Received Date:7th February, 2025
Revision Date: 15th February, 2025
Accepted Date:28th February, 2025

Rainfall Prediction using Wavelet Transform and Transformers

Bishnu Bashyal[1*], Darshan Bhusal[2], Kishor Singh3, Roshan Koju[4]
[1]Nepal College of Information Technology, Pokhara University, E-mail: bishan041@gmail.com
[2] Nepal College of Information Technology, Pokhara University, E-mail: darshanbhusal@gmail.com
[3]Nepal College of Information Technology, Pokhara University, E-mail: mekishorsingh2015@gmail.com
[4]Associate Professor, Nepal College of Information Technology, Pokhara University.
E-mail: contactroshankoju@gmail.com

Abstract — Rainfall prediction is crucial for various applications, yet time series data in meteorological datasets poses a significant challenge. This research proposes a novel approach that combines the power of wavelet transform and transformer architecture to develop an accurate rainfall prediction model. By decomposing time series data and leveraging self-attention mechanisms, this model captures complex temporal dependencies and spatial patterns. Through extensive evaluation and comparison using different metrics, the proposed algorithm demonstrates the superiority over existing methods in terms of predictive accuracy. The proposed model has been compared with LSTM model to evaluate its effectiveness in rainfall prediction and has measured a loss of 0.060, mean absolute error of 0.05, mean absolute percentage error of 0.05 and root mean square error of 0.10. The proposed model empowers decision-makers with reliable rainfall predictions, aiding improved planning and preparedness.

Keywords — Rainfall prediction, wavelet transform, transformers architecture, time series forecasting, meteorological data.

Introduction

Precipitation forecasting is an important aspect of meteorology, which significantly impacts areas such as agriculture, water resources management and disaster preparedness. [1]. Accurate and timely rainfall predictions are essential for informed decision-making, resource allocation, and risk mitigation. However, the inherent complexity of meteorological data, presents significant challenges in developing reliable rainfall prediction models. Because there is a complicated non-linear relationship in the process of turning precipitation into runoff, the relationship between rainfall and runoff is among the most complex hydrological phenomena. Due to the numerous variables involved in the physical process trial, this action is exceedingly tough to assimilate.[2]. This research endeavors to address the persistent issue in rainfall prediction by proposing an innovative approach that combines the power of wavelet transform and transformer architecture. Meteorological data, typically collected over time, often contains gaps due to various reasons, such as sensor malfunctions, network failures, or irregular data collection schedules [3]. The accurate forecasting of rainfall patterns is

essential for informed decision-making, resource management, and disaster preparedness [4]. Understanding when and where precipitation will occur can greatly benefit agriculture, water resource allocation, flood control, and infrastructure planning. However, this endeavor is not without its challenges, chief among them being the presence of time series data over long time.

Traditional approaches to predict rainfall, such as data imputation methods, have limitations when applied to meteorological datasets [5]. Moreover, prediction rainfall and other weather conditions using simple methods like mean or linear interpolation can lead to inaccurate predictions, as they do not capture the complex relationships inherent in meteorological data. Nonlinear patterns, seasonality, and the impact of different climatic conditions on rainfall events are some of these interactions. The accumulation of daily precipitation for the following day can be predicted using architecture based on deep learning. More precisely, it employs a layered perceptron for the task of the prediction and an autoencoder for minimizing and capturing non-linear connections between characteristics [6].

Transformer architecture has been adopted in a number of sectors as a result of recent advances in deep learning, including time series forecasting. Transformers, originally designed for natural language processing tasks, excel at capturing complex temporal dependencies and spatial patterns [7]. Their self-attention mechanism allows them to

** Corresponding Author*

weigh the importance of different data points dynamically, making them well-suited for handling irregularly spaced time series data.

Predicting rainfall accurately requires not just rainfall measurement but also the use of multiple meteorological characteristics such as humidity, wind speed, temperature, and air pressure [8]. Addressing the critical need for accurate rainfall prediction, where traditional methods may struggle due to reliance on single attributes and poor handling of temporal dependencies. Utilizing a combination of wavelet transforms and Transformer with GRU layers can effectively incorporate multiple meteorological parameters, capturing complex temporal and spatial relationships. This methodology enhances model's capability to forecast accurately, leading to more realistic predictions and better outcomes.

The objective of this study project is to:

- Develop a rainfall prediction model using wavelet transform and transformer architecture.

Literature Review

Precipitation prediction is important for climate research because it has a deep impact on various fields such as water resource management, agriculture, and disaster prevention. Over the years, various methodologies employing Machine Learning (ML), Deep Learning (DL), and statistical algorithms have been explored to enhance rainfall forecasting accuracy and reliability. This literature review synthesizes recent studies focusing on diverse techniques and datasets for rainfall prediction.

2.1 Comparative Study of Machine Learning Models

Ridwan et al. [2] conducted a comparative study to develop and evaluate ML models for rainfall prediction in Terengganu, Malaysia. They utilized data from ten stations within the study area and applied Bayesian Linear Regression (BLR), Decision Forest Regression (DFR), Boosted Decision Tree Regression (BDTR), and Neural Network Regression (NNR) algorithms. This study emphasizes the importance of considering different scenarios and time horizons for accurate rainfall forecasting.

