Received 11 January 2024, accepted 2 March 2024, date of publication 14 March 2024, date of current version 20 March 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3375365

RESEARCH ARTICLE

A Hybrid Transformer-LSTM Model With 3D Separable Convolution for Video Prediction

MAREETA MATHAI, (Graduate Student Member, IEEE), YING LIU, (Member, IEEE), AND NAM LING, (Life Fellow, IEEE)
Department of Computer Science and Engineering, Santa Clara University, Santa Clara, CA 95053, USA

Corresponding author: Mareeta Mathai (mmathai@scu.edu)

This work was supported in part by the National Science Foundation under Grant ECCS-2138635.

ABSTRACT Video prediction is an essential vision task due to its wide applications in real-world scenarios. However, it is indeed challenging due to the inherent uncertainty and complex spatiotemporal dynamics of video content. Several state-of-the-art deep learning methods have achieved superior video prediction accuracy at the expense of huge computational cost. Hence, they are not suitable for devices with limitations in memory and computational resource. In the light of Green Artificial Intelligence (AI), more environment friendly deep learning solutions are desired to tackle the problem of large models and computational cost. In this work, we propose a novel video prediction network 3DTransLSTM, which adopts a hybrid transformer-long short-term memory (LSTM) structure to inherit the merits of both self-attention and recurrence. Three-dimensional (3D) depthwise separable convolutions are used in this hybrid structure to extract spatiotemporal features, meanwhile enhancing model efficiency. We conducted experimental studies on four popular video prediction datasets. Compared to existing methods, our proposed 3DTransLSTM achieved competitive frame prediction accuracy with significantly reduced model size, trainable parameters, and computational complexity. Moreover, we demonstrate the generalization ability of the proposed model by testing the model on dataset completely unseen in the training data.

INDEX TERMS 3D separable convolution, deep learning, depthwise convolution, LSTM, pointwise convolution, self-attention, spatiotemporal modeling, transformer, visual communications, video prediction.

I. INTRODUCTION

With the increasing advent of powerful graphics processing units (GPUs), deep learning is the foremost option of many artificial intelligence (AI) applications and it has been a crucial part in the advancement of many computer vision (CV) algorithms. In this work, we focus on the task of video frame prediction. A video frame prediction model generates future frames from past frames, by learning the complex spatiotemporal content and dynamics of the video data. It finds a wide range of real-world applications such as video coding [1], autonomous vehicles [2] and anomaly detection [3].

Many existing video prediction models are based on 2-dimensional convolutional neural networks (2D CNNs) [4], [5], [6], [7], [8], 3-dimensional (3D) CNNs [9], [10], or recurrent neural networks (RNNs) [11], [12]. While CNNs can extract local features, RNNs are specifically used to learn sequential representations. To benefit from both CNNs and RNNs, other approaches [13], [14], [15], [16], [17], [18], [19], [20] proposed to combine CNN and RNN and learn spatiotemporal features from video data. Among these works, many adopted the long short-term memory (LSTM) as their RNN structure, which led to the family of ConvLSTM models.

Transformers which have primarily demonstrated success in natural language processing (NLP) [21] and several vision tasks [22], [23], [24], [25] were also recently utilized for video prediction [26], [27], [28]. Transformer models are capable of capturing long-range dynamics without the vanishing gradient problem of recurrent networks and have the advantage of parallelism with the self-attention

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

mechanism [21]. However, the accuracy of transformer models usually comes at the price of huge computational cost [29].

Recently, researchers across the NLP and CV communities [30], [31] advocated to shift the focus to Green AI with energy-efficient deep learning solutions, rather than continuously pushing red AI methods to reach state-of-the-art (SOTA) results using massive computational power. Our work focuses on developing an efficient video frame prediction model with reduced model size, fewer parameters, and low computational complexity, while achieving competitive prediction accuracy.

Since both transformer and ConvLSTM models have achieved superior accuracy in predictive learning, in this work, we propose a hybrid transformer-LSTM (3DTransLSTM) model to predict future video frames. To learn spatiotemporal dynamics from video data, the proposed 3DTransLSTM network adopted 3D separable convolutions to extract features along the temporal, height, and width dimensions. The 3D separable convolutions not only offer higher prediction accuracy than 2D convolutions, but also reduce the computational complexity compared to standard 3D convolutions. Our main contributions can be summarized as follows:

- For the first time in the literature, we proposed a hybrid transformer-LSTM (3DTransLSTM) network for the video prediction task. On the one hand, the transformer module can leverage long-range correlations among multiple successive video frames, and parallelize the computation with its self-attention mechanism. On the other hand, the LSTM module can enable spatiotemporal information flow vertically within each time step and horizontally among multiple time steps.
- The proposed 3DTransLSTM adopts 3D convolutions to effectively learn spatiotemporal dynamics. To reduce computational cost, the standard 3D convolution is decomposed into a 3D depthwise convolution and a pointwise convolution, which reduced the model size, trainable parameters, and floating-point operations (FLOPs). To the best of our knowledge, this is the first time that 3D separable convolution is utilized in a hybrid transformer-LSTM network.
- Qualitative and quantitative experimental results on popular video prediction datasets show that, compared to SOTA methods, the proposed model achieves competitive video frame prediction accuracy with significantly smaller model size, fewer model parameters, and less computational cost. Further, we demonstrated the generalization ability of the proposed model by testing the model on video sequences completely unseen in the training dataset.

The remaining of the paper is organized as follows. In Section II, we discuss existing video frame prediction methods and explain the motivation behind our proposed model. In Section III, we provide the preliminaries on 3D depthwise separable convolutions. Section IV elaborates on our proposed 3DTransLSTM model in detail. Section V presents experiments on four video prediction datasets and comparison studies with prior arts. Section VI concludes the paper and discusses future research directions.

II. RELATED WORK

A. VIDEO PREDICTION METHODS

Existing video prediction methods can be broadly classified into four categories: CNN-based methods, RNN-based methods, generative adversarial network (GAN)-based methods, and transformer-based methods.

Many 2D CNN-based video prediction approaches [4], [5], [32], [33] were devised to model spatiotemporal dynamics in video data. In [4], the content encoder and motion encoder focused on the static scene information and the temporal dynamics of consecutive frames, respectively. In [32], a convolutional encoder-decoder network was proposed to explicitly incorporate a time-related input variable to model temporal correlations. Deformable convolutions were used to fuse features from previous frames in [33].

Certain 2D CNN-based frame prediction schemes were proposed for inter-frame prediction [6], [7], [8] in traditional video coding such as the high efficiency video coding (HEVC) [34] and versatial video coding (VVC) [35], or in learning-based video coding. For example, a 2D CNN-based deep network was proposed for both uni-directional and bi-directional frame prediction in HEVC and avoided coding additional motion information [6]. Another CNN-based multi-resolution video prediction network (VPN) [7] utilized two sub-VPN architectures in cascade to generate virtual reference frame from previously coded frames in HEVC. Further, recurrent and bi-directional in-loop prediction modules were proposed in [8] as part of a deep learning-based video compression system.

3D CNN is another way to extract spatiotemporal features. It was used along with optical flow images to predict future frames based on a single image in [9]. Spatially displaced convolution network (SDC-Net) [10] utilized a 3D CNN for video prediction, conditioning on both past frames and past optical flows.

Using CNNs alone can only take into account local structures or short-range dependencies in video data due to the limited size of convolution kernels. To effectively capture long-range correlations in a video sequence, methods based on RNNs [11], [12] were proposed to predict future frames. For example, an LSTM-based encoder-decoder network was developed in [11]. It used an encoder LSTM to map an input video sequence into a fixed length representation, which was then decoded using single or multiple decoder LSTMs to predict future frames. Folded recurrent neural network (FRNN) [12] presented a recurrent auto-encoder with state sharing between the encoder and the decoder. It utilized stacked double-mapping gated recurrent unit (GRU) layers to enable bidirectional information flow between the input and the output.

Although RNNs effectively learn sequential representations, they fail to accurately learn spatial content [13]. To address this issue, convolutions were incorporated into LSTMs to form ConvLSTMs [13], where the internal fully connections in LSTM were replaced by convolution operations. For example, ConvLSTM was utilized in [4] for motion prediction, and was used in [5] to generate appearance features of the next frame from two previous frames. In addition, the stacked ConvLSTM architecture was explored in the dynamic neural advection (DNA) module [14], which predicted the distribution of each pixel in the current frame based on the previous frame. E3D-LSTM [15] integrated 3D convolutions into LSTM to capture short-term frame dependencies and utilized a gate-controlled self-attention module to perceive long-term correlations. The PredRNN [16] network proposed the popular spatiotemporal LSTM (ST-LSTM) structure. It adopted a temporal memory cell and a novel spatiotemporal memory cell to simultaneously memorize spatial and temporal information. Later on, several video prediction methods adopted ST-LSTM as their building blocks [17], [18], [19], [20]. For example, the memory-in-memory (MIM) [17] model improved PredRNN [16] by replacing the simple forget gate in the ST-LSTM block with two cascaded memory transitions, which more effectively captured non-stationary dynamics. CrevNet [18] used a reversible auto-encoder and stacked ST-LSTM blocks for future frame prediction and object detection. PredRNN-V2 [19] improved PredRNN [16] by introducing a memory decoupling loss to ST-LSTM to keep the memory cells from learning redundant features. Other ConvLSTM or convolutional GRU (ConvGRU) approaches such as TrajGRU [36], PredRNN++ [37], Conv-TTLSTM [38], STGRU [39] and ASTM [40] were also developed for the video prediction task.

Due to the mean-squared error (MSE) loss adopted in model training, CNN-based video prediction models tend to generate blurry predicted frames which are inconsistent with human perception. To overcome this limitation, GAN-based models adopt adversarial training such that the predicted frames are sharper and present more details than pure CNN-based methods. BeyondMSE [41] was the pioneer in applying adversarial training for video prediction. It used a multi-scale architecture and an image gradient difference loss along with the MSE loss. Dual-Motion GAN (DM-GAN) [42] used a dual adversarial training mechanism with two pairs of generator and discriminator to generate future frames and future flows simultaneously. CycleGAN [43] adopted a forward-backward prediction scheme by training a generator to produce both future frames and past frames. Attention-based inter-frame prediction method in [44] enhanced coding efficiency of VVC by incorporating GAN-based deep attention map estimation and deep frame interpolation methods.

In recent years, transformers have been developed for NLP and CV tasks. Compared with RNN-based methods, the transformer architecture can extract long-term dependencies more efficiently and get rid of the limitation of seriality. In particular, a few approaches combined transformers and CNNs for video frame prediction. For instance, ConvTransformer [26] used an end-to-end encoder-decoder transformer architecture for video interpolation and extrapolation tasks. It proposed multi-head convolutional self-attention layers with 2D convolutions in both the encoder and decoder. The temporal convolutional transformer network (TCTN) [27] used a transformer-based encoder for video prediction, where 3D convolutional layers were employed to extract short-term dependencies and masked self-attention layers were used to capture long-term dependencies. The video prediction transformer (VPTR) [28] proposed to separately perform spatial attention and temporal attention. First, spatial attention was performed locally on each feature patch using multi-head self-attention (MHSA), followed by a 2D separable convolution-based feed-forward neural network. Afterwards, a temporal MHSA was adopted to model the temporal dependency between frames.

