Supplementary Materials: *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

1. MotionGRU: Implementation Details

In this section, we will give a detailed description of the implementation details for MotionGRU from the **tensor view**. We first recall the equations of MotionGRU:

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

where subscript t denotes the time step, the superscript $l \in \{1, \cdots, L\}$ denotes the current layer, σ donates the sigmoid function, $*$ and $\odot$ denote the convolution and the Hadamard product respectively.

Learned Motion. $\mathcal{H}_t^l \in \mathbb{R}^{C\times H\times W}$ donates the hidden state of original predictive models. For memory efficiency, we employ the Encoder-Decoder structure, the encoder use the stride 2 convolution and $\text{Enc}(\mathcal{H}_t^l) \in \mathbb{R}^{\frac{C}{4}\times\frac{H}{2}\frac{W}{2}}$. $\mathcal{F}_t'$ and $\mathcal{D}_t^l$ present the learn transient variation and trending momentum respectively, which present the pixel-wise offsets of the state. Here, $\mathcal{F}_t', \mathcal{D}_t^l \in \mathbb{R}^{2k^2\times\frac{H}{2}\times\frac{W}{2}}$, where k is the filter size. $(\mathcal{F}_t')_{:,m,n}$ is a 1-dim tensor with size $2k^2$ and represent the vertical and horizontal offsets of the pixel at (m,n) position and the adjacent area with size k^2. Thus, the motion filter $\mathcal{F}_t^l \in \mathbb{R}^{2k^2\times\frac{H}{2}\times\frac{W}{2}}$ donates the learned motion-based state transition. With kernel W_{hm}, the channel number of the tensor is change to k^2. The Broadcast means the operation to broadcast and transpose the $k^2\times\frac{H}{2}\times\frac{W}{2}$ tensor to $\frac{C}{4}\times\frac{H}{2}\times\frac{W}{2}\times k^2$. $m_t^l \in \mathbb{R}^{\frac{C}{4}\times\frac{H}{2}\times\frac{W}{2}\times k^2}$ is the mask for the motion of the $k\times k$ filter area.

[*]Equal contribution

Warp. The Warp donates the warp operation with bilinear interpolation in a unit square. It can select the pixels in $\text{Enc}(\mathcal{H}_t^l)$, which point out to the $k\times k$ filter area by offset $\mathcal{F}_t^l$. It can be formulated as follows:

$$\begin{aligned}
\text{Warp}(\mathcal{H},\mathcal{F}) &= \Big(\mathcal{H}'_{c,m,n,i}\Big)_{\frac{C}{4}\times\frac{H}{2}\times\frac{W}{2}\times k^2} \\
&= \Big(\mathcal{H}_{c,(m+p_{ix}-\mathcal{F}_{i,m,n}),(n+p_{iy}-\mathcal{F}_{i+k^2,m,n})}\Big)_{\frac{C}{4}\times\frac{H}{2}\times\frac{W}{2}\times k^2},
\end{aligned} \tag{2}$$

where $i \in \{1,2,\cdots,k^2\}$, $p_{ix} = [\frac{i}{k}] - [\frac{k}{2}]$, $p_{iy} = (i \bmod k) - [\frac{k}{2}]$. The value of select pixel is calculated by the bilinear interpolation in the unit square as follows:

$$\begin{aligned}
&\mathcal{H}_{c,m',n'} \\
&= \begin{bmatrix}1-m'' & m''\end{bmatrix} \begin{bmatrix}\mathcal{H}_{c,\lfloor m'\rfloor,\lfloor n'\rfloor} & \mathcal{H}_{c,\lfloor m'\rfloor,\lceil n'\rceil} \\ \mathcal{H}_{c,\lceil m'\rceil,\lfloor n'\rfloor} & \mathcal{H}_{c,\lceil m'\rceil,\lceil n'\rceil}\end{bmatrix} \begin{bmatrix}1-n'' \\ n''\end{bmatrix}
\end{aligned} \tag{3}$$

where $\lfloor\cdot\rfloor$ and $\lceil\cdot\rceil$ donate the floor function and ceiling function respectively, $m'' = m' - \lfloor m'\rfloor$, $n'' = n' - \lfloor n'\rfloor$.