2.2 Deep Learning Architecture for Precipitation Prediction

Hernandez et al. [9] proposed a DL-based architecture for predicting daily precipitation accumulation. Their model integrates an autoencoder to capture nonlinear relationships between features and a multilayer perceptron for prediction. The results show better performance in terms of mean square error (MSE) and root mean square error (RMSE) compared to other approaches.

2.3 3D Convolutional Neural Network for Spatial Rainfall Prediction

A 3D convolutional neural network (CNN) developed by [10] demonstrates the capability to predict precipitation spatial distribution using meteorology field data. The contiguous United States' daily precipitation and meteorological data spanning 39 years were used in this study. The model's architecture includes max-pooling ,3D convolution layers, subsampling, and fully connected layers, showcasing promising results for spatial rainfall forecasting.

2.4 Long Short-Term Memory (LSTM) Techniques in Rainfall Prediction

Imrus Salehin et al. [8] proposed an LSTM-based model for rainfall prediction, leveraging six parameters collected from six regions. Their approach achieved 76% accuracy, highlighting the effectiveness of LSTM in sequence data analysis. Similarly, Dhital et al. [11] utilized LSTM for forecasting precipitation and air temperature in Nepal, achieving high accuracy and robustness. They employed averaging and linear interpolation techniques for data pre-processing, optimizing the model using Stochastic Gradient Descent and Adam optimizer.

2.5 Application of LSTM in Precipitation Forecasting

In [12], LSTM models were deployed for precipitation prediction based on meteorological data from Jingdezhen City. By selecting significant input variables and refining them based on relative importance, the LSTM model demonstrated improved prediction accuracy compared to classical statistical and ML algorithms.

By utilizing various machine learning and deep learning algorithms such as LSTM, RNN, CNN and auto encoders, researchers have successfully captured temporal and spatial dependencies in rainfall data. Research on rainfall prediction has increased as a result of the integration of multiple data sources, such as satellite imaging and data from weather stations. Methods like stochastic gradient descent and the Adam optimizer have made it possible to optimize model parameters and reduce prediction errors. All things considered, these methodological advancements show great potential for creating more precise and trustworthy rainfall forecasting systems. This will have a big effect on industries like water resource management, agriculture, and disaster preparedness that rely on rainfall forecasts.

Research Methodology

3.1 Data Collection

A good result is possible if only the data is collected from reliable sources, such as weather stations, authentic portals or remote sensing platforms. It is important to ensure that the dataset includes relevant meteorological parameters like humidity, wind speed, temperature, and atmospheric pressure along with rainfall. In this research, the data is collected from Opendatanepal and contains data from 1981-01-01 to 2019-12-31. Data is extracted for 93 weather stations spanning 62 districts in Nepal and contains 18 attributes.

3.2 Pre-processing

The pre-processing of data involves removal of irrelevant columns, checking the presence of null and negative values, and normalizing data using MinMaxscalar. Given the focus of this study, emphasize techniques that maintain data integrity and minimize information loss.

Wavelet Decomposition: It includes applying the wavelet transform to the pre-processed data to decompose it into different frequency components at multiple scales. Here, Discrete Wavelet Transform (DWT) is implemented using the apply_wavelet_transform function, decomposes each time series into approximation coefficients using the discrete wavelet transform (DWT) with the Daubechies 1 (db1) wavelet. The DWT splits a time series into approximation (or smoothing) and detail (or high-frequency) factors. Here, we retain only the approximation coefficients, which capture the coarse-scale information of the original time series while discarding the fine-scale details.

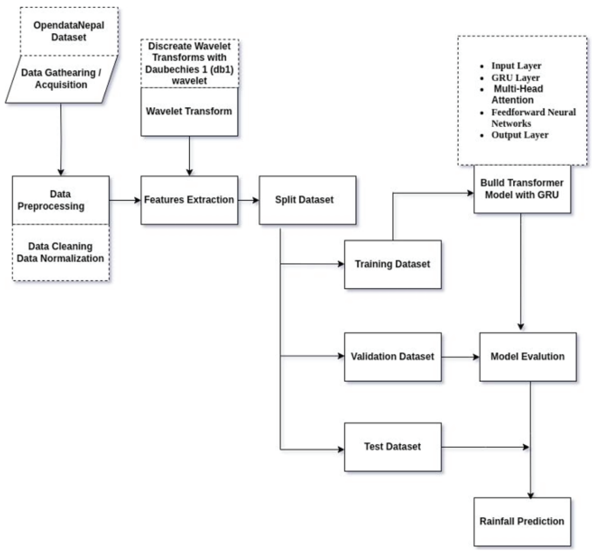

Figure 1: Proposed Methodology

This research proposes a model that includes the integration of wavelet transform and a transformer, WT-Transformer model for time-series forecasting, for rainfall prediction. The developed transformer model forecasts the features produced by wavelet transform (WT). The encoder compresses the meaningful information in the input sequence into a fixed-length vector in the transformer, and the decoder transforms it into an output. Each layer in encoder contains. two sublayers: a fully connected neural network and a multi-head self-attention layer. Each sub-layer uses normalization and the residual connection to boost performance. The encoder and the decoder are comparable. The only difference is the addition of two multi-head self-attention layers. In contrast to the original decoder, we do not employ the mask attention method in this case because the decoder only accepts historical data without any knowledge about the future. In fact, none of the efforts have so far emphasized on developing a uniquely built transformer-based deep neural network model linked with wavelet decomposition for rainfall prediction. The combination of wavelet transforms and transformer aims to capture both the time-frequency characteristics of the rainfall data. The wavelet transform helps in decomposing the signal into different frequency components, while the transformer effectively models the temporal dependencies and relationships within each frequency band. The best feature of this model is its ability to handle long-range dependencies and nonlinear behaviour (with the aid of transformers) to represent the time series' non-stationarity (with the aid of wavelet decomposition). This hybrid approach aims to improve the model's ability to handle the complexity and dynamics of rainfall time series data.

