Robust Spatiotemporal Convolutional Long Short-Term Memory Algorithm for Video Prediction

Wael Saideni, Fabien Courrèges, David Helbert, Jean Pierre P Cances

Robust Spatiotemporal Convolutional Long Short-Term Memory Algorithm for Video Prediction

Wael SAIDENI
XLIM Research Institute
UMR CNRS 7252
Limoges, France
wael.saideni@xlim.fr

Fabien COURREGES
XLIM Research Institute
UMR CNRS 7252
Brive-la-Gaillarde, France
fabien.courreges@unilim.fr

David HELBERT
XLIM Research Institute
UMR CNRS 7252
Poitiers, France
david.helbert@univ-poitiers.fr

Jean Pierre CANCES
XLIM Research Institute
UMR CNRS 7252
Limoges, France
cances@ensil.unilim.fr

***Abstract*—The use of recurrent neural networks in several applications has allowed to capture impressive results, especially in various applications such as video prediction and it has become a promising direction of scientific research. In this paper, we introduce a novel algorithm for video prediction called "Robust Spatiotemporal Convolutional Long Short-Term Memory (Robust-ST-ConvLSTM) Algorithm" that outperforms the state-of-the-art approaches. Robust-ST-ConvLSTM is a memory flow algorithm based on higher order ConvLSTM. This memory flow algorithm is holding the spatiotemporal information to optimize and control the prediction abilities of the ConvLSTM cell. Our approach is developed in the specific context of predicting future frames based on historical observations, and we experimentally validate the ability of the proposed algorithm on two spatiotemporal datasets, including a moving variant of MNIST dataset of handwritten digits, and KTH which is a human motion dataset.**

***Index Terms*—Video prediction, deep learning, neural networks, computer vision, ConvLSTM, memory flow, hidden states**

I. Introduction

Video prediction or predicting what happens in the next frames is the key component of intelligent decision making systems. It is also, an emerging field of computer vision and deep learning that is facing many challenges. Actually, these predictive systems have many real-world applications such as video surveillance or human and buildings security which is one of the most frequently debated issues nowadays.

Video prediction networks are based on historical information gathered from continuous and unlabeled video frames. These networks aim to forecast future frames in a video after having some previous images. Formally, we suppose $X_t \in \mathbb{R}^{w \times h \times c}$ is the t-th frame of a dynamic scene $X = (X_{t-n}, ..., X_t)$ with n frames, where w, h, and c denote width, height and number of channels, respectively. The main target from this project is to predict the next m frames $Y = (Y_{t+1}, ..., Y_{t+m})$ from the input X.

Since deep learning has demonstrated its performances and effectiveness in video processing and because feature extraction from images using traditional supervised machine learning methods is challenging, deep learning for video prediction could be a promising research direction and a very powerful tool for better performances. However, despite the huge advancement is deep learning models, video prediction remains a big challenge in terms of image deblurring and long-term prediction. Therefore, we propose in this paper a new algorithm called Robust Spatiotemporal Convolutional Long Short-Term Memory (Robust-ST-ConvLSTM) algorithm for video prediction as a long-term prediction algorithm that can forecast more than 20 frames. Also, it outperforms the state-of-the-art architectures. Thus, we empirically demonstrate the performance of our model on the main two challenges mentioned above. In fact, the main cell of our architecture is a modified ConvLSTM cell. Actually, ConvLSTM architecture is commonly used for spatiotemporal predictive systems with a traditional roadway for the memory state. However, this approach is outdated and not optimal for many reasons and that's why some changes on its main structure have been done. Firstly, the predictive algorithms based on ConvLSTM focus on the long-term stochastic properties of the data rather than its spatial distortion. Nevertheless, for video prediction, spatial and temporal data structure are important and should be taken into account. Secondly, the spatiotemporal cell state designed will handle low-level and semantic aspects of the spatiotemporal video data which are significantly important to generate future frames. Thirdly, the memory flow will propose new cell state and hidden state transition functions different from the ordinary ones which have not fully used the homogeneity of the input and the output space. Finally, the ConvLSTM model proposes an explicit temporal information encoding in each cell [32]. This first-order Markovian architecture updates the hidden states with information from the previous time step only which harden the capturing of long-range temporal correlations. In addition, most first-order RNN, i.e the current hidden state is updated based on one previous hidden state, suffer from the gradient vanishing problem in back-propagation resulting in a difficulty in learning RNN to model long-term dependency in data [33].

The remaining paper is organized as follows: Section II discusses related works. In Section III, we describe the main idea behind our proposed algorithm and its key components. In Section IV, we evaluate the capability of Robust-ST-ConvLSTM for multi-step video prediction on two spatiotemporal datasets, including a synthetic dataset of handwritten digits and a human motion dataset and report its performance by comparing it against the state-of-the-art

algorithms. Finally, Section V provides conclusion and the future research directions.

II. Related Works

A. Optical flow based methods

Many research projects have proposed video prediction solutions based on optical flow or dense trajectory [1]–[4]. In fact, optical flow is applied to report motion information about objects of successive frames. Technically, these approaches take the given dynamic scene as input to forecast the optical flow of the future frame. The obtained result is then merged with the last input frame to generate the future predicted video frame. However, those approaches that necessitate supervised training, use training datasets that contain optical flow information which is not obviously provided in the commonly used video datasets.

B. Deep Learning based methods

While the optical flow based models use the motion information to predict the frames, neural network approaches analyze the frames and extract their features in order to exploit the spatiotemporal representation to forecast the next frames. In this section, recent deep learning models for video prediction will be discussed after being classified into three categories: recurrent neural networks, convolutional networks and generative networks.

Although these neural networks based methods are better than the traditional optical-flow-based solutions in terms of performances, they are challenging and produce sometimes blurry results. Obviously, it is a promising research area.

1) Recurrent models

Recurrent networks are commonly used for video sequences related problems since they are considered as sequential data with spatio-temporal representation.

Recurrent neural networks (RNN) have demonstrated considerable success in video prediction research works that are detailed in [7]–[23] . In fact, along with the advancements in neural networks architectures, video prediction has been studied extensively in recent years.

Zhang et al. proposed a ConvLSTM-based architecture where hidden states are updated along a z-order curve [12]. The model presents a novel training approach based on two Z-Order Recurrent Networks (Znet): Znet-Predictor and Znet-Probe. Since the majority of video prediction algorithms based on ConvLSTM have duplicated features with same functionality in both cell state and hidden state of the LSTM unit, Znet came up with a novel routes updating to enhance the hidden states. Technically, to trick the neural network, the model is set to choose inputs that minimize the loss function instead of updating weights and biases that minimize the cost.

W. Lotter et al. [15] presented a predictive neural network (PredNet) architecture. This network aims to forecast future video frames in dynamic scenes. Technically, every layer in the network makes local predictions and only sends the deviations from those predictions to the following layers. The PredNet model is a series of recurrent blocks that make local predictions, then subtracted from the input before being forwarded to the subsequent network layer.