Gated Output. $\mathcal{H}_t' \in \mathbb{R}^{\frac{C}{4}\times\frac{H}{2}\times\frac{W}{2}\times k^2}$ donates the wrapped feature map. The decoder squeezes $\mathcal{H}_t'$ to $\mathbb{R}^{\frac{C}{4}\times\frac{H}{2}\times\frac{W}{2}}$ by 1×1 convolution and deconvolution to $\mathbb{R}^{C\times H\times W}$. The output gate $g_t \in \mathbb{R}^{C\times H\times W}$ is calculated from the concatenation of the input $\mathcal{H}_t^l$ and the decoded feature $\text{Dec}(\mathcal{H}_t')$ by 1×1 convolution $W_{1\times 1}$. The output $\mathcal{X}_t^l \in \mathbb{R}^{C\times H\times W}$ has the same shape as $\mathcal{H}_t^l$ and presents the motion-based transited state.

2. Visualization of Ablation Study

In addition to the intutive showcase in Figure 6 of the main text, we will provide more visualization for the ablation study to make each part's effect of MotionRNN more comprehensible.

Figure 1 visualizes more ablation cases. Without transient variation, the predictions lose pivotal details of hand and leg movement. Without trending momentum, the model fails to predict the human's back position correctly.

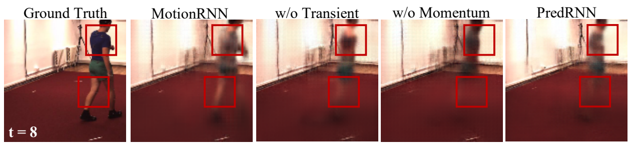

Figure 1. Visualization of ablation study. **Click to see the video**. Adobe reader is required to view the animation.

3. The Detail of Motion Trend Visualization

In this section, we will detail the visualization of learned trending momentum $\mathcal{D}_t^l$ corresponding to Figure 9 in the main text, which indicates the learned motion trend.

Take the Radar Shanghai dataset as an example. The input frame is with the resolution of $64 \times 64 \times 1$. Hidden states have 64 channels. After the patch operation with 4 patch number and the encoder with stride 2 inside the MotionGRU, the hidden state is converted to $\mathrm{Enc}(H)_t^l \in \mathbb{R}^{16\times 8\times 8}$. $\mathcal{D}_t^l \in \mathbb{R}^{18\times 8\times 8}$ can be split into two tensors. Each of them is in $\mathbb{R}^{9\times 8\times 8}$ and presents the learned offsets of 3×3 filter area in vertical and horizontal respectively. The learned offsets just present the motion. For visualization, we first calculate the mean along the channel dimension and get the x-offset and y-offset of every pixel of the $\mathrm{Enc}(\mathcal{H}_t^l)$. After the length regularization, we use the arrows to show the direction of the offsets, which represents the motion tendency.

Following the above method, we can also visualize the learned motion trend of the human case in Figure 1 of the main text. As shown in Figure 2, the learned motion trend indicates a left-to-right tendency, which is just the global movement and matches our expectation.

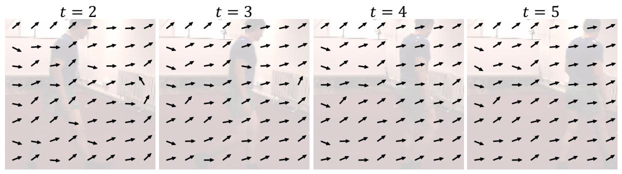

Figure 2. Visualization of learned motion trend in human case.

4. Guangzhou Benchmark

Besides the radar data from Shanghai, we also explore another challenging radar dataset [5], which its radar echos are collected every 6 minutes, from May 1st, 2014 to June 30th, 2014 in Guangzhou. The Guangzhou dataset is full of rain, and it can be used as an example of weather forecasting in rainy areas.

Setups. We also follow the experimental setting MIM [5], which has achieved the **state-of-the-art** performance in the Guangzhou dataset. We use the previous 10 frames to generate the future 10 frames. As for evaluation metrics, we use the CSI with 30 dBZ, 40 dBZ, 50 dBZ as thresholds.

Results. As shown in Table 1, the predictive models can get a consistent improvement in CSI with different thresholds, even in the **state-of-the-art** method MIM. Especially, MotionRNN can significantly promote CSI50. It means we can enhance the forecasts of severe weather through MotionRNN. Furthermore, as shown in Figure 3, the prediction results are more precise, indicating that the MotionRNN are better in detail prediction and more accurate on thick clouds prediction.

Table 1. Quantitative results of the Radar Echo dataset upon different network backbones. A higher CSI means a better performance.

