Memory In Memory: Differential Recurrent Neural Networks for Spatiotemporal Prediction

Anonymous CVPR submission

Paper ID ****

Abstract

A spatiotemporal predictive model aims to generate future frames by observing existing context, in which stationarity and non-stationarity are two significant features. Most previous work based on recurrent networks, with relatively simple memory transition functions over time, mainly focus on the stationary component. However, it proves difficult for these models to handle the more complicated non-stationary variations. In this paper, we present a differential recurrent neural network named Memory In Memory (MIM). The core of our model is the MIM block that takes in the differential features between adjacent recurrent states, and models the spatiotemporal stationary and non-stationary patterns separately. It uses two self-renewed memory units to handle these two components. The MIM model achieves the state-of-the-art results on three spatiotemporal prediction datasets, including a synthetic dataset and two real datasets, and is believed to be a potentially generic architecture for time-series modeling.

1. Introduction

In recent years, spatiotemporal predictive learning techniques have made great improvements in real applications, including traffic flows prediction [32] and precipitation forecasting [22, 30]. A well-performed predictive learning system is expected to learn the intrinsic variations in the consecutive spatiotemporal context, which can be seen as a combination of two components: the stationary and the non-stationary.

A great challenge in spatiotemporal prediction is that most existing work, including the ConvLSTM network [22] and PredRNN [30], focuses on stationary variations with simple transition functions between the recurrent states. This leads to a phenomenon that the predicted frames generated by these models become increasingly blurred over time. It is possibly caused by the inconsistency between the non-stationary spatiotemporal dynamics and an approximately linear extrapolation system.

Although in these LSTM [10] based models, recurrent gate structures (especially the forget gate) are used to deliver, select, and discard information in the memory states temporal transitions paths. These gate structures are a little bit simple for capturing the complicated and non-stationary variations in the high-dimensional spatiotemporal data. In our preliminary experiments, we find that the forget gates in the PredRNN model [30] do not work in an appropriate way. For most of time, over 80 percent of them are saturated, implying that this model always remembers similar variations. In another word, what is learned is mainly the stationary component.

Inspired by this observation, we focus on a more appropriate memory transition function for spatiotemporal modeling. We assume that this transition function is a combination of the stationary and non-stationary variations, and treat them separately in a more explicit way. It is simpler to capture the stationary variations, such as a fixed speed of a moving object. It can be seemed as the very short-term trends in spatiotemporal data. The non-stationary component, such as the noise, the sudden changes, or the unexpected occlusions of multiple moving objects, evolves over time and is more complicated to capture. It could be partially tackled by solving the long-term reasoning problem. But there is still a long way towards spatiotemporal reasoning.

Our idea is more straightforward. Noting that the differential features are widely exploited in non-deep autoregressive models for non-stationary dynamics [20] while not fully exploited in deep networks, we propose a new recurrent block named Memory In Memory (MIM) to leverage the differential information between adjacent recurrent states. Inside the MIM block, we inject a second-order recurrent module to replace the original recurrent memory transitions. This module has two inner recurrent units respectively for capturing the non-stationary variations from the differential features, and the stationary variations. In LSTM terms, this module can be seen as an improved version of forget gate that models the memory state transitions. But in fact, it is no longer a standard gate structure. Instead, the two inner units hold their own memory cells and are self-renewed over time,

as the outer memory cell in the MIM does. This mechanism is therefore intuitively named "memory in memory". The proposed Differential Recurrent Neural Network (DRNN) achieves the state-of-the-art results on three video spatiotemporal prediction datasets, including a synthetic dataset, a real human activity dataset, and a real traffic flows dataset.

2. Related Work

In recent years, deep learning has made breakthroughs in several areas. Many great work based on convolutional neural network [15] and recurrent neural network [25] provides a lot of useful insights for spatiotemporal prediction. Inspired by language modeling, Ranzato *et al.* [21] defined a RNN architecture predicting frames in a discrete space of patch clusters. Srivastava *et al.* [24] defined a sequence to sequence LSTM framework for video prediction. But this sequence to sequence model can only capture temporal dependencies. Shi *et al.* [22] combined RNN and CNN by proposing a convolutional LSTM network, which learns spatial and temporal dependencies simultaneously. Finn *et al.* [8] developed an action-conditioned video prediction model that predicts a distribution over pixel motion from previous frames. Villegas *et al.* [27] also built a model upon ConvLSTMs with optical flow guided features. Patraucean *et al.* [19] defined a model that generates optical flow frames for every input frame, then applies the optical flow frame to the current frame to generate the next frame. Kalchbrenner *et al.* [12] proposed the Video Pixel Network (VPN) that encodes the time, space and color structures of video tensors as a four-dimensional dependency chain. This model uses the well-established PixelCNNs [26], and achieves sharp prediction results. However it suffers from a high computational complexity. Wang *et al.* [30] presented a spatiotemporal LSTM unit with dual memory structures. This structure has a zigzag memory flow and provides a great modeling capability of short-term modeling. Mathieu *et al.* [17] defined a CNN-based model using generative adversarial networks (GAN) [9, 7]. Vondrick *et al.* [28] also used adversarial learning to generate frames from noise vectors by applying it on 3D CNNs.

More recently, there have been some methods that aim to model the stochasticity in spatiotemporal predictive learning. These methods argue that future uncertainty is the cause of the blurry predictions. Henceforth, Variational Autoencoder (VAE) [14] is introduced to spatiotemporal predictive learning to generate stochastic latent variables and use them as a component in the prediction making process. Lee *et al.* [16] designed a similar approach to explicitly model the stochasticity in future predictions and combined it with the adversarial training method. Denton *et al.* [6] proposed another VAE based video prediction model that learns a prior of uncertainty in a given environment. These methods increase the diversity of the prediction results, but are difficult to evaluate and required to run a great number of times. In this paper, we focus on the deterministic part of spatiotemporal predictive learning, divide it into stationary and non-stationary modeling, and attempt to capture complex data variations in the historical observations.

