PFST-LSTM: A SpatioTemporal LSTM Model With Pseudoflow Prediction for Precipitation Nowcasting

Chuyao Luo, Xutao Li, and Yunming Ye

Abstract—**Precipitation nowcasting is an important task, which can serve numerous applications such as urban alert and transportation. Previous studies leverage convolutional recurrent neural networks (RNNs) to address the problem. However, they all suffer from two inherent drawbacks of the convolutional RNN, namely, the lack of a memory cell to preserve the fine-grained spatial appearances and the position misalignment issue when combining current observations with previous hidden states. In this article, we aim to overcome the defects. Specifically, we propose a novel pseudo flow spatiotemporal LSTM unit (PFST-LSTM), where a spatial memory cell and a position alignment module are developed and embedded in the structure of LSTM. Upon the PFST-LSTM units, we develop a new sequence-to-sequence architecture for precipitation nowcasting, which can effectively combine the spatial appearances and motion information. Extensive empirical evaluations are conducted on synthetic MovingMNIST++ and CIKM AnalytiCup 2017 datasets. Our experimental results demonstrate the superiority of the proposed PFST-LSTM over the state-of-the-art competitors. To reproduce the results, we release the source code at: https://github.com/luochuyao/PFST-LSTM.**

Index Terms—**Deep learning, image sequence prediction, precipitation nowcasting.**

I. Introduction

PRECIPITATION nowcasting aims to predict the kilometer-wise rainfall intensity within next 6 h [1]. It is an important and useful tool for weather forecasting [2], which can not only help to prevent natural disasters caused by heavy rains but also serve agriculture activity arrangements, transportation route decisions, and people's daily trip planning. However, due to the inherent complexities of atmosphere and nonlinear cloud dynamics, the problem is very challenging [3].

Traditionally, numerical weather prediction (NWP) methods [4] make use of a set of fluid dynamics and the thermodynamics equations. The forecast solutions are derived with numerical simulations from the given initial and boundary conditions. However, NWP methods can only work at the moderate scale and cannot make kilometer-wise predictions accurately. Moreover, their computational costs are too high to be deployed in nowcasting applications [5], which need real-time updates frequently. Another line of studies considers the problem as a radar echo map extrapolation task [6], [7]. Optical flow [8]–[10], and correlation analysis [11] are two classical methods, which estimate the motion fields of precipitation particles for extrapolation. However, the two methods make an estimation based merely on current observations, ignoring a large volume of historical records.

Recently, researchers have resorted to deep recurrent neural networks (RNN) [12] to exploit huge records for radar echo map extrapolation. For example, Shi *et al.* proposed two typical methods under the sequence-to-sequence framework [13], [14], which are ConvLSTM [15] and ConvGRU [1]. These two methods combine the strength of the convolution kernels in image generation with the sequence prediction ability of RNN. Shi *et al.* further improved the ConvGRU model and thus proposed TrajGRU by developing adaptive kernel neighborhoods for extrapolations [1]. Though the three models can offer better performance than the conventional optical flow and correlation analysis methods, the resulting explorations have blurry effects. To address the drawback, Tran *et al.* revised the structure of TrajGRU and integrated the mean square error (MSE), mean absolute error, and structural similarity index (SSIM) [16] as the loss function in [17]. Besides, Wang *et al.* introduced a spatiotemporal memory cell and put forward a novel convolutional LSTM unit to build a predictive RNN (PredRNN) network for extrapolations [18].

The deep RNN models, albeit delivering state-of-the-art performance, fail to accurately model the overall appearances and trajectories of precipitation particles. This is because[1] the models either lack a memory cell for preserving fine-grained spatial appearances or inaccurately combine the previous hidden states with current input in RNN, or both. The limitations come from the inherent structures in the convolutional RNN units, such as ConvLSTM and ConvGRU. On the one hand, though the convolutional kernels are incorporated into LSTM and GRU to extract spatial features, the units only maintain a memory cell for capturing the temporal dynamics, where the spatial appearances are not memorized. Besides, in ConvLSTM and ConvGRU, the current input and hidden states of the previous time step are combined without a position alignment. As the precipitation particles are constantly moving, such position mismatches may cause noises to trajectories prediction.

Manuscript received August 8, 2020; revised October 24, 2020; accepted November 23, 2020. Date of publication November 26, 2020; date of current version January 6, 2021. This work was supported in part by the Shenzhen Science and Technology Program under Grant JCYJ20180507183823045 and JCYJ20200109113014456. *(Corresponding author: Yunming Ye.)*

The authors are with the Department of Computer Science, Harbin Institute of Technology, Shenzhen 518055, China, and also with Shenzhen Key Laboratory of Internet Information Collaboration, Shenzhen 518055, China (e-mail: luochuyao.dalian@gmail.com; lixutao@hit.edu.cn; yeyunming@hit.edu.cn).

Digital Object Identifier 10.1109/JSTARS.2020.3040648

[1] The full account of our findings will be discussed in Section II.

In this article, we aim to overcome the defects of the convolutional RNN units and develop a novel RNN block [12], namely, the pseudoflow spatial-temporal LSTM (PFST-LSTM). In the block, a spatial memory cell is introduced, which can preserve the spatial details level by level. Moreover, a pseudoflow module is proposed, which aligns the current input and previous hidden states with an estimated flow field. By doing so, the developed RNN block can better model the appearances and trajectories of the precipitation particles. Upon the block, we build up a new sequence-to-sequence architecture to address the extrapolation problem, where the appearance details in the spatial memory cells are transferred along both spatial and temporal dimensions in a zigzag way, and motion information is delivered only along the time dimension. Extensive experimental results have been conducted to demonstrate the effectiveness of the proposed PFST-LSTM model.

For clarity, we summarize the main contributions of this article as follows:

1) We propose a novel convolutional RNN block, which overcomes the defects of ConvLSTM and ConvGRU by introducing a spatial memory cell and a pseudoflow alignment module.
2) Upon the block, we construct a sequence-to-sequence method PFST-LSTM for radar echo map extrapolation. In the method, spatial appearances and motion information are effectively combined, where the former is delivered in a zigzag direction, while the latter one is conveyed horizontally.
3) Extensive empirical evaluations on the MovingMNIST++ and CIKM AnalytiCup 2017 datasets are conducted. Our experimental results show that the proposed PFST-LSTM yields significantly better performance than the state-of-the-art methods.

II. Related Work

As the key of the precipitation nowcasting, the radar echo extrapolation can be described as follows. Given a sequence of past radar observations $X_{1:t}$ ($X_{1:t} \in R^{t\times K_1\times K_2\times K_3}$), the extrapolation model aims to forecast the future maps $X_{t+1:T}(X_{t+1:T} \in R^{(T-t)\times K_1\times K_2\times K_3})$. Here, K_1, K_2, and K_3 denote the width, height, and channel of radar maps, respectively. In general, the channel number K_3 is one. The t and $(T-t)$ are the lengths of the input and output of the radar image sequences, respectively. Then, we review the related models that can be applied in this setting for extrapolation.

A. ConvLSTM

ConvLSTM was proposed in [15]. As the radar echo map extrapolation aims to predict a sequence of images, which have spatial structures, the ConvLSTM units replace the full connections in LSTM with convolutions. Specifically, in the ConvLSTM units the input modulation gate g_t, input gate i_t, forget gate f_t, memory cell C_t, output gate o_t, and hidden state H_t are updated as follows:

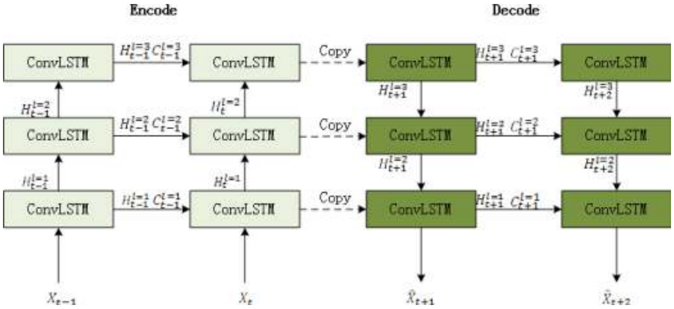


Fig. 1. Sequence-to-sequence extrapolation architecture based on ConvLSTM.

$$
\begin{aligned}
g_t &= \tanh(W_{xg} * X_t + W_{hg} * H_{t-1} + b_g)\\
i_t &= \sigma(W_{xi} * X_t + W_{hi} * H_{t-1} + b_i)\\
f_t &= \sigma(W_{xf} * X_t + W_{hf} * H_{t-1} + b_f)\\
C_t &= f_t \circ C_{t-1} + i_t \circ g_t\\
o_t &= \sigma(W_{xo} * X_t + W_{ho} * H_{t-1} + b_o)\\
H_t &= o_t \circ \tanh(C_t) \qquad (1)
\end{aligned}
$$

where σ represents sigmoid activation function, $*$ is the convolution operation, and $\circ$ is the Hadamard product. The W denotes the convolution kernel matrix to be learned and we use subscripts to differentiate various convolution matrices. Upon the stacks of such units, a sequence-to-sequence architecture is built for radar echo map extrapolation, shown as in Fig. 1. We can see from the figure that the information in memory cell C_t is delivered horizontally along the time dimension, capturing the temporal dynamics. In the vertical dimension, however, only hidden states are transferred. Hence, the ConvLSTM cannot memorize the fine-grained spatial appearances. Moreover, we can see from (1) that when calculating g_t, i_t, f_t and o_t, the current input X_t and hidden state H_{t-1} are directly summed after a convolution operation. As the precipitation particles constantly have displacements, there is inevitably a position mismatch between X_t and H_{t-1}. Such a simple combination may lead to noises.

B. ConvGRU

ConvGRU was developed in [1], [19]. As a simplified version of LSTM, GRU [20] contains fewer parameters and is often easier to train. Hence, Shi *et al.* further develop ConvGRU units, which change the full connections in GRU into convolutions for precipitation nowcasting [1]. The update gate Z_t, reset gate R_t and hidden state H_t in ConvGRU units are computed as follows:

$$
\begin{aligned}
Z_t &= \sigma(W_{xz} * X_t + W_{hz} * H_{t-1} + b_z)\\
R_t &= \sigma(W_{xr} * X_t + W_{hr} * H_{t-1} + b_i)\\
H'_t &= f(W_{xh} * X_t + R_t \circ (W_{hh} * H_{t-1}) + b_h)\\
H_t &= (1 - Z_t) \circ H'_t + Z_t \circ H_{t-1}.
\end{aligned} \qquad (2)
$$

Based on the ConvGRU units, a similar sequence-to-sequence architecture is stacked for extrapolations. According to update equations and the architecture, it is easy to see that the ConvGRU units also lack the ability to carefully model the spatial appearances and to tackle the position misalignment issue.

