MotionRNN: A Flexible Model for Video Prediction with Spacetime-Varying Motions

Haixu Wu*, Zhiyu Yao*, Jianmin Wang, Mingsheng Long (✉)
School of Software, BNRist, Tsinghua University, China
{whx20,yaozy19}@mails.tsinghua.edu.cn, {jimwang,mingsheng}@tsinghua.edu.cn

Abstract

This paper tackles video prediction from a new dimension of predicting spacetime-varying motions that are incessantly changing across both space and time. Prior methods mainly capture the temporal state transitions but overlook the complex spatiotemporal variations of the motion itself, making them difficult to adapt to ever-changing motions. We observe that physical world motions can be decomposed into transient variation *and* motion trend*, while the latter can be regarded as the accumulation of previous motions. Thus, simultaneously capturing the transient variation and the motion trend is the key to make spacetime-varying motions more predictable. Based on these observations, we propose the* MotionRNN *framework, which can capture the complex variations within motions and adapt to spacetime-varying scenarios. MotionRNN has two main contributions. The first is that we design the* MotionGRU *unit, which can model the transient variation and motion trend in a unified way. The second is that we apply the MotionGRU to RNN-based predictive models and indicate a new flexible video prediction architecture with a* Motion Highway*, which can significantly improve the ability to predict changeable motions and avoid motion vanishing for stacked multiple-layer predictive models. With high flexibility, this framework can adapt to a series of models for deterministic spatiotemporal prediction. Our MotionRNN can yield significant improvements on three challenging benchmarks for video prediction with spacetime-varying motions.*

1. Introduction

Real-world motions are extraordinarily complicated and are always varying in both space and time. It is extremely challenging to accurately predict motions with space-time variations, such as the deformation, accumulation, or dissipation of radar echoes in precipitation forecasting. Recent advanced deterministic video prediction models, such as PredRNN [36], MIM [37] and Conv-TT-LSTM [26] mainly focus on capturing the simple state transitions across time. They overlook the complex variations within the motions so that they cannot predict accurately under the highly changing scenario. Besides, optical-flow based methods [22, 20] use local-invariant state transitions to capture the short-term temporal dependency but lack the characterization of long-term motion trends. These methods may degenerate significantly when modeling ever-changing motions.

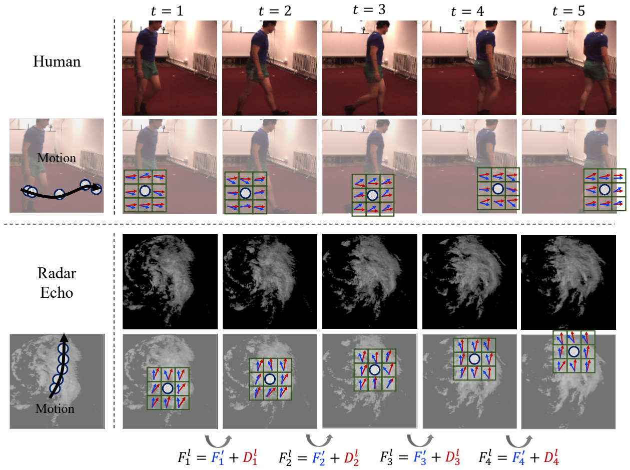

Figure 1. Two cases of real-world spacetime-varying motions. The movements $\mathcal{F}_t^l$ (shown in **black** arrows) of human legs or radar echoes can be decomposed into *transient variation* and *motion trend*. Our MotionRNN captures the transient variation $\mathcal{F}_t'$ (blue arrows) and the motion trend $\mathcal{D}_t^l$ (red arrows) simultaneously.

We observe that physical world motions can be naturally decomposed into the *transient variation* and *motion trend*. The transient variation can be seen as the deformation, dissipation, speed change or other variations of each local region instantly. As shown in Figure 1, when a person is running, different parts of the body will have various transient movement changes across time, *e.g.* the left and the right legs are taken forward alternately. Moreover, the natural spatiotemporal processes are following the rule of the trend, especially for physical motions. In the running scenario of Figure 1, the body sways up and down at each time step, but

[*] Equal contribution

the man keeps moving forward from left to right following the unchanging tendency. The motion follows the characteristics behind the physical world in a video sequence, such as inertia for objects, meteorology for radar echoes, or other physical laws, which can be seen as the *motion trend* of the video. Considering the decomposition of the motion, we should capture the transient variation and the motion trend for better space-time varying motion prediction.

We go beyond the previous state-of-the-art methods for deterministic spatiotemporal prediction [36, 37, 26] and propose a novel **MotionRNN** framework. To enable more expressive modeling of the spacetime-varying motions, MotionRNN adapts a **MotionGRU** unit for high-dimensional hidden-state transitions, which is specifically designed to capture the transient variation and the motion trend respectively. Inspired by the residual shortcuts in the ResNet [10], we improve the **Motion Highway** across layers within our framework to prevent the captured motions from vanishing and provide useful contextual spatiotemporal information for the MotionRNN. Our MotionRNN is flexible and can be easily adapted to the existing predictive models. Besides, MotionRNN achieves new state-of-the-art performance on three challenging benchmarks: a real-world human motion benchmark, a precipitation nowcasting benchmark, and a synthetic varied flying digits benchmark. The contributions of this paper are summarized as follows:

- Based on the key observation that the motion can be decomposed to transient variation and the motion trend, we design a new MotionGRU unit, which could capture the transient variation based on the spatiotemporal information and obtain the motion trend from the previous accumulation in a unified way.

- We propose the MotionRNN framework, which unifies the MotionGRU and a new Motion Highway structure to make spacetime-varying motions more predictable and to mitigate the problem of motion vanishing across layers in the existing predictive models.

- Our MotionRNN achieves the new state-of-the-art performance on three challenging benchmarks. And it is flexible to be applied together with a rich family of predictive models to yield consistent improvements.

