Journal of Computer Science

Research Article

Efficient Rainfall Forecasting Using Sequential Momentum-Based Artificial Neural Networks for Urban Flood Management

Sherin K K[1] and G. Suganthi[2]

[1]*Department of Computer Science, Nesamony Memorial Christian College, Affiliated to Manonmaniam Sundaranar University, Tirunelveli, Tamil Nadu, 627012, India*
[2]*Department of Computer Science, Women's Christian College, Nagercoil, Affiliated to Manonmaniam Sundaranar University Tirunelveli, Tamil Nadu, 627012, India*

Article history
Received: 12-07-2025
Revised: 24-10-2025
Accepted: 22-01-2026

Corresponding Author:
Sherin K K
Department of Computer Science, Nesamony Memorial Christian College, Affiliated to Manonmaniam Sundaranar University, Tirunelveli, Tamil Nadu, 627012, India
Email: sherin.sini@gmail.com

Abstract: Urban floods cause severe disruption in densely populated cities and damage essential public infrastructure. Rapid urbanization and weak drainage systems further increase the risk during heavy rainfall. Accurate short-term rainfall forecasting is required to support early warning and reduce losses. This research focuses on improving rainfall prediction accuracy using an Artificial Neural Network trained with a new optimizer named Sequential Momentum Gradient Descent. The study addresses the drawback of traditional gradient methods that use fixed momentum in training. A fixed momentum reduces adaptability and limits the model's ability to learn changing weather trends. The proposed Sequential Momentum Gradient Descent adjusts momentum dynamically based on the current error rate. This adaptive process allows faster convergence and better stability for time-dependent rainfall data. The optimizer enhances learning efficiency by increasing momentum when the model progresses and reducing it during unstable learning stages. The system uses nine major weather features that represent regional variations in rainfall intensity. Experiments are carried out using weather datasets from Aminjikarai, Velachery, Mudichur, and Mahabalipuram regions in Chennai. The proposed model achieves higher prediction accuracy between 94.7 and 95.6 percent on all datasets. Training time is reduced by more than half compared to other baseline optimizers. The results show that the adaptive momentum control improves rainfall forecasting reliability under diverse climatic patterns. This approach contributes to practical urban flood management systems that depend on precise and fast rainfall prediction.

Keywords: Flood Prediction, Artificial Neural Network, Sequential Momentum Gradient Descent, Time Series Data, Climate Data, Rainfall Forecasting

Introduction

Flooding is a serious problem for large cities (Kundzewicz et al., 2013). Sudden rainfall frequently causes major floods in urban areas (Price and Vojinovic, 2008). The impact of sudden rain is severe in densely populated cities with poor drainage systems (Walczykiewicz and Skonieczna, 2020). Chennai, a city in India, regularly faces flooding issues during unexpected heavy rainfall (Shivakumar et al., 2023). This flooding leads to loss of life and damage to buildings, roads, and essential city services. Predicting sudden heavy rain accurately can significantly reduce these risks by enabling authorities to prepare more effectively (Piran, 2024).

Accurate prediction of heavy rainfall requires reliable weather forecasting methods. Traditional forecasting methods often give inaccurate results due to their inability to handle rapid changes in weather conditions (Tripathy et al., 2021; Potter et al., 2025). Artificial Neural Networks (ANN) have improved rainfall prediction by learning patterns from past weather data. ANNs use historical weather parameters such as temperature, humidity, wind speed, and rainfall (Bączkiewicz et al., 2021; Barrera-Animas et al., 2021). These parameters are used to find patterns and forecast future rain. Existing ANN methods have yielded promising results in rainfall and flood prediction (Mosavi et al., 2018; Elsafi, 2014).

However, they still struggle to predict sudden and extreme rainfall events accurately. Existing methods usually treat weather data as individual data points without considering the sequential and dynamic nature of weather changes (Das et al., 2024). Standard gradient descent methods commonly used in Artificial Neural Networks (ANNs) have fixed momentum values. This fixed momentum approach results in slow training and unstable prediction outcomes. It also does not adapt well to changes in weather patterns over time (Litta et al., 2013; Rajendra et al., 2019). These weaknesses limit the ability of existing Artificial Neural Network (ANN) models to predict sudden rain events reliably.

The primary issue is the inadequate handling of time series weather data by standard Artificial Neural Network (ANN) models. Weather data changes continuously, with rainfall intensity fluctuating rapidly within short periods (Shekar et al., 2024). Traditional ANN methods do not adjust their learning approach dynamically during training. Due to this, existing models often fail to predict rainfall accurately, especially during rapidly changing weather conditions. This inaccurate prediction becomes a critical problem in city flood management systems. Cities such as Chennai experience heavy flooding due to sudden, unexpected rainfall. Therefore, an ANN method is needed that efficiently and accurately handles time series data.

This research introduces a new method called Sequential Momentum Gradient Descent (SMGD). The proposed SMGD solves the existing problems faced by traditional gradient descent methods in ANN training. Standard gradient descent methods use fixed momentum throughout the training process. In contrast, SMGD dynamically changes the momentum during training based on the error rate. At each training epoch, the error is calculated, and the momentum value is adjusted. If the error decreases compared to the previous epoch, the momentum increases by a small amount. If the error increases, the momentum value is reduced by half. This adaptive momentum adjustment accelerates learning during periods when the model makes accurate predictions. It also reduces the momentum value when prediction becomes less precise, preventing significant errors and instability. The adaptive momentum adjustment provided by SMGD is particularly beneficial for weather data. Weather parameters, such as temperature, humidity, wind speed, and rainfall, often show unpredictable and rapid changes. The SMGD approach quickly learns these changing patterns by adjusting its momentum according to the prediction errors at each step. This method prevents the model from overshooting the best solution and ensures smooth and stable learning. Thus, SMGD significantly improves the prediction accuracy and training speed of ANNs in handling dynamic weather data.

This study tests the proposed ANN-SMGD method using real weather data from the Chennai region. Data from four key locations within Chennai, namely Aminjikarai, Velachery, Mudichur, and Mahabalipuram, are used for training and evaluating the model. The dataset includes daily weather data collected from 2015 to 2020 by the Chennai Meteorological Department. Before training the ANN, data preprocessing methods, such as interpolation and normalization, are employed. These preprocessing steps ensure that the ANN receives well-formatted data for efficient learning. The proposed ANN-SMGD method is trained using 80% of the collected dataset. The remaining 20% is reserved for testing and evaluating the model's performance. The performance is measured in terms of accuracy, precision, recall, and F1-score. Results from the testing datasets show that ANN-SMGD achieves high accuracy levels across all four locations studied. For example, the model achieves 95.6% accuracy on the Aminjikarai dataset and 94.7% accuracy on the Velachery dataset. Precision, recall, and F1-score values also remain high, indicating that the model reliably predicts rainfall. Furthermore, the ANN-SMGD method significantly reduces training and testing time compared to standard methods. Training is completed in approximately 13.7 minutes, and testing occurs within 430 milliseconds. This fast prediction capability supports real-time decision-making during heavy rainfall events.

Literature Review

Several studies have been conducted to improve flood prediction using Artificial Neural Networks (ANN) and related methods. Shekar et al. (2024) studied rainfall-runoff modelling using ANN, CNN, and a combined CNN-RNN model in the Bardha Watershed, India. They used rainfall, maximum and minimum temperature, and discharge data. Their results showed that the CNN-RNN model outperformed the ANN and other models. However, the study faced limitations in computational complexity due to the integration of deep models, which affected real-time flood forecasting.

Saharudin et al. (2023) proposed a flood forecasting system using weather parameters like rainfall and temperature. They used Recurrent Neural Networks (RNN) to predict flood conditions. RNNs worked well with sequential weather data. However, the model struggled to handle sudden changes in weather patterns, which reduced prediction accuracy during extreme events.

Abebe and Endalie (2023) compared ANN and Adaptive Neuro-Fuzzy Inference System (ANFIS) models for rainfall prediction at meteorological stations in Ethiopia. They used geographical and periodicity components instead of climatic parameters. Although ANN showed good performance, the ANFIS model outperformed ANN in all cases. A major drawback in

their work was the exclusion of real-time weather parameters, which limited flood forecasting applications.

Dai et al. (2021) developed an ensemble learning approach for coastal flood forecasting using Internet of Things (IoT) data. They combined Backpropagation Neural Networks (BPNN) and Random Forest models with a Bayesian model combination. This method improved prediction accuracy. However, the complex structure required large computational resources, which reduced the speed of real-time predictions.

Khateeb et al. (2025) assessed the effectiveness of ANN models in spatial downscaling of precipitation data. Their study focused on increasing the spatial resolution of precipitation data from 20 km to 5 km. Although the ANN model worked well during winter, it showed poor performance during summer due to the high variability in rainfall. This limited the model's reliability for flood prediction in all seasons.

Thankappan et al. (2023) proposed an Adaptive Momentum Backpropagation (AM-BP) algorithm for flood prediction and management using IoT data. They integrated an adaptive momentum method with the backpropagation algorithm to improve prediction accuracy. The model achieved 96% accuracy. However, the approach required large high-performance computing resources, which made it less suitable for small-scale applications.

Komiya et al. (2025) introduced Informed Neural Networks (INNs) for flood forecasting with limited training data. Their model incorporated domain knowledge to improve prediction when data were scarce. Although INNs performed well under data-limited conditions, their dependence on expert domain knowledge restricted their broader application across different regions.

Khoirunisa et al. (2021) develop an Artificial Neural Network model combined with weather API data to predict rainfall and calculate runoff in the Jenelata Sub-watershed. The model uses spatial and hydrological data processed with the SCS method. The main drawback is the incomplete and inconsistent rainfall data, which lowers the accuracy of the predictions. Another study uses a multilayer perceptron neural network to forecast floods based on historical flood data. The performance is measured using the Nash coefficient and Root Mean Square Error values. The model shows good forecasting ability but suffers from high computational cost and a strong tendency to overfit during training.

René et al. (2018) showcased a remote-controlled and dynamic pluvial flood forecasting system for Castries, Saint Lucia. Dynamic forecasting is crucial for mitigating flooding in urban areas. Data was fed into the MS SQL database server using a specific command and response protocol. After sending the query to the counting device, the host computer received a response. The method involved a process of bias adjustments and uncertainty quantification. Forecasting was enhanced by including natural rainfall projections for the next three hours, which were then combined to create a 12-hour forecast. The data was divided into three categories based on percentile thresholds, which helped evaluate the forecasts.

Song et al. (2019) proposed an analysis method for flood forecasting and issuing appropriate warnings, focusing on urban-stream flooding. The method considered water level changes during rainfall. Water level analysis was performed by estimating upstream and downstream water levels about rainfall intensity and duration. The Hydrologic Engineering Centre's Hydrologic Modelling System (HEC-HMS) and River Analysis System (HEC-RAS) were utilized to calculate rainfall runoff and water flow rates. A flash flood warning was issued due to changes in backwater conditions. The results were compared to real-time water level observations from Dorim Stream.

Dai et al. (2021) suggested an artificial neural network-based model for predicting flood depth in the following hour. The model's performance was demonstrated through a case study of urban flooding in Macau, China, during six typhoons. Urban weather, geographic elevation, flood typhoon optimal track, tides, and submerged area water depth increment were evaluated as inputs, with flood depth as the output. The model was trained and tested by constructing four models with different sample sizes. The work could be improved by addressing the model's inadequate input. The provided model's structure is overly complex, and the predicted value is excessively high.

Table 1 compares the state-of-the-art methods for flood prediction using ANN and their variants. Existing studies demonstrate that ANN-based models can enhance flood prediction by utilizing climate and weather parameters. However, they suffer from problems such as slow learning, poor handling of time-series weather changes, dependence on complex computation, and lack of adaptability to sudden weather shifts. These problems affect the speed and accuracy of early flood warnings, especially in metropolitan cities with complex environmental dynamics. This research proposes a Sequential Momentum Gradient Descent-based Artificial Neural Network (ANN) model to address these limitations by enhancing training speed and accuracy for early flood prediction.