C. Trajgru

TrajGRU was first established in [1]. Because the convolutional filters in ConvLSTM and ConvGRU are location-invariant, not suitable to model the trajectories patterns, TrajGRU learns and adopts dynamic recurrent connections for convolution GRU. Specifically, its main formulas are given as follows:

$$
\begin{aligned}
&U_t, V_t = \gamma(X_t, H_{t-1})\\
&\quad Z_t = \sigma\left(W_{xz} * X_t + \sum_{l=1}^{L} W_{hz}^{l} * \text{warp}(H_{t-1}, U_{t,l}, V_{t,l})\right)\\
&\quad R_t = \sigma\left(W_{xr} * X_t + \sum_{l=1}^{L} W_{hr}^{l} * \text{warp}(H_{t-1}, U_{t,l}, V_{t,l})\right)\\
&\quad H_t' = f\Bigg(W_{xh} * X_t + R_t\\
&\qquad\qquad \circ \left(\sum_{l=1}^{L} W_{hh}^{l} * \text{warp}(H_{t-1}, U_{t,l}, Vt, l)\right)\Bigg)\\
&\quad H_t = (1 - Z_t) \circ H_t' + Z_t \circ H_{t-1}
\end{aligned}
\tag{3}
$$

where L is the total number of allowed links, U_t and V_t are the flow fields, which store the dynamic connection generated by a two-layer network γ, and the $warp$ function selects the dynamic connections via the bilinear sampling kernels [21], [22]. We can see that TrajGRU does not have a special design for preserving spatial appearance details. Though it tries to address the position mismatch problem, its $warp$ operator takes only X_t and H_{t-1} as input, which may not be sufficient.

D. PredRNN

PredRNN was developed in [18]. To better model the spatial correlation and temporal dynamics, a spatiotemporal LSTM unit (ST-LSTM) is designed in PredRNN. ST-LSTM incorporates a spatial memory cell. In addition to the conventional LSTM operations on temporal dimension, the main formulae related to the spatial memory cell are defined as follows:

$$
\begin{aligned}
&g_t = \tanh(W_{xg} * X_t \mathbb{I}_{\{l=1\}} + W_{hg} * H_t^{l-1} + b_g)\\
&i_t = \sigma(W_{xi} * X_t \mathbb{I}_{\{l=1\}} + W_{hi} * H_t^{l-1} + W_{mi} \circ M_t^{l-1} + b_i)\\
&f_t = \sigma(W_{xf} * X_t \mathbb{I}_{\{l=1\}} + W_{hf} * H_t^{l-1} + W_{mf} \circ M^{l-1} + b_f)\\
&M_t^l = f_t \circ M_t^{l-1} + i_t \circ g_t\\
&o_t = \sigma(W_{xo} * X_t \mathbb{I}_{\{l=1\}} + W_{ho} * H_t^{l-1} + W_{mo} \circ M_t^l + b_o)\\
&H_t^l = o_t \circ \tanh(M_t^l)
\end{aligned}
\tag{4}
$$

where H_t^l and M_t^l are the hidden state and spatial memory state of the lth level at time t, respectively, and $\mathbb{I}$ denotes the indicate function. Different from LSTM, the input modulation gate, input gate, forget gate, and output gate is updated by the hidden state of the previous layer at the current time rather than that at the previous time step. In other words, the unit aims to memorize the spatial information layer by layer. Hence, it can better model the spatial appearances. Then, to enhance the modeling capability for short-term sequence dynamics, Wang *et al.* further improved LSTM and proposed Causal LSTM [23] unit, which integrates a temporal memory into the spatial memory. In addition, a gradient highway unit was also developed to tackle the gradient issue for long-term prediction. Upon the two special designs, an enhanced version (named PredRNN++) was built and more promising results were delivered.

In [24], an E3D-LSTM model was put forward by replacing 2-D into 3-D convolution with attention mechanism. Instead of stacking the spatiotemporal LSTM units into a sequence-to-sequence predictor, all these models adopt a conventional sequence prediction scheme and forecasts only one image at each time. To generate a sequence prediction, the forecast result is recursively treated as input. Despite the introduction of an extra memory cell to capture the spatial appearance details, since the temporal dimension is still updated as conventional LSTM, the position misalignment problem remains.

E. Other Related Models

Our work is also related to video prediction studies. For example, Finn *et al.* [25] proposed an action-conditional prediction method, which models the pixel motion explicitly with a motion distribution. Bert *et al.* [26] introduced to dynamically adapt the weights of convolutions upon the given input for video prediction, instead of keeping them fixed as conventional convolution operation. Wang *et al.* proposed the memory in memory (MIM) neural network [27] to model the nonstationary and approximately stationary spatiotemporal features for prediction. In addition, in [28] a convolution network was proposed by nicely leveraging the context features to enhance the video prediction performance. Reda *et al.* developed a spatially displaced convolution in [29]. Adversarial learning networks [30], [31] and partial differential equation networks [32], [33] were also put forward for video prediction.

In this article, we aim to overcome the deficiency of the existing methods by developing a novel LSTM unit, which can not only preserve the fine-grained spatial appearances but also address the position mismatch problem. The performance of our model also can be shown by comparing the above models in Section IV.

III. Proposed Model

A. Pseudoflow Spatiotemproal LSTM Units

As noted in related work, due to the lack of a spatial memory cell, the previous convolutional RNN models cannot produce promising appearances for radar echo map extrapolation. The remedy to this issue is simple and easy. We follow [18] and embed a spatial memory cell into the structure of LSTM.

Our solution of the position misalignment is more sophisticated. To better understand how the problem appears, we give

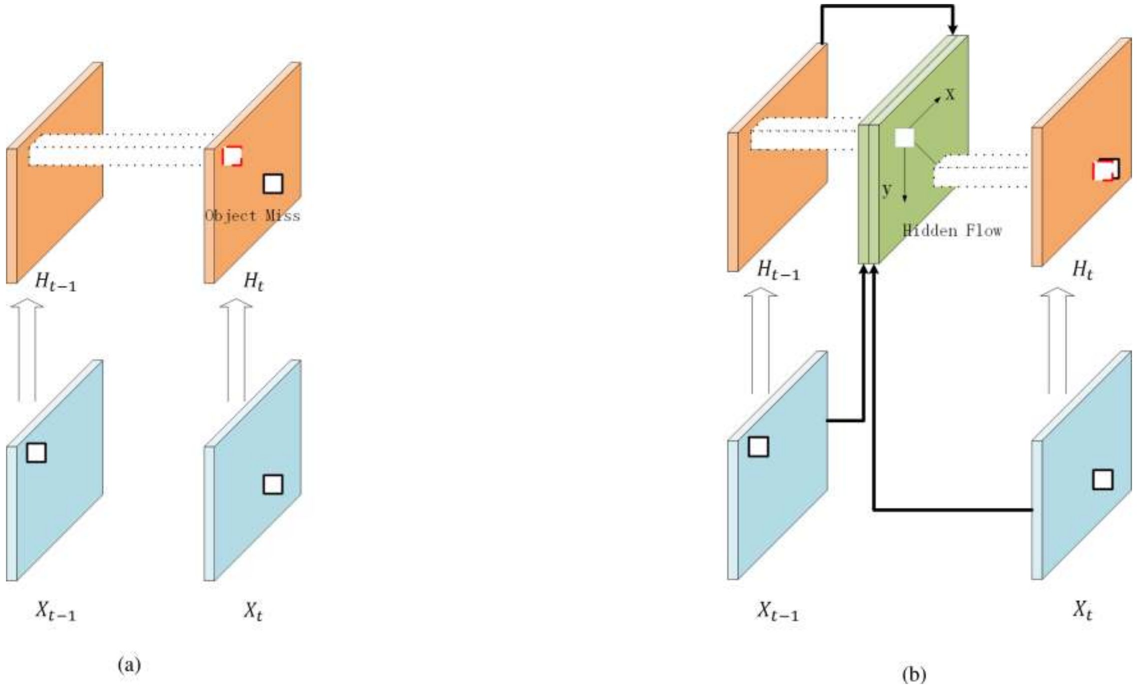


Fig. 2. Illustration of the mismatch problem and our solution. The left column shows the cause of mismatch problems. The right column presents our notion of the solution. The blue block and orange block denote the feature map of input and hidden state, respectively. In input feature maps, we leverage a black square to denote the same moving object at different timestamps. In the hidden state, the red square indicates the actual position that we perform convolution. We can see there is a mismatch between red and black square in existing RNN while the issue will be addressed by a hidden flow (shows as green blocks) in our solution. (a) An illustration example of the position misalignment problem in existing convolutional RNNs. (b) The notion of our solution.

in Fig. 2(a) an illustrative example. In the example, a square moves from the upper left corner at time step $t-1$ to the bottom right one at time step t, which is a rapid movement. If we apply ConvLSTM or ConvGRU to the case, the hidden state H_{t-1} will reside the square features in the upper left part. As a result, H_{t-1} and X_t can be hardly aligned, even after convolutional operations. Thus, the direct summation of previous hidden state and current observation may result in noises. We note that the rapid movement in two adjacent time steps is not a hypothesis for radar echo map extrapolation. There are two reasons: 1) the echo map interval is often around several minutes, e.g., six minutes in our experiments, because radars need to scan in many angles to form an image; 2) the precipitation particles move fast and vary simultaneously, thanks to the complexity of atmosphere.

Inspired by the optical flow methods, we approach the problem by adding a pseudoflow generation operator, which can estimate the displacement for each position. Calibrating the hidden state at the previous time step with the estimated displacements, the positions are expected to be aligned with current observations. For clarity, the notion of our approach is depicted in Fig. 2(b).

