DiffCast: A Unified Framework via Residual Diffusion for Precipitation Nowcasting

Demin Yu, Xutao Li,* Yunming Ye, Baoquan Zhang, Chuyao Luo, Kuai Dai
Harbin Institute of Technology, Shenzhen
deminy@stu.hit.edu.cn, {lixutao, yeyunming, baoquanzhang}@hit.edu.cn,
luochuyao.dalian@gmail.com, daikuai_hit@163.com

Rui Wang, Xunlai Chen
Shenzhen Meteorological Bureau
cxlxun@163.com

Abstract

Precipitation nowcasting is an important spatio-temporal prediction task to predict the radar echoes sequences based on current observations, which can serve both meteorological science and smart city applications. Due to the chaotic evolution nature of the precipitation systems, it is a very challenging problem. Previous studies address the problem either from the perspectives of deterministic modeling or probabilistic modeling. However, their predictions suffer from the blurry, high-value echoes fading away and position inaccurate issues. The root reason of these issues is that the chaotic evolutionary precipitation systems are not appropriately modeled. Inspired by the nature of the systems, we propose to decompose and model them from the perspective of global deterministic motion and local stochastic variations with residual mechanism. A unified and flexible framework that can equip any type of spatio-temporal models is proposed based on residual diffusion, which effectively tackles the shortcomings of previous methods. Extensive experimental results on four publicly available radar datasets demonstrate the effectiveness and superiority of the proposed framework, compared to state-of-the-art techniques. Our code will be made publicly available soon.

1. Introduction

Precipitation nowcasting, aiming at providing a high spatio-temporal resolution rainfall prediction of very short-range (*e.g.*, 0∼6 hours), is a crucial task in meteorological science and smart city applications [26, 33]. The core of the task is to predict the future radar echoes sequence based on current
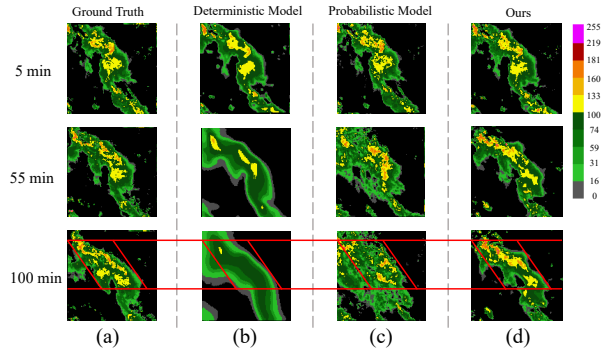
*Corresponding author

Figure 1. We evaluate different models' performances in precipitation nowcasting. The deterministic model (b) can nicely capture the moving trends but tends to get blurry appearance and underestimate the high-value echoes. The probabilistic model (c) can capture the appearance details well, but the predicted rainfall positions are inaccurate. Our method (d) is able to generate accurate prediction with nice appearance details.

observations. In essence, it is a spatio-temporal prediction problem, but with some quite distinguishing traits.

Unlike conventional video or spatio-temporal prediction tasks [9, 18], which aim to forecast a deterministic trajectory of moving objects, precipitation particles constitute a chaotic evolutionary system, governed not only by the deterministic moving trends of global system, but also by the stochastic variations (growth or decay) of local particles. More importantly, the global moving trends and local variations are inherently coupled. For example, when cold and warm air masses cross in a region, continuous rainfall is caused by an evolutionary convection with lasting growth and decay of precipitation particles, rather than a simple movement of seen rainfall belt.

However, most of previous spatio-temporal methods [3, 8–11, 26, 34] are developed for conventional tasks, which

1

fail to analyze and model this nature. As a result, their performance is unsatisfactory for precipitation nowcasting. In terms of methodology, we categorize these methods into deterministic and probabilistic models. Each type has its own disadvatanges. Deterministic models [8, 9, 27] leverage convolution recurrent neural network, transformer or their variants to forecast the future echoes sequence by directly optimizing the distance or similarity to ground-truth. This line of methods can nicely capture the moving trends, but the forecast is getting more and more blurry as the lead time increases, shown as in Figure 1(b). Moreover, the high-value echoes, which signify heavy storms, tend to be badly underestimated. The reason is that these methods overlook modelling the local stochastics. To address the blurry issue, researchers recently start to turn to probabilistic generation methods with generative adversarial networks (GANs) [4, 38] or diffusion models [10, 13]. Although the forecast images of such methods are with realistic details, the predicted rainfall positions are often very inaccurate, shown as in Figure 1(c). This is because these methods model the whole precipitation system in a stochastic manner, where the freedom of generation is too high to maintain a good prediction accuracy.

In this paper, we propose a flexible and unified end-to-end framework, called Diffcast, which nicely decomposes and models the global determinism and local stochastics of precipitation systems. In the framework, we decompose the evolving system as a motion trend and its stochastic residual, and design a motion component and a temporal residual diffusion component to model them, respectively. The motion component serves as a deterministic predictive backbone for global trajectory of precipitation system, and the carefully-designed temporal residual diffusion component accounts for modeling the residual distribution which signifies local stochastics. Specially, in the diffusion component, we design a Global Temporal UNet(GTUnet), which carefully utilizes multi-scale temporal features, including the global motion prior, sequence segment consistency and inter frame dependency, as diffusion conditions to model the temporal evolution of residual distribution. In the framework, the deterministic predictive component and local stochastic diffusion component are simultaneously trained in an end-to-end manner. Hence, they interplay with each other natrually. Our framework has three striking advantages. First, it is a robust and easy-to-train method without mode collapse compared to adversarial networks. Second, any type of deterministic spatio-temporal prediction models (reccurent-based or recurrent-free) can be easily equipped into the framework as predictive backbones. Third, the prediction performance of backbones, in terms of both accuracy metrics and appearance details, are significantly improved with our framework.

In summary, our main contributions are summarized as:

- We firstly propose to model the precipitation evolution from the perspective of global deterministic motion and local stochastic variations with residual mechanism.
- We propose a flexible precipitation nowcasting framework that can equip any deterministic backbones (*i.e.* recurrent-based and recurrent-free models) to generate accurate and realistic prediction.
- We develop a way to jointly train deterministic predictive model and probabilistic diffusion model in an end-to-end manner from the inspiration of precipitation chaotic evolution.
- We equip several notable backbones into our framework and conduct extensive experiments on four publicly available radar echo datasets, and the results show that our framework significantly improves the performance, delivering state-of-the-art results for precipitation nowcasting.

2. Related Work

2.1. Spatio-temporal Predictive Models

Deterministic predictive models are the mainstream of the existing approaches for spatio-temporal prediction [2, 23, 36]. They can be roughly categorized into two groups: recurrent-based models and recurrent-free models. The recurrent-based models learn a hidden state from the historical sequence and generate the future frames recurrently with the hidden state [3, 26, 27, 30, 35]. For example, Shi *et al.* proposed ConvLSTM [26] and ConvGRU [27] by integrating the convolution operator into the recurrent neural network. MAU [3] enhanced the video prediction by developing an attention-based predictive unit with a better temporal receptive field. PhyDnet [11] incorporated partial differential equation (PDE) constraints in the recurrent hidden state. The recurrent-free based models [23, 38] encode the given input frames into hidden states and decode all the predictive frames at once, instead of in a recurrent way. For example, SimVP [9] designed a scheme to encode and decode the information via simple convolutions. Earthformer [8] leveraged transformer to build the encoder-decoder for prediction. However, all the deterministic predictive methods suffer from the aforementioned blurry issue and high-value echoes fading away issue for precipitation nowcasting, because of the neglect of local stochastics.

Probabilistic predictive models are designed to capture the spatio-temporal uncertainty by estimating the conditional distribution of future state. These models aim to enhance the realism of predictions based on adversarial training [4, 20, 25, 32] or variational autoencoders [1, 7, 32, 38]. However, these methods directly model the whole precipitation system as stochastics and thus introduce uncontrollable randomness that harms to prediction accuracy. Moreover, these methods often suffer from mode collapse issue and

are not easy to train.

2.2. Spatio-temporal Diffusion Models

Thanks to the high-fidelity generative capabilities and stable training merit, diffusion models, garnered growing interest for spatio-temporal prediction lately [12, 15, 16, 22, 37]. The mainstream of the methods is based on denoising diffusion probabilistic models [6, 14]. For instance, MCVD [34] introduced a random mask scheme on frames and utilized conditional denoising network to learn temporal dependency between frames for prediction. LDCast [19] applied a channel-based conditional latent diffusion model to predict the evolution of precipitation with denoising mechanism. PreDiff [10] incorporated prior knowledge via extra knowledge control network to limit the generative process so as to align the prediction with domain-specific prior knowledge based on latent diffusion models. However, all the methods regard the whole system evolving stochastically and model it with a diffusion process. In this way, too much freedom is introduced for diffusion models to precisely control the generation. As a result, the predictive frames are with realistic details, but the positions tend to be mismatched.

3. Task Definition and Preliminaries

3.1. Task Definition

We follow [8, 10, 33] to formulate precipitation nowcasting as a spatio-temporal prediction problem. Given L_{in} initial frames $x = [x_i]_{i=-L_{in}}^0 \in \mathbb{R}^{L_{in} \times H \times W \times C}$, the prediction aims to model the conditional probabilistic distribution $p(y|x)$ of the following L_{out} frames $y = [y_i]_{i=1}^{L_{out}} \in \mathbb{R}^{L_{out} \times H \times W \times C}$, where H and W donate the spatial resolution of frame, and C donates the number of measurements at each space-time coordinate. In our radar echoes setting, $C = 1$ is used. Note that we denote the i-th frame, y_i, with a subscript i, and use a superscript t to indicate the t-th diffusion step state y^t in the following.

3.2. Preliminary:Diffusion

Denoising diffusion probabilistic models (DDPMs) [14, 28] aim to learn the data distribution $p(y)$ by training a model to reverse a Markov noising process that progressively corrupts the data.