Proposed Sequential Momentum Gradient Descent

Gradient Descent is a widely used optimization method for training Artificial Neural Networks (ANNs) (Mehmood et al., 2023). It updates the weights and biases of the network to reduce the error between predicted and actual outputs (Cheridito et al., 2022). In standard Gradient Descent, the weights are updated based on the negative gradient of the cost function (Abiodun et al., 2018). The basic weight update rule is given by:

Table 1: State-of-the-Art Comparison of Flood Prediction Methods Using ANN and Variants

Author (Year)	Purpose	Methods and Algorithms Used	Significance	Disadvantages
Shekar et al. (2024)	Rainfall-runoff modelling in Bardha Watershed	ANN, CNN, CNN-RNN	CNN-RNN achieved better prediction	High computational complexity limits real-time use
Saharudin et al. (2023)	Flood forecasting using weather parameters	RNN	Effective in handling sequential data	Poor performance during sudden weather changes
Abebe and Endalie (2023)	Rainfall prediction at meteorological stations	ANN, ANFIS	ANFIS outperformed ANN	No use of real-time weather parameters
Dai et al. (2021)	Coastal flood forecasting with IoT data	BPNN, Random Forest, Bayesian combination	Increased accuracy with ensemble model	High computational demand reduced real-time speed
Khateeb et al. (2025)	Spatial downscaling of precipitation data	ANN	Good performance in winter seasons	Poor summer performance limits overall reliability
Thankappan et al. (2024)	Flood prediction and management using IoT data	Adaptive Momentum Backpropagation	Achieved 96% prediction accuracy	Required high-performance computing resources
Komiya et al. (2025)	Flood forecasting with limited data	Informed Neural Networks (INNs)	Good performance with small datasets	Required strong expert domain knowledge
Khoirunisa et al. (2021)	Rainfall and runoff prediction in Jenelata	ANN, SCS method	Combined weather and spatial data	Incomplete rainfall data lowered prediction accuracy
René et al. (2018)	Dynamic pluvial flood forecasting in urban areas	Remote-controlled system, MS SQL-based	Dynamic short-term forecasting improved	Complex system integration required
Song et al. (2019)	Urban stream flood warning system	HEC-HMS, HEC-RAS models	Effective flash flood warnings issued	Depended heavily on precise upstream and downstream data
Dai et al. (2021)	Short-term flood depth prediction in Macau	ANN model	Evaluated urban flood prediction for typhoons	Complex model structure and excessive predicted values

$$W_{\text{new}} = W_{\text{old}} - \alpha \frac{\partial C}{\partial W} \quad (1)$$

Where W denotes the weight, α is the learning rate, and $\frac{\partial C}{\partial W}$ is the gradient of the cost function C with respect to W

Although Gradient Descent works well in many applications, it struggles to handle time series data effectively. Weather data, which includes parameters such as temperature, humidity, and rainfall, fluctuates continuously over time. Standard Gradient Descent treats each data point independently without considering the sequential nature of time series. This results in poor learning of temporal patterns and limits the ANN's ability to predict sudden changes in weather conditions.

Another problem with Gradient Descent is its slow convergence (Khan et al., 2021). The method uses a fixed learning rate during training, which does not adapt to the changing nature of the error landscape. As a result, the model either converges very slowly or oscillates around minima without reaching an optimal solution. This behavior worsens when training with time series data where patterns change dynamically over time. To address these limitations, the proposed method introduces a Sequential Momentum Gradient Descent approach. This method extends the standard Gradient Descent by dynamically adjusting the momentum during training based on the error rate. It enhances learning, particularly when working with sequential and time-dependent datasets, such as climate data. In the standard Momentum Gradient Descent, the velocity term is introduced to speed up convergence. The velocity update and weight update equations are:

$$v_{\text{new}} = \beta v_{\text{old}} - \alpha \frac{\partial C}{\partial W} \quad (2)$$

$$W_{\text{new}} = W_{\text{old}} + v_{\text{new}} \quad (3)$$

Where β is the momentum coefficient and v is the velocity. However, in traditional methods, β is fixed throughout the training process. A fixed momentum cannot adjust to the changing error behavior in time series data.

Sequential Momentum Gradient Descent proposes an adaptive momentum adjustment based on the error rate between epochs. The method begins by initializing the weights, biases, and a small initial momentum value β_0. The initial velocity v is set to zero. For each epoch, the error rate e is calculated using the mean squared error:

$$e = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad (4)$$

Where y_i is the actual output, $\hat{y}_i$ is the predicted output, and n is the number of training samples. The

momentum β is updated based on the change in error. If the current error e is less than the previous error e_{prev}, then the momentum is increased:

$$\beta = \beta_{\text{prev}} + \Delta\beta \tag{5}$$

Where $\Delta\beta$ is a small positive value. If the current error e is greater than the previous error e_{prev}, then the momentum is reduced by half:

$$\beta = \frac{\beta_{\text{prev}}}{2} \tag{6}$$

After updating the momentum, the new velocity and weight updates are calculated using the equations:

$$v_{\text{new}} = \beta v_{\text{old}} - \alpha \frac{\partial C}{\partial W} \tag{7}$$

$$W_{\text{new}} = W_{\text{old}} + v_{\text{new}} \tag{8}$$

The method continuously adjusts β at every epoch based on the observed error. This dynamic adjustment allows the network to speed up learning when the model is moving in the right direction and to slow down learning when the model faces difficulties.

Sequential Momentum Gradient Descent offers several improvements over the standard Gradient Descent. First, it speeds up convergence by increasing momentum when the model learns well. Second, it prevents overshooting by reducing momentum when the model struggles. Third, it captures sequential patterns in time series data more effectively because the training adapts to the dynamic nature of the data. Fourth, it provides smoother learning curves and better stability during training. By addressing the issues of slow learning and poor time series handling, Sequential Momentum Gradient Descent enhances the training efficiency and prediction accuracy of ANN models. It helps the ANN model learn both short-term fluctuations and long-term patterns in weather data, such as temperature, humidity, and rainfall. This is very important for early and accurate flood prediction in metropolitan cities where rapid changes in weather conditions can lead to severe flooding events. Algorithm 1 and Figure 1 show the entire flow of the proposed SMGD for ANN Training.

Algorithm 1 Sequential Momentum Gradient Descent (SMGD) for ANN Training

1: Initialize weights W and biases b
2: Set learning rate alpha
3: Set initial momentum beta 0
4: Initialize velocity V to 0
5: Set initial previous error e prev to a large value
6: Set momentum adjustment step delta beta
7: **for** each epoch **do**
8: Perform forward pass and compute output y pred
9: Compute error e as mean squared error
10: **if** e less than e prev **then**
11:Update momentum beta as beta prev plus delta beta
12: **else**
13: Update momentum beta as beta prev divided by 2
14: **end if**
15: Compute gradients dW and db
16:Update velocity V as beta times V minus alpha times dW
17: Update weights W as W plus V
18: Update biases b as b plus beta times V
19: Update previous error e prev as e
20: Update previous momentum beta prev as beta
21: **end for**
22: Return updated weights W and biases b

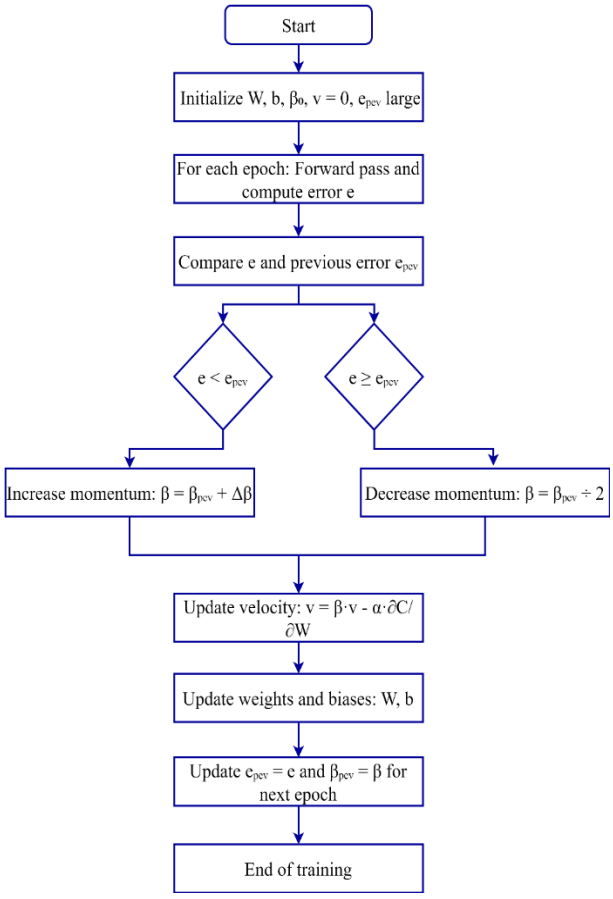


Fig. 1: The entire flow of proposed SMGD for ANN Training

Materials and Methods

The proposed methodology uses weather data from Aminjikarai, Velachery, Mudichur, and Mahabalipuram for early rainfall prediction. It comprises three main stages: Preprocessing, rainfall classification using an

Artificial Neural Network (ANN), and dynamic optimization using the Sequential Momentum Gradient Descent algorithm. Preprocessing applies linear interpolation, normalization, and integer mapping to prepare the input data. The ANN model processes the features to classify rainfall intensity. The training process adjusts learning using adaptive momentum to improve accuracy and convergence. Figure 2 shows the overall process flow of the proposed rainfall prediction and optimization method.

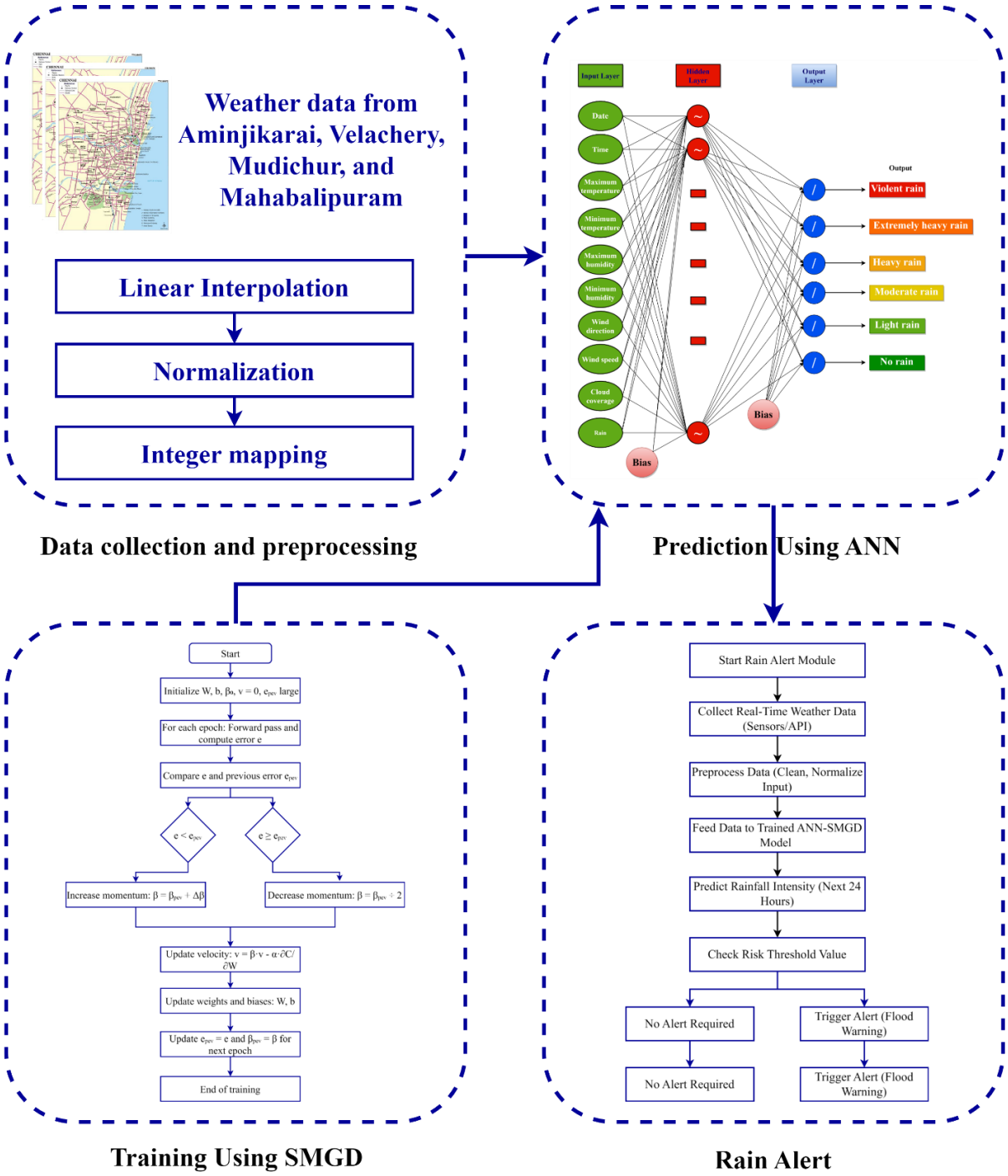


Fig. 2: Overall process flow of the proposed rainfall prediction and optimization method

