RV-GAN: Recurrent GAN for Unconditional Video Generation

Sonam Gupta  Arti Keshari  Sukhendu Das
Visualization and Perception Lab, Department of Computer Science
Engineering, Indian Institute of Technology, Madras, India
csl18d005@cse.iitm.ac.in, cs19s008@cse.iitm.ac.in, sdas@iitm.ac.in

Abstract

Generative models aiming to generate content from noise have achieved high-fidelity synthesis for image data. However, obtaining comparable performance in the field of unconditional video generation still remains challenging. In this work, we propose a recurrent GAN architecture to model the high-dimensional video data distribution. Recurrent networks by design are able to generate complex, long sequences in an autoregressive fashion. However, the standard LSTM unit for videos (ConvLSTM) is not ideally suited for the task of unconditional video generation. Therefore, we propose a simple yet effective LSTM variant called as TransConv LSTM (TC-LSTM) by modulating the conventional ConvLSTM to have a transpose convolutional structure in input-to-state transitions. This enables the network to model both spatial and temporal relationships across layers simultaneously inside the TC-LSTM unit. TC-LSTM unit acts as a building block of our generator. Extensive quantitative and qualitative analysis shows that RV-GAN outperforms state-of-the-art methods by a significant margin on Moving MNIST, MUG, Weizmann and UCF101 datasets. Additionally, owing to the recurrent structure, our method is able to generate high-quality videos, up to 2 times longer (32 frames) than training videos at inference time. Further analysis confirms that the proposed architecture is generic and can be easily adapted to other tasks like class-conditional video synthesis and text-to-video synthesis.

1. Introduction

Video generation is a complex task as it requires to model spatial as well as temporal dynamics simultaneously. Video generation task provides a means for unsupervised feature representation learning from the vast amount of unlabeled data available on the internet. Study [41] shows an improvement in the performance of downstream tasks like action classification, by using the learned weights of the discriminator.

Following the success of Convolutional GANs in image generation literature, many of the recent works in unconditional video synthesis have proposed 3D-CNN GANs [30,41,42]. These methods suffer from the following drawbacks: at inference, 3D CNN models can generate reasonable quality video sequences for a fixed length video on which it has been trained. The number of generated frames can be increased by increasing the time dimension of the input noise, but this leads to deteriorated results (see figure 4). Thus, to faithfully generate longer videos, more convolutional layers are required at the training time which leads to an increase in the number of parameters.

A few other works [37,39] modelled video synthesis as a two-step process. In the first step, latent vectors corresponding to each frame of the video are generated. In the second step, each frame is generated by using a 2D convolutional image generator. Although the two-step process reduces the complexity of the task, it lacks in the following: (1) It struggles to maintain the same appearance throughout the video, (2) spatio-temporal consistency is not modeled properly as temporal relationship is learned only in latent space and there is a lack of information transfer between consecutive frames (see figures 3, 4).

Videos are a sequence of frames, where consecutive frames will have high correlation. To exploit this inherent property of videos, we propose to use recurrent GAN namely, RV-GAN consisting of a recurrent generator with stacked TC-LSTM layers and two CNN based discriminators. Because of this hybrid design, the generator enjoys the sequential modelling capabilities of RNNs whereas discriminators leverage CNN properties to achieve good classification performance. In other words, our model is able to extract both local and global dynamics of the video. The architecture of the generator is inspired by video prediction models solving the task of future frame generation [36,44–47]. However, these networks are not designed to work for unconditional video generation setting where the generator learns a mapping between low-dimensional latent space and high dimensional video space. Thus, we propose an effective modification to Convolutional LSTM (ConvL-
STM) [47] that facilitates the interaction between low dimensional features (from the previous layer) with high dimensional features (of previous time step) via transpose convolution. This enables the vertical flow of low-level information from latent space input to RGB frame output. This modified LSTM block is the core component of RV-GAN and is referred to as TransConv LSTM (TC-LSTM).

To summarize, following are the key contributions of our work:

1. We propose a recurrent GAN network, RV-GAN with recurrent generator and Convolutional Image and Video Discriminators. A novel TC-LSTM unit is used as the constituent unit for unconditional video generation.

2. We showcase the generalization capability of the proposed architecture on longer sequence generation (upto 32 frames) by training on shorter sequence. We further demonstrate the application of our model on two conditional video generation tasks, namely: class-conditional and text-to-video synthesis.