3.4 Split data and Training

Split the preprocessed dataset into training, validation, and test sets. Ensure that the temporal order of the data is preserved to mimic real-time forecasting scenarios. The split data ratio is 55:21:24 with respect to training, validation and testing allowing large test set for excellence model performance.

Analyze the performance of the trained model on the test dataset. Evaluate the performance of the trained model root mean square error (RMSE) and mean absolute error (MAE), among other appropriate metrics for rainfall prediction.

3.5 Wavelet Transform

Wavelet transform is a versatile mathematical technique with many applications in artificial intelligence (AI), primarily for signal and data analysis. It decomposes complex signals or data into basic functions called wavelets, representing signal frequency components at different scales for efficient representation. In AI, wavelet transform is crucial for feature extraction, particularly in image processing, where wavelet coefficients capture textures and edges for tasks like classification and object detection. Additionally, it aids in denoising noisy data while preserving essential features, enhancing reliability for subsequent AI processing.

For time series data prevalent in fields like finance and forecasting, wavelet transform decomposes data into frequency components, facilitating trend, seasonal pattern, and anomaly identification for predictive modelling and decision-making. Wavelet-based image compression methods, such as JPEG, efficiently reduce image file sizes while maintaining quality, vital for image-related tasks in AI like storage, transmission, and analysis. In healthcare, wavelet transform assists in processing biomedical signals like ECG and EEG, aiding anomaly detection, diagnostic feature extraction, and disease classification improvement.

3.5.1 Discrete Wavelet Transforms with Daubechies 1 (db1) wavelet

In order to analyze the data using a discrete set of wavelets, we have implemented Discrete Wavelet Transform (DWT) using the apply_wavelet_transform function, decomposes each time series into approximation coefficients using the discrete wavelet transform (DWT) with the Daubechies 1 (db1) wavelet. The DWT decomposes a time series into approximation (or smoothing) coefficients and detail (or high-frequency) coefficients. Here, we retain only the approximation coefficients, which capture the coarse-scale information of the original time series while discarding the fine-scale details. This can be useful for reducing noise or extracting essential features from the time series data. Finally, the transformed approximation coefficients are stored in numpy arrays X_train1, X_val1, and X_test1, ready for further processing or model training.

3.6 Transformer Architecture

The architecture of the transformer proposed for this work is as shown in the figure below. This architecture combines elements of both GRU (Gated Recurrent Unit) and Transformer models to create a deep GRU-Transformer hybrid neural network. The input layer receives sequences of data with a shape of (10, 9), indicating sequences of length 10 with 9 features each. The data flows through two stacked GRU layers, each followed by Layer Normalization, to capture temporal dependencies within the sequences. Subsequently, the processed sequences are passed through two stacked Multi-Head Attention layers, allowing the model to attend to relevant information across different positions in the sequences. Each Multi-Head Attention layer is also followed by Layer Normalization to stabilize training. The output of the attention layers is then fed into a feedforward neural network (FFNN) with a hidden dimension of 16 and ReLU activation, followed by a linear output layer. A fully connected layer with ReLU activation and dropout regularization follows, and global average pooling is used to aggregate data across the sequence dimension. The Adam optimizer is used to optimize the model, and the mean absolute error (MAE) metric and mean squared error (MSE) loss are used to assess the model. This architecture is designed to effectively capture temporal dependencies and long-range interactions within sequential data while also leveraging self-attention mechanisms for contextual learning.

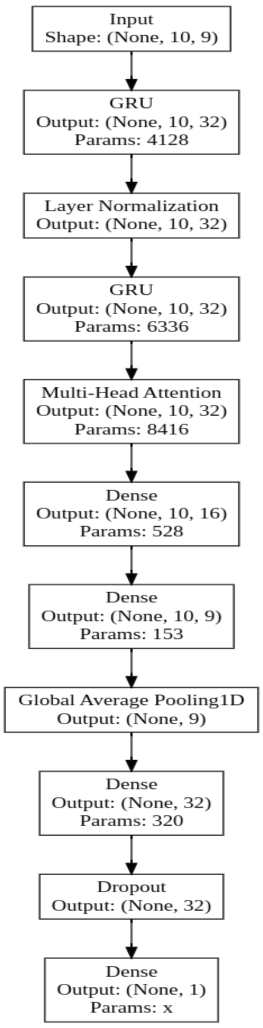

Figure 2: Transformer Architecture

Result and Discussion

4.1 Data Description

The rainfall data used in the research is csv file which consist the information of different attributes likely to impact the cumulative rainfall. It consists of 21 columns including the label column and other 18 columns with different attributes like radiation, wind, latitude, longitude, latitude, evaporation etc. while there are altogether 29006 rows consisting of rainfall on different occasions.

4.2 Model Implementation

The developed model uses a wavelet transform and transformers for making ultimate rainfall prediction.

