Spatiotemporal Rainfall Forecasting Using Deep Learning Under Computational Constraints: An IMERG-Based Study

Muhammad Izaaz Hazmii Suhaimi, Fazlina Ahmat Ruslan, Noorfadzli Abd Razak, Mohd Azri Abdul Aziz and Juliana Johari*

***Abstract*— Accurate short-term rainfall forecasting is essential for flood risk management, especially in tropical regions prone to localized convective storms. This study evaluates the performance of three deep learning architectures, ConvLSTM, 3D-CNN, and Temporal Convolutional Network (TCN), for 6-hour rainfall forecasting using satellite-based IMERG precipitation data. All models were trained, validated and tested on a consistent spatial-temporal dataset and compared based on standard metrics including RMSE, MAE, training stability, and spatial prediction quality. Quantitative results show that ConvLSTM achieved the lowest RMSE (0.3932 mm) and MAE (0.1148 mm), outperforming 3D-CNN and TCN across all evaluation criteria. Visual inspections of cumulative rainfall forecasts further confirm ConvLSTM's ability to preserve convective structure and suppress background noise, while TCN struggled with spatial generalization and temporal consistency. Spatial RMSE maps averaged across the forecast horizon revealed that ConvLSTM maintained low error regions throughout the domain. These findings underscore the effectiveness of spatiotemporal modelling in rainfall forecasting and indicate that ConvLSTM shows promising performance for operational early warning systems, though further validation is required.**

***Index Terms*— Computational constraints, deep learning, IMERG, rainfall forecasting, spatiotemporal analysis.**

I. Introduction

Short-term precipitation forecasting (nowcasting with lead times up to 6 hours) is vital for weather-dependent decision-making and early flood warnings [1], [2]. In tropical regions such as Peninsular Malaysia, intense convective rainfall can trigger devastating floods. For example, in December 2021, unprecedented rainfall inundated parts of the west coast under nearly four meters of water, turning roads into rivers [2]. Timely and accurate rainfall forecasts enable authorities to issue warnings and manage water resources, potentially saving lives and reducing economic losses.

Traditionally, Numerical Weather Prediction (NWP) models provide short-range forecasts but at coarse resolution and with high computational cost, running on supercomputers [3]. These physics-based models often struggle with short-term rainfall prediction due to initialization uncertainties and the chaotic nature of convective processes [1]. In contrast, data-driven approaches using deep learning have shown promise in capturing the complex spatiotemporal patterns of rainfall from historical data. For instance, Convolutional LSTM (ConvLSTM) networks can learn the evolution of 2D precipitation fields over time and have been successfully applied to nowcasting radar and satellite rainfall [1], [4]. Similarly, 3D convolutional neural networks and other CNN-based models have been explored for video-like precipitation sequence prediction, sometimes outperforming traditional extrapolation methods [5]. Another approach, Temporal Convolutional Networks (TCN), uses 1D dilated causal convolutions for sequence modeling and has demonstrated advantages in capturing long-term dependencies with stable gradients [6]. TCNs have yielded competitive results in weather and hydrological forecasts while being more lightweight and parallelizable than recurrent models [6], [7].

Despite these advances, deploying deep learning models in practice especially in developing regions or agencies with limited computing infrastructure remains challenging. High-resolution data and large neural networks demand substantial memory and processing power. There is a need for forecasting solutions that balance accuracy with computational efficiency, enabling wider adoption. Recent studies highlight the importance of lightweight models that can run on standard PCs or minimal hardware without sacrificing too much accuracy [7], [8]. In this work, we focus on making deep learning-based rainfall forecasting more accessible under computational constraints. We utilize NASA's IMERG (Integrated Multi-satellitE Retrievals for GPM) satellite precipitation dataset, which provides global rainfall estimates every 30 minutes [9] and tailor our models and data processing to fit on a single mid-range GPU.

The primary objective of this study is to evaluate and compare ConvLSTM, 3D-CNN, and TCN models for a six-hour ahead rainfall forecasting task, focusing on their performance trade-offs in terms of accuracy and efficiency

This manuscript is submitted on July 26, 2025, revised on October 2, 2025, accepted on October 23, 2025, and published on April 30, 2026.

Muhammad Izaaz Hazmii Suhaimi, Juliana Johari, Fazlina Ahmat Ruslan, Noorfadzli Abd Razak and Mohd Azri Abdul Aziz are with the Faculty of Electrical Engineering, Universiti Teknologi MARA, 40450 Shah Alam, Selangor, Malaysia.

*Corresponding author
Email address: julia893@uitm.edu.my

1985-5389/© 2026 The Authors. Published by UiTM Press. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

when operated on limited hardware. Peninsular Malaysia serves as the study region, leveraging its rich record of tropical rainfall events derived from the IMERG dataset.

This research provides several key contributions. First, it offers a comparative analysis of three deep learning architectures (ConvLSTM, 3D-CNN, and TCN) applied to spatiotemporal rainfall forecasting using the same IMERG dataset. This side-by-side comparison under identical experimental conditions highlights the distinct strengths and limitations of each approach for short-term precipitation prediction.

Second, the study addresses hardware-constrained model design, detailing strategies to reduce computational load. These include defining a limited spatial domain (bounding box over Peninsular Malaysia) and applying spatial resolution downsampling. Neural network models are optimized in size to fit within the memory limits of an 8GB GPU, demonstrating that meaningful forecasts can be achieved without reliance on expensive, high-performance computing resources.

Finally, by emphasizing models that can run on modest hardware, this work supports practical integration into electrical and electronic systems such as distributed sensor networks, edge-computing devices, and telecommunication-based flood early warning systems. This connection ensures that advances in rainfall forecasting can be directly embedded into real-world monitoring and decision-making infrastructures.

Performance evaluation is conducted using Root Mean Square Error (RMSE, mm/hr) and Mean Absolute Error (MAE, mm/hr), computed per 30-min forecast frame. For cumulative evaluations (e.g., 6-hour totals), metrics are reported in millimetres (mm). The reported benchmark values (e.g., RMSE in mm/hr) serve as baseline references for future research and improvements. These results are further contextualized in terms of model complexity, training duration, and the inherent trade-off between accuracy and computational cost.

Lastly, the study underscores the broader implications for real-world deployment. Demonstrating that deep learning models can operate on modest hardware with acceptable predictive performance supports the broader goal of adopting AI-based forecasting methods in resource-constrained environments. Such an approach holds potential benefits for local meteorological agencies and disaster management authorities, particularly in regions with limited access to advanced computing infrastructure.

The following sections review related researches (Section 2), describe the study data, model architectures, and experimental setup (Section 3), present the results and discussion of model performance (Section 4), and conclude with reflections on the feasibility and future directions of low-resource operational forecasting (Section 5).

II. Literature Review

A. Deep Learning for Rainfall Nowcasting

Recent years have seen rapid growth in deep learning applications for precipitation nowcasting (0–6h forecasts). ConvLSTM, introduced by [10] has become a cornerstone for modelling spatiotemporal weather data. In a ConvLSTM, convolutional layers within LSTM cells enable the network to capture both spatial structures and temporal dynamics of evolving rainfall fields. Numerous studies after 2019 have validated ConvLSTM's effectiveness for short-term rainfall prediction. For example, [4] demonstrated a ConvLSTM-based model that significantly improved convective rainfall nowcasts when incorporating satellite cloud imagery and wind field data. The ConvLSTM model achieved higher Critical Success Index scores than traditional optical-flow extrapolation, highlighting the benefit of deep spatiotemporal memory in predicting localized heavy rain. Other researchers have proposed enhancements like encoder–decoder ConvLSTM networks with attention mechanisms or hybrid models (e.g., combining ConvLSTM with U-Net architectures) to further refine precipitation nowcasts [1], [11]. Overall, ConvLSTM-based approaches often set the benchmark for nowcasting accuracy, especially in tasks involving gridded meteorological data such as weather radar echoes or satellite precipitation estimates.

