IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 2, 2020, accepted August 26, 2020, date of publication September 8, 2020, date of current version September 21, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3022774

Partial Convolutional LSTM for Spatiotemporal Prediction of Incomplete Data

HYESOOK SON, (Student Member, IEEE), AND YUN JANG, (Member, IEEE)
Department of Computer Engineering, Sejong University, Gwangjin-gu 05006, South Korea

Corresponding author: Yun Jang (jangy@sejong.edu)

This work was supported in part by the Basic Research Program through the National Research Foundation of Korea (NRF) funded by the MSIT under Grant 2019R1A4A1021702, and in part by the Institute of Information & communications Technology Promotion (IITP) funded by the Korea Government [Ministry of Science and ICT (MSIT)] (Development of Big data and AI based Energy New Industry type Distributed resource Brokerage System) under Grant 2019-0-00374.

ABSTRACT Advanced data analysis techniques facilitate data-driven spatiotemporal prediction in various fields. However, in real-world data, missing values are inevitable, which causes the data incomplete and makes predictions more challenging. Although we can train complex spatiotemporal correlations with deep learning techniques, most deep learning networks require data without any missing values. In this paper, we propose a novel deep learning framework that manages missing values in the grid-based data structure. We design a partial convolutional long-short-term-memory (PConvLSTM) by combining partial convolution for inpainting and convolutional long-short-term-memory (ConvLSTM) for spatiotemporal prediction. We treat incomplete spatiotemporal data with the partial convolution and train spatiotemporal dependencies with the ConvLSTM structure. The trained PConvLSTM can predict continuous spatial data with missing regions in incomplete input data. We also train the network using incomplete spatiotemporal data without ground truth to enhance practicality. Existing deep learning networks to interpolate missing data are mostly trained by applying ground truth data without missing regions. We show that PConvLSTM achieves higher prediction accuracies compared to ConvLSTM for incomplete data without ground truth.

INDEX TERMS Partial convolution, LSTM, incomplete data.

I. INTRODUCTION

Many researchers have collected spatiotemporal data and analyzed the information in various fields, such as weather, traffic, soil, satellite, and video. Lately, the demand for spatiotemporal data prediction increases dramatically due to the advance of deep learning techniques. Dixon *et al.* [1] present a deep learning technique for data-driven spatiotemporal prediction, which trains nonlinear spatiotemporal correlations. Xingjian *et al.* [2] introduce a convolutional Long-Short-Term-Memory (ConvLSTM) that captures spatial and temporal dependencies by integrating Long-Short-Term-Memory (LSTM) structures with convolutional operations. However, the spatiotemporal prediction is still challenging due to the complex spatiotemporal correlation.

Generally, real-world data can be erratic, and missing values almost inevitably occur, which causes the data incomplete. For example, air quality data is one of the spatiotemporal data, which is mostly measured at irregularly located stations. When data obtained from arbitrary locations are transformed in a gridded data structure, empty cells tend to be generated, resulting in the spatially irregular distribution of data. Besides, sampling time for measuring data at each station may be irregular due to the equipment failure. For these reasons, it is not always achievable to obtain complete data that are without any missing data. Incomplete spatiotemporal data, such as air quality data, degenerate prediction model performance, and result in biased predictions. Therefore, it is challenging to train deep learning models to predict incomplete spatiotemporal data. For more accurate predictions with such spatiotemporal data, we must overcome the following challenges.

- C1 Data must be handled, although the data measure locations are irregularly distributed, as shown in Figure 1 (a).
- C2 Data measured at arbitrary time intervals must be supervised, as illustrated in Figure 1 (b).
- C3 Spatio and temporal irregularities in data must be considered at the same time, as presented in Figure 1 (c).

The associate editor coordinating the review of this manuscript and approving it for publication was Michael Lyu.

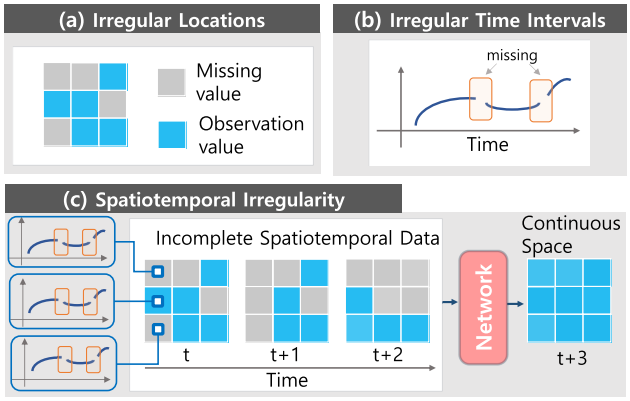


FIGURE 1. Spatiotemporal irregularity. (a) The sampling locations are irregularly distributed. (b) The sampling time intervals are irregular. (c) The spatial locations for data sampling vary randomly over time. The proposed deep learning network predicts future continuous spatial data from the past incomplete spatiotemporal data.

C4 We must be able to train the network, although there is no ground truth.

Spatially irregular missing data stated in C1 make it hard to estimate spatial data distribution [3]. To obtain continuous spatial data, we can determine the missing regions with deep learning networks. For example, deep learning models for inpainting and denoising restore damaged regions of a 2D grid structured image. In the image inpainting study, Liu *et al.* [4] have proposed a partial convolution in which the convolutional result is only affected by valid pixels. Most inpainting deep learning studies, such as partial convolution, utilize images that are intact in ground truth for the training. In this work, however, we can only use irregularly sampled data in the space. Since we do not have complete data, it is not possible to train deep learning networks using ground truth, as in the previous research. To resolve the ground truth problem, some researchers have reconstructed noisy images to train models without ground truth [5], [6]. However, the researchers have assumed a particular statistical distribution of noise, such as Gaussian noise. Therefore, the model training with the assumption of the noise distribution becomes frequently impractical because the actual noise distribution in the real data may differ from the assumed noise distribution [7].

As pointed in C2, it is not easy to apply the data sampled at irregular time intervals directly in the existing prediction models. Before the predictive model training, we need to estimate the missing data mostly by imputation techniques such as ARIMA and interpolation in the preprocessing stage. Since the imputation technique determines the data used for the training, the deep learning outputs change consequently. In this work, we do not know the actual values in the missing data, which makes it challenging to comprehend how the imputation affects deep learning trainings. Therefore, it is crucial to develop a deep learning network that predicts the data without imputation as a preprocessing.

As indicated in C3, Figure 1 (c) illustrates spatiotemporal data in which the spatial sampling locations fluctuate randomly over time. Even if the data sampling locations are arbitrary due to missing values, we maintain a grid-based data structure. We process irregularly sampled spatiotemporal data in the grid structure, as shown in Figure 1 (c). Spatiotemporal deep learning model, such as ConvLSTM proposed by Yu *et al.* [8], is trained with data in a grid structure without any missing value. In this work, however, we cannot use existing deep learning models such as ConvLSTM because the available data positions vary irregularly in space and time. We also find it difficult to obtain complete data utilized as ground truth, as stated in C4. Ground truth data can be very expensive or even impossible to obtain. As mentioned earlier, while we can train deep learning models for image denoising without ground truth, studies on other deep learning models that train spatiotemporal data without ground truth are not spotted easily.

In this paper, we study a predictive deep learning model trained with incomplete spatiotemporal data. Figure 1 (c) represents the grid-based spatiotemporal data that contain the missing values and illustrates our research goals. We propose a deep learning network that predicts future continuous spatial data from the past incomplete spatiotemporal data. We utilize incomplete spatiotemporal data to train the grid-based network without ground truth. Grid-based networks such as ConvLSTM have good performance only for gird structured data and all input data must be the same size. Many researchers have studied graph-based models to overcome the limitations of the grid-based model. The researchers have treated the data with irregular sampling patterns as graph structure data. Spatial-temporal graph neural network (STGNN) is trained with spatiotemporal data in a graph structure, but most STGNNs predict only graph nodes [9]. However, since we intend to construct a prediction model for continuous space and the input of the existing STGNN is a graph with fixed nodes, it is not easy to predict incomplete spatiotemporal data with the existing STGNN framework. Therefore, in this study, we tackle the C1-C4 mentioned above with a grid-based network rather than an STGNN framework.

Graler *et al.* [10] report that spatiotemporal interpolation using both past and future data together enables more accurate prediction than spatial interpolation. Therefore, we borrow the idea by Graler *et al.*, and we propose a partial convolutional LSTM, which is called PConvLSTM, by combining partial convolution and ConvLSTM. PConvLSTM extracts spatial patterns from incomplete data with partial convolution and controls the information flow with LSTM. Since PConvLSTM handles incomplete data at the network layer, we do not need to perform the imputation as a preprocessing. We also introduce an algorithm that trains the network without ground truth. We apply our model to the radar echo dataset and Moving MNIST dataset. For both datasets, our technique produces higher prediction

performance than existing approaches. The contributions of this paper are summarized as follows.