Preprocessing Module for Weather Data

The preprocessing module in this research prepares raw weather data for effective learning and prediction. The raw dataset contains temperature, humidity, wind speed, cloud coverage, and rainfall records. These records are collected at different times and may contain missing values, irregular formats, and different ranges. Such issues reduce the learning efficiency and accuracy of Artificial Neural Networks (ANN). The preprocessing module addresses these problems through two key steps: Handling missing values and data normalization. The first step in preprocessing is the identification and handling of missing values. Missing entries are replaced using linear interpolation. This method fills the missing value based on the average of its neighboring values. Let x_i and x_{i+2} be the known values, and x_{i+1} be the missing value. The interpolated value is computed as:

$$x_{i+1} = \frac{x_i + x_{i+2}}{2} \quad (9)$$

This method ensures that the time dependency in weather sequences remains preserved. After interpolation, the dataset becomes complete and ready for scaling. The second step is normalization. Each numeric feature is scaled to a common range between 0 and 1. This prevents features with large ranges from dominating others. Min-Max normalization is applied using the following Equation:

$$x_{\text{norm}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \quad (10)$$

Here, x is the original value, $x_{\min}$ is the minimum value of the feature, and $x_{\max}$ is the maximum value. This transformation helps maintain numerical stability during ANN training. Categorical attributes, such as wind direction and cloud coverage, are encoded using an integer mapping. Each unique category is assigned a fixed numeric code. For example, "North" is mapped to 1, "South" to 2, and so on. This transformation converts qualitative data into a usable form for the ANN.

The cleaned and normalized data are then stored in a structured format. Each record now contains consistent and scaled features. This standard format ensures that the ANN can learn the patterns in weather parameters without bias from data irregularities. The preprocessing module directly improves the accuracy, convergence speed, and reliability of the prediction system.

Proposed ANN Architecture for Flood Prediction

ANNs are highly effective in learning from complex and nonlinear data patterns, which is critical for predicting floods based on weather parameters (Alaloul et al., 2018). Their layered structure enables deep information processing, allowing the model to discover hidden relationships among inputs like temperature, humidity, and rainfall. The basic architecture of an ANN consists of three major layers: The input layer, the hidden layers, and the output layer (McCabe and Denham, 2021; Singh et al., 2021). Each layer contains nodes called neurons. Figure 3 shows the basic structure of a standard ANN.

In the input layer, each neuron represents one input feature. These inputs are processed and transferred to hidden layers through weighted connections. Each connection has an associated weight, and each neuron applies an activation function to the weighted sum of its inputs. The purpose of hidden layers is to learn complex patterns from the input features. Finally, the output layer processes the final result based on the learned patterns. Mathematically, the working of a neuron in a hidden layer can be described by:

$$z = \sum_{i=1}^{n}(w_i x_i) + b \quad (11)$$

Where z is the input to the activation function, w_i are the weights, x_i are the input features, and b is the bias term. The output a of the neuron is obtained by applying an activation function f to z:

$$a = f(z) \quad (12)$$

In this research, the Rectified Linear Unit (ReLU) activation function is selected for the hidden layers. ReLU is defined as:

$$f(x) = \max(0, x) \quad (13)$$

The ReLU function introduces non-linearity into the network, improving convergence speed by mitigating the vanishing gradient problem. The proposed Artificial Neural Network (ANN) architecture for flood prediction is illustrated in Figure 4. The input layer consists of nine features related to weather conditions. These features include date, time, maximum temperature, minimum temperature, maximum humidity, minimum humidity, wind direction, wind speed, cloud coverage, and rainfall.

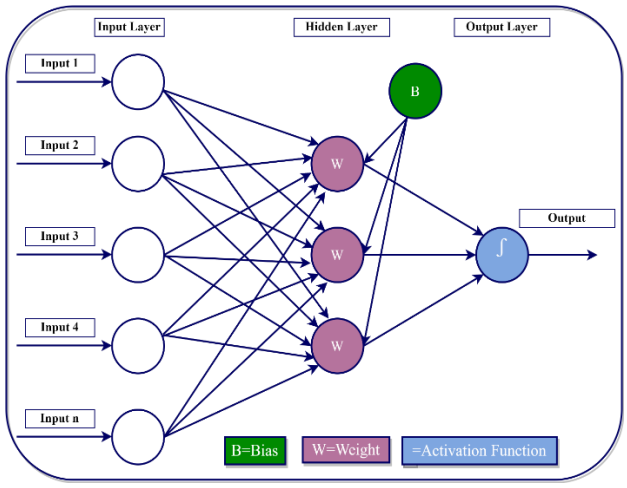


Fig. 3: The basic structure of a standard Artificial Neural Network (ANN)

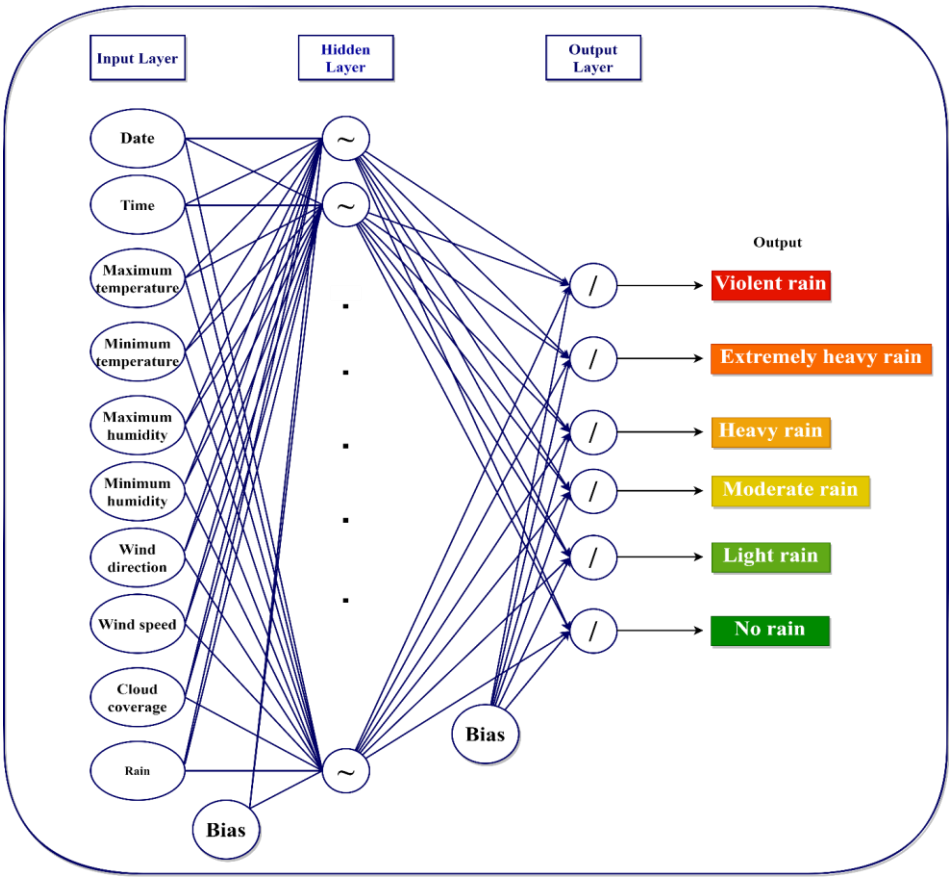


Fig. 4: Proposed ANN architecture for flood prediction based on weather data

Each input feature is represented as a separate green node. This layer collects essential environmental parameters that influence rainfall intensity and the occurrence of flooding events. The wide range of features ensures that the ANN captures the complete atmospheric situation.

The design of the hidden layer is critical for capturing nonlinear relationships between input parameters. The proposed model uses two hidden layers. Each hidden layer contains eight neurons, allowing for a balance between learning capability and computational efficiency. The number of neurons is selected after several experimental trials to achieve stable training and good prediction accuracy. Each neuron in the hidden layers uses the ReLU activation function to process the weighted sum of the previous layer's outputs.

The output layer consists of a single neuron with a linear activation function. The linear activation is suitable for regression tasks, such as predicting a continuous flood risk score. The final prediction output is a value that represents the likelihood or severity of flooding, which supports early warning systems.

The overall computation from input to output can be expressed as:

$$\text{Output} = f(W_2 \times f(W_1 \times X + b_1) + b_2) \quad (14)$$

Where X is the input feature vector, W_1 and W_2 are the weight matrices for input to hidden and hidden to output layers, b_1 and b_2 are the bias vectors, and f represents the activation functions used at each layer.

Bias terms are included in each hidden and output layer to improve the model's flexibility. A bias allows the activation function of a neuron to shift along the input axis, enabling better fitting of data points during training. The inclusion of bias is shown clearly in Figure 4. The SMGD algorithm is used to train the proposed ANN. SMGD dynamically adjusts the momentum during training based on error trends, thereby improving learning speed and enabling the model to adapt to the dynamic nature of weather data.

Rain Alert

The rain alert module shown in Figure 5 illustrates the complete operation of real-time rainfall risk prediction and alert generation. The process begins with the activation of the rain alert module. This component is continuously monitored to ensure uninterrupted tracking of local weather conditions.

