Self-Supervised Learning with Probabilistic Density Labeling for Rainfall Probability Estimation
— Supplementary Material —

Junha Lee[1,3*], Sojung An[2*], Sujeong You[3], Namik Cho[1]
[1]Seoul National University
[2]Korea Institute of Atmospheric Prediction Systems
[3]Korea Institute of Industrial Technology
{junhalee,nicho}@snu.ac.kr, ssojungan@kiaps.org, sjyou21@kitech.re.kr

In this supplementary material, we provide additional details and experimental results to support the main submission. Section A shows experiment settings in the main paper. Section B presents visualization results showing the impact of changes in probabilistic density labeling smoothing parameters. Section C provides mask reconstruction results for the pre-training. Section D shows the practical applicability across various rainfall patterns using KDE visualization. Section E describes and discusses the additional qualitative results.

A. Implementation details

We extensively experimented with parameters to tailor configurations for the real-world dataset, considering their noise levels and dynamic characteristics to ensure a fair comparison of models. We utilized Optuna[1] to optimize parameters, randomly employing values of 3 configs as follows:

- β_1: [0.5, 0.9, 0.95]
- β_2: [0.95, 0.999]
- learning rate: [1e-6, 5e-6, 5e-5, 1e-4, 5e-4, 1e-3, 5e-3].

We found the most suitable parameters through 30 trials, incorporating pruning. To enhance the efficiency of the optimization process, we employed pruning to discard trials that are expected to be unpromising in the future.

Our experiments are conducted on the NVIDIA A100 GPUs. SSLPDL and the baseline models are tailored to operate efficiently on a single GPU without model parallelization. Training configuration employs an AdamW optimizer with momentum parameters β_1=0.5 and β_2=0.95, and a weight decay of 0.01. Additionally, we defined the loss weight of the cross-entropy to w_i=1, 5, 10. We trained the baseline frameworks by comparing the settings identified by Optuna with their officially released original papers: ConvLSTM[2], Metnet[3], MAE[4], Swin-UNet, and PostRainBench[5].

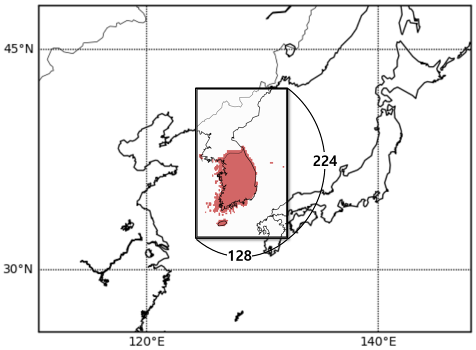

Figure 1. RDAPS domain visualization. The geographical scope encompasses East Asia, the prediction area defined by RDAPS, and serves as the input data. We align this area with the latitude coordinates of the Korean Peninsula dataset to match the ground truth. When verifying the data, sea observations are adjusted using limited data, ensuring that only land observations (color pixels) are verified to uphold reliability.

A.1. ConvLSTM

ConvLSTM [3] was presented as a network for predicting space-time patterns by applying convolutions to the recurrent state transitions of an LSTM cell. KIM *et al.* [1]

[*] Equal contribution

[1] https://optuna.org/

[2] https://github.com/ndrplz/ConvLSTM_pytorch

[3] https://github.com/osilab-kaist/KoMet-Benchmark-Dataset

[4] https://github.com/facebookresearch/mae_st

[5] https://github.com/yyyujintang/PostRainBench

Block	Layer	Resolution	Channels
Input	-	224 × 128	16
Masking	Tokenization	224 × 128 → 448	16 → 1024
	Masking	(4 × 448) → (4 × 448 × (1-M))	1024
Encoder	Embedding	(4 × 448 × (1-M))	1024
	MLP × 24	(4 × 448 × (1-M))	1024
	Layer Norm	(4 × 448 × (1-M))	1024
Decoder	Mask tokens appending	(4 × 448 × (1-M)) → (4 × 448)	1024
	Embedding	(4 × 448)	1024
	MLP × 8	(4 × 448)	1024
	Layer Norm	(4 × 448)	1024
	MLP	(4 × 448)	1024 → (c × 32)
Output	-	224 × 128	c

Table 1. The details of the MAE on our dataset. Mask tokens appending involves appending the number of pixels equivalent to the mask ratio (M) alongside the unmasked pixels to form the final composition of pixels. Layer norm represents the Layer Normalization layers, and $(\cdot)$ denotes the flattened shape of each axis.

utilized ConvLSTM to analyze three-dimensional patterns in NWP forecasts, enabling spatial pattern analysis through Convolution and correlation analysis between variables using LSTM. We employed ConvLSTM configured with a window size of 3 and a learning rate of 1e-3 for evaluation.