- We propose a deep learning network capable of spatiotemporal prediction with incomplete data. The proposed deep learning network extracts spatial patterns from data sampled at irregular locations.
- Our deep learning network predicts future continuous spatial data from data with missing values, but we do not perform any imputation or interpolation before training the deep learning network.
- We train the network without ground truth for spatiotemporal prediction. To the best of our knowledge, we first attempt to train a deep learning network to predict incomplete spatiotemporal data without ground truth.

II. RELATED WORK

Many researchers have studied deep learning models for image restorations such as super-resolution [11], denoising [12], [13], and inpainting [4]. Especially, Liu *et al.* [4] have introduced a partial convolution to ensure that the network output depends only on valid pixels. They have trained the network to fill irregular holes in the image with partial convolution and mask-update steps. Yeh *et al.* [14] have proposed a deep convolutional generative adversarial network (DCGAN) for image inpainting and introduced prior loss so that DCGAN realistically estimates missing pixels. Yu *et al.* [15] have proposed a deep learning-based user-guided system for image inpainting. They have modified the partial convolution to make a gated convolution and accumulated the gated convolution to form a generative adversarial network. The gated convolution allows us to train the optimal mask without the rule-based mask update step in the partial convolution. Recently, deep learning architectures for video inpainting have been introduced. Wang *et al.* [16] have combined a 3D completion network (3DCN) and 3D-2D combined completion network (CombCN) to recover missing regions of video frames. 3DCN is a 3D CNN for capturing the temporal structure of the video. 3DCN improves CombCN performance by delivering temporal information to CombCN for inpainting. CombCN inpaints each frame with a 2D convolution operation and supports the 3DCN to capture the temporal structure. Kim *et al.* [17] have proposed a 3D-2D encoder-decoder network architecture using ConvLSTM for the frame inpainting in a damaged video. They have designed the network to train aggregated feature maps from the past and future frames for inpainting.

Most data recovery tasks such as denoising and inpainting assume that ground truth exists and use it for network training. However, occasionally it is not easy to obtain ground truth data. Therefore, some researchers have proposed deep learning architectures that can be trained without ground truth. Lehtinen *et al.* [5] have extracted inputs and targets from corrupted distributions and applied them in the deep learning training. Soltanayev and Chun [6] have trained the network to optimize Stein's unbiased risk estimator (SURE), assuming a gaussian noise model. They used CNN and proposed a loss function to train CNN without ground truth [5], [6]. Deep learning techniques, which do not use ground truth, have been examined with image data but have not been extended to spatiotemporal data. In this paper, we devise a network to handle not only loss but also missing regions within the layer. We also train the proposed network with spatiotemporal data without ground truth.

Deep learning models for predicting spatiotemporal data have also been studied. Xingjian *et al.* [2] have introduced ConvLSTM to train spatiotemporal patterns. They have shown that ConvLSTM can predict with higher performance than fully connected LSTM (FC-LSTM). In various research fields, researchers have been using ConvLSTM. Song *et al.* [18] have proposed Pyramid Dilated Bidirectional ConvLSTM (PDB-ConvLSTM) for the video saliency detection. In PDB-ConvLSTM, the convolution operation in ConvLSTM is substituted by the dilated convolution, and multi-scale spatial features are extracted using various dilation factors. Besides, PDB-ConvLSTM captures temporal dependency in both directions by exchanging information between forward and backward ConvLSTM units. Zhao *et al.* [19] have presented ConvLSTM for predicting tongue movements. They have trained the ConvLSTM with ultrasound video data and showed higher predictive performance than 3DCNN. Yuan *et al.* [20] have proposed a Hetero-ConvLSTM framework for predicting traffic accidents. They have included spatial graph features to consolidate the spatial relationship of roads and environmental features such as weather and road conditions in the model training. Moreover, they have used the moving window to divide the prediction area into several regions. The Hetero-ConvLSTM is trained for each sub-region, and they have merged all outputs to obtain the final prediction. Zhe *et al.* [21] have proposed a model for gesture recognition. To apply short-term spatiotemporal features and long-term spatiotemporal features, their model combines 3D-CNN and ConvLSTM layers. Luo *et al.* [22] have integrated ConvNet and ConvLSTM as an auto-encoder for anomaly detection in videos. Wang *et al.* [23] have derived a new spatiotemporal LSTM (ST-LSTM) unit using a memory capable of storing both spatial appearance and temporal changes. ConvLSTM-based models utilize data with regular grid structures without missing values. However, spatiotemporal data measured with sensors, such as particulate matters and solar power generation, are collected at irregular locations. For the prediction of such data, grid mapping is required to train existing networks. However, irregularities result in missing regions after mapping the data points on a grid. Lee and Shin [24] have introduced a technique for estimating missing regions with inverse distance weighting interpolation and deep learning training. However, interpolation might produce biased results depending on techniques [25]. When training deep learning with biased data, biased outputs are produced. Therefore, it is necessary to design a prediction network without preprocessing steps like an interpolation.

For the prediction of non-regular data, STGNN has been proposed, which employs graph-structured data for the training rather than grid-structured data. Structural-RNN (S-RNN) [26] predicts node labels in the spatiotemporal graph by combining the RNN modeling node factor and the RNN modeling edge factor. Li *et al.* [27] have trained the Diffusion Convolutional Recurrent Neural Network (DCRNN) by modeling the data collected from the sensors on the road in the graph structure as a diffusion process. Yu *et al.* [8] have presented a spatio-temporal convolutional block that computes the graph convolution and gated temporal convolution to process both temporal and spatial information of the graph. Lin *et al.* [28] have constructed a graph of sparse monitoring stations in STGNN framework and predicted the air quality at each station. Most STGNNs including the studies introduced above, only predict the value of each node [9]. However, our interest is not just predicting the nodes of the graph, but also predicting values in the continuous space. Besides, if the spatial locations where the data are sampled vary over time, it is not straightforward to apply existing STGNNs such as S-RNN, DCRNN, STGCN. Therefore, in this work, we design a grid-based network with incomplete spatiotemporal data. Moreover, the new network is trained without ground truth for the predictions in the continuous space.

III. METHODOLOGY

In this work, we propose a deep learning framework for predicting incomplete spatiotemporal data. We introduce the deep learning layers, including partial convolution and ConvLSTM that are the bases of the proposed network, and then we specify the equations of the proposed network in Section III-A. We also describe how to train the network without ground truth in Section III-B. We define a loss function in Section III-C to optimize the network that is suitable for predicting continuous spatial data.

A. NETWORK STRUCTURE

1) PARTIAL CONVOLUTION

The partial convolution is proposed for image inpainting by Liu *et al.* [4]. The input of the partial convolution consists of a randomly damaged image and a binary mask, including 1 for valid pixels and 0 for invalid pixels that indicate the hole locations in the image. The partial convolution is computed with the image and mask as follows.

$$x' = \begin{cases} \mathbf{W}^T \left(\mathbf{X} \odot \mathbf{M}\right) \dfrac{\text{sum}\left(\mathbf{1}\right)}{\text{sum}\left(\mathbf{M}\right)} + b, & \text{if sum}\left(\mathbf{M}\right) > 0 \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where $\odot$ indicates the Hadamard product. $\mathbf{X}$ is the pixel value for the convolution window, and $\mathbf{M}$ is the mask corresponding to the window position. $\mathbf{W}$ and b are the convolution filter weight and bias, respectively, and $\mathbf{1}$ has the same shape as $\mathbf{M}$ but all elements are 1. $\frac{\text{sum}(\mathbf{1})}{\text{sum}(\mathbf{M})}$ is a factor to scale only valid pixels. In Equation (1), multiplying the input by the mask sets the invalid pixel values to 0s, which excludes the effect of the invalid pixels. In addition, the partial convolution applies an appropriate scaling factor for inputs where the valid and invalid pixels are unbalanced. Therefore, the partial convolution becomes a convolution that depends only on the valid pixel values.

