seasonal ARIMA models for seasonal rainfall prediction. These papers demonstrate how to utilize ARIMA models to model hydrologic data.

The purpose of the optimal model order is a critical difficulty when using ARIMA models. The issue of model identification for hydrological time series data was specifically addressed in reference [15]. However, which limits its capacity to properly capture non-stationary trends. As a result, statistical methods, such as ARIMA, are more appropriate to linear applications and may have drawbacks when dealing with non-linear and non-stationary data. This constraint is a disadvantage when employing statistical approaches since they may not be appropriate for capturing complicated and non-linear relationships in data. Non-linear and non-stationary applications frequently necessitate additional modelling techniques in addition to classic statistical methodologies. As a result, while statistical approaches like ARIMA have their uses in specific circumstances, their utility in non-linear and non-stationary data analysis may be limited.

The free dataset from the China Meteorological Agency (CMA) was used to predict rainfall in [20]. Nave Bayes, Back-propagation Neural Network, and Support Vector Machine were among the approaches compared. This technique, however, is better suited for linear applications, making reliable prediction for non-linear datasets difficult. In [24], a forecast model for proactive cloud auto-scaling was developed by forecasting resource consumption in multivariate time series data using Multivariate Fuzzy-LSTM (MF-LSTM). This LSTM network, however, did not address the vanishing gradient problem. In [25] designed LSTM-based Differential RNN (dRNN) to capture considerable spatiotemporal dynamics. Derivatives of States (DoS) were used in the procedure. This approach, however, was not intended for crowd scene analysis and did not give an end-to-end solution. Several LSTM variants, such as Bidirectional LSTM [26], Multidimensional LSTM (MDLSTM) [27], and Grid LSTM [28], have been proposed to improve on the original LSTM design. While these variations performed better in some areas, they did not directly address the issue of vanishing gradients.

3. Proposed Methodology

Because RNNs contain an element of memory that allows them to maintain large backlogs of datasets, they are particularly well suited for rainfall prediction. Traditional RNNs, on the other hand, suffer from the vanishing gradient problem, which might restrict their accuracy in long-term predictions. The suggested Intensified LSTM model addresses this issue by combining the capabilities of RNNs with Long Short-Term Memory (LSTM). The improved LSTM's increased memory capacity allows for the storage and use of a significant amount of previously collected data. This strategy succeeds in resolving the gradient disparity issue, which results in a more accurate precipitation prediction. The machine learning algorithms, such as neural networks, are best suited for non-linear applications, such as the prediction of precipitation, whereas statistical techniques are best suited for linear applications. The neural networks known as Feed Forward (FFNN) are ineffective at forecasting precipitation because they do not take into account the state of the data prior to the current time. However, recurrent neural networks (RNN) are designed to capture temporal dependencies and store data from earlier temporal steps.

1. Recurrent Neural Networks

Predicting enlargement actions since classifications of variable-length vectors with a temporal component is critical in the setting of machine learning. Recurrent Neural Networks (RNNs) excel in capturing dependencies between prior terms in a series, making them ideal for forecasting weather patterns. A large number of data sets can be used to create prediction models with the help of RNN. A RNN's primary mode of operation consists of making predictions at each stage. A RNN generates a departure or prediction at a predetermined time t using the entry vector at that point in time and the hidden state from the previous step.

$$\mathrm{Kt} = \mathrm{G}(\mathrm{Xh}\,[\mathrm{Kt}-1, \mathrm{Xt}\,] + \mathrm{bh}\,)$$

$$\mathrm{Ot} = \mathrm{f}(\mathrm{Wo} * \mathrm{Kt} + \mathrm{bo})$$

Kt is the hidden state at time t in the RNN model, while Ot is the anticipated emission. Weights Xh and Wo are attributed to abandoned and television couches. Additionally, bh and bo are the respective biases of the hidden capa and the exit capa. While the function of activation g is responsible for turning on the hidden sofa, the function of transmission f is in charge of making predictions.