3. Extensive experimentation on benchmark datasets, both quantitatively and qualitatively along with the ablation studies demonstrates the superiority of our model over the state-of-the-art methods.

2. Related Work

Recently, the problem of video generation has attracted significant attention by research community. The task is challenging because it requires the network to generate realistic videos. Thus, to perform well, one of the critical requirements is to learn a good quality spatio-temporal representation from the training data. Two of the popular learning strategies used in the literature are GANs [9] and VAEs [25]. The complexity of the task varies based on the input that is used to condition the generation process. The methods that use some form of conditioning as input, for eg. first frame, few frames, etc are a bit easier to model than the one trained merely on latent noise. In this paper, we focus on the latter task, where no conditioning is provided as input i.e. unconditional video generation.

Conditional Video Generation One of the ways to reduce the complexity of modelling the high-dimensional video data is to provide additional information to the network such as class labels, captions, optical flow, few frames etc. This guides the network by revealing the spatial structure, content, and motion of the underlying video to be modelled. Few well-explored conditional video generation tasks are: Image to video generation [7,49], video-to-video translation [5,27] and future frame prediction [3,36,44–46]. The conditioning input can also be from other domains like text, audio etc. Some of these tasks are text-to-video generation [6,17,23], audio-to-video synthesis [12,19]. Although, our network does not require such conditioning, it can be generalized to these scenarios as illustrated in section 5.7.

Unconditional Video Generation refers to the generation of new video samples from training data distribution using latent noise vectors. To reduce the complexity of the task, several works try latent space decomposition into different video attributes, for example foreground, background, motion, content, objects, etc. VGAN [41] proposes a two-stream generator and a video discriminator for synthesizing the moving foreground and static background of the video separately. G3AN and G3AN++ [13,42] aim to disentangle content and motion and introduce multi-stream convolutional architecture, where various branches are responsible for modelling spatial, spatio-temporal and temporal features. V3GAN [22] decomposes a video into foreground, background and motion using three branch convolutional generator and proposes a feature level masking strategy and shuffling loss to improve the decomposition. In-MoDeGAN [43] assumes that the motion in a video can be represented in orthogonal basis vectors. They try to control video motion in latent space using a 3D Conv based generator and temporal pyramid discriminator.

Other direction of research is to utilize an image generator for generating each frame of the video. TGAN [30] proposes a dual generator approach where the temporal generator synthesizes the latent vectors corresponding to each frame of the video and an image generator maps these vectors to frames. MoCoGAN [39] replaces the temporal generator with a GRU for modelling motion in latent space. Similarly, [20,37] attempt to generate higher resolution videos ranging from 128x128 to 512x512 and longer video generation ranging from 32 to 64 frames. In [34], Skorokhodov et al. builds on top of styleGAN2 model, and proposes a continuous time generator and a single hyper-network based discriminator. In addition to these, [48] explore the use of transformer architecture for latent space, although high computational cost of transformer models for video data hinders the research in this direction. Limited works are present for higher resolution and longer video generation models due to unavailability of high-end GPU machines. We could not compare our model with these methods because of the high computational requirement for training these on our datasets.

RNN-GAN frameworks have been applied in diverse domains such as text [50], finance [33], sensors [2], music [28], medical [8] etc. However, most of these networks operate on same input and output size. Applying similar architecture in unconditional video generation domain is not straightforward due to the requirement of mapping low-dimensional latent noise vector into the high-dimensional video data. Our propose recurrent framework RV-GAN solves the task with a simplistic change in the existing ConvLSTM framework.
3. Preliminaries

3.1. Convolutional LSTM

LSTM [16] is a stable and powerful RNN module that has been proven to be successful for sequence modeling problems [4, 11, 29, 38]. It is able to model long-term dependencies by using cell state $C_t$ that can accumulate information. The information in the cell-state is updated with the help of self-parameterized controlling gates. An LSTM consists of three gates, namely, input, output and forget gates. On arrival of a new input, the previous cell state $C_{t-1}$ is updated depending upon which gate is activated. For instance, the information in the input will be accumulated if the input gate is activated. Similarly, the information in $C_{t-1}$ can be forgotten if forget gate is activated. Finally, the output gates control the fraction of information of the latest cell state $C_t$ that would be propagated to the final (hidden) state $H_t$. LSTM were designed to work with 1D data.