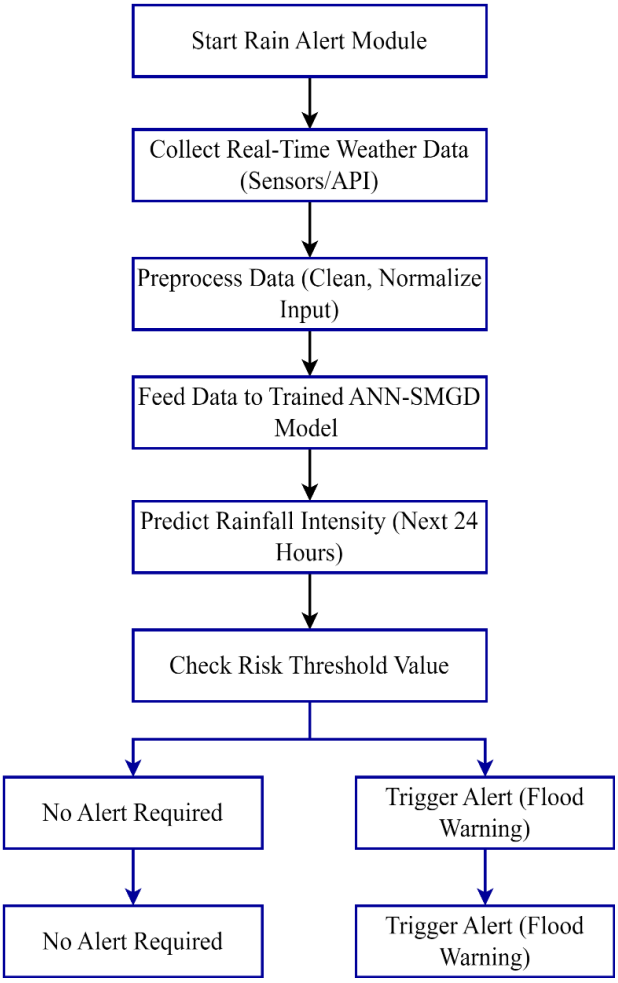


Fig. 5: Process Flow of Rain Alert Module Based on ANN-SMGD Model

Once initiated, the system collects real-time weather data. This data is obtained from sensors or trusted weather APIs. These weather readings include temperature, humidity, wind direction, wind speed, cloud coverage, and rainfall. The collected raw weather data then undergoes a preprocessing stage. This step prepares the data for effective learning and prediction. The processed data is then passed into the trained ANN-SMGD model. This neural model uses an improved learning mechanism based on Sequential Momentum Gradient Descent. It receives the inputs and generates predictions regarding rainfall intensity over the next 24 hours. This prediction result includes classified rainfall categories such as light, moderate, heavy, and extremely heavy rain. The rainfall classification follows four quantitative thresholds based on daily precipitation levels. Light rain is below 15 mm, moderate rain ranges between 15 mm and 64 mm, and heavy rain covers 65 mm to 124 mm. Extreme rain occurs when daily precipitation exceeds 125 mm and triggers the flood alert module.

After prediction, the system checks the rainfall category against a defined risk threshold. This threshold defines the level beyond which the system considers a rainfall event as a potential flood threat. The decision module compares the predicted rainfall value to this threshold. If the predicted rainfall value remains below the threshold, the system concludes that no immediate risk exists. The module then remains inactive until the next data input arrives. If the predicted rainfall exceeds the threshold, the system identifies a flood risk. At this stage, the module initiates a warning response. The system activates the flood warning alert. This alert is sent to relevant authorities and automated public notification systems. This ensures a fast response in high-risk areas. This process provides early and accurate warning signals to minimize the impact of flooding. It supports time-critical decision-making in urban and semi-urban environments. By combining real-time weather data, effective preprocessing, and optimized prediction, this alert module increases city-level preparedness during monsoon or extreme rainfall periods.

Results and Discussion

System Details

The simulation results for the suggested flood forecasting model were generated using an AI server with the following configuration: Intel-Xeon W-2245 (16.5 MB cache, 8 cores, 16 threads, 3.90 GHz to 4.70 GHz Turbo, 155 W), NVIDIA RTX A5000, M.2 512GB PCIe NVMe Solid State Drive (SSD) and 128 GB, DDR4-RAM. The proposed forecasting model training and parameter tuning were carried out using Python machine learning and deep learning packages, which include (Tensorflow, PyTorch, scikit-learn, and numpy). These Python packages make it simple to construct a variety of AI models in this comparative analysis.

Study Area

The study area for this research is the Chennai basin, located along the southeastern coast of India. Chennai, a major metropolitan city, experiences frequent flooding due to heavy rainfall, combined with urban expansion and a poorly maintained drainage network. The Chennai basin is characterized by a complex network of rivers, lakes, and water bodies that interact with the urban landscape, as shown in Figure 6(a). The natural landscape of the Chennai basin features important water systems, including the Coovum and Adyar rivers. These rivers, along with numerous natural lakes, have historically managed floodwaters during periods of heavy rainfall. Figure 6(a) illustrates the distribution of water bodies across the basin before significant human interference. Areas covered by green on the map represent natural water storage zones that reduce the

impact of floods under natural conditions. Flood possibilities under natural landscape conditions, with minimal human activities, are illustrated in Figure 6(b). In this scenario, flood-prone regions are primarily limited to areas near rivers and lakes. The purple areas represent zones with moderate flood risks. These risks are primarily due to the natural accumulation of rainwater in low-lying regions. The map clearly shows that under natural conditions, large portions of the Chennai basin remained safe from flood risks. However, due to rapid urbanization, industrial expansion, and encroachments on natural drainage systems, flood vulnerability has increased drastically. Figure 6(c) presents the flood possibilities in the landscape influenced by human activities. The map indicates a significant increase in flood-prone areas, shown in purple. Built-up areas have expanded across natural water flow paths and storage zones, resulting in severe waterlogging during intense rainfall events. The comparison between Figures 6(b) and 6(c) clearly shows the adverse impact of unplanned development on the flood risk profile of the Chennai basin. Urbanization has reduced the natural infiltration of rainwater and blocked traditional drainage paths, resulting in surface runoff accumulating rapidly. This accumulation results in flash floods even during moderate rainfall.

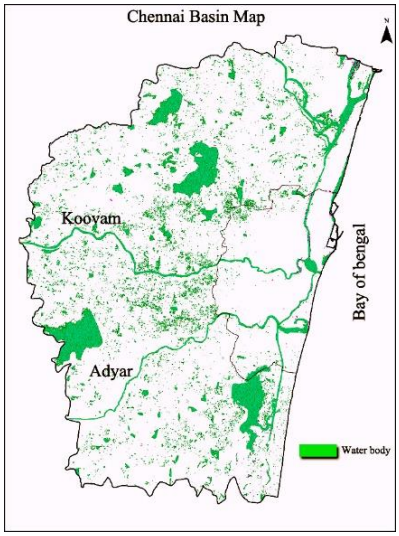


Fig. 6 (a): Chennai basin map

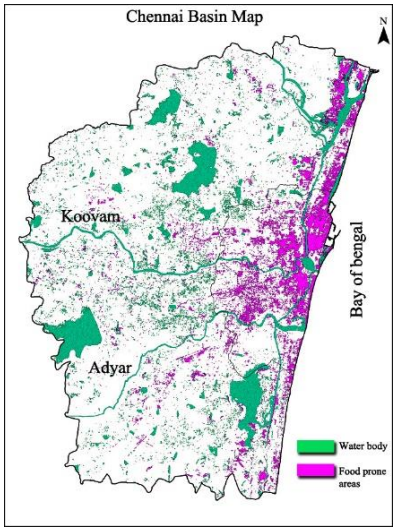


Fig. 6 (b): Flood possibilities in the landscape without the influence of human activities

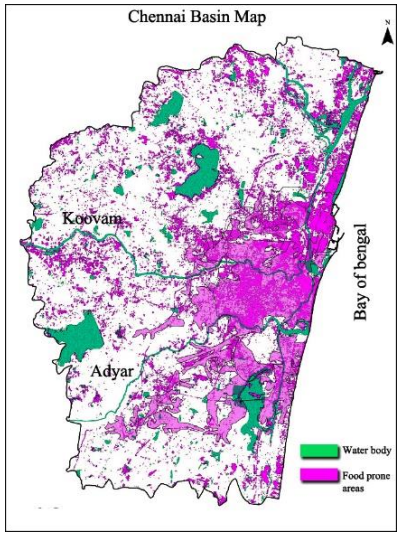


Fig. 6 (c): Flood possibilities in the landscape due to human activities such as urbanization and industrialization

Data Set Details

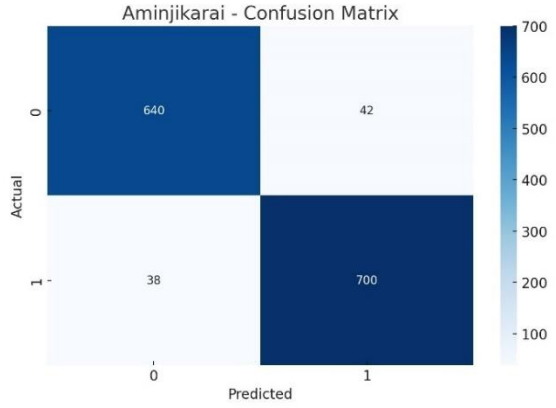
The dataset used in this research is collected from the Chennai Meteorological Department. The data covers the period from 2015 to 2020 and focuses on four critical regions: Aminjikarai, Velachery, Mudichur, and Mahabalipuram. These regions are situated within the Chennai basin and are highly susceptible to flooding during heavy rainfall. The dataset contains detailed weather information collected daily with multiple readings for each day. A total of 17,520 records were obtained. Each day includes various readings to capture the changing weather conditions throughout the day. This detailed data collection supports the development of a more accurate prediction model, particularly for sudden rainfall and flood events. The dataset was separated into two parts for model development. Eighty percent of the data was used for training, and twenty percent was used for testing. Table 2 presents the list of attributes collected. Each record includes the date and time when the weather data was measured. Temperature attributes contain the daily maximum and minimum temperatures at each location. These values are recorded as numbers and help in understanding the heat patterns that affect evaporation and rainfall. Humidity information includes the daily maximum and minimum humidity. Humidity controls atmospheric stability and influences rainfall formation, and it is a key attribute for flood prediction. Wind conditions are described using two attributes: Wind direction and wind speed. Wind direction is stored as text indicating the main wind flow on a specific day. Wind speed is recorded numerically and captures the strength of winds that can impact rainfall patterns. Cloud coverage information is included as a text attribute describing the cloud presence at the location. Cloud coverage strongly influences solar radiation and the likelihood of rainfall. The final attribute is rain, recorded as the maximum daily precipitation in numeric form. This attribute provides a direct measurement of rainfall intensity.

Table 2: Weather attributes details

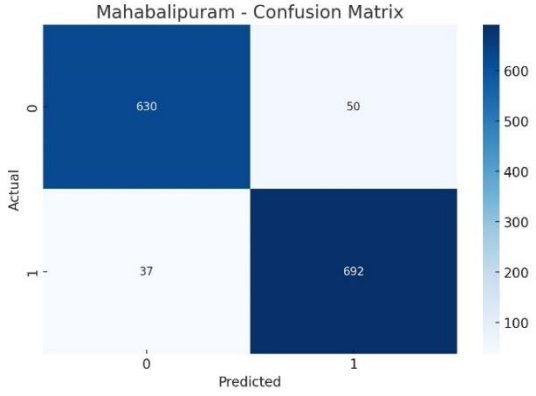
Attribute Name	Attribute Details	Data Type
Date	Date when the weather data was obtained	String
Time	The time when the weather data was obtained	String
Maximum temperature	Daily maximum temperature at a specific location	Numeric
Minimum temperature	Daily minimum temperature at a specific location	Numeric
Maximum humidity	Daily maximum humidity at a specific location	Numeric
Minimum humidity	Daily minimum humidity at a specific location	Numeric
Wind direction	Wind direction for a specific day	String
Wind speed	Wind speed on a specific day	Numeric
Cloud coverage	Cloud coverage details of the location where rain falls	String
Rain	Maximum daily precipitation at a specific location	Numeric

State-of-the-Art Accuracy Comparison

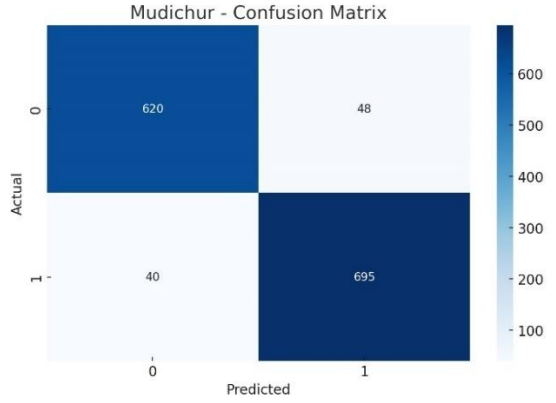
The performance of the proposed ANN-SMGD model was evaluated on real-world weather datasets from four locations: Aminjikarai, Velachery, Mudichur, and Mahabalipuram. Each dataset contained 3,504 records, representing 20% of the total dataset collected. The evaluation considered accuracy, precision, recall, and F1-score. The results were derived based on the confusion matrices plotted for each location (Fig. 7). The confusion matrix for Aminjikarai (Fig. 7a) shows that the ANN-SMGD model achieves strong performance with high true positive and true negative values. The confusion matrix for Mahabalipuram (Fig. 7b) indicates slight misclassifications but still maintains excellent predictive reliability. For Mudichur (Fig. 7c), the model demonstrates robust results with balanced performance across both classes. The confusion matrix for Velachery (Fig. 7d) also exhibits consistent performance with slightly higher false positives but high correct prediction rates overall.