3. Preliminaries

PredRNN [30] has made a remarkable improvement in video prediction. It allows the memory states and hidden states to be delivered in two directions: across cascaded RNN layers vertically and through all RNN states horizontally. This mechanism is implemented by the spatiotemporal LSTM (ST-LSTM) units, which almost double the gate structures in traditional convolutional LSTMs [22]. Via these gates, the output states of a certain ST-LSTM unit are determined by the spatiotemporal memory $\mathcal{M}_t^{l-1}$ passed from the previous layer, as well as the temporal memory $\mathcal{C}_{t-1}^{l}$ delivered from the previous time stamp. Calculations inside a spatiotemporal LSTM unit at time stamp t and layer l are shown as follows:

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

where σ is the sigmoid activation function, $*$ and $\odot$ denote the convolution and the Hadamard product operators respectively. The input gate i_t, input modulation gate g_t, forget gate f_t and output gate o_t control spatiotemporal information flows. The unit at time stamp t and layer l has four inputs: $\mathcal{X}_t$, which is either the input frame for $l = 1$ or the output hidden states by the previous layer $\mathcal{H}_t^{l-1}$ for $l > 1$; $\mathcal{H}_{t-1}^{l}$ and $\mathcal{C}_{t-1}^{l}$, the hidden states and memory states from the previous time stamp; as well as $\mathcal{M}_t^{l-1}$, the spatiotemporal memory states either from the top layer at the previous time stamp or the last layer at the current time stamp. All states are represented by $\mathbb{R}^{C\times W\times H}$ dimensional tensors, where the first dimension is the number of their channels, and the following two dimensions denote the height and width of feature maps. The structure of ST-LSTM is shown in Figure 2 (left).

The biggest highlight of PredRNN is its zigzag memory flow of the spatiotemporal memory $\mathcal{M}$. It provides PredRNN a great modeling capability of the short-term trends in longer

pathways through the vertical layers. Meanwhile, the temporal memory $\mathcal{C}$ offers this model a shorter pathway from previous inputs to later outputs [10], thus eases the vanishing gradient problem. This dual memory mechanism enables the PredRNN model to effectively model the shape deformations and motion trajectories in spatiotemporal sequences. We exploit this network as basic building blocks of our proposed model, as it can capture accurate spatial information. However, the original PredRNN also suffers the problem of blurry predictions as it still uses the simple forget gate inherited from ConvLSTM. In conclusion, the complex non-stationary dynamics cannot be fully captured by such simple temporal transitions.

4. Methods

In this section, we give detailed descriptions of the Differential Recurrent Neural Network architecture, which is designed to further deliver the hidden representations diagonally to generate differential features. As mentioned above, the differential features for non-stationary dynamics were not fully exploited in previous work. To address it, we further derive the Memory In Memory (MIM) block, which constitutes two cascaded temporal memory units to explicitly capture the stationary and the non-stationary variations respectively.

4.1. Differential Recurrent Neural Network

The architecture of Differential Recurrent Neural Network (DRNN) should enable deliveries of all necessary information flows to capture the long-term and short-term, stationary and non-stationary dynamics. A schematic of DRNN is shown in Figure 1. While the PredRNN model can fully capture the stationary dynamics, the non-stationary dynamics remain under-explored. Our key idea is to further model the differential dynamics between consecutive states, based on the **difference-stationary** assumption that differencing a non-stationary sequence will likely lead to a stationary one [20].

As the hidden states in PredRNN can be used as abstract representations of the input video frames, differencing the hidden states of two adjacent time stamps can reveal the variations of two input frames more evidently. Compared with PredRNN, a node at time stamp $t \neq 1$ and layer $l \neq 1$ takes the hidden state $\mathcal{H}_{t-1}^{l-1}$ from the node at time stamp $t-1$ and layer $l-1$ as extra inputs. These connections are shown as orange arrows in Figure 1. As the first layer doesn't have any previous layer, we simply use ST-LSTM [30] to generate its hidden presentations. Note that, the temporal differencing is performed by subtracting hidden state $\mathcal{H}_{t-1}^{l-1}$ from the hidden state h_t^{l-1} in MIM. As mentioned before, the Memory In Memory (MIM) block has two cascaded temporal memory units to capture the stationary and the non-stationary dynamics respectively. Thus, we modify the