Method	CSI30	CSI40	CSI50
TrajGRU[2]	0.251	0.201	0.150
E3D-LSTM[3]	0.350	0.309	0.254
+ MotionRNN	0.379	0.366	0.333
ConvLSTM[1]	0.385	0.360	0.315
+ MotionRNN	0.411	0.387	**0.350**
PredRNN[4]	0.401	0.378	0.306
+ MotionRNN	0.424	0.391	0.345
MIM[5]	0.429	0.399	0.317
+ MotionRNN	**0.431**	**0.403**	0.343

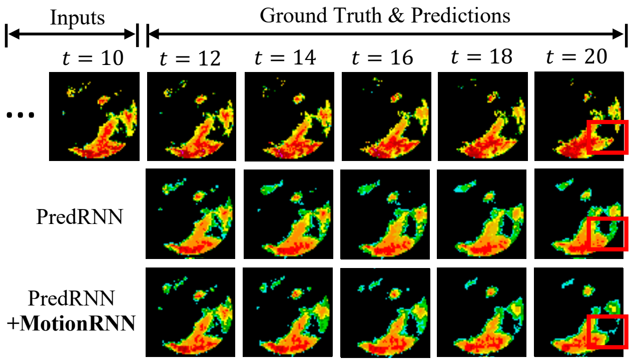

Figure 3. Prediction examples on the Radar Echo.

5. More Qualitative Results

We will show more qualitative cases in this section. Areas to focus on are highlighted in red.

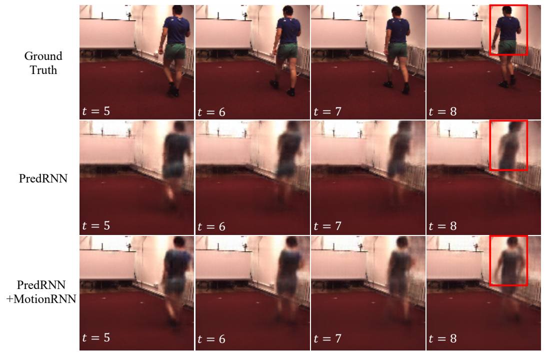

Figure 4. Prediction examples on the Human3.6M.

In human motion prediction (Figure 4), MotionRNN presents the details of human arms, while the predicted arms by PredRNN are vanished. In Figure 5, with MotionRNN, predictive models can generate the results with better sharpness. In synthetic V-MNSIT dataset (Figure 6), our approach can generate eidetic results of number “3”. These qualitative results show that MotionRNN can significantly improve the accuracy and sharpness of the prediction.

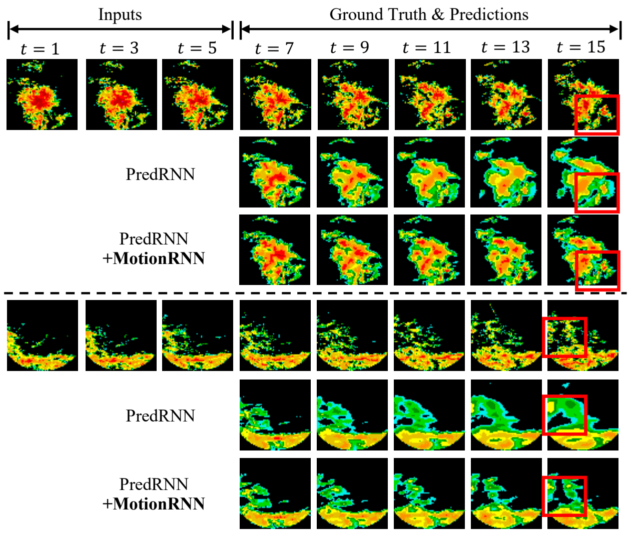

Figure 5. Prediction examples on the Radar Shanghai.

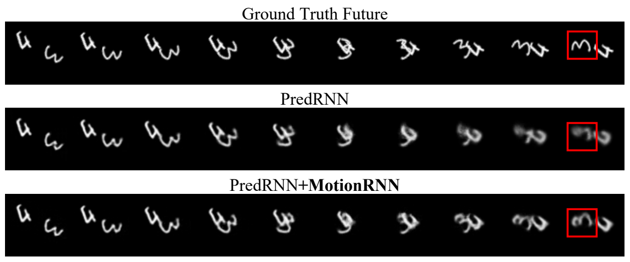

Figure 6. Prediction examples on the V-MNIST.

References

[1] 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. 2

[2] 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. 2

[3] 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. 2

[4] 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. 2

[5] 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. 2