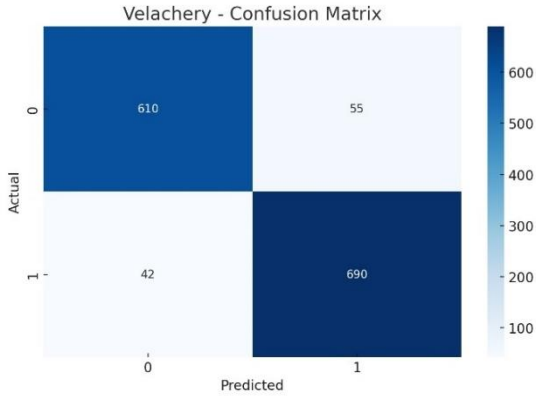
(a) Confusion Matrix for Aminjikarai Dataset



(b) Confusion Matrix for the Mahabalipuram Dataset



(c) Confusion Matrix for Mudichur Dataset



(d) Confusion Matrix for Velachery Dataset

Fig. 7: Confusion Matrices for Rainfall Prediction across Four Locations in the Chennai Basin

The detailed calculation formulas are given as follows:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (15)$$

$$\text{Sensitivity (Recall)} = \frac{TP}{TP+FN} \quad (16)$$

$$Precision = \frac{TP}{TP+FP} \quad (17)$$

$$F1-score = \frac{2(Recall \times Precision)}{Recall+Precision} \quad (18)$$

Tables 3 to 6 present the performance comparison of the proposed model against the existing methods by Shekar et al. (2024); Saharudin et al. (2023); Abebe and Endalie (2023); Dai et al. (2021); Khateeb et al. (2025); Thankappan et al. (202); Komiya et al. (2025); Khoirunisa et al. (2021); René et al. (2018); Song et al. (2019); Dai et al. (2021).

The comparative performance analysis based on the Aminjikarai dataset is presented in Table 3. The ANN-SMGD method achieved an accuracy of 95.6%, which is higher compared to the previous best result of 88.2% achieved by Dai et al. (2021). The precision, recall, and F1-score for ANN-SMGD are 94.3, 94.9, and 94.6%, respectively. These results indicate a significant improvement in capturing true positive events without increasing false positives.

Table 3: State-of-the-Art Accuracy Comparison for Aminjikarai Dataset

Method	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Shekar et al. (2024)	87.5	85.2	88.0	86.5
Saharudin et al. (2023)	85.0	83.4	84.5	83.9
Abebe and Endalie (2023)	84.3	82.5	84.2	83.3
Dai et al. (2021)	88.2	86.0	88.5	87.2
Khateeb et al. (2025)	85.6	83.3	85.1	84.1
Thankappan et al. (2023)	88.0	85.7	87.0	86.3
Komiya et al. (2025)	86.3	83.9	85.6	84.7
Khoirunisa et al. (2021).	84.7	82.0	83.9	82.9
René et al. (2018)	85.8	83.2	84.7	83.9
Song et al. (2019)	86.7	84.3	86.1	85.2
ANN-SMGD (Proposed)	95.6	94.3	94.9	94.6

Table 4: State-of-the-Art Accuracy Comparison for Mahabalipuram Dataset

Method	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Shekar et al. (2026)	86.2	84.1	86.7	85.4
Saharudin et al. (2023)	84.0	82.7	83.5	83.1
Abebe and Endalie (2023)	83.5	81.9	83.3	82.6
Dai et al. (2021)	87.8	85.7	87.5	86.6
Khateeb et al. (2025)	85.2	82.9	84.6	83.7
Thankappan et al. (2023)	87.5	85.2	86.4	85.8
Komiya et al. (2025)	85.7	83.1	85.0	84.0
Khoirunisa et al. (2021)	84.5	81.7	83.8	82.7
René et al. (2018)	85.6	83.0	84.5	83.7
Song et al. (2019)	86.3	83.9	85.6	84.7
ANN-SMGD (Proposed)	95.2	93.5	94.9	94.2

Table 5: State-of-the-Art Accuracy Comparison for Mudichur Dataset

Method	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Shekar et al. (2024)	86.5	84.5	86.9	85.7
Saharudin et al. (2023)	83.8	82.0	83.0	82.5
Abebe and Endalie (2023)	83.0	81.5	82.7	82.1
Dai et al. (2021)	87.5	85.4	87.2	86.3
Khateeb et al. (2025)	85.0	82.6	84.4	83.5
Thankappan et al. (2023)	87.3	85.0	86.2	85.6
Komiya et al. (2025)	85.4	83.0	84.8	83.9
Khoirunisa et al. (2021)	83.9	81.3	83.2	82.2
René et al. (2018)	85.1	82.5	84.0	83.2
Song et al. (2019)	86.1	83.6	85.4	84.5
ANN-SMGD (Proposed)	95.0	93.3	94.6	94.0

Table 6: State-of-the-Art Accuracy Comparison for Velachery Dataset

Method	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Shekar et al. (2024)	86.0	83.9	86.5	85.2
Saharudin et al. (2023)	83.5	82.2	83.3	82.7
Abebe and Endalie (2023)	83.0	81.7	82.5	82.1
Dai et al. (2021)	87.0	85.0	86.8	85.9
Khateeb et al. (2025)	84.9	82.5	84.2	83.3
Thankappan et al. (2023)	87.1	84.8	86.1	85.4
Komiya et al. (2025)	85.2	82.7	84.5	83.6
Khoirunisa et al. (2021)	83.6	80.9	82.8	81.8
René et al. (2018)	84.8	82.2	83.8	83.0
Song et al. (2019)	85.9	83.5	85.2	84.3
ANN-SMGD (Proposed)	94.7	93.0	94.2	93.6

Table 4 presents the accuracy comparison for the Mahabalipuram dataset. ANN-SMGD achieved an accuracy of 95.2%, outperforming the highest accuracy of 87.8% recorded by Dai et al. (2021). The precision, recall, and F1-score for ANN-SMGD are 93.5, 94.9, and 94.2%. This consistency across metrics confirms the robustness of the proposed method under varied weather patterns. Table 5 shows the results for the Mudichur dataset. ANN-SMGD reached an accuracy of 95.0%. The second-best accuracy is 87.5%, as reported by Dai et al. (2021). The precision, recall, and F1-score for ANN-SMGD are 93.3, 94.6, and 94.0%. The high recall value in Mudichur proves that the model correctly detects the majority of rain events. The Velachery dataset results are presented in Table 6 Here, ANN-SMGD achieved an accuracy of 94.7%, higher than the 87.0% obtained by Dai et al. (2021). The precision, recall, and F1-score are 93.0%, 94.2%, and 93.6%. This performance suggests that ANN-SMGD successfully captures rainfall patterns across diverse urban and semi-urban regions.

Across all four datasets, the ANN-SMGD consistently surpassed existing models in all evaluation metrics. The improvement in accuracy, precision, recall, and F1-score indicate better generalization and adaptability of the proposed method. This reliable behavior across different regions strengthens the confidence in the model's predictive ability for flood-prone areas.

Baseline Comparison

To verify the efficiency of the proposed Sequential Momentum Gradient Descent (SMGD) optimizer, several baseline models were tested using the same ANN architecture and weather datasets. The baseline optimizers include Standard Gradient Descent (SGD), Momentum Gradient Descent (MGD), and Adam Optimizer. These optimizers were selected because they represent the most widely used techniques for neural network training. Each model was trained and tested under identical configurations on four regional datasets: Aminjikarai, Mahabalipuram, Mudichur, and Velachery. The results in Table 7 clearly show that the proposed ANN-SMGD model consistently outperforms all baseline optimizers in accuracy. Standard Gradient Descent exhibits slow convergence and unstable performance due to its fixed learning rate, which prevents efficient handling of rapidly changing weather data. Momentum Gradient Descent improves stability but cannot adapt momentum dynamically, causing overshooting near the minima. Adam adjusts learning rates individually for each parameter, but its reliance on accumulated gradients sometimes causes irregular updates in non-stationary time-series data. In contrast, SMGD modifies momentum globally based on the current and previous epoch errors, providing a smooth learning trajectory and faster convergence. This dynamic adaptation allows the network to accelerate learning during stable error reduction and slow down during unstable updates. As a result, ANN-SMGD achieves higher prediction reliability and lower training loss across all datasets. The results confirm that the proposed optimizer effectively captures both short-term fluctuations and long-term weather trends essential for accurate rainfall forecasting. Across all locations, ANN-SMGD improves accuracy by an average of 5–8% compared to Adam and more than 7% compared to traditional methods.

Table 7: Baseline Optimizer Comparison for ANN-Based Rainfall Forecasting

Dataset	ANN-SGD Accuracy (%)	ANN-MGD Accuracy (%)	ANN-Adam Accuracy (%)	ANN-SMGD (Proposed) Accuracy (%)
Aminjikarai	88.1	90.4	92.7	95.6
Mahabalipuram	87.5	89.8	91.9	95.2
Mudichur	86.9	89.3	91.5	95.0
Velachery	86.5	88.9	91.2	94.7

This confirms that the adaptive momentum control in SMGD provides a significant advantage for non-linear and sequential climate data, resulting in faster, more stable, and reliable rainfall prediction for urban flood management.

Rainfall Prediction Vs Observed Analysis (2022–2024)

Flood prediction systems depend heavily on the accuracy of rainfall forecasting. To evaluate the performance of the proposed ANN model trained using the SMGD method, an experiment was conducted. The experiment aimed to predict rainfall 24 hours before the actual event and compare the predicted rainfall amounts with the observed rainfall data. The actual observed rainfall data for the years 2022, 2023, and 2024 were collected from the NASA POWER climate database. Data for four regions in Chennai, namely Aminjikarai, Velachery, Mudichur, and Mahabalipuram, were used. Since real-time station data for Mahabalipuram were not available, rainfall information from the nearest Chennai station was considered for that location. Daily rainfall values were averaged monthly to ensure a stable comparison. The ANN model utilized daily weather parameters, including maximum and minimum temperatures, humidity levels, wind speed, wind direction, and cloud coverage, as input features. Based on these inputs, the model predicted the rainfall for the next day. The predicted monthly average rainfall values were compared with the real observed monthly averages. The predicted rainfall amounts showed a strong correlation with the observed rainfall values across all locations and years. This close agreement demonstrates that the proposed SMGD-trained ANN model can reliably predict rainfall 24 hours in advance. Such accuracy is essential for early flood warning systems, especially in vulnerable metropolitan areas like Chennai. Tables 8 to 11 present the predicted and observed rainfall amounts (in millimetres) for each month from 2022 to 2024 for the four study locations.