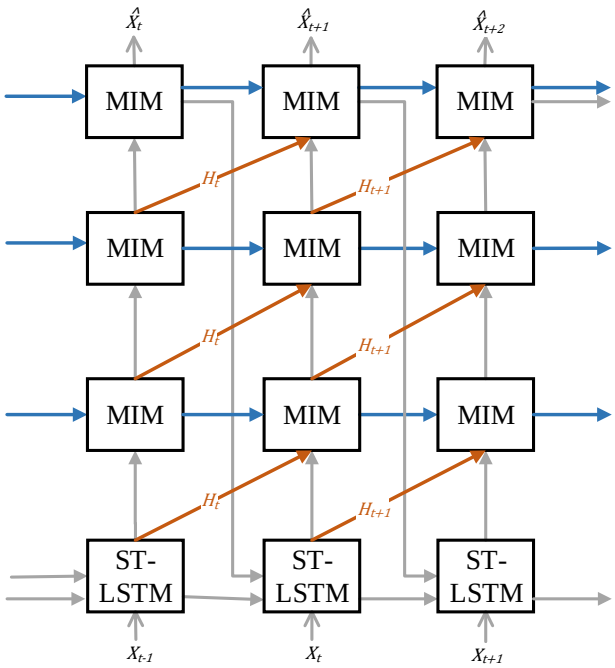

Figure 1: The structure of DRNN with 4 layers. The vertical (gray) arrows denote the zigzag information flows of the spatiotemporal memory $\mathcal{M}$ designed in PredRNN; the diagonal (orange) arrows are added to further deliver the hidden states to generate differential features $\mathcal{D}$; and the horizontal (blue) arrows are modified to deliver the temporal memory $\mathcal{C}$ designed in LSTM, as well as the stationary memory $\mathcal{S}$ and non-stationary memory $\mathcal{N}$ designed in the MIM.

horizontal (blue) arrows from PredRNN to further deliver the two temporal memories (denoted by $\mathcal{S}$ for stationary memory and $\mathcal{N}$ for non-stationary memory), as well as the learned differential representations (denoted by $\mathcal{D}$) from the previous time stamp.

4.2. Memory In Memory

Our observation is that, the complex dynamics in spatiotemporal sequences can be handled more effectively as a combination of stationary variations and non-stationary variations. Suppose we have a spatiotemporal sequence showing a walking man for example. The speed of the man can be seen as stationary variation and the swing of the man's arms should be considered as non-stationary variation, which we should make more effort to model them. Unfortunately, the forget gate in vanilla LSTM and ST-LSTM is a simple gating structure that struggles to capture the non-stationary variations in spatiotemporal sequences. In our preliminary experiments, we find that the majority of forget gates in the PredRNN model [30] are saturated, implying that the units always remember similar variations. In another word, what is learned is mainly the stationary component.

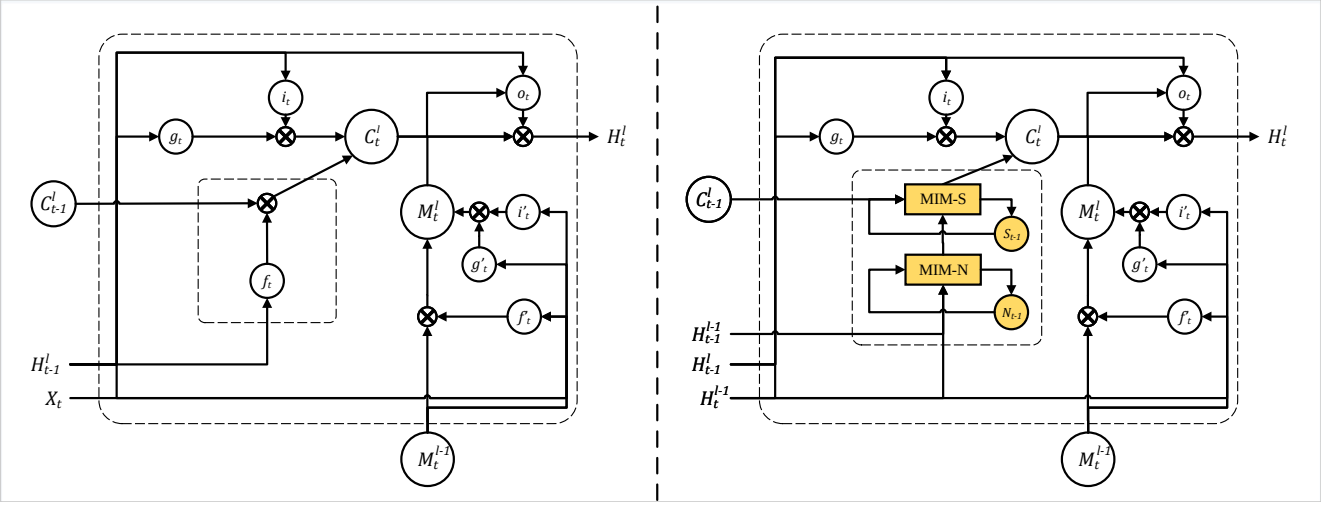

Figure 2: The ST-LSTM block [30] in the left plot and the proposed Memory In Memory (MIM) block in the right plot. MIM is designed to introduce two recurrent units (yellow squares) to replace the forget gate (dashed box) in ST-LSTM. MIM-S denotes the stationary unit and MIM-N denotes the non-stationary unit. Note that the MIM block cannot be used in the first layer so the input $\mathcal{X}_t$ is replaced by $\mathcal{H}_t^{l-1}$.

The Memory In Memory (MIM) block is enlightened by the idea of modeling the non-stationary variations using a series of cascaded memory transitions instead of the simple, saturation-prone forget gate in ST-LSTM. As compared in Figure 2 (the dashed boxes), two cascaded temporal memory recurrent units are designed to replace the temporal forget gate f_t in ST-LSTM. The first unit additionally taking $\mathcal{H}_{t-1}^{l-1}$ as input is used to capture the non-stationary variations based on the differencing $(\mathcal{H}_{t-1}^{l-1} - \mathcal{H}_t^{l-1})$ between two consecutive frames, so we name it **non-stationary unit** (shown as **MIM-N** in Figure 2). The non-stationary unit generates differential features $\mathcal{D}_t^l$ based on the difference-stationary assumption [20]. The other unit takes as inputs the output $\mathcal{D}_t^l$ of the non-stationary unit and the temporal memory $\mathcal{C}_t^{l-1}$ of the outer ST-LSTM to capture the general stationary variations in spatiotemporal sequences, so we call it **stationary unit** (shown as **MIM-S** in Figure 2).