A.2. Metnet

Sonderby *et al.* [4] proposed Metnet for precipitation nowcasting. Metnet consists of ConvLSTM cells that employ an axial attention mechanism. Kim *et al.* [1] used Metnet for a precipitation correction that outperformed other architectures. Based on our experimental findings, we chose a window size of 3 and an axial channel of 32 for this official paper. We set the window size to three for the model validation and the learning rate of 5e-3.

A.3. MAE

We used a 3D patch size of 4×16×16 for time, height, and width. Additionally, we adopt two separable positional embeddings for the encoder and decoder for space-time positional embeddings. Note that the embeddings are learnable positional embedding [6]. Table 1 shows the details of the structure we applied to our dataset using MAE's network. For pre-training, we follow the official configurations. We use the AdamW optimizer with a batch size of 64. We employed the same pre-training settings as in Table 2 for our model.

Config	Pre-training	Transfer learning
optimizer	AdamW	AdamW
optimizer momentum	β_1, β_2=0.9, 0.95	β_1, β_2=0.5, 0.95
weight decay	0.05	0.01
learning rate	1.6e-3	1e-4
learning rate schedule	cosine decay	-
warmup epochs	120	-
epochs	1000	200
batch size	64	32
gradient clipping	0.02	-
checkpoint	$\mathcal{L}_{rec}$	mIoU

Table 2. Hyperparameters setting

A.4. Swin-Unet and PostRainBench

PostRainBench [5] is a post-processing model based on Swin-Unet, incorporating a channel attention module to aggregate spatial information. In line with the official paper, we trained the model using multitask learning that involves both regression and segmentation tasks. The training configuration was set with a batch size of 32 and a learning rate of 1e-4. For the segmentation task, we used class weights of [1, 5, 10], which were in line with those used in our models.

A.5. Ours

Our encoder employs an InternImage [7] architecture. The proposed model uses a spatiotemporal masking method. We set a spatial patch size of 16×16 and a temporal patch size of 4. For a 16×224×128 input, the patch size was 4×16×16. Our pre-training configuration follows that presented in [7]. Table 3 presents details on the blocks of the encoder network and the size of input variables according to each block in our dataset. Vanilla UperNet is structured in the decoder for both reconstruction and segmentation tasks.

The following hyper-parameters are set for our model. Pre-training configuration employs an AdamW optimizer with momentum parameters β_1=0.9 and β_2=0.95, and a weight decay of 0.05. We set the learning rate of 1.6e-3, beginning with a warm-up period spanning 120 epochs and culminating in 1000 epochs. We utilize a batch size of 64 and apply gradient clipping at 0.02 to promote stability during training. For the training phase, the learning rate is reduced to 1e-4, with the batch size adjusted to 32, optimizing the network for subsequent downstream tasks. The detailed

Block	Layer	Resolution	Channels
Input	-	224 × 128	16
Masking	Tokenization	224 × 128 → (4 × 448)	16 → 1024
	Masking	(4 × 448) → (4 × 448 × (1-M))	1024
	Mask tokens appending	(4 × 448 × (1-M)) → (4 × 448)	1024
	Embedding	(4 × 448)	1024
	Layer norm	(4 × 448)	1024
	-	(4 × 448) → 224 × 128	1024 → 16
Stem	Conv3×3	224 × 128 → $224/2^i$ × $128/2^i$	16 → 64
	Batch Norm	$224/2^i$ × $128/2^i$	64
	GELU	$224/2^i$ × $128/2^i$	64
	Conv3×3	$224/2^i$ × $128/2^i$	64
	Batch norm	$224/2^i$ × $128/2^i$	64
	GELU	$224/2^i$ × $128/2^i$	64
	Dropout	$224/2^i$ × $128/2^i$	64
Stage$_{i\in\{1,2,3,4\}}$	Layer norm	$224/2^i$ × $128/2^i$	$64\times2^{i-1}$
	Deformable Conv	$224/2^i$ × $128/2^i$	$64\times2^{i-1}$
	Dropout	$224/2^i$ × $128/2^i$	$64\times2^{i-1}$
	Residual	$224/2^i$ × $128/2^i$	$64\times2^{i-1}$
	Layer norm	$224/2^i$ × $128/2^i$	$64\times2^{i-1}$
	Conv3×3	$224/2^i$ × $128/2^i$ → $224/2^{i+1}$ × $128/2^{i+1}$	$64\times2^{i-1}$ → 64×2^i