The partial convolution includes a mask update step. When we perform the partial convolution on the input image and input mask, the position of the valid pixel in the output varies according to the input image. To propagate updated mask, the partial convolution updates the valid pixel position. If there is more than one valid pixel in the convolution window of Equation (1), the pixel corresponding to the convolution window is set to a valid pixel because it includes the computation with valid pixel values. We can update the mask as follows.

$$m' = \begin{cases} 1, & \text{if sum}\left(\mathbf{M}\right) > 0 \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

2) CONVOLUTIONAL LONG-SHORT-TERM-MEMORY (CONVLSTM)

To improve the fully connected LSTM (FC-LSTM), which does not include spatial correlation, ConvLSTM is designed as a spatiotemporal prediction network to include all spatiotemporal information. ConvLSTM is trained for spatial correlation with the convolution operations in state-to-state and input-to-state transitions, and for temporal correlation with cyclic structures. The equations for ConvLSTM are as follows.

$$i_t = \sigma\left(\mathcal{W}_{xi} * \mathcal{X}_t + \mathcal{W}_{hi} * \mathcal{H}_{t-1} + b_i\right) \tag{3}$$

$$f_t = \sigma\left(\mathcal{W}_{xf} * \mathcal{X}_t + \mathcal{W}_{hf} * \mathcal{H}_{t-1} + b_f\right) \tag{4}$$

$$\mathcal{C}_t = f_t \odot \mathcal{C}_{t-1} + i_t \odot tanh\left(\mathcal{W}_{xc} * \mathcal{X}_t + \mathcal{W}_{hc} * \mathcal{H}_{t-1} + b_c\right) \tag{5}$$

$$o_t = \sigma\left(\mathcal{W}_{xo} * \mathcal{X}_t + \mathcal{W}_{ho} * \mathcal{H}_{t-1} + b_o\right) \tag{6}$$

$$\mathcal{H}_t = o_t \odot tanh\left(\mathcal{C}_t\right), \tag{7}$$

where $*$, $\odot$, and σ represent the convolution operator, Hadamard product, and element-wise sigmoid function, respectively. The input $\mathcal{X}_t$, cell state $\mathcal{C}_t$, and hidden state $\mathcal{H}_t$ are 3D tensors. $\mathcal{C}_{t-1}$ indicates the previous long-term state, and $\mathcal{H}_{t-1}$ is the previous short-term-state. The input gate i_t, forget gate f_t, and output gate o_t are the 3D tensors for the cell and control the information flow along with the cell state. $\mathcal{W}$ is the convolutional kernel, and b is the bias. As represented in Equation (3)-(7), the hidden states are updated by the convolution of input $\mathcal{X}_t$ and the convolution of the previous hidden state $\mathcal{H}_{t-1}$. The convolution does not differentiate observations and missing values. Therefore, ConvLSTM takes missing values as information with observations. In this paper, we propose a ConvLSTM network that is able to distinguish missing values. We design the network that updates the hidden state only on observations.

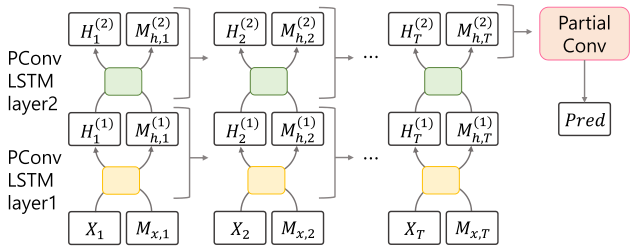


FIGURE 2. The architecture of our network configured with two PConvLSTM layers and one partial convolution layer.

3) PARTIAL CONVOLUTIONAL LONG-SHORT-TERM-MEMORY NETWORK

In this work, we propose a Partial Convolutional Long-Short-Term-Memory Network (PConvLSTM) by combining partial convolution with ConvLSTM, which means that we replace the convolution with the partial convolution in ConvLSTM. To apply the partial convolution in PConvLSTM, we create a mask that indicates the positions of valid pixels not only in the input but also in the hidden state. Figure 2 illustrates our network architecture with two PConvLSTM layers. As presented in Figure 2, the layer receives input and mask and updates them. We add a partial convolutional layer in the last so that the network output is depend only on observations.

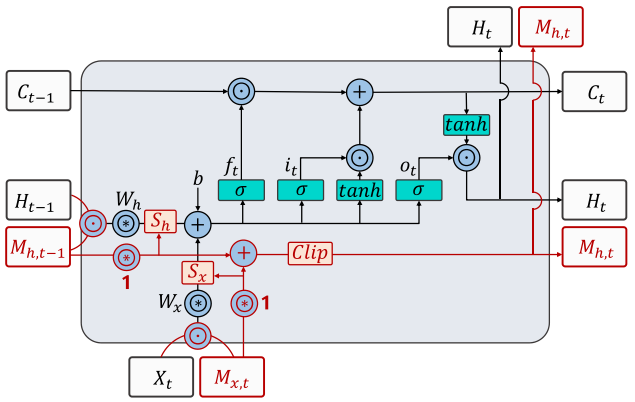


FIGURE 3. Structure of our PConvLSTM. The red lines indicate the differences from the structure of ConvLSTM.

Figure 3 shows the structure of the proposed PConvLSTM. The cell of PConvLSTM is composed of the input, input mask, hidden state, a mask of hidden state, and cell state. The input $\mathcal{X}_t$, cell state $\mathcal{C}_t$, hidden state $\mathcal{H}_t$, input gate i_t, forget gate f_t, and output gate o_t of PConvLSTM are the 3D tensors that compose a cell. $\mathcal{M}_{x,t}$ represents a mask corresponding to $\mathcal{X}_t$, and $\mathcal{M}'_{x,t}$ indicates an updated mask. Similarly, $\mathcal{M}_{h,t-1}$ is a mask of $\mathcal{H}_{t-1}$ and $\mathcal{M}'_{h,t-1}$ represents an updated $\mathcal{M}_{h,t-1}$. $\mathcal{M}_{h,t}$ is a mask of $\mathcal{H}_t$ updated from $\mathcal{M}'_{h,t-1}$. Similar to the partial convolution, we calculate the mask scaling factors $\mathcal{S}_x$ and $\mathcal{S}_h$ to properly scale irregularly distributed data. The equations of PConvLSTM are as follows.

$$\mathcal{X}'_t = \mathcal{X}_t \odot \mathcal{M}_{x,t} \tag{8}$$

$$\mathcal{H}'_{t-1} = \mathcal{H}_{t-1} \odot \mathcal{M}_{h,t-1} \tag{9}$$

$$\mathcal{M}'_{x,t} = \mathbf{1} * \mathcal{M}_{x,t} \tag{10}$$

$$\mathcal{M}'_{h,t-1} = \mathbf{1} * \mathcal{M}_{h,t-1} \tag{11}$$

$$\mathcal{S}_x = \frac{ks_x}{\left(\mathcal{M}'_{x,t} + \epsilon\right)} \odot clip\left(\mathcal{M}'_{x,t}\right) \tag{12}$$

$$\mathcal{S}_h = \frac{ks_h}{\left(\mathcal{M}'_{h,t-1} + \epsilon\right)} \odot clip\left(\mathcal{M}'_{h,t-1}\right) \tag{13}$$

$$i_t = \sigma\Big(\left(\mathcal{W}_{xi} * \mathcal{X}'_t\right) \odot \mathcal{S}_x + \left(\mathcal{W}_{hi} * \mathcal{H}'_{t-1}\right) \odot \mathcal{S}_h + b_i \Big) \tag{14}$$

$$f_t = \sigma\Big(\left(\mathcal{W}_{xf} * \mathcal{X}'_t\right) \odot \mathcal{S}_x + \left(\mathcal{W}_{hf} * \mathcal{H}'_{t-1}\right) \odot \mathcal{S}_h + b_f \Big) \tag{15}$$

$$\mathcal{C}_t = f_t \odot \mathcal{C}_{t-1} + i_t \odot tanh\Big(\left(\mathcal{W}_{xc} * \mathcal{X}'_t\right) \odot \mathcal{S}_x + \left(\mathcal{W}_{hc} * \mathcal{H}'_{t-1}\right) \odot \mathcal{S}_h + b_c \Big) \tag{16}$$

$$o_t = \sigma\Big(\left(\mathcal{W}_{xo} * \mathcal{X}'_t\right) \odot \mathcal{S}_x + \left(\mathcal{W}_{ho} * \mathcal{H}'_{t-1}\right) \odot \mathcal{S}_h + b_o \Big) \tag{17}$$

$$\mathcal{H}_t = o_t \odot tanh\left(\mathcal{C}_t\right) \tag{18}$$

$$\mathcal{M}_{h,t} = clip\left(\mathcal{M}'_{x,t} + \mathcal{M}'_{h,t-1}\right), \tag{19}$$

where ks_x and ks_h represent input-to-state kernel size and state-to-state kernel size, respectively. In the network implementation, we set ϵ to 10^{-8}. The *clip* is a function to adjust the scope of mask and is defined as follows.

$$clip\left(k\right) = \begin{cases} 1, & \text{if } k >= 1 \\ 0, & \text{otherwise} \end{cases} \tag{20}$$

In Equation (8) and (9), we multiply the input and hidden states by their corresponding masks so that the cell result depends only on valid input and valid hidden state. We update the mask by a convolution with the weight $\mathbf{1}$, as seen in Equation (10) and (11). Before the first input comes into the cell, we initialize both the hidden state $\mathcal{H}$ and the hidden state mask $\mathcal{M}_h$ to zero. Initializing to zero means that there is no meaningful state information in the hidden state until input data comes in. We also initialize the mask of the hidden state to 0 to signify that there is no information in the hidden state. The mask scaling factors $\mathcal{S}_x$ and $\mathcal{S}_h$ shown in Equation (12) and (13) are the ratio of the updated mask and kernel size in the convolution window. By multiplying $clip\left(\mathcal{M}'_{x,t}\right)$, the ratio at the masked location is zero. Similar to the partial convolution, we utilize the mask scaling factor to adjust scaling only with unmasked input and unmasked hidden state. Gates (i_t, f_t, o_t), cell state, and hidden state in Equation (14)-(18) are updated with the partial convolution input $\mathcal{X}'_t$ and hidden state $\mathcal{H}'_{t-1}$. Finally, we calculate the mask $\mathcal{M}_{h,t}$ of the hidden state $\mathcal{H}_t$ as presented in Equation (19). $\mathcal{H}_t$ is calculated with the input $\mathcal{X}'_t$ updated with the partial convolution and hidden state $\mathcal{H}'_{t-1}$. Therefore, we compute a valid position in $\mathcal{H}_t$ as the union of a valid position in the input $\mathcal{X}'_t$ and a valid position in the $\mathcal{H}'_{t-1}$.