We utilize a combination of multiple one-layer convolutional signals and a bilinear sampling $warp$ function to realize the notion. The convolutional signal combination accounts for displacement estimation and the $warp$ function makes the calibration. In particular, we model the displacement D as

$$\begin{aligned} D_t = X_{t-1} * W_{x'd} + X_t * W_{xd} + H_{t-1} * W_{hd} \\ + M_t * W_{md} + b_d \end{aligned} \tag{5}$$

Here, D_t is a K_1-by-K_2-by-2 tensor, K_1 and K_2 denote the height and width of input X, and the third dimension denotes the displacements along with vertical and horizontal directions. We can see that D_t depends on the inputs of previous and current time steps X_t and X_{t-1}, hidden state of previous time step H_{t-1} and spatial memory at current time M_t. The D_t models some motion field which has some key differences from the conventional optical flow. One key difference is that conventional optical flow methods learn the motion filed by setting up an objective function with the assumption that the brightness of pixels is invariant. Instead, in our approach, the motion field D^t is generated by some convolutions and fusions on various feature maps (e.g., X_t, X_{t-1}, H_{t-1}, and M_t). The motion field mimics the optical flow but does not have the assumption on pixel brightness. Thus, we mention it as the pseudoflow. After the displacement estimation, we adopt the bilinear sampling kernels [21], [22] as our $warp$ function to calibrate the positions in H_{t-1} and C_{t-1}

$$\begin{aligned} H'_{i,j,c} &= \sum_{k_1=1}^{K_1}\sum_{k_2=1}^{K_2} H_{k_1,k_2,c} \\ &\max(0, 1-|i+D_{i,j,0}-k_1|)\max(0, 1-|j+D_{i,j,1}-k_2|) \\ C'_{i,j,c} &= \sum_{k_1=1}^{K_1}\sum_{k_2=1}^{K_2} C_{k_1,k_2,c} \\ &\max(0, 1-|i+D_{i,j,0}-k_1|)\max(0, 1-|j+D_{i,j,1}-k_2|). \end{aligned} \tag{6}$$

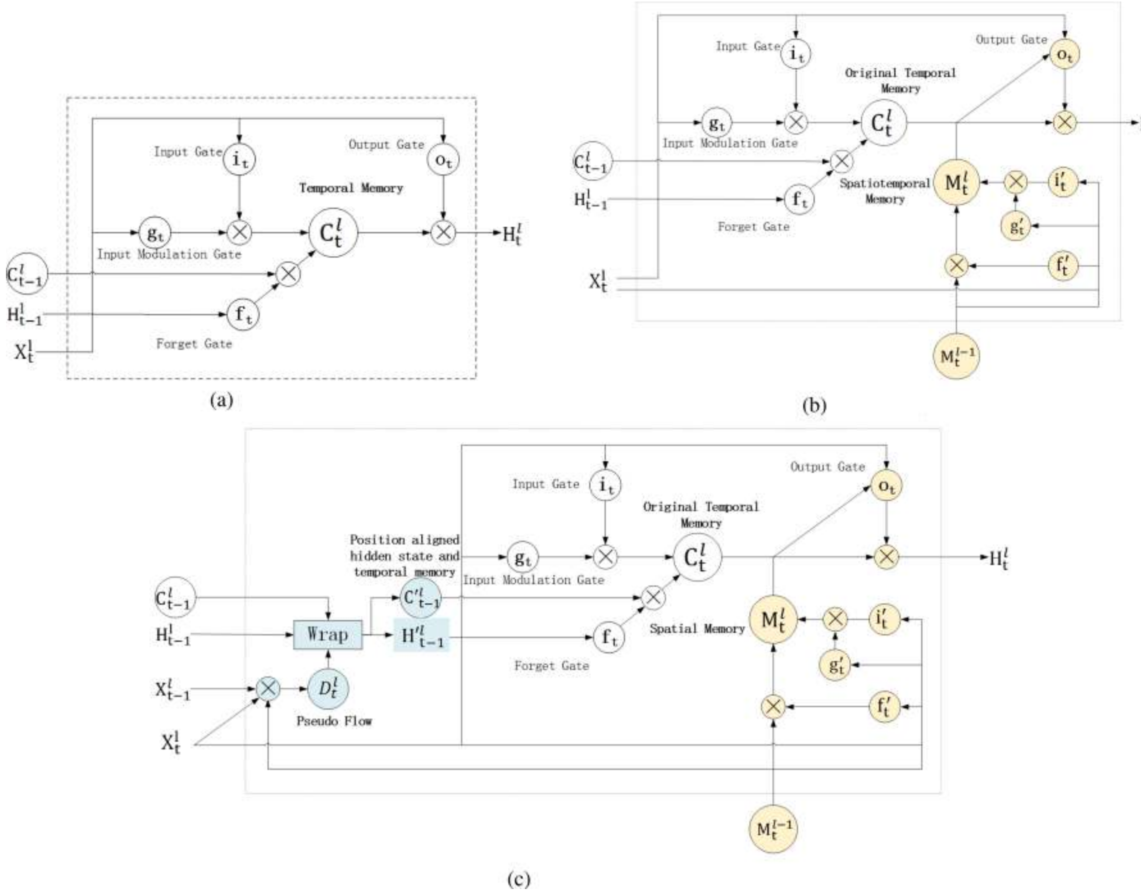


Fig. 3. Comparison of three different LSTM units. (a) Conventional LSTM. (b) ST-LSTM. (c) PFST-LSTM.

Here, i and j are the indices of positions and c is the index for channels. H' and C' denote the calibrated results of H and C with our pseudoflow. For brevity, we omit the subscript t for timestaps. The $warp$ operator utilizes a weighted summation of hidden or memory state at the previous time step to calculate aligned state, where the weights are determined according to the vertical and horizontal distances after the position calibration. We note that as the spatial memory state M_t models the appearances at the current time step, which does not need an alignment operation.

Embedding the spatial memory cell and the position alignment module into LSTM, we obtain our pseudo flow spatiotemporal LSTM unit (PFST-LSTM). As for a comparison with ConvLSTM and ST-LSTM, we depict their structures as ours in Fig. 3. We observe that compared to LSTM, ST-LSTM introduces an extra spatial memory cell, shown as the orange parts in Fig. 3(b). Our PFST-LSTM is more advanced, and not only embeds the spatial memory cell (orange parts), but also develops a position alignment module (blue parts). The two special designs are able to remedy the drawbacks of existing convolutional RNNs. Specifically, the computation of the unit is formulated as follows:

$$
\begin{aligned}
g_t &= \tanh(W_{xg} * X_t^l + W_{hg} * H'^l_{t-1} + b_g)\\
i_t &= \sigma(W_{xi} * X_t^l + W_{hi} * H'^l_{t-1} + b_i)\\
f_t &= \sigma(W_{xf} * X_t^l + W_{hf} * H'^l_{t-1} + b_f)\\
C_t^l &= f_t \circ C'^l_{t-1} + i_t \circ g_t\\
g'_t &= \tanh(W'_{xg} * X_t^l + W_{mg} * M_t^{l-1} + b'_g)\\
i'_t &= \sigma(W'_{xi} * X_t^l + W_{mi} * M_t^{l-1} + b'_i)\\
f'_t &= \sigma(W'_{xf} * C_t^l + W_{mf} * M_t^{l-1} + b'_f)\\
M_t^l &= f'_t \circ M_t^{l-1} + i'_t \circ g'_t\\
o_t &= \sigma(W_{xo} * X_t^l + W_{ho} * H'^l_{t-1} + W_{co} \circ C_t^l\\
&\quad + W_{mo} * M_t^l + b_o)\\
H_t^l &= o_t \circ \tanh(W_{1\times1} * [C_t^l, M_t^l]).
\end{aligned}
\tag{7}
$$

As the spatial memory state M_t and hidden state H_t will also be delivered along the spatial dimension, we utilize a superscript l to denote the lth layer in the spatial dimension. We can see that the input module gate g_t, input gate i_t, forget gate f_t and memory state C_t^l related to the temporal cell are updated with calibrated hidden and memory states H'^{l-1}_t and M'^{l-1}_t. The corresponding gates and states on the spatial cell are renewed layer by layer, whose input and memory information are X_t^l and M_t^{l-1}. Next, we discuss how to utilize the PFST-LSTM units for echo map extrapolation.

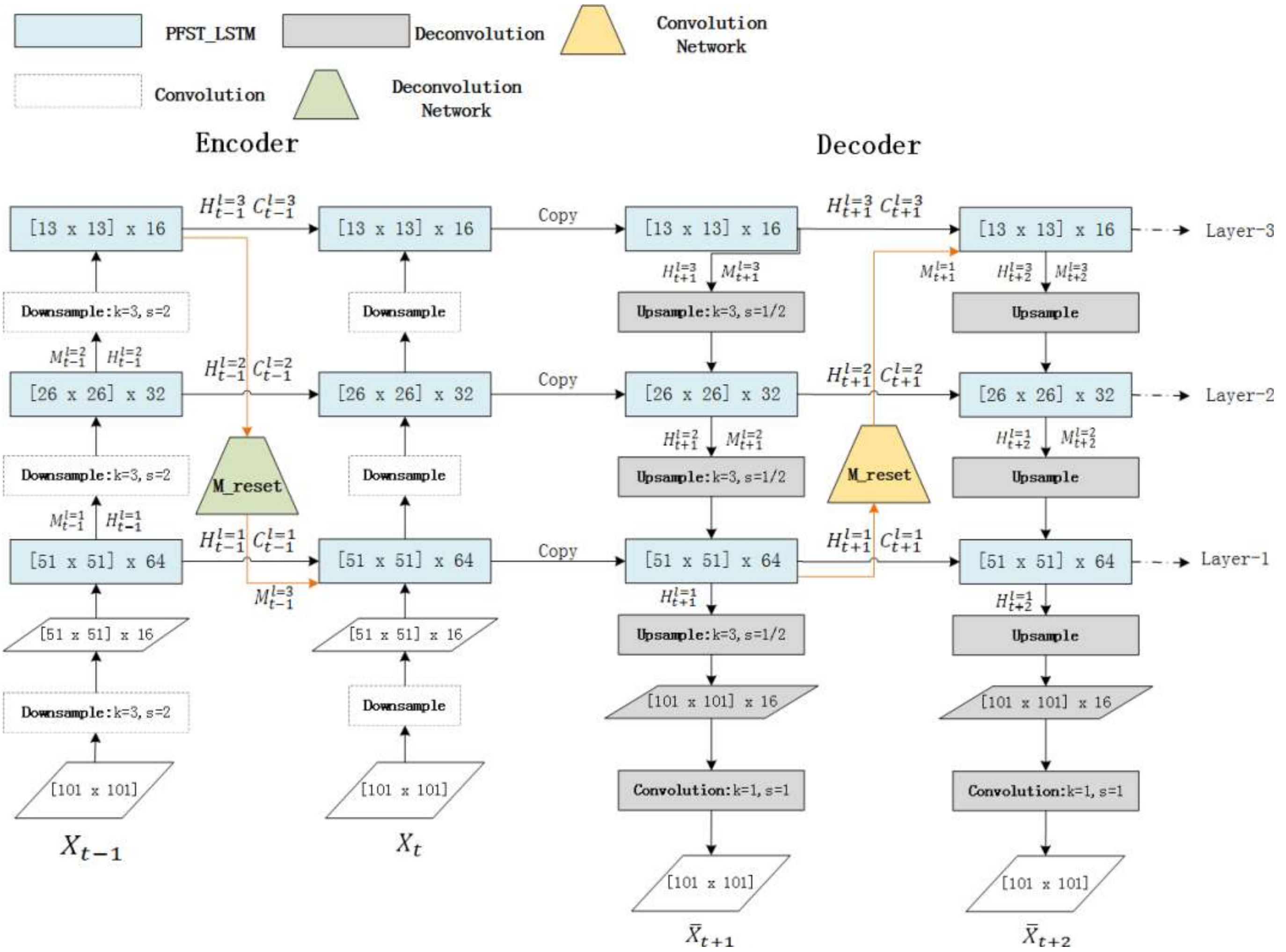


Fig. 4. Three-layer sequence-to-sequence encode-decode architecture based on PFST-LSTM units.