Specifically, the diffusion model consists of two processes: a forward nosing process and a reverse denoising process. The former is a parameter-free process with progressively noising. We can get the t-th diffusion state from the observation y^0 with a special property:

$$q(y^t|y^0) = \mathcal{N}(y^t; \sqrt{\overline{\alpha}_t}y^0, (1-\overline{\alpha}_t)I), \qquad (1)$$

where $y^T \sim \mathcal{N}(0,1)$ denotes a sample from a pure Gaussian distribution and y^0 is a sample from target distribution. The coefficients are defined as $\overline{\alpha}_t = \prod_{t=1}^T \alpha_t, \alpha_t = 1 - \beta_t$,

where $\beta_t \in (0,1)$ is predefined by an incremental variance schedule. As for the reverse denoising process, it is defined by the following Markov chain with parameterized Gaussian transitions:

$$p_\theta(y^{t-1}|y^t) = \mathcal{N}(y^{t-1}; \mu_\theta(y^t, t), \sigma_t^2 I), \qquad (2)$$

where μ_θ donates the posterior mean function. The reverse function $p_\theta(y^{t-1}|y^t)$ aims to remove the noise added in the forward noising process and we follow [14] to set the parameterization:

$$\mu_\theta(y^t, t) = \frac{1}{\sqrt{\alpha_t}}(y^t - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(y^t, t)), \qquad (3)$$

where $\epsilon_\theta(y^t, t)$ is a trainable denoising function that estimates the corresponding noise conditioned on the current step t. The parameters θ can be optimized by the following loss:

$$L(\theta) = \mathbb{E}_{y^0, t, \epsilon}\|\epsilon - \epsilon_\theta(y^t, t)\|^2, \qquad (4)$$

where $y^t = \sqrt{\overline{\alpha}_t}y^0 + \sqrt{1-\overline{\alpha}_t}\epsilon$. After training, we can recover y^{t-1} from y^t with

$$y^{t-1} = \frac{1}{\sqrt{\alpha_t}}(y^t - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(y^t, t)) + \sigma_t\epsilon, \qquad (5)$$

where σ_t is a variance hyperparameter. Finally, by drawing a sample from the prior distribution $p(y^T)$ and iteratively applying Equation (5), a sample from the target distribution $p(y^0)$ can be derived.

4. Overall Framework

Our framework, DiffCast, is designed to decompose the precipitation systems as the global motion trend and local stochastic residual, and utilizes a deterministic component and stochastic diffusion component to model them, respectively. The overview of our framework is illustrated in Figure 2 (a). Specifically, we apply a deterministic backbone as base predictor $\mathcal{P}_{\theta_1}$ to capture the global motion trajectory denoted as μ. Note that any type of deterministic prediction models, *e.g.* SimVP [24], Earthformer [8], ConvGRU [27] etc., can be equipped into our framework without additional adaptation (Section 4.1). Then, the deterministic motion prior μ is leveraged for producing a residual sequence r with ground truth y to represent the local stochastics. We propose an auxiliary stochastic diffusion component to model the evolution of the residual distribution (Section 4.2). In the component, a sophisticated Global Temporal UNet(GTUnet) is designed to exploit the global motion prior, sequence segment consistency and inter-frame dependency for the evolution modelling (Section 4.3). Finally, the probabilistic predictive residual $\hat{r}$ and the deterministic prediction μ are added to form the final prediction $\hat{y}$.
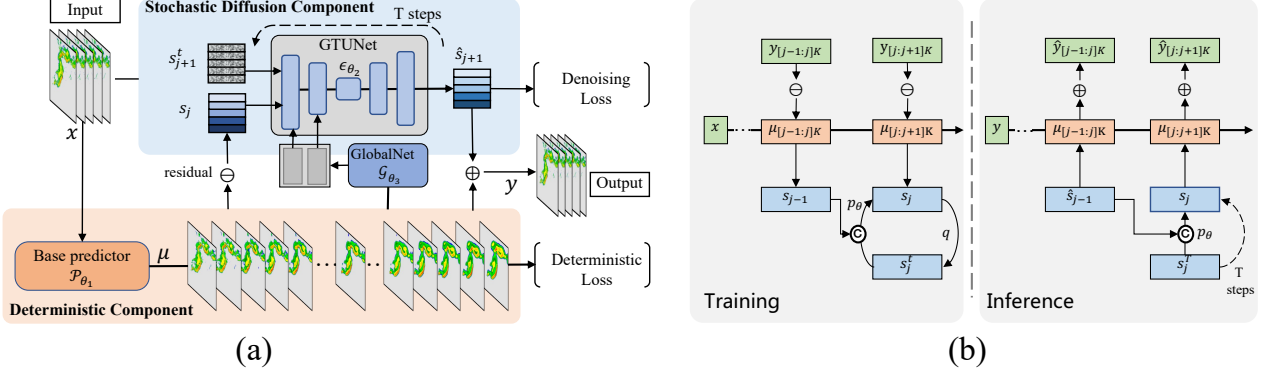
Figure 2. The overview of our *DiffCast* framework for precipitation nowcasting is shown in (a). The DiffCast models the precipitation process from two perspective: deterministic component and stochastic component. The former accounts for predicting a global motion trend by a coarse forecast, while the latter aims to incorporate stochasticity with auxiliary-conditioned diffusion into the coarse forecast by residual. The sub-figure (b) indicates the computing flow of our framework for training and inference, respectively. The green, orange and blue rectangles represent, respectively, radar echos segment, output of deterministic predictor and residual segment for diffusion model.

4.1. Deterministic Predictive Backbone

As a generic module, the deterministic component allows any prediction models of recurrent-based and recurrent-free architectures to be the base predictor $\mathcal{P}_{\theta_1}(\cdot)$. We denote the output of the base predictor as $\mu = [\mu_i]_{i=1}^{L_{out}} \in \mathbb{R}^{L_{out} \times H \times W \times C}$. The deterministic predictive backbone learns to model $p_{\theta_1}(\mu|x)$ by using the given input frames x with any pixel-wise lose, *e.g.* mean-squared error (MSE):

$$\mathcal{L}_{\mathcal{P}} = \mathbb{E}[||\mu - y||^2], \qquad (6)$$

or other natively designed loss function for the backbone. Here θ_1 means the parameters of this part. We term the loss as deterministic loss in our framework. Though the prediction μ in this way still suffers from the blurry and high-value echoes fading away issues, it is able to capture the global motion trend, which will help us to decompose the local stochastic residual from the precipitation systems next. Also, μ will provide necessary information for diffusion component to model the evolution of residual later in Section 4.3.

4.2. Stochastic Residual Prediction

As aforementioned, precipitation systems evolve with a global motion trend and local stochastics. The predictive backbone has offered us a global motion trend with μ. Next, we introduce how to model the local stochastics. Our notion is to compute the residual r between the ground-truth y and μ to represent the local stochastics:

$$r = y - \mu. \qquad (7)$$

However, the computation of r here involves the ground-truth, which thus cannot be directly used in the prediction. Our idea is using r as supervised information to train a diffusion model to predict the residual evolution. Since r denotes a sequence from 1 to L_{out}, we model its evolution

in an autoregressive manner. In other words, the diffusion model needs to establish the following distribution:

$$p_{\theta_2}(r_i|\hat{r}_{i-1}), \qquad (8)$$

where $\hat{r}_{i-1}$ indicates the predictive residual of the $(i-1)$-th frame, and θ_2 denotes the parameters of the diffusion model. Suppose that we use a T-step denoising diffusion to model the distribution, following Equation (2) we have:

$$p_{\theta_2}(r_i|\hat{r}_{i-1}) = p(r^T)\prod_{t=1}^{T} p_{\theta_2}(r_i^{t-1}|r_i^t, \hat{r}_{i-1}), \qquad (9)$$

where $r^T \sim \mathcal{N}(0, I)$ and t is the denoising step; and r_i^t represents the t-th denoising state for the i-th frame residual. Similar to Equation (3), in the denoising process, learning to recover the residual state r_i^{t-1} from state r_i^t is equivalent to estimating the corresponding noise ϵ added in the t-th step corruption. Hence, we can use noise estimation as optimization objective. Suppose that each step of the diffusion process shares the same denoiser function $\epsilon_{\theta_2}(r_i^t, \hat{r}_{i-1}, t)$, which takes the previous diffusion state r_i^t and the most recent predictive residual $\hat{r}_{i-1}$ as input. We thus have the following objective function:

$$\mathcal{L}_{\epsilon} = \mathbb{E}_{(r_i, r_{i-1})\sim r, t, \epsilon\sim\mathcal{N}(0,I)}||\epsilon - \epsilon_{\theta_2}(r_i^t, \hat{r}_{i-1}, t)||^2, \quad (10)$$

where the t-th denoising state r_i^t can be computed as:

$$r_i^t = \sqrt{\overline{\alpha}_t}\underbrace{(y - \mathcal{P}_{\theta_1}(x))_i}_{residual} + \sqrt{1 - \overline{\alpha}_t}\epsilon. \qquad (11)$$

We term the objective in Equation (10) as denoising loss. To capture the interplay between the deterministic predictive backbone and the stochastic residual prediction, we train our framework in an end-to-end manner with the following
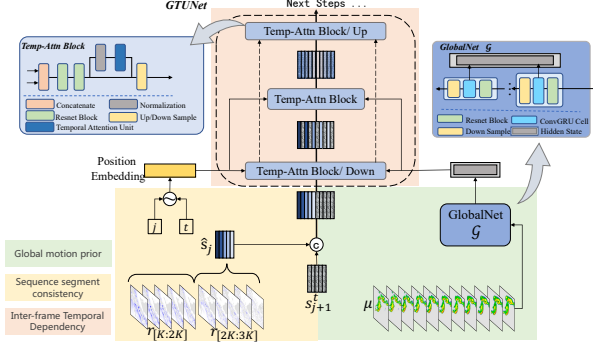
4

Figure 3. Illustration of our Global Temporal Unet (*i.e.*, GTUnet).

combined loss function:

$$\mathcal{L} = \alpha \sum_{r_i \in r} \mathcal{L}_\epsilon(r_i) + (1 - \alpha)\mathcal{L}_\mathcal{P}, \quad (12)$$

where $\alpha \in [0, 1]$ is a weight factor to balance two losses. Once well trained, we can predict the residual r_i by the previous residual $\hat{r}_{i-1}$ with iteratively denoising from a Gaussian noise sample by Equation (5). Repeating this L_{out} times leads us to an estimated residual sequence $\hat{r} = [\hat{r}_i]_{i=1}^{L_{out}}$. Note that $\hat{r}_0 = 0$ in the setting. Finally, we compute the final prediction result $\hat{y}$ as:

$$\hat{y} = \hat{r} + \mu. \quad (13)$$

4.3. Global Temporal UNet (GTUNet)

In this subsection, we introduce the detailed diffusion component for the stochastic residual prediction. In the diffusion component we design a Global Temporal UNet, which effectively exploits multi-scale temporal features, namely global motion prior, sequence segment consistency, and inter-frame temporal for the residual evolution prediction. Figure 3 illustrates the details of the diffusion model blocks. Next, we elaborate the key blocks, respectively.