B. MOTIVATION OF THE PROPOSED METHOD

Although the aforementioned SOTA methods achieved accurate video prediction results, their accuracy comes at a price of big model size, large amount of model parameters and heavy computational complexity. For example, the transformer-based models TCTN [27] and VPTR [28] have large model size and FLOPs due to standard 3D convolutions and complicated attention mechanisms, respectively. The ConvLSTM-based models E3D-LSTM [15] and CrevNet [18] adopted standard 3D convolutions too. MIM [17] also has relatively larger model size and FLOPs since it adopted additional memory modules inside the original ST-LSTM blocks.

In this work, we aim at developing a lightweight video prediction network which still offers competitive frame prediction accuracy. To achieve this goal, we design a hybrid architecture that benefits from both transformer and LSTM structures. This is the first time in the literature that such hybrid structure is proposed for video frame prediction. Besides, while existing video prediction networks adopt 2D convolutions [4], [5], [6], [8], [19], [26], [28], [32], [33] or standard 3D convolutions [15], [27], our proposed network adopts the idea of 3D separable convolution. Separable convolution was first conceived in MobileNet [45] to develop lightweight models suitable for mobile and embedded devices. Later on, 2D separable CNN was utilized for faster video segmentation [46], moving object detection [47], and violence detection [48]. To alleviate the computation burden of standard 3D convolution in deep networks, 3D separable CNN was proposed for dynamic hand gesture recognition and video moving object segmentation [49], [50], [51]. In our preliminary work [20], 3D separable convolution-based ST-LSTM was developed with a reversible architecture for video frame prediction. In this work, we propose to use 3D separable convolutions in a hybrid transformer-LSTM network. This not only leverages spatiotemporal correlations, but also effectively reduces model size, trainable parameters, and computational cost.

III. PRELIMINARIES

In this section, we contrast the 2D, 2D separable, 3D and 3D separable convolutions and explain the rationale behind the usage of depthwise separable convolutions.

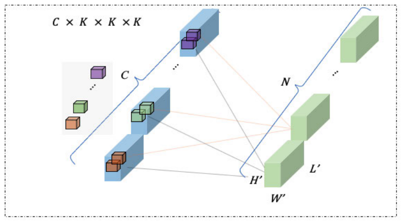
A. 2D CONVOLUTION VERSUS 2D SEPARABLE CONVOLUTION

For an RGB image of size $C \times H \times W$ where C is the number of channels, H is the height and W is the width of the image, 2D convolutions use N 3D filters of size $C \times K \times K$ (channel $\times$ height $\times$ width) to convolve the image in the height and width directions to produce feature maps of size $N \times \check{H} \times \check{W}$. To reduce its computational complexity, a 2D convolution can be separated into a 2D depthwise convolution and a 1D pointwise convolution. In a 2D depthwise convolution, filters of size $1 \times K \times K$ (channel $\times$ height $\times$ width) convolve with each input channel to produce an intermediate output $C \times \check{H} \times \check{W}$. Afterwards, 1D pointwise convolutions convolve the intermediate output in the channel direction with N filters of size $C \times 1 \times 1$ (channel $\times$ height $\times$ width), to generate the final output feature maps of size $N \times \check{H} \times \check{W}$.

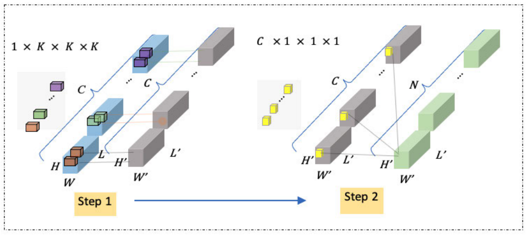
B. 3D CONVOLUTION VERSUS 3D SEPARABLE CONVOLUTION

Video prediction needs a model to learn spatiotemporal information abundantly and deeply. To achieve this goal, prior arts adopted standard 3D convolutions [15], [18], [27]. Let $\mathbf{X}_{in} \in \mathbb{R}^{C \times L \times H \times W}$ be a 4D video tensor, where C, L, H, and W represent the channel, number of successive video frames, height, and width. As illustrated in Fig. 1 (a), a standard 3D convolution uses a 4D filter of size $C \times K \times K \times K$ (channel $\times$ time $\times$ height $\times$ width) which moves in three directions (time, height, width) to generate a 3D output tensor of size $L' \times H' \times W'$. N such filters would create the final output tensor $\mathbf{X}_{out} \in \mathbb{R}^{N \times L' \times H' \times W'}$. The number of floating-point multiplications involved in such a standard 3D convolution is $C \times K \times K \times K \times N \times L' \times H' \times W'$.

Although standard 3D convolution is amply used for video prediction, it has made the network architectures much complex and model sizes bigger. To reduce the computational complexity, a standard 3D convolution can be separated into a 3D depthwise convolution and a pointwise convolution, which are combinedly termed as 3D depthwise separable convolution. As shown in Fig. 1 (b) Step 1, the depthwise convolution applies filters of size $1 \times K \times K \times K$ to each of the C input channels to produce an intermediate output of size $C \times L' \times H' \times W'$. This process involves $C \times K \times K \times K \times L' \times H' \times W'$ multiplications. The intermediate output goes through the pointwise convolution described in Fig. 1 (b) Step 2, where filters of size $C \times 1 \times 1 \times 1$ are applied along the channel direction to produce an output of size $1 \times L' \times H' \times W'$. The final 4D output tensor of size



(a)



(b)

FIGURE 1. (a) The standard 3D convolution, and (b) the 3D depthwise separable convolution.

$N \times L' \times H' \times W'$ is generated by applying N such pointwise filters. Such a pointwise convolution involves $C \times N \times L' \times H' \times W'$ multiplications. To compare the computational cost of the 3D depthwise separable convolution with the standard 3D convolution, we compute the ratio of the number of multiplications involved in these two types of convolutions as

$$\begin{aligned}&\frac{\text{3D depthwise separable convolution}}{\text{standard 3D convolution}}\\&= \frac{\begin{array}{c}C \times K \times K \times K \times L' \times H' \times W' + \\ C \times N \times L' \times H' \times W'\end{array}}{C \times K \times K \times K \times N \times L' \times H' \times W'}\\&= \frac{1}{N} + \frac{1}{K^3}\end{aligned} \tag{1}$$

Therefore, the decomposed convolution can reduce the computational cost of the standard 3D convolution by $\frac{1}{N} + \frac{1}{K^3}$ where N is the number of output channels and K is the filter dimension in time, height, and width.

IV. PROPOSED METHOD

In this section, we elaborate our algorithm in detail. Fig. 2 shows the overall framework of the proposed architecture for three time steps $t-1, t, t+1$. The proposed video frame prediction network enables temporal information flow indicated by the four arrows connecting adjacent time steps.

A. PROBLEM STATEMENT

Consider the video prediction process at time step t. The network takes a 4D tensor $\mathbf{X}_t \in \mathbb{R}^{C \times L \times H \times W}$ as the input, which represents L successive video frames with frame indices $t, t+1, \cdots, t+L-1$, where C, H and W represent the channel, height and width of each frame. The problem of video prediction is to predict the L frames $\widehat{\mathbf{X}}_{t+1} \in \mathbb{R}^{C \times L \times H \times W}$ with frame indices $t+1, t+2, \cdots, t+L$ given

$\mathbf{X}_t$ and can be formulated as

$$\widehat{\mathbf{X}}_{t+1} = Net(\mathbf{X}_t|\theta), \tag{2}$$

where $Net(\cdot)$ represents the frame prediction network, and θ is the collection of trainable model parameters.

B. ALGORITHM OVERVIEW

As shown in the middle column of Fig. 2, at time step t, each frame in the 4D video tensor $\mathbf{X}_t \in \mathbb{R}^{C\times L\times H\times W}$ is spatially split into $p \times p$ patches, forming a tensor $\mathbf{P}_t \in \mathbb{R}^{Cp^2\times L\times \frac{H}{p}\times\frac{W}{p}}$, where $\frac{H}{p} \times \frac{W}{p}$ is the resulting number of patches, and Cp^2 is the length of each flattened patch. Next, $\mathbf{P}_t$ is processed by spatial embedding and positional encoding to generate an output tensor $\mathbf{Z}_t \in \mathbb{R}^{d_{\text{model}}\times L\times \frac{H}{p}\times\frac{W}{p}}$, where d_{model} is the embedded dimension. $\mathbf{Z}_t$ is then passed through six 3DTransLSTM blocks, which leverage transformers and LSTM to extract spatiotemporal features and generate an output tensor $\widehat{\mathbf{Z}}_{t+1} \in \mathbb{R}^{d_{\text{model}}\times L\times \frac{H}{p}\times\frac{W}{p}}$. It is then processed by the prediction head to generate the predicted patches $\widehat{\mathbf{P}}_{t+1} \in \mathbb{R}^{Cp^2\times L\times \frac{H}{p}\times\frac{W}{p}}$, which is reshaped to form the final output frames $\widehat{\mathbf{X}}_{t+1} \in \mathbb{R}^{C\times L\times H\times W}$. In the following subsections, we elaborate the proposed network components in detail.

C. SPATIAL EMBEDDING AND POSITIONAL ENCODING

The proposed spatial embedding module processes the input patches $\mathbf{P}_t \in \mathbb{R}^{Cp^2\times L\times \frac{H}{p}\times\frac{W}{p}}$ to produce the embedded feature maps. It adopts two 2D depthwise separable convolutional layers. As shown in (3), 2D depthwise separable convolution G is adopted to output intermediate feature map $\mathbf{J}_1 \in \mathbb{R}^{d_{\text{model}}\times L\times \frac{H}{p}\times\frac{W}{p}}$, followed by another 2D depthwise separable convolution S, which outputs $\mathbf{J}_2 \in \mathbb{R}^{d_{\text{model}}\times L\times \frac{H}{p}\times\frac{W}{p}}$. Both G and S are applied frame by frame. $\mathbf{J}_1$ and $\mathbf{J}_2$ are then added and passed through the *Dropout* layer to output the final embedded feature maps $\mathbf{Emb}_t \in \mathbb{R}^{d_{\text{model}}\times L\times \frac{H}{p}\times\frac{W}{p}}$.

$$\begin{aligned}\mathbf{J}_1 &= LeakyReLU\left(G^{Cp^2\times1\times1} * (G^{1\times7\times7} \otimes \mathbf{P}_t)\right)\\ \mathbf{J}_2 &= LeakyReLU\left(S^{d_{\text{model}}\times1\times1} * (S^{1\times5\times5} \otimes \mathbf{J}_1)\right)\\ \mathbf{Emb}_t &= Dropout\ (\mathbf{J}_1 + \mathbf{J}_2)\end{aligned} \tag{3}$$

In (3), $\otimes$ is the 2D depthwise convolution operation, and $*$ is the 1D pointwise convolution operation. G adopts a 2D depthwise convolution with Cp^2 filters of size $1 \times 7 \times 7$ followed by a 1D pointwise convolution with d_{model} filters of size $(Cp^2) \times 1 \times 1$. Similarly, S adopts a 2D depthwise convolution with d_{model} filters of size $1 \times 5 \times 5$ followed by a 1D pointwise convolution with d_{model} filters of size $d_{\text{model}} \times 1 \times 1$.