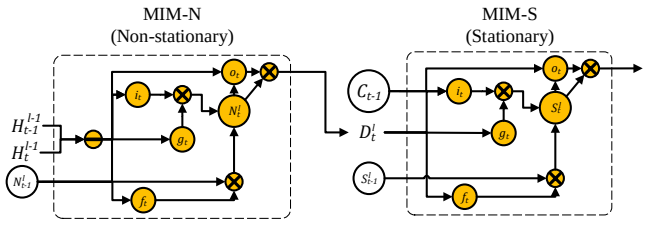

Figure 3: Non-stationary unit (MIM-N) and stationary unit (MIM-S), which are organized in a cascade structure in the MIM block. Note that non-stationary variation is modeled by sequence differencing.

It is important to emphasize that, by differencing the consecutive frames, the non-stationary variations are made more prominent while the stationary variations are significantly suppressed. Thus, the non-stationary unit will not memorize much information from the previous state. By replacing the forget gate in ST-LSTM with the output of the cascaded non-stationary and stationary units (as shown in Figure 2), the non-stationary dynamics can be captured more effectively. Key calculations inside a MIM unit can be shown as follows:

$$
\begin{aligned}
g_t &= \tanh(W_{xg} * \mathcal{H}_t^{l-1} + W_{hg} * \mathcal{H}_{t-1}^{l} + b_g) \\
i_t &= \sigma(W_{xi} * \mathcal{H}_t^{l-1} + W_{hi} * \mathcal{H}_{t-1}^{l} + b_i) \\
\mathcal{D}_t^l &= \text{MIM-N}(\mathcal{H}_t^{l-1}, \mathcal{H}_{t-1}^{l-1}, \mathcal{N}_{t-1}^{l}) \\
\mathcal{C}_t^l &= \text{MIM-S}(\mathcal{D}_t^l, \mathcal{C}_{t-1}^{l}, \mathcal{S}_{t-1}^{l}) + i_t \odot g_t \\
g_t' &= \tanh(W_{xg}' * \mathcal{H}_t^{l-1} + W_{mg} * \mathcal{M}_t^{l-1} + b_g') \\
i_t' &= \sigma(W_{xi}' * \mathcal{H}_t^{l-1} + W_{mi} * \mathcal{M}_t^{l-1} + b_i') \\
f_t' &= \sigma(W_{xf}' * \mathcal{H}_t^{l-1} + W_{mf} * \mathcal{M}_t^{l-1} + b_f') \\
\mathcal{M}_t^l &= f_t' \odot \mathcal{M}_t^{l-1} + i_t' \odot g_t' \\
o_t &= \sigma(W_{xo} * \mathcal{H}_t^{l-1} + W_{ho} * \mathcal{H}_{t-1}^{l} + W_{co} * \mathcal{C}_t^l + W_{mo} * \mathcal{M}_t^l + b_o) \\
\mathcal{H}_t^l &= o_t \odot \tanh(W_{1\times1} * [\mathcal{C}_t^l, \mathcal{M}_t^l]),
\end{aligned}
\tag{2}
$$

where MIM-N and MIM-S denote the stationary unit and the non-stationary unit respectively; $\mathcal{S}$ and $\mathcal{N}$ denote the horizontally-transited memory cells in the two corresponding recurrent units; $\mathcal{D}$ denotes the differential features, which are learned by the non-stationary unit and fed into the stationary unit. The cascade structure enables end-to-end modeling of both stationary and non-stationary dynamics.

A schematic of the stationary and non-stationary units is presented in Figure 3. We present the detailed calculations

of the non-stationary unit (MIM-N) as follows:

$$
\begin{aligned}
g_t &= \tanh(W_{xg} * (\mathcal{H}_t^{l-1} - \mathcal{H}_{t-1}^{l-1}) + W_{ng} * \mathcal{N}_{t-1}^{l} + b_g) \\
i_t &= \sigma(W_{xi} * (\mathcal{H}_t^{l-1} - \mathcal{H}_{t-1}^{l-1}) + W_{ni} * \mathcal{N}_{t-1}^{l} + b_i) \\
f_t &= \sigma(W_{xf} * (\mathcal{H}_t^{l-1} - \mathcal{H}_{t-1}^{l-1}) + W_{nf} * \mathcal{N}_{t-1}^{l} + b_f) \\
\mathcal{N}_t^l &= f_t \odot \mathcal{N}_{t-1}^{l} + i_t \odot g_t \\
o_t &= \sigma(W_{xo} * (\mathcal{H}_t^{l-1} - \mathcal{H}_{t-1}^{l-1}) + W_{no} * \mathcal{N}_t^{l} + b_o) \\
\text{MIM-N}(\mathcal{H}_t^{l-1}, &\mathcal{H}_{t-1}^{l-1}, \mathcal{N}_{t-1}^{l}) = o_t \odot \tanh(\mathcal{N}_t^l),
\end{aligned}
\tag{3}
$$

where all gates g_t, i_t, f_t and o_t are updated by the frame difference $(\mathcal{H}_t^{l-1} - \mathcal{H}_{t-1}^{l-1})$, which highlights the non-stationary variations in the spatiotemporal sequence, and focuses the unit more on the non-stationary dynamics.