Global motion prior refers to the global motion trend information derived from the base deterministic prediction μ. As shown in green part of Figure 3, we design a ConvRNN-liked structure, GlobalNet $\mathcal{G}_{\theta_3}$ to extract the global motion information from the deterministic prediction as:

$$h = \mathcal{G}_{\theta_3}(\mathcal{P}_{\theta_1}(x)). \quad (14)$$

Here θ_3 represents the parameters of the GlobalNet. As shown in dark blue part of Figure 3, GlobalNet is a multi-layer architecture with multiple temporal blocks. Each temporal block is composed of a down sample operator, a ConvGRU operator and a Resnet operator. By incorporating the derived hidden state h as an extra condition into diffusion model, the residual prediction r is re-expressed as:

$$r = [p_{\theta_2}(r_i|\hat{r}_{i-1}, h)]_{i=1}^{L_{out}}, \quad (15)$$

and the denoiser function is changed into:

$$\epsilon_{\theta_2}(r_i^t, \hat{r}_{i-1}, \mathcal{G}_{\theta_3}(\mathcal{P}_{\theta_1}(x)), t). \quad (16)$$

The objective function in Equation (12) is then parameterized by $(\theta_1, \theta_2, \theta_3)$.

Sequence segment consistency. To better maintain the sequence consistency of residual evolution, we propose to partition the residual sequence r into multiple segments s for prediction, because recent studies [23] show that multi-input multi-output is a better paradigm than single-in single-out in recurrent spatio-temporal prediction. As shown in yellow part of Figure 3, we construct the segment and denote the j-th segment as:

$$s_{j-1} = r_{[(j-1)K:jK]}, \quad (17)$$

where K indicates the length of each segment and $j = 0, 1, ..., \lceil \frac{L_{out}}{K} \rceil$. Then we have $s_j \in \mathbb{R}^{K \times H \times W \times C}$. In this way, the diffusion is changed to model the segment-level temporal distribution as:

$$s = [p_{\theta_2}(s_j|\hat{s}_{j-1}, h)]_{j=1}^{\lceil \frac{L_{out}}{K} \rceil}. \quad (18)$$

Specifically, we incorporate the segment condition with channel concatenation between t-th denoising state s_j^t and previous segment s_{j-1}. Moreover, as the forecast lead time increases, the residual inevitably becomes larger because of the increased uncertainty. Hence, it is important to explicitly indicate the position of a segment. To this end, we add an extra position embedding on segment index j, in addition to the denoising step t. With the above changes, the objective of diffusion model for residual evolution becomes:

$$\mathcal{L}_\epsilon = \mathbb{E}||\epsilon - \epsilon_{\theta_2}(s_j^t, \hat{s}_{j-1}, \mathcal{G}_{\theta_3}(\mathcal{P}_{\theta_1}(x)), t, j)||^2. \quad (19)$$

Note that when predicting the first residual segment s_1, we use $s_0 = 0$ following [34].

Inter-frame temporal dependency. To better model the inter-frame dependency within a segment, our Global Temporal Unet is carefully designed with temporal attention blocks, as shown in orange part of Figure 3. It is indeed a variant of UNet in DDPM with temporal evolution. The temporal attention block, as shown in grey blue part of Figure 3, is constructed by concatenate operator, Resnet operator, normalization operator and temporal attention unit operator, followed by an up/downsample operator, which makes the predicted residual in the segment become temporal dependent. With the developed temporal evolution UNet structure, the inter-frame dependency can be effectively taken into account.

4.4. Training and Inference

For clarity, we summarize the computing flow of training and inference of our framework in Figure 2 (b). In the training phase, the deterministic predictor $\mathcal{P}_{\theta_1}$ first produces

μ. Then the residual r is computed as Equation (7) and grouped into segments s by Equation (17). Given a segment s_{j-1}, we predict the segment s_j by our diffusion model p_{θ_2}. Here the diffusion model is updated conventionally, namely sampling a step t, adding noise as scheduled, using the denoiser to predict noise and calculating loss as Equation (19) to update parameters $(\theta_1, \theta_2, \theta_3)$. As for inference, the computing flow is similar with the only difference of diffusion part, where we first sample the T-state segment s_j^T from $\mathcal{N}(0, I)$ and perform T-step denoising with ϵ_θ to recover $\hat{s}_{j-1}$. Once all required residual segments are obtained we can compute the final prediction as Equation (13).

5. Experiments

5.1. Experimental Setting

Dataset. The **SEVIR** [33], as a widely used dataset for precipitation nowcasting, contains 20,393 weather events of radar frame sequence with a length of 4-hour and size of 384 km×384 km, where every pixel stands for 1km×1km and the temporal resolution is 5 minutes. The **MeteoNet** [17] covers a large area of 550 km×550 km in France, and records over three years observations with temporal resolution of 6 minutes. The **Shanghai_Radar** [5] is generated by volume scans in intervals of approximately 6 minutes from 2015 to 2018 in Pudong, Shanghai with a spatial size of 501 km ×501 km. The **CIKM**[1] records precipitation events in 101 km×101 km area of Guangdong, China. Each sequence settles 15 radar echo maps as a sample and the temporal resolution is 6 minutes. More details of these datasets are provided in Appendix.

Data preprocess. As for all prediction sequences, we mainly focus on modeling the precipitation event. Hence, following [5], we separate the continuous sequence into multiple events for MeteoNet and Shanghai-Radar corpus. As mentioned in [38] that the increasing length of initial frames L_{in} cannot provide substantial improvements for forecast, we set the challenging prediction task to predict 20 frames given 5 initial frames (*i.e.* $5 \to 20$) except for the CIKM dataset, where only $5 \to 10$ can be used due to its sequence length limitation. For all the datasets, we keep the original temporal resolution but downscale the spatial size to 128×128, due to the limitation of our computation resource.

Evaluation. To evaluate the accuracy of nowcasting, we calculate the average Critical Success Index (CSI) and Heidke Skill Score (HSS) following [10, 21, 33] at different thresholds (the detailed way to compute CSI and HSS can be found at appendix). The CSI, similar to IoU, is to measure the degree of pixel-wise matching between predictions and ground truth after thresholding them into 0/1 matrices. Following [8, 10], we also report the CSIs at different pool-

---

[1]https://tianchi.aliyun.com/dataset/1085

ing scales, which relax the pixel-wise matching to evaluate the accuracy on neighborhood aggregations. Additionally, LPIPS and SSIM are also utilized to measure the visual quality of prediction.

Training details. We train our DiffCast framework for 200K iterations using Adam optimizer with a learning rate of 0.0001. As for diffusion setting, we follow the standard setting of diffusion model in [14] to set the diffusion steps as 1000 and denoising steps for inference as 250 with DDIM [29]. We set default loss weight factor $\alpha = 0.5$ to balance the deterministic loss and denoisng loss in Equation (12). As for baseline methods, their configurations are tuned correspondingly for different datasets. All experiments run on a computer with a single A6000 GPU.

5.2. Experimental Results

As mentioned in Section 4.1 that our DiffCast can flexibly utilize various type of deterministic models as a base predictor, we select some notable deterministic models as our comparison baselines. Among them, the SimVP [9] and Earthformer [8] apply recurrent-free strategy to generate all frames at once, while the MAU [3], ConvGRU [27] and PhyDnet [11] are designed with recurrent strategy to generate frames one by one. Moreover, we also utilize two diffusion-based approaches [10, 34] and a GAN-based model [4] for comparison. We evaluate all this deterministic models and stochastic generative models, as well as our framework equipped with every deterministic model on four real-world precipitation datasets. The experimental results are shown in Table 1.

From the results of Table 1, we make the following observations: (i) Equipped into our DiffCast framework, the performances of backbones are significantly improved and the improvements are from 2% to 20%, in terms of CSI and HSS, and more improvements can be observed for pooling CSI. This verifies the effectiveness of our framework to boost the prediction accuracy of backbones. (ii) In terms of LPIPS and SSIM, which measures the visual quality of predictions, our DiffCast framework also makes a significant improvement. Especially for LPIPS, the improvements are $16.9\% \sim 66.7\%$. This suggests that our framework indeed improves the visual quality of backbone predictors. (iii) Compared to the state-of-the-art GAN and diffusion baselines, namely MCVD, PreDiff and STRPM, the proposed framework also performs better. The observation validates that modeling the precipitation system with a global trend and local stochastics is better than modeling the whole system as stochastics.

To investigate how the prediction performance changes against the lead time, we depict the curves in Figure 4. We observe that as the lead time increases, the performance of all method decreases, because the uncertainty for prediction is enlarged. However, the performance of methods with our

Table 1. Experiment results on four radar datasets. Relative improvements are shown with brackets.