4.2.1 Wavelet Transform:

This is a module implemented for feature extraction and therein reduce the decomposition helps to identify and extract relevant information from the signal while potentially discarding less important or noisy components. Features that have little correlation with the target variable or do not provide meaningful information for the predictive task can be discarded. The data frame is there by reduced to 9 columns.

This research applies the wavelet transform to each series of data in the training, validation, and test sets (X_train1, X_val1, and X_test1, respectively). The wavelet transforms, implemented using the apply_wavelet_transform function, decomposes each time series into approximation coefficients using the discrete wavelet transform (DWT) with the Daubechies 1 (db1) wavelet. The DWT decomposes a time series into approximation (or smoothing) coefficients and detail (or high-frequency) coefficients. Here, we retain only the approximation coefficients, which capture the coarse-scale information of the original time series while discarding the fine-scale details. This can be useful for reducing noise or extracting essential features from the time series data. Finally, the transformed approximation coefficients are stored in numpy arrays X_train1, X_val1, and X_test1, ready for further processing or model training.

4.2.2 Transformer:

This module is implemented to capture temporal dependencies and then training the model to learn patterns within the data. It allows the model to weigh the importance of different elements in the sequence when making predictions. The building block and parameters involved in the model are as follows:

Table 1

Building block and parameters involved

Layer Name	Output Shape
input_layer	(None, 10, 9)
gru	(None, 10, 32)
layer_normalization	(None, 10, 32)
gru_1	(None, 10, 32)
layer_normalization_1	(None, 10, 32)
multi_head_attention	(None, 10, 32)
layer_normalization_2	(None, 10, 32)
multi_head_attention_1	(None, 10, 32)
layer_normalization_3	(None, 10, 32)
dense	(None, 10, 16)
dense_1	(None, 10, 9)
layer_normalization_4	(None, 10, 9)
global_average_pooling1d	(None, 9)
dense_2	(None, 32)
dropout	(None, 32)
dense_3	(None, 1)

The "gru_transformer" model architecture is intended to record sequential data's temporal dependencies like time series while minimizing the computational complexity compared to traditional transformer architectures. The input layer receives sequences of data with a shape of (None, 10, 9), indicating a variable batch size, sequences of length 10, and 9 features per time step. Following the input layer, a GRU (Gated Recurrent Unit) layer with 32 units processes the input sequences. GRU is a form of recurrent neural network (RNN) which has the capacity to efficiently capture temporal dependencies while mitigating the vanishing gradient problem commonly encountered in standard RNNs.

After the GRU layer, normalization is applied to adjust the activation of each layer and ensure a more stable learning process. The model then employs a multi-head attention system that allows it to focus on multiple parts of the input stream concurrently, enhancing its ability to capture relevant information across various temporal scales. Subsequent dense layers further process the representations obtained from the attention mechanism before producing final predictions. The model ends with a dense output layer followed by layer normalization and global average pooling to produce predictions for each time step in the sequence. Additionally, a dropout layer is included before the final dense layer to prevent overfitting and improve generalization performance.

Overall, this architecture combines the strengths of GRU-based recurrent processing with attention mechanisms to accurately model long-term dependencies and make predictions on sequential data like rainfall prediction.

Training Process: For a specified number of 30 epochs, the model is fitted to the training data (X_train1, y_train1) as part of the training process. Additionally, the validation data (X_val1, y_val1) is used to evaluate the model's performance on data not used for training, helping to assess generalization. The ModelCheckpoint callback (cp1) is included in the training process, ensuring that the model's weights are saved in the specified directory ('model1/') after each epoch if the performance on the validation set improves. The goal is to minimize the absolute gap between the predicted and actual values. Mean absolute error (MAE) is a loss function and it is also employed as the metric to monitor during the training period.

The optimizer selected for the training process is Adam, a widely used optimization algorithm, having a learning rate of 0.00001. This rate determines the step throughout optimization, and a smaller learning rate is often chosen for fine-tuning to ensure convergence. The training is performed in batches of size 4 (batch_size=4), which means that the model parameters are updated after processing four samples. The primary aim of this training process is to track the model's performance on the validation set, use the Model Checkpoint callback to save the best weights, and continuously adjust the model's weights to reduce the mean absolute error on the training set.

4.3 Model Performance Analysis

The model, trained on a dataset of 29,006 rows divided into training, validation, and testing sets with 16,000, 6,000, and 7,006 rows respectively, demonstrated a steady performance across various metrics over 30 epochs. The loss function reached a minimum of 0.056% for training and 0.05% for validation, while the Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE) both achieved minima of approximately 0.05%. Additionally, the Root Mean Square Error (RMSE) was achieved at about 0.10 units indicating the low error magnitude of the Transformers model's predictions. These metrics indicate that the model achieved stable and accurate performance throughout the training phase.