2. Related Work

2.1. Deterministic Video Prediction

Recurrent neural networks (RNNs) have been wildly used in the field of video prediction to model the temporal dependencies in the video [19, 17, 24, 7, 6, 15, 22, 12, 32, 20, 9, 38, 26]. To learn spatial and temporal content and dynamics in a unified network structure, Shi *et al.* [21] proposed the convolutional LSTM (ConvLSTM), extending the LSTM with convolutions to maintain spatial information in the sequence model. The fusion of CNNs and LSTMs makes the predictive models capable to capture the spatiotemporal information. Finn *et al.* extended ConvLSTM for robotics to predict the transformation kernel weights between robot states. Wang *et al.* [36] introduced PredRNN, which makes the memory state update along a zigzag state transition path across stacked recurrent layers using the ST-LSTM cell. For capturing long-term dynamics, E3D-LSTM [35] incorporated 3D convolution and memory attention into the ST-LSTM, which can capture the long-term video dynamics. Su *et al.* [26] presented a high-order convolution LSTM (Conv-TT-LSTM) to learn the spatiotemporal correlations by combining the history convolutional features.

Still, previous spatiotemporal predictive models mainly focus on spatiotemporal state transitions but ignore internal motion variations. When it comes to instantly-changing motions, these predictive models may not behave well. To learn the coherence between frames, some video prediction methods are based on the optical flow [27, 23]. SDC-Net [20] learns the transformation kernel and kernel offsets between frames based on the optical flow. TrajGRU [22] also follows the idea of optical flow to learn the receptive area offsets for a special application of precipitation nowcasting. Villegas *et al.* [31] leveraged the optical flow for short-term dynamic modeling. These optical-flow based methods capture the short-term temporal dynamics effectively. However, they only treat the video as the instantaneous translation of pixels between adjacent frames and may ignore the motion trend of object variations.

Note that these methods are generally based on the RNN, such as LSTMs. In this paper, we propose a flexible external module for RNN-based predictive models without changing their original predictive framework. Unlike previous predictive learning methods, our approach focuses on modeling the within-motion variations, which could learn the explicit transient variation and remember the motion trend in a unified way. Our method naturally complements existing methods for learning spatiotemporal state transitions and can be applied with them for more powerful video prediction.

2.2. Stochastic Video Prediction

In addition to these deterministic video prediction models, some recent literature has explored the spatiotemporal prediction problem by modeling the future uncertainty. These models are based on adversarial training [16, 33, 28] or variational autoencoders (VAEs) [1, 28, 7, 14, 30, 4, 8]. These stochastic models could partially capture the spatiotemporal uncertainty by estimating the latent distribution for each time step. They did not attempt to explicitly model the motion variation, which is different from our MotionRNN. Again, MotionRNN can be readily applied with these stochastic models by replacing their underlying backbones.

3. Methods

Recall our observation as shown in Figure 1: real-world motions can be decomposed into the *transient variation* and *motion trend*. In the spirit of this observation, we propose the flexible MotionRNN framework with a motion highway, which could effectively enhance the ability to adapt to the spacetime-varying motions and avoid the motion vanishing. Further, we propose a specifically designed unit named MotionGRU, which can capture the transient variation and motion trend in a unified recurrent cell. This section will first describe the MotionRNN architecture and illustrate how to adapt MotionRNN to the existing RNN-based predictive models. Next, we will present the unified modeling of transient variation and motion trend in the MotionGRU unit.

3.1. MotionRNN

Typically, RNN-based spatiotemporal predictive models are in the forms of stacked blocks, as shown in Figure 2. Here we use each block to indicate the predictive RNN unit, such as ConvLSTM [21] or ST-LSTM [36]. In this framework, the hidden states transit between predictive blocks and are controlled by the inner recurrent gates. However, when it comes to spacetime-varying motions, the gate-controlled information flow would be overwhelmed by incessantly making quick responses to the transient variations of motions. Besides, it also lacks motion trend modeling.

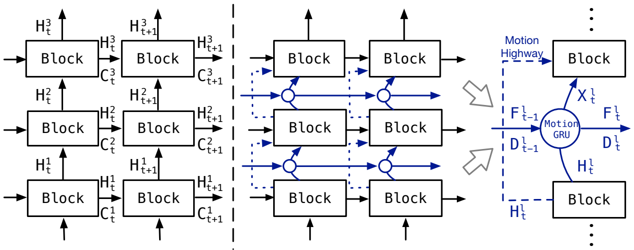

Figure 2. An overview of typical architecture of predictive frameworks: RNN-based spatiotemporal predictive networks (**left**), MotionRNN framework (**right**) which embeds the MotionGRU (blue circles) between layers of the original models. The blue dashed lines between stacked layers present the Motion Highway.

To tackle the challenge of spacetime-varying motions modeling, the MotionRNN framework incorporates the MotionGRU unit between the stacked layers as an operator without changing the original state transition flow (Figure 2). MotionGRU can capture the motion and conduct a state transition to the hidden states based on the learned motion. However, we find that motion will blur and even vanish when the transited features pass through multi-layers. Motivated by this observation, MotionRNN introduces the Motion Highway to provide an alternative quick route for the motion context information. We find that Motion Highway could effectively avoid motion blur and constrain the object in the right location from the visualization in Figure 6.

In detail, the MotionRNN framework inserts the MotionGRU between layers of the original RNN blocks. Take ConvLSTM [21] as an example. After the first layer, the overall equations for the l-th layer at time step t are as follows:

$$\begin{aligned}\mathcal{X}_t^l, \mathcal{F}_t^l, \mathcal{D}_t^l &= \mathrm{MotionGRU}(\mathcal{H}_t^l, \mathcal{F}_{t-1}^l, \mathcal{D}_{t-1}^l)\\ \mathcal{H}_t^{l+1}, \mathcal{C}_t^{l+1} &= \mathrm{Block}(\mathcal{X}_t^l, \mathcal{H}_{t-1}^{l+1}, \mathcal{C}_{t-1}^{l+1})\\ \mathcal{H}_t^{l+1} &= \mathcal{H}_t^{l+1} + (1 - o_t) \odot \mathcal{H}_t^l,\end{aligned} \tag{1}$$

where $l \in \{1, 2, \cdots, L\}$. Tensors $\mathcal{F}_t^l$ and $\mathcal{D}_t^l$ denote the transient filter and the trending momentum from MotionGRU respectively. We will give detailed descriptions to MotionGRU in the next section. The input $\mathcal{X}_t^l$ of the Block has been transited by MotionGRU. $\mathcal{H}_{t-1}^{l+1}$, $\mathcal{C}_{t-1}^{l+1}$ are the hidden state and memory state from the previous time step respectively, which are the same as original predictive blocks. o_t is the output gate of the RNN-based predictive block, which reveals the constantly updated memory in LSTMs.