Method	SEVIR						MeteoNet					
	↑CSI	↑CSI-pool4	↑CSI-pool16	↑HSS	↓LPIPS	↑SSIM	↑CSI	↑CSI-pool4	↑CSI-pool16	↑HSS	↓LPIPS	↑SSIM
SimVP[9]	0.2662	0.2844	0.3452	0.3369	0.3914	0.6304	0.3346	0.3383	0.4143	0.4568	0.3523	0.7557
DiffCast_SimVP	**0.3077**	**0.4122**	**0.5683**	**0.4033**	**0.1812**	**0.6354**	**0.3511**	**0.5081**	**0.7155**	**0.4846**	**0.1198**	**0.7887**
	(+15.59%)	(+44.94%)	(+64.63%)	(+19.71%)	(+53.70%)	(+0.79%)	(+4.93%)	(+50.19%)	(+72.70%)	(+6.09%)	(+65.99%)	(+4.37%)
Earthformer[8]	0.2513	0.2617	0.2910	0.3073	0.4140	**0.6773**	0.3296	0.3428	0.4333	0.4604	0.3718	0.7899
DiffCast_Earthformer	**0.2823**	**0.3868**	**0.5362**	**0.3623**	**0.1818**	0.6420	**0.3402**	**0.5020**	**0.7092**	**0.4696**	**0.1236**	**0.7967**
	(+12.34%)	(+47.80%)	(+84.26%)	(+17.90%)	(+56.09%)	(-5.21%)	(+3.22%)	(+46.44%)	(+63.67%)	(+2.00%)	(+66.76%)	(+0.86%)
MAU[3]	0.2463	0.2566	0.2861	0.3004	0.3933	0.6361	0.3232	0.3304	0.4165	0.4451	0.3089	0.7897
DiffCast_MAU	**0.2716**	**0.3789**	**0.5414**	**0.3506**	**0.1874**	**0.6729**	**0.3490**	**0.503**	**0.7114**	**0.4822**	**0.1213**	0.7665
	(+10.27%)	(+47.66%)	(+89.23%)	(+16.71%)	(+52.35%)	(+5.79%)	(+7.98%)	(+52.24%)	(+70.80%)	(+8.34%)	(+60.73%)	(-2.94%)
ConvGRU[27]	0.2416	0.2554	0.3050	0.2834	0.3766	**0.6532**	0.3400	0.3578	0.4473	0.4667	0.2950	**0.7832**
DiffCast_ConvGRU	**0.2772**	**0.3809**	**0.5463**	**0.3551**	**0.1880**	0.6188	**0.3512**	**0.4930**	**0.7001**	**0.4862**	**0.1244**	0.7761
	(+14.74%)	(+49.14%)	(+79.11%)	(+25.30%)	(+50.08%)	(-5.27%)	(+3.29%)	(+37.79%)	(+56.52%)	(+4.18%)	(+57.83%)	(-0.91%)
PhyDnet[11]	0.2560	0.2685	0.3005	0.3124	0.3785	**0.6764**	0.3384	0.3824	0.4986	0.4673	0.2941	**0.8022**
DiffCast_PhyDnet	**0.2757**	**0.3797**	**0.5296**	**0.3584**	**0.1845**	0.6320	**0.3472**	**0.5066**	**0.7200**	**0.4802**	**0.1234**	0.7788
	(+7.70%)	(+41.42%)	(+76.24%)	(+14.72%)	(+51.2%)	(-6.56%)	(+2.60%)	(+32.48%)	(+44.40%)	(+2.76%)	(+58.04%)	(-2.92%)
MCVD[34]	0.2148	0.3020	0.4706	0.2743	0.217	0.5265	0.2336	0.3841	0.6128	0.3393	0.1652	0.5414
PreDiff[10]	0.2304	0.3041	0.4028	0.2986	0.2851	0.5185	0.2657	0.3854	0.5692	0.3782	0.1543	0.7059
STRPM[4]	0.2512	0.3243	0.4959	0.3277	0.2577	0.6513	0.2606	0.4138	0.6882	0.3688	0.2004	0.5996

Method	Shanghai_Radar						CIKM					
	↑CSI	↑CSI-pool4	↑CSI-pool16	↑HSS	↓LPIPS	↑SSIM	↑CSI	↑CSI-pool4	↑CSI-pool16	↑HSS	↓LPIPS	↑SSIM
SimVP[9]	0.3841	0.4467	0.5603	0.5183	0.2984	0.7764	**0.3021**	0.3530	0.4677	**0.3948**	0.3134	0.6324
DiffCast_SimVP	**0.3955**	**0.5116**	**0.6576**	**0.5296**	**0.1571**	**0.7902**	0.2999	**0.3657**	**0.5260**	0.3874	**0.2223**	**0.6391**
	(+2.97%)	(+14.53%)	(+17.37%)	(+2.18%)	(+47.35%)	(+1.78%)	(-0.73%)	(+3.60%)	(+12.47%)	(-1.87%)	(+29.07%)	(+1.06%)
Earthformer[8]	0.3575	0.4008	0.4863	0.4843	0.2564	0.7750	**0.3153**	0.3547	0.4927	0.3828	0.3857	**0.6510**
DiffCast_Earthformer	**0.3751**	**0.4855**	**0.6212**	**0.5069**	**0.1586**	**0.7851**	0.3099	**0.3807**	**0.5509**	**0.3947**	**0.2259**	0.6313
	(+4.92%)	(+21.13%)	(+27.74%)	(+4.67%)	(+38.14%)	(+1.30%)	(-1.71%)	(+7.33%)	(+11.81%)	(+3.11%)	(+41.43%)	(-3.03%)
MAU[3]	0.3996	0.4695	0.5787	0.5356	0.2735	0.7303	0.2936	0.3152	0.4144	0.3660	0.3999	0.6277
DiffCast_MAU	**0.4089**	**0.5212**	**0.6658**	**0.5475**	**0.1618**	**0.7879**	**0.3158**	**0.3803**	**0.5443**	**0.4085**	**0.2205**	**0.6498**
	(+2.33%)	(+11.01%)	(+15.05%)	(+2.22%)	(+40.84%)	(+7.89%)	(+7.56%)	(+20.65%)	(+31.35%)	(+11.61%)	(+44.86%)	(+3.52%)
ConvGRU[27]	0.3612	0.4439	0.5596	0.4899	0.2564	0.7795	0.3092	0.3533	0.4686	**0.4007**	0.3135	**0.6601**
DiffCast_ConvGRU	**0.3738**	**0.4923**	**0.6596**	**0.4945**	**0.1563**	**0.7809**	**0.3143**	**0.3681**	**0.5117**	0.3967	**0.2201**	0.6418
	(+3.49%)	(+10.90%)	(+17.87%)	(+0.94%)	(+39.04%)	(+0.18%)	(+1.65%)	(+4.19%)	(+9.20%)	(-1.00%)	(+29.79%)	(+2.77%)
PhyDnet[11]	0.3653	0.4552	0.5980	0.4957	0.1894	0.7751	0.3037	0.3442	0.4655	0.3931	0.3631	**0.6540**
DiffCast_PhyDnet	**0.3671**	**0.4907**	**0.6493**	**0.4986**	**0.1574**	**0.7780**	**0.3131**	**0.3836**	**0.5550**	**0.3990**	**0.2270**	0.6156
	(+0.49%)	(+7.80%)	(+8.58%)	(+0.59%)	(+16.90%)	(+0.37%)	(+3.10%)	(+11.45%)	(+19.23%)	(+1.50%)	(+37.48%)	(-5.87%)
MCVD[34]	0.2872	0.3984	0.5675	0.4036	0.2081	0.5119	0.2513	0.3095	0.4955	0.3294	0.2528	0.5358
PreDiff[10]	0.3583	0.4389	0.5448	0.4849	0.1696	0.7557	0.3043	0.3681	0.5117	0.3967	0.2201	0.6418
STRPM[4]	0.3606	0.4944	0.6783	0.4931	0.1681	0.7724	0.2984	0.3590	0.5020	0.3870	0.2397	0.6443

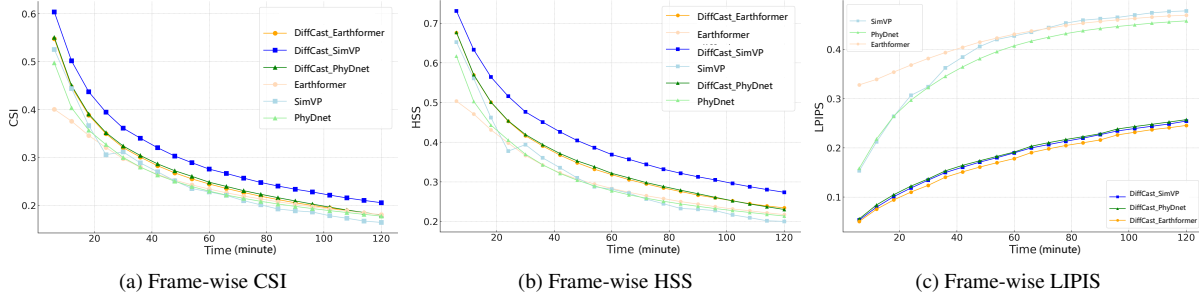(a) Frame-wise CSI   (b) Frame-wise HSS   (c) Frame-wise LIPIS

Figure 4. Performance changes against different lead time in terms of CSI, HSS and LPIPS. For a better vision, we only show the curves of SimVP, Earthformer and PhyDnet with or without our framework.

framework is always better than that without it. This again validates the effectiveness of the proposed framework.

In Figure 5, we show and compare the results of all the methods for a precipitation event. We observe that the prediction of SimVP is blurry, as it does not model the local stochastics. MCVD, Prediff and STRPM deliver better visual details than SimVP, but the positions of green and yellow parts are less accurate. This is because these methods model the intact system in a stochastic generation way, where the freedom of generation is too high to maintain the accuracy. When equipped SimVP into the proposed DiffCast framework, not only realistic appearance details are produced, but also the positions are very accurate. At the

Table 2. Analysis of the necessity of deterministic loss.

Method	CSI	CSI-pool4	CSI-pool16	HSS	LPIPS	SSIM
DiffCast_SimVP($\alpha=0$)	0.2578	0.2726	0.3049	0.3154	0.3849	0.6570
DiffCast_Simvp($\alpha=0.5$)	**0.3077**	**0.4122**	**0.5683**	**0.4033**	**0.1812**	0.6354
DiffCast_Simvp($\alpha=1$)	0.2430	0.3065	0.3999	0.2989	0.1831	**0.6824**

100 minute prediction, we can see that DiffCast_SimVP accurately predicts two closing rain belts of yellow, but none of the comparison methods achieves this. The observations again validate the superiority of our framework.
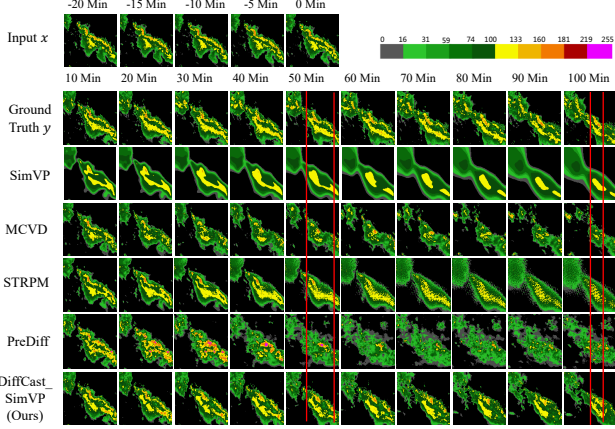
Figure 5. A visual comparison example on a precipitation event from SEVIR. The results of Earthformer, MAE, ConvGRU are similar, which is put into Appendix due to the space limitation.