We then add the updated input mask $\mathcal{M}'_{x,t}$ and the updated hidden state mask $\mathcal{M}'_{h,t-1}$ for the union operation and then apply the *clip* function. Note that we apply the *clip* function to the updated mask so that the mask element ranges from 0 to 1. Since the hidden state accumulates information from the input and previous state [29], the mask of the hidden state accumulates the valid positions of the hidden state over the sequence.

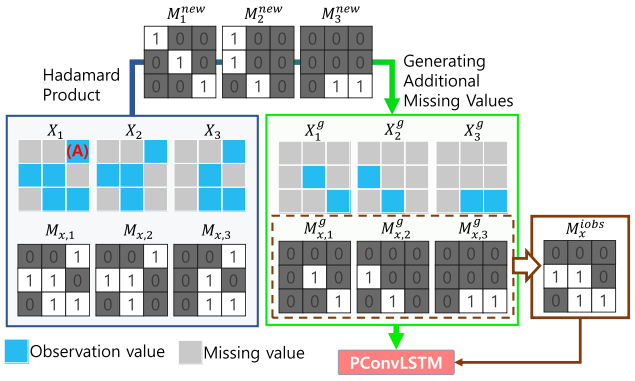


FIGURE 4. The process of the proposed generating missing technique. This shows an example with an input sequence of three time steps. We multiply a new mask sequence $\{\mathcal{M}_1^{new}, \mathcal{M}_2^{new}, \mathcal{M}_3^{new}\}$ by the original input sequence $\{\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3\}$ and the original input mask sequence $\{\mathcal{M}_{x,1}, \mathcal{M}_{x,2}, \mathcal{M}_{x,3}\}$. We obtain the input sequence with additional missing values $\{\mathcal{X}_1^g, \mathcal{X}_2^g, \mathcal{X}_3^g\}$ and the input mask sequence with additional zero values $\{\mathcal{M}_{x,1}^g, \mathcal{M}_{x,2}^g, \mathcal{M}_{x,3}^g\}$. We use these two sequences for the network training.

B. GENERATING MISSING TECHNIQUE

The proposed network predicts $\hat{\mathcal{X}_{t+1}}$ for a future frame $\mathcal{X}_{t+1}$ with the previous k frames $\mathcal{X}_{t-k+1}, \ldots, \mathcal{X}_t$. Our training goal is to optimize the weights of the network so that $Loss\left(\mathcal{X}_{t+1}, \hat{\mathcal{X}_{t+1}}\right)$ is minimized. However, since $\mathcal{X}_{t+1}$ is a frame that contains missing values, we need an algorithm that trains the network without ground truth. We propose a generating missing technique for the network to predict missing regions with missing data effectively. The proposed technique trains the network by generating additional missing values on the input sequences that already have missing values. Figure 4 presents our proposed generating missing technique as an example of the length-3 sequence data $\{\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3\}$. We create a new mask sequence $\{\mathcal{M}_1^{new}, \mathcal{M}_2^{new}, \mathcal{M}_3^{new}\}$ during the training process and multiply it by the original input sequence $\{\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3\}$ and the original input mask sequence $\{\mathcal{M}_{x,1}, \mathcal{M}_{x,2}, \mathcal{M}_{x,3}\}$ as shown in Figure 4. Note that we set all missing values to zeros. We obtain the input sequence with additional missing values $\{\mathcal{X}_1^g, \mathcal{X}_2^g, \mathcal{X}_3^g\}$ and the input mask sequence with additional zero values $\{\mathcal{M}_{x,1}^g, \mathcal{M}_{x,2}^g, \mathcal{M}_{x,3}^g\}$, and we use these two sequences for the PConvLSTM network training. We also use $\mathcal{M}_x^{iobs}$, which is the union of $\{\mathcal{M}_{x,1}^g, \mathcal{M}_{x,2}^g, \mathcal{M}_{x,3}^g\}$ for training and we describe the detail of $\mathcal{M}_x^{iobs}$ in the following Section III-C.

As explained earlier, we create a new mask sequence, $\mathcal{M}^{new} = \{\mathcal{M}_{t-k+1}^{new}, \ldots, \mathcal{M}_t^{new}\}$, to generate additional zero values randomly in the input mask sequence, $\mathcal{M}_x = \{\mathcal{M}_{x,t-k+1}, \ldots, \mathcal{M}_{x,t}\}$. We multiply the original input mask $\mathcal{M}_{x,i}$ by $\mathcal{M}_i^{new}$ to create a mask $\mathcal{M}_{x,i}^g$ with additional zero values. We multiply the input $\mathcal{X}_i$ by $\mathcal{M}_i^{new}$ to obtain the input $\mathcal{X}_i^g$ with additional missing values as follows.

$$\mathcal{M}_{x,i}^g = \mathcal{M}_{x,i} \odot \mathcal{M}_i^{new}, \quad i = t-k+1, \ldots, t \tag{21}$$

$$\mathcal{X}_i^g = \mathcal{X}_i \odot \mathcal{M}_i^{new}, \quad i = t-k+1, \ldots, t \tag{22}$$

From Equation (21) and (22), we use $\mathcal{X}^g = \{\mathcal{X}_{t-k+1}^g, \ldots, \mathcal{X}_t^g\}$ with additional missing values as the input sequence and $\mathcal{M}_x^g = \{\mathcal{M}_{x,t-k+1}^g, \ldots, \mathcal{M}_{x,t}^g\}$ as the input mask sequence to the network. We generate a new $\mathcal{M}^{new}$ for each epoch during the training. The network is trained with the new data that have additional missing values randomly per epoch. Note that we randomize $\mathcal{M}^{new}$ for each epoch to avoid losing data completely. In the generating missing technique, the network is trained to predict space-time information with fewer data by additionally generating missing data. Therefore, our network tends to predict more successfully future values of data with more missing values than training data. The proposed generating missing technique also trains the network to predict data at any location, although there is no data sample at the location. Besides, the incomplete data may always have samples at specific locations. In the sensor data, for example, there is always no observation value at a location where there is no sensor. In the original input sequence of Figure 4, (A) is the location where the observation value always exists. However, after the additional missings, all data are missing at (A). Therefore, we train the network to predict data that always miss at the location (A).

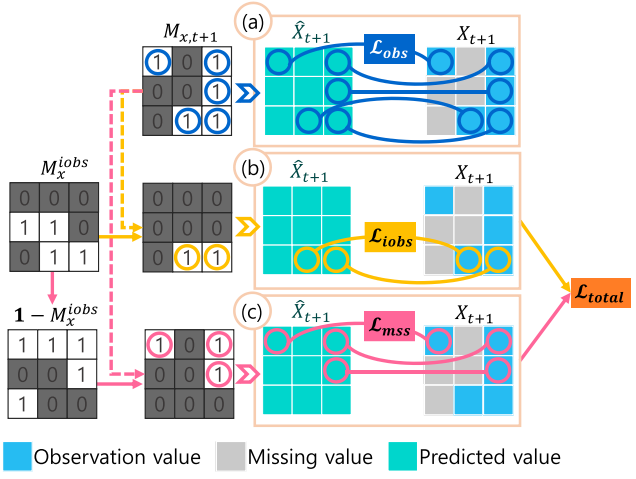


FIGURE 5. Loss functions. We define three different loss functions. $\mathcal{L}_{obs}$, $\mathcal{L}_{iobs}$ and $\mathcal{L}_{mss}$ according to the masks.

C. LOSS FUNCTION

In this work, we define three different loss functions, $\mathcal{L}_{obs}$, $\mathcal{L}_{iobs}$ and $\mathcal{L}_{mss}$. Besides, we employ a total variation (TV) loss function $\mathcal{L}_{tv}$ for the spatial smoothness [30]. Figure 5 illustrates how to compute the losses using $\mathcal{M}_{x,t+1}$

and $\mathcal{M}_x^{iobs}$. As presented in Figure 5 (a), we define $\mathcal{L}_{obs}$ as follows.

$$\mathcal{L}_{obs} = \frac{1}{N} \left\| \mathcal{M}_{x,t+1} \odot \left(\hat{\mathcal{X}_{t+1}} - \mathcal{X}_{t+1} \right) \right\|_1, \tag{23}$$

where $\mathcal{M}_{x,t+1}$ is the mask of $\mathcal{X}_{t+1}$ and N is the size of $\mathcal{X}_{t+1}$, which indicates N = height × width of $\mathcal{X}_{t+1}$. In Equation (23), we calculate the loss only for the observations except for the missing locations. The TV loss $\mathcal{L}_{tv}$ is defined as follows.

$$\mathcal{L}_{tv} = \sum_{(u,v)\in R,(u,v+1)\in R} \frac{\left\| \tilde{\mathcal{X}_{t+1}}(u, v+1) - \tilde{\mathcal{X}_{t+1}}(u, v) \right\|_1}{N} + \sum_{(u,v)\in R,(u+1,v)\in R} \frac{\left\| \tilde{\mathcal{X}_{t+1}}(u+1, v) - \tilde{\mathcal{X}_{t+1}}(u, v) \right\|_1}{N}, \tag{24}$$

where R is the region of 1-pixel dilation of the missing region and $\tilde{\mathcal{X}_{t+1}}$ is the output frame, but the areas where the element of $\mathcal{M}_{x,t+1}$ is 1 are set to valid pixels of $\mathcal{X}_{t+1}$. Without the proposed generating missing technique, the network is optimized with $\mathcal{L}_{obs}+\lambda_{tv}\mathcal{L}_{tv}$, a weighted combination of $\mathcal{L}_{obs}$ and $\mathcal{L}_{tv}$, where λ_{tv} is the weights of $\mathcal{L}_{tv}$.