The last equation presents the motion highway, which compensates the predictive block's output by the previous hidden state $\mathcal{H}_t^l$. We reuse the output gate to expose the desired unchanging content information. This highway connection provides extra details to the hidden states and balances the invariant part and the changeable motion part.

Note that MotionRNN does not change the state transition flows in the original predictive models. Thus, with this high flexibility, MotionRNN can adapt to a rich family of predictive frameworks, such as ConvLSTM [21], PredRNN [36], MIM [37], E3D-LSTM [35], and other RNN-based predictive models. It can significantly enhance spacetime-varying motion modeling of the existing predictive models.

3.2. MotionGRU

As mentioned above, towards modeling the spacetime-varying motions, our approach presents the MotionGRU unit to conduct motion-based state transitions by modeling the motion variation. In video prediction, the motion can be presented as pixels displacement corresponding to the hidden states transitions in RNNs. We use the MotionGRU to learn the *pixel offsets* between adjacent states. The learned pixel-wise offsets are denoted by *motion filter* $\mathcal{F}_t^l$. Considering that real-world motions are the composition of transient variations and motion trends, we specifically design two modules in the MotionGRU to model these two components respectively (Equation 4).

3.2.1 Transient Variation

In a video, the transient variation at each time step is not only based on the spatial context but also presents high temporal coherence. For example, the waving hands of a man follow a nearly continuous arm rotation angle between adjacent frames. Motivated by the spatiotemporal coherence of

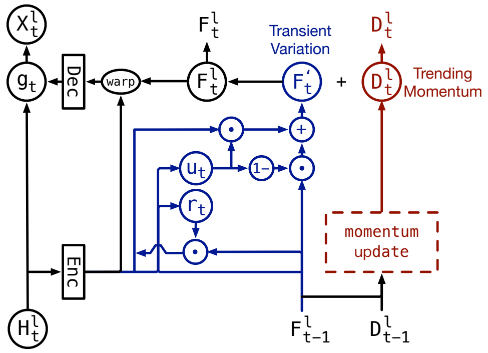

Figure 3. MotionGRU unit's architecture. The blue part is to capture the transient variation $\mathcal{F}'_t$. The trending momentum $\mathcal{D}^l_t$ accumulates the motion tendency in an accumulation way (red part).

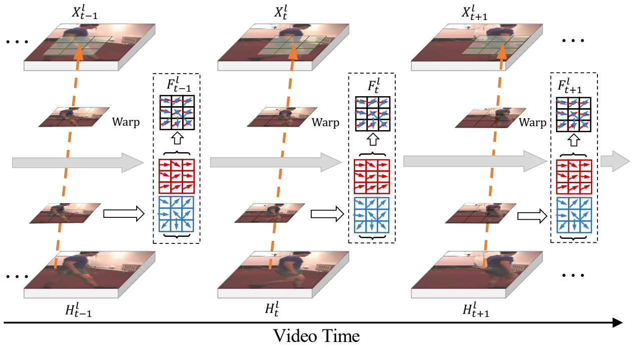

Figure 4. State transitions by MotionGRU. The motion filter $\mathcal{F}^l_t$ is combined by the transient variation (blue square) and trending momentum (red square). The new transited state is obtained by the Warp operation based on the learned motion filter.

transient variations, we adapt a ConvGRU [22] to learn the transient variation. With this recurrent convolutional network, the learned transient variation could consider the instant states and maintain the spatiotemporal coherence of variations. The equations of the transient-variation learner of the l-th MotionGRU at time step t are shown as follows:

$$\begin{aligned}
u_t &= \sigma\left(W_u * \text{Concat}([\text{Enc}(\mathcal{H}^l_t), \mathcal{F}^l_{t-1}])\right) \\
r_t &= \sigma\left(W_r * \text{Concat}([\text{Enc}(\mathcal{H}^l_t), \mathcal{F}^l_{t-1}])\right) \\
z_t &= \tanh\left(W_z * \text{Concat}([\text{Enc}(\mathcal{H}^l_t), r_t \odot \mathcal{F}^l_{t-1}])\right) \\
\mathcal{F}'_t &= u_t \odot z_t + (1 - u_t) \odot \mathcal{F}^l_{t-1}.
\end{aligned} \tag{2}$$

We use $\mathcal{F}'_t = \text{Transient}\left(\mathcal{F}^l_{t-1}, \text{Enc}(\mathcal{H}^l_t)\right)$ to summarize the above equations. σ is the sigmoid function, W_u, W_r and W_z denotes the 1×1 convolution kernel, $*$ and $\odot$ denote the convolution operator and the Hadamard product respectively. u_t and r_t are the update gate and reset gate in ConvGRU [22], and z_t is the reseted feature of current moment. $\text{Enc}(\mathcal{H}^l_t)$ encodes the input from the last predictive block. $\mathcal{F}^l_{t-1}$ presents motion filter from the previous time step for capturing the transient variations. Transient variation $\mathcal{F}'_t$ for current frame is calculated with the update gate u_t. Note that transient variation $\mathcal{F}^l_t$ presents the transition of each pixel's position between adjacent states. Thus, all the gates, z_t, and $\mathcal{F}^l_t$ are in the offset space, which are learned filters and different from spatiotemporal states $\mathcal{H}^l_t, \mathcal{C}^l_t$.