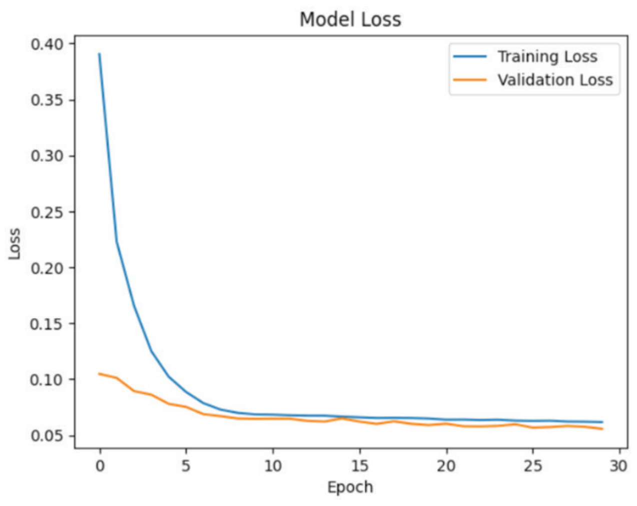

Figure 3: Loss Vs Epoch Plot

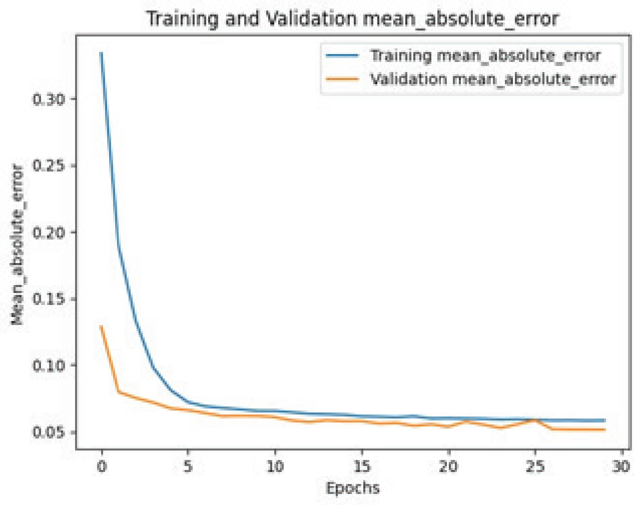

Figure 4: MAE Vs Epochs

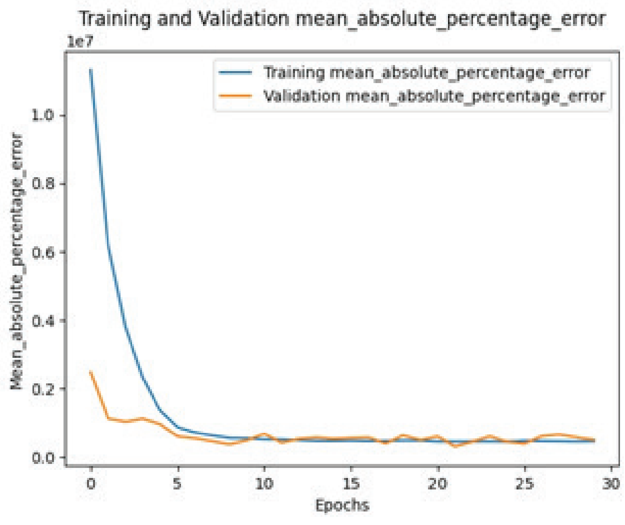

Figure 5: MAPE vs. Epoch

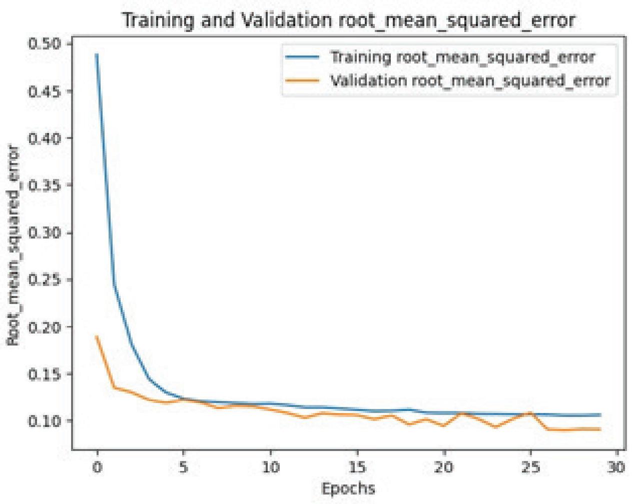

Figure 6: Root Mean Square Error

4.4 Actual vs Predictions

This portion consists of prediction of rainfall using transformer under different cases of datasets