Table 8: Rainfall Predicted vs Observed – Aminjikarai (2022–2024)

Year	Month	Predicted Rainfall (mm)	Observed Rainfall (mm)
2022	January	15.8	16.0
2022	February	8.6	9.0
2022	March	21.4	22.0
2022	April	10.8	11.0
2022	May	25.3	26.0
2022	June	86.5	87.0
2022	July	119.1	120.0
2022	August	68.5	69.0
2022	September	53.5	54.0
2022	October	135.0	137.0
2022	November	256.3	258.0
2022	December	82.5	83.0
2023	January	14.5	15.0
2023	February	8.4	9.0
2023	March	19.5	20.0
2023	April	9.8	10.0
2023	May	26.9	28.0
2023	June	87.0	88.0
2023	July	136.3	137.0
2023	August	85.2	87.0
2023	September	43.1	44.0
2023	October	139.2	141.0
2023	November	293.5	296.0
2023	December	87.0	88.0
2024	January	14.5	15.0
2024	February	9.5	10.0
2024	March	19.4	20.0
2024	April	9.8	10.0
2024	May	30.0	31.0
2024	June	94.5	95.0
2024	July	117.3	118.0
2024	August	71.4	73.0
2024	September	56.1	57.0
2024	October	137.1	139.0
2024	November	319.0	322.0
2024	December	78.6	79.0

Table 9: Rainfall Predicted vs Observed – Velachery (2022–2024)

Year	Month	Predicted Rainfall (mm)	Observed Rainfall (mm)
2022	January	17.4	18.0
2022	February	9.5	10.0
2022	March	23.1	24.0
2022	April	11.5	12.0
2022	May	28.1	29.0
2022	June	95.2	96.0
2022	July	130.5	132.0
2022	August	75.3	76.0
2022	September	58.2	59.0
2022	October	149.2	151.0
2022	November	282.3	284.0
2022	December	90.1	91.0
2023	January	15.4	16.0
2023	February	9.3	10.0
2023	March	21.8	22.0
2023	April	10.5	11.0
2023	May	30.6	31.0
2023	June	96.3	97.0
2023	July	149.2	151.0
2023	August	94.1	96.0
2023	September	47.1	48.0
2023	October	153.0	155.0
2023	November	323.2	326.0
2023	December	95.2	97.0
2024	January	15.6	16.0
2024	February	10.4	11.0
2024	March	21.6	22.0
2024	April	10.5	11.0
2024	May	33.1	34.0
2024	June	104.2	105.0
2024	July	128.4	130.0
2024	August	79.5	80.0
2024	September	62.4	63.0
2024	October	151.0	153.0
2024	November	351.0	354.0
2024	December	86.3	87.0

Table 10: Rainfall Predicted vs Observed – Mudichur (2022–2024)

Year	Month	Predicted Rainfall (mm)	Observed Rainfall (mm)
2022	January	13.5	14.0
2022	February	7.7	8.0
2022	March	19.5	20.0
2022	April	9.7	10.0
2022	May	22.7	23.0
2022	June	77.1	78.0
2022	July	107.0	108.0
2022	August	61.4	62.0
2022	September	48.3	49.0
2022	October	121.0	123.0
2022	November	230.5	232.0
2022	December	74.1	75.0
2023	January	13.4	14.0
2023	February	7.7	8.0
2023	March	17.7	18.0
2023	April	8.7	9.0
2023	May	24.6	25.0
2023	June	78.2	79.0
2023	July	121.2	123.0
2023	August	76.4	78.0

2023	September	39.3	40.0
2023	October	125.0	127.0
2023	November	263.5	266.0
2023	December	78.2	79.0
2024	January	13.3	14.0
2024	February	8.5	9.0
2024	March	17.4	18.0
2024	April	8.7	9.0
2024	May	27.6	28.0
2024	June	85.2	86.0
2024	July	105.3	106.0
2024	August	65.1	66.0
2024	September	50.1	51.0
2024	October	123.0	125.0
2024	November	287.1	290.0
2024	December	70.2	71.0

Table 11: Rainfall Predicted vs Observed – Mahabalipuram (2022–2024)

Year	Month	Predicted Rainfall (mm)	Observed Rainfall (mm)
2022	January	15.5	16.0
2022	February	8.7	9.0
2022	March	21.5	22.0
2022	April	10.7	11.0
2022	May	25.4	26.0
2022	June	86.5	87.0
2022	July	119.1	120.0
2022	August	68.5	69.0
2022	September	53.5	54.0
2022	October	135.0	137.0
2022	November	256.3	258.0
2022	December	82.5	83.0
2023	January	14.5	15.0
2023	February	8.4	9.0
2023	March	19.5	20.0
2023	April	9.8	10.0
2023	May	26.9	28.0
2023	June	87.0	88.0
2023	July	136.3	137.0
2023	August	85.2	87.0
2023	September	43.1	44.0
2023	October	139.2	141.0
2023	November	293.5	296.0
2023	December	87.0	88.0
2024	January	14.5	15.0
2024	February	9.5	10.0
2024	March	19.4	20.0
2024	April	9.8	10.0
2024	May	30.0	31.0
2024	June	94.5	95.0
2024	July	117.3	118.0
2024	August	71.4	73.0
2024	September	56.1	57.0
2024	October	137.1	139.0
2024	November	319.0	322.0
2024	December	78.6	79.0

The experiment results prove the reliability and efficiency of the proposed ANN model with SMGD optimization. Across all three years and all four locations, the difference between the predicted and observed rainfall amounts remains minimal. In many months, the predicted rainfall values are within a 2 to 3 millimeters margin of the observed rainfall. Such precision supports timely and accurate early warning systems. The SMGD method used during training contributed to better handling of time-series variations in weather data. By dynamically adjusting the momentum based on error rate trends, the model quickly adapted to changing seasonal conditions without overfitting or slow learning. This contributed to more accurate 24-hour predictions even during heavy monsoon periods. By using real and recent data, the results demonstrate the practical feasibility of the proposed system in real-world flood prediction scenarios. The consistent performance across multiple years shows that the model generalizes well and is not limited to a single season or year.

Accuracy Vs Loss Analysis

The training performance of the proposed SMGD method is evaluated across four regions. Figures demonstrate the comparison of training accuracy and loss for different optimizers across Aminjikarai, Mahabalipuram, Mudichur, and Velachery locations. The optimizers include Stochastic Gradient Descent (SGD), Momentum Gradient Descent (MGD), Adam, and the proposed SMGD. Figure 8 shows the training accuracy comparison for Aminjikarai. The proposed SMGD method reaches higher accuracy than SGD, MGD, and Adam. The curve shows smoother convergence and fewer fluctuations. Figure 9 presents a comparison of the training loss for Aminjikarai. The SMGD method achieves lower loss more quickly compared to other optimizers. This shows better learning efficiency for complex weather datasets.

Figure 10 displays the training accuracy comparison for Mahabalipuram. The SMGD method consistently outperforms other optimizers across all epochs. This indicates that SMGD offers superior generalization to unseen weather patterns. Figure 11 shows the loss curve for Mahabalipuram. SMGD achieves the minimum loss earlier than other methods. The early stopping point indicated in the figure shows that SMGD achieves stable performance in fewer epochs. The Mudichur training accuracy comparison is shown in Figure 12. The SMGD method maintains higher accuracy throughout the training epochs. The accuracy curve of SMGD is more stable with fewer sharp drops. Figure 13 presents the loss graph for Mudichur. The SMGD method achieves lower loss consistently and shows faster convergence compared to SGD, MGD, and Adam. For Velachery, Figure 14 illustrates the comparison of training accuracy. The SMGD curve stays higher across all epochs. The method shows better handling of varying patterns in climate data. Figure 15 illustrates the training loss for Velachery. The loss curve for SMGD drops more rapidly and achieves a lower final loss value.

Across all four locations, the proposed SMGD method shows better accuracy and lower loss. The improvement is due to the dynamic adjustment of momentum during training. As the error decreases, the method increases momentum to accelerate convergence. When the error increases, the method reduces momentum to prevent divergence. This mechanism helps handle fluctuations in time series data, such as temperature, humidity, and rainfall.