Similarly, the detailed calculations of the stationary unit (MIM-S) are shown as follows:

$$
\begin{aligned}
g_t &= \tanh(W_{dg} * \mathcal{D}_t^{l} + W_{cg} * \mathcal{C}_{t-1}^{l} + b_g) \\
i_t &= \sigma(W_{di} * \mathcal{D}_t^{l} + W_{ci} * \mathcal{C}_{t-1}^{l} + b_i) \\
f_t &= \sigma(W_{df} * \mathcal{D}_t^{l} + W_{cf} * \mathcal{C}_{t-1}^{l} + b_f) \\
\mathcal{S}_t^l &= f_t \odot \mathcal{S}_{t-1}^{l} + i_t \odot g_t \\
o_t &= \sigma(W_{do} * \mathcal{D}_t^{l} + W_{co} * \mathcal{C}_{t-1}^{l} + W_{so} * \mathcal{S}_t^{l} + b_o) \\
\text{MIM-S}(\mathcal{D}_t^{l}, &\mathcal{C}_{t-1}^{l}, \mathcal{S}_{t-1}^{l}) = o_t \odot \tanh(\mathcal{S}_t^l),
\end{aligned}
\tag{4}
$$

which takes the memory states $\mathcal{C}_{t-1}^{l}$ and the differential features $\mathcal{D}_{t-1}^{l}$ of MIM-N as input. As can be validated, the stationary unit provides a gating mechanism to adaptively decide whether to trust the original memory or the differential feature of MIM-N. If the differential features vanish, indicating that the non-stationary dynamics is not prominent, then the unit will mainly reuse the original memory. Otherwise, if the differential features are prominent, then the unit will overwrite the memory to focus more on the non-stationary dynamics.

5. Experiments

In this section, we perform extensive evaluation of the proposed Memory in Memory (MIM) approach. For each evaluation dataset, we will introduce the dataset details and the implementation details of our proposed approach on it. At last, we report the performance of our MIM model and analyze experimental results both qualitatively and quantitatively.

We use three video prediction datasets: a synthetic dataset with moving digits, a real video dataset with human actions and a standard traffic flow prediction dataset. Our model has 4 layers in all experiments, including a ST-LSTM layer as the first layer and three MIM blocks. The number of channels in each MIM block is 64, as a balance of prediction accuracy and memory cost efficiency. We train all of the models with L2 loss, using the ADAM optimizer [13] with a 0.001 starting learning rate. The training process is stopped after $100,000$ iterations. The batch size of each iteration is set to 8. Note that we extend the layer normalization [2] to 3D tensors and apply it on the MIM and ConvLSTM models, based on the idea that the training process of deep convolutional networks can be stabilized by reducing the covariate shift problem [11]. Note that the MIM blocks in the first time stamp do not have any previous hidden representations as input, so the non-stationary units take matrices filled with zero as initialization. All experiments are implemented in TensorFlow [1].

Baseline Models: We mainly focus the comparative study with the following state-of-the-art methods: **ConvLSTM** [22] is the first deep network that uses convolution in LSTM and shows great results on video prediction; **FRNN** Oliu *et al.* [18] uses ConvGRU units and shares states to avoid re-encoding of the predictions. The ConvGRU nodes from encoder do not share weights with the nodes from decoder in this method, which is inherited from the encoder-predictor recurrent architecture [25, 4]; **VPN** [12] has shown a great performance in many video prediction tasks. As a strong competitor, we include VPN model to demonstrate the importance of exploiting the differential patterns and the cascaded memory recurrent units.

5.1. Moving MNIST Dataset

Implementation Moving MNIST dataset consists of $64 \times 64 \times 1$ grayscale sequences of length 20 displaying pairs of digits moving around the image (10 for the inputs and 10 for the predictions). The sequences are generated by the method described in the work of Srivastava *et al.* [24], following the experimental settings in PredRNN [30]. Before training the model, we apply max-min normalization to scale the data from its original intensities to $[0, 1]$. In particular, to reduce the training time and memory usage on the Moving MNIST dataset, we reshape each $64 \times 64 \times 1$ input image into a $16 \times 16 \times 16$ tensor. By doing this, we significantly reduce the parameters and training time of MIM, while the performance is affected marginally. Besides, the scheduled sampling strategy [3] is applied to all of the models to stitch the discrepancy between training and inference.

Results We measure the per-frame structural similarity index measure (SSIM) [31], the mean square error (MSE) and the mean absolute error (MAE) to evaluate our models, as shown in table 1. A lower RMSE or MAE denotes better prediction performance, so does a higher SSIM. Additionally, we also include FC-LSTM [24], TrajGRU [23], CDNA [8] and DFN [5] for comparison. MIM reduces the former best MSE and MAE results to 52.1 and 120.8 respectively. The SSIM result also reaches 0.874. We also design two ablation experiments, by respectively removing the stationary units or non-stationary units, to verify necessity of the cascaded temporal memory recurrent units. As table 2 shown, non-

Table 1: A comparison for predicting 10 frames on the moving MNIST dataset. All of the models are trained with 10 target frames and have comparable numbers of parameters.

MODEL	SSIM	MSE	MAE
FC-LSTM	0.690	118.3	209.4
CONVLSTM	0.707	103.3	182.9
TRAJGRU	0.713	106.9	190.1
CDNA	0.721	97.4	175.3
DFN	0.726	89.0	172.8
FRNN	0.813	69.7	150.3
VPN	0.870	64.1	131.0
PREDRNN	0.867	56.8	126.1
MIM	**0.874**	**52.0**	**116.5**

Table 2: Ablation study of the MIM block with either stationary unit or non-stationary unit removed. These experiments also predict 10 frames on the moving MNIST dataset.

MODEL	SSIM	MSE	MAE
MIM (WITHOUT MIM-N)	0.858	54.4	124.8
MIM (WITHOUT MIM-S)	0.853	55.7	125.5
MIM	**0.874**	**52.0**	**116.5**

stationary units work better than stationary units and they work best when they are cascaded.