B. CNN and 3D-CNN Approaches

Convolutional neural networks (CNNs) offer another route for modelling spatiotemporal data by treating sequences of images as a "video" volume. A 3D-CNN applies convolution filters in both spatial dimensions and the time dimension simultaneously, extracting features across space and time in one unified framework [11], [12]. This approach has been explored for very short-term radar rainfall predictions. [5] developed a 3D-CNN model to nowcast convective storms using 3D weather radar data, and found it outperformed a traditional 3D extrapolation scheme (3DNOW) and persistence in terms of threat scores at 0–10 min leads. The 3D-CNN captured the growth and movement of rain cells better at immediate lead times, though performance slightly degraded at longer leads (e.g., 10 min) compared to the simpler advection model. In the context of hourly satellite data, 3D-CNNs can learn precipitation evolution over several hours by leveraging local spatiotemporal correlations. However, pure CNN-based models may struggle with longer-term dependencies or extreme events unless given very deep networks or large training data. To mitigate this, some studies use CNNs in combination with recurrent layers or as part of multi-stream ensembles [11], [13]. Still, 3D-CNNs remain appealing due to their straightforward fully-convolutional structure and parallel computation, potentially offering faster inference than recurrent models. One must consider that 3D convolutions can be memory-intensive; thus, model architecture must be tuned carefully for hardware limits.

C. Temporal Convolutional Networks (TCN):

TCNs are a class of models originally developed for sequence modelling tasks, which have recently been applied in weather and hydrological forecasting. A TCN uses 1D convolution along the time axis with dilated filters and causal padding, enabling it to capture long-range temporal patterns without recurrence. The advantages of TCNs include a large receptive field, stable gradients, and inherent parallelism in

TABLE 1. IMERG SPATIAL RESOLUTIONS AND ASSOCIATED USE CASES

Resolution (H×W)	Approx. Grid Size	Spatial Resolution	Approx. Grid Area (km²)	Type	Use Case
1800 ×3600	~11 × 11 km	0.1° × 0.1°	~123 km²	Native IMERG V07 grid	Highly detailed city-wide predictions
900 × 1800	~22 × 22 km	0.2° × 0.2°	~484 km²	Downsampled (this study)	Balanced detail and efficiency for districts
450 × 900	~44 × 44 km	0.4° × 0.4°	~1,936 km²	Further downsampled (hypothetical)	Broad regional trends or large-scale flooding

Based on downscaling approaches and hydrological modelling use cases from [16], [17]

TABLE 2. DATASET SPLIT WITH PERIODS AND APPROXIMATE SAMPLE COUNTS.

Dataset	Period	Samples (approx.)	Purpose	Dataset
Training	2014–2021	~36,000	Model fitting	Training
Validation	2022	~4,500	Early stopping & hyperparameter tuning	Validation
Test	2023	~4,500	Independent generalization evaluation	Test

training [6]. These properties make TCNs an attractive alternative to RNN-based models like LSTMs, especially when long sequences are involved. [7] applied a TCN to local weather station time-series data (temperature, humidity, etc.) and reported better forecast accuracy than a comparable LSTM, while also noting the model could run efficiently on a standard computer. In precipitation forecasting, TCNs have been used in a variety of ways. Some works treat multi-grid inputs as multiple channels in a TCN, essentially flattening the spatial dimension into a vector per time step. This allows the TCN to learn from all locations' histories at once, but can introduce a very high-dimensional input. Others have integrated TCNs with spatial feature extractors – for instance, using a preliminary CNN to encode each radar or satellite image into a feature vector, and then feeding the sequence of vectors into a TCN. Other studies on rainfall-runoff modelling have employed deep temporal convolution networks combined with attention or hybrid components to forecast rainfall or streamflow, showing that TCN-based models can match or exceed LSTM performance in certain cases [7], [14]. A key consideration is that TCNs, being fully convolutional, require careful design of dilation factors and kernel sizes to effectively cover the needed forecast horizon. Additionally, while TCNs excel at temporal patterns, they do not intrinsically model spatial relationships – hence our study evaluates how a TCN handles gridded rainfall data versus the explicitly spatiotemporal ConvLSTM and 3D-CNN.

In summary, prior research provides evidence that ConvLSTM, 3D-CNN, and TCN each hold promises for precipitation forecasting, but there is a knowledge gap in direct comparisons of these architectures on a common problem under resource constraints. Our work extends the literature by benchmarking these three approaches on the same IMERG satellite rainfall dataset, and by addressing the practical challenge of operational deployment on limited hardware.

III. RESEARCH METHODOLOGY

A. Study Area and Data

Our case study focuses on Peninsular Malaysia (West Malaysia), a region characterized by a tropical monsoon climate with abundant rainfall. We defined a spatial bounding box roughly covering latitudes 1°N–7°N and longitudes 99°E–105°E, which encapsulates the Peninsular Malaysian landmass and surrounding waters. All data were extracted for this region from the IMERG dataset. IMERG (Integrated Multi-satellitE Retrievals for GPM) is a satellite-based precipitation product from NASA's Global Precipitation Measurement mission, providing 30 minutes estimates of rainfall on a ~0.1° latitude/longitude grid [9]. In this study, we used IMERG 30 minutes accumulated rainfall from recent years. The use of satellite data is motivated by its complete spatial coverage, which is valuable in regions with sparse ground radar or rain gauge networks.

To prepare the data for model training under hardware limitations, we applied several preprocessing steps:

1) Spatial Cropping and Downsampling:

Using the defined bounding box (98.000°E–107.000°E, 0.000°N–9.000°N), we extracted a regional rainfall time series from the global IMERG grids. The original IMERG dataset is provided on a global grid of 1800 × 3600 cells at 0.1° × 0.1° spatial resolution, which corresponds to an approximate grid spacing of ~11 × 11 km near the equator (cell area ~123 km²) [15]. This native resolution, while adequate for global and regional studies, can be computationally expensive for localized modelling.

To optimize memory usage and improve computational efficiency, we downsampled by a factor of 2 using bilinear interpolation, producing tensors of 900 × 1800 cells at 0.2° × 0.2° resolution (~22 × 22 km grid spacing, ~484 km² per cell). The interpolation preserved dominant rainfall patterns while

smoothing fine-scale noise, making the data more suitable for district-scale hydrological modelling. The final cropped domain over Peninsular Malaysia typically resulted in ~30 × 30 effective grid cells.

The resolution tiers and their respective use cases are summarized in Table 1. The 0.1° (native) and 0.2° (downsampled) tiers are consistent with scales commonly used in IMERG-based flood and hydrological studies [16], [17].A further 0.4° hypothetical resolution (~44 × 44 km grid spacing) is also shown, representing a coarse option for regional flood-trend or climate-scale applications.