Back-propagation in a neural network typically calculates the change in weight (w) based on the error (E) associated with the weight (K) assigned in the previous state. The formula for the computation is as follows:

$$\Delta = \frac{\partial \mathrm{E}}{\partial \mathrm{K}}$$

Back-Propagation over Time (BPTT) is a popular training method for Recurrent Neural Networks (RNNs). It entails modifying the network's weights to reduce error by back-propagating through a specific number of states, indicated as "n." The purpose is to compute the weight change by adding the errors of the back-propagated states and updating the weights accordingly.

$$\Delta = \sum_{j=1}^{n} \frac{\partial \mathrm{Ei}}{\partial \mathrm{Ki}}$$

Training classical RNNs with BPTT can be difficult owing to concerns such as exploding and vanishing gradients. When the gradients are back-propagated through time, they either grow exponentially or diminish exponentially. As a result, failures from later time steps have difficulties reaching and affecting older time steps, resulting in inefficient network parameter adjustments.

The Long Short-Term Memory (LSTM) unit was created to address these issues. At each stage of network training, the LSTM unit drastically decreases loss. This is accomplished by incorporating a memory cell that enables the network to retain and update information over longer sequences, essentially overcoming the vanishing gradient problem. The LSTM architecture incorporates specialized gates that control the flow of information (input, forget, and output).

The network can better gather and use relevant information from previous time steps by including LSTM units into the RNN architecture. The ability of the LSTM unit to retain and convey information over longer periods allows for more precise updates of network parameters, which improves the overall training process and improves the model's prediction performance.

2. Long Short Term Memory (LSTM):

The Excel LSTM type recurrent neural networks (RNN) [18]. It is frequently used for tasks like time series analysis, recognition of speech, and natural language processing because it is capable of recording long-term dependencies. The LSTM server is capable of processing the stream of entries. A vector is used to represent each component of a series.

What information from the prior cell state should be erased is decided by the forget gate. Its inputs are the current input and the prior concealed state, and its output falls between [0, 1]. The forget gate's equation is as follows:

$$f_t = \sigma(W_f * [h_{t-1}, x_t] + b_f)$$

Where, h_t-1 is the prior hidden state, x_t is the current input, W_f and b_f are the forget gate's weights and biases, and f_t is the forget gate's output.

What fresh information should be kept in buffer that to be decided by input function gate. It also accepts the current input as well as the prior concealed state as inputs. The input gate's equation is as follows:

$$i_t = \sigma(W_i * [h_{t-1}, x_t] + b_i)$$

Where, W_i and b_i are the input gate's weights and biases, and i_t is the output of the input gate.

Cell State Update: The prior cell state is combined with the new input data to update the cell state. The cell state update equation is as follows:

$$C_t = f_t * C_{t-1} + i_t * \tanh(W_C * [h_{t-1}, x_t] + b_C)$$

where W_C and b_C are the biases and weights for updating the cell state, and C_t represents the updated cell state.

Output Gate: Using the updated cell state, the output gate chooses the LSTM cell's output. It accepts the current input as well as the prior hidden state as inputs. The output gate's equation is:

$$o_t = \sigma(W_o * [h_{t-1}, x_t] + b_o)$$

where W_o and b_o are the output gate's weights and biases, and o_t is the output gate output.

Applying the output gate to the updated cell state results in the computation of the hidden state. The hidden state's equation is as follows:

$$h_t = o_t * \tanh(C_t)$$

The vanishing gradient problem can be reduced and long-term dependencies in sequential data can be captured using LSTM networks.

The following are the activation functions for the various components of an LSTM (Long Short-Term Memory) network:

$Input\ Gate\ (It)$:

$It\ =\ \sigma(Wixt\ +\ Uih(t-1)\ +\ bi)$

$Output\ Gate\ (Ot)$:

$Ot\ =\ \sigma(Woxt\ +\ Uoh(t-1) +\ bo)$

$Forget\ Gate\ (Ft)$:

$Ft\ =\ \sigma(Wftxt\ +\ Ufh(t-1)\ +\ bf)$

$Candidate\ Vector\ (C0t)$:

$C0t\ =\ tanh(Wcxt\ +\ Uch(t-1)\ +\ bc)$

$Cell\ State\ (Ct)$:

$Ct\ =\ Ft\ \odot\ Ct-1\ +\ It\ \odot\ C0t$

$Hidden\ State\ (ht)$:

$ht\ =\ Ot\ \odot\ tanh(Ct)$

LSTMs can retain significant information for extended periods of time by updating and preserving the cell state, which makes them useful for tasks like sequence categorization and machine translation.