3.2.2 Trending Momentum

In the running scenario, the man's body sways up and down at each step while the man keeps moving forward. In this case, the motion is following a forward trend. In video prediction, we usually have to go through the whole frame sequence to get the motion trend. However, the future is unreachable. This dilemma is similar to reward prediction in reinforcement learning. Inspired by *Temporal Difference* learning [27], we use an accumulating way to capture the pattern of motion variation. We use the previous motion filter $\mathcal{F}^l_{t-1}$ as the estimation of the current motion trend and get the momentum update function as follows:

$$\mathcal{D}^l_t = \mathcal{D}^l_{t-1} + \alpha\left(\mathcal{F}^l_{t-1} - \mathcal{D}^l_{t-1}\right), \tag{3}$$

where α is the step size of momentum update and $\mathcal{D}^l_t$ is the learned trending momentum. We denote the above equation as $\mathcal{D}^l_t = \text{Trend}\left(\mathcal{F}^l_{t-1}, \mathcal{D}^l_{t-1}\right)$. With momentum update, $\mathcal{D}^l_t$ convergences to the weighted sum of motion filters $\mathcal{F}^l_t$, which can be viewed as the motion trend in the past period. In the running example (Figure 4), $\mathcal{F}^l_{t-1}$ presents the motion of the last moment and $\mathcal{D}^l_t$ denotes the forward trend learned from the past. By momentum updating, this tendency estimation is of larger coefficient over time. Note that the trending momentum $\mathcal{D}^l_t$ is the momentum update of motion filter $\mathcal{F}^l_t$ and is also in the offset space, which presents the learned motion trend of pixels in a video.

3.2.3 Overall Procedure for MotionGRU

By implementing the key observation of motion decomposition, we design MotionGRU as the following procedure:

$$\begin{aligned}
\mathcal{F}'_t &= \text{Transient}\left(\mathcal{F}^l_{t-1}, \text{Enc}(\mathcal{H}^l_t)\right) \\
\mathcal{D}^l_t &= \text{Trend}\left(\mathcal{F}^l_{t-1}, \mathcal{D}^l_{t-1}\right) \\
\mathcal{F}^l_t &= \mathcal{F}'_t + \mathcal{D}^l_t \\
m^l_t &= \text{Broadcast}\left(\sigma(W_{\text{hm}} * \text{Enc}(\mathcal{H}^l_t))\right) \\
\mathcal{H}'_t &= m^l_t \odot \text{Warp}\left(\text{Enc}(\mathcal{H}^l_t), \mathcal{F}^l_t\right) \\
g_t &= \sigma\left(W_{1\times 1} * \text{Concat}([\text{Dec}(\mathcal{H}'_t), \mathcal{H}^l_t])\right) \\
\mathcal{X}^l_t &= g_t \odot \mathcal{H}^l_{t-1} + (1 - g_t) \odot \text{Dec}(\mathcal{H}'_t),
\end{aligned} \tag{4}$$

where t denotes the time step and $l \in \{1, \cdots, L\}$ denotes the current layer, Transient$(\cdot)$ and Trend$(\cdot)$ present the transient-variation learner and trending-momentum updater respectively. $\mathcal{F}'_t$ and $\mathcal{D}^l_t$ denote the transient variation and trending momentum of the current frame. Based on the observation of motion decomposition, the motion filter $\mathcal{F}_t^l$ is the combination of transient variation and trending momentum. m_t^l is the mask for motion filter and Broadcast$(\cdot)$ means the broadcast operation with kernel W_{hm} to keep tensor dimension consistent to $\mathcal{H}_t'$.

For the state transition, we use the warp operation [2, 3] to map the pixels from the previous state to the position in the next state, which is widely used in different fields of video analysis, such as video style transfer [5] and video restoration [34]. Here Warp$(\cdot)$ denotes the warp operation with bilinear interpolation. As shown in Figure 4, warping the previous state by the learned motion $\mathcal{F}_t^l$, we can explicitly incorporate motion variation into the transition of hidden states. More details about the warp operation in MotionGRU can be found in the supplementary materials. As shown in Figure 3, the final output $\mathcal{X}_t^l$ of MotionGRU is a gate g_t controlled result from the input $\mathcal{H}_t^l$ and the decoder output, in which the decoder output has been explicitly transited by warp operation based on the motion filter $\mathcal{F}_t^l$.

Overall, by capturing the transient variation and motion trend separately and fusing them in a unified unit, MotionGRU can effectively model the spacetime-varying motions. With MotionGRU and Motion Highway, our MotionRNN framework can be applied to scenarios with ever-changing motions, which seamlessly compensates existing models.

4. Experiments

We extensively evaluate our proposed MotionRNN on the following three challenging benchmarks.

Human motions. This benchmark is built on the Human3.6M [11] dataset, which contains human actions from real world of 17 different scenarios with 3.6 million poses. We resize each RGB frame to the resolution of 128×128. Real-world human motion is much more complicated. For example, when a person is walking, different parts of the human body will have diverse transient variations, *e.g.* the arms and legs are bending, the body is swaying. The complex motion variations will make the prediction of real human motion a really challenging task.

Precipitation nowcasting. Precipitation nowcasting is a vital application of video prediction. It is challenging to predict the accumulation, deformation, dissipation, or diffusion of radar echos reflecting severe weather. This benchmark uses the Shanghai radar dataset, which contains evolving radar maps from Shanghai weather bureau. The Shanghai dataset has $40,000$ consecutive radar observations, collected every 12 minutes, with $36,000$ sequences for training and $4,000$ for testing. Each frame is resized to the resolution of 64×64.

Varied moving digits. We introduce the Varied Moving MNIST (V-MNIST) dataset consisting of sequences of frames with a resolution of 64×64. Previous Moving MNIST [25] or Moving MNIST++ [22] digits move with a lower velocity without digits variations. By contrast, our varied Moving MNIST forces all digits to move, rotate, and scale simultaneously. The V-MNIST are generated on the fly by sampling two different MNIST digits, with $100,000$ sequences for training and $10,000$ for testing.

Backbone models. To verify the universality of MotionRNN, we use the following predictive models as our backbone models including ConvLSTM [21], PredRNN [36], MIM [37] and E3D-LSTM [35]. On all benchmarks, our MotionRNN based on these models has four stacked blocks with 64-channel hidden states. For E3D-LSTM, we replace the encoder and decoder inside the MotionGRU with 3D convolutions to downsample the 3D feature map to 2D and keep the other operations unchanged.