Notably, the design of the PFST-LSTM unit is not a direct combination of the TrajGRU and ST-LSTM. Compared with TrajGRU, the main difference is that the flow is generated with the contextual input x_{t-1} and x_t in our model, which produces the pseudo optical flow D. Besides, the spatial representation in spatial memory and hidden state also enriches the form of D. The ablation experiment in Section IV-B.4 will also demonstrate the superiority of our method.

B. PFST-LSTM Architecture

We stack the PFST-LSTM units into a sequence-to-sequence predictor, shown as in Fig. 4. The predictor comprises two parts. One is the encoder to extract spatial appearances features and temporal dynamics from a sequence of echo maps observed, and the other is the encoder, which leverages extraction results for predicting the further echo map sequence. Both the encoder and decoder have a three-layer PFST-LSTM structure, where downsample and upsample convolutional kernels are inserted in between two PFST-LSTM layers. As the PFST-LSTM unit maintains both temporal and spatial memory cells, we can see that their states C_t^l and M_t^l are, respectively, delivered horizontally and vertically along with H_t^l. By doing so, the motion patterns can be modeled time by time and the spatial appearances be preserved layer by layer. Furthermore, we expect the preserved spatial appearances also to be effectively combined with the temporal motion features. To this end, a special three-layer deconvolution subnetwork is developed for the encoder, shown as a green trapezoid in Fig. 4. The subnetwork enlarges the most abstract memory state $M_{t-1}^{l=3}$ at the previous time step into appropriate size and combines it with $H_{t-1}^{l=1}$ and $C_{t-1}^{l=1}$ as input at the current step, shown as an orange arrow in the encoder. Symmetrically, a convolution subnetwork is also developed for the decoder. The scheme makes the spatial appearances are conveyed in a zigzag manner, which effectively integrates them with motion patterns.

IV. Experiments

A. Experiment on MovingMNIST++

1) Dataset and Parameter Setting: Following [1], we generate synthetic image sequences by randomly selecting three digits and gradually applying the movement, rotation, and illumination on them to make motion patterns. The synthetic data set contains

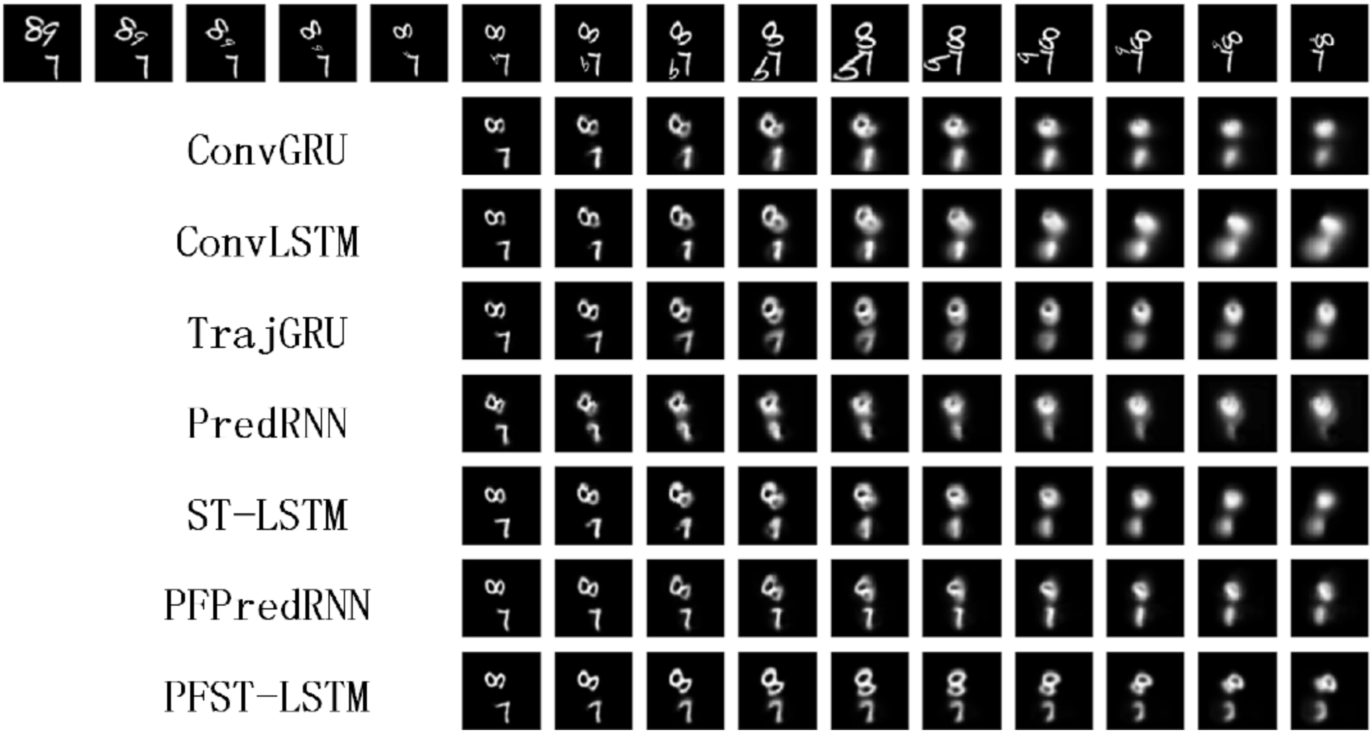


Fig. 5. Prediction results of all methods on an example from MovingMNIST++. The first five images in the first row are the input, and the remainders denote the ground-truth output.

TABLE I
PARAMETER SETTINGS TO GENERATE THE MOVINGMNIST++

Variables	Training Set	Test Set 1	Test Set 2	Test Set 3
max_veloctiy	3.6	5.6	5.6	4.6
initial_velocity	[0.0,3.6]	[0.0,5.6]	[0.0,5.6]	[0.0,4.6]
scale_variation_range	[0.9,1.1]	[0.7,1.3]	[0.6,1.4]	[0.5,1.5]
rotation_angle_range	[-30,30]	[-45,45]	[-45,45]	[-40,40]
global_rotation_angle_range	[-20,20]	[-30,30]	[-45,45]	[-30,30]
illumination_factor_range	[0.6,1.0]	[0.7,1.3]	[0.8,1.2]	[0.8,1.2]

TABLE II
RESULTS ON THE THREE TEST SETS (UNIT: MSE $\times 10^{-2}$)

Method	Test Set 1	Test Set 2	Test Set 3
ConvGRU	6.28	8.13	9.91
ConvLSTM	6.00	7.41	8.79
TrajGRU	5.83	7.23	8.67
PredRNN	6.27	7.71	9.26
ST-LSTM	6.02	7.49	8.96
PFPredRNN	5.90	7.16	8.73
PFST-LSTM	**5.80**	**7.12**	**8.51**

8000 training samples and 2000 validation samples. As for the test set, we generate 4000 samples with different parameter settings, shown as in Table I. The prediction is performed with a sequence of five images as input and a corresponding sequence of ten images as output. In this experiment, we normalize all images into $[-1.0, 1.0]$ and leverage a batch size of 16. All models adopt the early stopping strategy and are trained with a learning rate of 0.0004. The max epoch is set to 800.

2) Results and Analysis: Following [17], MSE is adopted as the evaluation metric. Table II shows the results of all the methods. We note that method PredRNN and ST-LSTM are the variants of PFST-LSTM and PFPredRNN without position alignment module, respectively. The difference of the models include PredRNN and PFPredRNN is that ST-LSTM and PFST-LSTM adopt the developed sequence-to-sequence architecture while PredRNN and PFPredRNN leverage the conventional sequence prediction scheme (as introduced in Section II). We observe from the table that the proposed PFST-LSTM consistently outperforms the state-of-the-art baselines on three different test sets. PFST-LSTM shows an obvious improvement in ST-LSTM, which is attributed to its position alignment module. The same conclusion also can be obtained by comparing PredRNN and PFPredRNN. Moreover, we find that ST-LSTM and PFST-LSTM deliver better results than PredRNN and PFST-PredRNN, respectively, which implies the effectiveness of the developed sequence-to-sequence architecture. Among all the baseline methods, TrajGRU performs the best, because it considers the position mismatch problem. The fact further verifies our motivation to position calibration. For a visual comparison, we show in Fig. 5, the prediction sequences on an example of all methods. We observe that predictions by ConvLSTM, ConvGRU, and PredRNN are the worst, which tend to become blurry rapidly. This is because they all neglect the position mismatch problem. However, PFST-LSTM, TrajGRU,

TABLE III
COMPARISON RESULTS ON THE CIKM ANALYTICUP 2017 COMPETITION DATASET IN TERMS OF HSS, CSI, AND MSE

dBZ threshold	HSS ↑				CSI ↑				MSE ↓
	5	20	40	avg	5	20	40	avg	
ConvLSTM[15]	0.7115	0.5282	0.1112	0.4503	0.7797	0.4698	0.0608	0.4368	98.57
ConvGRU	0.7122	0.5350	0.1258	0.4577	0.7802	0.4701	0.0687	0.4397	90.31
CDNA	0.7001	0.5128	0.1097	0.4409	0.7714	0.4459	0.0593	0.4256	94.86
DFN	0.7085	0.5459	0.1301	0.4615	0.7780	0.4738	0.0714	0.4410	88.03
TrajGRU	0.7057	0.5546	0.1939	0.4847	0.7715	0.4858	0.1100	0.4558	90.64
PredRNN	0.7176	0.5464	0.1678	0.4773	0.7770	0.4774	0.0947	0.4497	99.29
PredRNN++	0.7159	0.5598	0.1689	0.4816	0.7741	0.4888	0.0948	0.4526	89.69
MIM	0.7266	0.5522	0.1851	0.4879	0.7837	0.4843	0.1054	0.4578	86.91
E3DLSTM	0.7296	0.5136	0.1260	0.4564	0.7910	0.4445	0.0684	0.4346	95.15
PFST-LSTM	**0.7301**	**0.5652**	**0.2256**	**0.5070**	**0.7946**	**0.4979**	**0.1306**	**0.4744**	**82.11**

and PFPredRNN, which both carefully address the problem, yield very promising results. In terms of position accuracy and spatial appearances, PFPredRNN and PFST-LSTM are better than TrajGRU, owing to its extra spatial memory cell. With the ablation of the position alignment module, the performance of PredRNN and ST-LSTM degenerates, which again validates the effectiveness of the position calibration scheme in PFST-LSTM. In terms of the architecture of models, PFST-LSTM is better than PFPredRNN due to the downsample operation between different layers, which makes the shallow layer also obtain a larger field of view.