To preserve the positional information of the input video sequence, fixed positional encoding $\mathbf{Pos} \in \mathbb{R}^{d_{\text{model}}\times L\times \frac{H}{p}\times\frac{W}{p}}$ is calculated via (4) [26], where i represents the channel index, $0 \le i < d_{\text{model}}$, l is the temporal index $0 \le l < L-1$, (h, w) are the spatial indices, and $k = 10^4$.

$$\mathbf{Pos}_{i,l,h,w} = \begin{cases} \sin\left(\dfrac{l}{k^{\frac{i}{d_{\text{model}}}}}\right), & i \text{ even},\\[2ex] \cos\left(\dfrac{l}{k^{\frac{i-1}{d_{\text{model}}}}}\right), & i \text{ odd}.\end{cases} \tag{4}$$

$\mathbf{Pos}$ is added to the embedded feature maps $\mathbf{Emb}_t$ to generate the output $\mathbf{Z}_t \in \mathbb{R}^{d_{\text{model}}\times L\times \frac{H}{p}\times\frac{W}{p}}$ by

$$\mathbf{Z}_t = \mathbf{Emb}_t + \mathbf{Pos}. \tag{5}$$

D. 3D TRANSFORMER-LSTM

The stack of six 3DTransLSTM blocks across three time steps $t-1$, t, $t+1$ as shown in Fig. 2 are described in detail in Fig. 3 (a). Take time step t as an example: the positioned feature maps $\mathbf{Z}_t \in \mathbb{R}^{d_{\text{model}}\times L\times \frac{H}{p}\times\frac{W}{p}}$ pass through these six 3DTransLSTM blocks to output $\widehat{\mathbf{Z}}_{t+1}$. In each 3DTransLSTM block, the input is processed by two parallel branches. Branch 1 is indicated by the blue arrow. It extracts features within time step t through transformers, and outputs $\mathbf{Z}_t^{out1}$. Branch 2 enables information flow across three time steps $t-1, t, t+1$ through ST-LSTM structures, and outputs $\mathbf{Z}_t^{out2}$. In the following we give detailed descriptions of Branch 1 and Branch 2.

Branch 1: Its network structure is separately depicted in Fig. 4. It adopts two sub-blocks: 1) 3D separable convolution-based self-attention (*3DSepSA*), and 2) 3D separable convolution-based feed-forward network (*3DSepFFN*). Layer normalization (*LN*) is applied before each sub-block, and residual connection is applied after each sub-block.

1) 3DSepSA

As shown in Fig. 4 (a), the input tensor $\mathbf{Z}_t$ first goes through a *LN* layer by

$$\mathbf{Z}_t^{attn\_in} = LN(\mathbf{Z}_t). \tag{6}$$

As shown in Fig. 4 (b), the *3DSepSA* module then takes $\mathbf{Z}_t^{attn\_in} \in \mathbb{R}^{d_{\text{model}}\times L\times \frac{H}{p}\times\frac{W}{p}}$ as the input and calculates three tensors $\overline{\mathbf{Q}}, \overline{\mathbf{K}}, \overline{\mathbf{V}}$ of dimension $d_{\text{model}} \times L \times \frac{H}{p} \times \frac{W}{p}$ using 3D separable convolutional kernels W_q, W_k, W_v as follows

$$\begin{aligned}\overline{\mathbf{Q}} &= W_q^{d_{\text{model}}\times1\times1\times1} * \left(W_q^{1\times3\times3\times3} \circledast \mathbf{Z}_t^{attn\_in}\right),\\ \overline{\mathbf{K}} &= W_k^{d_{\text{model}}\times1\times1\times1} * \left(W_k^{1\times3\times3\times3} \circledast \mathbf{Z}_t^{attn\_in}\right),\\ \overline{\mathbf{V}} &= W_v^{d_{\text{model}}\times1\times1\times1} * \left(W_v^{1\times3\times3\times3} \circledast \mathbf{Z}_t^{attn\_in}\right),\end{aligned} \tag{7}$$

where $*$ denotes 1D pointwise convolution and $\circledast$ denotes 3D depthwise convolution.

To calculate the attention of the above tensors, we first transpose the tensors $\overline{\mathbf{Q}}, \overline{\mathbf{K}}$ and $\overline{\mathbf{V}}$ to dimension $\frac{H}{p} \times \frac{W}{p} \times L \times d_{\text{model}}$. Let $\frac{H}{p} \times \frac{W}{p}$ be the batch size, then the batch has $\frac{H}{p} \times \frac{W}{p}$ samples, and each sample is a tensor of size $L \times d_{\text{model}}$, denoted by $\mathbf{Q} \in \mathbb{R}^{L\times d_{\text{model}}}$, $\mathbf{K} \in \mathbb{R}^{L\times d_{\text{model}}}$,

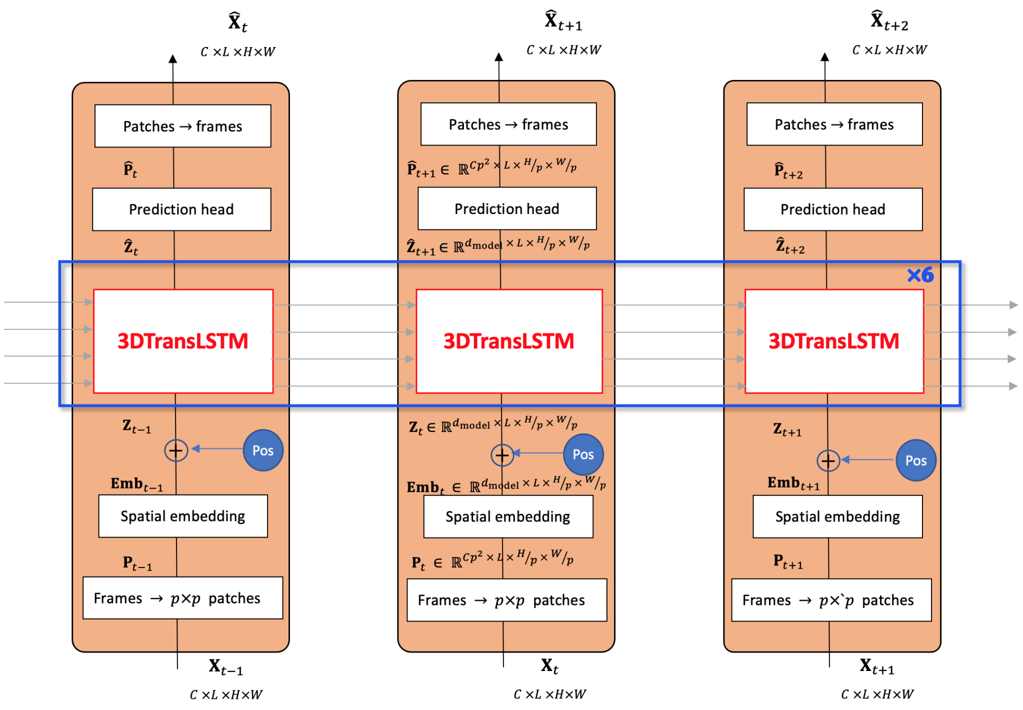


FIGURE 2. The overall architecture of the proposed network for three time steps $t-1$, t, and $t+1$.

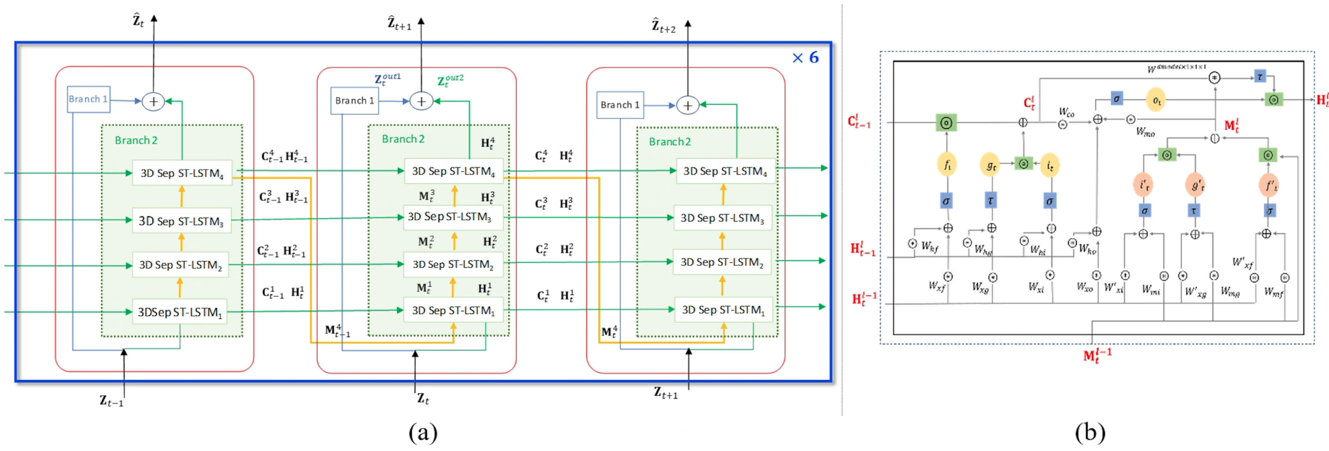


FIGURE 3. (a) The stack of six 3DTransLSTM blocks for time steps $t-1, t, t+1$. Each 3DTransLSTM block consists of Branch 1 (blue arrow) and Branch 2 (green arrow). Branch 2 further consists of 4 3D separable convolution-based ST-LSTM (3D SepST-LSTM) layers. (b) The structure of the l-th 3D SepST-LSTM layer at time step t.

and $\mathbf{V} \in \mathbb{R}^{L\times d_{\text{model}}}$. Within each sample, the self-attention $\mathbf{Z} \in \mathbb{R}^{L\times d_{\text{model}}}$ among L temporal elements is calculated as

$$\mathbf{Z} = Softmax\left(Mask(\frac{\mathbf{QK}^T}{\sqrt{d_{\text{model}}}})\right)\mathbf{V}. \tag{8}$$

We use a single-head masked self-attention where the masking mechanism only allows a position to look at the previous tokens and prevents the leaking of information to future tokens [27]. We mask out the attention to future tokens by setting their attention scores to $-\infty$, which generates zero weights after they are passed through the $Softmax(\cdot)$ function.