2) Temporal Resolution and Sequencing

The Integrated Multi-satellitE Retrievals for GPM (IMERG) dataset provides precipitation estimates at a native temporal resolution of 30 minutes [15]. In this study, we retain this resolution to capture fine-scale temporal variability essential for short-term rainfall forecasting.

Each training sample comprises a sequence of 12 consecutive 30-minute rainfall frames (i.e., 6 hours of historical data) as input and 12 subsequent 30-minute frames (i.e., 6 hours of future data) as the prediction target. This many-to-many forecasting structure enables the model to learn complex spatiotemporal patterns across a 6-hour context window and generate forecasts extending 6 hours into the future, with outputs available at 30 minutes intervals.

Formally, the task is defined as follows: given rainfall observations at times t−5.5 h, t−5 h, ..., t, the model predicts rainfall at times t+0.5 h, t+1 h, ..., t+6 h. This configuration was chosen based on both empirical evaluation and prior literature, which suggest that a 6-hour input window offers an effective trade-off between capturing temporal dependencies and maintaining computational efficiency. Preliminary experiments confirmed that extending the input sequence beyond 12 frames provided negligible gains in accuracy, while shorter sequences degraded forecast performance.

3) Normalization

No explicit normalization or scaling was applied to the rainfall input or output data. The model was trained directly on raw rainfall values (in mm/hr), including extreme events. This decision preserved the physical interpretability of the model outputs and allowed the loss function to focus more accurately on high-intensity rainfall patterns critical for flood forecasting.

While normalization is common in deep learning to stabilize training, in our case scaling rainfall values could suppress the relative importance of high-intensity extremes. By training on raw mm/hr values, the model directly optimizes errors in physical rainfall units, ensuring that heavy-rainfall events, which drive flood hazards are prioritized in the loss function. Although this may slow convergence slightly, it improves interpretability and maintains a physically consistent error scale.

4) Dataset Split

To ensure temporal integrity and avoid data leakage, the IMERG rainfall dataset was split chronologically into training, validation, and independent test sets. The training set spanned January 1, 2014 – December 31, 2021, while the validation set covered January 1 – December 31, 2022 and was used for early stopping and hyperparameter tuning. An independent test set from January 1 – December 31, 2023 was reserved exclusively for final evaluation of generalization. These temporal split mimics real-world forecasting scenarios, where models trained on past years are applied to predict unseen future events.

Table X summarizes the dataset division with corresponding sample counts. The test set was never used during model training or validation and was only applied once after model development to provide an unbiased estimate of performance.

B. Model Architectures

We implemented three deep learning model architectures ConvLSTM, 3D-CNN, and TCN each tailored for the 6-hour ahead rainfall forecasting task.

1) ConvLSTM Architecture:

The ConvLSTM model is designed to process sequences of 2D rainfall maps by integrating convolutional operations into the LSTM's gating mechanisms. The input consists of a sequence of 30 minutes rainfall images with spatial dimensions H×W (approximately 30×30 in our downsampled grid). At each timestep, a ConvLSTM layer receives the current image along with the previous hidden and cell states—both of which preserve the same spatial dimensions—and produces updated states.

Our implementation adopts an encoder-predictor structure. Specifically, three ConvLSTM layers (each with 16 filters and a 5×5 kernel) are stacked to form the encoder, which sequentially processes twelve input frames (6 h at 30-min). A final 3D convolutional layer (3D-CNN) then generates rainfall predictions for n future hours based on the final hidden state. All hidden and cell states are zero-initialized at the beginning of each sequence.

We do not apply explicit downsampling operations such as pooling or strided convolutions within the ConvLSTM layers, thereby preserving spatial resolution throughout the model. Instead, the receptive field is expanded by stacking multiple ConvLSTM layers, allowing the model to learn both local and global spatial patterns. Although the input rainfall maps are downsampled externally during preprocessing to reduce computational load, the model itself maintains this resolution, ensuring spatial continuity is retained across timesteps. This design enables ConvLSTM to effectively track the movement and transformation of rainfall cells over time through the propagation of hidden states [4].

One challenge associated with ConvLSTM is its memory consumption: each gate contains convolutional weights and maintains a full hidden state for every spatial location. To mitigate this, we limit the number of filters, keeping the parameter count manageable (~0.5 million) and ensuring the model fits within GPU memory constraints. We also applied gradient checkpointing during training to reduce memory usage. Despite its computational cost, ConvLSTM is well-suited to leverage spatial continuity in rainfall, offering the potential for improved prediction accuracy.

2) 3D-CNN Architecture

The 3D Convolutional Neural Network (3D-CNN) model treats each input as a spatiotemporal volume with shape

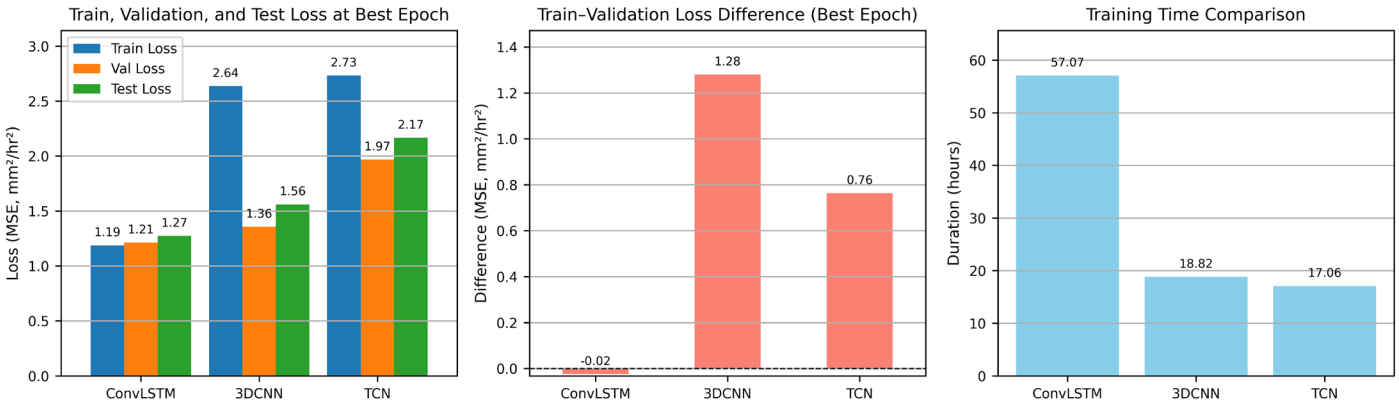

Fig. 1. Training and evaluation comparison results for all models. (a) Training, validation, and test loss at the best epoch (MSE, mm²/hr²). (b) Train–validation loss difference at the best epoch (MSE, mm²/hr²). (c) Total training time (hours). Subfigures (a)–(c) are shown from left to right.

(Channels, Time, Height, Width) = (4, 12, H, W), where each sample contains 12 consecutive 30-minute rainfall maps along with static features such as elevation, latitude, and longitude. These form a video-like tensor that is processed by a series of 3D convolutional layers. Each 3D convolution uses a kernel size of 3×3×3 (in time × height × width), enabling the model to learn spatial and temporal dependencies simultaneously [11].

Our implementation comprises three sequential 3D convolutional blocks. Each block consists of a 3D convolution layer with 32 filters, followed by batch normalization, ReLU activation, and 3D dropout (with a rate of 0.3) for regularization. No pooling or strided convolution is applied, ensuring that the original spatiotemporal resolution is preserved throughout the network. The receptive field is instead expanded through the depth of the stacked convolutional layers.