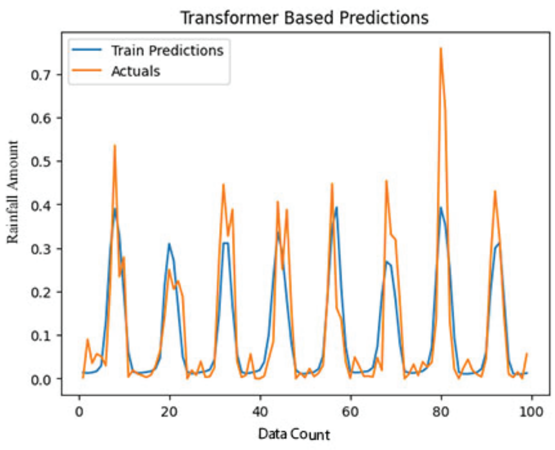

Figure 7: Actual vs Train Predictions

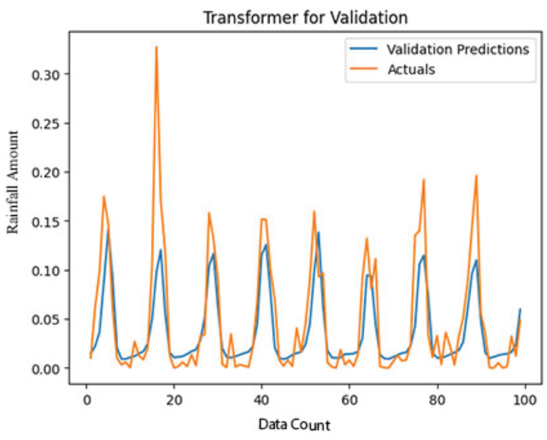

Figure 8: Actual vs Train Predictions

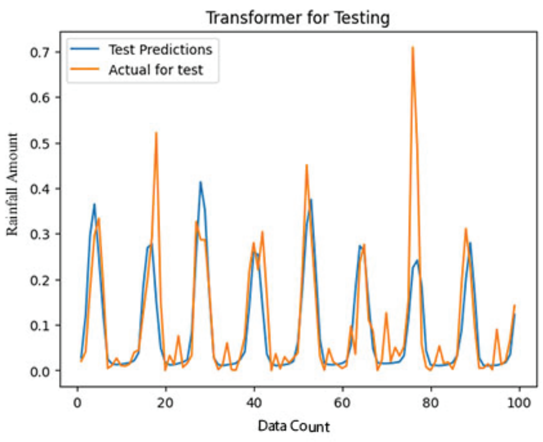

Figure 9: Actual vs Testing Predictions

4.5 LSTM (Long Short-Term Memory)

The Long Short-Term Memory (LSTM) network is a specialized type of Recurrent Neural Network (RNN) designed to manage and learn from long-term dependencies in sequential data. They are well-suited for processing sequences of data with long-term dependencies, such as time series or natural language, by maintaining a cell state that can capture and remember information over time. LSTMs accomplish this by employing a system of gates that manage the flow of information through the network, enabling them to selectively keep or discard information as necessary. While LSTMs are effective for modeling sequences, they may struggle with capturing global dependencies across long sequences due to their sequential nature and limitations in parallelization.

LSTM are well-suited for processing sequences of data with long-term dependencies, such as time series or natural language, by maintaining a cell state that can capture and remember information over time. LSTMs accomplish this by employing a system of gates that manage the flow of information through the network, enabling them to selectively keep or discard information as necessary. While LSTMs are effective for modeling sequences, they may struggle with capturing global dependencies across long sequences due to their sequential nature and limitations in parallelization.

4.5.1 LSTM Architecture:

The architecture of the LSTM model implemented in this work is as follows:

Table 2

LSTM Architecture

Layer (type)		Output Shape
LSTM	(None, 10, 64)	
Dropout	(None, 10, 64)	
LSTM	(None, 32)	
Dropout	(None, 32)	
Dense	(None, 16)	
Dense	(None, 8)	
Dense	(None, 1)	

The architecture includes two layers, each layer successed by a dropout layer that can prevent overfitting. The first LSTM layer processes input sequences of length 10 and outputs sequences of 64-dimensional hidden states, effectively capturing temporal dependencies within the data. The second LSTM layer further refines the representation to produce a final output of dimensionality 32. Dropout layers are employed after each LSTM layer to randomly drop connections during training, reducing the likelihood of overfitting. The final layers consist of dense (fully connected) layers, with the last layer producing a single output representing the predicted value. Overall, this LSTM architecture comprises approximately 32,033 trainable parameters and is capable of learning complex temporal patterns from sequential input data, making it suitable for tasks such as time series forecasting or sequence prediction.

4.5.2 Model Performance Analysis of LSTM

The model, trained on a dataset of 29,006 rows divided into training, validation, and testing sets of 16,000, 6,000, and 7,006 rows respectively, demonstrated notable performance over 30 epochs. The training phase achieved a minimum loss of approximately 0.062% and a minimum validation loss of 0.058%. The Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE) reached minima of around 0.062% and 0.4%, respectively. Additionally, the Root Mean Square Error (RMSE) was about 0.11 units, reflecting a low magnitude of prediction error. These metrics indicate that the model achieved lower stable and accurate performance throughout the training phase as compared to proposed transformers model.