Table 3. We provide the details of the SSLPDL encoder on our dataset. Mask tokens appending involves appending the number of pixels equivalent to the mask ratio (M) alongside the unmasked pixels to form the final composition of pixels. Conv3×3, Batch norm and Layer norm is the 2D convolutional layer with 3 × 3 kernel, Batch normalization layers, and Layer Normalization layers, respectively. $(\cdot)$ denotes the flattened shape of each axis. Residual represents residual layers for adding the previous stage output features.

architecture configurations of *Encoder* of SSLPDL are described in Table 3.

B. Probabilistic density labeling

Representing the probability of rainfall instances through probabilistic density labeling, we investigate various gradients to identify smooth transition points in probability values by adjusting the α parameter. Figure 2 shows the probability values of group 2 and group 3 of precipitation, smoothed as α is adjusted. We enhanced the model generalization performance through gradient-based smoothing adjustments and effectively prevented overfitting. Label smoothing moderates the confidence assigned to the true labels, preventing the model from exhibiting excessive certainty.

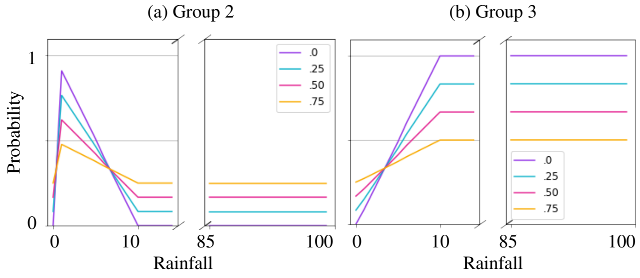

Figure 2. Visualization results illustrate the impact of varying the smoothing parameter α on the probability estimates for two distinct rainfall intensity groups: (a) Group 2 with rain $0.1 \leq y < 10$, and (b) Group 3 with moderate to heavy rain $10 \leq y < 100$. Each line corresponds to a different α value, demonstrating the effect of parameter adjustment on the transition smoothness of probability estimates.

C. Mask reconstruction results

In this section, we analyze the reconstruction results obtained during pre-training. Figure 3 illustrates the reconstruction performance for input data comprising 16 meteorological variables, based on observations collected on August 7, 2022, at 07 UTC. These variables, observed across multiple vertical atmospheric levels, are crucial for understanding weather phenomena, including precipitation forecasting.

As shown in Figure 3, the proposed model demonstrates the capability to accurately reconstruct nonlinear and highly variable parameters, such as rainfall. This achievement highlights the model's ability to represent physical flows in real-world meteorological datasets. The numerical model exhibits strong interdependence among variables, even under a 90% masking ratio. The reconstructed outputs closely align with the ground truth, accurately capturing complex variables such as relative humidity (RH) and rainfall (RAIN), which are known for their significant variability. Pre-training enables the model to infer context