To enhance spatial discrimination, a spatial attention mechanism is introduced using a 1×1×1 convolution followed by a sigmoid activation. The resulting attention map modulates the feature activations to emphasize important spatial regions. This attention-weighted representation is then passed through a final 1×1×1 convolutional layer that produces n predicted rainfall maps, where n = 12 corresponds to the next 6 hours of 30-minute forecasts. The output tensor has the shape (Batch, n, Height, Width).

Unlike ConvLSTM, which explicitly models temporal dependencies through recurrent hidden states, the 3D-CNN captures temporal dynamics implicitly via spatiotemporal convolutions. This structure enables more parallel computation and faster training. Although it may be less effective than ConvLSTM in capturing long-range temporal patterns, the 3D-CNN offers competitive performance for near-term rainfall forecasting and is well-suited for deployment on consumer-grade GPUs, with approximately 0.3 million parameters.

3) TCN Architecture

The Temporal Convolutional Network (TCN) model is attractive for its computational efficiency, parallelism, and suitability for sequence modelling without relying on recurrence [6]. In our implementation, the TCN processes sequences of rainfall data by treating each spatial location independently as a separate time series. Each input sample consists of 12 consecutive 30-minute rainfall maps and accompanying static features (e.g., DEM, latitude, longitude), resulting in a tensor of shape (Batch, Time, Channels, Height, Width).

To introduce spatial awareness while maintaining efficiency, we first apply a shallow 2D convolution (with ReLU activation) to each frame independently using a shared spatial feature extractor. This results in a transformed feature representation that retains the original spatial resolution. The temporal evolution at each spatial grid point is then modelled independently using a lightweight 1D convolutional network. Specifically, we apply two stacked 1D convolutional layers with a kernel size of 3, preserving sequence length via padding. The final output comprises n = 12 predicted rainfall maps corresponding to the next 6 hours, each at 30-minute intervals. The output tensor has shape (Batch, 12, Height, Width).

Unlike ConvLSTM and 3D-CNN models, which jointly capture spatial and temporal dependencies through recurrence or 3D kernels, respectively, the TCN focuses purely on temporal dynamics at the pixel level. This design enables high computational efficiency and scalability, making it well-suited for real-time applications, despite its limited capacity to model fine-grained spatial interactions. With approximately 0.1 million parameters, the model remains lightweight and easily deployable on resource-constrained hardware.

In our implementation, the TCN model processes each spatial location independently by modelling its temporal sequence using stacked 1D dilated convolutions. To introduce some spatial awareness, a shared 2D convolutional layer is applied initially to extract local spatial features across the input frames. However, during the forecasting stage, the model primarily focuses on learning temporal dependencies at each pixel and does not explicitly account for spatial interactions between grid locations.

To provide a unified overview of the architectural configurations and training hyperparameters, Table 3 summarizes the main components of each model. This includes layer types, kernel sizes, hidden channels, parameter counts, as

TABLE 3. SUMMARY OF MODEL ARCHITECTURES AND TRAINING HYPERPARAMETERS

Model	Architecture Summary	Key Hyperparameters	Parameters (approx.)
ConvLSTM	3 stacked ConvLSTM layers (16 filters each, 5×5 kernel), followed by 3D Conv output	Batch size = 1, Optimizer = Adam (lr=0.001, weight decay=1e−5), Dropout = 0.3, Epochs = 100, Early stopping patience = 10, Random seed = 42	~0.5M
3D-CNN	3 convolutional blocks (32 filters each, 3×3×3 kernel) with BatchNorm + ReLU + 3D Dropout (0.3), followed by spatial attention (1×1×1 conv)		~0.3M
TCN	Shared 2D Conv (3×3, ReLU) for each frame → 2 stacked 1D temporal convolutions (kernel size=3)		~0.1M

well as training setup such as optimizer, learning rate, batch size, number of epochs, and fixed random seeds. Such detail enhances reproducibility and allows readers to directly compare the computational complexity and design choices across models.

In summary, whereas ConvLSTM and 3D-CNN architectures jointly encode spatiotemporal patterns through recurrent units and 3D kernels, respectively, the TCN adopts a temporally focused approach with minimal spatial modelling. Despite these design differences, all three models produce outputs on the same downsampled spatial grid, enabling direct and consistent comparison using spatial error metrics.

C. Training and Implementation

All models—ConvLSTM, 3D-CNN, and TCN—were implemented using the PyTorch framework and trained under identical hardware and data conditions to ensure a fair comparison. The hardware setup consisted of an NVIDIA RTX 2080 SUPER GPU (8 GB VRAM), an AMD Ryzen 7 5700X 8-core CPU, and 32 GB RAM.

To ensure reproducibility, we fixed random seeds across NumPy and PyTorch (seed = 42) and disabled non-deterministic CuDNN behavior. Training used the Adam optimizer with an initial learning rate of 0.001, weight decay of 1e-5, and a learning rate scheduler that reduced the learning rate by a factor of 0.5 upon plateauing of the validation loss. A maximum of 100 epochs was allowed with early stopping patience of 15 epochs, and gradient clipping with a maximum norm of 5.0 was applied to stabilize ConvLSTM training. Dropout regularization (p = 0.3) was used in the Conv3D layers to mitigate overfitting. The full set of architectural specifications and training hyperparameters is summarized in Table 3 for clarity and reproducibility.

Due to memory and computational constraints, we used a batch size of 1 for all models. The Adam optimizer was employed with an initial learning rate of 0.001. Early stopping was applied based on the validation Mean Squared Error (MSE); if no improvement was observed for 100 consecutive epochs, training was halted to prevent overfitting. Training durations varied by model complexity, with ConvLSTM requiring approximately 57.07 hours, 3D-CNN taking 18.82 hours, and TCN completing in 17.06 hours (Fig. 1 (c)).

ConvLSTM demanded additional memory management strategies due to its recurrent nature. We applied gradient checkpointing to reduce GPU memory usage by recomputing intermediate activations during backpropagation, at the cost of increased training time.

Each model was trained to map the preceding 6 hours of rainfall input to a forecast of the subsequent 6 hours using chunked IMERG rainfall tensors. We saved the model weights at the epoch yielding the best validation loss.

1) Performance Metrics

To assess forecasting accuracy, we computed Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) over the entire validation set, including all time steps and spatial grid points. These metrics are defined as (1) and (2):

$$\mathrm{RMSE} = \sqrt{\frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{N}(p_{i,j} - o_{i,j})^2} \tag{1}$$

$$MAE = \frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{N}|p_{i,j} - o_{i,j}|j \tag{2}$$

where $p_{i,j}$ and $o_{i,j}$ are the predicted and observed rainfall at grid cell j and time step i, M is the number of temporal samples, and N is the number of spatial grid points. These metrics were computed on the test dataset at the 6-hour lead time. RMSE emphasizes large deviations (especially relevant for extreme rainfall), while MAE provides an intuitive measure of average error.

2) Convergence and Learning

Fig. 1 (a) and (b) shows the training and validation loss at the best epoch for each model, along with the absolute difference between the two. ConvLSTM exhibited the smallest train-validation gap (−0.02), indicating strong generalization. In contrast, 3D-CNN and TCN showed larger gaps (1.28 and 0.76 respectively), suggesting mild overfitting. Fig. 1 (c) compares the total training duration, highlighting the computational intensity of ConvLSTM despite its superior accuracy.

provides epoch-wise performance trends. ConvLSTM demonstrated smooth and stable convergence across all metrics. TCN showed rapid early learning but plateaued quickly with higher residual errors. 3D-CNN displayed fluctuating behaviour, likely due to its sensitivity to spatiotemporal kernel configurations.