C. Lu et al. [18] propose a Flexible Spatio-Temporal Network (FSTN). This model enables the generation of the frames lying between the observed frames in order to output slow-motion video sequences. Also, it proposes a novel loss function to optimize the training phase of the model. The architecture described above is based on two main models: extrapolation model and interpolation model. Both of them are considered as spatio-temporal autoencoders. However, the extrapolation model has a guided training phase by the the ground truth frames feeding each layer by the supervised information needed, while the interpolation model does not need the ground truth images. Another difference of the two models lies in their definition. The interpolation is the estimation of a value between given data points but the extrapolation is useful when looking for a value that is either higher or lower than the values in the dataset.

A recent RNN architecture was proposed by Wang et al. in [24]. The idea behind this research work remain behind the new spatiotemporal LSTM (ST-LSTM) unit that take out and memorize spatial appearances and temporal variations simultaneously since for video prediction we need to consider both the spatial and the temporal structures. In fact, the Predictive Recurrent Neural Network (PredRNN) is based on spatiotemporal memory flow which allows the memory cells to move vertically across stacked RNN layers and horizontally through all RNN states. This approach is different from stacked LSTM. Actually, in stacked LSTM, memory states are updated independently from the visual features which means that the first layer of the present time step could ignore the information memorized by the last layer at the previous time step. However, in PredRNN, a memory cell is introduced to handle the information between different time steps. Another problem is solved in [31]. The new memory cell can handle long-term and short-term information at the same time which can limit the predictive performances of the model. So, a pair of memory cells is used and and explicitly decoupled in order to satisfy the different variations. This model reduce the loss of visual information from the very first layer to the top of the recurrent network. Furthermore, another learning strategy was proposed called reverse scheduled sampling. This strategy enables to learn temporal dynamics from longer periods of the input video and reduce the training discrepancy between the encoding network and the prediction network.

2) Convolutional models

Different from recurrent neural networks, convolutional networks are feed-forward neural networks that are commonly used for computer vision challenges such as visual prediction. Many models are based on convolutions for video prediction. One of these architectures is a multi-model combining temporal and spatial sub-networks which is proposed in [25] and called MixPred. The future frame prediction approach

described is divided into two parts: a temporal model for modeling the time series of the input video and a spatial sub-network to model the spatial texture on the content. Then, the authors tested an information fusion method for feature map interaction between the two parts. This approach allows to copy the unchanging pixels from the last frame thanks to the temporal mask which means that the predicted frame has the same clearness as the original frame. Also, synchronously exchanging temporal and spatial information enables to fill the changed pixels in order to have a complete predicted image. This model uses only convolutional layers but it could be theoretically enhanced by using another models like the generative networks. However, the architecture of this approach, in particular the choice of the hyperparameters, differs with the model which their performances are compared to. The model described above could be used not only in future frames prediction but also in several applications such as object tracking, action recognition and video compression.
In [26], the model trains a deep neural network to generate video frames by flowing pixel values from existing ones instead of initializing them from scratch. The model, called Deep Voxel Flow (DVF), takes usually 3 frames from the video scene without pre-processing: two frames are taken as input and the third frame is used as the generated target. This approach is based on the idea of borrowing voxels (3D-pixels) from the adjacent frames to generate more realistic results. The architecture is composed of a convolutional encoder-decoder to forecast the voxel flow and a volume sampling layer to generate the target image.
As in [25], the model can predict the in-between frames (interpolation) and the future frames (extrapolation) of the input dynamic scene. The voxel flow, used to sample the input frames with the volume sampling function to synthesize the target frame, has two main components: the spatial component and the temporal one. The spatial element is the optical flow for the predicted frame and the temporal part is used to form a color in that frame.
The framework described above aims to predict one frame but it can naturally be extended to a multi-frame prediction framework with a fairly simple manipulation. In fact, the target becomes a 3D volume and not 2D image and the learning rate will be reduced to maintain stability in the training phase. In addition, the spatiotemporal coherence is maintained because of the preserve of local correlations due to the convolutions across the temporal layers.
The strength of this model is that it combines the advantages of the optical-flow-based approach and the newer neural-network-based models. Also, it can be trained and tested on any real-world video with any resolution. However, it fails in scenes with repetitive patterns. Also, it generates some blurry scenes, like most of neural-network-based implementations.

3) Generative models

Generative models are used to generate new samples from the same distribution as the input data. The target behind training generative models is to learn a probability distribution that is similar to the data's probability distribution. In video prediction, the models described above aim to output a single eventual outcome. However, generative approaches generate a wide spectrum of feasible predictions.
The most common network structure in the field of video prediction and image generation in general is Generative Adversarial Networks (GAN). These networks are composed of two sub-networks jointly trained, the discriminator and the generator, to create fake samples that look like real data. Technically, the generator fools the discriminator by generating new samples from a random noise (e.g. Gaussian noise). Then the discriminator features the probability distribution function that describes real data. Nevertheless, in video prediction, some conditions could be added to the general implementation of GAN in order to forecast the future frames.
In [27], a generative approach was proposed to prediction frames based on cycle GAN. The main model is composed of one generator and two discriminators. In fact, the generator uses the retrospective cycle to predict both future and past frames and we train it with reversed input sequences. Moreover, one discriminator is dedicated to identify fake frames while the other is implemented to distinguish the sequences that contain fake frames which is crucial in forecasting temporally consistent frames. Technically, the loss function and the network architecture make this approach special when we compare it with the general formulation of GAN networks. Since this model enables to predict a limited number of frames before generating blurry images, a multi-frame prediction strategy is employed. The model starts by forecasting the next frame from an input video. Then, it constructs a new input video by concatenating the last frames of the input video and the predicted frame. Finally, the new input video will enable the prediction of the next frame. This strategy is repeated until we get the desired number of predicted frames.
In [28], the authors insisted on the fact that conditional Generative Adversarial Networks (cGAN) are suitable for video frames prediction because it can guarantee the spatio-temporal coherence between the predicted frames and the input video.
Another approach is discussed in [29] and is based on the idea of dividing the video signal into two parts: content and motion. Content to specify the objects in the sequence and motion to describe their movements. The model is based on mapping a sequence of random vectors to a sequence of frames in order to generate the predicted videos. These random vectors are composed of two parts: one for the content and the other for the motion. Since this framework is based on GAN, discriminators are used to learn motion and content decomposition in an unsupervised way by introducing a new adversarial learning scheme.

III. Overview of the proposed Robust Spatiotemporal ConvLSTM algorithm

A. From LSTM to ConvLSTM

The idea behind the proposed algorithm is based on Convolutional LSTM (ConvLSTM) which is Long Short Term

Memory (LSTM) network applied on high dimensional data.