Convolutional LSTM (ConvLSTM) [47] is a variation of LSTM [16] that takes a 3D tensor as input, where two of the dimensions corresponds to height and width of the input data and third is the channel dimension. Thus, it can model the spatio-temporal features simultaneously by preserving the spatial information which would otherwise be lost in a standard LSTM. ConvLSTM determines the future state of a cell from the inputs and past states of its local neighbours, by using a convolutional structure in both the input-to-state and state-to-state transitions. In ConvLSTM, the input and output size are kept same by applying padding before convolution operation. The key equations of ConvLSTM are as follows:

$$g_t = \tanh (W_{xg} \odot X_t + W_{bg} \odot H_{t-1} + b_g)$$
$$i_t = \sigma (W_{xi} \odot X_t + W_{bi} \odot H_{t-1} + b_i)$$
$$f_t = \sigma (W_{xf} \odot X_t + W_{hf} \odot H_{t-1} + b_f)$$
$$C_t = f_t \odot C_{t-1} + i_t \odot g_t$$
$$o_t = \sigma (W_{xo} \odot X_t + W_{ho} \odot H_{t-1} + b_o)$$
$$H_t = o_t \odot \tanh (C_t)$$

Here, $\sigma$ represents the Sigmoid operation, $\odot$ represents the convolution operation and $\circ$ shows Hadamard product. $i_t, f_t, o_t, g_t$ are the input, forget, output and input-modulation gates respectively. $\{X_1, \ldots, X_t\}, \{C_1, \ldots, C_t\}, \{H_1, \ldots, H_t\}$ corresponds to the inputs, cell states and hidden states.

4. Proposed Method: RV-GAN

In this section, we first discuss two straightforward extensions of ConvLSTM networks for generative setting where the network learns to map an input noise vector to video. We then discuss the proposed RV-GAN architecture in detail.

4.1. Baselines

A stacked LSTM recurrent generator for video synthesis should satisfy following properties: (i) the spatial structure in the hidden states should be preserved. (ii) The size of the hidden states should increase as we move deeper across the layers in the network. In other words, more spatial context should be captured for higher resolution video synthesis. Keeping these in mind, we propose two plausible baselines using transpose convolution as illustrated in Figure 1. To satisfy property (ii), we introduce Transpose convolution or UpSampling block after each ConvLSTM layer.

However, we find that the model with upsampling layer introduces training instability and leads to mode collapse. On the other hand, the model with transpose convolutional layer generates samples where the appearance of the person gets distorted over time. This is because such a design fails to encode the change in spatial features between LSTM layers across time steps. To address above limitations, we propose to integrate the transpose convolution operation inside the LSTM unit. This leads to incorporation of learnable parameters of transpose convolution into the recurrent state transition over time. We call this modified LSTM unit as TransConv LSTM (TC-LSTM).

The overall architecture of our proposed RV-GAN model is shown in Figure 2(a). It consists of a recurrent generator and two convolutional discriminators, namely, image and
Let the training data consist of $N$ video samples where each video is represented as $V = \{X_1, X_2, ..., X_T\}$. Here $X_i, 1 \leq i \leq T$ represents the $i^{th}$ frame of the input video. Video generation problem can then be described as generating the output video $\hat{V} = \{\hat{X}_1, \hat{X}_2, ..., \hat{X}_T\}$ with $T$ frames from an input noise vector $z$, randomly sampled from Normal distribution. The core of the recurrent generator architecture is the proposed TC-LSTM unit as discussed next.

### 4.2. TransConv-LSTM (TC-LSTM)

A single unit of the proposed TC-LSTM is illustrated in Figure 2 (right). It takes 3 inputs: $X_t$, the input noise vector or hidden state from previous TC-LSTM layer; $H_{t-1}$: the hidden state from previous time step; and $C_{t-1}$: cell state from previous time step. We use transpose convolution between input-to-state transitions in the input gate $i_t$, input-modulation gate $g_t$, forget gate $f_t$ and output gate $o_t$. Thus, the computation of upsampled feature maps rely on two factors, cell states and hidden state of previous timestamps and input from previous layer. This design enhances the modeling capability of short-term spatio-temporal dynamics of the network. The key equations of TC-LSTM are given in equation 2, where $\circ_T$ denotes the transpose convolution, $\circ$ denotes the convolution and $\odot$ denotes the Hadamard product.