TABLE 3. SUMMARY OF MODEL PERFORMANCE METRICS AT THE BEST EPOCH.

Model	Validation Loss (MSE, mm²/hr²)	Train Loss (MSE, mm²/hr²)	MAE (mm/hr)	RMSE (mm/hr)	Loss Difference (mm²/hr²)
ConvLSTM	1.2123	1.1876	0.1148	0.3932	−0.0247
3D-CNN	1.3563	2.6358	0.2608	0.4449	1.2795
TCN	1.9682	2.7315	0.2805	0.5478	0.7633

3) Inference Speed

We evaluated the inference time per forecast sequence on the same GPU. TCN was the most efficient, requiring approximately 0.1 seconds per inference. 3D-CNN followed with 0.3 seconds, while ConvLSTM, due to its sequential nature, required 0.5 seconds per forecast. All models are thus capable of near real-time operation with 30 minutes update cycles.

4) Experimental Robustness

Each experiment was repeated three times with different random seeds to ensure robustness. The reported performance differences were consistent across runs, with low standard deviation in both RMSE and MAE, confirming the reliability of the results.

D. Results and Discussion

This section presents the evaluation of the proposed deep learning models, ConvLSTM, 3D-CNN, and TCN across quantitative accuracy, training dynamics, computational efficiency, and spatial prediction quality. All models were assessed under consistent experimental conditions using the same input-output sequences of 6-hour cumulative IMERG rainfall tensors.

To ensure a comprehensive comparison, the models were evaluated using both numerical error metrics and visual inspection of forecast maps. The primary quantitative metrics used were validation loss, mean absolute error (MAE), and root mean squared error (RMSE), each computed across the entire validation set. In addition, training loss, train–validation loss gap, and total training time were considered to assess convergence behaviour and generalization. Finally, qualitative comparisons were made using predicted rainfall maps against ground truth observations.

1) Quantitative Results

Table 3 presents the quantitative evaluation of the three models, ConvLSTM, 3D-CNN, and TCN, based on six key metrics: training loss (MSE, mm²/hr²), validation loss (MSE, mm²/hr²), independent test loss (MSE, mm²/hr²), mean absolute error (MAE, mm/hr), root mean squared error (RMSE, mm/hr), and the difference between training and validation loss (MSE, mm²/hr²) at the best epoch. Together with the architectural and hyperparameter details summarized in Table 2, these results provide a comprehensive view of model performance, predictive accuracy, and generalization ability.

Among the models, ConvLSTM consistently outperformed the others across all metrics. It achieved the lowest validation loss of 1.2123 MSE (mm²/hr²), training loss of 1.1876 MSE (mm²/hr²), and test loss of 0.541 MSE (mm²/hr²), with a MAE of 0.115 mm/hr and an RMSE of 0.393 mm/hr. These results reflect ConvLSTM's strong ability to capture both spatial and temporal rainfall patterns accurately. The minimal train–val loss difference of −0.0247 MSE (mm²/hr²) further indicates that ConvLSTM avoided overfitting and maintained stable generalization on unseen data.

In contrast, 3D-CNN exhibited higher validation loss (1.3563 MSE, mm²/hr²) and substantially larger training loss (2.6358 MSE, mm²/hr²), resulting in a train–val gap of 1.2795 MSE (mm²/hr²), the largest among the models. Although its test loss (0.499 MSE, mm²/hr²) was competitive, this suggests that the model relied heavily on fitting the training distribution and did not generalize as reliably. Its MAE and RMSE (0.261 mm/hr and 0.445 mm/hr, respectively) were superior to TCN's but

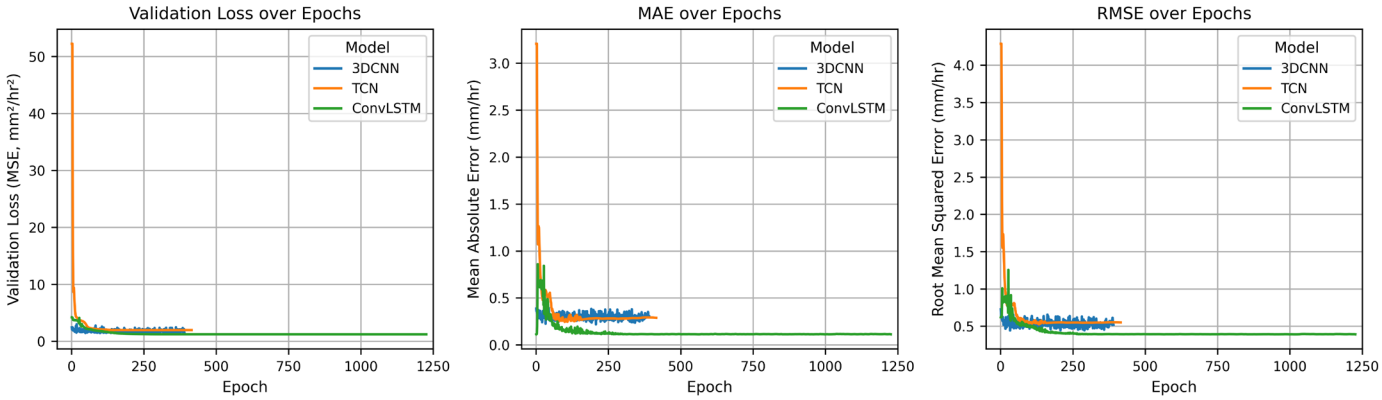

Fig. 2. Epoch-wise performance trends for all models. (a) Validation loss, (b) mean absolute error (MAE), and (c) root mean square error (RMSE). Subfigures (a) - (c) are arranged from left to right.

inferior to ConvLSTM's.

TCN recorded the highest validation loss (1.9682 MSE, mm²/hr²), MAE (0.281 mm/hr), and RMSE (0.548 mm/hr). While its train–val loss difference of 0.7633 MSE (mm²/hr²) was less severe than 3D-CNN's, its independent test loss (0.535 MSE, mm²/hr²) confirmed weaker performance when exposed to new events. These results suggest that TCN, despite its efficiency and fast convergence, struggled to capture the complex spatiotemporal dependencies required for accurate rainfall forecasting.

To interpret the error metrics in context, MAE reflects the average magnitude of errors in predictions, making it intuitive for understanding deviations from observed rainfall. For instance, an MAE of 0.115 mm/hr means that on average, ConvLSTM's predictions were only off by ~0.11 mm/hr, a highly acceptable margin for practical rainfall forecasting. RMSE emphasizes larger errors and is more sensitive to extremes, making it critical for flood-related applications. Lower RMSE values, particularly those under 0.4 mm/hr, indicate strong robustness against extreme rainfall errors.

These differences are further illustrated in Fig. 1a–c, where (a) compares training, validation, and test losses at the best epoch, (b) highlights train–val loss differences, and (c) compares training durations. The numerical and visual analyses consistently identify ConvLSTM as the most accurate and stable model, with the smallest error margins and minimal overfitting, albeit at higher computational cost.