Next, the self-attentions of all samples in the batch are grouped and transposed to form the output self-attention $\tilde{\mathbf{Z}}_t^{attn} \in \mathbb{R}^{d_{\text{model}}\times L\times\frac{H}{p}\times\frac{W}{p}}$, which is then processed by 3D depthwise separable convolution W_A and a $Dropout$ layer as shown in (9),

$$\mathbf{Z}_t^{attn} = Dropout\left(W_A^{d_{\text{model}}\times1\times1\times1} * (W_A^{1\times3\times3\times3} \circledast \tilde{\mathbf{Z}}_t^{attn})\right). \tag{9}$$

Afterwards, a residual connection from the input $\mathbf{Z}_t$ is applied to $\mathbf{Z}_t^{attn}$ to produce the final output of the attention module $\mathbf{Z}_t^{attn\_out}$ as

$$\mathbf{Z}_t^{attn\_out} = \mathbf{Z}_t^{attn} + \mathbf{Z}_t. \tag{10}$$

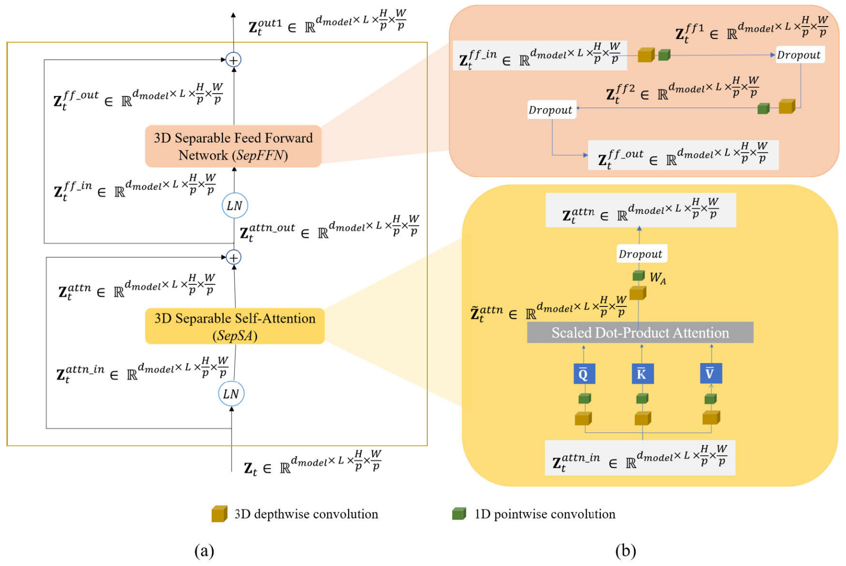

3D Separable Feed Forward Network (SepFFN)
3D Separable Self-Attention (SepSA)
LN
Dropout
Scaled Dot-Product Attention
Q
K
V
W_A
3D depthwise convolution
1D pointwise convolution
(a)
(b)

FIGURE 4. (a) Branch 1 of the 3DTransLSTM block, (b) ***3DSepFFN*** module (top), ***3DSepSA*** module (bottom).

2) 3DSepFFN

The second sub-block of Branch 1, *3DSepFFN*, takes the layer normalized $\mathbf{Z}_t^{attn\_out}$ as the input,

$$\mathbf{Z}_t^{ff\_in} = LN(\mathbf{Z}_t^{attn\_out}). \tag{11}$$

As shown in Fig. 4 (b), inside the *3DSepFFN* sub-block, there are two 3D depthwise separable convolutional layers W_{ff1} and W_{ff2}, each followed by a *Dropout* layer,

$$\begin{aligned}\mathbf{Z}_t^{ff1} &= LeakyReLU\Big(W_{ff1}^{d_{\text{model}}\times1\times1\times1} *\\ &\qquad (W_{ff1}^{1\times3\times3\times3} \circledast \mathbf{Z}_t^{ff\_in})\Big),\\ \mathbf{Z}_t^{ff2} &= W_{ff2}^{d_{\text{model}}\times1\times1\times1} * \Big(W_{ff2}^{1\times3\times3\times3} \circledast Dropout(\mathbf{Z}_t^{ff1})\Big),\\ \mathbf{Z}_t^{ff\_out} &= Dropout\left(\mathbf{Z}_t^{ff2}\right).\end{aligned} \tag{12}$$

The residual connection from the *3DSepSA* sub-block $\mathbf{Z}_t^{attn\_out} \in \mathbb{R}^{d_{\text{model}}\times L\times\frac{H}{p}\times\frac{W}{p}}$ and the output from the *3DSepFFN* sub-block $\mathbf{Z}_t^{ff\_out} \in \mathbb{R}^{d_{\text{model}}\times L\times\frac{H}{p}\times\frac{W}{p}}$ are added to form the final output $\mathbf{Z}_t^{out1} \in \mathbb{R}^{d_{\text{model}}\times L\times\frac{H}{p}\times\frac{W}{p}}$ of Branch 1,

$$\mathbf{Z}_t^{out1} = \mathbf{Z}_t^{ff\_out} + \mathbf{Z}_t^{attn\_out}. \tag{13}$$

Branch 2: At time step t, Branch 2 shown by the green arrow in Fig. 3 (a) enables spatiotemporal information flow. It contains 4 layers of 3D separable convolution-based ST-LSTM (3D SepST-LSTM), and outputs $\mathbf{Z}_t^{out2} \in \mathbb{R}^{d_{\text{model}}\times L\times\frac{H}{p}\times\frac{W}{p}}$.

As shown in Fig. 3 (a), at the l-th 3D SepST-LSTM layer, $l = 1, 2, 3, 4$, the inputs are the temporal memory cell $\mathbf{C}_{t-1}^l$ and hidden state $\mathbf{H}_{t-1}^l$ delivered horizontally from the same l-th layer of the previous time step $t-1$, as well as the spatiotemporal memory cell $\mathbf{M}_t^{l-1}$ and the hidden state $\mathbf{H}_t^{l-1}$ delivered vertically from the $(l-1)$-th 3D SepST-LSTM layer of the current time step t. The outputs are the spatiotemporal memory cell $\mathbf{M}_t^l$, the hidden state $\mathbf{H}_t^l$, and the temporal memory cell $\mathbf{C}_t^l$. While $\mathbf{M}_t^l$ is delivered vertically to the $(l+1)$-th 3D SepST-LSTM layer, $\mathbf{C}_t^l$ is delivered horizontally to the $(t+1)$-th time step, and $\mathbf{H}_t^l$ is delivered both vertically and horizontally. It is noteworthy that when $l = 1$, the input spatiotemporal memory cell $\mathbf{M}_t^0 = \mathbf{M}_{t-1}^4$, which is the output spatiotemporal memory cell of the 4-th 3D SepST-LSTM layer of the previous time step $t-1$, and the input hidden state $\mathbf{H}_t^0 = \mathbf{Z}_t$, which is the positional feature input of the entire stack of six 3DTransLSTM blocks.

All the gates and the input $\mathbf{Z}_t$, hidden state $\mathbf{H}_t^l$ and memory cells $\mathbf{M}_t^l$ and $\mathbf{C}_t^l$ are 4D tensors of size $d_{\text{model}} \times L \times \frac{H}{p} \times \frac{W}{p}$, where d_{model} is the number of channels, L is the temporal length, $\frac{H}{p}$ and $\frac{W}{p}$ are the height and width.

Fig. 3 (b) shows the structure of the l-th 3D SepST-LSTM layer at time step t. One set of input gate i_t, forget gate f_t, and input modulation gate g_t are generated by processing hidden states $\mathbf{H}_t^{l-1}$ and $\mathbf{H}_{t-1}^l$ by 3D separable convolution as shown in (14a). Here $\circledast$ is the 3D depthwise convolution, $*$ is the 1D pointwise convolution, σ is the sigmoid function and τ is the tanh function.

$$\begin{aligned}i_t &= \sigma\Big(W_{xi} * (W_{xi} \circledast \mathbf{H}_t^{l-1}) + W_{hi} * (W_{hi} \circledast \mathbf{H}_{t-1}^l)\Big)\\ f_t &= \sigma\Big(W_{xf} * (W_{xf} \circledast \mathbf{H}_t^{l-1}) + W_{hf} * (W_{hf} \circledast \mathbf{H}_{t-1}^l)\Big)\\ g_t &= \tau\Big(W_{xg} * (W_{xg} \circledast \mathbf{H}_t^{l-1}) + W_{hg} * (W_{hg} \circledast \mathbf{H}_{t-1}^l)\Big)\end{aligned} \tag{14a}$$

An extra set of input gate i_t', forget gate f_t', and input modulation gate g_t' are generated by processing the hidden state $\mathbf{H}_t^{l-1}$ and spatiotemporal memory cell $\mathbf{M}_t^{l-1}$ again using

3D separable convolution, as shown in (14b).

$$
\begin{aligned}
i'_t &= \sigma\Big(W'_{xi} * (W'_{xi} \circledast \mathbf{H}_t^{l-1}) + W_{mi} * (W_{mi} \circledast \mathbf{M}_t^{l-1})\Big)\\
f'_t &= \sigma\Big(W'_{xf} * (W'_{xf} \circledast \mathbf{H}_t^{l-1}) + W_{mf} * (W_{mf} \circledast \mathbf{M}_t^{l-1})\Big)\\
g'_t &= \tau\Big(W'_{xg} * (W'_{xg} \circledast \mathbf{H}_t^{l-1}) + W_{mg} * (W_{mg} \circledast \mathbf{M}_t^{l-1})\Big)
\end{aligned} \tag{14b}
$$

Afterwards, the output temporal memory cell $\mathbf{C}_t^l$ and spatiotemporal memory cell $\mathbf{M}_t^l$ are generated by (14c) and (14d), respectively, where $\odot$ represents the Hadamard product.

$$\mathbf{C}_t^l = f_t \odot \mathbf{C}_{t-1}^l + i_t \odot g_t \tag{14c}$$

$$\mathbf{M}_t^l = f'_t \odot \mathbf{M}_t^{l-1} + i'_t \odot g'_t \tag{14d}$$

The output gate o_t is generated by processing $\mathbf{H}_t^{l-1}$, $\mathbf{H}_{t-1}^l$, $\mathbf{M}_t^l$, and $\mathbf{C}_t^l$ again by 3D separable convolution, as shown in (14e).

$$
\begin{aligned}
o_t = \sigma\Big(& W_{xo} * (W_{xo} \circledast \mathbf{H}_t^{l-1}) + W_{ho} * (W_{ho} \circledast \mathbf{H}_{t-1}^l)\\
&+ W_{mo} * (W_{mo} \circledast \mathbf{M}_t^l) + W_{co} * (W_{co} \circledast \mathbf{C}_t^l)\Big)
\end{aligned} \tag{14e}
$$

Finally, the hidden state $\mathbf{H}_t^l$ is generated by

$$\mathbf{H}_t^l = o_t \odot \tau\Big(W^{2d_{\text{model}} \times 1 \times 1 \times 1} * [\mathbf{C}_t^l, \mathbf{M}_t^l]\Big), \tag{14f}$$

where $[\mathbf{C}_t^l, \mathbf{M}_t^l]$ is the channel concatenation of $\mathbf{C}_t^l$ and $\mathbf{M}_t^l$. In equations (14a), (14b), (14e), the depthwise 3D convolution kernal size is set as $1 \times 3 \times 3 \times 3$, and the pointwise convolution kernel size is set as $d_{\text{model}} \times 1 \times 1 \times 1$.