B. Experiment on Radar Data

1) Dataset: The radar echo map dataset is from the CIKM AnalytiCup 2017 competition, which covers 101 × 101 km area in Shenzhen. Each radar echo map contains 101 × 101 pixels, and each pixel denotes a square of 1 × 1 km. The dataset was originally divided into a training set with 10 000 sequences and two test sets with 2000 sequences. We randomly select 2000 sequences from the training set as a validation set and select 1023 sequences, which contain the high echo area (at least one pixel with the reflectivity $>$ 40dBZ) from the two test sets as a test set. Each sequence covers 90 min with an interval of 6 min. Hence, it contains 15 echo maps. Our objective is utilizing the first five echo maps as input and predicting the next ten ones.

2) Parameter Setting and Evaluation Metrics: The detailed parameters of the proposed PFST-LSTM are shown in Fig. 4. Moreover, we set the number of channels for all the spatial memory state to be 16. The parameter settings of all the baseline methods follow [17] and [18]. All the methods are trained with a learning rate of 0.0004 and the early stopping strategy, and Adam optimizer is adopted. We normalize all the echo maps into $[-1.0, 1.0]$ for learning, and batch size is set to four among all models.

Following [15], we evaluate the results according to the Heidk Skill Score (HSS) [34] and Critical Success Index (CSI) [1]. To this end, we apply the following transformation to convert the pixel value p in ground-truth and the predicted echo maps into the reflectivity dBZ

$$\mathrm{dBZ} = p \times 95/255 - 10. \tag{8}$$

Then, we change the ground echo maps into binary matrices according to a threshold τ. If the reflectivity is larger than the threshold, it is set to 1, otherwise, it is set to 0. By comparing the binary matrices of ground truth and prediction, we obtain TP, FP, TN, and FN, which denote the numbers of the true positive, false positive, true negative, false negative elements. The HSS and CSI are computed as follows:

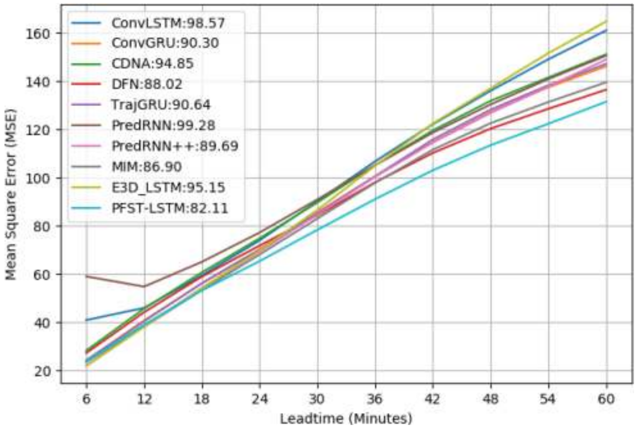


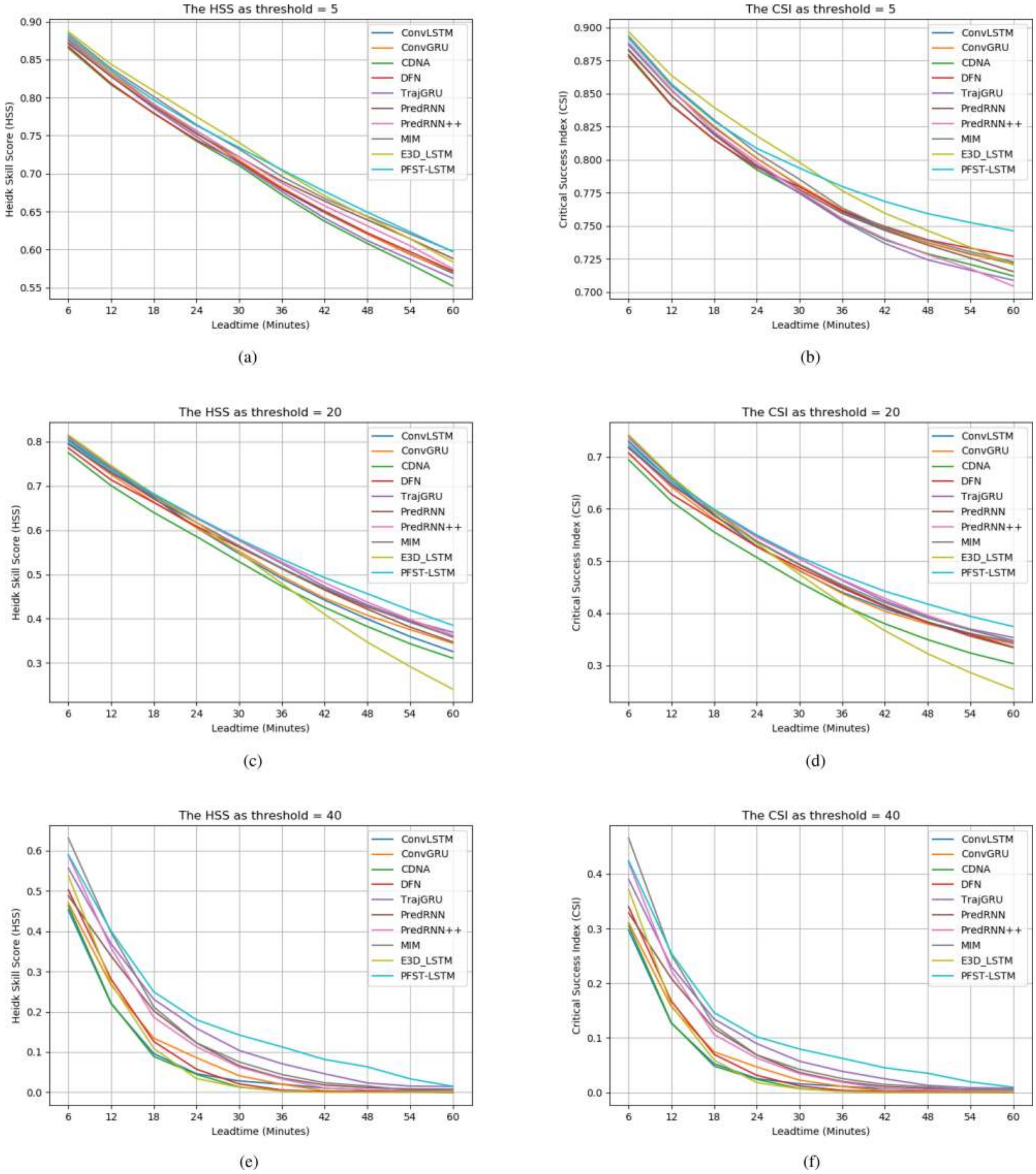
Fig. 6. Performance changes against different nowcast lead time in terms of MSE. (Best view in color).

$$\mathrm{HSS} = \frac{2 * (\mathrm{FN} * \mathrm{TN} - \mathrm{FN} * \mathrm{FP})}{(\mathrm{TP} + \mathrm{FN}) * (\mathrm{FP} + \mathrm{FN}) + (\mathrm{TP} + \mathrm{FP}) * (\mathrm{FP} + \mathrm{TN})} \tag{9}$$

$$\mathrm{CSI} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN}} \tag{10}$$

Specifically, we select 5, 20, and 40 dBZ as the threshold, respectively. The HSS and CSI are integrated metrics that take both probabilities of detection and false alarm rate into account and can directly reflect the goodness of a model. The larger the HSS and CSI are, the better the performance is. Besides, we also utilize MSE to evaluate the performance.

3) Results and Analysis: Table III shows the results. We can see that PFST-LSTM outperforms all methods under different dBZ thresholds especially in the highest thresholds. In terms of HSS, the prediction of our method is only 0.07% and 0.96% higher than the second one as the threshold is 5 and 20 respectively. However, when the threshold is 40, it is improved to 16.36%. A similar phenomenon also can be seen for CSI. It implies that our model can more accurately predict the region of high rainfall. For the MIM model, although its performance is



(a)

(b)

(c)

(d)

(e)

(f)

Fig. 7. Performance changes against different nowcast lead time in terms of HSS and CSI scores. (Best view in color). (a) HSS $\tau = 5$. (b) CSI $\tau = 5$. (c) HSS $\tau = 20$. (d) CSI $\tau = 20$. (e) HSS $\tau = 40$. (f) CSI $\tau = 40$.

better than other baselines, it is inferior to the proposed PFST-LSTM model. Among all the methods, ConvLSTM, ConvGRU, CDNA, and DFN perform the worst, as these approaches neither address the mismatch problem nor carefully model the spatial appearance.

To present the performance at different nowcasting lead time, we plot the frame-wise scores of the MSE in Fig. 6. We can see that as the lead time increases all the models' predictions become worse. However, our model PFST-LSTM in general delivers the best performance. As for E3DLSTM, it yields the best prediction in the first 6 min. Nevertheless, its performance is getting worse gradually and becomes the worst among all the models at the 10_{th} prediction. Moreover, we also depict in Fig. 7, the HSS and CSI curves w.r.t. different lead time under thresholds 5,

TABLE IV
COMPARISON RESULTS OF ABLATION STUDY ON THE CIKM ANALYTICUP 2017 COMPETITION DATASET IN TERMS OF HSS, CSI, AND MSE

dBZ threshold	HSS ↑				CSI ↑				MSE ↓
	5	20	40	avg	5	20	40	avg	
Conv-LSTM	0.7100	0.5340	0.0900	0.4446	0.7731	0.4665	0.0480	0.4292	88.00
Traj-LSTM	0.7215	0.5474	0.1811	0.4833	0.7860	0.4777	0.1022	0.4553	87.91
ST-Traj-LSTM	0.7156	0.5364	0.1681	0.4734	0.7818	0.4710	0.0941	0.4490	89.21
ST-LSTM	0.7181	0.5372	0.1569	0.4707	0.7841	0.4764	0.0870	0.4492	88.74
PF-LSTM	0.7207	0.5638	0.1296	0.4714	0.7837	0.4937	0.0706	0.4494	82.81
PFST-LSTM	**0.7301**	**0.5652**	**0.2256**	**0.5070**	**0.7946**	**0.4979**	**0.1306**	**0.4744**	**82.12**

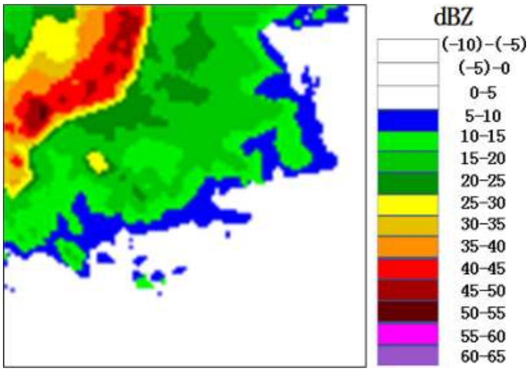


Fig. 8. Color code of the rada echo color map.