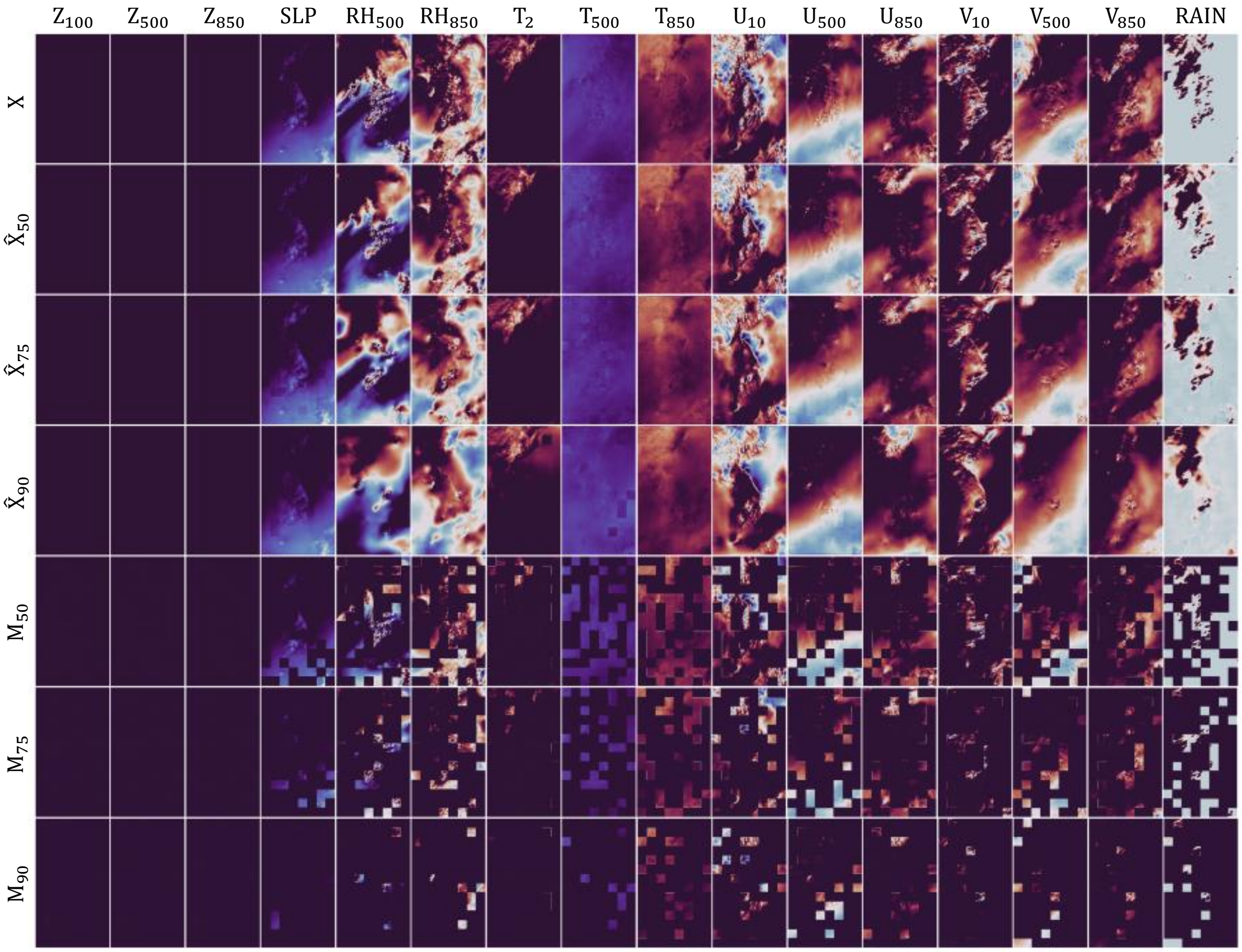

Figure 3. Variable reconstruction results using the pre-trained model on data from August 7, 2022, at 07 UTC. X, $\hat{X}$, and M denote input, reconstructed input, and masked input, respectively. The subscripts $(\cdot)$ of variables and M represent the vertical levels and mask ratio, respectively. The first row visualizes the normalized variables. **Rows 2-4** show the results $\hat{\text{X}}$ of reconstructing the masked pixels. **Rows 5-7** visualize the variables with mask. For the visualization, the masked values were set to -100, and a range of [-1, 1] by conducting z-score.

from incomplete data, resulting in improved prediction accuracy. However, higher masking percentages introduce limitations, such as noticeable blurring in reconstructed values. For instance, surface temperature (T_2), a key factor in rainfall prediction, is sensitive to minor discrepancies, leading to reduced downstream performance. Excessive masking also restricts the model's ability to learn robust representations, causing it to rely on superficial patterns in training data and hindering generalization to unseen datasets. By optimizing the masking ratio and leveraging pre-training capabilities, we aim to refine the reconstruction process while preserving the model's generalization performance.

D. Evaluation of components under different rainfall patterns.

We evaluated our model's practical applicability by analyzing the August 2022 data based on the four weather clusters in the Republic of Korea proposed in prior research [2], as shown in Figure 5. We extracted the components based on empirical orthogonal function and then clustered them using K-means clustering. The precipitation cases were grouped into four clusters, containing 44, 633, 12, and 55 cases, respectively. We visualize the kernel density estimation (KDE) of samples, representing the data's probability density function. Our results show that the predicted data patterns were closer to the ground truth and were adjusted to reflect the precipitation patterns better.