These error magnitudes are also meaningful in a real-world context. In tropical regions, hourly rainfall rates often range between 0–5 mm/h during light to moderate rain, while convective storms can exceed 10 mm/h [18]. Therefore, an MAE of approximately 0.1 mm/h and RMSE below 0.4 mm/h indicate that the models, particularly ConvLSTM, are capable of capturing the dominant rainfall distribution with high fidelity. The relatively low MAE also suggests that the models are not merely fitting to frequent no-rain conditions but are effectively capturing actual rainfall dynamics.

2) Training Behaviour and Convergence

The training dynamics of each model were further analysed to assess their learning stability and generalization behaviour over time. Fig. 2 presents the epoch-wise evolution of three key indicators: validation loss, mean absolute error (MAE), and root mean squared error (RMSE).

ConvLSTM demonstrated smooth and consistent convergence across all metrics. Its validation loss steadily decreased without significant fluctuations, indicating stable gradient updates and effective learning. Both MAE and RMSE curves followed a similarly stable downward trend, reflecting its robustness in learning temporal rainfall dependencies.

TCN, on the other hand, exhibited rapid convergence during the early epochs, achieving a significant reduction in validation loss within the first 10 - 15 epochs. However, the model quickly plateaued, with minimal improvement in subsequent iterations. This behaviour suggests that TCN is efficient in learning basic patterns but lacks the capacity to model more complex spatiotemporal interactions needed for higher precision in long-range rainfall forecasting.

3D-CNN showed comparatively unstable convergence behaviour. Its validation loss and error metrics fluctuated across epochs, with signs of overfitting emerging in the later stages of training. This volatility may be attributed to the model's sensitivity to spatial kernel configurations and lack of explicit temporal modelling, which limits its ability to extract consistent features from sequential rainfall inputs.

Overall, the learning curves reinforce the quantitative findings in Section D.2: while TCN and 3D-CNN offer faster convergence, their limited capacity to generalize to unseen data results in higher prediction errors. ConvLSTM, despite requiring more epochs and computational resources, consistently achieved better generalization and lower validation errors. This relationship between training dynamics and generalization is further supported by the metrics discussed in Section D.2, where ConvLSTM's smooth convergence translated into the lowest validation errors, while the faster convergence of TCN and fluctuating behaviour of 3D-CNN coincided with higher forecasting errors.

3) Computational Efficiency

In addition to forecast accuracy, computational cost plays a key role in determining the operational viability of deep learning models, particularly for time-sensitive or resource-constrained environments such as regional weather services or edge-based flood warning systems.

Fig. 1(c) compares the total training time required for each model. ConvLSTM incurred the highest training cost, requiring approximately 57.07 hours to converge. This is expected given its recurrent structure, which processes spatiotemporal sequences step-by-step and introduces memory-intensive operations such as gating and backpropagation through time. Additionally, gradient checkpointing was used to manage memory, further increasing computational overhead.

In contrast, 3D-CNN and TCN completed training significantly faster, requiring 18.82 hours and 17.06 hours, respectively. These models leverage parallel convolutional operations without recurrence, making them more efficient on modern GPU hardware.

Beyond training time, we also measured GPU memory consumption. ConvLSTM peaked at approximately 6.5 GB, 3D-CNN used around 5.0 GB, and TCN remained below 4.0 GB, confirming that all three models are capable of running on an 8 GB GPU. However, ConvLSTM leaves little headroom for higher-resolution data or longer input sequences without additional optimization.

This analysis reveals a clear trade-off between model accuracy and computational efficiency. ConvLSTM provides the highest predictive performance but at the cost of longer training time and higher memory usage. TCN, on the other hand, is highly lightweight and fast to train but exhibits noticeably lower accuracy, especially under heavy rainfall scenarios. 3D-CNN strikes a middle ground in both training cost and accuracy.

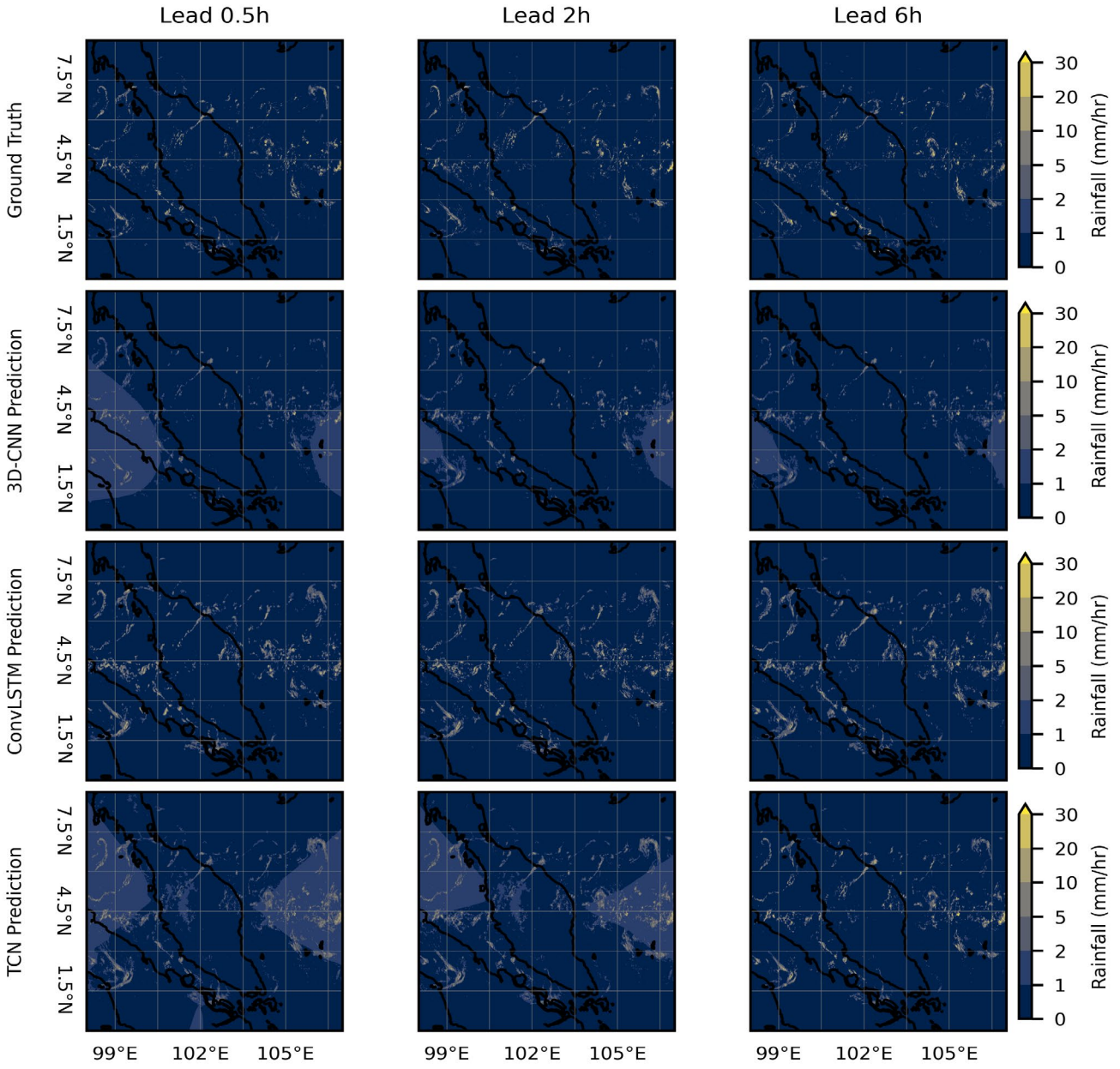

Fig. 3. Spatial rainfall predictions at short (0.5 h), medium (2 h), and long (6 h) lead times. Ground truth and predictions from 3D-CNN, ConvLSTM, and TCN models are shown for comparison across different forecast horizons.

