Supporting Information for "Probabilistic Forecasting Model for Tropical Cyclone Intensity Based on Diffusion Model"

Jingjia Luo [1,2] , Peng Yang [1,2] and Fan Meng [1,2,*]

1 . State Key Laboratory of Climate System Prediction and Risk Management (CPRM), Nanjing University of Information Science and Technology, Nanjing, China

2 . School of Artificial Intelligence, Nanjing University of Information Science and Technology, Nanjing, China

* Correspondence: meng@nuist.edu.cn

1. Baseline Models Parameters

1.1. CNN

This model serves as our primary convolutional baseline. It employs a deep 2D convolutional neural network (CNN) encoder to extract features from spatiotemporal image sequences, which are subsequently fused with auxiliary scalar variables for the final regression prediction.

Image Encoder: The encoder consists of six convolutional blocks, each using the ReLU activation function:

- Block 1: Conv2D (16 filters, 5×5 kernel), MaxPooling, BatchNormalization
- Block 2: Conv2D (32 filters, 5×5 kernel), MaxPooling, BatchNormalization
- Block 3: 3× Conv2D (64 filters, 3×3 kernels), MaxPooling
- Block 4: 3× Conv2D (128 filters, 3×3 / 2×2 / 3×3 kernels), MaxPooling
- Block 5: 3× Conv2D (256 filters, 3×3 kernels), MaxPooling
- Block 6: 3× Conv2D (512 filters, 3×3 kernels), followed by 1× Conv2D (1024 filters, 2×2 kernel)

Feature Fusion: The encoded spatiotemporal features are flattened and concatenated with the auxiliary feature vector.

Output MLP:

- Dense (1000 units, ReLU activation) → Dropout (rate = 0.2)
- Dense (200 units, ReLU activation) → Dropout (rate = 0.2)
- Output Layer: Dense (1 unit, linear activation)

Optimizer: Adam

Loss Function: Mean Squared Error (MSE)

1.2. ConvLSTM

This model employs ConvLSTM2D layers to directly process the 5D spatiotemporal image sequence input, thereby capturing the temporal evolution and spatial dependencies simultaneously.

ConvLSTM Layers: Two stacked ConvLSTM2D layers are used:

- Layer 1: 64 filters, 3×3 kernel, return_sequences=True
- Layer 2: 128 filters, 3×3 kernel, return_sequences=False

Feature Fusion: The feature maps from the final time step are flattened and concatenated with the auxiliary feature vector.

Output MLP:

- Dense (256 units, ReLU activation)
- Output Layer: Dense (1 unit, linear activation)

Optimizer: Adam

Loss Function: Mean Squared Error (MSE)

1.3. Transformer Model

This model adopts the encoder part of the standard Transformer architecture. To handle image-based inputs, each time-step image is first passed through a lightweight CNN to extract feature embeddings suitable for the Transformer encoder.

Model Configuration:

- Hidden dimension: 128
- Encoder layers: 4
- Attention heads: 8
- Feed-forward dimension: 512

Pooling: The encoder outputs a sequence of feature representations corresponding to all time steps, which are aggregated using global average pooling.

Output MLP:

- Dense (128 units, ReLU activation)
- Output Layer: Dense (1 unit, linear activation)

Optimizer: Adam

Loss Function: Mean Squared Error (MSE)

1.4. MC Dropout

This method introduces stochasticity by applying Dropout during both training and inference. By performing multiple stochastic forward passes at inference time, the model estimates predictive uncertainty based on the variability of its outputs.

Base Architecture: Identical to the Spatiotemporal 2D CNN baseline (Section 1.1)

Dropout Probability: 0.2 (applied in the Output MLP layer)

Loss Function: Mean Squared Error (MSE)

Optimizer: Adam

Monte Carlo (MC) Samples: 120

1.5. Deep Ensemble

The Deep Ensemble method trains multiple independent models with different random seeds. Predictions from all members are aggregated to estimate the mean prediction and associated uncertainty.

Base Architecture: Identical to the Spatiotemporal 2D CNN baseline (Section 1.1)

Loss Function: Mean Squared Error (MSE)

Optimizer: Adam

Ensemble Size: 10 independently trained networks. During inference, each network generates 12 predictions, resulting in a total of 120 ensemble members.

1.6. GAN

This model is a conditional generative adversarial network (cGAN) designed for regression tasks, used to generate predictive distributions.

Generator:

- Architecture: Based on the Spatiotemporal 2D CNN baseline (Section 1.1)

- To introduce stochasticity, a latent noise vector z(e.g., 100-dimensional vector sampled from a standard normal distribution) is concatenated with the combined feature vector of flattened image features and auxiliary features, then fed into the output MLP

Discriminator:

- Architecture: A multilayer perceptron (MLP)
- Input: A concatenated vector of the generator's intermediate combined features (flattened image features + auxiliary features) and the intensity value (either real or generated)
- Hidden Layers: Three dense layers (512, 256, 128 units) with LeakyReLU activation
- Output Layer: Dense (1 unit, Sigmoid activation)

Optimizers: Independent Adam optimizers for generator and discriminator

Loss Functions:

- Discriminator Loss: Binary Cross-Entropy (BCE)
- Generator Loss: Mean Squared Error (MSE) for regression + BCE for adversarial loss

Uncertainty Quantification: During inference, 120 different latent noise vectors ($z_1, \ldots, z_{120}$) are sampled and input to the generator, producing 120 distinct intensity predictions. This ensemble is used to estimate the predictive distribution.

All baseline models were trained on a single NVIDIA A800 GPU for 300 epochs, with a learning rate of $10^{-4}$and a batch size of 50. To ensure a fair comparison among baseline models, all probabilistic methods were evaluated under comparable computational resources. The MC Dropout model performed 120 stochastic forward passes during inference.The Deep Ensemble model consisted of 10 independently trained networks, each generating 12 samples, resulting in a total of 120 ensemble members.The GAN model was similarly configured to generate 120 ensemble members through sampling different latent noise vectors.