Implementation details. Our method is trained with the $L1 + L2$ loss [35] to enhance the sharpness and smoothness of the generated frames simultaneously, using the ADAM [13] optimizer with an initial learning rate of 3×10^{-4}. The momentum factor α is set to 0.5. For memory efficiency, the learned filter size of MotionGRU is set to 3×3. The batch size is set to 8, and the training process is stopped after $100,000$ iterations. All experiments are implemented in PyTorch [18] and conducted on NVIDIA TITAN-V GPUs.

4.1. Human Motion

Setups. We follow the experimental setting in MIM [37], which uses the previous 4 frames to generate the future 4 frames. As for evaluation metrics, we use the frame-wise structural similarity index measure (SSIM), the mean square error (MSE), the mean absolute error (MAE) to evaluate our models. Besides these common metrics, we also use the Fréchet Video Distance (FVD) [29], which is a metric for qualitative human judgment of generated videos. The FVD could measure both the temporal coherence of the video content and the quality of each frame.

Results. As shown in Table 1, our proposed MotionRNN promotes diverse backbone predictive models with consistent improvement in quantitative results. Significantly, with MotionRNN the performance improves **29%** in MSE and **22%** in MAE using the PredRNN as the backbone. Our approach also promotes the FVD, which means the prediction performs better in motion consistency and frame quality. To our best knowledge, MotionRNN based on PredRNN has achieved the **state-of-the-art** performance on Human3.6M. As for qualitative results, we show a case of walking in Figure 5. In this case, the human has a left movement tendency with transient variations across different body parts. The frames generated by MotionRNN are richer in detail and less blurry than those of other models, especially for the

Table 1. Quantitative results of Human3.6M upon different network backbones ConvLSTM [21], MIM [37], PredRNN [36] and E3D-LSTM [35]. A lower MSE, MAE or FVD, or a higher SSIM indicates a better prediction.

Method	SSIM	MSE/10	MAE/100	FVD
TrajGRU [22]	0.801	42.2	18.6	26.9
Conv-TT-LSTM [26]	0.791	47.4	18.9	26.2
ConvLSTM [21]	0.776	50.4	18.9	28.4
+ MotionRNN	0.800	44.3	18.6	26.9
MIM [37]	0.790	42.9	17.8	21.8
+ MotionRNN	0.841	35.1	14.9	18.3
PredRNN [36]	0.781	48.4	18.9	24.7
+ MotionRNN	0.846	**34.2**	**14.8**	**17.6**
E3D-LSTM [35]	0.869	49.4	16.6	23.7
+ MotionRNN	**0.881**	44.5	15.8	21.7

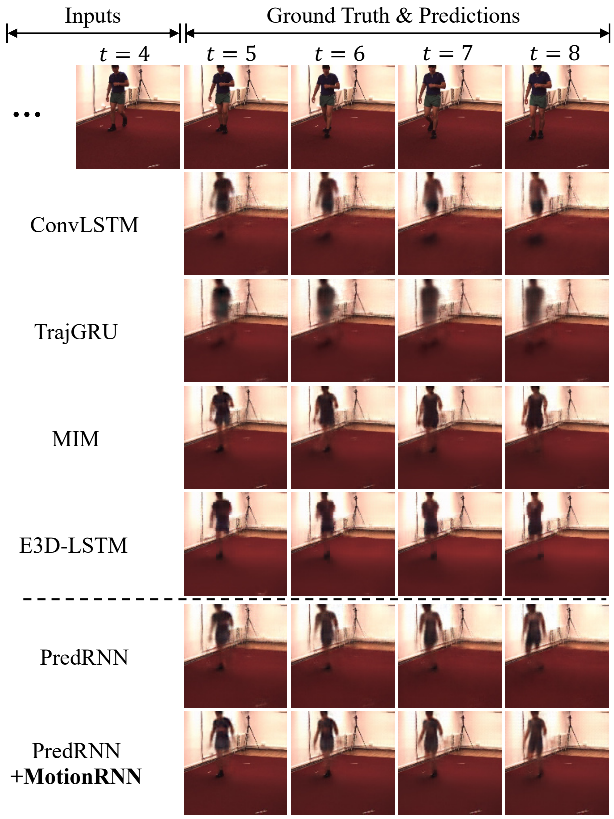

Figure 5. Prediction frames on the human motion benchmark.

arms and legs. PredRNN and MIM may present the prediction in good sharpness but fail to bend the left elbow, and the predicted legs are also blurry. By contrast, MotionRNN could predict the sharpest sequence compared with previous methods and largely enrich the detail for each part of the body, especially for the arms and legs. What's more, the pose prediction for the arms and the legs is also predicted more precisely, which means our approach could not only maintain the details but perform well in motion capturing.

Table 2. Parameters and computations comparison of MotionRNN using diverse backbone models. FLOPs denotes the number of multiplication operations for a human sequence prediction, which predicts the future 4 frames based on the previous 4 frames.

Method	Params(MB)	FLOPs(G)	MSEΔ
ConvLSTM	4.41	31.6	-
+ MotionRNN	5.21(↑ 18%)	36.6(↑ 16%)	12%
PredRNN	6.41	46.0	-
+ MotionRNN	7.01(↑ 9.3%)	49.5(↑ 7.6%)	29%
MIM	9.79	70.2	-
+ MotionRNN	10.4(↑ 6.2%)	73.7(↑ 5.0%)	18%
E3D-LSTM	20.4	292	-
+ MotionRNN	21.3(↑ 4.4%)	303(↑ 3.8%)	10%

Table 3. The ablation of MotionRNN with respect to Motion Highway (**MH**), Transient Variation (**TV**) and Trending Momentum (**TM**) on the Human3.6M dataset. Δ denotes the MSE improvements over PredRNN.