Branch 1 + Branch 2: As shown in Fig. 3 (a), the output $\mathbf{Z}_t^{out1}$ from Branch 1 and the output $\mathbf{Z}_t^{out2}$ from Branch 2 are added together to form the output of one 3DTransLSTM block. Six such 3DTransLSTM blocks are stacked to generate the final output $\widehat{\mathbf{Z}}_{t+1} \in \mathbb{R}^{d_{\text{model}} \times L \times \frac{H}{p} \times \frac{W}{p}}$.

E. PREDICTION HEAD

In Fig. 2, the final output of the 3DTransLSTM blocks $\widehat{\mathbf{Z}}_{t+1}$ is processed by a prediction head using pointwise convolution $W_{\widehat{Z}}$ with Cp^2 filters of size $d_{\text{model}} \times 1 \times 1 \times 1$ to output the predicted frame patches $\widehat{\mathbf{P}}_{t+1} \in \mathbb{R}^{Cp^2 \times L \times \frac{H}{p} \times \frac{W}{p}}$ as

$$\widehat{\mathbf{P}}_{t+1} = W_{\widehat{Z}}^{d_{\text{model}} \times 1 \times 1 \times 1} * \widehat{\mathbf{Z}}_{t+1}. \tag{15}$$

These frame patches are reshaped to generate the L predicted frames $\widehat{\mathbf{X}}_{t+1} \in \mathbb{R}^{C \times L \times H \times W}$ with frame indices $t+1 : t+L$.

V. EXPERIMENTAL STUDIES

In this section, we demonstrate the effectiveness of the proposed method through experiments conducted on four video prediction datasets, including both synthetic and real-word datasets. We analyze the performance of the proposed model in terms of prediction accuracy and model efficiency, and compare it with SOTA methods.

A. TRAINING AND INFERENCE STRATEGY

To train the network, we choose the widely used MSE as the loss function. The MSE between the ground-truth frame $\mathbf{Y}_k$ with time index k and the corresponding predicted frame $\widehat{\mathbf{Y}}_k$ is calculated as follows:

$$
\begin{aligned}
\text{MSE} = \frac{1}{C \times H \times W} \sum_{c=1}^{C} \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} \big(&\mathbf{Y}_k(c,i,j)\\
&- \widehat{\mathbf{Y}}_k(c,i,j)\big)^2
\end{aligned} \tag{16}
$$

where C, H, and W denotes the number of channels, the height and width of the frame, respectively.

As described in Section IV, in each time step, our proposed network takes L frames as the input (for example, frame t to frame $t+L-1$), and generates L predicted frames (for example, frame $t+1$ to frame $t+L$), with one time index shift. It is noteworthy that the video sequence length L can be different during the training and inference stages, and L can also vary during the inference stage, while the dimensions of the trained filters are fixed.

We train our model *Net* to predict the next L frames $\widehat{\mathbf{X}}_{t+1} = \{\widehat{\mathbf{Y}}_{t+1}, \widehat{\mathbf{Y}}_{t+2}, \cdots, \widehat{\mathbf{Y}}_{t+L}\} \in \mathbb{R}^{C \times L \times H \times W}$ by learning from the previous L frames $\mathbf{X}_t = \{\mathbf{Y}_t, \mathbf{Y}_{t+1}, \cdots, \mathbf{Y}_{t+L-1}\}$ in three successive iterations. In iteration 1, $\mathbf{X}_t$ is used to predict $\widehat{\mathbf{X}}_{t+1}$. In iteration 2, $\mathbf{X}_{t+1} \in \mathbb{R}^{C \times L \times H \times W}$ is used to predict $\widehat{\mathbf{X}}_{t+2} \in \mathbb{R}^{C \times L \times H \times W}$. In iteration 3, $\mathbf{X}_{t+2} \in \mathbb{R}^{C \times L \times H \times W}$ is used to predict $\widehat{\mathbf{X}}_{t+3} \in \mathbb{R}^{C \times L \times H \times W}$. Since there are three iterations and each iteration has L predicted frames, the MSE loss used to train the network is averaged over $3L$ predicted frames.

During inference, the model takes the previous K frames $\mathbf{X}_t = \{\mathbf{Y}_t, \mathbf{Y}_{t+1}, \cdots, \mathbf{Y}_{t+K-1}\} \in \mathbb{R}^{C \times K \times H \times W}$ as the input, and predicts the future N frames $\widehat{\mathbf{X}}_{t+K} = \{\widehat{\mathbf{Y}}_{t+K}, \widehat{\mathbf{Y}}_{t+K+1}, \cdots, \widehat{\mathbf{Y}}_{t+K+N-1}\} \in \mathbb{R}^{C \times N \times H \times W}$. The inference process is given in Algorithm 1.

Algorithm 1 The Inference Process

```
Input: $\mathbf{X}_t = \{\mathbf{Y}_t, \mathbf{Y}_{t+1}, \cdots, \mathbf{Y}_{t+K-1}\}$
$\mathbf{X}_{in} \leftarrow \mathbf{X}_t$
for $i = 0, 1, 2,$ to $N-1$ do
    $\{\widehat{\mathbf{Y}}_{t+1}, \widehat{\mathbf{Y}}_{t+2}, \cdots, \widehat{\mathbf{Y}}_{t+K+i}\} \leftarrow Net(\mathbf{X}_{in})$
    $\mathbf{X}_{in} \leftarrow Concat(\mathbf{X}_{in}, \widehat{\mathbf{Y}}_{t+K+i})$
end for
Output: $\widehat{\mathbf{X}}_{t+K} = \{\widehat{\mathbf{Y}}_{t+K}, \widehat{\mathbf{Y}}_{t+K+1}, \cdots, \widehat{\mathbf{Y}}_{t+K+N-1}\}$
```

As described in Algorithm 1, the inference process consists of N iterations. In the i-th iteration, the last predicted frame $\widehat{\mathbf{Y}}_{t+K+i}$ is concatenated with the current input video sequence to form a new input sequence $\mathbf{X}_{in} \in \mathbb{R}^{C \times (K+i+1) \times H \times W}$ of $K+i+1$ frames. After N iterations, a total of N future frames are predicted, denoted as $\widehat{\mathbf{X}}_{t+K} = \{\widehat{\mathbf{Y}}_{t+K}, \widehat{\mathbf{Y}}_{t+K+1}, \cdots, \widehat{\mathbf{Y}}_{t+K+N-1}\}$.

The input and output sequence length L during training, the input video frame number K and the predicted future frame number N during inference are provided for each dataset in Section V-B.

B. DATASETS

We conducted experimental studies on the synthetic MovingMNIST (MMNIST) dataset [13] and real-world datasets KTH Action [52] and Human 3.6m [53]. To evaluate the generalization ability of the proposed method, we also trained the model on the KITTI [54] dataset and tested it on an unseen Caltech Pedestrian [55] dataset.

1) MMNIST

This is a widely used synthetic grayscale video prediction dataset, with a frame size of $1 \times 64 \times 64$. The first dimension represents the grayscale channel. Two digits continuously move in the frame, initialized at a random location. The digits move in a constant velocity and angle, bouncing off the edges of the frame. Following TCTN [27], we used two subsets of this dataset for training: MMNIST-2K with 2,000 video sequences and MMNIST-10K with 10,000 video sequences. Each sequence has 20 frames. We trained one model on each subset and tested both trained models on another unseen MMNIST subset of 3,000 video sequences (MMNIST-3K). During training, the input and output sequence lengths were both $L = 17$. During testing, we used $K = 10$ previous frames to predict $N = 10$ future frames.

2) KTH ACTION

KTH is a grayscale dataset originally used for action recognition. It has video sequences of 25 individuals doing six types of actions: walking, running, jogging, boxing, hand waving and clapping. Videos are shot with static cameras at 25 frames per second (fps) in four settings, namely, outdoors, outdoors with scale variations, outdoors with different clothes, and indoors. Individuals 1-16 were used for training and individuals 17-25 were used for testing. The training set contained 8,488 sequences and the testing set had 5,041 sequences. Each sequence had 20 frames, and each frame was resized to $1 \times 64 \times 64$, where the first dimension represented the grayscale channel. Similar to the MMNIST dataset, during training, the input and output sequence length is $L = 17$ frames. During testing, the previous $K = 10$ frames were used to predict $N = 10$ future frames.

3) HUMAN 3.6m

This is a complex human pose action dataset, originally with 3.6 million RGB images. It comprises of video sequences with humans performing different types of actions. Each sequence has 8 frames. We followed the experiment setting in [19] and chose only the "walking" scenario. The original $3 \times 1000 \times 1000$ resolution frames were resized to $3 \times 128 \times 128$ resolution in our experiments, where the first dimension represented the RGB channels. Subjects S1, S5, S6, S7, S8 were used for training which had 2,624 video sequences, and subjects S9, S11 were used for testing which had 1,135 video sequences. For training we set the input and output sequence length as $L = 5$ and for testing we used the previous $K = 4$ frames to predict the future $N = 4$ frames.

4) KITTI AND CALTECH PEDESTRIAN

The KITTI and Caltech Pedestrian datasets are two car-mounted camera video datasets with real-world scenarios, widely used for video frame prediction. The KITTI dataset was used for model training. It consisted of 3,150 sequences and each sequence had 13 frames. A subset of the Caltech dataset was used for model testing. This dataset was collected from a vehicle driving through regular traffic in an urban environment. It had 1,983 sequences and each sequence had 11 frames. We followed [56] to preprocess the datasets. The frames from both datasets were center-cropped and resized to $3 \times 128 \times 160$, where the first dimension represented the RGB channels. During training, we set the input and output sequence length to be $L = 10$. During testing, the previous $K = 10$ frames were used to predict the next $N = 1$ frames.

C. EXPERIMENT SETTINGS

The models were trained with the PyTorch framework using an NVIDIA Tesla V100 32 GB GPU. The ADAM optimizer was used to minimize the MSE loss between the ground-truth and the predicted frames. To prevent overfitting, dropout layers of the proposed network adopt a dropout rate of 0.05 during training. The model was trained for 200 epochs for MMNIST and KTH Action, and 300 epochs for Human 3.6m and KITTI. We set the patch size as $p = 4$ and the hidden dimension as $d_{\text{model}} = 128$ for MMNIST, KTH Action, and Human 3.6m. For KITTI dataset, we set the patch size as $p = 2$ and the hidden dimension as $d_{\text{model}} = 256$. We used a stride of 1 for the depthwise separable convolutions in our network.

D. METHODS IN COMPARISON

We compared the performance of our proposed model to seven existing methods. They included three transformer-based models: ConvTransformer [26], TCTN [27], and VPTR [28], and four RNN-based models: FRNN [12], E3D-LSTM [15], MIM [17], and PredRNN-V2 [19]. All the models were implemented with the same experiment settings proposed in their original works. To verify the generalization ability of our proposed model, we chose the following existing methods for comparison studies: the ConvLSTM-based models DNA [14], CrevNet [18], the GAN-based DM-GAN [42], the transformer-based VPTR [28], as well as the CNN-based BeyondMSE [41].