20, 40 dBZ, respectively. We can see that E3DLSTM and MIM deliver the best performance at the beginning but their performance becomes worse as the lead time increases. The proposed PFST-LSTM delivers promising performance at the beginning and produces the best overall performance in all the models.

Similar to MovingMNIST++, we visualize the prediction results in Fig. 9, where the mapping between reflectivity value and color is shown in Fig. 8. In Fig. 9, the first line denotes the ground truth and the other lines denote the prediction results by different models. We observe from the last few ground truth images that the shape of the region with high rainfall intensity is horizontal and continuous. However, the prediction results of all the models except for PFST-LSTM upward tilt or some of them are discontinuous. For instance, we mark the high echo value regions with black lines in the last ground-truth image, which forms a template. Accordingly, we put the template into the last column of predictions and find that the PFST-LSTM matches with it best, which is because of its special designs on position calibrations and spatial appearances preserver. Besides, only PredRNN and MIM model can generate the red part at the last prediction. However, their appearances and position for high echo value regions have a huge gap compared with ground truth. Moreover, we also observe that the CDNA and E3DLSTM tend to underestimate the high echo value regions (deep yellow or red parts in ground-truth), which are critical for precipitation nowcasting. It also explains why the metrics HSS and CSI are much smaller than other models while the threshold is 40.

4) Ablation Study: To investigate the effectiveness of our proposed mechanisms, we conduct the ablation studies and summarize the results in Table IV. Before analyzing the results, we first explain the name of the ablation models. In the table, Conv-LSTM denotes the architecture (in Section III-B) built with the ConvLSTM unit. Traj-LSTM is the TrajGRU variant by replacing GRU with LSTM. Again, ST-Traj-LSTM indicates the TrajLSTM model with an embedded spatial memory in each building unit. ST-LSTM and PF-LSTM represent the degeneration models of our PFST-LSTM by removing the spatial memory and pseudoflow subcomponent, respectively. We observe that PFST-LSTM outperforms all the models under different dBZ thresholds. Among all the models, the PF-LSTM achieves competitive MSE with PFST-LSTM but worse HSS and CSI, especially in the higher dBZ threshold. Besides, the ST-LSTM also yields much worse results than PFST-LSTM. The result validates the effectiveness of spatial memory and pseudo optical flow subcomponents. As for Traj-LSTM, which also leverages the spatial transformer to address the spatial nonalignment issue, its performance is also inferior to our PFST-LSTM because it does not have the special designs to preserve the spatial details. The model ST-Traj-LSTM, which consists of the spatial transformer and spatial memory mechanism, delivers even worse performance than Traj-LSTM. The reason may be that the spatial transformer and the spatial memory cannot be simultaneously well-trained. Conv-LSTM, which has neither preserves the spatial details nor tackles the misalignment issue, performs the worst.

Similarly, we also depict the performance of the different models w.r.t. the nowcasting lead time in Figs. 10 and 12. It can be seen that the proposed PFST-LSTM always performs the best. As for PF-LSTM, which excludes the spatial memory subcomponent, delivers competitive performance at 5 dBZ threshold but much worse prediction at 40 dBZ threshold than PFST-LSTM. Similar observations can be made for ST-LSTM. Again, we find that Traj-LSTM, ST-Traj-LSTM, and Conv-LSTM yield worse performance than our PFST-LSTM at different nowcasting lead times.

In Fig. 13, we also show an example to visually compare the models. We find that Conv-LSTM underestimates the area and strength of high echo value parts (red parts ground-truth image). Traj-LSTM better predicts the high echo values but the positions are not accurate. ST-TrajLSTM delivers better position prediction but the high echo values are underestimated. ST-LSTM produces a better high echo value region area (yellow parts), but the specific values are still underestimated (supposed to be red, instead of yellow). PF-LSTM shows the promising prediction, but the area of yellow part is overestimated and the dark yellow part is smaller than the ground truth. Overall, PFST-LSTM produces the best results, in terms of position and value accuracies. However, we find that it also cannot accurately predict the red part in the 10th prediction, namely the high value

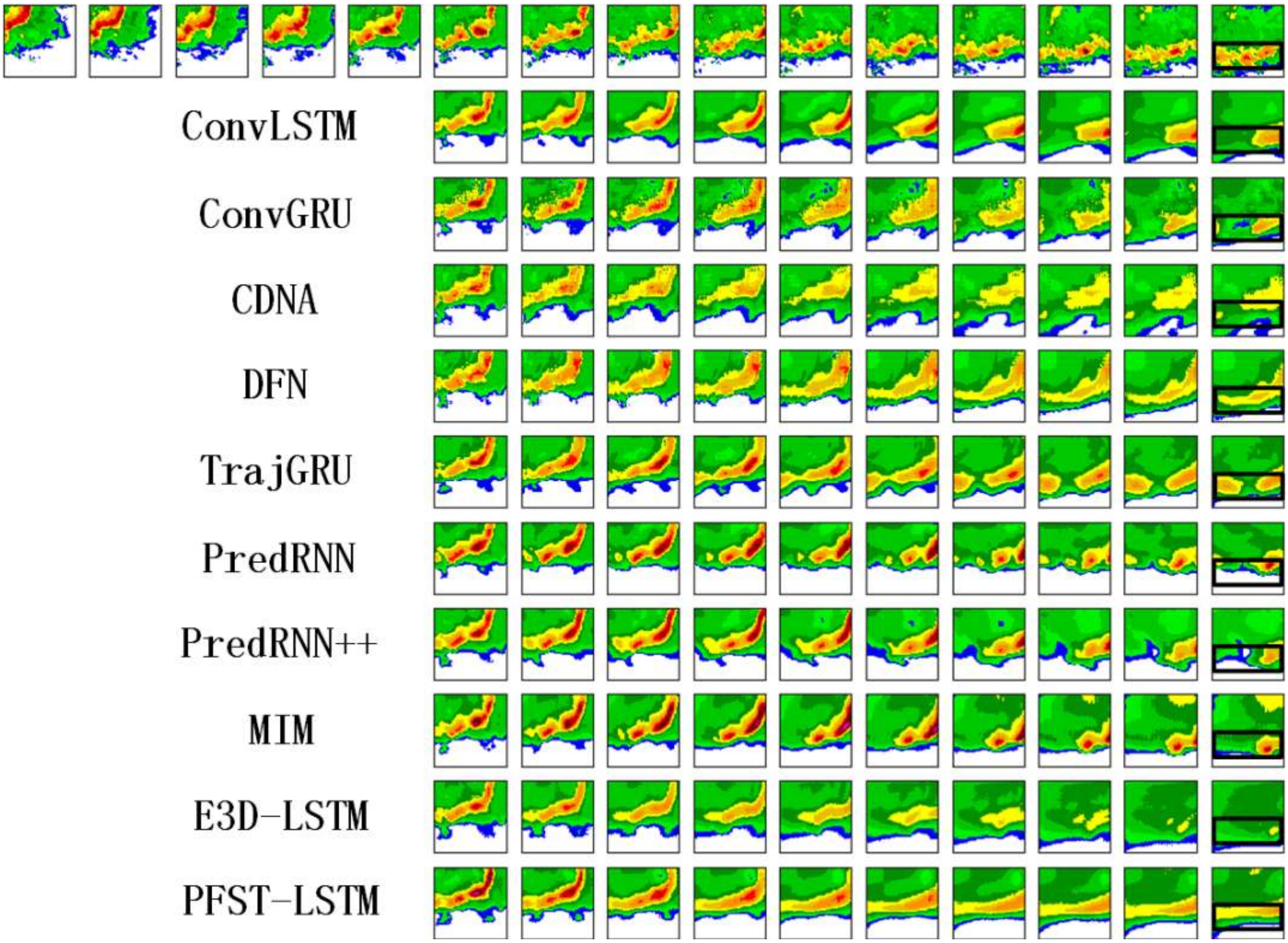


Fig. 9. Prediction results of all methods on an example from the CIKM AnalytiCup 2017 competition. The first five images in the first row are the input, and the remainders denote the ground-truth output (Best view in color).

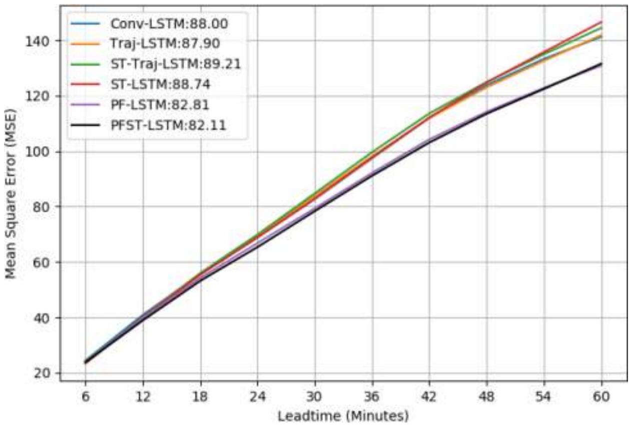


Fig. 10. Performance changes against different nowcast lead time in ablation study in terms of MSE. (Best view in color).