1) LSTM

Long short Term Memory Network is considered as an advanced type of RNN that was designed and developed by Hochhreiter and Schmidhhuber (1997) [34] to solve the vanishing gradient problem of standard RNNs. Theoretically, RNNs are designed to learn long term dependencies. However, in practice, many issues appear such as vanishing gradient that prevents those neural network to learn long term dependencies. Therefore, it has been proven that LSTM is a powerful tool to remember information for longer period of time. Indeed,the main idea behind LSTM consists of connecting the previous information to the future task.

The main structure of LSTM based neural networks is the same: it consists of a chain of LSTM modules. However, the structure of those modules depends on the application.

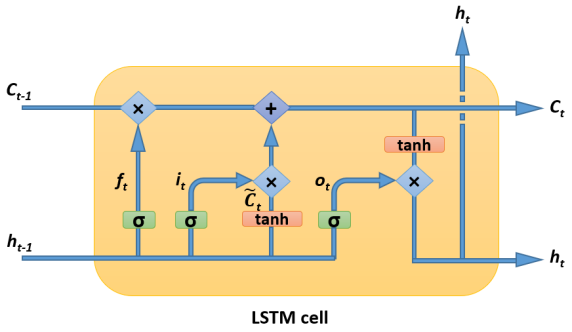

Fig. 1. The structure of a standard LSTM module

One of the most powerful components of LSTM is the cell state which is represented by the horizontal line on the top of Figure 1. It is used to handle the main information through the whole network and from one LSTM block to another. This function is controlled by 3 different structures: the forget gate, the input gate and the output gate.

When we look at the LSTM cell in Figure 1, we notice that the information coming from the cell state c_t passes through the forget gate that decide which information is going to be forgotten thanks to the sigmoid layer that outputs a number between 0 and 1. Then, the input gate uses the input x_t and the hidden state h_{t-1} to update the cell state. Then, a tanh layer outputs new candidate values $\hat{c}_t$ that have the possibility to be added to the cell state. The cell state update is created from the combination of $\hat{c}_t$ and i_t.

Now, everything is ready to update the cell state. Firstly, the old cell state c_{t-1} is multiplied by f_t then $i_t * \hat{c}_t$ is added. Finally, the output of the LSTM unit will be based on the cell state c_t, the input x_t and the hidden state h_{t-1}. Indeed, a sigmoid gate is applied to decide the parts of the cell state that will be involved in the output process. Then, the cell state c_t is put through tanh and then multiplied by the output of the sigmoid layer. The main target of this last step is to output the new hidden state h_t. To sum up the mechanism of LSTM: This neural network unit has 3 inputs: the input x_t, the cell state c_{t-1} and the hidden state h_{t-1} that will be passed through 3 different gates in order to output 2 structures: the cell state c_t and the new hidden state h_t. The mechanism described above is explained by the following equations:

$$\begin{aligned} i_t &= \sigma(w_i \times x_t + s_i \times h_{t-1}) \\ f_t &= \sigma(w_f \times x_t + s_f \times h_{t-1}) \\ o_t &= \sigma(w_o \times x_t + s_o \times h_{t-1}) \\ \hat{c}_t &= tanh(w_{\hat{c}} \times x_t + s_{\hat{c}} \times h_{t-1}) \\ c_t &= f_t \circ c_{t-1} + i_t \circ \hat{c}_t \\ h_t &= o_t \circ tanh(\hat{c}_t) \end{aligned} \tag{1}$$

Where σ is the sigmoid function, $\times$ is a simple multiplication and $\circ$ denotes the Hadamard product.

2) ConvLSTM

Although LSTM is considered as a powerful network for dealing with temporal relationship, its main drawback is that it is unable to handle spatial information because we need to flatten high dimensional data to 1D vectors to be compatible to the input common structure. However, Spatiotemporal data are commonly used is many applications such as video surveillance. So, we were forced to look for a new structure where we take advantage of LSTM by integrating spatiotemporal data. Convolutional LSTM is used to capture the spatial dimension for the prediction mode. The special feature of ConvLSTM is that the inputs x_t, the cell states c_t, the hidden states h_t and the 3 gates are 3D tensors. In addition, the convolution operation is used instead of simple matrix multiplication as shown in the following equations:

$$\begin{aligned} I_t &= \sigma(W_i * X_t + S_i * H_{t-1}) \\ F_t &= \sigma(W_f * X_t + S_f * H_{t-1}) \\ O_t &= \sigma(W_o * X_t + S_o * H_{t-1}) \\ \hat{C}_t &= tanh(W_{\hat{c}} * X_t + S_{\hat{c}} * H_{t-1}) \\ C_t &= F_t \circ C_{t-1} + I_t \circ \hat{C}_t \\ H_t &= O_t \circ tanh(\hat{C}_t) \end{aligned} \tag{2}$$

Where $*$ denotes the convolution operation and $\circ$ denotes the Hadamard product.

B. Robust Spatiotemporal ConvLSTM proposed algorithm

The Proposed Robust Spatiotemporal ConvLSTM (Robust-ST-ConvLSTM) algorithm is a memory flow algorithm based on higher order ConvLSTM. To make it simple, the novel algorithm aim to decide the cell state C_t not only from the previous hidden state H_{t-1} but also from N previous hidden states ($H_{t-2}, ..., H_{t-N}$) (N will be fixed by the user and it can only affect the computational time). The second part of the algorithm is to implement a memory flow to hold spatiotemporal information to optimize and control the prediction capacities of ConvLSTM. In fact, the memory flow will be a second cell state for spatiotemporal data. However, the cell state will not be removed and will handle temporal

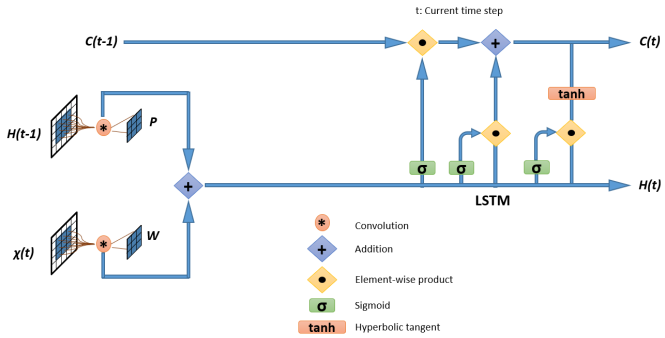

Fig. 2. The structure of convolutional LSTM

data.