In the training introduced in Section III-B, we create additional missing values during the training and then represent the locations of all missing values in the input sequence as the sum of $\mathcal{M}_x^g$. The union of masks with additional missing values is presented as follows.

$$\mathcal{M}_x^{iobs} = clip\left(\sum_i^k \mathcal{M}_{x,i}^g \right). \tag{25}$$

If the element of $\mathcal{M}_x^{iobs}$ is 1, it indicates that there is at least one observation value, whereas if the element is 0, this indicates that all of the elements are missing at the corresponding position in the input sequence. As seen in Figure 5 (b) and Figure 5 (c), we define $\mathcal{L}_{iobs}$ and $\mathcal{L}_{mss}$ using $\mathcal{M}_x^{iobs}$ as follows.

$$\mathcal{L}_{iobs} = \left\| \mathcal{M}_{x,t+1} \odot \mathcal{M}_x^{iobs} \odot \left(\hat{\mathcal{X}_{t+1}} - \mathcal{X}_{t+1} \right) \right\|_1 \tag{26}$$

$$\mathcal{L}_{mss} = \left\| \mathcal{M}_{x,t+1} \odot \left(\mathbf{1} - \mathcal{M}_x^{iobs} \right) \odot \left(\hat{\mathcal{X}_{t+1}} - \mathcal{X}_{t+1} \right) \right\|_1. \tag{27}$$

$\hat{\mathcal{X}_{t+1}}$ is the output of the network which utilizes $\mathcal{X}^g$ and $\mathcal{M}_x^g$ as inputs. In Equation (26), only if the element of $\mathcal{M}_{x,t+1}$ and the element of $\mathcal{M}_x^{iobs}$ are both 1, the element of $\mathcal{M}_{x,t+1} \odot \mathcal{M}_x^{iobs}$ is 1, otherwise it is 0. Therefore, $\mathcal{L}_{iobs}$ is the loss function for the positions where at least one observation exists in the input sequence among the positions where observation exist in $\mathcal{X}_{t+1}$. In Equation (27), $\left(\mathbf{1} - \mathcal{M}_x^{iobs}\right)$ is the inverted $\mathcal{M}_x^{iobs}$. Therefore, $\mathcal{L}_{mss}$ is the loss function for the positions where there is no observation in the input sequence among the positions where the observatiosn exist in $\mathcal{X}_{t+1}$. In the generating missing technique, the network is optimized with a weighted combination of $\mathcal{L}_{iobs}$, $\mathcal{L}_{mss}$, and $\mathcal{L}_{tv}$. We define the total loss function as follows.

$$\mathcal{L}_{total} = \mathcal{L}_{iobs} + \lambda_m \mathcal{L}_{mss} + \lambda_{tv} \mathcal{L}_{tv}, \tag{28}$$

where λ_m denotes the weight of the loss function $\mathcal{L}_{mss}$. The larger the value of λ_m, the more weight for prediction of the unsampled regions. We set λ_m to 6 and λ_{tv} to 0.001 for all experiments in this work.

IV. EXPERIMENTS

In this section, we analyze the prediction accuracy of the proposed networks with two datasets. We have designed the experiments to compare the incomplete spatiotemporal data predictions of deep learning networks. Figure 6 presents four architectures of the network models, including ConvLSTM, PConvLSTM-C, PConvLSTM-P, and PConvLSTM-P (G). ConvLSTM is configured with two ConvLSTM layers and one convolutional layer. PConvLSTM-C is the network in which ConvLSTM layer is replaced by PConvLSTM layer from ConvLSTM. PConvLSTM-P is the network we proposed in Section III. PConvLSTM-P (G) has the same structure as PConvLSTM-P and is trained with the proposed generating missing technique. PConvLSTM-P (G) is optimized with $\mathcal{L}_{total}$ defined in Equation (28), whereas the other three networks are trained using $\mathcal{L}_{obs}$ and $\mathcal{L}_{tv}$ defined in Equation (23) and (24) respectively.

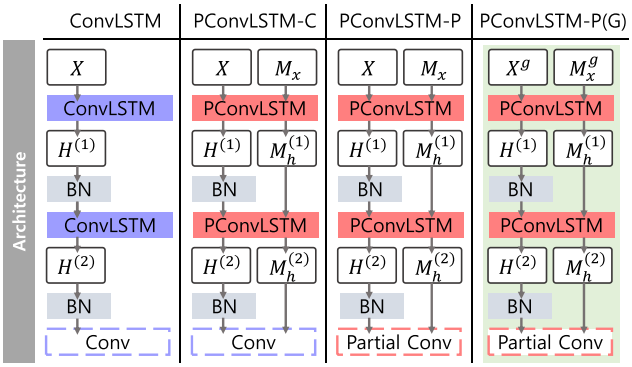


FIGURE 6. Comparison of four network architectures, ConvLSTM, PConvLSTM-C, PConvLSTM-P, and PConvLSTM-P (G). BN indicates the Batch Normalization layer.

We compare ConvLSTM and PConvLSTM-C to examine the performance of the PConvLSTM layer. We also compare the PConvLSTM-C and PConvLSTM-P to analyze the predictions after combining the partial convolutional layer and the PConvLSTM layer. We examine the predictions of PConvLSTM-P and PConvLSTM-P (G) to evaluate the proposed generating missing technique. We also compare these four networks with a 2-phase approach which consists of interpolation and prediction. Luo *et al.* [31] propose a generative model for multivariate time series data imputation. Without a complete dataset, it is impossible to calculate the imputation accuracy. Therefore, they first impute the dataset, and then they train the classifier with the imputed data to evaluate the imputation performance with the classification accuracy. Similarly, in the 2-phase approach, we fill the missing region with the radial basis function (RBF) interpolation and then train ConvLSTM with the interpolated data.

A. DATA DESCRIPTION

We utilize the Moving MNIST dataset and the Radar Echo dataset for the model evaluation. In this section, we present the preprocessing of each dataset.

1) MOVING MNIST DATASET

We use the Moving MNIST dataset [32] containing 10,000 sequences. Each sequence consists of 64 × 64 image frames and displays two moving numbers, which are randomly selected from the MNIST dataset. Each number is assigned a velocity whose direction is chosen uniformly randomly on a unit circle and whose magnitude is also chosen uniformly at random over a fixed range. We use 8,000 training sets, 1,000 validation sets, and 1,000 test sets. We apply nine frames as input data and the tenth frame as the output reference.

2) RADAR ECHO DATASET

We have obtained the radar echo dataset from the CIKM AnalytiCup 2017 challenge[1]. Shenzhen Meteorological Bureau collected the radar echo dataset for three years, which includes radar maps measured 15 times (1.5 hours total) at 6-minute intervals over an area of $101km \times 101km$. In the radar map, 1km is represented by one pixel, and each pixel contains a radar reflectivity value. Therefore, each map size is 101 × 101. We split the datasets for training, validation, and testing at an 8:1:1 ratio, respectively. The number of sequences in the training set is 9,600, and the number of sequences in the validation set and the test set is 1,200. We then divide each dataset with a sliding window of time length seven and make each sequence consists of 7 frames. We apply the first six frames as input data and the last frame as the output reference.