\begin{align}
  g_t &= \tanh (W_{xg} @_T X_t + W_{hg} @ H_{t-1} + b_g) \\
  i_t &= \sigma (W_{xi} @_T X_t + W_{hi} @ H_{t-1} + b_i) \\
  f_t &= \sigma (W_{xf} @_T X_t + W_{hf} @ H_{t-1} + b_f) \\
  C_t &= f_t \odot C_{t-1} + i_t \odot g_t \\
  o_t &= \sigma (W_{xO} @_T X_t + W_{ho} @ H_{t-1} + b_o) \\
  H_t &= o_t \odot \tanh (C_t)
\end{align} 

(2)

### 4.3. Generator Architecture

Choice of kernel size, stride and padding in transpose convolution operations decide the factor with which the input will be scaled up spatially. Thus, the same 5 layer network can be used for generating higher resolution video as well. Hidden states ($H_t$) and cell states ($C_t$) of the TC-LSTM are initialized to zero which corresponds to no past memory.

Like a standard LSTM, TC-LSTM can also be used as a building block for complex architectures. For the unconditional video generation task, we use the recurrent generator architecture as illustrated in Figure 2. It consists of a stack of five TC-LSTM layers. We use a kernel size of 4 with a stride of 2 for input-to-state transition in TC-LSTM. The last TC-LSTM layer outputs 64 channels. Thus, a 2D convolution layer is used to obtain the final frame with 3 channels (RGB). The input at each time step is the same noise vector randomly sampled from the latent space.
4.4. Discriminator Architecture

Similar to MoCoGAN [39], we use two separate discriminators, Image Discriminator $D_I$ and Video Discriminator $D_V$. $D_V$ has 5 convolutional 3D layers modeled as conv(2+1)D block. It accepts the entire video as input. $D_I$ has 5 convolutional 2D layers and accepts a randomly sampled frame of the video as input. Image discriminator helps to maintain the quality of the individual frames. Video discriminator helps to improve the spatio-temporal consistency of the generated output.

4.5. Loss Functions

We use adversarial loss [9] to train the proposed RV-GAN network. Let the recurrent generator be denoted by $G$. The generator tries to generate realistic videos so that it can fool the discriminator whereas the discriminator tries to distinguish the generated videos from real ones by classifying them as fake or real. The loss functions for generator and discriminators are defined as follows:

$$
\min_G \max_{D_I, D_V} L_I(G, D_I) + L_V(G, D_V)
$$

where,

$$
L_I(G, D_I) = E_{s(v) \sim p_{\text{data}}} \left[ \log(D_I(s(v))) \right] + E_{z \sim p_z} \left[ \log(1 - D_I(s(G(z)))) \right]
$$

$$
L_V(G, D_V) = E_{v \sim p_{\text{data}}} \left[ \log(D_V(v)) \right] + E_{z \sim p_z} \left[ \log(1 - D_V(G(z))) \right]
$$

5. Experiments and Results

5.1. Datasets and Setup

We trained our model on four datasets as described below.

**Moving MNIST** is a synthetic dataset of handwritten digits consisting of 10,000 videos. Each video sequence contains 2 digits moving independently across the frame.

**Weizmann** [10] action dataset consists of 93 videos of 9 people performing 10 different actions, such as running, bending, jumping. The video frames have been flipped for augmentation purposes.

**UCF101** [35] dataset contains 13,320 real-world video clips, categorized in 101 action classes. We rescale the frames to 85x64 followed by center-cropping to 64x64 for Weizmann and UCF101 datasets, similar to [30].

**MUG** [1] dataset contains 908 video sequences of 52 individuals performing facial expressions. We chose only 6 expressions- fear, anger, sadness, disgust, happiness and surprise. Faces are cropped in each frame on the basis of landmarks and then resized to 64x64.

For all our experiments, we randomly chose 16 frames with step sizes 1 and 2. The input noise ($z$) dimension is set to 128. The output videos contain 16 frames at a resolution of 64x64. A batch size of 16 is used. The learning rate for generator and discriminators is set to $10^{-4}$. The networks are trained using Adam optimizer [24] with $b_1 = 0.5$ and $b_2 = 0.999$. Source code will be made public.