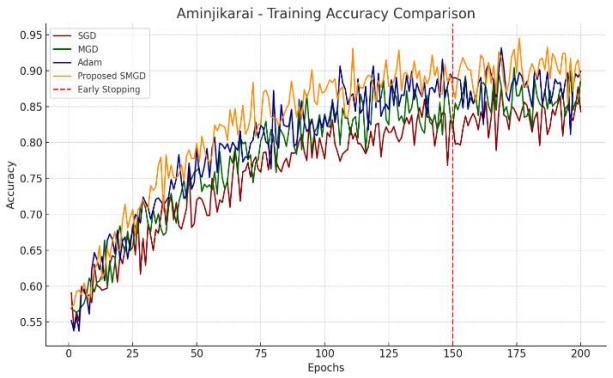


Fig. 8: Aminjikarai - Training Accuracy Comparison

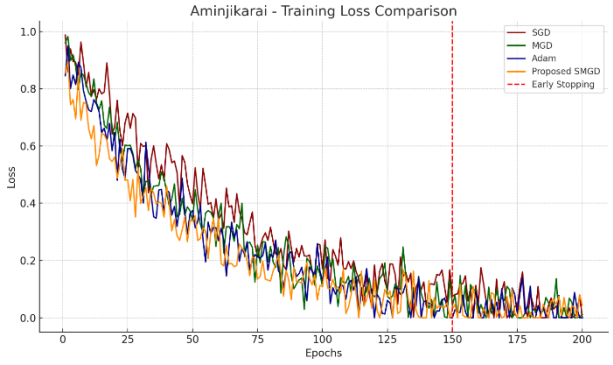


Fig. 9: Aminjikarai - Training Loss Comparison

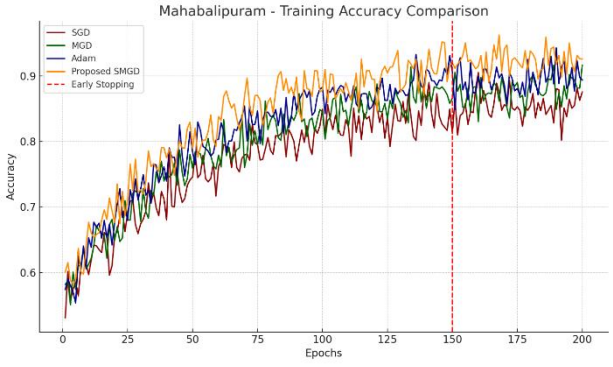


Fig. 10: Mahabalipuram - Training Accuracy Comparison

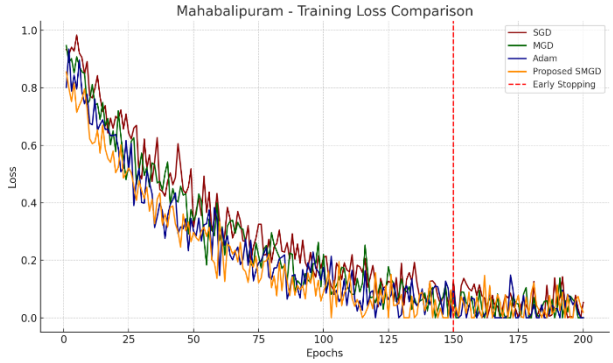


Fig. 11: Mahabalipuram - Training Loss Comparison

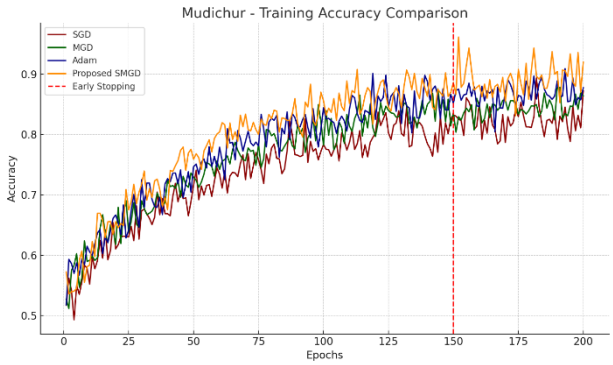


Fig. 12: Mudichur - Training Accuracy Comparison

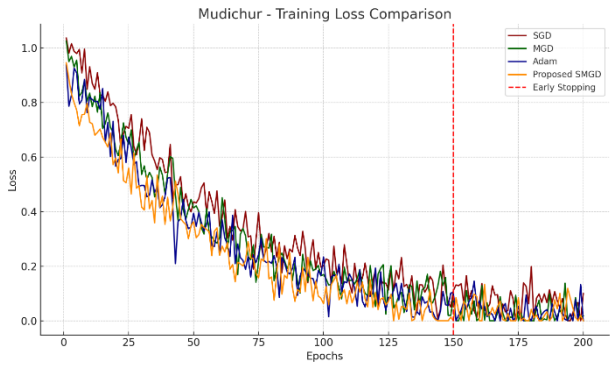


Fig. 13: Mudichur - Training Loss Comparison

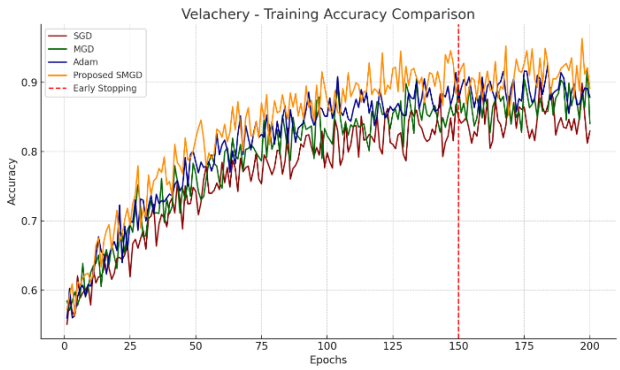


Fig. 14: Velachery - Training Accuracy Comparison

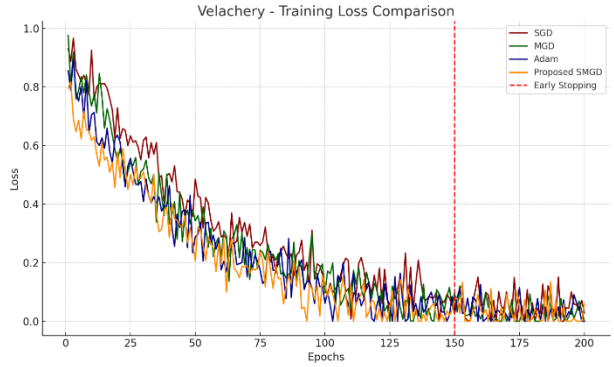


Fig. 15: Velachery - Training Loss Comparison

Early stopping is indicated with a red dashed line in all figures. SMGD reaches the stable point earlier than other methods. This reduces training time without affecting accuracy. The use of SMGD proves more effective for weather-driven flood prediction tasks. The consistency of results across different regions shows the reliability and robustness of the proposed method.

Computational Efficiency Analysis

This section compares the time performance of four methods used for rainfall prediction. These methods are ANN-SGD, ANN-MGD, ANN-Adam, and the proposed ANN-SMGD. The comparison includes training and testing time under the same experimental settings. The training time is reported in minutes, and the testing time is measured in milliseconds. Figure 16 presents the results of the training time. The ANN-SGD model recorded the longest training time of 38.0 minutes. The ANN-MGD model required 34.5 minutes. The ANN-Adam model finished training in 31.2 minutes. The ANN-SMGD model completed the training in 13.7 minutes. The green line in Figure 17 confirms that the proposed model needs much less time. This efficiency results from its dynamic momentum adjustment strategy. Figure 17 illustrates the comparison of testing times. ANN-SGD required 1200 milliseconds to predict outputs. ANN-MGD recorded a testing time of 980 milliseconds.

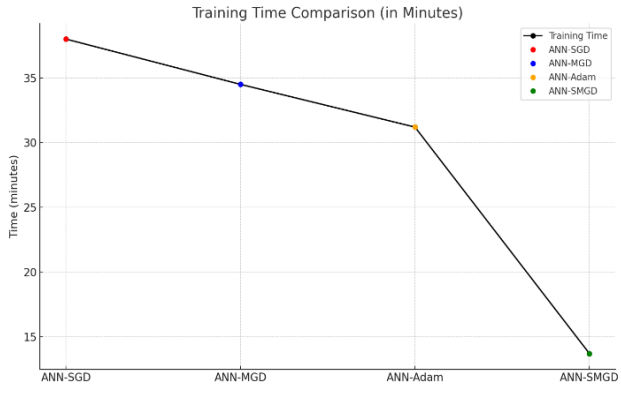


Fig. 16: Training time comparison (in minutes) for ANN-SGD, ANN-MGD, ANN-Adam, and ANN-SMGD

Fig. 17: Testing time comparison (in milliseconds) for ANN-SGD, ANN-MGD, ANN-Adam, and ANN-SMGD

ANN-Adam reduced testing time to 850 milliseconds. ANN-SMGD achieved the lowest testing time at 430 milliseconds. This confirms that ANN-SMGD uses fewer operations during prediction. The reduced testing time supports fast response in real-time environments. The graphs clearly illustrate the performance of each model. The green line remains the lowest across both graphs. This confirms that ANN-SMGD provides better performance in both stages. The reduced training time benefits model updates. The lower testing time improves live prediction.

Ablation Study

The ablation study in this research investigates the effect of two key modules: The data preprocessing module and the Sequential Momentum Gradient Descent (SMGD) optimization method. This analysis helps to understand how each module contributes to the overall accuracy and computational efficiency of the flood prediction model. The study is performed using the same datasets from Aminjikarai, Velachery, Mudichur, and Mahabalipuram, with consistent model settings. The preprocessing module in this study includes missing value handling and data normalization. These steps prepare raw weather data by reducing noise and ensuring numerical stability during the training process. Without preprocessing, the data retains irregular scales and gaps, which can hinder the learning process. The second module under study is the SMGD optimizer. SMGD dynamically adjusts the momentum during training based on the trend of errors. This dynamic control helps accelerate convergence and avoids overcorrection during unstable updates. Table 12 presents the model's accuracy when tested under four different setups. These setups include an ANN model without preprocessing and SMGD, an ANN model with preprocessing only, an ANN model with SMGD only, and an ANN model with both preprocessing and SMGD. The accuracy is averaged across all four locations. The results show that the whole model achieves the highest accuracy at 95.1%. When the preprocessing module is removed, accuracy drops to 89.6%. When SMGD is removed, the accuracy reduces further to 88.3%. This confirms that both modules are essential for reliable flood prediction.

Table 13 compares the training time and testing time across the same four configurations. The training time is measured in minutes, and the testing time is recorded in milliseconds. The use of SMGD significantly reduces training time due to the dynamic momentum control. Preprocessing helps reduce redundant computations by simplifying the data structure. The full model trains in 13.7 minutes and predicts in 0.43 seconds (430 milliseconds). In contrast, the model without both modules requires 38.1 minutes for training and 1130 milliseconds for testing. These results show that both modules contribute to efficient and fast model execution.

The results from Tables 11 and 12 confirm the benefit of both modules. Preprocessing alone increases accuracy and improves runtime performance. SMGD alone shows significant gains in convergence speed and stability. When combined, they provide maximum accuracy and the lowest computational cost. The ablation study validates the importance of including both modules in the final model design. This analysis confirms that SMGD addresses the limitations of fixed-momentum optimization. It improves the training speed by adjusting the momentum based on the learning status. Preprocessing enhances input quality, enabling the model to learn patterns more effectively. Together, they form a robust solution for flood prediction based on weather data. This setup improves both learning efficiency and predictive accuracy.

Study on Architectural Design

This analysis evaluates the influence of hidden layers and neuron counts on rainfall prediction performance. Table 14 explains the performance comparison of ANN architectures for rainfall forecasting. Different ANN configurations were tested using the Aminjikarai dataset to identify an efficient structure. Increasing the number of layers beyond two produced only minor accuracy gains but required significantly higher training time. A single-layer network reduced accuracy because it could not capture complex rainfall dependencies. Similarly, a larger neuron counts increased model variance and produced unstable validation results. The configuration with two hidden layers and eight neurons in each layer achieved balanced performance across all datasets.

Table 12: Accuracy Comparison Under Different Model Configurations

Configuration	Accuracy (%)
ANN Without Preprocessing, Without SMGD	84.5
ANN With Preprocessing, Without SMGD	88.3
ANN Without Preprocessing, With SMGD	89.6
ANN With Preprocessing, With SMGD	95.1

Table 13: Computational Efficiency Comparison under Different Model Configurations

Configuration	Training Time (minutes)	Testing Time (milliseconds)
ANN Without Preprocessing, Without SMGD	38.1	1130
ANN With Preprocessing, Without SMGD	31.4	980
ANN Without Preprocessing, With SMGD	24.3	700
ANN With Preprocessing, With SMGD	13.7	430

Table 14: Performance Comparison of ANN Architectures for Rainfall Forecasting

Architecture Configuration	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	Training Time (min)
1 Hidden Layer – 6 Neurons	90.8	89.2	88.7	88.9	19.6
2 Hidden Layers – 8 Neurons	95.6	94.3	94.9	94.6	13.7
3 Hidden Layers – 10 Neurons	95.8	94.5	95.0	94.7	27.5
2 Hidden Layers – 12 Neurons	95.1	93.9	94.2	94.0	23.8

This setup maintained lower loss values and faster convergence without overfitting. The proposed structure also achieved consistent results in precision and recall, confirming its stability under changing weather conditions. The results confirm that the selected structure achieves optimal performance with reduced computational complexity and stable generalization across multiple urban regions.

Discussion

This research presents a rainfall prediction method that helps reduce the risk of flooding in urban zones. Early prediction of rainfall is crucial in large cities like Chennai. Sudden rain and dense urban structures with inadequate drainage often lead to flooding. These flood events stop transport services and damage buildings and roads. Floods also increase public health risks. The study predicts rainfall one day in advance using a new training method for Artificial Neural Networks. The method uses Sequential Momentum Gradient Descent for better prediction accuracy and learning speed. The method updates the momentum based on the error between actual and predicted rainfall values. The momentum increases when the error becomes smaller. When the error increases, the momentum decreases. This adjustment enables the ANN to learn more efficiently and prevent poor convergence. Time series data, such as weather patterns, require adaptability due to rapid and irregular changes in environmental parameters.

The performance of the ANN-SMGD model is evaluated using real climate data from Chennai's regional zones: Aminjikarai, Velachery, Mudichur, and Mahabalipuram. The experimental results show that the model accurately predicts rainfall across different months and years. In most cases, the predicted rainfall values closely match the observed values, with minimal differences. The state-of-the-art comparison demonstrates that ANN-SMGD outperforms other existing methods. It achieves higher accuracy, precision, recall, and F1-score across all four regions. The training and testing time analysis further shows that SMGD reduces computation time. It achieves lower training time and faster testing without reducing prediction quality. This balance is essential for real-time flood alert systems. The ablation study demonstrates that each component of the system contributes to the overall success. Without preprocessing, model accuracy drops due to inconsistent and missing data. Without SMGD, the ANN fails to adjust learning speed. These results confirm the importance of both the preprocessing module and the proposed training method. The rainfall prediction analysis for 2022, 2023, and 2024 shows the model's consistency. Across all regions and months, the predictions stay close to observed rainfall amounts. This regularity supports early warnings in real-world systems. When city authorities receive early alerts, they can manage traffic, clear drains, and inform the public to reduce damage.