E. EVALUATION METRICS

The quality of the predicted frame $\widehat{\mathbf{Y}}_k$ compared to the original frame $\mathbf{Y}_k$, was evaluated using the peak signal-to-noise ratio (PSNR) defined as

$$\text{PSNR} = 10 \log_{10} \frac{255^2}{\text{MSE}(\mathbf{Y}_k, \widehat{\mathbf{Y}}_k)}. \tag{17}$$

Besides, the structural similarity index (SSIM) was also calculated. It is a metric consistent with human subjective opinion and is defined as [57]:

$$\text{SSIM}(\mathbf{Y}_k, \widehat{\mathbf{Y}}_k) = l(\mathbf{Y}_k, \widehat{\mathbf{Y}}_k)^{\alpha} \times c(\mathbf{Y}_k, \widehat{\mathbf{Y}}_k)^{\beta} \times g(\mathbf{Y}_k, \widehat{\mathbf{Y}}_k)^{\gamma}, \tag{18}$$

TABLE 1. Quantitvative results on MMNIST-3K and KTH Action datasets. The best, second-best, and third-best results of each metric are highlighted in red, blue and brown, respectively.

Methods	MMNIST-3K (Trained on MMNIST-2K)		MMNIST-3K (Trained on MMNIST-10K)		KTH Action		Model size (MB)	Params (M)	GFLOPs
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM			
VPTR [28]	19.12	0.854	19.40	0.8773	28.93	0.891	1988.2	167.9	201.8
TCTN [27]	20.45	0.886	22.66	0.918	27.83	0.874	344.1	31.9	511.3
MIM [17]	20.78	0.882	23.14	0.927	28.96	0.903	211.9	41.6	795.2
E3D-LSTM [15]	18.83	0.843	20.86	0.893	27.15	0.872	201.5	38.7	253.6
PredRNN-V2 [19]	20.52	0.891	23.78	0.921	28.56	0.884	93.1	23.9	116.6
FRNN [12]	17.89	0.824	19.23	0.837	26.12	0.771	89.2	-	-
ConvTransformer [26]	17.68	0.833	19.41	0.810	27.61	0.815	81.0	20.2	228.0
3DTransLSTM	21.04	0.893	23.57	0.921	28.78	0.890	56.6	11.6	140.1

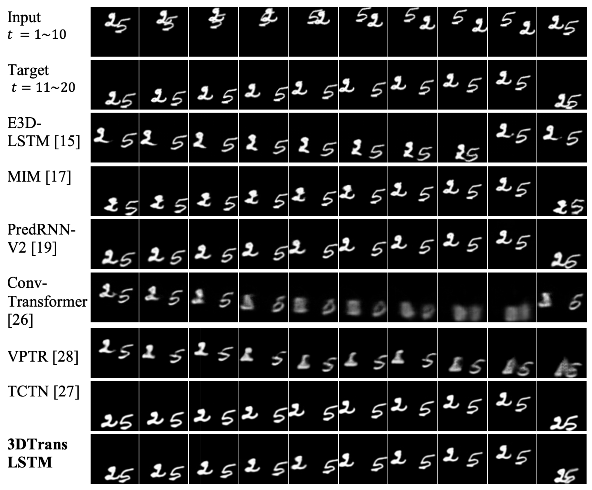


FIGURE 5. The visual results on the MMNIST-3K dataset [13] for models trained on the MMNIST-10K dataset.

where l, c, and g are the luminance, contrast and structure comparison measures between $\mathbf{Y}_k$ and $\widehat{\mathbf{Y}}_k$, and α, β and γ are parameters to define the relative importance of these three components [57]. The final reported PSNR and SSIM were averaged over all N predicted future frames. Higher PSNR and SSIM values indicate that the predicted frames have higher quality.

Besides the aforementioned accuracy metrics, the model efficiency is also evaluated by calculating the model size measured in megabytes (MB) and the number of trainable parameters measured in millions (M). We also use the evaluation metric - giga floating point operations (GFLOPs) to infer the computational complexity of the model. GFLOPs calculate the total number of floating point operations, such as addition, subtraction, multiplication and division needed for model inference. Lower GFLOPs usually indicate less computationally expensive models.

F. EXPERIMENTAL RESULTS

1) MMNIST

Though the dynamics of MMNIST seem simple, the frequent occlusion and overlapping of the digits make the prediction complex. Table 1 compares the frame prediction accuracy performance between the proposed model and existing models on the MMNIST-3K test set after the models were trained on the MMNIST-2K and MMNIST-10K datasets. The best, second-best, and third-best results of each metric are highlighted in red, blue and brown, respectively. Our proposed model 3DTransLSTM achieved the highest PSNR and SSIM scores when it was trained on MMNIST-2K, and it achieved the second highest PSNR and SSIM scores when it was trained on MMNIST-10K. It is superior in terms of model size and parameters, and it requires the second fewest GFLOPs for inference. When the models were trained on MMNIST-10K, PredRNN-V2 [19] achieved the highest PSNR, but our proposed model has 39.2% decrease in model size and 51.5% decrease in parameters, compared to PredRNN-V2. To visually demonstrate the effectiveness of our proposed model, Fig. 5 shows the qualitative results of models trained on MMNIST-10K. We observe that ConvTransformer [26] and VPTR [28] generated blurry results and E3D-LSTM [15] failed to correctly predict the relative positions of the digits. The results of our proposed 3DTransLSTM model correctly captured the trajectory of moving digits and looked similar to the ground truth without blurriness. MIM [17], PredRNN-V2 [19], and TCTN [27] produced good visual results at the expense of huge model size, parameters and computational complexity compared to our proposed model.

2) KTH ACTION

The KTH Action dataset has a similar frame size (64 × 64) and input/output sequence length as the MMNIST dataset. The quantitative results are also summarized in Table 1. Obviously, the proposed 3DTransLSTM model achieved the third highest PSNR and SSIM values. Although the MIM [17] and VPTR [28] models achieved slightly higher PSNR and SSIM values than 3DTransLSTM, their model sizes were much larger, and they required much more parameters and GFLOPs. Fig. 6 shows the predicted frames of different models. We observe that the proposed 3DTransLSTM model predicted the motion of the hands correctly and it preserved the structure of the person. In contrast, the transformer-based models VPTR [28], ConvTransformer [26], and the ConvLSTM-based model E3D-LSTM [15] generated blurry results and had missing arms. The visual results of MIM [17], TCTN [27], and PredRNN-V2 [19] are close to the proposed

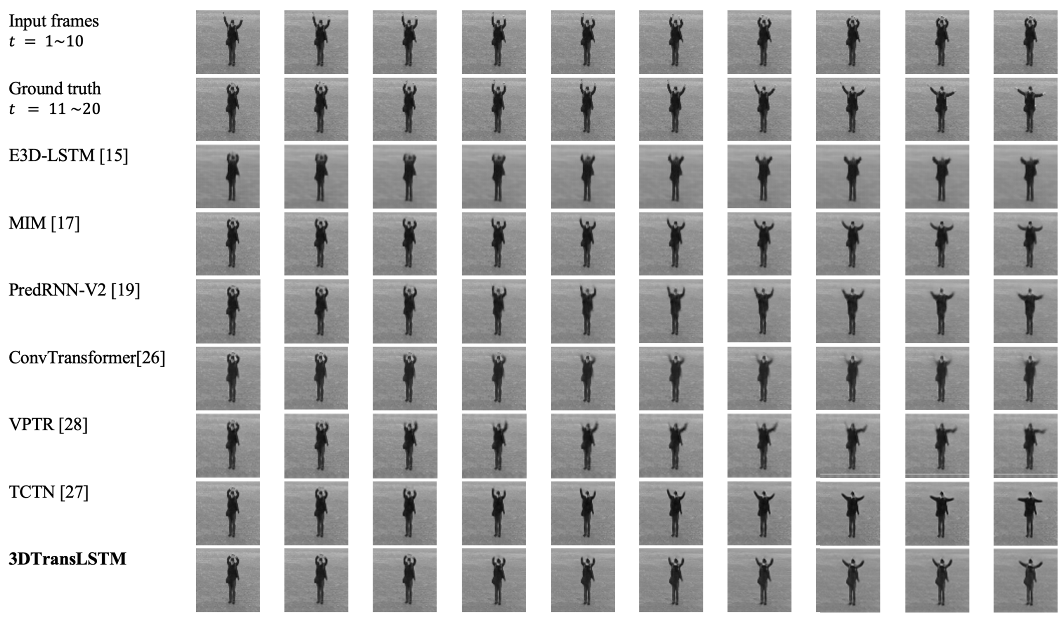


FIGURE 6. **The visual results on the KTH Action dataset [52]. The proposed 3DTransLSTM model predicted the motion of the hands correctly and preserved the structure of the person, unlike the blurry results of VPTR [28], ConvTransformer [26] and E3D-LSTM [15].**

TABLE 2. **Quantitative results on human 3.6m dataset. ∗ The results are reported in [17]. The best, second-best, and third-best results of each metric are highlighted in red, blue and brown, respectively.**

Methods	PSNR	SSIM	Model size (MB)	Params (M)	GFLOPs
VPTR [28]	**29.31**	**0.893**	2513.9	198.8	180.9
TCTN [27]	**27.63**	**0.839**	344.1	32.1	492.4
E3D-LSTM* [15]	19.79	0.735	263.5	40.9	395.7
MIM* [17]	21.80	0.790	196.2	41.9	775.2
PredRNN-V2 [19]	20.73	0.790	156.6	**24.6**	**176.3**
FRNN* [12]	21.16	0.771	**109.2**	-	-
ConvTransformer [26]	20.97	0.798	**81.0**	**20.1**	**228.7**
3DTransLSTM	**28.60**	**0.875**	**56.2**	**11.6**	**107.0**

TABLE 3. **Quantitative results on Caltech Pedestrian dataset. ∗ The results are reported in [18]. † The results are reported in [58]. The best, sec'ond-best, and third-best results of each metric are highlighted in red, blue and brown, respectively.**

Methods	PSNR	SSIM	Model size (MB)	Params (M)	GFLOPs
CrevNet* [18]	**29.25**	**0.925**	2700	-	**350.4**
VPTR [28]	**29.12**	**0.921**	2513.9	208.7	201.6
BeyondMSE† [41]	24.87	0.881	**218.4**	-	-
DM-GAN* [42]	26.29	0.899	-	**113**	-
DNA† [14]	-	0.896	-	>160	-
3DTransLSTM	**28.81**	**0.915**	**207.2**	**43.1**	**55.9**

3DTransLSTM, but they require much more computational cost.