5.3. Analysis and Discussions

Why is the deterministic loss necessary? Our framework has two loss functions, namely the deterministic loss and denoising loss. We can see from Equation (19) that the denoising loss is related to all the parameters θ_1, θ_2 and θ_3. Hence, by optimizing the denoising loss, we can also update the backbone predictor. Then, one would like to ask whether the deterministic loss is really necessary.

To investigate the question, we evaluate DiffCast_SimVP with different weight factor α in Equation (12) and show the results in Table 2. Note that DiffCast_SimVP degenerates into SimVP when $\alpha = 0$. We observe that SimVP even performs better than DiffCast_SimVP without deterministic loss, i.e., $\alpha = 1$. To further understand the reason, we use an example to depict the prediction, μ, positive and negative residuals r with and without deterministic loss $\mathcal{L}_p$ in Figure 6. We find that when both losses are used (*i.e.*, $\alpha = 0.5$), the prediction is accurate and with realistic details. In this case the μ indeed generates a global trend without details, and residual parts account for making up the details. However, if we remove the deterministic loss (*i.e.*, $\alpha = 1$), the prediction is with realistic details but not accurate. Moreover, in this case the μ plays a very minor role while residual becomes the main component. This is easy to understand. When the deterministic loss is removed (*i.e.*, $\alpha = 1$), our DiffCast cannot nicely decompose the precipitation system into a global trend and local stochastics, which can be validated by Figure 7a where the deterministic loss never decreases. Instead, in this case the DiffCase degenerates into a model that regards the whole system as stochastics. Hence, its prediction is with realistic details but not accurate. Moreover, we can see from Figure 7b that the deterministic loss enhances both the convergence speed and convergence quality for our DiffCast framework.
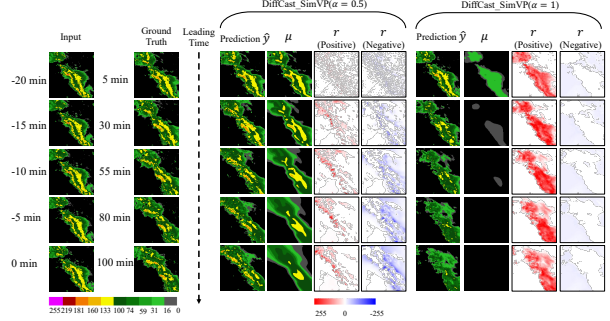

Figure 6. An illustration example of the prediction $\hat{y}$, μ and residual r with or without deterministic loss for DiffCast_SimVP.
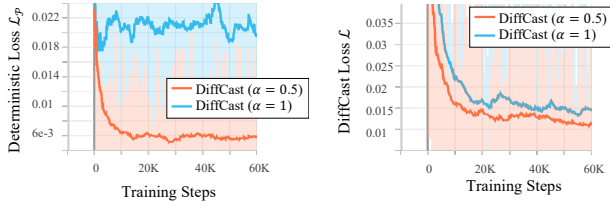

(a) Deterministic Loss $\mathcal{L}_\mathcal{P}$ (b) Overall Loss $\mathcal{L}$

Figure 7. Analysis of loss trends.

Table 3. Analysis of our end-to-end training on SEVIR

Method	CSI	CSI-pool4	CSI-pool16	HSS	LPIPS	SSIM
DiffCast_SimVP	**0.3077**	**0.4122**	**0.5683**	**0.4033**	0.1812	**0.6354**
DiffCast_simvp-frozen	0.2739	0.3807	0.5417	0.3563	0.1948	0.6315
DiffCast_Earthformer	**0.2823**	**0.3868**	**0.5362**	**0.3623**	0.1818	**0.6420**
DiffCast_Earthformer-frozen	0.2622	0.3676	0.5183	0.3400	0.1776	0.6341
DiffCast_PhyDnet	**0.2757**	**0.3797**	**0.5296**	**0.3584**	0.1845	**0.6320**
DiffCast_PhyDnet-frozen)	0.2603	0.3715	0.5469	0.3360	0.1939	0.6230

Table 4. The ablation of DiffCast with respect to GlobalNet $\mathcal{G}_{\theta_3}$ on the SEVIR dataset.

Method	CSI	CSI-pool4	CSI-pool16	HSS	LPIPS	SSIM
DiffCast_SimVP	**0.3077**	**0.4122**	**0.5683**	**0.4033**	0.1812	**0.6354**
DiffCast_SimVP w/o $\mathcal{G}$	0.2719	0.3729	0.5471	0.3522	0.2135	0.6315
DiffCast_Earthformer	**0.2823**	**0.3868**	**0.5362**	**0.3623**	0.1818	**0.6420**
DiffCast_Earthformer w/o $\mathcal{G}$	0.2558	0.3442	0.4811	0.3298	0.1918	0.6018
DiffCast_PhyDnet	**0.2757**	**0.3797**	**0.5296**	**0.3584**	0.1845	**0.632**
DiffCast_PhyDnet w/o $\mathcal{G}$	0.2648	0.3641	0.5049	0.3411	0.1760	0.6568

Why do we apply end-to-end training rather than a two-stage training? Our DiffCast framework is trained in an end-to-end manner with both the deterministic and stochastic components. One interesting question is whether it is really better than a two-stage strategy. In two-stage strategy, we first train a deterministic backbone, then equip it into our framework and keep it frozen. In this case, our framework degenerates into a Predictor-Corrector paradigm, where the diffusion model works as a corrector to the prediction of backbone.

Table 3 shows the results. We find that the end-to-end way is absolutely better than the two-stage manner. This is because when trained in end-to-end manner, the deterministic and stochastic component can interplay with other. For example, the denoising loss of stochastic part will also produce gradients for the update of deterministic backbone. As a result, the backbone may become better.

Is our GlobalNet $\mathcal{G}_\theta$ **effective?** We show the ablation study on the designed GlobalNet in Table 4. We can see that removing GlobalNet degrades the performance, which indicates its effectiveness.

6. Conclusion

In this paper, we propose an unified and flexible framework for precipitation nowcasting based on residual diffusion model, which nicely decomposes and models the system evolution with a global deterministic trend component and a local stochastic component. Extensive experiments on four real-world datasets verifies the effectiveness of the proposed framework.

References

[1] Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy Campbell, and Sergey Levine. Stochastic variational video prediction. In *International Conference on Learning Representations*, 2018.

[2] Cong Bai, Feng Sun, Jinglin Zhang, Yi Song, and Shengyong Chen. Rainformer: Features extraction balanced network for radar-based precipitation nowcasting. *IEEE Geoscience and Remote Sensing Letters*, 19:1–5, 2022.

[3] Zheng Chang, Xinfeng Zhang, Shanshe Wang, Siwei Ma, Yan Ye, Xiang Xinguang, and Wen Gao. Mau: A motion-aware unit for video prediction and beyond. *Advances in Neural Information Processing Systems*, 34:26950–26962, 2021.

[4] Zheng Chang, Xinfeng Zhang, Shanshe Wang, Siwei Ma, and Wen Gao. Strpm: A spatiotemporal residual predictive model for high-resolution video prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13946–13955, 2022.

[5] Lei Chen, Yuan Cao, Leiming Ma, and Junping Zhang. A deep learning-based methodology for precipitation nowcasting with radar. *Earth and Space Science*, 7(2): e2019EA000812, 2020.

[6] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[7] Jean-Yves Franceschi, Edouard Delasalles, Mickaël Chen, Sylvain Lamprier, and Patrick Gallinari. Stochastic latent residual video prediction. In *International Conference on Machine Learning*, pages 3233–3246. PMLR, 2020.

[8] Zhihan Gao, Xingjian Shi, Hao Wang, Yi Zhu, Yuyang Bernie Wang, Mu Li, and Dit-Yan Yeung. Earthformer: Exploring space-time transformers for earth system forecasting. *Advances in Neural Information Processing Systems*, 35:25390–25403, 2022.

[9] Zhangyang Gao, Cheng Tan, Lirong Wu, and Stan Z Li. Simvp: Simpler yet better video prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3170–3180, 2022.

[10] Zhihan Gao, Xingjian Shi, Boran Han, Hao Wang, Xiaoyong Jin, Danielle Maddix, Yi Zhu, Mu Li, and Yuyang Wang.

Prediff: Precipitation nowcasting with latent diffusion models. *arXiv preprint arXiv:2307.10422*, 2023.

[11] Vincent Le Guen and Nicolas Thome. Disentangling physical dynamics from unknown factors for unsupervised video prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11474–11484, 2020.

[12] William Harvey, Saeid Naderiparizi, Vaden Masrani, Christian Weilbach, and Frank Wood. Flexible diffusion modeling of long videos. *Advances in Neural Information Processing Systems*, 35:27953–27965, 2022.

[13] Yusuke Hatanaka, Yannik Glaser, Geoff Galgon, Giuseppe Torri, and Peter Sadowski. Diffusion models for high-resolution solar forecasts. *arXiv preprint arXiv:2302.00170*, 2023.

[14] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[15] Tobias Höppe, Arash Mehrjou, Stefan Bauer, Didrik Nielsen, and Andrea Dittadi. Diffusion models for video prediction and infilling. *Transactions on Machine Learning Research*, 2022.

[16] Chiyu Jiang, Andre Cornman, Cheolho Park, Benjamin Sapp, Yin Zhou, Dragomir Anguelov, et al. Motiondiffuser: Controllable multi-agent motion prediction using diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9644–9653, 2023.

[17] Gwennaëlle Larvor, Léa Berthomier, Vincent Chabot, Brice Le Pape, Bruno Pradel, and Lior Perez. Meteonet, an open reference weather dataset, 2020.

[18] Alex X Lee, Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine. Stochastic adversarial video prediction. *arXiv preprint arXiv:1804.01523*, 2018.

[19] Jussi Leinonen, Ulrich Hamann, Daniele Nerini, Urs Germann, and Gabriele Franch. Latent diffusion models for generative precipitation nowcasting with accurate uncertainty quantification. *arXiv preprint arXiv:2304.12891*, 2023.