3. Improved Long Short Term Memory (ILSTM):

A novel technique termed Sigmoid Weighted Linear Units (SWLUs) was presented to overcome the problems and difficulties connected with the activation functions utilized in the LSTM architecture [31]. SWLUs were developed to address the vanishing gradient problem that occurs during back-propagation in regular LSTM networks. SWLUs work by multiplying input values by a weighting factor before passing them through the sigmoid activation function. This adjustment aids in the regulation and management of gradients during back-propagation, ensuring that they remain within specific levels.

This method resulted in the development of the improved LSTM model. To further solve the vanishing gradient problem, it makes use of SWLUs within the LSTM architecture. The SWLU change the activation functions with increased entry numbers, enabling a more fluid gradient debit and increasing the gradient size during propagation. This improvement helps to lessen the issue of erratic gradients, which can interfere with the effective learning and prediction capabilities of standard LSTM networks. The enhanced LSTM model boosts the network's capacity to collect and transmit gradients, resulting in better training and more accurate predictions. SWLUs are incorporated into the LSTM design in the upgraded LSTM model. This approach enhances overall performance while assisting in the resolution of issues related to the functional activations used in LSTM networks.

The operation of the upgraded LSTM model depends on a number of essential elements:

- The input gate employs an activation function that is sigmoid multiplied by the input. It controls how much of the input is permitted to pass through and hence governs the movement of evidence into the cell state.
- Forget Gate (F): A function of sigmoid activation is also used in the forget gate. It specifies how much information from the preceding time step should be ignored or discarded from the cell state.
- The applicant vector (C') employs a tanh motivation function multiplied by the input. It computes a new candidate value that may be added to the cell state.
- Output Gate (O): The output gate is connected with a softmax activation function. The amount of the cell state that should be output as the hidden state is specified. The relevance of the current cell state to the final output is determined by this gate.
- Hidden state, denoted by the letter "h," is the output of the LSTM unit. It contains information about the

LSTM's present state and is used to produce predictions or to pass to following time steps.

- Memory state (C): The memory's state, denoted by the letter "C," is in charge of storing and keeping long-term information across numerous time steps. It is updated by taking the input barrier, forget gate, and candidacy vector into account, allowing the LSTM to capture and store pertinent information.

A range of values between 0 and infinity is produced by changing the input gate in the Improved LSTM model. With this adjustment, the vanishing gradient issue is successfully avoided, which can happen when the input vector's range of values is constrained. This innovation makes it possible for the network to effectively capture and transmit gradients throughout the network, increasing training and prediction capabilities.

$$\mathrm{LT} = \mathrm{Xa(t)} * \mathrm{sigmoid}(\mathrm{Ki}\,[\mathrm{Y(t)}, (\mathrm{Ct} - 1)\,] + \mathrm{bi})$$

The value obtained from the input gate is multiplied by the current input in the improved LSTM model according to the equation above. The result of this multiplication is the candidate vector, which may be zero or infinity. The model can effectively capture additional tiny changes in the time series data by enabling the candidate vector to take values in this broader range. This improvement allows the learning process to more effectively update the information stored in the cell state vector, resulting in enhanced accuracy and performance in capturing and modelling the data's dynamics.

$$\mathrm{C\,0\,t} = \mathrm{X(t)} * \tanh(\mathrm{Wc}\,[\mathrm{ht} - 1, \mathrm{X(t)}] + \mathrm{bc})$$

The mathematical formulas for the output gate and hidden state in the Intensified LSTM can be expressed as follows:

$Output\ Gate\ (O)$:

$$O = \sigma(Wo * xt + Uo * ht - 1 + bo)$$

In these equations, represents the activation function of the sigmoid, * denotes element-wise multiplication, ⊙ denotes the Hadamard thing, W and U are weight matrices, xt is the input the vector at time step t, ht-1 is the previous time step's hidden state, and bo is the output gate's bias term.

$Hidden\ State\ (ht)$:

$$ht = O \odot tanh(Ct)$$

An optimizer is used in the upgraded LSTM model to boost prediction accuracy even further. The Adam optimizer is specifically utilized in the back-propagation learning process. Based on the training data, this optimizer continuously updates and optimizes the weights, allowing the model to learn and enhance its predictive skills. The proposed improved LSTM model outperforms other prediction models in several performance measures thanks to the use of the Adam optimizer. This shows that the combination of enhanced LSTM, RNN architecture, and weighted linear units is extremely efficient at predicting rainfall.