Indeed, the novel algorithm uses a stack of ConvLSTM units to learn the spatial correlations and the temporal dynamics from the input video. These feature will be used later to forecast the future frames. So, a novel transition function is introduced based on spatiotempral memory flow and is able to leverage a deterministic number of previous hidden states. In the original implementation of ConvLSTM, the temporal memory states C_t are updated only from one time step to another. However, in video prediction, the consecutive frames are having a close data distributions in the spatial dimensions and also many temporal correlations. That's why, we need to exploit these properties to make better predictions in terms of quality performances. Therefore, we believe that this higher order ConvLSTM based on memory flow will take advantage from the global motion changes of the consecutive frames and the information of the spatiotemporal memory to predict future frames. It is obvious that the memory state is updated horizontally in the original stacked ConvLSTM model. However, the previous model is enhanced by updating the memory state horizontally (the cell state) and also vertically (the spatiotemporal memory state) as illustrated in Figure 3. This approach ameliorates the way we handle the spatiotemporal information from the input to the output and enables to connect all the recurrent units of the entire network. The main equations of the new robust spatiotemporal unit represented in Figure 4 are:

$$
\begin{aligned}
I_t &= \sigma(W_i * X_t + f(H^l_{t-1}, ..., H^l_{t-N})) \\
F_t &= \sigma(W_f * X_t + f(H^l_{t-1}, ..., H^l_{t-N})) \\
\hat{C}_t &= tanh(W_{\hat{c}} * X_t + f(H^l_{t-1}, ..., H^l_{t-N})) \\
C^l_t &= F_t \circ C^l_{t-1} + I_t \circ \hat{C}_t \\
I'_t &= \sigma(W'_i * X_t + M'_i * STM^{l-1}_t) \\
F'_t &= \sigma(W'_f * X_t + M'_f * STM^{l-1}_t) \\
\hat{C}'_t &= tanh(W'_{\hat{c}} * X_t + M'_{\hat{c}} * STM^{l-1}_t) \\
STM^l_t &= F'_t \circ STM^{l-1}_t + I'_t \circ \hat{C}'_t \\
O_t &= \sigma(W_{ox} * X_t + f(H^l_{t-1}, ..., H^l_{t-N}) \\
&\quad + W_{oc} * C^l_t + W_{ostm} * STM^l_t) \\
H^l_t &= O_t * tanh(W_{1\times1} * [C^l_t, STM^l_t])
\end{aligned}
\tag{3}
$$

Where σ is the sigmoid activation function, $*$ and $\circ$ denote the convolution operator and the Hadamard product respectively. Like the original ConvLSTM, I_t and I_te: the input gates, F_t and F'_t: the forget gates, $\hat{C}_t$ and $\hat{C}'_t$: the potential candidates for the cell states,O_t: the output gate. X_t denotes the input at the time step t. H^l_t denotes the hidden state of the lth layer at the time step t. C^l_t is the memory state of the lth layer at the time step t. STM^l_t denotes the spatiotemporal memory of the lth layer at the time step t. f is the function that should be designed to combine N previous hidden states. The design of the f is quite difficult since it must satisfy the following conditions: the spatial structure of the hidden states must be preserved, the size of the filters that control the previous hidden states must increase with the time steps in order to capture the context of these structures and finally, the algorithm's complexity must not explode.

To implement f we tested two main approaches. The first approach aims to return the mean value of all elements in the input tensor that handle the previous hidden states. In Robust-ST-ConvLSTM, the feedback signal is generated by combining multiple preceding hidden states. Therefore, the state of the N-order Robust-ST-ConvLSTM is recursively updated with the following f function:

$$
f(H^l_{t-1}, ..., H^l_{t-N}) = \frac{1}{N} \sum_{n=1}^{N} W_{hn} H^l_{t-n} \tag{4}
$$

Analogous to the filter structures used in signal processing, the second approach in designing the f function in inspired from recursive least squares filters [35]. It is now based on the weighted sum of the previous hidden states. Consequently, f is straightforward:

$$
f(H^l_{t-1}, ..., H^l_{t-N}) = \frac{1}{N} \sum_{n=1}^{N} \alpha^n_n W_{hn} H^l_{t-n} \tag{5}
$$

Where α is the forgetting factor. The parameter α $(0 < \alpha < 1)$ gives more weight to recent hidden states.

The gates of the Robust Spatiotemporal unit are no longer dependent on the the hidden state and the temporal memory state from the previous time step of the same layer. However, they depend on the previous hidden states from previous time

steps at the same layer and the spatiotemporal memory state. To be clear, the first layer in a stacked ConvLSTM model at time step t receives the spatiotempral memory of the last layer in the stacked model of the previous time step as illustrated in Figure 3 ($STM_t^1 = STM_{t-1}^L$ with L is the number of stacked layers).

So, we adopt the original structure of ConvLSTM and we added a second gated structure for the spatiotemporal memory STM_t^l. However, the final hidden state H_t^l depends on the combination of the temporal memory state C_t^l and the spatiotemporal memory state STM_t^l.

The spatiotemporal memory parameter is dedicated to reduce the loss of spatiotempral information in the video sequences from the first layer to the last layer of the network. Besides, the previous hidden states used as input for the ConvLSTM blocks are implemented to expand the visibility of the neural units about the context of the current events at different time steps.

It is clear that the proposed model increases the number of parameters when we compare it with the standard ConvLSTM but it will prevent as from unnecessarily expanding the ConvLSTM model to obtain the same performances.

IV. Performance evaluation, comparison and discussion

A. Datasets

As far as we are concerned, there are currently no datasets for video prediction because it is an emerging area of research. However, researchers basically use motion video datasets such as KTH and MovingMNIST used to compare the performances of our proposed algorithm with the state-of-the-art approaches.

1) KTH

This dataset [5] has 2391 video sequences of 6 human actions (Walking , Jogging, Running, Boxing, Hand waving, Hand clapping) performed by 25 people in 4 different scenarios (outdoors s1, outdoors with scale variation s2, outdoors with different clothes s3 and indoors s4 as illustrated in Figure 5).

Static cameras were used to capture the video scenes with 25fps as a frame rate. The sequences have a length of 4 seconds in average with a frame size of 160×120. The videos are stored in 600 video files for each combination of 25 subjects, 6 actions and 4 scenarios.

2) Moving MNIST

The commonly used Moving MNIST dataset [6] contains 10,000 sequences of moving handwritten digits with random velocities. Each video of the dataset is a 20-frame dynamic scene and its frame size is 64×64.

B. Compared methods and performance metrics

1) Compared methods

To evaluate the performance of our proposed Robust-ST-ConvLSTM, we compare it with the performance of some advanced video prediction models:

- **ConvLSTM:** is commonly used for spatiotemporal predictive systems with a traditional roadway for the memory state. This algorithm is mentioned in almost every research work as the least efficient approach. However, it is the source of inspiration for video prediction algorithms based on recurrent neural networks.
- **PredRNN 2017:** based on the spatiotemporal LSTM (ST-LSTM) unit that take out and memorize spatial appearances and temporal variations simultaneously.
- **PredRNN 2021:** In this algorithm, a pair of memory cells is used and explicitly decoupled in order to enhance the performances of the previous algorithm and surpass its limitations. In addition, another learning strategy was proposed called reverse scheduled sampling.