[20] Chuyao Luo, Xutao Li, Yunming Ye, Shanshan Feng, and Michael K Ng. Experimental study on generative adversarial network for precipitation nowcasting. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–20, 2022.

[21] Chuyao Luo, Guangning Xu, Xutao Li, and Yunming Ye. The reconstitution predictive network for precipitation nowcasting. *Neurocomputing*, 507:1–15, 2022.

[22] Kangfu Mei and Vishal Patel. Vidm: Video implicit diffusion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 9117–9125, 2023.

[23] Shuliang Ning, Mengcheng Lan, Yanran Li, Chaofeng Chen, Qian Chen, Xunlai Chen, Xiaoguang Han, and Shuguang Cui. Mimo is all you need: a strong multi-in-multi-out baseline for video prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1975–1983, 2023.

[24] Kashif Rasul, Calvin Seward, Ingmar Schuster, and Roland Vollgraf. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *International Conference on Machine Learning*, pages 8857–8868. PMLR, 2021.

[25] Suman Ravuri, Karel Lenc, Matthew Willson, Dmitry Kangin, Remi Lam, Piotr Mirowski, Megan Fitzsimons, Maria Athanassiadou, Sheleem Kashem, Sam Madge, et al. Skilful precipitation nowcasting using deep generative models of radar. *Nature*, 597(7878):672–677, 2021.

[26] 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.

[27] Xingjian Shi, Zhihan Gao, Leonard Lausen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. Deep learning for precipitation nowcasting: A benchmark and a new model. *Advances in neural information processing systems*, 30, 2017.

[28] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[29] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2020.

[30] Cheng Tan, Zhangyang Gao, Lirong Wu, Yongjie Xu, Jun Xia, Siyuan Li, and Stan Z Li. Temporal attention unit: Towards efficient spatiotemporal predictive learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18770–18782, 2023.

[31] Cheng Tan, Siyuan Li, Zhangyang Gao, Wenfei Guan, Zedong Wang, Zicheng Liu, Lirong Wu, and Stan Z Li. Openstl: A comprehensive benchmark of spatio-temporal predictive learning. In *Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

[32] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. Mocogan: Decomposing motion and content for video generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1526–1535, 2018.

[33] Mark Veillette, Siddharth Samsi, and Chris Mattioli. Sevir: A storm event imagery dataset for deep learning applications in radar and satellite meteorology. *Advances in Neural Information Processing Systems*, 33:22009–22019, 2020.

[34] Vikram Voleti, Alexia Jolicoeur-Martineau, and Chris Pal. Mcvd-masked conditional video diffusion for prediction, generation, and interpolation. *Advances in Neural Information Processing Systems*, 35:23371–23385, 2022.

[35] Yunbo Wang, Mingsheng Long, Jianmin Wang, Zhifeng Gao, and Philip S Yu. Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms. *Advances in neural information processing systems*, 30, 2017.

[36] Haixu Wu, Zhiyu Yao, Jianmin Wang, and Mingsheng Long. Motionrnn: A flexible model for video prediction with spacetime-varying motions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15435–15444, 2021.

[37] Sihyun Yu, Kihyuk Sohn, Subin Kim, and Jinwoo Shin. Video probabilistic diffusion models in projected latent space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18456–18466, 2023.

[38] Yuchen Zhang, Mingsheng Long, Kaiyuan Chen, Lanxiang Xing, Ronghua Jin, Michael I Jordan, and Jianmin Wang. Skilful nowcasting of extreme precipitation with nowcastnet. *Nature*, 619(7970):526–532, 2023.

DiffCast: A Unified Framework via Residual Diffusion for Precipitation Nowcasting

Supplementary Material

7. Datasets Detials

SEVIR, the Storm EVent ImagRy (SEVIR) [33] is an annotated, curated and spatio-temporally aligned dataset across five multiple data types including visible satellite imagery, infrared satellite imagery (mid-level water vapor and clean longwave window), NEXRAD radar mosaic of VIL(vertically integrated liquid mosaics) and ground lightning events. In this paper, we focus on the short term weather forecasting task and select all the radar mosaics of VIL as the main data. The dataset contains 20393 weather events from multiple sensors in 2017-2020. Each event consists of a 4-hour length sequence of images sampled in 5 minute steps covering 384 km×384 km patches sampled at locations throughout the continental U.S.. As our task is to predict the future VIL up to 20 frames (120 min) given 5 observed frames (30 min), we following [8] to sample the 25 continuous frames with stride = 12 in every event and split the dataset into training, validation and test sets with the time point January 1, 2019 and June 1, 2019, respectively. The frames are rescaled back to the range 0-255 and binarized at thresholds [16,74,133,160,181,219] to calculate the CSI and HSS following original settings in [33].

    MeteoNet [17] is a multimodel dataset including full time series of satellite and radar images, weather models and ground observations. It covers geographic areas of 550 km×550 km in the northwestern quarter of France and a span over three years, and records every 6 min from 2016 to 2018. Like the SEVIR, we split the radar sequence from 2016 to 2018 into training, validation and test sets with the time point January 1, 2018 and June 1, 2018, respectively. Then, we apply Algorithm 1 to filter precipitation events with a stride-20 sliding window to reduce the noise in the data. Note that a mean pixel threshold T_pixel is used as a filter to precipitation events. The data range of frames in MeteoNet is set to [0-70] and the thresholds are set to [12, 18, 24, 32] following [17] for the CSI and HSS evaluation.

    Shanghai_Radar [5] is a dataset contains continuous radar echo frames generated by volume scans in intervals of approximately 6 minute from October 2015 to July 2018 in Pudong, Shanghai. Every radar echo map covers 501 km×501 km area. We follow [5] to preprocess the echo sequence and also apply Algorithm 1 to filter 25-frame weather event datasets. The data range of frames in Shanghai Radar is set to [0-70] and the thresholds are set to [20, 30, 35, 40] following [17] for the CSI and HSS computation.

    CIKM is a radar dataset from CIKM AnalytiCup 2017 Competition, recording precipitation samples in 101 km×101km area of Guangdong, China. Each sample settles 15 historical radar echo maps as a sample in which the time interval between two consecutive maps is 6-minute. We follow [21] to process the dataset to pad each echo map into 128×128 and follow the original setting to split training, validation and test sets. We transform the pixel in each frame to the reflectivity of [0,76] dBZ and use the thresholds [20,30,35,40] to compute the CSI and HSS.

    The lengths of event sequences in each dataset are set to 25 frames except for the CIKM dataset with 15 frames. Compared to most of the existing studies, which aim to make an hour prediction (*e.g.*, 10 frames with a 6-minute interval), our tasks (except for CIKM dataset) are for the forecast in two hours (*i.e.* 20 frames) in this paper, which are more challenging. Although some recent studies attempt to achieve two hours prediction by frame interpolation (*e.g.*, predicting 10 frames with a 12-minute interval), this trick simplifies the complexity of precipitation dynamics and results in a degrading temporal resolution for prediction.

Algorithm 1 Weather Event Filtering

1: Given continuous frames s, pixel threshold T_{pixel}
2: $i \leftarrow 10$
3: $L_{in}, L_{out} \leftarrow 5, 20$
4: $event\_set \leftarrow \{\}$
5: **while** $i + L_{out} < \text{Len}(s)$ **do**
6:     **if** $\text{Mean}(s[i]) > T_{pixel}$ **then**
7:         $event \leftarrow s[i - L_{in} : i + L_{out}]$
8:         $event\_pixel \leftarrow \sum_{frame \in event} \text{Mean}(frame)$
9:         **if** $event\_pixel >= (L_{in} + L_{out})T_{pixel}/2$ **then**
10:             Add event to $event\_set$.
11:             $i \leftarrow i + L_{out}$
12:             Continue
13:         **end if**
14:     **end if**
15:     $i \leftarrow i + 1$
16: **end while**
17: Return $event\_set$

---

8. DiffCast: Implementation Details

In this section, we will give a detailed description of the implementation for DiffCast's training and inference process and our experimental settings as well.

    Training and Inference. The DiffCast is trained with an end-to-end manner as shown in Figure 2 (b), where the base

Algorithm 2 Training of The Framework

1: **while** not converged **do**
2:     Sampling a sequence $(x, y) \sim \mathcal{D}$, where $len(x) = L_{in}, len(y) = L_{out}$
3:     Making basic prediction $\mu = \mathcal{P}_{\theta_1}(x)$, where $len(\mu) = L_{out}$
4:     Building residual sequence r following Eq. (7)
5:     Grouping segments s_j from r following Eq. (17)
6:     Extracting global hidden state h following Eq. (14)
7:     Sampling diffusion step $t \sim \mathcal{U}(0, ..., T)$
8:     $\mathcal{L}_\epsilon \leftarrow 0$
9:     **while** $j < \lceil \frac{L_{out}}{K} \rceil$ **do**
10:         $\epsilon \sim \mathcal{N}(0, I)$
11:         Disturbing s_j to s_j^t following Eq. (1)
12:         Getting denoising loss $\mathcal{L}_\epsilon^j$ following Eq. (19)
13:         $\mathcal{L}_\epsilon = \mathcal{L}_\epsilon + \mathcal{L}_\epsilon^j$
14:     **end while**
15:     Computing deterministic loss $\mathcal{L}_\mathcal{P}$ following Eq. (6)
16:     Computing final loss $\mathcal{L}$ following Eq. (12) given α
17:     $(\theta_1, \theta_2, \theta_3) \leftarrow (\theta_1, \theta_2, \theta_3) - \nabla_{(\theta_1, \theta_2, \theta_3)} \mathcal{L}$
18: **end while**

Algorithm 3 Inference of The Framework

1: Given initial frames x
2: Making basic prediction $\mu = \mathcal{P}_{\theta_1}(x)$
3: Extracting global hidden state h following Eq. (14)
4: $j \leftarrow 0, \hat{s}_{j-1} \leftarrow 0$
5: **while** $j < \lceil \frac{L_{out}}{K} \rceil$ **do**
6:     $s_j^T \sim \mathcal{N}(0, 1)$
7:     **while** Reverse diffusion from $t = T$ to $t = 1$ **do**
8:         $\epsilon \sim \mathcal{N}(0, I)$
9:         Estimating target noise ϵ_{θ_2} following Eq. (16)
10:         Recovering s_j^{t-1} from s_j^t following Eq. (5)
11:     **end while**
12:     Getting current residual segment $\hat{s}_j$
13: **end while**
14: Computing target frames $\hat{y}$ following Eq. (13)

Figure 8. CSI with threshold

deterministic predictor and residual diffusion model are optimized within the same training iteration. The complete training procedure is summarized in Algorithm 2. In the inference phase, the framework also utilizes the base predictor to estimate the global trend and then apply the diffusion model to generate the residual segments autoregressively. The final prediction is obtained by combining the two components. The inference procedure is summarized in Algorithm 3.