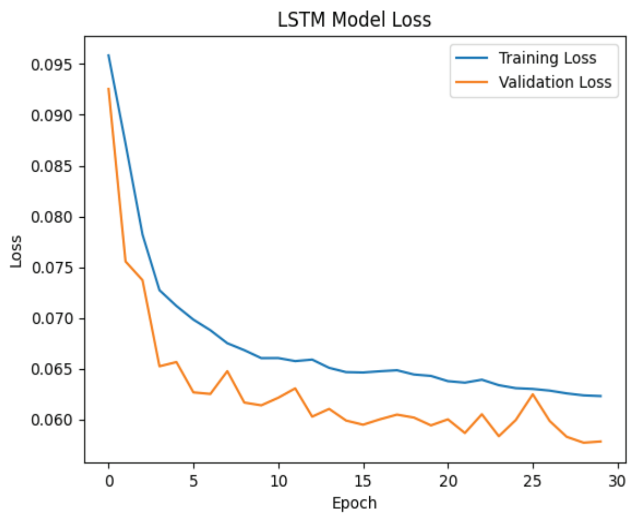

Figure 8: Model Loss LSTM

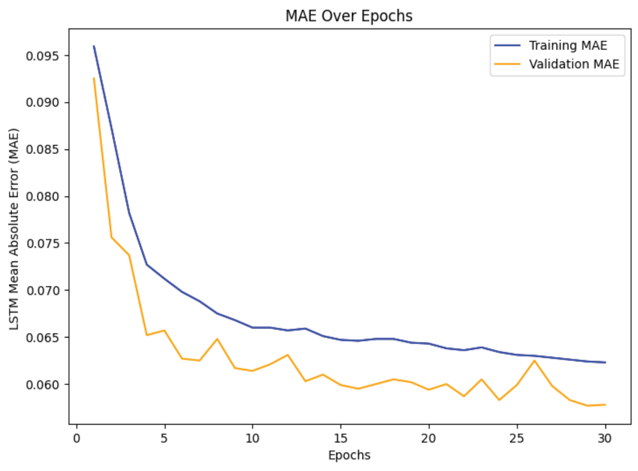

Figure 9: Mean Absolute Error

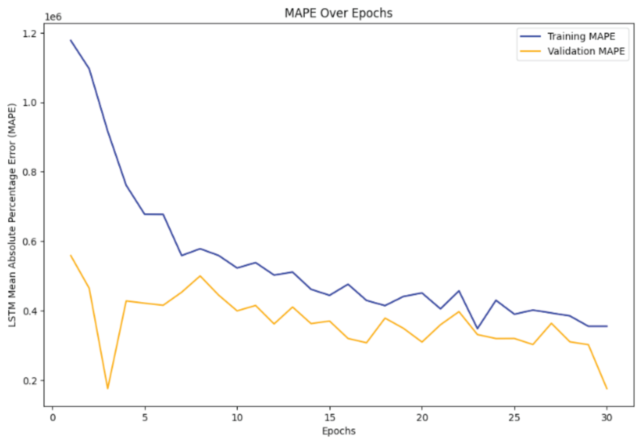

Figure 10: Mean Absolute Percentage Error

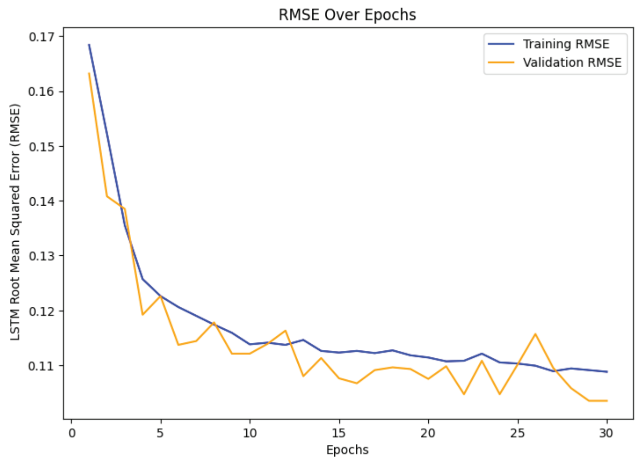

Figure 11: LSTM Root Mean Square Error

4.5.3 Actual vs Prediction of LSTM

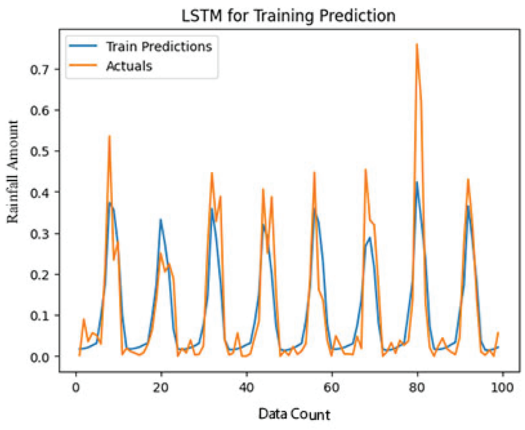

Figure 12: LSTM Actual vs Training Predictions

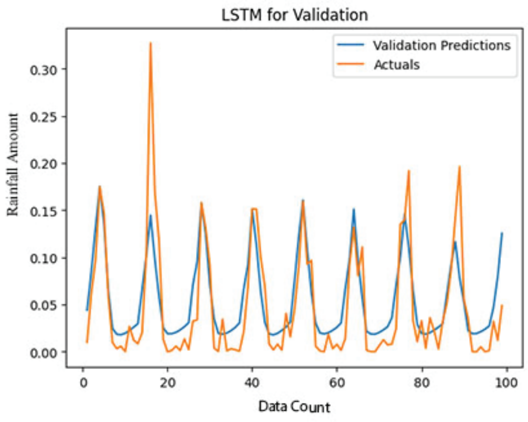

Figure 13: LSTM Actual vs validation Predictions

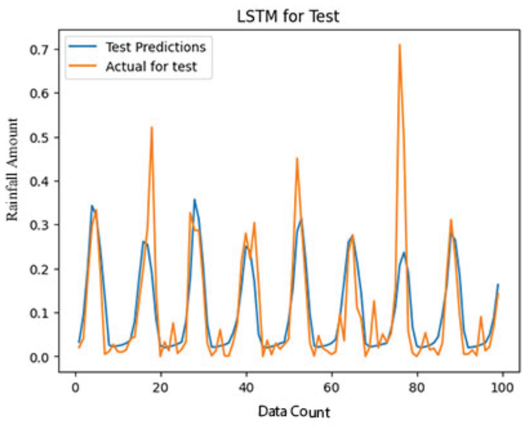

Figure 14: LSTM Actual vs Test Predictions

Conclusion and Future Recommendation

In this work, we propose a model that combines wavelet transform and transformer that predicts rainfall. The datasets used for the experiments consists of 29002 rows of data collected from opendatanepal. Furthermore, it contains data from 1981-01-01 to 2019-12-31 extracted from 93 weather stations spanning 62 districts in Nepal and contains 18 attributes. The data thus obtained is feature extracted using wavelet transform which thereby reduced the number of attributes to nine. The proposed Transformer architecture was designed and hyper parameters such as epochs, batch size, optimizer with learning rate, and dropout were tuned extensively to obtain best performing model configuration. The proposed algorithm has achieved a model loss of 0.056, mean absolute error of 0.05, and root mean square error of 0.10 and prove upper hand when compared with state of art algorithm LSTM. As for LSTM, with the same dataset model loss of 0.062, mean absolute error of 0.062, mean absolute percentage error of 0.062 and root mean square error of 0.12 is measured.

In the future, studies should concentrate on examining how well other variants of Transformer perform when trained on the dataset and how they perform when predicting other weather variable like humidity and temperature.

Acknowledgements

This research was jointly conducted by Nepal College of Information Technology and has completed with joint effort of Supervisor and Colleague.

References

[1] T. Anuradha, P. S. G. Aruna Sri Formal, and J. RamaDevi, "Hybrid model for rainfall prediction with statistical and technical indicator feature set," Expert Systems with Applications, vol. 249, p. 123260, Sep. 2024, doi: 10.1016/j.eswa.2024.123260.

[2] W. M. Ridwan, M. Sapitang, A. Aziz, K. F. Kushiar, A. N. Ahmed, and A. El-Shafie, "Rainfall forecasting model using machine learning methods: Case study Terengganu, Malaysia," Ain Shams Engineering Journal, vol. 12, no. 2, pp. 1651–1663, Jun. 2021, doi: 10.1016/j.asej.2020.09.011.