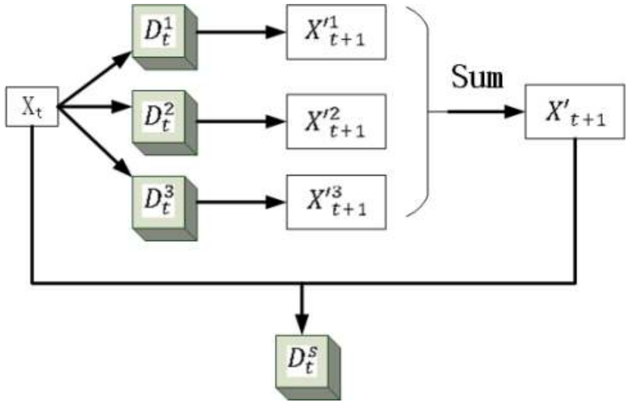
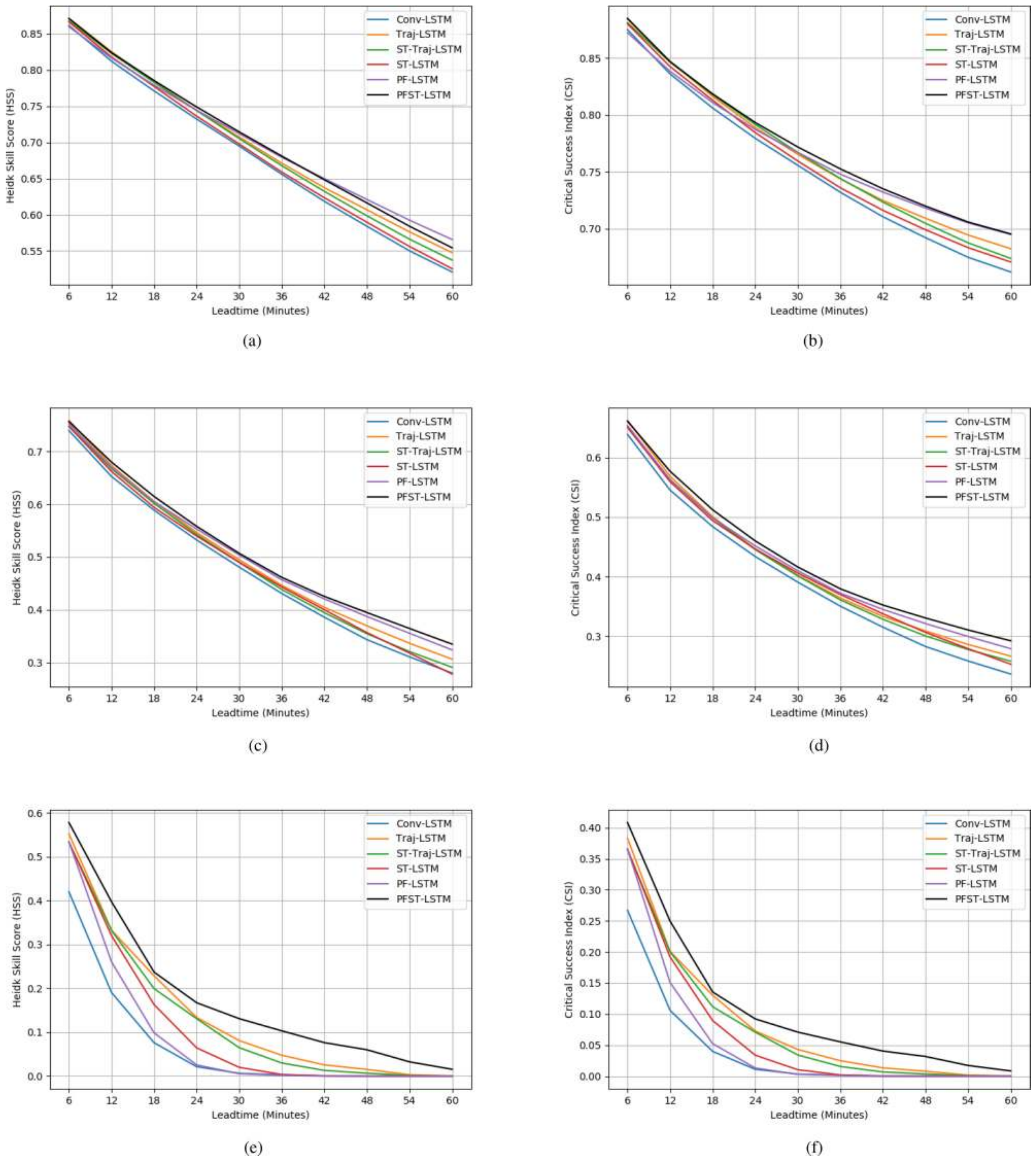


Fig. 11. Generation process of synthetic flow D^s.

part is still underestimated, which could be an interesting issue to study in the future.

5) Visualization and Understanding of Pseudoflow: One of the main contributions of our model is the introduction of pseudoflow to address the position misalignment issue. To investigate and understand the effectiveness of the component, we show and compare in Fig. 14, the pseudoflow and optical flow computed by our methods and predicted image sequence. The second line denotes the ground-truth optical flow computed from the predicted images [10]. As there are three layers utilized the



(a)

(b)

(c)

(d)

(e)

(f)

Fig. 12. Performance changes against different nowcast lead time in ablation study in terms of HSS and CSI scores. (Best view in color). (a) HSS $\tau = 5$. (b) CSI $\tau = 5$. (c) HSS $\tau = 20$. (d) CSI $\tau = 20$. (e) HSS $\tau = 40$. (f) CSI $\tau = 40$.

pseudo flows, we depict them, respectively, in the third, fourth, and fifth lines of the figure, which are denoted as $\{D^l\}_{l=1}^3$, respectively. In the last line, we utilize the $\{D^l\}_{l=1}^3$ to synthesize a total flow D^s. Specifically, the D^s is computed as Fig. 11. Given input X_t at the current time, we utilize the flow D_t^l to calculate the next image $X_{t+1}'^l$ from a different layer. Then, the three images from the layers are added together to obtain the final output X_{t+1}'. Finally, we compute D^s based on X_t and X_{t+1}'. From the figure, we make three interesting observations. (1) The $\{D^l\}_{l=1}^3$ in the layers are not equal to the ground-truth optical flow, but mimic it. Hence, we name $\{D^l\}_{l=1}^3$ as pseudoflow. (2) In $\{D^l\}_{l=1}^3$, the high reflectivity region is more clear in pseudoflow than that in the optical flow, especially for D^2 and D^3, which indicates that the movement of the high reflectivity region is carefully modeled. (3) We can see that the synthesized total flow D^s is quite similar to the ground truth optical flow,

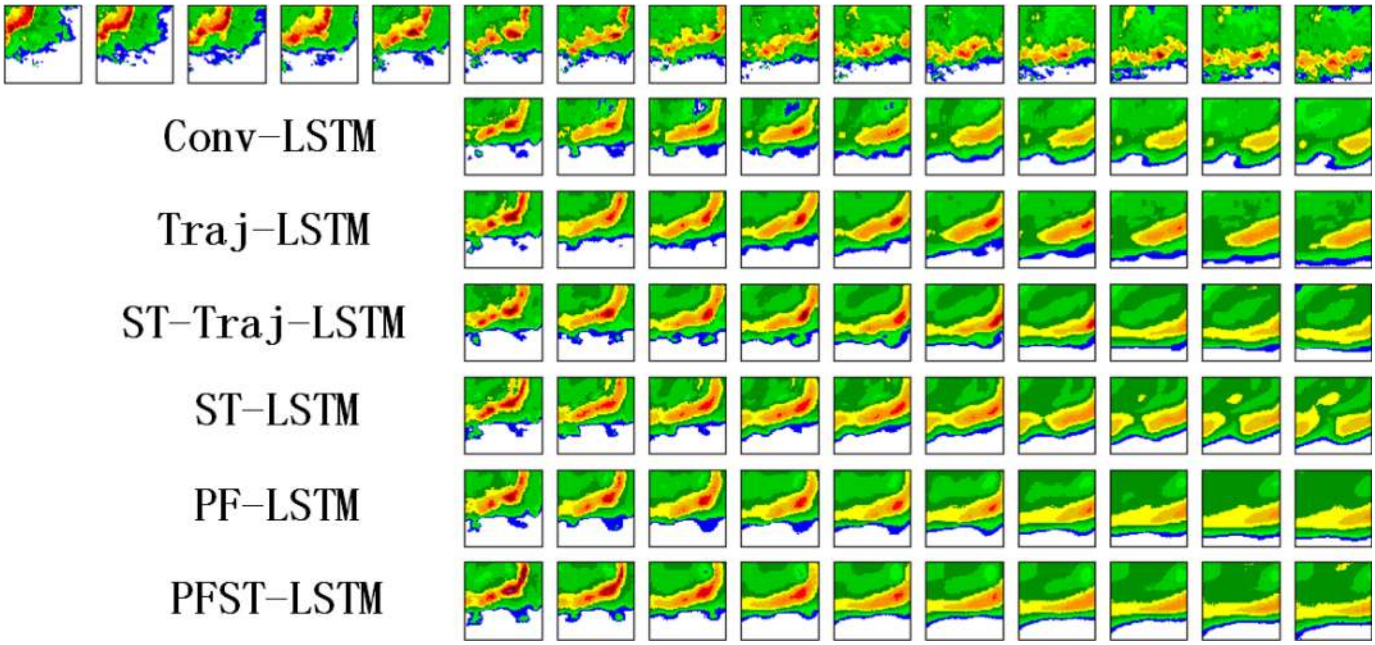


Fig. 13. Ablation results of on an example from the CIKM AnalytiCup 2017 competition. The first five images in the first row are the input, and the remainders denote the ground-truth output (Best view in color).

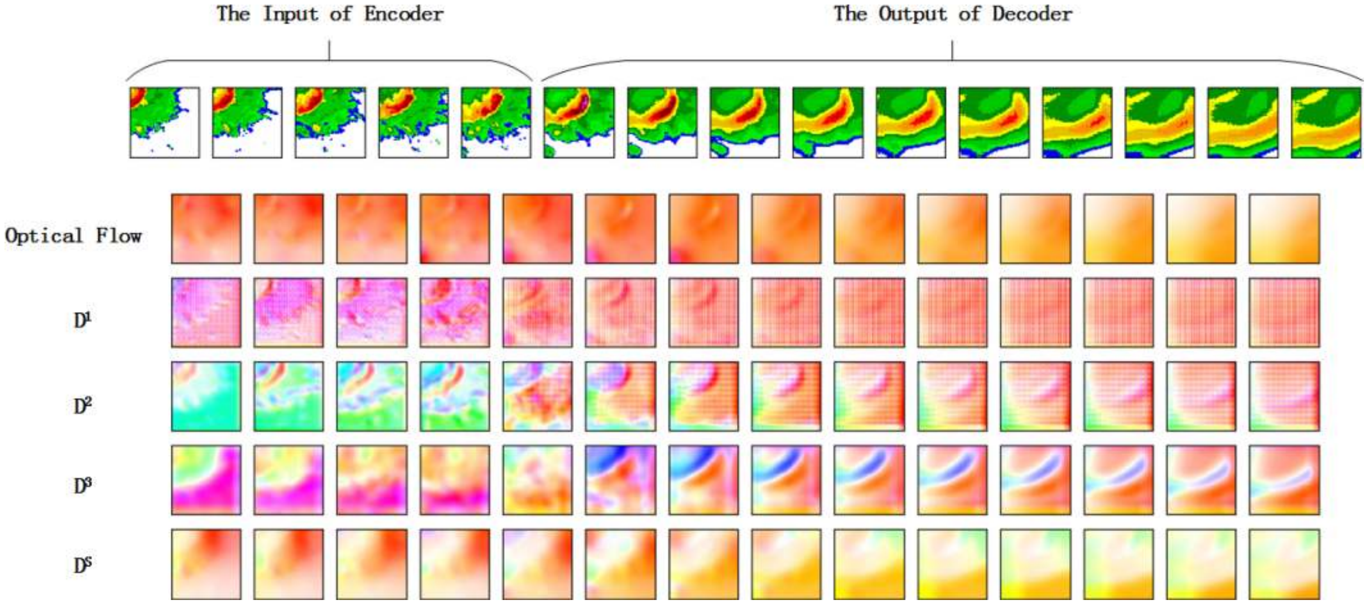


Fig. 14. Visualization of optical flow and pseudo flow from different times on an example from the CIKM AnalytiCup 2017 competition. The first line is the input and the output of our model. The second line denotes the optical flow of the first line. The third line to the fifth line is pseudo flows of different layers, respectively. The last line presents the synthetic flow D^s. (Best view in color).

which validates that pseudo flow indeed models the motion fields.