The sequences in Moving MNIST dataset have less variations as they are generated by two moving digits with fixed speed. To verify that our model can capture the variations (especially non-stationary variations) better, we add different acceleration to the speed and generate a new Moving MNIST dataset with variant speed. The results of PredRNN [30], PredRNN++ [29] and MIM are shown as table 3. These experiments also predict 10 frames based on 10 observations. Obviously, MIM achieves better performence on datasets which have more variations.

Table 3: A comparison for predicting 10 frames on the moving MNIST dataset with acceleration. All of the models are also trained with 10 target frames and have comparable numbers of parameters.

MODEL	SSIM	MSE	MAE
PREDRNN	0.	0.	0.
PREDRNN++	0.	0.	0.
MIM	**0.**	**0.**	**0.**

In Figure 6, we visualize the corresponding generated images. The predictions of FRNN are blurred, while PredRNN and MIM are better. Note that MIM predicts the shapes of the digits better, such as the upper part of the digit "7". Figure 4 shows the frame-wise MSE of 2 models (PredRNN and MIM) over the test sets. As the number of predicted frames grows, the curve of MIM gets lower, indicating that our cascaded temporal memory recurrent units are necessary.

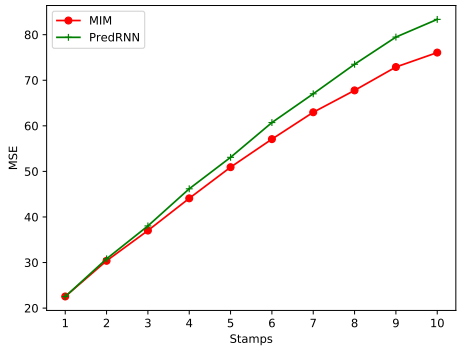

Figure 4: Frame-wise MSE over the MNIST test sets. Lower curves denote higher accuracy.

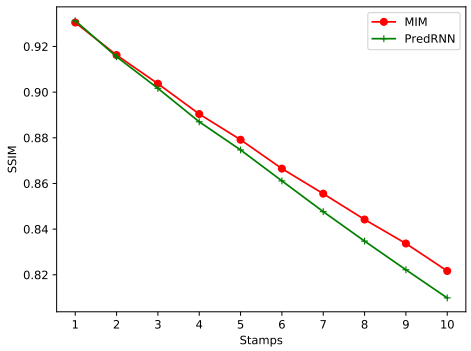

Figure 5: Frame-wise SSIM over the MNIST test sets. Higher curves denote higher accuracy.

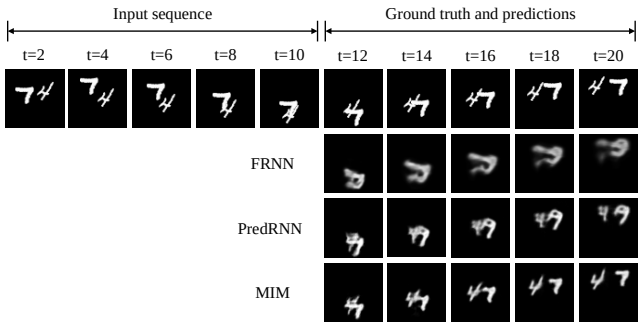

Figure 6: A demonstration of predictions on moving MNIST test set. All models predict 10 frames into the future by observing 10 frames. Frames are shown at a two frames interval.

Figure 5 shows the frame-wise SSIM of PredRNN and MIM, which confirms the same conclusion.

5.2. KTH Action Dataset

Implementation KTH consists of 600 videos of 15-20 seconds with 25 persons performing 6 human actions. Each frame in these sequences is grayscale and with the size of $120 \times 160 \times 1$, which will be resized into shapes of $128 \times 128 \times 1$ pixels. The dataset has been split into persons 1 to 16 for training, and 17 to 25 for testing, also following the experimental settings in [30]. In particular, all of the models are trained to generate subsequent 10 frames from prior 10 frames, and tested to predict 20 frames from 10 frames. We also apply max-min normalization to scale the data from the its original intensities to $[0, 1]$ and reshape each $128 \times 128 \times 1$ input image into a $32 \times 32 \times 16$ tensor in the same way as we did on moving MNIST dataset. The scheduled sampling strategy [3] is also applied to the experiments on KTH action dataset.

Table 4: A comparison for predicting 20 frames on the KTH action dataset. All of the models are trained with 10 target frames and made to predict 20 future frames at test time. They have comparable numbers of parameters.

MODEL	PSNR	SSIM
CONVLSTM	23.58	0.712
TRAJGRU	26.97	0.790
DFN	27.26	0.794
MCNET	25.95	0.804
FRNN	26.12	0.771
PREDRNN	27.55	0.801
MIM	**28.16**	**0.806**

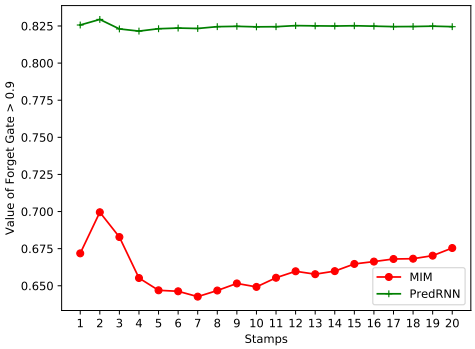

Figure 7: Percentage of value which greater than 0.9 from the forget gate (KTH dataset).