Method	MH	TV	TM	$\frac{\text{MSE}}{10}$	Δ
PredRNN				48.4	-
+ Motion Highway	√			42.5	12%
+ MotionGRU w/o Momentum		√		41.5	14%
+ MotionGRU w/o Transient			√	43.5	10%
+ MotionGRU		√	√	40.3	17%
+ MotionRNN w/o Momentum	√	√		38.9	20%
+ MotionRNN w/o Transient	√		√	40.6	16%
+ MotionRNN	√	√	√	**34.2**	**29%**

Parameters and computations analysis. We measure the complexity in terms of both model size and computations, as shown in Table 2. MotionRNN improves the performance of the PredRNN significantly (MSE: 48.4 → 34.2, SSIM: 0.781 → 0.846) with only 9.3% additional parameters and 7.6% increased computations. The increase of the model size is the same among different predictive frameworks because MotionRNN is only used as an external operator for hidden states across layers. The growth of the computations is also controllable. Based on these observations, we can see that our MotionRNN is a flexible model, which can improve the performance significantly on spatiotemporal variation modeling without significant sacrifice in model size or computation cost.

Ablation study. As shown in Table 3, we analyze the effectiveness of each part from our MotionRNN. Only by adopting the Motion Highway we could get a fairly good promotion (12%↑) indicating the Motion Highway can maintain the information of the motion context and compensate existing models for the additional useful information. Only adopting the MotionGRU without Motion Highway makes the MotionRNN achieve 17% improvement. From the quantitative results described in Table 3, we could easily find that the Motion Highway and MotionGRU can promote each other and achieve better improvement (**29%**↑). Furthermore, from the qualitative results shown in Figure 6, we can find without the Motion Highway, the predictions lose details of the arms and have the positional skewing. Thus we can verify the effect of our Motion Highway, which can compensate necessary content details to the MotionGRU and constrained the motion in the right area. More visualization can be found in supplementary materials. Besides, the learned trending momentum and transient variation give 9% and 13% extra promotions respectively, indicating that both parts of motion decomposition are effective for video prediction.

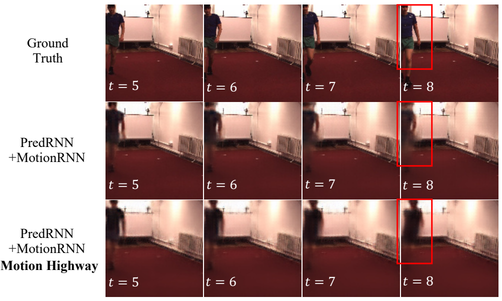

Figure 6. The qualitative case for the ablation study of the Motion Highway, using the red box to box out of the body.

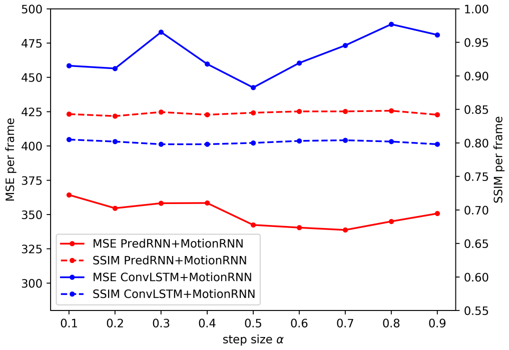

Figure 7. The sensitivity analysis of hyper-parameter α.

Hyper-parameters. We show the sensitivity analysis of the training hyper-parameter α for trending momentum in Figure 7. Our MotionRNN based on PredRNN and ConvLSTM achieves great performance when $\alpha = 0.5$ and is robust and easy to tune in the range of 0.5 to 0.7. We have similar results on the other two benchmarks and thus set α to 0.5 throughout the experiments.

4.2. Precipitation Nowcasting

Setups. We forecast the next 10 radar echo frames from the previous 5 observations, covering weather conditions in the next two hours. We use the gradient difference loss (GDL) [16] to measure the sharpness of the prediction frames. A lower GDL indicates a higher sharpness similarity of ground truth. Further, for radar echo intensities, we convert the pixel values in dBZ and compare the Critical Success Index (CSI) with 30 dBZ, 40 dBZ, 50 dBZ as thresholds, respectively. CSI is defined as $\mathrm{CSI} = \frac{\mathrm{Hits}}{\mathrm{Hits}+\mathrm{Misses}+\mathrm{FalseAlarms}}$, where hits correspond to the true positive, misses correspond to the false positive, and false alarms correspond to the false negative. A higher CSI indicates better forecasting performance. Compared with MSE, the CSI metric is particularly sensitive to the high-intensity echoes, always with high changeable motions.

Table 4. Quantitative results of the Shanghai dataset upon different network backbone. A lower GDL or a higher CSI means a better prediction performance.

Method	SSIM	GDL	CSI30	CSI40	CSI50
TrajGRU	0.815	13.9	0.576	0.545	0.484
Conv-TT-LSTM	0.820	13.6	0.571	0.530	0.469
ConvLSTM	0.837	12.3	0.624	0.605	0.560
+ MotionRNN	0.850	11.9	0.646	0.629	0.586
MIM	0.849	11.3	0.654	0.646	0.609
+ MotionRNN	0.863	11.1	0.668	0.654	0.614
PredRNN	0.841	11.9	0.633	0.622	0.581
+ MotionRNN	0.865	10.9	**0.678**	**0.664**	**0.623**
E3D-LSTM	0.842	12.7	0.615	0.615	0.590
+ MotionRNN	**0.880**	**9.67**	0.671	0.659	0.621

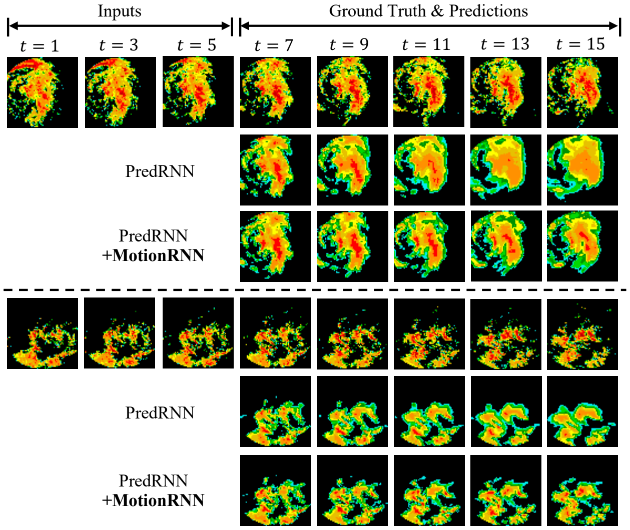

Figure 8. Prediction examples on the Shanghai radar echo dataset.