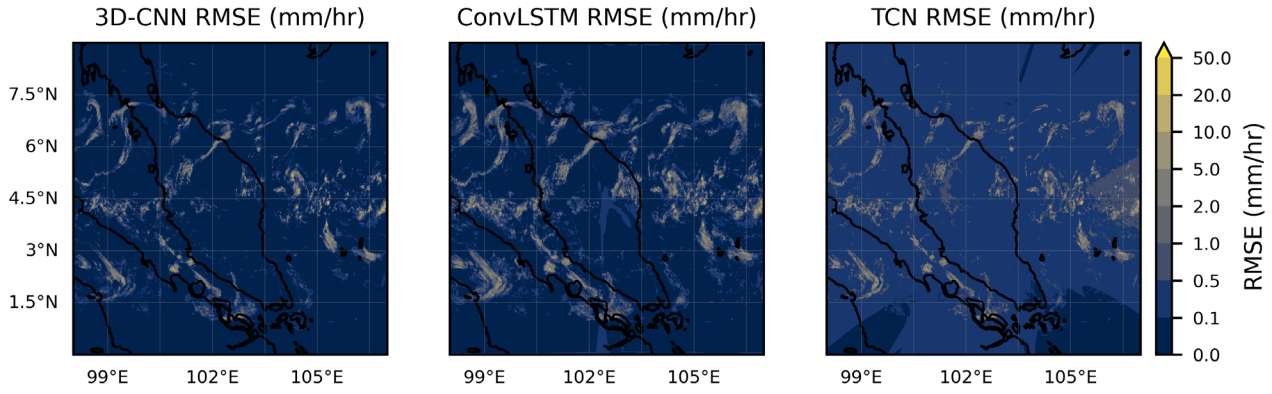

Fig. 4. Spatial RMSE maps averaged over the 6-hour forecast (12 forecast frames). Higher values indicate greater prediction errors across Peninsular Malaysia.

These trade-offs are important in real-world deployments. In high-performance settings where predictive accuracy and spatial detail are critical, such as urban flood early warning systems, ConvLSTM may be the most suitable despite its higher cost. For applications requiring fast inference and low overhead, such as regional monitoring or agricultural advisories, models like TCN or 3D-CNN may suffice.

Importantly, all three models were successfully trained and tested on a single consumer-grade GPU, highlighting their feasibility in low-resource environments. This opens the door for practical use in institutions lacking access to large-scale cloud infrastructure. Future work may further optimize deployment using model compression techniques such as quantization, pruning, or knowledge distillation [8], enabling faster inference without significant performance loss.

4) Limitations

Although IMERG provides valuable global coverage and high temporal resolution, it has known limitations. Its ~0.1° (~10 km) native resolution may not capture fine-scale convective structures typical of tropical storms, potentially underestimating localized rainfall peaks that are critical for urban flood forecasting. IMERG retrievals are also subject to satellite sampling gaps, retrieval algorithm uncertainties, and gauge-correction biases, especially in regions with sparse ground observations such as parts of Peninsular Malaysia. These uncertainties may propagate into the model forecasts, influencing both error magnitudes and spatial patterns. Future work should therefore validate forecasts against independent ground-based radar or rain-gauge networks to better quantify these biases.

In addition, training directly on raw rainfall values, while preserving interpretability, may reduce training stability compared to normalized inputs. This trade-off was acceptable given our emphasis on extreme-rainfall sensitivity, but should be revisited in future work. Alternative approaches such as selective normalization (e.g., log-scaling or quantile transformation) or loss reweighting could offer a balance between physical interpretability and stable optimization.

5) Qualitative Forecast Evaluation

While quantitative metrics provide an overall summary of model performance, visual inspection of spatial rainfall outputs and localized errors is essential to assess the practical reliability of predictions. This section presents qualitative analyses using representative forecast samples, regional bias patterns, and spatial RMSE distributions.

Fig. 3 illustrates spatial rainfall predictions at short (0.5 h), medium (2 h), and long (6 h) lead times, comparing the outputs of 3D-CNN, ConvLSTM, and TCN models against IMERG ground truth. The ground truth maps show localized convective rainfall cells across Peninsular Malaysia, with sharp spatial boundaries and high-intensity areas particularly along the west coast.

Among the predictions, ConvLSTM demonstrates the closest resemblance to the ground truth across all lead times. It preserves rainfall structure and intensity, while maintaining spatial coherence and reducing spurious background rainfall. Notably, ConvLSTM sustains predictive skill even at 6 h lead time, showing its strength in temporal consistency and spatiotemporal learning.

By contrast, 3D-CNN captures the broader rainfall extent but suffers from blocky noise artifacts, especially at longer lead times. High-intensity regions tend to be oversmoothed, and faint background rainfall appears in otherwise dry regions, suggesting its convolutional depth diffuses finer spatial details.

The TCN model produces the weakest predictions, with scattered low-intensity rainfall across the domain and diffuse patterns that fail to capture the spatial structure of convective systems. The degradation becomes more pronounced with lead time, reflecting the model's limited spatial filtering capacity.

These qualitative differences support the quantitative evaluation in previous sections and highlight ConvLSTM's superiority in both short- and longer-range rainfall forecasts.

To further characterize temporal error accumulation, Fig. 4 presents spatial RMSE maps for each model, obtained by averaging RMSE values at each grid cell across the 12 forecasted timesteps. The revised colour scale enhances contrast, allowing clearer identification of localized errors and background noise.

ConvLSTM maintains the lowest RMSE across most of the domain, especially in central and southern regions. The updated visualization reinforces its consistent performance in both spatial coherence and temporal rainfall evolution. 3D-CNN exhibits higher noise across broader areas, with noticeable patchiness particularly around the northern and coastal zones. This indicates less stable spatial generalization and suggests that 3D-CNN may introduce spurious fluctuations when learning spatiotemporal features. TCN displays widespread spatial error with diffuse RMSE patterns and prominent artifacts in the northeast and border regions, consistent with its tendency to generate temporally fragmented predictions.

This visualization confirms the robustness of ConvLSTM, while highlighting the noise sensitivity of 3D-CNN and the instability of TCN. Despite challenges in interpreting RMSE in low-rainfall areas, this spatial-temporal aggregation provides valuable insight into model behaviour across the full forecast horizon.

IV. Conclusion

This study investigated the effectiveness of three deep learning models, 3D-CNN, ConvLSTM, and TCN, for short-term rainfall forecasting using IMERG satellite data. Among the models, ConvLSTM demonstrated the best overall performance, achieving the lowest RMSE and MAE scores while accurately capturing spatial rainfall structures and temporal evolution, particularly during high-intensity events. 3D-CNN showed moderate success but struggled with temporal consistency, while TCN exhibited significant limitations in both spatial coherence and rainfall intensity prediction.