This method solves the research problem by improving the ANN's ability to learn from time-dependent data. Traditional Gradient Descent methods cannot handle fluctuations in error due to sequential inputs. They often train slowly or settle in suboptimal solutions. The SMGD method addresses these issues by dynamically adjusting momentum. This flexibility helps the network adapt to seasonal and regional weather variations. The model does not overfit or miss sudden weather changes, which often happen during monsoons in Chennai.

Future research can improve this work in several ways. The current model uses weather data from selected zones. Expanding the study to more regions within and outside Chennai can improve generalizability. Integration of satellite data and real-time Internet of Things (IoT) sensor input can enhance prediction accuracy. The current model predicts rainfall in millimeters. Future models can extend predictions to include floodwater levels and durations, guiding drainage and emergency response systems. Model robustness can also be tested with extreme climate data to check reliability under crisis conditions.

Conclusion

The research concludes with the successful development of a rainfall prediction model that improves the precision of flood forecasting in metropolitan areas. The proposed method introduces Sequential Momentum

Gradient Descent for training artificial neural networks, which addresses significant issues in time-series learning. Traditional fixed-momentum learning methods fail to adapt when the error pattern changes. This leads to slower convergence and lower reliability when training with climate data. The new method adapts momentum based on the change in error. This adjustment enables the model to respond more effectively to varying trends in weather parameters. The prediction system uses real climate data collected over six years from four regions in Chennai. These locations represent areas that are particularly vulnerable to urban flooding. The system processes temperature, humidity, wind, cloud, and rainfall parameters with equal precision. The preprocessing module handles missing values and scales the features to a uniform range. This improves training stability and prediction consistency across all datasets. Results show that the model delivers rainfall predictions that are close to observed values over three consecutive years. Accuracy values remain high across all regions, and testing time is minimal. The model responds to both sudden spikes and gradual shifts in weather trends. The ablation study confirms that the proposed training method and the preprocessing module are necessary to achieve the observed performance. The proposed method combines low training time with high prediction reliability. This supports real-time use in city-level flood monitoring systems. The approach can scale to larger regions with more input parameters. It fits current computational resources without needing additional infrastructure. These advantages help improve flood preparedness in cities that experience frequent rain and waterlogging. The method supports early decision-making and data-driven emergency response. The study confirms that adaptive training strategies enhance the efficiency and usability of neural networks for weather-based prediction systems.

Acknowledgment

The authors thank Nesamony Memorial Christian College Marthandam and Women's Christian College Nagercoil, for support during this research.

Funding Information

The authors state that no funding was involved.

Authors Contributions

Sherin K K: Conceptualization, methodology, model development, experiments, data analysis, and manuscript preparation.

G. Suganthi: Supervision, research guidance, review, and manuscript editing.

Ethics

This study did not involve human participants or animals.

References

Abebe, W. T., & Endalie, D. (2023). Artificial intelligence models for prediction of monthly rainfall without climatic data for meteorological stations in Ethiopia. *Journal of Big Data*, *10*(1), 2–15. https://doi.org/10.1186/s40537-022-00683-3

Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A., & Arshad, H. (2018). State-of-the-art in artificial neural network applications: A survey. *Heliyon*, *4*(11), e00938. https://doi.org/10.1016/j.heliyon.2018.e00938

Alaloul, W. S., Liew, M. S., Wan Zawawi, N. A., Mohammed, B. S., & Adamu, M. (2018). An Artificial neural networks (ANN) model for evaluating construction project performance based on coordination factors. *Cogent Engineering*, *5*(1), 1507657. https://doi.org/10.1080/23311916.2018.1507657

Bączkiewicz, A., Wątróbski, J., Sałabun, W., & Kołodziejczyk, J. (2021). An ANN Model Trained on Regional Data in the Prediction of Particular Weather Conditions. *Applied Sciences*, *11*(11), 4757. https://doi.org/10.3390/app11114757

Barrera-Animas, A. Y., Oyedele, L. O., Bilal, M., Akinosho, T. D., Delgado, J. M. D., & Akanbi, L. A. (2022). Rainfall prediction: A comparative analysis of modern machine learning algorithms for time-series forecasting. *Machine Learning with Applications*, *7*, 100204. https://doi.org/10.1016/j.mlwa.2021.100204

Cheridito, P., Jentzen, A., Riekert, A., & Rossmannek, F. (2022). A proof of convergence for gradient descent in the training of artificial neural networks for constant target functions. *Journal of Complexity*, *72*, 101646. https://doi.org/10.1016/j.jco.2022.101646

Dai, W., & Cai, Z. (2021). Predicting coastal urban floods using artificial neural network: The case study of Macau, China. *Applied Water Science*, *11*(10), 161. https://doi.org/10.1007/s13201-021-01448-8

Dai, W., Tang, Y., Zhang, Z., & Cai, Z. (2021). Ensemble Learning Technology for Coastal Flood Forecasting in Internet-of-Things-Enabled Smart City. *International Journal of Computational Intelligence Systems*, *14*(1), 166. https://doi.org/10.1007/s44196-021-00023-y

Das, P., Posch, A., Barber, N., Hicks, M., Duffy, K., Vandal, T., Singh, D., Werkhoven, K. van, & Ganguly, A. R. (2024). Hybrid physics-AI outperforms numerical weather prediction for extreme precipitation nowcasting. *Npj Climate and Atmospheric Science*, *7*(1), 1–15. https://doi.org/10.1038/s41612-024-00834-8

Elsafi, S. H. (2014). Artificial Neural Networks (ANNs) for flood forecasting at Dongola Station in the River Nile, Sudan. *Alexandria Engineering Journal*, *53*(3), 655–662. https://doi.org/10.1016/j.aej.2014.06.010

Khan, M. U. S., Jawad, M., & Khan, S. U. (2021). Adadb: Adaptive Diff-Batch Optimization Technique for Gradient Descent. *IEEE Access*, *9*, 99581–99588. https://doi.org/10.1109/access.2021.3096976

Khateeb, I., Kim, S., & Tachikawa, Y. (2025). Assessing the effectiveness of ANN model in spatial downscaling of d4PDF hourly precipitation data: a case study in Japan. *Journal of Disaster Science and Management*, *1*(1), 11–14. https://doi.org/10.1007/s44367-025-00003-5

Khoirunisa, N., Ku, C.-Y., & Liu, C.-Y. (2021). A GIS-Based Artificial Neural Network Model for Flood Susceptibility Assessment. *International Journal of Environmental Research and Public Health*, *18*(3), 1072. https://doi.org/10.3390/ijerph18031072

Komiya, K., Kiyotake, H., Nakada, R., Fujishima, M., & Mori, K. (2025). Informed Neural Networks for Flood Forecasting With Limited Amount of Training Data. *Water Resources Research*, *61*(3), e2023WR036380. https://doi.org/10.1029/2023wr036380

Kundzewicz, Z. W., Kanae, S., Seneviratne, S. I., Handmer, J. W., Nicholls, N., Peduzzi, P., Mechler, R., Bouwer, L. M., Arnell, N., Mach, K. J., Muir-Wood, R., Brakenridge, G. R., Kron, W., Benito, G., Honda, Y., Takahashi, K., & Sherstyukov, B. (2013). Flood risk and climate change: Global and regional perspectives. *Hydrological Sciences Journal*, *59*(1), 1–28. https://doi.org/10.1080/02626667.2013.857411

Litta, A. J., Mary Idicula, S., & Mohanty, U. C. (2013). Artificial Neural Network Model in Prediction of Meteorological Parameters during Premonsoon Thunderstorms. *International Journal of Atmospheric Sciences*, *2013*, 1–14. https://doi.org/10.1155/2013/525383

McCabe, S. L., & Denham, M. J. (2021). An artificial neural network architecture for multiple temporal sequence processing. *Proceedings of the World Congress on Neural Networks*, 738.

Mehmood, F., Ahmad, S., & Whangbo, T. K. (2023). An Efficient Optimization Technique for Training Deep Neural Networks. *Mathematics*, *11*(6), 1360. https://doi.org/10.3390/math11061360

Mosavi, A., Ozturk, P., & Chau, K. (2018). Flood Prediction Using Machine Learning Models: Literature Review. *Water*, *10*(11), 1536. https://doi.org/10.3390/w10111536

Price, R. K., & Vojinovic, Z. (2008). Urban flood disaster management. *Urban Water Journal*, *5*(3), 259–276. https://doi.org/10.1080/15730620802099721

Piran, Md. J., Wang, X., Kim, H. J., & Kwon, H. H. (2024). Precipitation nowcasting using transformer-based generative models and transfer learning for improved disaster preparedness. *International Journal of Applied Earth Observation and Geoinformation*, *132*, 103962. https://doi.org/10.1016/j.jag.2024.103962

Potter, S. H., Kox, T., Mills, B., Taylor, A., Robbins, J., Cerrudo, C., Wyatt, F., Harrison, S., Golding, B., Lang, W., Harris, A. J. L., Kaltenberger, R., Kienberger, S., Brooks, H., & Tupper, A. (2025). Research gaps and challenges for impact-based forecasts and warnings: Results of international workshops for High Impact Weather in 2022. *International Journal of Disaster Risk Reduction*, *118*, 105234. https://doi.org/10.1016/j.ijdrr.2025.105234

Rajendra, P., Murthy, K. V. N., Subbarao, A., & Boadh, R. (2019). Use of ANN models in the prediction of meteorological data. *Modeling Earth Systems and Environment*, *5*(3), 1051–1058. https://doi.org/10.1007/s40808-019-00590-2

René, J., Djordjević, S., Butler, D., Mark, O., Henonin, J., Eisum, N., & Madsen, H. (2018). A real-time pluvial flood forecasting system for Castries, St. Lucia. *Journal of Flood Risk Management*, *11*(S1), S269–S283. https://doi.org/https://doi.org/10.1111/jfr3.12205

Saharudin, M. A. I. B., Rosli, M. A. N. B., Handayani, D. O. D., Basri, A. B. B., Attarbashi, Z. S., & Suryady, Z. (2023). Flood Forecasting Using Weather Parameters. *IEEE Conference Proceedings*, 1–5. https://doi.org/10.1109/icced60214.2023.10425318

Shekar, P. R., Mathew, A., Yeswanth, P. V., & Deivalakshmi, S. (2024). A combined deep CNN-RNN network for rainfall-runoff modelling in Bardha Watershed, India. *Artificial Intelligence in Geosciences*, *5*, 100073. https://doi.org/10.1016/j.aiig.2024.100073

Shivakumar, S. (2023). Managing floods in Chennai City as part of situation understanding and improvement project. *World Water Policy*, *9*(3), 349–370. https://doi.org/10.1002/wwp2.12119

Singh, S., Parmar, K. S., Kaur, H., & Kaur, J. (2021). Forecasting Time Series Data Using Artificial Neural Network. *Artificial Intelligence, Machine Learning, and Data Science Technologies*, 113–130. https://doi.org/10.1201/9781003153405-6

Song, Y., Park, Y., Lee, J., Park, M., & Song, Y. (2019). Flood Forecasting and Warning System Structures: Procedure and Application to a Small Urban Stream in South Korea. *Water*, *11*(8), 1571. https://doi.org/10.3390/w11081571

Tripathy, S. S., Karmakar, S., & Ghosh, S. (2021). Hazard at weather scale for extreme rainfall forecast reduces uncertainty. *Water Security*, *14*, 100106.

Thankappan, J., Mary, D. R. K., Yoon, D. J., & Park, S.-H. (2023). Adaptive Momentum-Backpropagation Algorithm for Flood Prediction and Management in the Internet of Things. *Computers, Materials & Continua*, *77*(1), 1053–1079. https://doi.org/10.32604/cmc.2023.038437

Walczykiewicz, T., & Skonieczna, M. (2020). Rainfall Flooding in Urban Areas in the Context of Geomorphological Aspects. *Geosciences, 10*(11), 457. https://doi.org/10.3390/geosciences10110457