Results In the experiments on KTH action dataset, we use image-level PSNR and SSIM to evaluate our models, as shown in table 4. A higher PSNR denotes better prediction performance. We include MCNET [27] for comparison in these experiments. Note that all of the models are trained with 10 target frames and predicted 20 future frames. MIM consistently outperforms the baseline models. Figure 8 shows a person running from the near to the distant. Note that the person in the predictions of PredRNN keeps running nearby, while MIM makes the person run far away. To verify the necessity of our cascaded temporal memory recurrent units, we output the value of the forget gate from PredRNN and MIM and calculate the percentage of the value which is greater than 0.9, as shown in Figure 7. Note that we divide the outputs of the cascaded temporal memory recurrent units by the original cell state to get the "fake" forget gate value. Obviously, most of the forget gate value of PredRNN is greater than 0.9, while it is about 60% in MIM. Meanwhile, the value of "forget gate" in MIM is more volatile.

As shown in table 4, PredRNN++ [29] is still better than MIM. Considering most of the human actions in the KTH Action Dataset are simple, such as walking or jogging. We split the KTH Action Dataset into two subsets based on their variations. (methods). The results of PredRNN [30], PredRNN++ [29] and MIM on "stationary" KTH dataset are shown in table 5, while the results on "non-stationary" KTH dataset are shown in table 6. These experiments are also trained to generate subsequent 10 frames from prior 10 frames, and tested to predict 20 frames from 10 frames. Obviously, MIM achieves better performence on datasets which have more variations.

Table 5: A comparison for predicting 20 frames on the "stationary" KTH action dataset. All of the models are also trained with 10 target frames and made to predict 20 future frames at test time.

MODEL	SSIM	MSE	MAE
PREDRNN	0.	0.	0.
PREDRNN++	0.	0.	0.
MIM	**0.**	**0.**	**0.**

5.3. TaxiBJ Dataset

Implementation TaxiBJ contains traffic flow images collected consecutively from the GPS monitors of taxicabs in Beijing. Each frame in TaxiBJ is a $32 \times 32 \times 2$ grid of image. Two channels represent the traffic flow entering and leaving the same district at this time. We generate sequences from TaxiBJ dataset and split the whole dataset into a training set and a test set as described in the work of Zhang *et al.* [33]. Each sequence contains 8 consecutive frames, 4 for the inputs and 4 for the predictions. The frames are also scaled

Table 6: A comparison for predicting 20 frames on the "non-stationary" KTH action dataset. All of the models are also trained with 10 target frames and made to predict 20 future frames at test time.

MODEL	SSIM	MSE	MAE
PREDRNN	0.	0.	0.
PREDRNN++	0.	0.	0.
MIM	**0.**	**0.**	**0.**

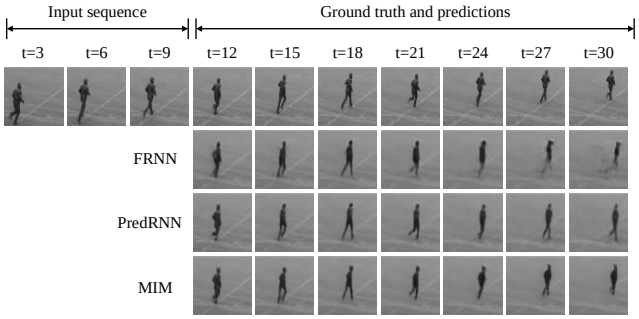

Figure 8: A demonstration of predictions on KTH action test set. All models predict 20 frames into the future by observing 10 frames. Frames are shown at a three frames interval.

to $[0, 1]$ and reshape to $16 \times 16 \times 8$ as described above. Particularly, the scheduled sampling strategy [3] is not used in these experiments.

Results The setting of the experiments on TaxiBJ dataset follows ST-ResNet [32], which has shown state-of-art results on TaxiBJ dataset. Note that ST-ResNet only predicts one frame in one pass due to its non-recurrent structures. By regarding previous generated images as inputs, we can easily make it predict longer into the future. Compared with PredRNN, MSE of MIM does not decrease much in the first few frames. But as the number of predicted frames increases, the gap becomes larger, shown as table 7. Figure 9 shows the corresponding generated images on TaxiBJ. For easy observation, we visualize the difference between the predictions and the ground truth images. Obviously, MIM shows good performance in all predicted frames, especially the last few frames.

6. Conclusions

We investigates the non-stationarity underlying the spatiotemporal sequences, which forms the main obstacle in spatiotemporal predictive learning. Modern recurrent networks are powerful for modeling stationary sequences, while the attention mechanism has been introduced to model the non-stationary patterns. However, in spatiotemporal predictive learning, the attention mechanism proves less effective. This paper enables non-stationary modeling in video prediction by a differential recurrent neural network taking advantages of the differential information between adjacent recurrent states. A Memory In Memory (MIM) block is derived to model the spatiotemporal variations, which uses two cascaded memory recurrent units to handle non-stationary and stationary spatiotemporal variations respectively. MIM achieves the state-of-the-art prediction performance on three video datasets: a synthetic dataset with moving digits, a real video dataset with human actions and a standard traffic flow prediction dataset.