2) Performance metrics

Because the results are video frames, we will use the most commonly used metrics to evaluate the quality of images between the ground truth and the prediction. Those metrics are Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) [30]. The PSNR measures, in decibels, the quality ratio between the original frame and the predicted one. The higher the PSNR, the better the quality of the predicted image.

The PSNR is calculated by:

$$PSNR(Y,\hat{Y}) = 10log_{10}\frac{max_{\hat{Y}}^2}{\frac{1}{N}\sum_{i=1}^{N}(Y-\hat{Y})^2} \tag{6}$$

Where Y is the ground truth, $\hat{Y}$ is the generated prediction, N is the number of pixels and $max_{\hat{Y}}$ is the maximum value of the frame intensities.

The SSIM measures the similarity between two images in terms of luminance, contrast and structure. It is calculated as follows:

$$SSIM(Y,\hat{Y}) = \frac{(2\mu_Y\mu_{\hat{Y}} + C_1) + (2\sigma_{Y\hat{Y}} + C_2)}{(\mu_Y^2 + \mu_{\hat{Y}}^2 + C_1)(\sigma_Y^2 + \sigma_{\hat{Y}}^2 + C_2)} \tag{7}$$

Where μ_Y and $\mu_{\hat{Y}}$ are the average of Y and $\hat{Y}$, respectively, σ_Y and $\sigma_{\hat{Y}}$ are the variance of Y and $\hat{Y}$, respectively, $\sigma_{Y\hat{Y}}$ is the covariance of Y and $\hat{Y}$. C_1 and C_2 are constants. The higher the SSIM, the greater similarity between two images.

C. Implementation details

The proposed algorithm is implemented with Python 3.6 and Pytorch 1.4.0 as a deep learning framework. Pytorch is used because it offers an effective way to manipulate tensors or multi-dimensional matrices needed to store and process multi-dimensional data.

We use Adam optimizer to train our model which is an optimization algorithm that combine the properties of AdaGrad and RMSProp algorithms to provide an optimization algorithm that is faster than the commonly used Stochastic Gradient Descent (SGD) algorithm especially with sparse data. A mini-batch of 2 sequences is chosen at each training iteration and it is reduced to the maximum to handle the out of memory problem of our GPU. We choose a learning rate of 0.0001

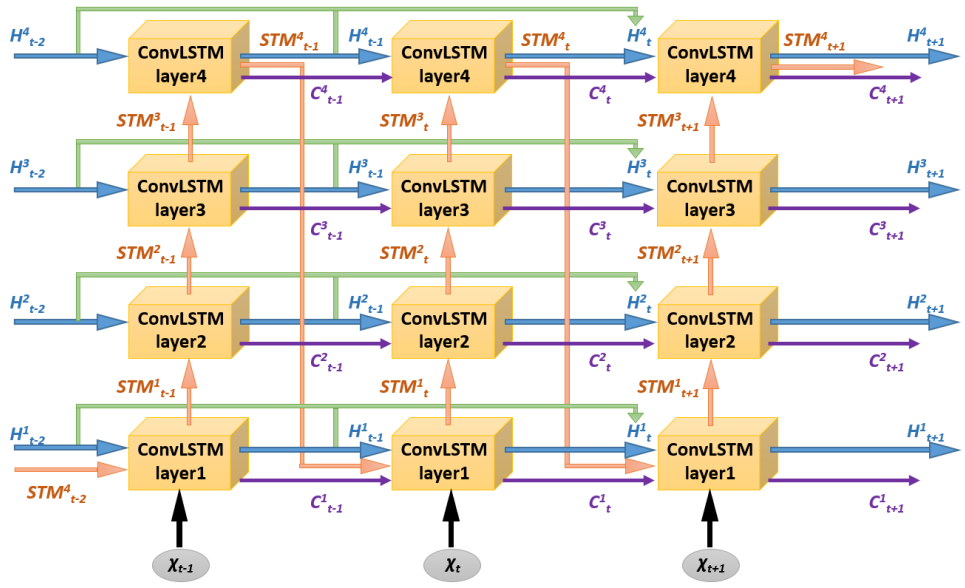

Fig. 3. The main structure of Robust Spatiotempral LSTM

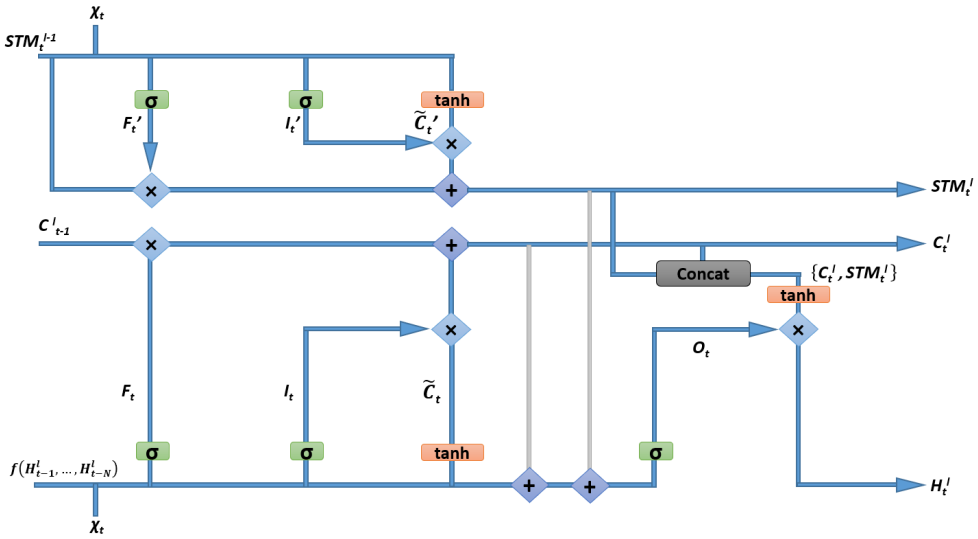

Fig. 4. Robust Spatiotemporal Unit

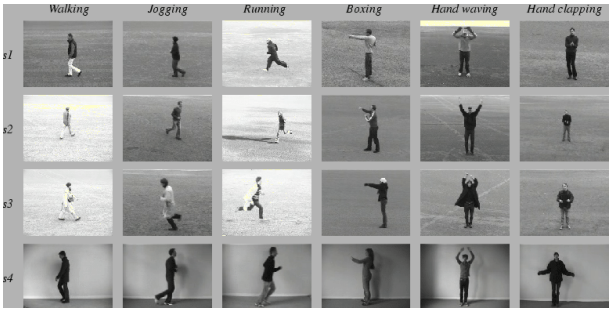

Fig. 5. KTH action dataset

and the training process is stopped after 100000 iterations. The main architecture of our proposed model is composed of 4 ConvLSTM layers for each time step as illustrated in Figure 3. We set the number of the hidden states used to strengthen the prediction process to 3. The initial tensor used to handle the hidden states tensors is filled with the scalar value 0 with the shape: [3, dimension of one hidden state]. The dimensions of the hidden state depend on the dimensions of the input frames.