Results. We provide quantitative results in Table 4, our MotionRNN using the state-of-the-art model E3D-LSTM achieves **24**% improvement on the GDL metric, indicating our MotionRNN could produce the most sharpness predicted sequence. With our MotionRNN, the predictive frameworks could significantly improve various CSI metrics with different thresholds, which demonstrates that our approach can make predictions well on the changeable radar echos. As shown in Figure 8, MotionRNN predicts the motion more precisely in qualitative results. In the top case, there is a cyclone in which the motion contains moving up and anticlockwise rotation. PredRNN could roughly predict the cyclone positions but suffers from blurring. Focusing on the center part of prediction at $t = 15$, MotionRNN forecasts the exact rotation trend, but the echoes predicted from PredRNN is just a block without cyclone rotation details. In the bottom case, the echoes have an upward-diffusion movement and a slight anticlockwise rotation simultaneously. Our approach provides more details for the diffusion than PredRNN. We find many small cloud clusters in the right-bottom area and a more subtle outline of the center part. We believe such an accurate, fine-grained prediction will be very valuable for severe weather forecasting.

Motion trend visualization. To have a better view of the learned motion trend, we visualized the $\mathcal{D}_t^1$, the trending momentum of the first layer MotionGRU. We use the arrows to show the direction of the offsets, which represent the motion trend. The details about visualization operations are shown in the supplementary materials. In Figure 9, the center arrows show a moving up and anticlockwise rotation. The bottom arrows indicate the downward-motion of a cyclone's small tile. This visualization exactly shows that MotionRNN could capture the motion trend and have the ability to model the spacetime-varying motion in radar.

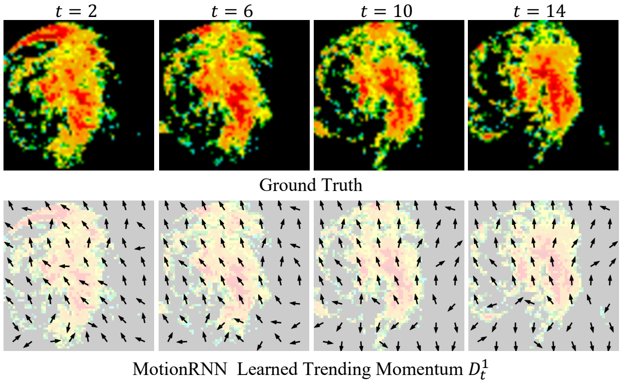

Figure 9. The visualization of learned motion tendency, which shows the motion direction. The arrows are calculated from $\mathcal{D}_t^1$.

4.3. Varied Moving Digits

Setups. We predict the future 10 frames based on the previous 10 frames. We use MSE, SSIM, GDL, and Peak Signal to Noise Ratio (PSNR) as evaluation metrics. Compared with the original Moving MNIST++ dataset, the sequences in our proposed varied Moving MNIST (V-MNIST) dataset are in lots of variations, such as faster moving speed, higher speed rotation, and scaling.

Results. With MotionRNN, the backbone models could get a consistent improvement in all metrics, as presented in Table 5. Especially in PredRNN, the predictions gain an excellent promotion in MSE and GDL. In the case of Figure 10, the digits move with rotation, which makes the prediction task harder. Previous models fail in giving the prediction with enough sharpness and clear strokes. As shown in the bottom line, with MotionRNN, PredRNN presents more satisfactory and sharper results.

Table 5. Quantitative results of V-MNIST. Higher PSNR means better prediction.

Method	MSE	SSIM	PSNR	GDL
TrajGRU	109	0.515	15.9	69.3
Conv-TT-LSTM	71.1	0.744	18.4	53.6
E3D-LSTM	57.6	0.852	19.7	44.6
+ MotionRNN	52.8	0.867	20.3	42.4
ConvLSTM	47.0	0.845	20.6	41.8
+ MotionRNN	44.4	0.861	20.9	40.3
MIM	34.6	0.888	22.3	34.6
+ MotionRNN	28.9	0.906	23.1	30.9
PredRNN	35.6	0.891	22.1	34.7
+ MotionRNN	**25.1**	**0.920**	**24.0**	**27.7**

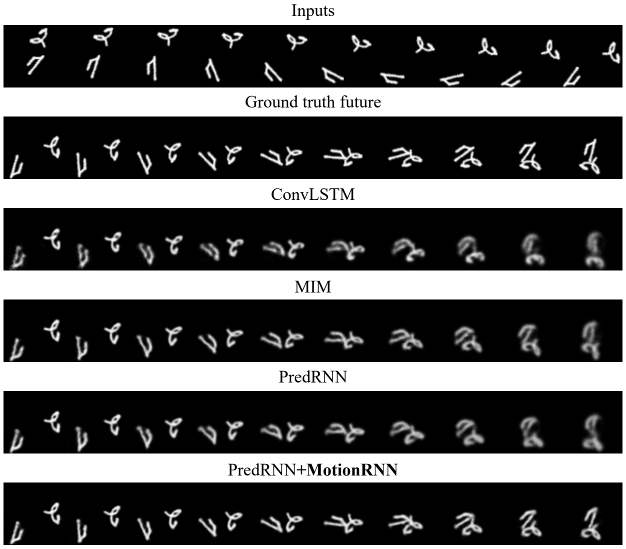

Figure 10. Prediction examples of V-MNIST.

5. Conclusion

In this paper, we have presented a flexible MotionRNN framework to predict spacetime-varying motions. Based on the observation that the motion can be decomposed to transient variation and the motion trend, we design the MotionGRU to capture the transient variation of the motion and the motion tendency respectively. By incorporating the MotionGRU to RNN-based predictive frameworks with the motion highway, MotionRNN can model the motion explicitly in state transitions and avoid the motion vanishing. With high flexibility, we apply MotionRNN with a series of predictive models to achieve significant promotions and state-of-the-art performance on three challenging prediction tasks.