Experimental details All experiments are conducted on a computer with NVIDIA A6000 GPU (48G memory) and all models, including DiffCast equipped with various backbones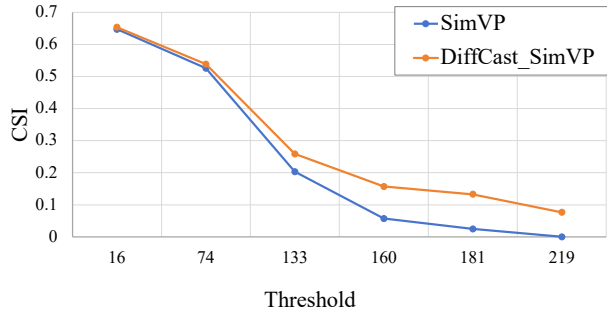 and single backbones, can fit in a single GPU. As for the implementation of various backbones, we easily rebuild the most backbones from OpenSTL [31] library to adapt with DiffCast. We construct the GTUNet with a hierarchical UNet architecture with temporal attention blocks. This structure is composed of four up/down layers, with a hidden size of 64, and is subsequently upscaled/downscaled by factors of 1,2,4,8, respectively. Despite the increased number of parameters (*e.g.*, from 20.13MB to 66.40 MB for DiffCast_MAU) and higher training costs (*e.g.*, from 18 hours at 12 batch size to 23 hours at 6 batch size for 30K iterations) associated with the DiffCast framework, we can attain a great paramount for modeling the distribution of stochastic temporal evolution. Furthermore, the DiffCast framework can expedite the forecast by utilizing optimization techniques such as DDIM, DPM-solver *etc*. This capability can fully meet the requirements of short-term precipitation forecasting task in real-world scenarios (*e.g.*, 17 seconds to produce 20-frame forecasts), enabling real-time predictions that are both more accurate and realistic.

9. Additional Analysis

In this section, we give an extra analysis on the design of framework loss.

We decompose the determinism and local stochastics in precipitation evolution and model them with a deterministic component and a residual diffusion component, respectively. Different from other two-stage frameworks, we train the overall framework in an end-to-end manner to simulate the interplay between the determinism and uncertainty, which indicate that the gradient from stochastic diffusion denoising loss can also optimize the deterministic backbone. In Table 5, we compare the performance between the pure deterministic backbones and the intermediate output μ from deterministic component in DiffCast. The results show that the stochastic loss indeed leads to a positive optimization on most of the deterministic backbones.

We point out that the conventional deterministic methods always under-estimate the high-value echoes with the increasing lead time (shown in Figure 1). To further inves-

Deterministic Method	CSI	CSI-pool4	CSI-pool16	HSS	LPIPS	SSIM
SimVP	0.2662	0.2844	0.3452	0.3369	0.3914	0.6570
DiffCast_Simvp-μ	**0.2690**	**0.2828**	**0.3134**	**0.3320**	**0.3961**	**0.6728**
Earthformer	**0.2513**	**0.2617**	**0.2910**	**0.3073**	**0.4140**	**0.6773**
DiffCast_Earthformer-μ	0.2490	0.2579	0.2834	0.3040	0.4391	0.6685
MAU	0.2463	0.2566	0.2861	0.3004	**0.3933**	0.6361
DiffCast_MAU-μ	**0.2542**	**0.2848**	**0.3157**	**0.3361**	0.3929	**0.7028**
ConvGRU	0.2560	0.2685	0.3005	0.3124	0.3785	**0.6764**
DiffCast_ConvGRU-μ	**0.2635**	**0.2873**	**0.3197**	**0.3350**	**0.3860**	**0.6818**
PhyDNet	0.2560	0.2685	0.3005	0.3124	0.3785	0.6764
DiffCast_PhyDNet-μ	**0.2659**	**0.2785**	**0.3105**	**0.3252**	**0.3748**	**0.6811**

Table 5. Analysis of performance for backbones with different optimization strategies on SEVIR.

tigate this, we show in Figure 8 the performance of Diff-Cast_SimVP, in terms of different thresholds. We observe that with the stochastic modeling the high-value echoes indeed can be more accurately maintained and predicted.