Finally, to demonstrate that the pseudoflow $\{D^l\}_{l=1}^3$ exactly calibrates the position of the hidden state at different kinds of the control gate, we compare the L1 norm and L2 norm between the input $X_t * W$ and hidden state $H_{t-1} * W$ with and without the pseudoflow. Table V shows the result. We can see that the value of L1 and L2 with pseudoflow calibration is smaller than that without the calibration in general. The results imply that the pseudoflow can align the hidden state toward the input, indicating its effectiveness in tackling the position mismatch issue.

V. Conclusion

In this article, we propose a novel convolutional LSTM unit for precipitation nowcasting, namely PFST-LSTM, which addresses the position mismatch issue and the lack of a spatial appearance preserver. A new sequence-to-sequence prediction

TABLE V
COMPARISON OF THE DIFFERENCE BETWEEN THE HIDDEN STATE AND INPUT AFTER CONVOLUTION WITH AND WITHOUT THE PSEUDOFLOW IN TERMS OF L1 NORM AND L2 NORM

Control Gate	Measurment	Condition	Layer 1	Layer 2	Layer 3
Input Gate	L1 Norm	Without Pseudo Flow	0.8935	1.9593	1.5521
		With Pseudo Flow	**0.8737**	**1.9505**	**1.4987**
	L2 Norm	Without Pseudo Flow	1.5560	**8.3136**	3.9716
		With Pseudo Flow	**1.4797**	8.3632	**3.7407**
Modulation Gate	L1 Norm	Without Pseudo Flow	0.4317	1.2081	1.3337
		With Pseudo Flow	**0.4140**	**1.2020**	**1.2865**
	L2 Norm	Without Pseudo Flow	0.4308	4.6381	2.8832
		With Pseudo Flow	**0.4003**	**4.6296**	**2.6856**
Forget Gate	L1 Norm	Without Pseudo Flow	0.7058	1.9608	1.9870
		With Pseudo Flow	**0.6878**	**1.9384**	**1.9261**
	L2 Norm	Without Pseudo Flow	1.0489	8.5831	6.0863
		With Pseudo Flow	**0.9970**	**8.5389**	**5.8136**
Output Gate	L1 Norm	Without Pseudo Flow	0.6360	1.8078	1.2268
		With Pseudo Flow	**0.6226**	**1.7957**	**1.1542**
	L2 Norm	Without Pseudo Flow	0.7941	7.9140	2.4407
		With Pseudo Flow	**0.7584**	**7.8898**	**2.1593**

architecture is also developed. Extensive experimental results have been reported, which demonstrate the superiority of the proposed model over the state-of-the-art approaches. Experimental results have validated its effectiveness. In the future, we would like to address the underestimation issue of high echo value regions.

REFERENCES

[1] X. Shi *et al.*, “Deep learning for precipitation nowcasting: A benchmark and a new model,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5617–5627.

[2] G. Ayzel, M. Heistermann, A. Sorokin, O. Nikitin, and O. Lukyanova, “All convolutional neural networks for radar-based precipitation nowcasting,” *Procedia Comput. Sci.*, vol. 150, pp. 186–192, 2019.

[3] J. Sun, “Use of NWP for nowcasting convective precipitation: Recent progress and challenges,” *Bull. Amer. Meteorol. Soc.*, vol. 95, no. 95, pp. 409–426, 2014.

[4] P. Bauer, A. Thorpe, and G. Brunet, “The quiet revolution of numerical weather prediction,” *Nature*, vol. 525, no. 7567, 2015, Art. no. 47.

[5] C. Wang and Y. Hong, “Application of spatiotemporal predictive learning in precipitation nowcasting,” in *AGU Fall Meet. Abstr.*, 2018, Art. no. H31H-1988.

[6] L. Tian, X. Li, Y. Ye, P. Xie, and Y. Li, “A generative adversarial gated recurrent unit model for precipitation nowcasting,” *IEEE Geosci. Remote Sens. Lett.*, vol. 17, no. 4, pp. 601–605, Apr. 2020.

[7] E. Shi, Q. Li, D. Gu, and Z. Zhao, “A method of weather radar echo extrapolation based on convolutional neural networks,” in *Proc. Int. Conf. Multimedia Model.*, 2018, pp. 16–28.

[8] W. Wong, L. H. Yeung, Y. Wang, and M. Chen, “Towards the blending of NWP with nowcast–operation experience in B 08 FDP,” in *Proc. WMO Symp. Nowcasting*, vol. 30, 2009, pp. 1–15.

[9] G. Ayzel, M. Heistermann, and T. Winterrath, “Optical flow models as an open benchmark for radar-based precipitation nowcasting (rainymotion v0. 1),” *Geoscientific Model Develop.*, vol. 12, no. 4, pp. 1387–1402, 2019.

[10] W.-C. Woo and W.-K. Wong, “Operational application of optical flow techniques to radar-based rainfall nowcasting,” *Atmosphere*, vol. 8, no. 3, 2017, Art. no. 48.

[11] G. Wang, W. Wong, L. Liu, and H. Wang, “Application of multi-scale tracking radar echoes scheme in quantitative precipitation nowcasting,” *Adv. Atmos. Sci.*, vol. 30, no. 2, pp. 448–460, 2013.

[12] Z. C. Lipton, J. Berkowitz, and C. Elkan, “A critical review of recurrent neural networks for sequence learning,” 2015, *arXiv: 1506.00019*.

[13] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.

[14] K. Cho, B. van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” in *Proc. Conf. Emp. Methods Natural Lang. Process.*, 2014, pp. 1724–1734.

[15] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, “Convolutional LSTM network: A machine learning approach for precipitation nowcasting,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 802–810.

[16] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, “Loss functions for image restoration with neural networks,” *IEEE Trans. Comput. Imag.*, vol. 3, no. 1, pp. 47–57, Mar. 2017.

[17] Q.-K. Tran and S.-k. Song, “Computer vision in precipitation nowcasting: Applying image quality assessment metrics for training deep neural networks,” *Atmosphere*, vol. 10, no. 5, 2019, Art. no. 244.

[18] Y. Wang, M. Long, J. Wang, Z. Gao, and S. Y. Philip, “PredRNN: Recurrent neural networks for predictive learning using spatiotemporal LSTMS,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 879–888.

[19] N. Ballas, L. Y. C. Pal, and A. Courville, “Delving deeper into convolution networks for learning video representation,” 2015, *arXiv: 1511.06432*.

[20] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” in *Proc. Conf. Neural Inf. Process. Syst. Workshop Deep Learn.*, 2014, pp. 1–9.

[21] M. Jaderberg *et al.*, “Spatial transformer networks,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2017–2025.

[22] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, “FlowNet 2.0: Evolution of optical flow estimation with deep networks,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2462–2470.

[23] Y. Wang, Z. Gao, M. Long, J. Wang, and S. Y. Philip, “PredRNN++: Towards a resolution of the deep-in-time dilemma in spatiotemporal predictive learning,” in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5123–5132.

[24] Y. Wang, L. Jiang, M.-H. Yang, L.-J. Li, M. Long, and L. Fei-Fei, “Eidetic 3D LSTM: A model for video prediction and beyond,” in *Proc. Int. Conf. Learn. Representations*, 2018, pp. 1–14.

[25] C. Finn, I. Goodfellow, and S. Levine, “Unsupervised learning for physical interaction through video prediction,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 64–72.

[26] X. Jia, B. De Brabandere, T. Tuytelaars, and L. V. Gool, “Dynamic filter networks,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 667–675.

[27] Y. Wang, J. Zhang, H. Zhu, M. Long, J. Wang, and S. Y. Philip, “Memory in memory: A predictive neural network for learning higher-order non-stationarity from spatiotemporal dynamics,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 9146–9154.

[28] W. Byeon, Q. Wang, R. Kumar Srivastava, and P. Koumoutsakos, “ContextVP: Fully context-aware video prediction,” in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 753–769.

[29] F. A. Reda *et al.*, “SDC-Net: Video prediction using spatially-displaced convolution,” in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 718–733.

[30] M. Mathieu, C. Couprie, and Y. LeCun, “Deep multi-scale video prediction beyond mean square error,” in *Proc. 4th Int. Conf. Learn. Representations*, 2016, pp. 1–14.

[31] C. Vondrick, H. Pirsiavash, and A. Torralba, “Generating videos with scene dynamics,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 613–621.

[32] Z. Long, Y. Lu, X. Ma, and B. Dong, “PDE-Net: Learning PDES from data,” in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 3208–3216.

[33] V. L. Guen and N. Thome, “Disentangling physical dynamics from unknown factors for unsupervised video prediction,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11 474–11 484.

[34] R. J. Hogan, C. A. Ferro, I. T. Jolliffe, and D. B. Stephenson, “Equitability revisited: Why the equitable threat score is not equitable,” *Weather Forecasting*, vol. 25, no. 2, pp. 710–726, 2010.

Chuyao Luo received the bachelor’s degree in Internet of Things engineering from Dalian Maritime University, Dalian, China, in 2017. He is currently working toward the Ph.D. degree with Harbin Institute of Technology, Shenzhen, China.

His research includes data mining, computer vision, time-series data prediction, and clustering.

Xutao Li received the bachelor’s degree in computer science from Lanzhou University of Technology, Lanzhou, China, in 2007, and the master’s and Ph.D. degrees in computer science from Harbin Institute of Technology, Shenzhen, China, in 2009 and 2013, respectively.

He is currently an Associate Professor at the Harbin Institute of Technology, Shenzhen, China. His research interests include data mining, machine learning, graph mining, and social network analysis, especially tensor-based learning, and mining algorithms.

Yunming Ye received the Ph.D. degree in computer science from Shanghai Jiao Tong University, Shanghai, China.

He is currently a Professor with the Harbin Institute of Technology, Shenzhen, China. His research interests include data mining, text mining, and ensemble learning algorithms.