The results emphasize the critical role of spatiotemporal modelling capabilities in rainfall forecasting. ConvLSTM's strong performance highlights its potential for deployment in real-time flood forecasting and early warning systems, though further validation with independent datasets and baselines is necessary. Beyond hydrology, such models can also be

embedded within electrical and electronic systems, including distributed sensor networks, IoT platforms, and telecommunication-based warning frameworks, enabling automated monitoring and rapid dissemination of alerts.

While this study primarily focused on continuous error metrics such as RMSE and MAE, we acknowledge that categorical metrics such as Probability of Detection (POD), False Alarm Ratio (FAR), and Critical Success Index (CSI/ETS) provide important complementary insights, particularly for threshold-based rainfall event detection and lead-time evaluation. These were not included in the present work due to computational and scope constraints, but will be incorporated in future studies to strengthen event-based verification and operational relevance.

Future work will explore model generalization across multiple geoclimatic regions, the integration of additional meteorological variables (e.g., wind, temperature, humidity), and the extension of forecast horizons. Furthermore, incorporating uncertainty quantification and coupling the model with hydrological or inundation simulators will enhance its applicability within an end-to-end flood risk management framework.

In addition, hybrid ConvLSTM architectures will be investigated to balance predictive skill with faster runtime. For example, restricting ConvLSTM computation to a low-dimensional latent space or coupling a CNN-based encoder with a lightweight temporal module such as the tested TCN could retain accuracy while improving computational efficiency. Multi-scale (pyramidal) recurrent variants may also be explored as lower-cost alternatives.

To enable deployment on edge devices such as local flood monitoring stations, model compression strategies including pruning, quantization-aware training, and knowledge distillation will be examined to reduce memory footprint and accelerate inference. Finally, releasing benchmark code, evaluation metrics, and preprocessed datasets will promote reproducibility and serve as a community resource for advancing rainfall forecasting research.

ACKNOWLEDGMENT

The authors express their gratitude as well as appreciation to the Faculty of Electrical Engineering, Universiti Teknologi MARA Shah Alam for their assistance.

CONFLICT OF INTEREST

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

ETHICS STATEMENT

This study did not involve human participants or animals.

REFERENCES

[1] R. Rahimi, P. Ravirathinam, A. Ebtehaj, A. Behrangi, J. Tan, and V. Kumar, 'Global Precipitation Nowcasting of Integrated Multi-satellitE Retrievals for GPM: A U-Net Convolutional LSTM Architecture', June 2024, doi: 10.1175/JHM-D-23-0119.1.

[2] S. Rahman, 'Malaysia's Floods of December 2021: Can Future Disasters be Avoided?', no. 2022, 2022.

[3] P. Hewage *et al.*, 'Temporal convolutional neural (TCN) network for an effective weather forecasting using time-series data from the local weather station', *Soft Comput.*, vol. 24, no. 21, pp. 16453–16482, Nov. 2020, doi: 10.1007/s00500-020-04954-0.

[4] W. Liu, Y. Wang, D. Zhong, S. Xie, and J. Xu, 'ConvLSTM Network-Based Rainfall Nowcasting Method with Combined Reflectance and Radar-Retrieved Wind Field as Inputs', *Atmosphere*, vol. 13, no. 3, Art. no. 3, Mar. 2022, doi: 10.3390/atmos13030411.

[5] D.-K. Kim *et al.*, 'Improving precipitation nowcasting using a three-dimensional convolutional neural network model from Multi Parameter Phased Array Weather Radar observations', *Atmospheric Res.*, vol. 262, p. 105774, Nov. 2021, doi: 10.1016/j.atmosres.2021.105774.

[6] Y. Lin, I. Koprinska, and M. Rana, 'Temporal Convolutional Attention Neural Networks for Time Series Forecasting', in *2021 International Joint Conference on Neural Networks (IJCNN)*, Shenzhen, China: IEEE, July 2021, pp. 1–8. doi: 10.1109/IJCNN52387.2021.9534351.

[7] P. Hewage *et al.*, 'Temporal convolutional neural (TCN) network for an effective weather forecasting using time-series data from the local weather station', *Soft Comput.*, vol. 24, no. 21, pp. 16453–16482, Nov. 2020, doi: 10.1007/s00500-020-04954-0.

[8] K. Gikunda and N. Jouandeau, 'Lightweight Deep Learning for Weather Prediction and Forecasting in Africa'.

[9] G. J. Huffman *et al.*, 'Integrated Multi-satellite Retrievals for the Global Precipitation Measurement (GPM) Mission (IMERG)', in *Satellite Precipitation Measurement: Volume 1*, V. Levizzani, C. Kidd, D. B. Kirschbaum, C. D. Kummerow, K. Nakamura, and F. J. Turk, Eds, Cham: Springer International Publishing, 2020, pp. 343–353. doi: 10.1007/978-3-030-24568-9_19.

[10] X. SHI, Z. Chen, H. Wang, D.-Y. Yeung, W. Wong, and W. WOO, 'Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting', in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2015. Accessed: June 01, 2025. [Online]. Available: https://proceedings.neurips.cc/paper/2015/hash/07563a3fe3bbe7e3ba84431ad9d055af-Abstract.html

[11] J. Liu, L. Xu, and N. Chen, 'A spatiotemporal deep learning model ST-LSTM-SA for hourly rainfall forecasting using radar echo images', *J. Hydrol.*, vol. 609, p. 127748, June 2022, doi: 10.1016/j.jhydrol.2022.127748.

[12] D.-K. Kim *et al.*, 'Improving precipitation nowcasting using a three-dimensional convolutional neural network model from Multi Parameter Phased Array Weather Radar observations', *Atmospheric Res.*, vol. 262, p. 105774, Nov. 2021, doi: 10.1016/j.atmosres.2021.105774.

[13] Q. Guo, Z. He, and Z. Wang, 'Monthly climate prediction using deep convolutional neural network and long short-term memory', *Sci. Rep.*, vol. 14, no. 1, p. 17748, July 2024, doi: 10.1038/s41598-024-68906-6.

[14] H. C. Kilinc, 'Daily Streamflow Forecasting Based on the Hybrid Particle Swarm Optimization and Long Short-Term Memory Model in the Orontes Basin', *Water*, vol. 14, no. 3, Art. no. 3, Jan. 2022, doi: 10.3390/w14030490.

[15] G. J. Huffman *et al.*, 'NASA global precipitation measurement (GPM) integrated multi-satellite retrievals for GPM (IMERG)', *Algorithm Theor. Basis Doc. ATBD Version*, vol. 4, no. 26, p. 30, 2015.

[16] Y. Shen, A. Xiong, Y. Wang, and P. Xie, 'Performance of high-resolution satellite precipitation products over China', *J. Geophys. Res. Atmospheres*, vol. 115, no. D2, 2010, doi: 10.1029/2009JD012097.

[17] L. Tu and L. Duan, 'Spatial downscaling analysis of GPM IMERG precipitation dataset based on multiscale geographically weighted regression model: a case study of the Inner Mongolia Reach of the Yellow River basin', *Front. Environ. Sci.*, vol. 12, Apr. 2024, doi: 10.3389/fenvs.2024.1389587.

[18] A. Naha Biswas, Y. H. Lee, W. Tao Yeo, W. C. Low, D. Y. Heh, and S. Manandhar, 'Statistical Analysis of Atmospheric Delay Gradient and Rainfall Prediction in a Tropical Region', *Remote Sens.*, vol. 16, no. 22, Art. no. 22, Jan. 2024, doi: 10.3390/rs16224165.