[3] T. A. Gahwera, O. S. Eyobu, and M. Isaac, "Analysis of Machine Learning Algorithms for Prediction of Short-Term Rainfall Amounts Using Uganda's Lake Victoria Basin Weather Dataset," IEEE Access, vol. 12, pp. 63361–63380, 2024, doi: 10.1109/ACCESS.2024.3396695.

[4] S. Benziane, "Survey: Rainfall Prediction Precipitation, Review of Statistical Methods," WSEAS TRANSACTIONS ON SYSTEMS, vol. 23, pp. 47–59, Jan. 2024, doi: 10.37394/23202.2024.23.5.

[5] Z. Sa'adi et al., "Evaluating Imputation Methods for rainfall data under high variability in Johor River Basin, Malaysia," Applied Computing and Geosciences, vol. 20, p. 100145, Dec. 2023, doi: 10.1016/j.acags.2023.100145.

[6] J. Hou, Y. Wang, J. Zhou, and Q. Tian, "Prediction of hourly air temperature based on CNN–LSTM," Geomatics, Natural Hazards and Risk, vol. 13, no. 1, pp. 1962–1986, Dec. 2022, doi: 10.1080/19475705.2022.2102942.

[7] Q. Wen et al., "Transformers in Time Series: A Survey," in Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, Macau, SAR China: International Joint Conferences on Artificial Intelligence Organization, Aug. 2023, pp. 6778–6786. doi: 10.24963/ijcai.2023/759.

[8] I. Salehin, I. M. Talha, Md. Mehedi Hasan, S. T. Dip, Mohd. Saifuzzaman, and N. N. Moon, "An Artificial Intelligence Based Rainfall Prediction Using LSTM and Neural Network," in 2020 IEEE International Women in Engineering (WIE) Conference on Electrical and Computer Engineering (WIECON-ECE), Bhubaneswar, India: IEEE, Dec. 2020, pp. 5–8. doi: 10.1109/WIECON-ECE52138.2020.9398022.

[9] E. Hernández, V. Sanchez-Anguix, V. Julian, J. Palanca, and N. Duque, "Rainfall Prediction: A Deep Learning Approach," in Hybrid Artificial Intelligent Systems, vol. 9648, F. Martínez-Álvarez, A. Troncoso, H. Quintián, and E. Corchado, Eds., in Lecture Notes in Computer Science, vol. 9648. , Cham: Springer International Publishing, 2016, pp. 151–162. doi: 10.1007/978-3-319-32034-2_13.

[10] G. Chen and W.-C. Wang, "Short-term precipitation prediction using deep learning," Geophysical Research Letters, vol. 49, no. 8, p. e2022GL097904, Apr. 2022, doi: 10.1029/2022GL097904.

[11] S. Dhital, K. Lamsal, S. Shrestha, and U. Bhurtyal, "Forecasting Weather using Deep Learning from the Meteorological Stations Data : A Study of Different Meteorological Stations in Kaski District, Nepal," Jul. 12, 2023. doi: 10.31223/X5CH4H.

[12] J. Kang, H. Wang, F. Yuan, Z. Wang, J. Huang, and T. Qiu, "Prediction of Precipitation Based on Recurrent Neural Networks in Jingdezhen, Jiangxi Province, China," Atmosphere, vol. 11, no. 3, p. 246, Feb. 2020, doi: 10.3390/atmos11030246.