3) HUMAN 3.6m

This human pose dataset has video frames of dimension 3 × 128 × 128, where the first dimension represents the RGB channels. Our proposed model predicts the future four frames based on previous four frames. Table 2 summarizes the quantitative results of the compared models. The proposed 3DTransLSTM achieved the smallest model size, fewest number of parameters and GFLOPs. In terms of prediction accuracy, it achieved the second highest PSNR and SSIM values. Although VPTR [28] achieved the highest frame

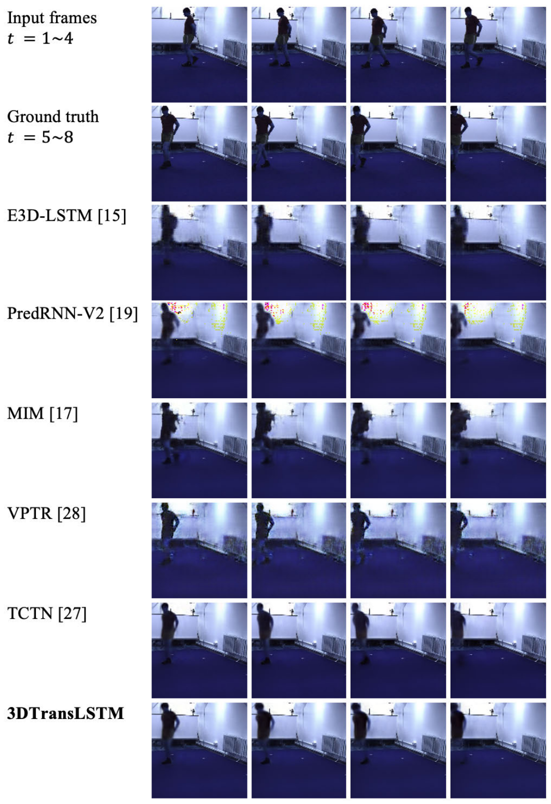


FIGURE 7. **The visual results on the human 3.6m dataset [53].**

TABLE 4. Ablation study on MMNIST-3K. The best and second-best results of each metric are highlighted in red and blue, respectively.

Model	Method	MMNIST-3K (Trained on MMNIST-2K)		Model size (MB)	Params (M)	GFLOPs
		PSNR	SSIM			
A	3DTransLSTM (traditional 2D)	19.44	0.844	745	78.7	934.7
B	3DTransLSTM (2D separable)	18.03	0.791	**50.9**	**10.5**	**126.1**
C	3DTransLSTM (traditional 3D)	**21.48**	**0.913**	1100	231.5	2914.3
	3DTransLSTM	**21.03**	**0.893**	**56.6**	**11.6**	**140.1**

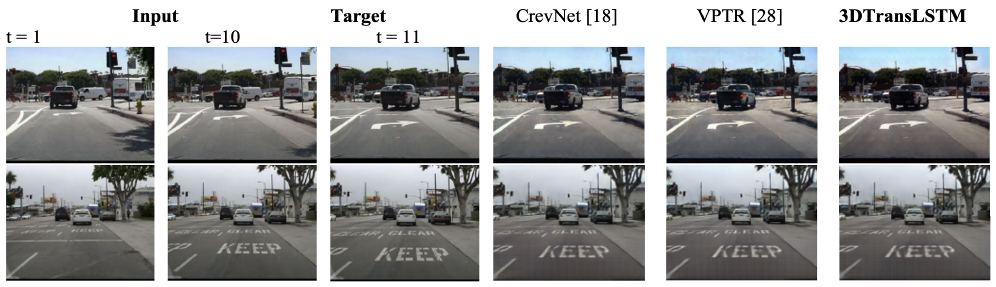


FIGURE 8. The visual results on the Caltech Pedestrian dataset [55] shows that our proposed model 3DTransLSTM produces results similar to high computational models such as CrevNet [18] and VPTR [28].

prediction accuracy, it was 44 times the size of our model, and it required 17 times more parameters than our model. Fig. 7 shows the visual quality of the predicted frames. We observe that the proposed 3DTransLSTM model is able to preserve the structure of the person's body and shape of arms. In contrast, the solely ConvLSTM-based models E3D-LSTM [15], MIM [17] and PredRNN-V2 [19] produced vague results and had inconsistencies in the physical appearance of the person, especially the shape of arms. Besides, the transformer-based model TCTN [27] also generated blurry results. The quantitative and qualitative results show that our model works effectively for RGB video frames.

4) KITTI AND CALTECH PEDESTRIAN

To evaluate the generalization ability of video prediction models, we trained the models on the KITTI dataset [54] using the previous 10 frames to predict the next one frame and evaluated the trained models on the Caltech Pedestrian dataset [55]. Both the KITTI and Caltech Pedestrian datasets are complex datasets comprised of real-world scenarios with multiple moving objects in the background. We observe from Table 3 that the proposed 3DTransLSTM achieved the best model efficiency in terms of model size, number of parameters, and GFLOPs. Besides, it achieved the third highest PSNR and SSIM values. Although CrevNet [18] has the highest PSNR and SSIM values, it required much larger model size, much more parameters and GFLOPs than our model did. Fig. 8 shows that our proposed model generated accurate visual results similar to CrevNet [18] and VPTR [28], producing clear objects and texts in the predicted frames.

G. ABLATION STUDY

We conducted a series of experiments with various designs of the proposed TransLSTM network to validate the advantages of using 3D separable convolutions. The models were trained on the MMNIST-2K dataset and tested on the MMNIST-3K dataset. The results are summarized in Table 4.

First, we designed Model A, which is the TransLSTM network with traditional 2D convolutions in the spatial embedding module, the self-attention and feed-forward layers of Branch 1, and in the ST-LSTM layers of Branch 2. It achieved an average PSNR of 19.44 dB and an average SSIM of 0.844. Its model size is 745MB. It has 78.7 M trainable parameters, and it requires 934.7 GFLOPs to conduct inference.

To reduce the model size, parameters, and computational complexity of Model A, we designed Model B, which is the TransLSTM network with 2D separable convolutions. It apparently reduces the model size, parameters, and computation complexity to 50.9 MB, 10.5 M, and 126.09 GFLOPs, respectively. Nevertheless, the video frame prediction accuracy has decreased to an average PSNR of 18.03 dB and an average SSIM of 0.791.

To leverage temporal information in the video sequence, we further designed Model C, the TransLSTM with traditional 3D convolutions in self-attention and feed-forward layers of Branch 1 and ST-LSTM layers of Branch 2. The results in Table 4 showed that the prediction accuracy was significantly improved compared to traditional 2D convolutions (Model A), achieving an average PSNR of 21.48 dB and an average SSIM of 0.913. However, this comes at a price of dramatically increased model size (1,100 MB), number of trainable parameters (231.5 M) and GFLOPs (2,914.3).

In contrast, our proposed model, 3DTransLSTM with 3D separable convolutions, significantly reduces the model size, number of parameters, and GFLOPs by 94.9%, 95%, and 95.2%, respectively, compared to Model C. Meanwhile, it only slightly decreased the prediction accuracy by 2.1% for the PSNR metric and by 2.2% for the SSIM metric.

The above ablation studies thoroughly validate the benefits of the proposed 3DTransLSTM with 3D separable convolutions. This network architecture not only leverages the spatiotemporal correlations in the video to provide competitive frame prediction accuracy, but also significantly reduces the model size, trainable parameters and computational complexity compared to traditional 3D convolutions. Therefore, it offers a good trade-off between model accuracy and model complexity.

VI. CONCLUSION

Video frame prediction is a challenging yet essential vision task in various real-world scenarios. In this era of Green AI, it is extremely important for machine learning models to offer both model efficiency and accuracy. In this paper, we devise a novel video prediction framework 3DTransLSTM, incorporating both transformer and LSTM structures with 3D separable convolutions. Extensive experimental results demonstrate the effectiveness of our proposed scheme on both synthetic and real-world datasets. Compared to existing approaches, our method is able to achieve competitive prediction accuracy with significantly reduced model size, number of parameters, and computational complexity. Hence, our model is more suitable for memory-constrained and computation resource-limited platforms, such as mobile and embedded devices. In the future, we plan to adapt this approach to various video processing tasks, such as reference frame generation in learning-based video coding, video super-resolution, and omnidirectional video frame prediction.

REFERENCES

[1] B. Liu, Y. Chen, S. Liu, and H.-S. Kim, "Deep learning in latent space for video prediction and compression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Nashville, TN, USA, Jun. 2021, pp. 701–710.

[2] Y. Zhou, H. Dong, and A. El Saddik, "Deep learning in next-frame prediction: A benchmark review," *IEEE Access*, vol. 8, pp. 69273–69283, 2020.

[3] W. Liu, W. Luo, D. Lian, and S. Gao, "Future frame prediction for anomaly detection—A new baseline," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, Jun. 2018, pp. 6536–6545.

[4] X. Lin, Q. Zou, X. Xu, Y. Huang, and Y. Tian, "Motion-aware feature enhancement network for video prediction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 2, pp. 688–700, Feb. 2021.

[5] S. Li, J. Fang, H. Xu, and J. Xue, "Video frame prediction by deep multi-branch mask network," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 4, pp. 1283–1295, Apr. 2021.

[6] H. Choi and I. V. Bajic, "Deep frame prediction for video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 7, pp. 1843–1855, Jul. 2020.

[7] J.-K. Lee, N. Kim, S. Cho, and J.-W. Kang, "Deep video prediction network-based inter-frame coding in HEVC," *IEEE Access*, vol. 8, pp. 95906–95917, 2020.

[8] R. Yang, R. Timofte, and L. Van Gool, "Advancing learned video compression with in-loop frame prediction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 5, pp. 2410–2423, May 2023.

[9] Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, and M. H. Yang, "Flow-grounded spatial–temporal video prediction from still images," in *Proc. Eur. Conf. Comput. Vis.*, Munich, Germany, Sep. 2018, pp. 600–615.

[10] F. A. Reda, G. Liu, K. J. Shih, R. Kirby, J. Barker, D. Tarjan, and B. Catanzaro, "SDC-Net: Video prediction using spatially-displaced convolution," in *Proc. Eur. Conf. Comput. Vis.*, Munich, Germany, Sep. 2018, pp. 718–733.

[11] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using LSTMs," in *Proc. PMLR Int. Conf. Mach. Learn.*, vol. 37, Lille, France, Jul. 2015, pp. 843–852.

[12] M. Oliu, J. Selva, and S. Escalera, "Folded recurrent neural networks for future video prediction," in *Proc. Eur. Conf. Comput. Vis.*, Munich, Germany, Sep. 2018, pp. 716–731.

[13] 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 *Proc. Int. Conf. Neural Inf. Process. Syst.*, vol. 28, Montreal, QC, Canada, Dec. 2015, pp. 802–810.

[14] C. Finn, I. Goodfellow, and S. Levine, "Unsupervised learning for physical interaction through video prediction," in *Proc. Adv. Neural Inf. Proc. Syst.*, vol. 29, Barcelona, Spain, Dec. 2016, pp. 1–9.

[15] Y. Wang, L. Jiang, M. H. Yang, L.-J. Li, M. Long, and L. Fei-Fei, "Eidetic 3D LSTM: A model for video prediction and beyond," in *Proc. Int. Conf. Learn. Represent.*, New Orleans, LA, USA, May 2019, pp. 1–14.

[16] Y. Wang, M. Long, J. Wang, Z. Gao, and S. Y. Philip, "PredRNN: Recurrent neural networks for predictive learning using spatiotemporal LSTMs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, vol. 30, Long Beach, CA, USA, Dec. 2017, pp. 879–888.

[17] Y. Wang, J. Zhang, H. Zhu, M. Long, J. Wang, and P. S. Yu, "Memory in memory: A predictive neural network for learning higher-order non-stationarity from spatiotemporal dynamics," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Long Beach, CA, USA, Jun. 2019, pp. 9154–9162.