B. GENERATING MISSING VALUES

In this work, we intentionally generate missing values to see how the networks are trained with incomplete spatiotemporal data. To generating missing values, we create irregular masks by multiplying two sets of masks, which are a random sample mask and a set of random walk masks as presented in Figure 7, which shows how we create missing values with the Moving MNIST dataset. Figure 7 (a) presents a random sample mask created by generating random samples with the 30% missing rate. We randomly set 0 and 1 at a ratio of 3:7 to form the random sample mask whose size is 64 × 64. To make the missing regions more irregular, we create random walk masks, as shown in Figure 7 (b). We make the random walk masks in the same way as described in work by Yu *et al.* [33]. First, we create a 64 × 64 image frame where all elements are 1. The random walk is performed by a specific step starting from a randomly selected point in the frame. The element of the mask selected in each step changes from 1 to 0. The element can be selected more than once with the random walk procedure. We set the total number of steps to 64^2. We create different random walk masks for each frame of the Moving MNIST dataset. To create the irregular masks as shown in Figure 7 (c), we multiply one random sample mask by each random walk mask. In the irregular mask, the part where the random sample mask is 0 indicates a common missing region in the entire radar echo dataset, whereas, in the irregular mask, the part where the random walk mask is 0 means a different missing region for each data frame. Figure 7 (d) displays the original data without any missing value. Figure 7 (e) reveals the missing data created by multiplying the original data in Figure 7 (d) by the irregular masks in Figure 7 (c). Note that in this work, we do not use the original data for the network training; instead, we create missing values for both input and output data. We generate missing values with irregular masks in the training set and validation set, and the missing rate in the Moving MNIST dataset is 47%. We create three more random sample masks in addition to the irregular masks in the test set. In order to compare the prediction accuracies according to the missing rates, we generate 35%, 55%, 75%, and 95% missing values in each test set. For each missing rate, we create all different random sample masks for all frames in the test set. We do not combine the random walk mask with three random sample masks. The random sample mask in the training set is fixed as one, but in the test set, the random sample mask is different for each frame. We examine the predictions of the deep learning networks for three test set masks. In the same way, we also create masks for the radar echo dataset. For the irregular masks of the radar echo dataset, we set the missing rate of the random sample mask to 95% and the number of steps in the random walk to 101^2. Then, the missing rate of the radar echo dataset by the irregular masks is 96%. In the generating missing technique, we created the new mask $\mathcal{M}_i^{new}$ in the same way as the random sample mask, and the missing rate is set to 10%.

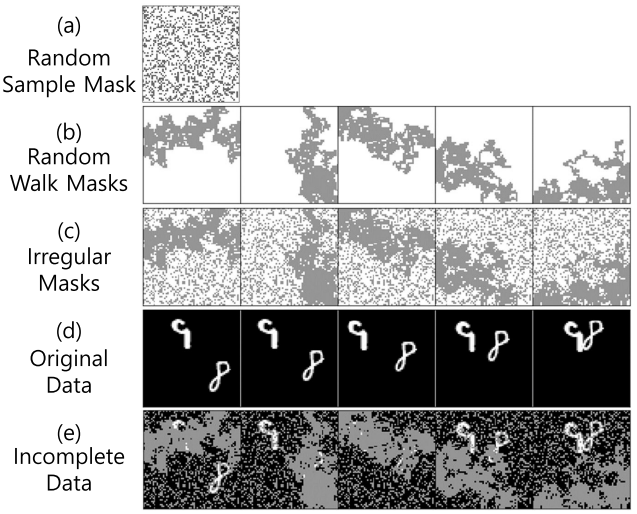


FIGURE 7. We create input mask sequence to generate missing values on the Moving MNIST dataset.

[1] https://tianchi.aliyun.com/competition/entrance/231596/information

TABLE 1. Performance comparison of ConvLSTM, PConvLSTM-C, PConvLSTM-P, and PConvLSTM-P (G) with the Moving MNIST test set. The highest value is highlighted in red, and the second-highest value is displayed in blue.

	Moving MNIST Dataset									
	35% random sample		55% random sample		75% random sample		95% random sample		(47%) Irregular	
method	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR
RBF+ConvLSTM	**0.8630**	**18.9437**	**0.8586**	**18.7769**	**0.8417**	**18.2048**	0.7043	14.5594	**0.8407**	**18.2113**
ConvLSTM	0.8489	17.6619	0.8364	17.3079	0.8076	16.4249	0.7368	14.4756	0.8207	16.7939
PCovLSTM-C	0.8505	17.7751	0.8429	17.4748	0.8207	16.7604	**0.7435**	14.7674	0.8236	16.9245
PConvLSTM-P	**0.8523**	**17.9255**	0.8449	17.6135	0.8238	16.9513	0.7417	**14.7880**	0.8252	17.0447
PConvLSTM-P (G)	0.8512	17.8995	**0.8451**	**17.6665**	**0.8255**	**17.0298**	**0.7444**	**14.8380**	**0.8259**	**17.0957**

C. RESULTS

We set the hidden state and kernel size of the four networks identically in each dataset. We create missing values in the training sets with the irregular masks described in Section IV-B. We scale the data range between 0 and 1 and train four networks using Adam optimizer [34] with a learning rate of 0.001. We set the batch size to 8 and the epoch size to 50. Then, we select the model with the lowest validation loss. We conduct experiments by TensorFlow [35] on a single NVIDIA TITAN Xp GPU. We generate missing values in the test sets with irregular masks and random sample masks as defined in Section IV-B. We evaluate the prediction performances of the networks for the five test sets with missing values. We compare the output frame and the target frame by calculating the Structural Similarity Index (SSIM) and the Peak signal-to-noise ratio (PSNR). SSIM is a perceptual quality metric that measures image similarity. PSNR is measured by calculating the mean squared error (MSE) between corresponding pixels. We use SSIM and PSNR, which are commonly applied criteria for the image comparison, to evaluate the predicted image quality. Higher SSIM and PSNR indicate better results.

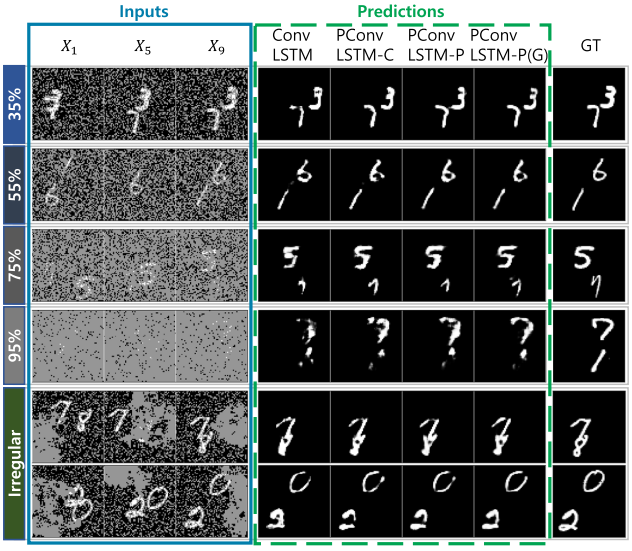


FIGURE 8. The prediction visualizations with the Moving MNIST dataset. The left pink box represents input sequences and the right green box presents the predictions with four networks and the ground truth(GT).

1) MOVING MNIST DATASET

We set the dimension of the hidden states to 32 and kernel size to 5 × 5. Table 1 presents the results for the Moving MNIST test set. RBF+ConvLSTM indicates the result of the 2-phase approach (RBF interpolation and ConvLSTM). RBF+ConvLSTM is the best when the missing rates are less than or equal to 75%. However, the SSIM of RBF+ConvLSTM is the lowest when the missing rate is 95%. Aside from these results, ConvLSTM without interpolation is always the worst. The results are generally improved in order of PConvLSTM-C, PConvLSTM-P, and PConvLSTM-P (G). Especially when the missing rate is 95%, the SSIM and PSNR of PConvLSTM-P (G) are the highest. However, when the missing rate is 35%, PConvLSTM-P performs better than PConvLSTM-P (G). With the irregular mask, the missing rate of the training set is about 47%, and we have trained PConvLSTM-P (G) with data having a missing rate higher than 47%. Consequently, the network trained by the generating missing technique performs well only on data with a missing rate higher than 47%.

We can also see the performance of four networks in Figure 8 that shows the prediction visualizations for the test sets. The left side of Figure 8 presents input sequences, and the right side visualizes the predictions of four networks and the ground truth (GT). In Figure 8, we can see that the predictions of three PConvLSTM networks, including (PConvLSTM-C, PConvLSTM-P, PConvLSTM-P (G)), are more apparent than one of the ConvLSTM network. In particular, for the test sets in which missing values are generated with the random sample masks, the output shapes of three PConvLSTM networks are more similar to the digits shown in the ground truth than one of the ConvLSTM network. The random sample mask is created in a different way compared to the irregular mask of the training set. Therefore, we can see that the PConvLSTM predicts more successfully than ConvLSTM for datasets with missing values, which are different from the training sets. As presented in Table 1 and Figure 8, the prediction performances of the network consisting of the PConvLSTM layer are always better than one of the ConvLSTM layer. Figure 9 shows the result of the 2-phase approach with the Moving MNIST dataset. The first row presents the

TABLE 2. Performance comparison of ConvLSTM, PConvLSTM-C, PConvLSTM-P, and PConvLSTM-P (G) with the radar echo test set. The highest value is highlighted in red, and the second-highest value is displayed in blue.