10. More Qualitative Results

In this section, we show more illustrative examples on different datasets to compare our DiffCast with baseline methods. As shown in Figure 9, 11, 10, 12, all deterministic backbones deliver blurry results after 60 minutes, with a phenomenon of high-value echoes and details fading away. However, when equipped into our DiffCast framework, all the prediction results of the backbones are consistently enhanced, where the forecast images are not blurry anymore, and the high-value echoes and details are carefully preserved. All the observations validate the effectiveness of the proposed DiffCast framework.
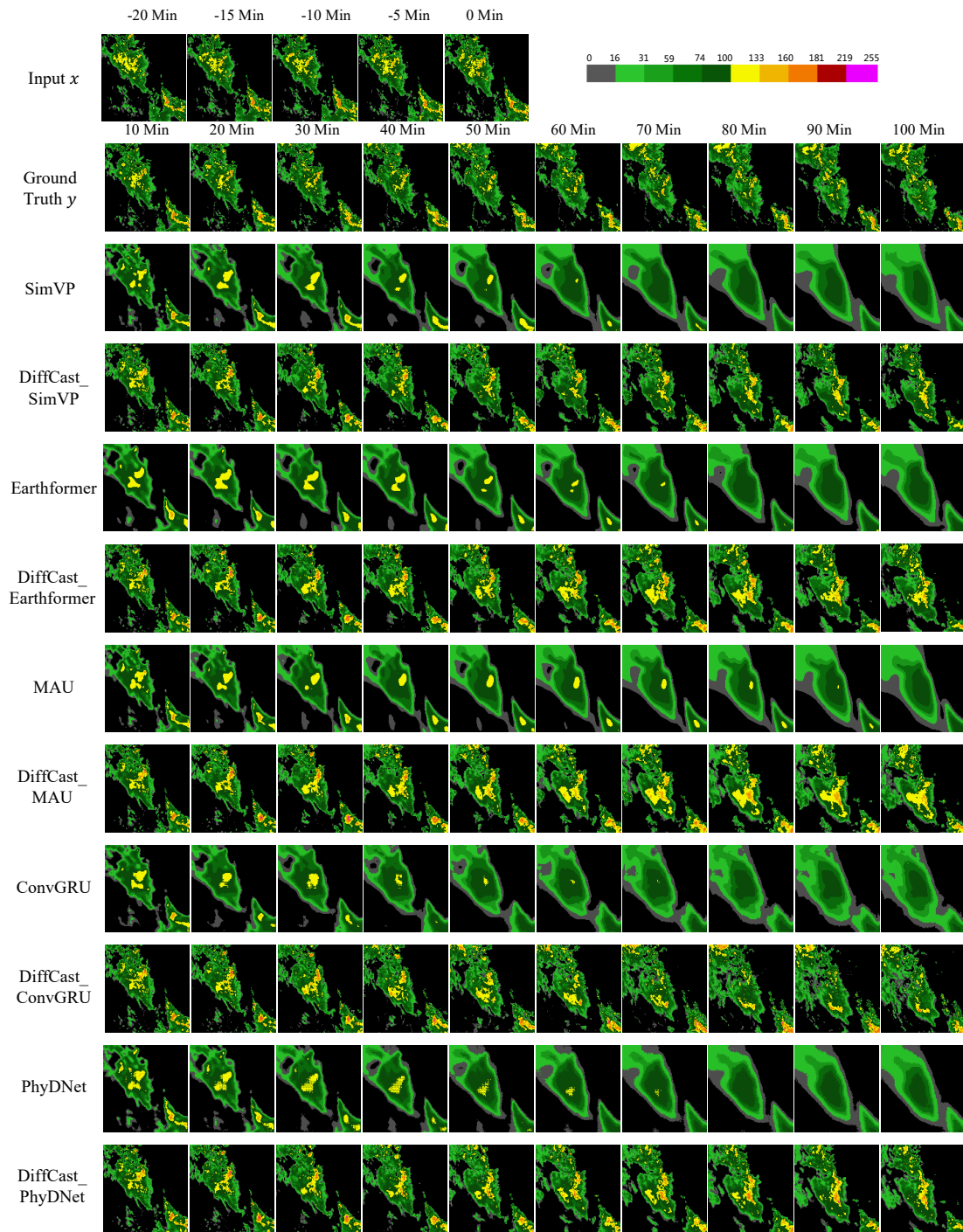
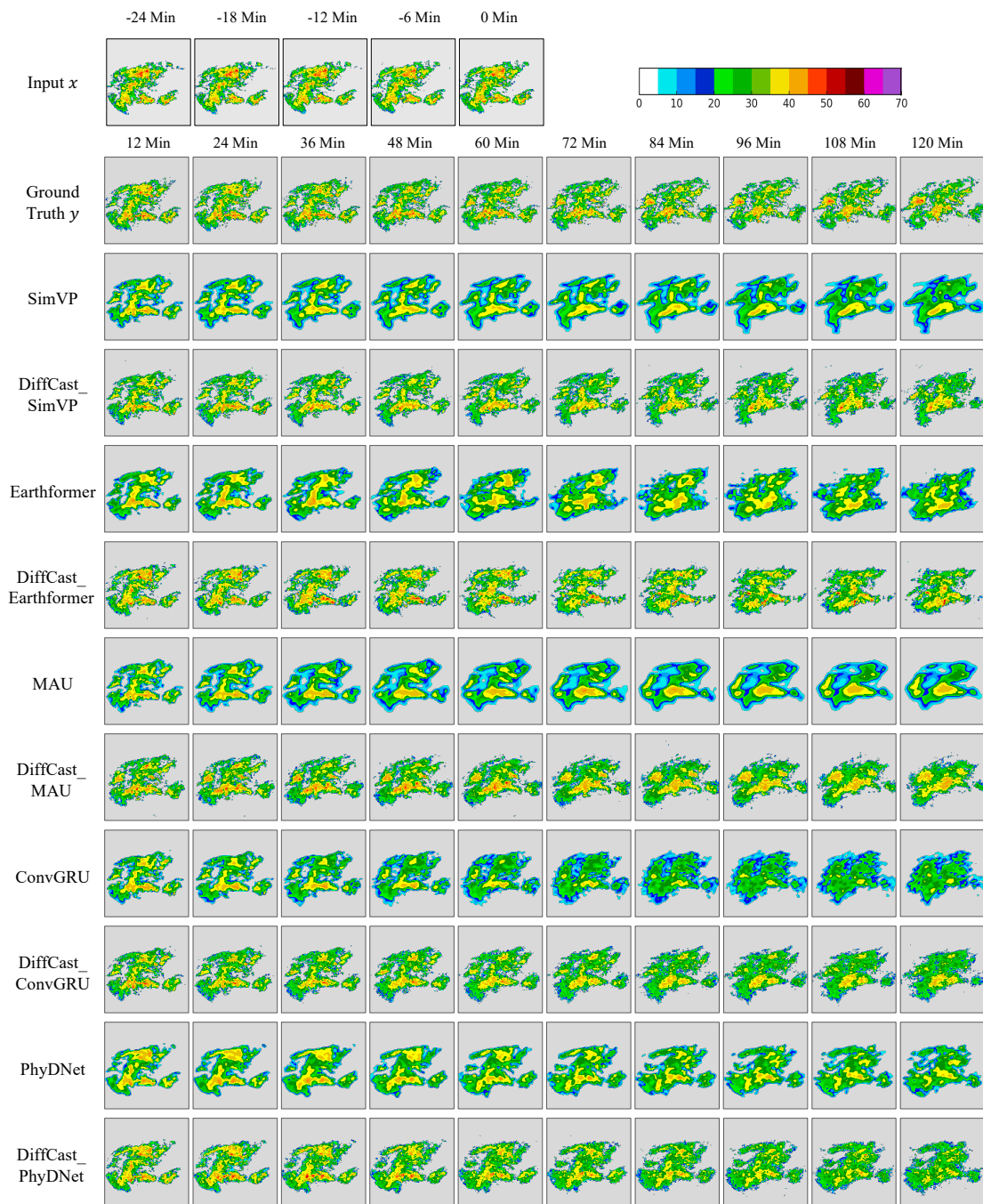Figure 9. Prediction examples on the SEVIR.
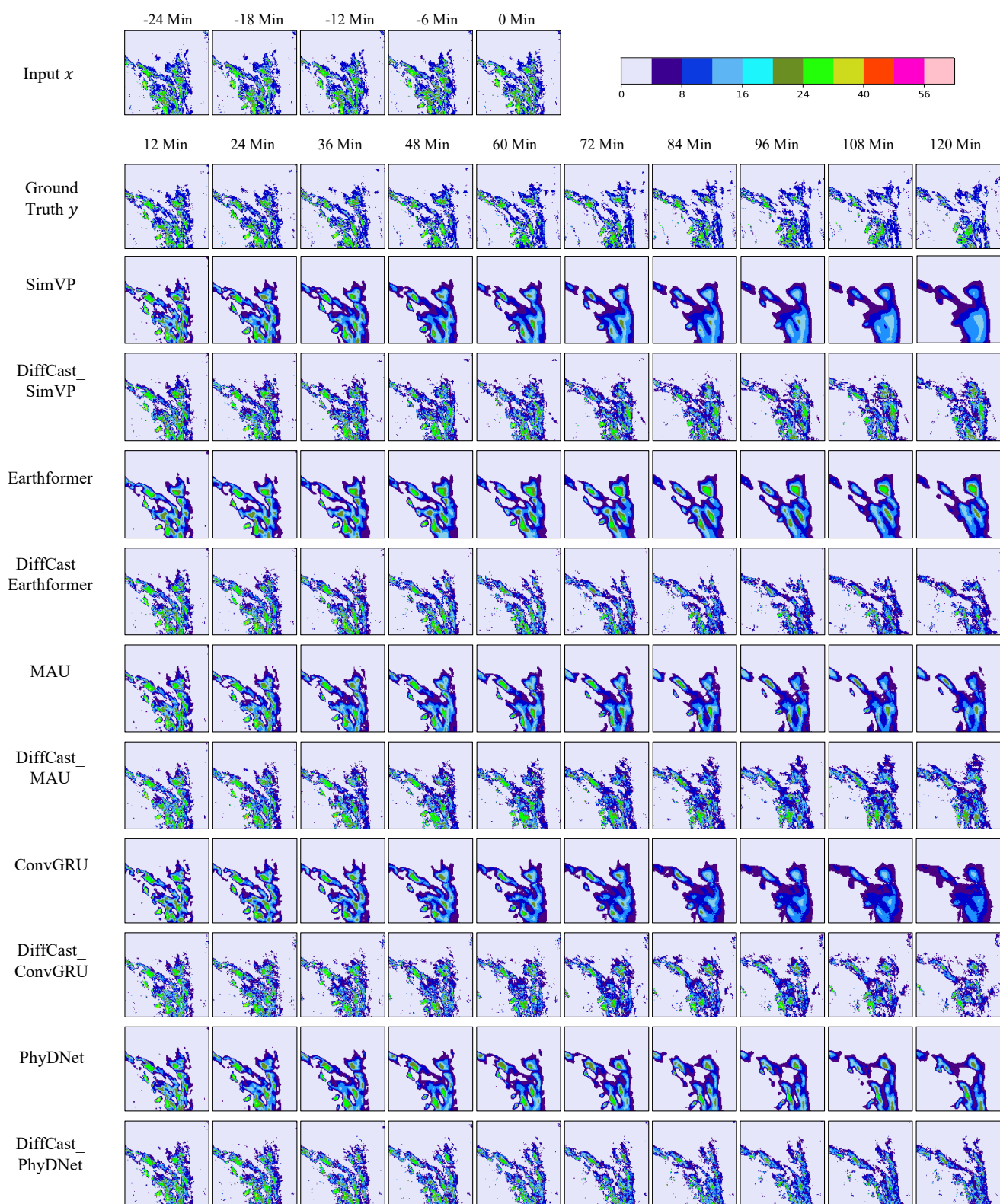
Figure 10. Prediction examples on the Shanghai Radar.
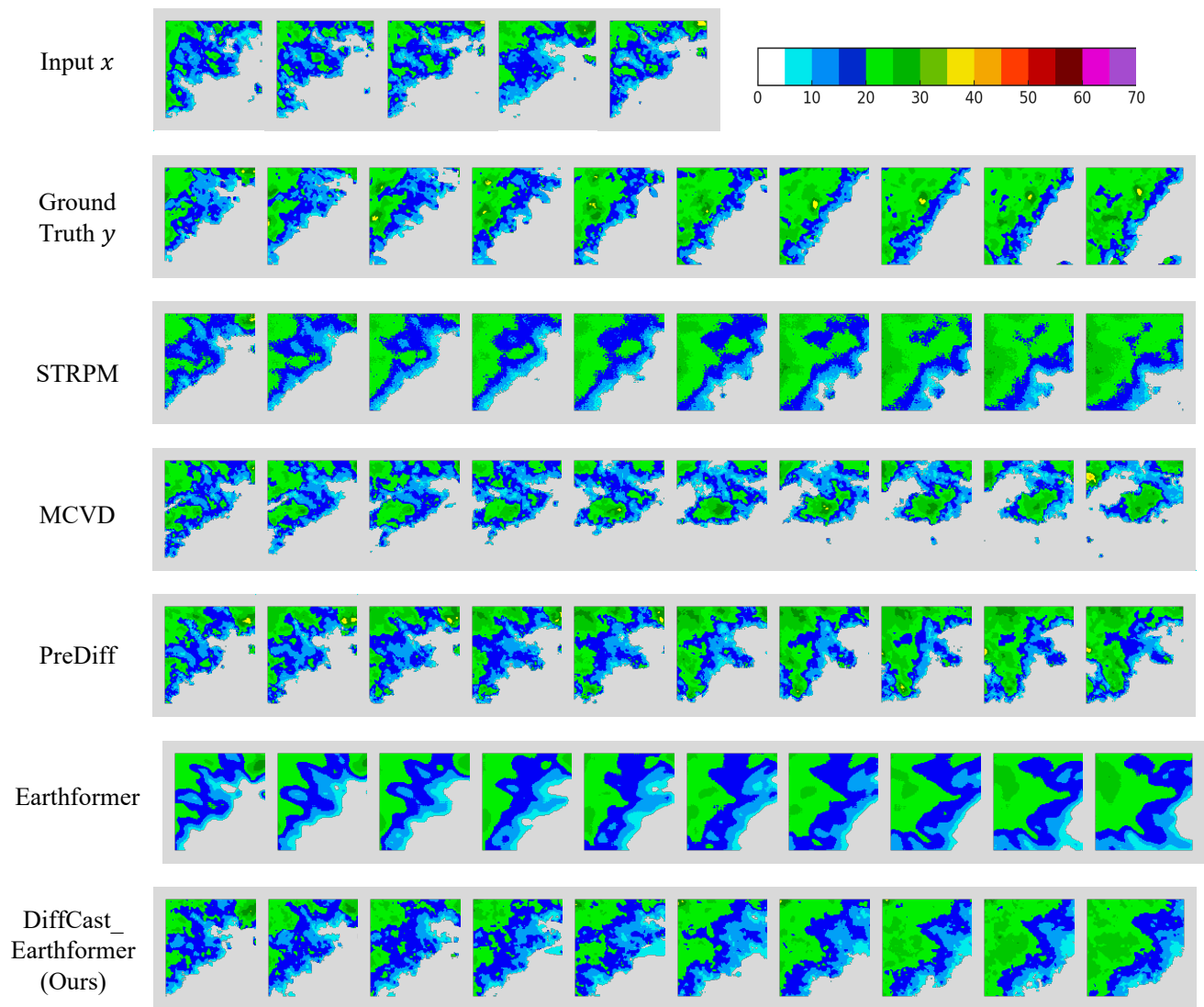
Figure 11. Prediction examples on the MeteoNet.

Figure 12. Prediction examples on the CIKM dataset.