5.2. Evaluation metrics

We use Fréchet Inception Distance (FID) [15] as the evaluation metric to measure the quality of the generated videos. FID metric is the squared Wasserstein distance between two multidimensional Normal distributions ($\mathcal{N}(\mu, \Sigma)$). In the case of video data, the feature embedding of generated and real video samples are calculated us-
also outperforms the baseline, suggesting that inclusion of
able to learn the spatio-temporal correlation well. RV-GAN
diverse and realistic. Table 1 indicates that our model is
enhancement of the metric. The high IS values obtained
sufficient number of jack action videos. For the MUG
dataset, our model is able to generate realistic samples that
foreground-background decomposition. MoCoGAN per-
completely in different directions, the network fails to learn
visual quality and temporal consistency. We note that for
Moving MNIST, we also compare our model with MoCoGAN, G3AN and
SVGAN in Table 2 using FVD metric. It can be seen that the de-
composition of video or latent space need not always result
in reduced complexity of the task. Recently, self-supervised
video GAN (SVGAN) [18] uses FVD for evaluation, hence we
also compare our model with MoCoGAN, G3AN and
SVGAN in Table 2 using FVD metric. It can be seen that
our method consistently achieves the lowest FVD. This fur-
ther confirms that RV-GAN is able to learn training data
transpose convolution inside the LSTM allows for improved
visual quality and temporal consistency. We note that for
Moving MNIST dataset, the FID values of V3GAN is the
highest. This is because, for such fine digits moving inde-
dependently in different directions, the network fails to learn
foreground-background decomposition. MoCoGAN
performs poorly on Moving MNIST. This suggests that the de-
composition of video or latent space need not always result
in reduced complexity of the task. Recently, self-supervised
video GAN (SVGAN) [18] uses FVD for evaluation, hence we
also compare our model with MoCoGAN, G3AN and
SVGAN in Table 2 using FVD metric. It can be seen that
our method consistently achieves the lowest FVD. This fur-
ther confirms that RV-GAN is able to learn training data
transposition of the feature distribution. FVD is then computed as :
\begin{equation}
FVD = \left(\frac{1}{2}\right) \left(\|\mu_g - \mu_r\|^2 + \text{tr}(\Sigma_g + \Sigma_r - 2\Sigma_g\Sigma_r)^{1/2}\right) \end{equation}
where \(\mu_r\) and \(\Sigma_r\) correspond to the mean and covariance matrix of
the real distribution and similarly \(\mu_g\) and \(\Sigma_g\) correspond to the
mean and covariance of the generated distribution. Lower
values of FID metric correspond to better quality of the gen-
erated videos.

We also report Fréchet Video Distance (FVD) [40]
metric to compare our method with the recently proposed
self-supervised video GAN [18]. FVD uses similar set-
tings as FID except for the 3D-CNN architecture. FVD
uses the I3D [4] model for embedding whereas FID uses the

For UCF101 dataset, we also report Inception score
(IS). [32]. IS is the KL divergence between class
conditional probability distribution \((p(y|x))\) and marginal
\((p(y))\) probability distribution which can be calculated as:
\begin{equation}
IS(G) = \exp\left(E_{x\sim G}KL(p(y|x)||p(y))\right) \end{equation}
High values of IS indicate better diversity and quality of the generated samples.

5.3. Quantitative Evaluation

We compare our method with state-of-the-art methods
[22, 25, 30, 39, 41] and the proposed baseline in section
4.1 using FID metric. Quantitative comparison of the four
datasets is reported in Table 1. To compute the FID value,
we have generated 5000 samples using the trained model. It
can be seen that our method RV-GAN outperforms almost
all other methods except V3GAN on UCF101. This might
be because V3GAN generates background of the frame sep-
ately which is same throughout the video, which helps in
enhancement of the metric. The high IS values obtained
on UCF101 dataset implies that the generated samples are
diverse and realistic. Table 1 indicates that our model is
able to learn the spatio-temporal correlation well. RV-GAN
also outperforms the baseline, suggesting that inclusion of