	Radar Echo Dataset									
	35% random sample		55% random sample		75% random sample		95% random sample		(96%) Irregular	
method	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR
RBF+ConvLSTM	0.4892	22.6642	0.5200	23.1262	0.5738	**23.7304**	**0.6422**	**23.0694**	**0.6181**	**22.1893**
ConvLSTM	0.4310	19.8774	0.4295	20.4559	0.3912	20.3205	0.2999	15.1848	0.2762	12.5033
PCovLSTM-C	0.6427	23.5287	0.6283	23.4299	0.5840	22.9498	0.3920	18.8317	0.3535	15.8513
PConvLSTM-P	**0.6499**	**23.6536**	**0.6419**	**23.5788**	**0.6211**	23.2588	0.4678	20.1493	0.4680	19.2032
PConvLSTM-P (G)	**0.6488**	**23.7088**	**0.6424**	**23.6272**	**0.6251**	**23.2866**	**0.5013**	**20.7076**	**0.5087**	**20.0096**

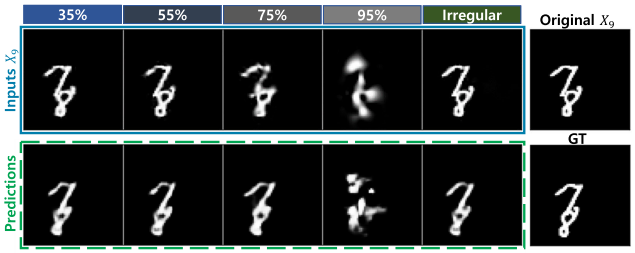


FIGURE 9. Results for the 2-phase approach on the Moving MNIST dataset. The first row shows the frames interpolated by RBF interpolation and the original frame. ConvLSTM is utilized for the predictions, and the second row represents the output frames and ground truth.

ninth frames of the interpolated input sequences by the RBF interpolation according to the missing rates. Moreover, the far right of the first row shows the original data without any missing value. The second row reveals the prediction results and the ground truth data of the ConvLSTM. We can see that the output prediction frame is significantly distorted when the missing rate is 95%.

2) RADAR ECHO DATASET

We set the dimension of the hidden states to 32 and kernel size to 3×3 in the four networks. Table 2 shows the results for the radar echo test set. In this experiment, ConvLSTM always produces the worst results. When the missing rate is more than 95%, the results with RBF+ConvLSTM are the best. However, when the missing rates are below 55%, the results with RBF+ConvLSTM are worse than ConvLSTM. The results for PConvLSTM-C, PConvLSTM-P, and PConvLSTM-P (G) are similar to those of the Moving MNIST dataset. The missing rate of the training set in which missing values were created with the irregular mask is 96%. Similar to the results of the Moving MNIST data, the PConvLSTM trained by the generating missing technique (PconvLSTM-P (G)) does not significantly improve the performance when the missing rate is 35%. However, if the missing rates are 55%, 75%, and 95%, PConvLSTM-P (G) outperforms PConvLSTM-P, even though the missing rate is lower than that of the training set. Figure 10 shows input sequences and predictions of the four networks, and the ground truth (GT). As the missing rate increases, the ConvLSTM produces artifacts such as holes in the middle of the space. We see that PConvLSTM-C and PConvLSTM-P could be better predictors than ConvLSTM but still produces hole marks. The PConvLSTM-P (G) produces the prediction most similar to the ground truth. We see that PConvLSTM-P (G) is the most successful in predicting continuous space when the missing rate is high or missing values are generated irregularly. Figure 11 presents the results of the 2-phase approach with the radar echo dataset. We can see artifacts in the output prediction frames when the missing rates are below 75%.

D. DISCUSSION

When the missing rate of the training set is higher than that of the test set, PConvLSTM-P (G) stably predicts better than PConvLSTM-P. This is because, as described in Section III-B, we have trained PConvLSTM-P (G) with data having additional missing values. Since we randomly generated missing values, we have trained PConvLSTM-P (G) to be more robust to irregularly distributed data. As a result, PConvLSTM-P (G) has higher performance for the irregular missing cases as well as test sets with high missing rates.

For 35% missing data, the generating missing technique does not seem to improve the network performance. In the experimental results for the Moving MNIST dataset, we have explained that this is because the missing rate (35%) of the missing data has lower than the missing rate (47%) of the training set. In the experiment results for the radar echo dataset, PConvLSTM-P (G) performs better than the PConvLSTM-P even when the missing rate (55%, 75%, and 95%) of the test set is lower than the missing rate (96%) of the training set. The cause of these results seems to be related to the characteristics of the radar echo dataset. In the radar echo dataset, there are many regions where pixel values are continuously changed. This feature makes the data robust against the occurrence of additional missing values. Therefore, data with a missing rate of 55% or higher is not significantly different from the training set. However, when the missing rate decreases to 35%, the difference between the training set and the test set increases, which implies that the generating missing technique does not help improve the performance. We further investigate this by producing missing values so that the missing rate of the training set and validation set of the radar echo dataset becomes 47%. To generate

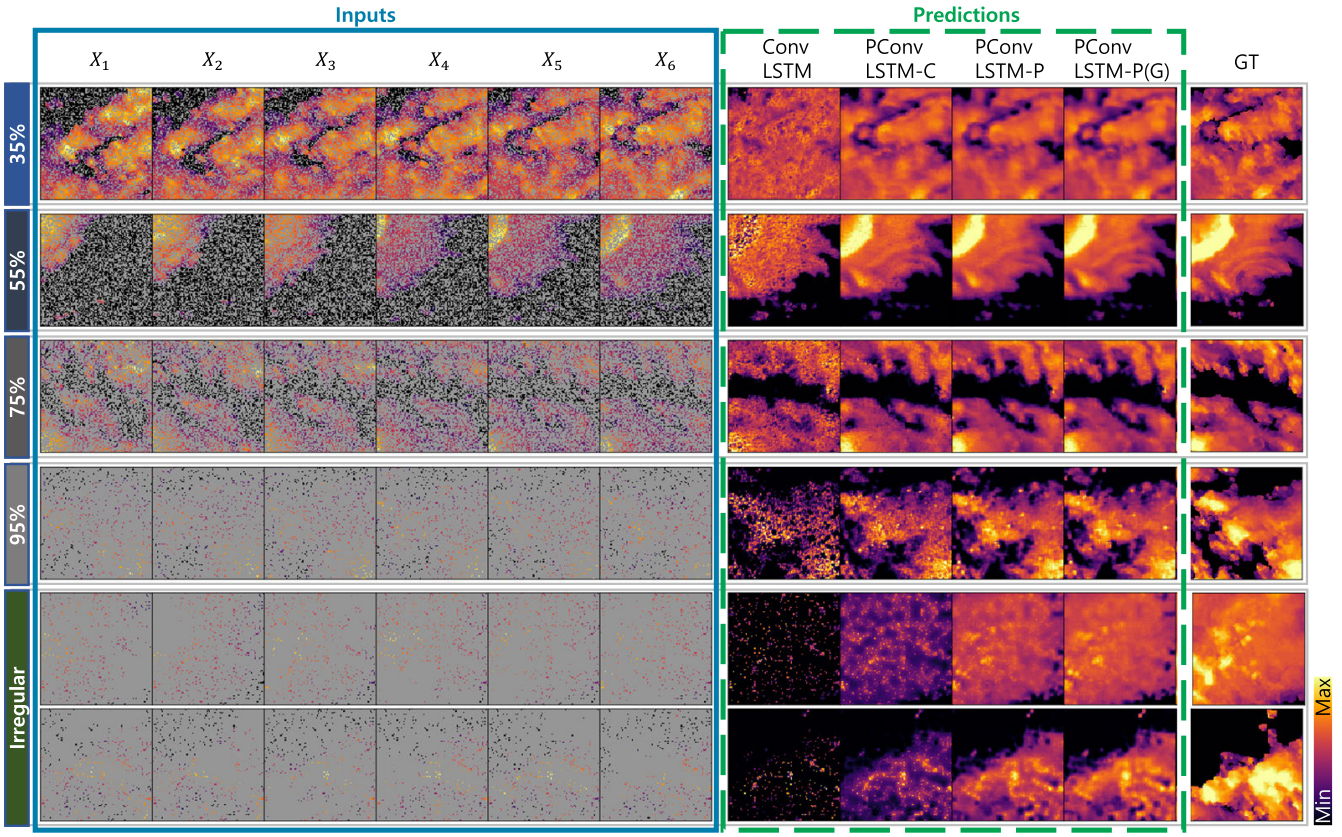


FIGURE 10. The prediction visualizations with the radar echo test set. The left side shows the input sequences and the right side presents the predictions with the four models and the ground truth.

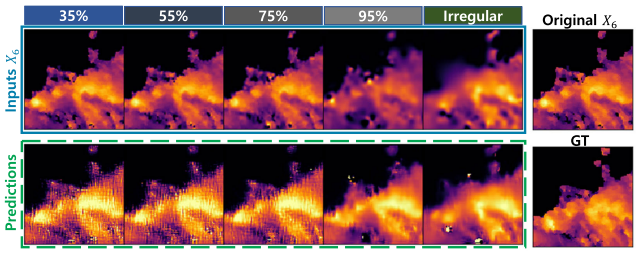


FIGURE 11. Results for the 2-phase approach on the radar echo dataset. The first row shows the frames interpolated by RBF interpolation and the original frame. ConvLSTM is utilized for the predictions, and the second row represents the output frames and ground truth.