Table 7: A comparison of MSE for predicting 4 frames on the TaxiBJ dataset. All of the models are trained with 4 target frames and have comparable numbers of parameters

MODEL	FRAME 1	FRAME 2	FRAME 3	FRAME 4
ST-RESNET	0.688	0.939	1.130	1.288
VPN	0.744	1.031	1.251	1.444
FRNN	0.682	0.823	0.989	1.183
PREDRNN	0.634	0.934	1.047	1.263
MIM	**0.554**	**0.737**	**0.887**	**0.999**

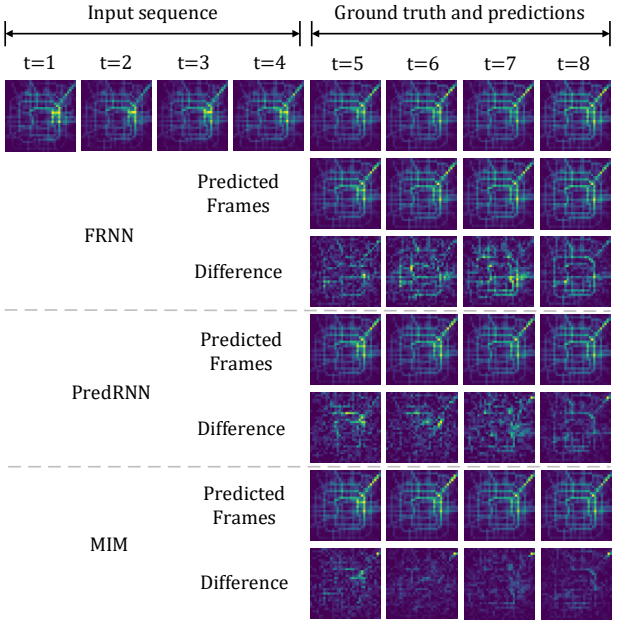

Figure 9: A demonstration of predictions on TaxiBJ test set. We also visualize the difference between the predictions and the ground truth images for easy observation

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016. 5

[2] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 5

[3] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 1171–1179, 2015. 5, 7, 8

[4] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014. 5

[5] B. De Brabandere, X. Jia, T. Tuytelaars, and L. Van Gool. Dynamic filter networks. In *NIPS*, 2016. 5

[6] E. Denton and R. Fergus. Stochastic video generation with a learned prior. *arXiv preprint arXiv:1802.07687*, 2018. 2

[7] E. L. Denton, S. Chintala, R. Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *NIPS*, pages 1486–1494, 2015. 2

[8] C. Finn, I. Goodfellow, and S. Levine. Unsupervised learning for physical interaction through video prediction. In *NIPS*, 2016. 2, 5

[9] I. J. Goodfellow, J. Pougetabadie, M. Mirza, B. Xu, D. Wardefarley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *NIPS*, 3:2672–2680, 2014. 2

[10] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 1, 3

[11] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 5

[12] N. Kalchbrenner, A. v. d. Oord, K. Simonyan, I. Danihelka, O. Vinyals, A. Graves, and K. Kavukcuoglu. Video pixel networks. In *ICML*, 2017. 2, 5

[13] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 5

[14] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012. 2

[16] A. X. Lee, R. Zhang, F. Ebert, P. Abbeel, C. Finn, and S. Levine. Stochastic adversarial video prediction. *arXiv preprint arXiv:1804.01523*, 2018. 2

[17] M. Mathieu, C. Couprie, and Y. LeCun. Deep multi-scale video prediction beyond mean square error. In *ICLR*, 2016. 2

[18] M. Oliu, J. Selva, and S. Escalera. Folded recurrent neural networks for future video prediction. *arXiv preprint arXiv:1712.00311*, 2017. 5

[19] V. Patraucean, A. Handa, and R. Cipolla. Spatio-temporal video autoencoder with differentiable memory. In *ICLR Workshop*, 2016. 2

[20] D. B. Percival and A. T. Walden. *Spectral Analysis for Physical Applications*. Cambridge University Press, 1993. 1, 3, 4

[21] M. Ranzato, A. Szlam, J. Bruna, M. Mathieu, R. Collobert, and S. Chopra. Video (language) modeling: a baseline for generative models of natural videos. *arXiv preprint arXiv:1412.6604*, 2014. 2

[22] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *NIPS*, pages 802–810, 2015. 1, 2, 5

[23] X. Shi, Z. Gao, L. Lausen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. Woo. Deep learning for precipitation nowcasting: A benchmark and a new model. In *Advances in Neural Information Processing Systems*, 2017. 5

[24] N. Srivastava, E. Mansimov, and R. Salakhutdinov. Unsupervised learning of video representations using lstms. In *ICML*, 2015. 2, 5

[25] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. *NIPS*, 4:3104–3112, 2014. 2, 5

[26] A. van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, et al. Conditional image generation with pixelcnn decoders. In *NIPS*, pages 4790–4798, 2016. 2

[27] R. Villegas, J. Yang, S. Hong, X. Lin, and H. Lee. Decomposing motion and content for natural video sequence prediction. In *International Conference on Learning Representations (ICLR)*, 2017. 2, 7

[28] C. Vondrick, H. Pirsiavash, and A. Torralba. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems*, pages 613–621, 2016. 2

[29] Y. Wang, Z. Gao, M. Long, J. Wang, and P. S. Yu. Predrnn++: Towards a resolution of the deep-in-time dilemma in spatiotemporal predictive learning. *arXiv preprint arXiv:1804.06300*, 2018. 6, 7

[30] Y. Wang, M. Long, J. Wang, Z. Gao, and S. Y. Philip. Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms. In *Advances in Neural Information Processing Systems*, pages 879–888, 2017. 1, 2, 3, 4, 5, 6, 7

[31] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *TIP*, 13(4):600, 2004. 5

[32] J. Zhang, Y. Zheng, and D. Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *AAAI*, pages 1655–1661, 2017. 1, 8

[33] J. Zhang, Y. Zheng, D. Qi, R. Li, and X. Yi. Dnn-based prediction model for spatio-temporal data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 92. ACM, 2016. 7