The entire training process was on an NVIDIA GeForce RTX 2060GPU, Intel(R) Core(TM) i7-9700K CPU (3.60 GHz), a 32GB device memory, and Windows 10 operating system.

D. Simulation results

1) on KTH dataset

Table I presents quantitative results of the proposed algorithm and state-of-the-art networks and the corresponding frame-wise comparisons are shown in Figure 6 and Figure 7. We adopt PSNR and SSIM as evaluation metrics. We can obviously confirm that our proposed algorithm show significant improvements in terms of short-term and long-term forecasting over the commonly used ConvLSTM approach. In fact, it increases the average PSNR and SSIM over the same number of predicted frames by 26% and 21.31%, respectively, by comparing it with the algorithm mentioned above. Also, it performs favorably against the PredRNN-v2017 and the PredRNN-v2021 algorithms of Wang et al. Our Robust-ST-ConvLSTM (with $\alpha = 0.9$) performs better than PredRNN-v2021 by 1.72% and 2.77% in terms of PSNR and SSIM, respectively. These empirical results demonstrate the effectiveness and the efficiency of the Robust Spatiotemporal Convolutional LongShort-Term Memory algorithm in predicting future frames. In accordance with these results, Figure 8 that compares representative generated frames, proves that our algorithm outperforms the state-of-the-art approaches in terms of future movement and frames details. Robust ST-ConvLSTM predicts more accurate motion trajectories into the future because of the memory flow component that strengthen the long term prediction ability of the ConvLSTM cell and also because of updating the ConvLSTM cell using information from some previous time steps.

we can notice also that the second approach in designing f which is inspired from recursive least squares filters slightly outperforms the first approach in terms of PSNR and SSIM. This means that further research work could be done in order to determine the optimal value of α that gives the best PSNR and SSIM performances. In this work, various values of α have been tested randomly ($0 < \alpha < 1$) and the optimum one among them was the selected value 0.9.

The presented results and the computational cost depend on the number of memory units used for feedback. In our implementation, we used 3 hidden states which means that we have 3r-order Robust-ST-ConvLSTM. Furthermore, the number of hidden states can affect the performances of our model in terms of the quality of its output and also in terms of the computational process.

From the previous observations about the value of α and the number of hidden states, we can confirm that a trade-off should be done between quality performances and computational costs, in future research work, to have the best performances without training a computationally very expensive algorithm.

2) on Moving MNIST

Table II presents the performance of the evaluated models on the Moving MNIST dataset by predicting the next 10 frames from the previous 10 input frames. We use the similarity index measure (SSIM) and the Peak signal-to-noise ratio (PSNR) for evaluation. As shown from table II and Figures 9 and 10, our architecture performs well against the state-of-the-art approaches in both metrics. Figure 9 and Figure 10 show the frame-wise PSNR and SSIM comparisons of different approaches on MNIST dataset. The results of these figures prove the ability of the Robust-ST-ConvLSTM in predicting future frames. Also, they prove that our approach outperforms the previous models on all the predicted frames. The memory flow algorithm based on 3rd order ConvLSTM with $\alpha = 0.9$ increases the average PSNR over the 10 predicted frames by 3.15% by comparing it with PredRNN (Wang et al., 2021). However, it outperforms the same approach by 0.22% in terms of SSIM. This means that this metric could not be a good evaluation metric in this case. Moreover, Our approach performs favorably against the traditional ConvLSTM approach in terms of PSNR and SSIM. It brings 14.59% PSNR improvement and 26.95% SSIM improvement over ConvLSTM based frames prediction approach. These numerical results are confirmed by Figure 11 that shows the quality of the 10 predicted frames generated by the different approaches. Robust-ST-ConvLSTM outputs clearer frames. However, the state-of-the-art algorithms produce blurry images. This means that Robust-ST-ConvLSTM is more precise and sure about the future variations which proves its robustness against the other long-term prediction algorithms mentioned above.

we can notice also that the recursive least squares filters based approach in designing f has approximately similar results as the first approach and that for different values of α. Different from KTH dataset, the value of the parameter α does not

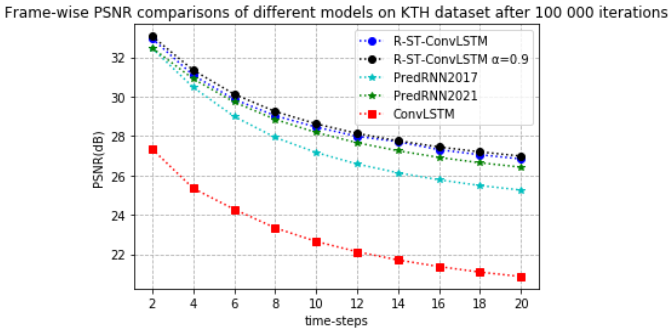

Fig. 6. Frame-wise PSNR comparisons of different models on KTH dataset after 100 000 iterations

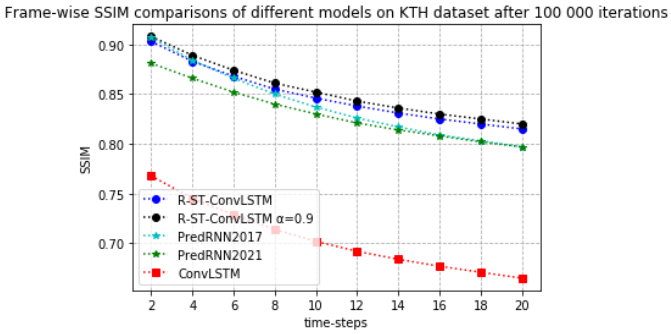

Fig. 7. Frame-wise SSIM comparisons of different models on KTH dataset after 100 000 iterations