47% missing data, we apply an irregular mask constructed by multiplying the random sample mask containing the missing rate of 30% with the random walk mask by a 101^2 step. Then, we train PConvLSTM-P and PConvLSTM-P (G) again. The two trained networks predict the 30% missing test set. As a result, the SSIM and PSNR of PConvLSTM-P are 0.6535 and 23.6931, respectively. The SSIM and PSNR of PConvLSTM-P (G) are 0.6686 and 24.0691, respectively, which are higher than ones of PConvLSTM-P. Therefore, the performance of PConvLSTM-P (G) trained with the 47% missing data for the 30% missing data is higher than that of PConvLSTM-P. Even if the missing rate in the radar echo dataset is as low as 30%, we observe that the generating missing technique improves the performance if the difference between the training set and the test set is not significant. From our experiments, it is seen that the generating missing technique may have different results depending on the data continuity characteristics. Therefore, we will enhance the algorithm to successfully predict data with fewer missing values than the training set regardless of the characteristics of data in the future.

We have also compared the results of the 2-phase approach for comprehensive experimental studies. The 2-phase approach becomes unstable when the missing rate of the test set is significantly different from the training set. This is because if the missing rate is different, the result estimated by the interpolation is different. If the difference between the training set and the test set is not significant enough, the 2-phase approach achieves excellent performance, but the larger the difference, the worse the results. In the proposed network, even if the missing rate difference increases, the artifacts occurring in the 2-phase approach are not produced. Also, the proposed network can be trained immediately without any pre-processing, as in the 2-phase approach.

V. CONCLUSION

In this paper, we proposed a new deep learning framework, Partial Convolutional Long-Short-Term-Memory (PConvLSTM). The proposed PConvLSTM extends ConvLSTM to handle incomplete spatiotemporal data with input frames and input mask by a partial convolution operation. To process the missing data, we designed the network so that the state is adjusted only by the observation values using a mask in the hidden state. We also introduced a training technique that does not use ground truth since many cases in the real-world data do not have ground truth. The proposed training technique is designed for the network to predict the data where the data is missing effectively. We have shown that it is possible to train incomplete spatiotemporal data by the proposed deep learning framework and training algorithm. As shown in the prediction results, PConvLSTM achieves higher accuracy than ConvLSTM. We confirmed that the PConvLSTM trained by the proposed generating missing technique could successfully predict data with large missing rate. In this paper, we focused on the missing problem with space rather than time. In the future, we will extend PConvLSTM to handle missing in time to enable missing frame estimation. Besides, the current one-step prediction will be extended to multi-step prediction using an encoder-decoder structure. Moreover, currently, PConvLSTM works only with grid-based data structures. We will develop a network supporting spatiotemporal graph-based data structure in which nodes and edges are missing irregularly.

REFERENCES

[1] M. F. Dixon, N. G. Polson, and V. O. Sokolov, "Deep learning for spatio-temporal modeling: Dynamic traffic flows and high frequency trading," *Appl. Stochastic Models Bus. Ind.*, vol. 35, no. 3, pp. 788–807, May 2019.

[2] S. Xingjian, 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.*, pp. 802–810, 2015.

[3] S. Liu, Y. Zhang, P. Ma, B. Lu, and H. Su, "A novel spatial interpolation method based on the integrated RBF neural network," *Procedia Environ. Sci.*, vol. 10, pp. 568–575, Jun. 2011.

[4] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 85–100.

[5] J. Lehtinen, J. Munkberg, J. Hasselgren, S. Laine, T. Karras, M. Aittala, and T. Aila, "Noise2noise: Learning image restoration without clean data," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 2965–2974.

[6] S. Soltanayev and S. Y. Chun, "Training deep learning based denoisers without ground truth data," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 3257–3267.

[7] H. Ji, C. Liu, Z. Shen, and Y. Xu, "Robust video denoising using low rank matrix completion," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 1791–1798.

[8] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Proc. 27th Int. Joint Conf. Artif. Intell.* Palo Alto, CA, USA: AAAI Press, Jul. 2018, pp. 3634–3640.

[9] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, pp. 1–21, 2020, doi: 10.1109/tnnls.2020.2978386.

[10] B. Graler, E. Pebesma, and G. Heuvelink, "Spatio-temporal interpolation using gstat," *RFID J.*, vol. 8, no. 1, pp. 204–218, 2016.

[11] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *Proc. Eur. Conf. Comput. Vis.* Zürich, Switzerland: Springer, 2014, pp. 184–199.

[12] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, Jul. 2017.

[13] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep CNN denoiser prior for image restoration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3929–3938.

[14] R. A. Yeh, C. Chen, T. Y. Lim, A. G. Schwing, M. Hasegawa-Johnson, and M. N. Do, "Semantic image inpainting with deep generative models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5485–5493.

[15] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Free-form image inpainting with gated convolution," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2019, pp. 4471–4480.

[16] C. Wang, H. Huang, X. Han, and J. Wang, "Video inpainting by jointly learning temporal structure and spatial details," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 5232–5239.

[17] D. Kim, S. Woo, J.-Y. Lee, and I. S. Kweon, "Deep video inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 5792–5801.

[18] H. Song, W. Wang, S. Zhao, J. Shen, and K.-M. Lam, "Pyramid dilated deeper ConvLSTM for video salient object detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 715–731.

[19] C. Zhao, P. Zhang, J. Zhu, C. Wu, H. Wang, and K. Xu, "Predicting tongue motion in unlabeled ultrasound videos using convolutional LSTM neural networks," in *Proc. ICASSP-IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, 2019, pp. 5926–5930.

[20] Z. Yuan, X. Zhou, and T. Yang, "Hetero-ConvLSTM: A deep learning approach to traffic accident prediction on heterogeneous spatio-temporal data," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, May 2019, pp. 984–992.

[21] G. Zhu, L. Zhang, P. Shen, and J. Song, "Multimodal gesture recognition using 3-D convolution and convolutional LSTM," *IEEE Access*, vol. 5, pp. 4517–4524, 2017.

[22] W. Luo, W. Liu, and S. Gao, "Remembering history with convolutional LSTM for anomaly detection," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2017, pp. 439–444.

[23] 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.

[24] S. Lee and J. Shin, "Hybrid model of convolutional LSTM and CNN to predict particulate matter," *Int. J. Inf. Electron. Eng.*, vol. 9, no. 1, pp. 1–5, 2019.

[25] D. W. Wong, L. Yuan, and S. A. Perlin, "Comparison of spatial interpolation methods for the estimation of air quality data," *J. Exposure Sci. Environ. Epidemiol.*, vol. 14, no. 5, pp. 404–415, 2004.

[26] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, "Structural-RNN: Deep learning on spatio-temporal graphs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 5308–5317.

[27] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. 6th Int. Conf. Learn. Represent. (ICLR)*, Vancouver, BC, Canada, Apr. 2018, pp. 1–33.

[28] Y. Lin, N. Mago, Y. Gao, Y. Li, Y.-Y. Chiang, C. Shahabi, and J. L. Ambite, "Exploiting spatiotemporal patterns for accurate air quality forecasting using deep learning," in *Proc. 26th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, 2018, pp. 359–368.

[29] T. H. Vu and J.-C. Wang, "Acoustic scene and event recognition using recurrent neural networks," *Detection Classification Acoustic Scenes Events*, vol. 2016, pp. 1–3, Sep. 2016.

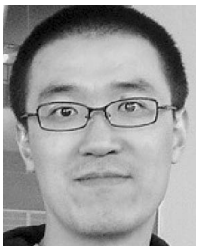
[30] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Proc. Eur. Conf. Comput. Vis.* Seattle, WA, USA: Springer, 2016, pp. 694–711.

[31] Y. Luo, Y. Zhang, X. Cai, and X. Yuan, "E2gan: End-to-end generative adversarial network for multivariate time series imputation," in *Proc. 28th Int. Joint Conf. Artif. Intell.* Palo Alto, CA, USA: AAAI Press, 2019, pp. 3094–3100.

[32] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using LSTMs," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 843–852.

[33] T. Yu, C. Lin, S. Zhang, S. You, X. Ding, J. Wu, and J. Zhang, "End-to-end partial convolutions neural networks for Dunhuang grottoes wall-painting restoration," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, Oct. 2019, pp. 1447–1455.
[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, May 2015, p. 13.
[35] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, and M. Kudlur, "Tensorflow: A system for large-scale machine learning," in *Proc. 12th USENIX Symp. Operating Syst. Design Implement. (OSDI)*, 2016, pp. 265–283.

HYESOOK SON (Student Member, IEEE) received the bachelor's degree in statistics from Sejong University, South Korea, in 2019. Her research interests include machine learning, predictive modeling, and data visualization.



YUN JANG (Member, IEEE) received the bachelor's degree in electrical engineering from Seoul National University, South Korea, in 2000, and the master's and Ph.D. degrees in electrical and computer engineering from Purdue University, in 2002 and 2007, respectively. From 2007 to 2011, he was a Postdoctoral Researcher with CSCS and ETH Zürich, Switzerland. He is currently an Associate Professor of computer engineering with Sejong University, Seoul, South Korea. His research interests include machine learning, interactive visualization, and visual analytics.