4. Result and Discussion

Tensorflow's backend and the Python keras library were both used to enhance the LSTM model for rainfall prediction. The number of lost Capas, the rate of learning, the number of points in each Capa, and the rate of loss were all calculated using hyperparameter searching. The optimal model and its associated hyperparameters were discovered using a search strategy that included more than 300 models with various combinations of hyperparameters. The rainfall forecast dataset included historical rainfall data for the Hyderabad region from 1980 to 2019. This dataset was divided into two parts: a training set (34 years from 1980 to 2019) and a testing set (2019). The model's experimental variables included the maximum and minimum temperature, maximum relative humidity, minimum absolute humidity, and wind.

During the formation phase, the system was fed experimental variables and the corresponding precipitation data, allowing it to comprehend and predict the variable of precipitation results. The simulation's results were produced using Jetbrains' Pycharm. Numerous predictive models, including as Holt-Winters, ARIMA, ELM, RNN with ReLU, RNN with SiLU, and LSTM with sigmoidal and hyperbolic activation functions, were examined in relation to the improved LSTM suggested version. Every model had its own performance metrics that took into account the RMSE, losses, precision, and learning rates. The superiority of RNN enhanced prediction models based on LSTM is shown in Table 1.

Table 1: Comparison of Accuracy of different method for rainfall prediction

Method	Accuracy in %
ELM	69.37
Hot-Winter	78.45
ARIMA	81.55
RNN with ReLU	86.11
RNN with Silu	86.32
LSTM	87.54
Improved LSTM	89.36

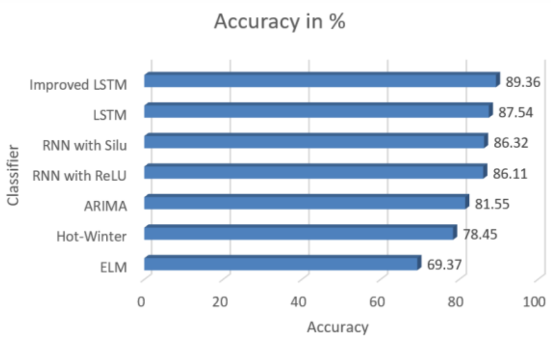

Fig 1: Comparison of Accuracy of different method for rainfall prediction

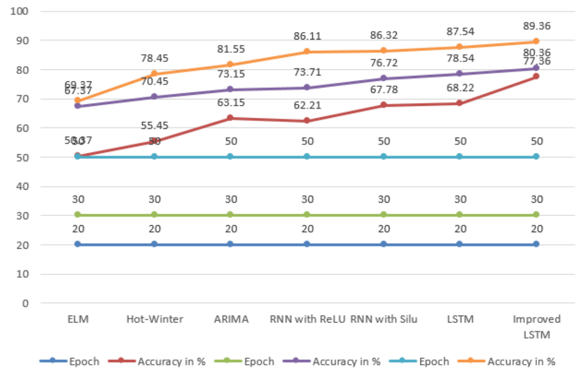

Fig 2: Comparison of prediction model with Neural Network

Using the rainfall dataset, the accuracy of several prediction algorithms was assessed. The ELM (Extreme Learning Machine) obtained an accuracy of 69.37%, according to the results. With an accuracy of 78.45%, the Holt-Winters technique did somewhat better. The ARIMA (Autoregressive Integrated Moving Average) model improved even further, reaching 81.55% accuracy. Moving on to the RNN (Recurrent Neural Network) models, the one with ReLU activation obtained 86.11% accuracy, while the one with Silu activation achieved 86.32%. The LSTM (Long Short-Term Memory) model outperformed the other techniques, with an accuracy of 87.54%. The proposed Improved LSTM model, on the other hand, had the maximum accuracy of 89.36%. These findings suggest that the Improved LSTM outperforms the other models in predicting rainfall.