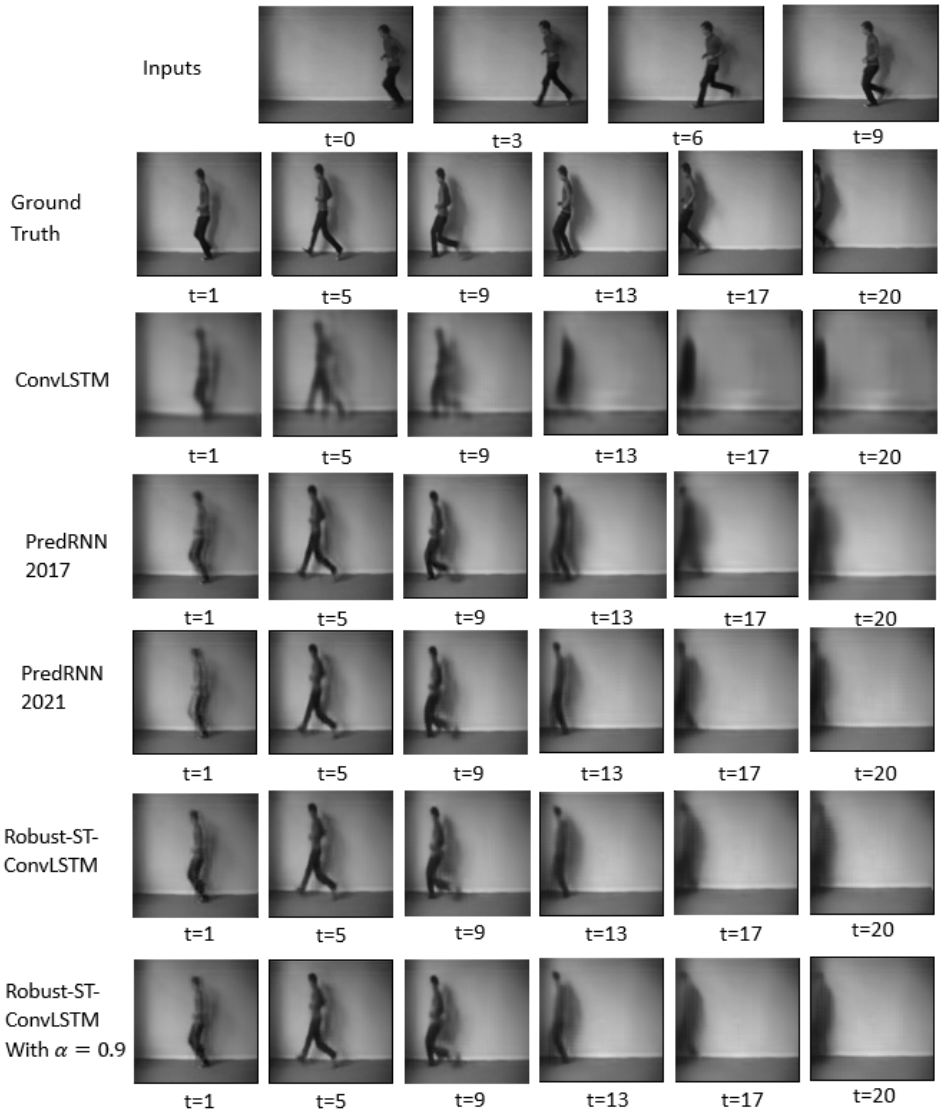

Fig. 8. Prediction examples on the KTH data set, where we predict 20 frames into the future based on the past 10 frames

TABLE I
QUANTITATIVE EVALUATION OF DIFFERENT ALGORITHMS ON THE KTH DATASET. THE METRICS ARE AVERAGED OVER THE 20 PREDICTED FRAMES. HIGHER SCORES INDICATE BETTER PREDICTION RESULTS

Model	PSNR(dB)	SSIM
ConvLSTM (Shi et al., 2015)	23.009	0.704
PredRNN (Wang et al., 2017)	27.624	0.839
PredRNN (Wang et al., 2021)	28.502	0.831
Robust-ST-ConvLSTM	**28.828**	**0.848**
Robust-ST-ConvLSTM with $\alpha = 0.9$	**28.992**	**0.854**

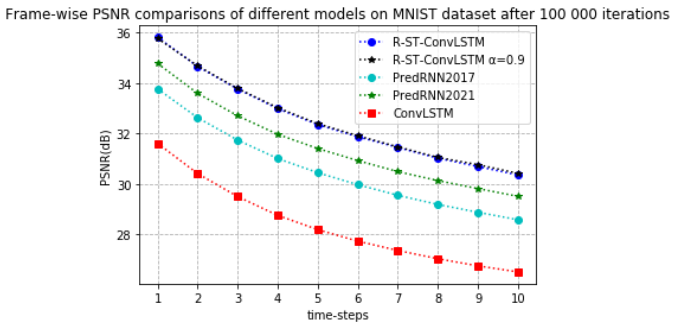

Fig. 9. Frame-wise PSNR comparisons of different models on MNIST dataset after 100 000 iterations

affect the quality performances of the outputs but it affects the computational cost of our algorithm since a number of multiplications are added to the calculation process. This means that, for MNIST dataset, only the first approach of designing f, which is based on returning the mean value of the previous hidden states, is taken into consideration.

V. CONCLUSION

Video prediction is considered as a powerful tool to understand and model dynamic scenes. Therefore, in this work,

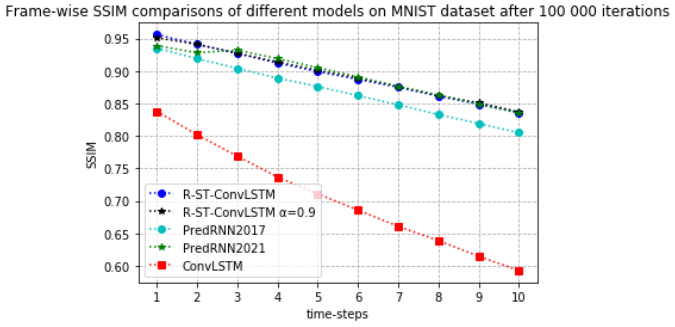

Fig. 10. Frame-wise SSIM comparisons of different models on MNIST dataset after 100 000 iterations

TABLE II
QUANTITATIVE EVALUATION OF DIFFERENT ALGORITHMS ON THE MNIST DATASET. THE METRICS ARE AVERAGED OVER THE 10 PREDICTED FRAMES. HIGHER SCORES INDICATE BETTER PREDICTION RESULTS

Model	PSNR(dB)	SSIM
ConvLSTM (Shi et al., 2015)	28.380	0.705
PredRNN (Wang et al., 2017)	30.569	0.869
PredRNN (Wang et al., 2021)	31.525	0.893
Robust-ST-ConvLSTM	**32.490**	**0.894**
Robust-ST-ConvLSTM with $\alpha = 0.9$	**32.520**	**0.895**

we propose a new recurrent neural network (Robust-ST-ConvLSTM) for video prediction. It is based on new robust spatiotemporal unit inspired from the well-known ConvLSTM structure. This spatiotemporal unit rely on two different approaches in order to strengthen its prediction abilities: a memory flow to handle the spatiotemporal information and a higher order ConvLSTM approach that enable the cell states to decide their values from previous hidden states. Our approach outperforms the state-of-the-art research works on different datasets, including KTH dataset for human motion and Moving MNIST.

In conclusion, video prediction is a promising research direction and can be used in different applications such as video surveillance, video compression and intelligent decision-making systems. While great work has been done in video prediction, there is still a place for improvement especially with the continuous advancement in deep learning techniques.