[18] W. Yu, Y. Lu, S. Easterbrook, and S. Fidler, "Efficient and information-preserving future frame prediction and beyond," in *Proc. Int. Conf. Learn. Represent.*, Apr. 2020, pp. 1–14.

[19] Y. Wang, H. Wu, J. Zhang, Z. Gao, J. Wang, P. S. Yu, and M. Long, "PredRNN: A recurrent neural network for spatiotemporal predictive learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 2, pp. 2208–2225, Feb. 2023.

[20] M. Mathai, Y. Liu, and N. Ling, "A lightweight model with separable CNN and LSTM for video prediction," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Austin, TX, USA, May 2022, pp. 516–520.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Proc. Syst.*, vol. 30, Long Beach, CA, USA, Dec. 2017, pp. 1–12.

[22] J. Li, W. Wang, J. Chen, L. Niu, J. Si, C. Qian, and L. Zhang, "Video semantic segmentation via sparse temporal transformer," in *Proc. ACM Int. Conf. Multimedia*, Chengdu, China, Oct. 2021, pp. 59–68.

[23] H. Yuan, Z. Cai, H. Zhou, Y. Wang, and X. Chen, "TransAnomaly: Video anomaly detection using video vision transformer," *IEEE Access*, vol. 9, pp. 123977–123986, 2021.

[24] M. Fujitake and A. Sugimoto, "Video sparse transformer with attention-guided memory for video object detection," *IEEE Access*, vol. 10, pp. 65886–65900, 2022.

[25] J. Wensel, H. Ullah, and A. Munir, "ViT-ReT: Vision and recurrent transformer neural networks for human activity recognition in videos," *IEEE Access*, vol. 11, pp. 72227–72249, 2023.

[26] Z. Liu, S. Luo, W. Li, J. Lu, Y. Wu, S. Sun, C. Li, and L. Yang, "ConvTransformer: A convolutional transformer network for video frame synthesis," 2020, *arXiv:2011.10185*.

[27] Z. Yang, X. Yang, and Q. Lin, "PTCT: Patches with 3D-temporal convolutional transformer network for precipitation nowcasting," 2021, *arXiv:2112.01085*.

[28] X. Ye and G. A. Bilodeau, "VPTR: Efficient transformers for video prediction," in *Proc. IEEE Int. Conf. Pattern Recognit.*, Montréal, QC, Canada, Aug. 2022, pp. 3492–3499.

[29] W. Li, X. Wang, X. Xia, J. Wu, J. Li, X. Xiao, M. Zheng, and S. Wen, "SepViT: Separable vision transformer," 2022, *arXiv:2203.15380*.

[30] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, "Green AI," *Commun. ACM*, vol. 63, no. 12, pp. 54–63, Nov. 2020, doi: 10.1145/3381831.

[31] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell, "On the dangers of stochastic parrots: Can language models be too big?" in *Proc. ACM Conf. Fairness, Accountability, Transparency*, Virtual Event Canada, Mar. 2021, pp. 610–623.

[32] V. Vukotić, S. L. Pintea, C. Raymond, G. Gravier, and J. C. V. Gemert, "One-step time-dependent future video frame prediction with a convolutional encoder–decoder neural network," in *Proc. Int. Conf. Image Anal. Process.*, Catania, Italy, Sep. 2017, pp. 140–151.

[33] M. A. Yilmaz and A. M. Tekalp, "DFPN: Deformable frame prediction network," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Anchorage, AK, USA, Sep. 2021, pp. 1944–1948.

[34] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[35] B. Bross, Y.-K. Wang, Y. Ye, S. Liu, J. Chen, G. J. Sullivan, and J.-R. Ohm, "Overview of the versatile video coding (VVC) standard and its applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 10, pp. 3736–3764, Oct. 2021.

[36] X. J. 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 *Proc. Adv. Neural Inf. Proc. Syst.*, vol. 30, Long Beach, CA, USA, Dec. 2017, pp. 5617–5627.

[37] 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," in *Proc. PMLR Int. Conf. Mach. Learn.*, Stockholm, Sweden, Jul. 2018, pp. 5123–5132.

[38] J. Su, W. Byeon, J. Kossaifi, F. Huang, J. Kautz, and A. Anandkumar, "Convolutional tensor-train LSTM for spatio-temporal learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 13714–13726.

[39] Z. Chang, X. Zhang, S. Wang, S. Ma, and W. Gao, "IPRNN: An information-preserving model for video prediction using spatiotemporal GRUs," in *Proc. IEEE Int. Conf. Image Process.*, Anchorage, AK, USA, Sep. 2021, pp. 2703–2707.

[40] Z. Chang, X. Zhang, S. Wang, S. Ma, Y. Ye, and W. Gao, "ASTM: An attention based spatiotemporal model for video prediction using 3D convolutional neural networks," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Shenzhen, China, Jul. 2021, pp. 1–6.

[41] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," in *Proc. 4th Int. Conf. Learn. Represent.*, San Juan, Puerto Rico, May 2016, pp. 1–14.

[42] Z. Yi, H. Zhang, P. Tan, and M. Gong, "DualGAN: Unsupervised dual learning for image-to-image translation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy, Oct. 2017, pp. 2868–2876.

[43] Y.-H. Kwon and M.-G. Park, "Predicting future frames using retrospective cycle GAN," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Long Beach, CA, USA, Jun. 2019, pp. 1811–1820.

[44] Q. N. Tran and S.-H. Yang, "Attention-based inter-prediction for versatile video coding," *IEEE Access*, vol. 11, pp. 84313–84322, 2023.

[45] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[46] A. Pfeuffer and K. Dietmayer, "Separable convolutional LSTMs for faster video segmentation," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Auckland, New Zealand, Oct. 2019, pp. 1072–1078.

[47] B. Hou, Y. Liu, and N. Ling, "A super-fast deep network for moving object detection," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Seville, Spain, Oct. 2020, pp. 1–5.

[48] Z. Islam, M. Rukonuzzaman, R. Ahmed, Md. H. Kabir, and M. Farazi, "Efficient two-stream network for violence detection using separable convolutional LSTM," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Shenzhen, China, Jul. 2021, pp. 1–8.

[49] X. Zhang, Y. Tie, and L. Qi, "Dynamic gesture recognition based on 3D separable convolutional LSTM networks," in *Proc. IEEE 11th Int. Conf. Softw. Eng. Service Sci. (ICSESS)*, Beijing, China, Oct. 2020, pp. 180–183.

[50] B. Hou, Y. Liu, N. Ling, L. Liu, Y. Ren, and M. K. Hsu, "F3DsCNN: A fast two-branch 3D separable CNN for moving object detection," in *Proc. Int. Conf. Vis. Commun. Image Process. (VCIP)*, Munich, Germany, Dec. 2021, pp. 1–5.

[51] B. Hou, Y. Liu, N. Ling, L. Liu, and Y. Ren, "A fast lightweight 3D separable convolutional neural network with multi-input multi-output for moving object detection," *IEEE Access*, vol. 9, pp. 148433–148448, 2021.

[52] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: A local SVM approach," in *Proc. 17th Int. Conf. Pattern Recognit.*, vol. 3, Cambridge, U.K., Aug. 2004, pp. 32–36.

[53] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3.6M: large scale datasets and predictive methods for 3D human sensing in natural environments," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 7, pp. 1325–1339, Jul. 2014.

[54] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013.

[55] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: A benchmark," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Miami, FL, USA, Jun. 2009, pp. 304–311.

[56] W. He, T. Xiong, H. Wang, J. He, X. Ren, Y. Yan, and L. Tan, "Radar echo spatiotemporal sequence prediction using an improved ConvGRU deep learning model," *Atmosphere*, vol. 13, no. 1, p. 88, Jan. 2022.

[57] Y. Liu, P. Du, and Y. Li, "Hierarchical motion-compensated deep network for video compression," *Proc. SPIE*, vol. 11730, pp. 124–131, Apr. 2021.

[58] Y.-H. Ho, C.-Y. Cho, and W.-H. Peng, "Deep reinforcement learning for video prediction," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Taipei, Taiwan, Sep. 2019, pp. 604–608.

MAREETA MATHAI (Graduate Student Member, IEEE) received the B.Comp. degree in computer engineering from the National University of Singapore, in 2011, and the M.Eng. degree from Nanyang Technological University, Singapore, in 2016. She is currently pursuing the Ph.D. degree in computer science and engineering with Santa Clara University. She is working with advisors Dr. Nam Ling and Dr. Ying Liu on efficient deep-learning methods for video prediction and video coding. Her research interests include deep learning, computer vision, predictive modeling, and video prediction.

YING LIU (Member, IEEE) received the B.S. degree in communications engineering from Beijing University of Posts and Telecommunications, Beijing, China, in 2006, and the M.S. and Ph.D. degrees in electrical engineering from The State University of New York, Buffalo, NY, USA, in 2008 and 2012, respectively. She is currently an Assistant Professor with the Department of Computer Science and Engineering, Santa Clara University, Santa Clara, CA, USA. Her main research interests include image and video processing, deep learning, and computer vision. She serves as an Associate Editor for IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY.

NAM LING (Life Fellow, IEEE) received the B.Eng. degree from the National University of Singapore, in 1981, and the M.S. and Ph.D. degrees from the University of Louisiana at Lafayette, Lafayette, LA, USA, in 1985 and 1989, respectively. From 2002 to 2010, he was the Associate Dean of the School of Engineering, Santa Clara University, Santa Clara, CA, USA. He was the Sanfilippo Family Chair Professor. He is currently the Wilmot J. Nicholson Family Chair Professor and the Chair of the Department of Computer Science and Engineering, Santa Clara University. He is/was also a Chair/Distinguished/Guest and Consulting Professor with several universities internationally. He has authored or coauthored more than 250 publications and seven adopted standard contributions. He has been granted 20 U.S. patents so far. He has delivered more than 120 invited colloquia worldwide. He is an IEEE Fellow due to his contributions to video coding algorithms and architectures. He is also an IET Fellow. He was a recipient of the IEEE ICCE Best Paper Award (First Place) and the Umedia Best/Excellent Paper Award (thrice). He received six awards from Santa Clara University, four at the university level and two at the school/college level. He has served as the General Chair/the Co-Chair for IEEE Hot Chips, VCVP (twice), IEEE ICME, Umedia (seven times), IEEE SiPS, and IEEE VCIP. He has also served as the Technical Program Co-Chair for IEEE ISCAS (twice), APSIPA ASC, IEEE APCCAS, IEEE SiPS (twice), DCV, and IEEE VCIP. He was the Technical Committee Chair of IEEE CASCOM TC and IEEE TCMM. He is the Chair of APSIPA U.S. Chapter. He has served as a Guest/Associate Editor for IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS, IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING, *JSPS* (Springer), and *MSSP* (Springer). He was named as an IEEE Distinguished Lecturer (twice) and an APSIPA Distinguished Lecturer. He was a Keynote Speaker of IEEE APCCAS, VCVP (twice), *JCPC*, IEEE ICAST, IEEE ICIEA, IET FC Umedia, IEEE Umedia, IEEE ICCIT, ICNLP/SSPS/CVPS, and Workshop at XUPT (twice).

• • •