Table 2: Summary of different method accuracy comparison

Method	Number of Epochs	Accuracy (%)	RMSE	Losses	Learning Rate (ms)
ELM	-	69.37	3.08	-	-
Hot-Winter	-	78.45	2.84	-	-
ARIMA	-	81.55	5.68	-	-
RNN with ReLU	50	86.11	0.76	0.5824	0.9
RNN with Silu	50	86.32	0.76	0.5769	0.75
LSTM	50	87.54	0.35	0.1274	0.5
Improved LSTM	50	89.36	0.33	0.0054	0.025

Several criteria were used to assess the efficacy of various prediction algorithms, including accuracy, root mean square error (RMSE), losses, and learning rate. Without defining the number of epochs, the ELM (Extreme Learning Machine) obtained an accuracy of 69.37%. The Holt-Winters approach improved accuracy to 78.45%, but it did not provide information on the number of epochs or losses. The ARIMA (Autoregressive Integrated Moving Average) model fared even better, with an accuracy of 81.55%, but did not offer epoch or loss information.

Moving on, both the RNN with activation of ReLU and the RNN with Silu activation had their RNN (Recurrent Neural Network) model trained for 50 epochs. The RNN with ReLU obtained an accuracy of 86.11%, an RMSE of 0.76, and losses of 0.5824 at a learning rate of 0.9 (ms). The RNN with Silu performed somewhat better with a learning rate of 0.75 (ms), achieving accuracy of 86.32%, RMSE of 0.76, and losses of 0.5769. Similarly trained across 50 epochs, the LSTM (Long Short-Term Memory) model outperformed the others with an accuracy of 87.54%. It had losses of 0.1274 and a lower RMSE of 0.35 with a learning rate of 0.5 (ms). The proposed Improved LSTM outperformed all previous methods.

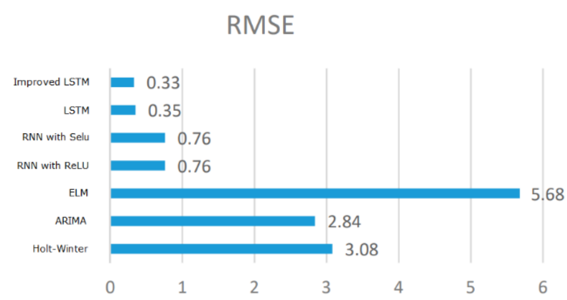

Fig 3: Different RMSE prediction model comparison

In terms of accuracy, RMSE, and losses, these data show that the Improved LSTM model beat the other techniques. The number of training epochs was consistent among the RNN models, although the learning rates differed. Overall, the Improved LSTM model had the highest accuracy and lowest mistakes, suggesting its efficiency and efficacy in rainfall prediction.

When the RMSE of the different methods is compared, the statistical methods (Holt-Winters and ARIMA) compete with the LSTM-based model. The Holt-Winters approach controls the data pattern with smoothing parameters, whereas ARIMA converts non-stationary data to stationary data before making predictions. The ELM approach, on the other hand, has a high RMSE due to the lack of weight adjusting. RNN models with ReLU and Silu activation functions perform equally, however the introduction of LSTM units in the RNN dramatically reduces error when compared to RNN without LSTM. Figure 3 depicts the RMSE comparison of all approaches utilized in this research.

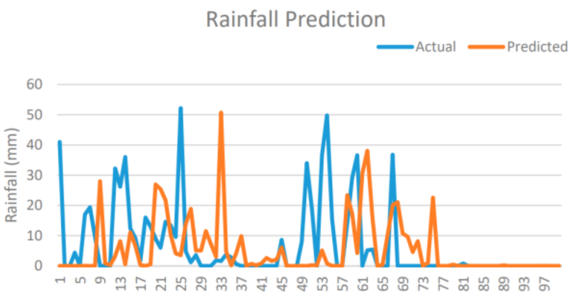

Fig 4: Prediction of rainfall as Actual and Predicted

Figure 4 depicts the actual and forecast rainfall levels plotted during a 97-day period from July to September 2014. The model was trained utilizing 34 years of past data as input, which was made possible by the RNN network's utilization of LSTM units. Except for a few peaks, the plot of projected rainfall values roughly matches the actual values. This variance could be caused by outliers in the dataset, which can cause considerable differences in successive readings. The prediction approaches seek to either eliminate or include these outliers by examining whether comparable patterns appear throughout the training phase. When the RMSE and real versus anticipated rainfall values are compared, the LSTM-based model performs well, closely reflecting the actual rainfall pattern. The inclusion of LSTM units in RNN enables for the processing of huge datasets as well as better capturing of temporal dependencies, resulting in better predictions.