ACKNOWLEDGMENTS

This work was supported in part by the sensors generation project of Nouvelle Aquitaine region (2018-1R50214).

REFERENCES

[1] J. Walker, A. Gupta, and M. Hebert, "Dense optical flow prediction from a static image," in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Dec. 2015, pp. 2443–2451.

[2] J. Walker, C. Doersch, A. Gupta, and M. Hebert, "An uncertain future: Forecasting from static images using variational autoencoders," in Proc. Eur. Conf. Comput. Vis. (ECCV), 2016, pp. 835–851.

[3] TC. Wang, MY. Liu, JY. Zhu, G. Liu, A. Tao, and J. Kautz, "Video-to-video synthesis," in Proc. Neural Inf. Process. Syst. (NIPS), 2018, pp. 1144–1156.

[4] N. Sedaghat, "Hybrid Learning of Optical Flow and Next Frame Prediction to Boost Optical Flow in the Wild," 2016, arXiv:1612.03777. Accessed: Apr. 7, 2017. [Online]. Available: https://arxiv.org/abs/1612.03777

[5] Christian Schuldt, Ivan Laptev, and Barbara Caputo, "Recognizing human actions: a local svm approach," in ICPR, 2004, pp. 32–36 Vol.3.

[6] N. Srivastava, E. Mansimov, and R. Salakhutdinov, "Unsupervised Learning of Video Representations using LSTMs," in ICML, 2015.

[7] A. Terwilliger, G. Brazil, and X. Liu, "Recurrent flow-guided semantic forecasting," in WACV, 2019.

[8] X. Shi, Z. Chen, H. Wang, D. Yeung, W. Wong, and W. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in NeurIPS, 2015.

[9] W. Lotter, G. Kreiman, and D. D. Cox, "Unsupervised learning of visual structure using predictive generative networks," arXiv:1511.06380, 2015.

[10] N. Wichers, R. Villegas, D. Erhan, and H. Lee, "Hierarchical long-term video prediction without supervision," in ICML, ser. Proceedings of Machine Learning Research, vol. 80, 2018.

[11] R. Villegas, J. Yang, Y. Zou, S. Sohn, X. Lin, and H. Lee, "Learning to generate long-term future via hierarchical prediction," in ICML, 2017.

[12] J. Zhang, Y. Wang, M. Long, W. Jianmin, and P. S. Yu, "Z-order recurrent neural networks for video prediction," in ICME, July 2019.

[13] 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:1412.6604, 2014.

[14] N. Srivastava, E. Mansimov, and R. Salakhutdinov, "Unsupervised Learning of Video Representations using LSTMs," in ICML, 2015.

[15] W. Lotter, G. Kreiman, and D. Cox, "Deep Predictive Coding Networks for Video Prediction and Unsupervised Learning," in ICLR (Poster), 2017.

[16] W. Byeon, Q. Wang, R. K. Srivastava, and P. Koumoutsakos, "Contextvp: Fully context-aware video prediction," in CVPR (Workshops), 2018.

[17] V. Patraucean, A. Handa, and R. Cipolla, "Spatio-temporal video autoencoder with differentiable memory," (ICLR) Workshop, 2015.

[18] C. Lu, M. Hirsch, and B. Scholkopf, "Flexible Spatio-Temporal ¨ Networks for Video Prediction," in CVPR, 2017.

[19] E. L. Denton and V. Birodkar, "Unsupervised learning of disentangled representations from video," in NeurIPS, 2017.

[20] J. Oh, X. Guo, H. Lee, R. L. Lewis, and S. P. Singh, "ActionConditional Video Prediction using Deep Networks in Atari Games," in NeurIPS, 2015.

[21] E. Denton and R. Fergus, "Stochastic video generation with a learned prior," in ICML, ser. Proceedings of Machine Learning Research, J. G. Dy and A. Krause, Eds., vol. 80, 2018.

[22] S. shahabeddin Nabavi, M. Rochan, and Y. Wang, "Future Semantic Segmentation with Convolutional LSTM," in BMVC, 2018.

[23] S. Vora, R. Mahjourian, S. Pirk, and A. Angelova, "Future segmentation using 3d structure," arXiv:1811.11358, 2018.

[24] Y. Wang, M. Long, J. Wang, Z. Gao, and S. Y. Philip, "PredRNN: Recurrent neural networks for predictive learning using spatiotemporal lstms," in NeurIPS, 2017, pp. 879–888

[25] J. Yan, G. Qin, R. Zhao, Y. Liang and Q. Xu, "Mixpred: Video Prediction Beyond Optical Flow," in IEEE Access, vol. 7, pp. 185654-185665, 2019, doi: 10.1109/ACCESS.2019.2961383.

[26] Z. Liu, R. A. Yeh, X. Tang, and Y. Liu, "Video frame synthesis using deep voxel Flow," in Proc. IEEE Int. Conf. Comput. Vis. (CVPR), Oct. 2017, pp. 44634471.

[27] Y.-H. Kwon and M.-G. Park, "Predicting future frames using retrospective cycle gan," in CVPR, 2019.

[28] Sergiu Oprea, Pablo Martinez-Gonzalez, Alberto Garcia-Garcia, John Alejandro Castro-Vargas, Sergio Orts-Escolano, Jose Garcia-Rodriguez, and Antonis Argyros. A review on deep learning techniques for video prediction. arXiv preprint arXiv:2004.05214, 2020

[29] S. Tulyakov, M.-Y. Liu, X. Yang, and J. Kautz, "MoCoGAN: Decomposing motion and content for video generation," in CVPR, June 2018

[30] A. Horé and D. Ziou, "Image Quality Metrics: PSNR vs. SSIM," 2010 20th International Conference on Pattern Recognition, Istanbul, 2010, pp. 2366-2369, doi: 10.1109/ICPR.2010.579.

[31] Yunbo Wang, Haixu Wu, Jianjin Zhang, Zhifeng Gao, Jianmin Wang, Philip S. Yu, Mingsheng Long, "PredRNN: A Recurrent Neural Network for Spatiotemporal Predictive Learning". arXiv:2103.09504, 2021

Fig. 11. Prediction examples on the MNIST data set, where we predict 10 frames into the future based on the past 10 frames

[32] S. Xingjian et al., "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in Proc. Adv. Neural Inf. Process. Syst., 2015, pp. 802–810.

[33] R. Soltani and H. Jiang, "Higher order recurrent neural networks," arXiv preprint arXiv:1605.00064, 2016.

[34] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735– 1780, 1997.

[35] J. Cances and V. Meghdadi, "Joint channel estimation and data demodulation algorithms for fast time varying band limited frequency selective Rayleigh fading channels: A comparison study," Annales Des Telecommun., vol. 55, no. 56, pp. 226–237, May/Jun. 2000.