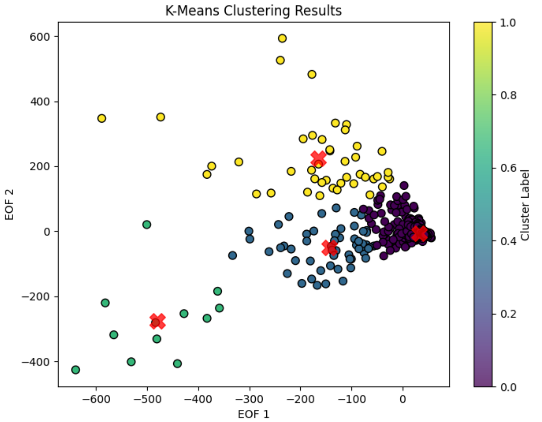

Figure 4. Weather clustering result on the RDAPS dataset.

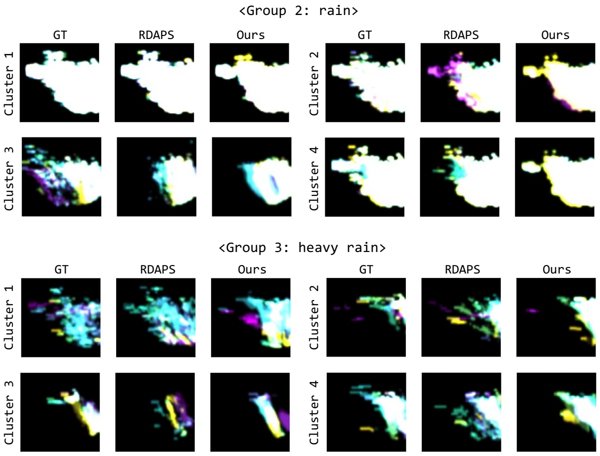

Figure 5. Evaluation of model's practical applicability based on the four weather clusters identified in the Republic of Korea through KDE analysis.

E. Qualitative results

For qualitative analysis, three instances (Figure 6, 7, and 8) of heavy rainfall events, which posed significant challenges for RDAPS prediction, were selected. The first row of each image depicts results obtained using traditional one-hot labeling, while the second row represents outcomes achieved through probabilistic density labeling. Unlike one-hot labeling, which tends to overfit specific groups without considering the imbalanced data characteristics, probabilistic density labeling offers greater flexibility across various groups, potentially mitigating data imbalances. These findings are anticipated to address the common issue of data imbalance in real-world datasets, providing a methodology applicable to diverse imbalanced data.

Figure 6 illustrates instances where all AI models predicted rainfall spreading in all directions but failed to forecast heavy rain inland. We present cases in which the proposed method exhibits limited performance in precipitation prediction and discuss the potential reasons behind this outcome. We empirically observed that prediction models struggle when sporadic showers rather than bands of rain clouds. Sporadic showers often arise from the complex nonlinear dynamics of the atmosphere and terrain, presenting irregular and intricate patterns that are challenging for models to learn. Rainfall patterns can significantly vary based on geographical and seasonal factors in each region. By incorporating comprehensive atmospheric conditions and terrain data, we can significantly enhance the accuracy of our results, enabling us to capture rainfall's diverse characteristics effectively.

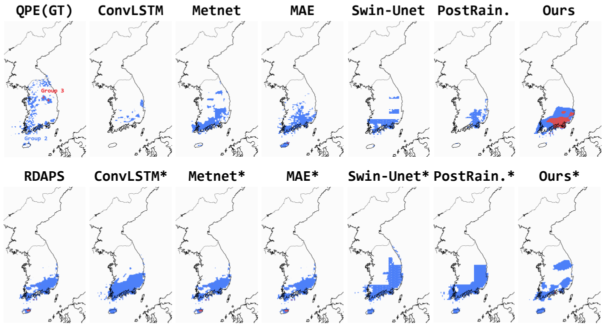

Figure 6. **Cluster 1.** Visualization result between benchmarks on August 15 2022 at 18 UTC (+28 h). As the stationary front moved southward, cold air from the northwest was advected into the upper atmosphere. At the same time, near the surface, abundant moisture remained due to recent rainfall, leading to a rise in temperature. This resulted in atmospheric instability due to the temperature difference between the upper and lower levels, leading to sporadic showers, characteristic of a typical localized precipitation event. Due to significant variations in rainfall intensity and amount within regions, ranging from several minutes to hours, prediction becomes challenging. High-resolution NWP models with temporal-spatial scales are required to accurately predict such concentrated heavy rainfall. * denotes that probabilistic density labeling is applied.