5. Conclusion

In this paper we suggested an improved rainfall prediction model in this study by adding weighted linear units in an advanced recurrent neural network (RNN). The Intensified LSTM model solves the shortcomings of standard prediction approaches such as statistical algorithms and basic RNNs. The Intensified LSTM model outperforms other rainfall prediction models by exploiting the capabilities of LSTM units and the improvements brought by weighted linear units. We assessed the accuracy and efficiency of the Intensified LSTM model using extensive testing and comparative analysis. The outcomes demonstrated its superiority over other prediction approaches such as ELM, Holt-Winters, ARIMA, and simple RNNs. The Intensified LSTM outperformed all other models in this investigation with an excellent accuracy of 89.36%. We also discovered that including weighted linear units aids in overcoming the obstacles of vanishing gradient and improves the learning process. The Intensified LSTM effectively mitigates the vanishing gradient problem by adding the sigmoid and tanh functions multiplied by the input in the input gate and candidate vector, respectively, allowing the model to capture and utilize more nuanced patterns and relationships in the data. The suggested improved LSTM model, powered by weighted linear units, provides an enhanced method for predicting rainfall. Its capacity to handle long-term dependencies, solve the vanishing gradient problem, and give greater accuracy demonstrates its promise for use in a variety of disciplines, including weather forecasting, agriculture, and disaster management. The improved LSTM model represents a considerable leap in rainfall prediction and shows great promise for future research and implementation.

References

[1] Jimoh, R.G.; Olagunju, M.; Folorunso, I.O.; Asiribo, M.A. Modeling rainfall prediction using fuzzy logic. Int. J. Innov. Res. Comput. Commun. Eng. 2013, 1, 929–936.

[2] Wu, J.; Liu, H.; Wei, G.; Song, T.; Zhang, C.; Zhou, H. Flash flood forecasting using support vector regression model in a small mountainous catchment. Water 2019, 11, 1327.

[3] Poornima, S.; Pushpalatha, M.; Sujit Shankar, J. Analysis of weather data using forecasting algorithms. In Computational Intelligence: Theories, Applications and Future Directions—Volume I. Advances in Intelligent Systems and Computing; Verma, N., Ghosh, A., Eds.; Springer: Singapore, 2019; Volume 798.

[4] Manideep, K.; Sekar, K.R. Rainfall prediction using different methods of Holt winters algorithm: A big data approach. Int. J. Pure Appl. Math. 2018, 119, 379–386.

[5] Gundalia, M.J.; Dholakia, M.B. Prediction of maximum/minimum temperatures using Holt winters method with excel spread sheet for Junagadh region. Int. J. Eng. Res. Technol. 2012, 1, 1–8.

[6] Puah, Y.J.; Huang, Y.F.; Chua, K.C.; Lee, T.S. River catchment rainfall series analysis using additive Holt–Winters method. J. Earth Syst. Sci. 2016, 125, 269–283.

[7] Patel, D.P.; Patel, M.M.; Patel, D.R. Implementation of ARIMA model to predict Rain Attenuation for KU-band 12 Ghz Frequency. IOSR J. Electron. Commun. Eng. (IOSR-JECE) 2014, 9, 83–87.

[8] Graham, A.; Mishra, E.P. Time series analysis model to forecast rainfall for Allahabad region. J. Pharmacogn. Phytochem. 2017, 6, 1418–1421. 15. Salas, J.D.; Obeysekera, J.T.B. ARMA model identification of hydrologic time series. Water Resour. Res. 1982, 18, 1011–1021.

[9] Chen, W.; Xie, X.; Wang, J.; Pradhan, B.; Hong, H.; Bui, D.T.; Duan, Z.; Ma, J. A comparative study of logistic model tree, random forest, and classification and regression tree models for spatial prediction of landslide susceptibility. Catena 2017, 151, 147–160.

[10] Kusiak, A.; Wei, X.; Verma, A.P.; Roz, E. Modeling and prediction of rainfall using radar reflectivity data: A data-mining approach. IEEE Trans. Geosci. Remote Sens. 2013, 51, 2337–2342.