Acknowledgments

This work was supported by National Key R&D Program of China (2017YFC1502003), NSFC grants (62022050, 62021002, 61772299, 71690231), Beijing Nova Program (Z201100006820041), and MOE Innovation Plan of China.

References

[1] Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H Campbell, and Sergey Levine. Stochastic variational video prediction. In *ICLR*, 2018.

[2] Michael J Black and Paul Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer vision and image understanding*, 63(1):75–104, 1996.

[3] Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert. High accuracy optical flow estimation based on a theory for warping. In *ECCV*, pages 25–36, 2004.

[4] Lluis Castrejon, Nicolas Ballas, and Aaron Courville. Improved conditional vrnns for video prediction. In *ICCV*, pages 7608–7617, 2019.

[5] Dongdong Chen, Jing Liao, Lu Yuan, Nenghai Yu, and Gang Hua. Coherent online video style transfer. In *ICCV*, pages 1105–1114, 2017.

[6] Bert De Brabandere, Xu Jia, Tinne Tuytelaars, and Luc Van Gool. Dynamic filter networks. In *NeurIPS*, pages 667–675, 2016.

[7] Emily Denton and Rob Fergus. Stochastic video generation with a learned prior. In *ICML*, pages 1182–1191, 2018.

[8] Jean-Yves Franceschi, Edouard Delasalles, Mickaël Chen, Sylvain Lamprier, and Patrick Gallinari. Stochastic latent residual video prediction. In *ICML*, pages 3233–3246, 2020.

[9] Vincent Le Guen and Nicolas Thome. Disentangling physical dynamics from unknown factors for unsupervised video prediction. In *CVPR*, pages 11474–11484, 2020.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[11] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, pages 1325–1339, 2013.

[12] Nal Kalchbrenner, Aaron van den Oord, Karen Simonyan, Ivo Danihelka, Oriol Vinyals, Alex Graves, and Koray Kavukcuoglu. Video pixel networks. In *ICML*, pages 1771–1779, 2017.

[13] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[14] Alex X Lee, Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine. Stochastic adversarial video prediction. In *ICLR*, 2019.

[15] Ziwei Liu, Raymond Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video frame synthesis using deep voxel flow. In *ICCV*, pages 4473–4481, 2017.

[16] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. In *ICLR*, 2016.

[17] Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L Lewis, and Satinder Singh. Action-conditional video prediction using deep networks in atari games. In *NeurIPS*, pages 2863–2871, 2015.

[18] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035, 2019.

[19] MarcAurelio Ranzato, Arthur Szlam, Joan Bruna, Michael Mathieu, Ronan Collobert, and Sumit Chopra. Video (language) modeling: a baseline for generative models of natural videos. *arXiv preprint arXiv:1412.6604*, 2014.

[20] Fitsum A Reda, Guilin Liu, Kevin J Shih, Robert Kirby, Jon Barker, David Tarjan, Andrew Tao, and Bryan Catanzaro. Sdc-net: Video prediction using spatially-displaced convolution. In *ECCV*, pages 718–733, 2018.

[21] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *NeurIPS*, pages 802–810, 2015.

[22] Xingjian Shi, Zhihan Gao, Leonard Lausen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. Deep learning for precipitation nowcasting: A benchmark and a new model. In *NeurIPS*, pages 5617–5627, 2017.

[23] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NeurIPS*, pages 568–576, 2014.

[24] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhutdinov. Unsupervised learning of video representations using LSTMs. In *ICML*, pages 843–852, 2015.

[25] Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. Training very deep networks. In *NeurIPS*, pages 2377–2385, 2015.

[26] Jiahao Su, Wonmin Byeon, Furong Huang, Jan Kautz, and Animashree Anandkumar. Convolutional tensor-train lstm for spatio-temporal learning. 2020.

[27] Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, pages 9–44, 1988.

[28] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. MoCoGAN: Decomposing motion and content for video generation. In *CVPR*, pages 1526–1535, 2018.

[29] Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018.

[30] Ruben Villegas, Arkanath Pathak, Harini Kannan, Dumitru Erhan, Quoc V Le, and Honglak Lee. High fidelity video prediction with large stochastic recurrent neural networks. In *NeurIPS*, pages 81–91, 2019.

[31] Ruben Villegas, Jimei Yang, Seunghoon Hong, Xunyu Lin, and Honglak Lee. Decomposing motion and content for natural video sequence prediction. In *ICLR*, 2017.

[32] Ruben Villegas, Jimei Yang, Yuliang Zou, Sungryull Sohn, Xunyu Lin, and Honglak Lee. Learning to generate long-term future via hierarchical prediction. In *ICML*, pages 3560–3569, 2018.

[33] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *NeurIPS*, pages 613–621, 2016.

[34] Xintao Wang, Kelvin CK Chan, Ke Yu, Chao Dong, and Chen Change Loy. Edvr: Video restoration with enhanced deformable convolutional networks. In *CVPRW*, pages 0–0, 2019.

[35] Yunbo Wang, Lu Jiang, Ming-Hsuan Yang, Li-Jia Li, Mingsheng Long, and Li. Fei-Fei. Eidetic 3D LSTM: A model for video prediction and beyond. In *ICLR*, 2019.

[36] Yunbo Wang, Mingsheng Long, Jianmin Wang, Zhifeng Gao, and S Yu Philip. PredRNN: Recurrent neural networks for predictive learning using spatiotemporal lstms. In *NeurIPS*, pages 879–888, 2017.

[37] Yunbo Wang, Jianjin Zhang, Hongyu Zhu, Mingsheng Long, Jianmin Wang, and Philip S. Yu. Memory in memory: A predictive neural network for learning higher-order non-stationarity from spatiotemporal dynamics. In *CVPR*, pages 9154–9162, 2019.

[38] Zhiyu Yao, Yunbo Wang, Mingsheng Long, and Jianmin Wang. Unsupervised transfer learning for spatiotemporal predictive networks. In *ICML*, 2020.