Figure 7 and 8 show prediction outcomes for the precipitation correction. Areas marked in blue (Group 2) indicate 'rain' events in the [0.1, 10) mm range, and those marked in red (Group 3) represent 'heavy rain' events above 10 mm. We observed that the RDAPS model captures both precipitation ranges reasonably well, yet it only partially successfully predicts regions that experienced 'heavy rain.' Without the labeling, the ConvLSTM, Metnet, MAE, Swin-Unet, PostRainBench, and our model could not accurately detect both precipitation ranges. In contrast, when probabilistic density labeling was applied. The benchmark and our models could predict precipitation patterns reasonably well when RDAPS captured the location of precipitation well, as shown in Figure 7. As illustrated in Figure 8, while the benchmark models exhibited errors in targeting precipitation away from the core when RDAPS failed to capture precipitation effectively, our model demonstrated improved performance and robustness in addressing these positional errors. Through analysis, we observed that our model leverages deformable convolution by flexibly aggregating neighborhood pixels to learn representations. Using hierarchical deformable convolution layers contributes to mitigating positional errors in RDAPS, suggesting that this approach can effectively enhance prediction accuracy in such cases. Notably, in Figure 8, our model can accurately predict the narrow core of heavy rain occurring inland, closely resembling the ground truth. The results also indicate robustness against precipitation cases with high uncertainty, typical during the summer season in the Korean Peninsula.

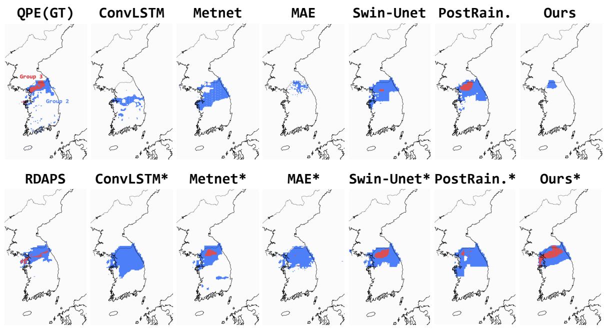

Figure 7. **Cluster 2.** Visualization result between benchmarks on August 7, 2022, at 00 UTC (+27 h). * denotes that probabilistic density labeling is applied.

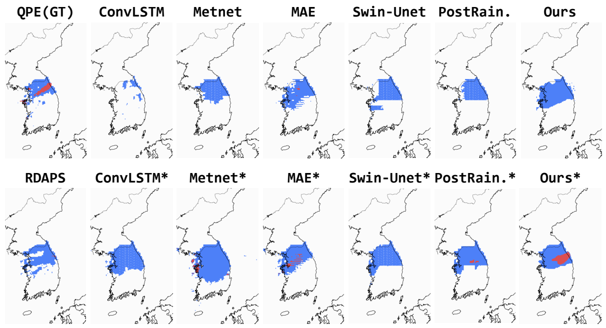

Figure 8. **Cluster 3.** Visualization result between benchmarks on August 8 2022 at 12 UTC (+29 h). * denotes that probabilistic density labeling is applied.

References

[1] Taehyeon Kim, Namgyu Ho, Donggyu Kim, and Se-Young Yun. Benchmark dataset for precipitation forecasting by post-processing the numerical weather prediction. *arXiv preprint arXiv:2206.15241*, 2022. 1, 2

[2] Chanil Park, Seok-Woo Son, Joowan Kim, Eun-Chul Chang, Jung-Hoon Kim, Enoch Jo, Dong-Hyun Cha, and Sujong Jeong. Diverse synoptic weather patterns of warm-season heavy rainfall events in south korea. *Monthly Weather Review*, 149(11):3875–3893, 2021. 4

[3] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, 28, 2015. 1

[4] Casper Kaae Sønderby, Lasse Espeholt, Jonathan Heek, Mostafa Dehghani, Avital Oliver, Tim Salimans, Shreya Agrawal, Jason Hickey, and Nal Kalchbrenner. Metnet: A neural weather model for precipitation forecasting. *arXiv preprint arXiv:2003.12140*, 2020. 2

[5] Yujin Tang, Jiaming Zhou, Xiang Pan, Zeying Gong, and Junwei Liang. Postrainbench: A comprehensive benchmark and a new model for precipitation forecasting. *arXiv preprint arXiv:2310.02676*, 2023. 2

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2

[7] Wenhai Wang, Jifeng Dai, Zhe Chen, Zhenhang Huang, Zhiqi Li, Xizhou Zhu, Xiaowei Hu, Tong Lu, Lewei Lu, Hongsheng Li, et al. Internimage: Exploring large-scale vision foundation models with deformable convolutions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14408–14419, 2023. 2