[11] Kannan, M.; Prabhakaran, S.; Ramachandran, P. Rainfall forecasting using data mining technique. Int. J. Eng. Technol. 2010, 2, 397–401.

[12] Mehrotra, R.; Sharma, A. A nonparametric nonhomogeneous hidden Markov model for downscaling of multisite daily rainfall occurrences. J. Geophys. Res. Atmos. 2005, 110.

[13] Nayak, D.R.; Mahapatra, A.; Mishra, P. A survey on rainfall prediction using artificial neural network. Int. J. Comput. Appl. 2013, 72, 16.

[14] Kashiwao, T.; Nakayama, K.; Ando, S.; Ikeda, K.; Lee, M.; Bahadori, A. A neural network-based local rainfall prediction system using meteorological data on the internet: A case study using data from the Japan meteorological agency. Appl. Soft Comput. 2017, 56, 317–330.

[15] Mislan, H.; Hardwinarto, S.; Sumaryono, M.A. Rainfall monthly prediction based on artificial neural network: A case study in Tenggarong Station, East Kalimantan, Indonesia. Procedia Comput. Sci. 2015, 59, 142–151.

[16] Niu, J.; Zhang, W. Comparative analysis of statistical models in rainfall prediction. In Proceedings of the 2015 IEEE International Conference on Information and Automation, Lijiang, China, 8–10 August 2015; pp. 2187–2190.

[17] Dash, Y.; Mishra, S.K.; Panigrahi, B.K. Rainfall prediction of a maritime state (Kerala), India using SLFN and ELM techniques. In Proceedings of the 2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT), Kannur, India, 6–7 July 2017; pp. 1714–1718.

[18] Poornima, S.; Pushpalatha, M.; Aglawe, U. Predictive analytics using extreme learning machine. J. Adv. Res. Dyn. Control Syst. 2018, 10, 1959–1966.

[19] Choi, E.; Schuetz, A.; Stewart, W.F.; Sun, J. Using recurrent neural network models for early detection of heart failure onset. J. Am. Med Inform. Assoc. 2016, 24, 361–370.

[20] Tran, N.; Nguyen, T.; Nguyen, B.M.; Nguyen, G. A multivariate fuzzy time series resource forecast model for clouds using LSTM and data correlation analysis. Procedia Comput. Sci. 2018, 126, 636–645.

[21] Zhuang, N.; Kieu, T.D.; Qi, G.J.; Hua, K.A. Deep differential recurrent neural networks. arXiv 2018, arXiv:1804.04192. 26. Schuster, M.; Paliwal, K.K. Bidirectional recurrent neural networks. IEEE Trans. Signal Process. 1997, 45, 2673–2681.

[22] Graves, A.; Mohamed, A.R.; Hinton, G. Speech recognition with deep recurrent neural networks. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013; pp. 6645–6649.

[23] Kalchbrenner, N.; Danihelka, I.; Graves, A. Grid long short-term memory. arXiv 2015, arXiv:1507.01526. 29.

[24] Salman, A.G.; Heryadi, Y.; Abdurahman, E.; Suparta, W. Single layer & multi-layer long short-term memory (LSTM) model with intermediate variables for weather forecasting. Procedia Comput. Sci. 2018, 135, 89–98. 30.

[25] Hochreiter, S.; Schmidhuber, J. Long short-term memory. Neural Comput. 1997, 9, 1735–1780.

[26] Elfwing, S.; Uchibe, E.; Doya, K. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. Neural Netw. 2018, 107, 3–11.

[27] Davuluri, R. ., Mallapragada, V. ., Mamillapalli, U. M. R. ., M, M. ., & Peeka, S. . (2023). Alzheimer's Disease Diagnosis Using CNN Based Pre-trained Models. International Journal on Recent and Innovation Trends in Computing and Communication, 11(4), 315–323. https://doi.org/10.17762/ijritcc.v11i4.6456

[28] Martin, S., Wood, T., Hernandez, M., González, F., & Rodríguez, D. Machine Learning for Personalized Advertising and Recommendation. Kuwait Journal of Machine Learning, 1(4). Retrieved from http://kuwaitjournals.com/index.php/kjml/article/view/156

[29] Dhabliya, D., Dhabliya, R. Key characteristics and components of cloud computing (2019) International Journal of Control and Automation, 12 (6 Special Issue), pp. 12-18.