<table>
<thead>
<tr>
<th>Method</th>
<th>MMNIST</th>
<th>WZM</th>
<th>MUG</th>
<th>UCF101</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FID↓</td>
<td>FID↓</td>
<td>FID↓</td>
<td>FID↓</td>
</tr>
<tr>
<td>VGAN [41]</td>
<td>-</td>
<td>158.1</td>
<td>160.7</td>
<td>115.1</td>
</tr>
<tr>
<td>TGAN [30]</td>
<td>-</td>
<td>99.8</td>
<td>97.1</td>
<td>110.5</td>
</tr>
<tr>
<td>MoCoGAN [39]</td>
<td>44.1</td>
<td>92.2</td>
<td>87.1</td>
<td>104.1</td>
</tr>
<tr>
<td>G3AN [42]</td>
<td>16.1</td>
<td>86.0</td>
<td>67.1</td>
<td>86.7</td>
</tr>
<tr>
<td>V3GAN [22]</td>
<td>59.9</td>
<td>62.6</td>
<td>53.5</td>
<td>80.2</td>
</tr>
<tr>
<td>Baseline</td>
<td>31.3</td>
<td>63.0</td>
<td>MC</td>
<td>94.6</td>
</tr>
<tr>
<td>RV-GAN (Ours)</td>
<td><strong>14.8</strong></td>
<td><strong>57.3</strong></td>
<td><strong>23.3</strong></td>
<td>82.3</td>
</tr>
</tbody>
</table>

Table 1. Quantitative Comparison of our method with SOTA methods using FID (lower the better) on Moving-MNIST, Weizmann, MUG and UCF101 datasets. MC represents the mode collapse.

<table>
<thead>
<tr>
<th>Method</th>
<th>Weizmann</th>
<th>MUG</th>
<th>UCF101</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FID↓</td>
<td>FID↓</td>
<td>FID↓</td>
</tr>
<tr>
<td>MoCoGAN [39]</td>
<td>194.34</td>
<td>102.2</td>
<td>869.41</td>
</tr>
<tr>
<td>G3AN [42]</td>
<td>117.69</td>
<td>89.73</td>
<td>687.67</td>
</tr>
<tr>
<td>SVGAN [18]</td>
<td>105.51</td>
<td>67.62</td>
<td>643.55</td>
</tr>
<tr>
<td>Ours</td>
<td><strong>91.49</strong></td>
<td><strong>49.2</strong></td>
<td><strong>623.90</strong></td>
</tr>
</tbody>
</table>

Table 2. Comparison of the performance of our method with SOTA methods using FVD metric.

<table>
<thead>
<tr>
<th>Architecture</th>
<th>WZM</th>
<th>MUG</th>
<th>UCF101</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FID↓</td>
<td>FID↓</td>
<td>FID↓</td>
</tr>
<tr>
<td>3 - Layer</td>
<td>63.64</td>
<td>52.70</td>
<td>101.1</td>
</tr>
<tr>
<td>4 - Layer</td>
<td>61.08</td>
<td>24.78</td>
<td>88.7</td>
</tr>
<tr>
<td>5 - Layer (Ours)</td>
<td><strong>57.32</strong></td>
<td><strong>23.31</strong></td>
<td><strong>82.3</strong></td>
</tr>
</tbody>
</table>

Table 3. Ablation with Number of TC-LSTM Layers in generator.
dataset consists of less number of videos, we perform linear interpolation similar to [31] to verify whether the network is memorizing the dataset. Qualitative results for this are available in supplementary material (SM). It can be seen that there is a smooth transition in generated videos with change in input noise. Thus, we can conclude that the network is not memorizing the dataset.

5.5. Ablation Study

Ablation of Architecture: We perform two ablation studies to assess the impact of each component of the proposed RV-GAN. Since there are no recurrent GANs for the task of unconditional video generation (to the best of our knowledge), we build a simple and reasonable baseline as specified in section 4.1. We use the same architecture as that of ours, Figure 2, but replace each TC-LSTM with a combination of ConvLSTM [47] and transpose convolutional layer in the generator as shown in Figure 1. The discriminator remains the same. This baseline emphasizes the need and significance of the proposed TC-LSTM.

Apart from this baseline, we also use the upsampling layer (using bilinear interpolation) after ConvLSTM layer, but this configuration proved to be unstable leading to mode collapse for all datasets. However, we did not come across mode collapse for TC-LSTM which shows that our model is stable. Results in Table 1 suggest that TC-LSTM have contributed to the improvement in the quality of the generated videos.

Study the number of layers: We further test the effect of the number of TC-LSTM layers using three variants as shown in Table 3. The three-layer variant contains three TC-LSTM layers with 512, 256 and 128 channels in hidden states, respectively. The four-layer variant has four TC-LSTM layers with 512, 256, 128 and 64 channels in hidden states, respectively, with a kernel size of 4x4. It can be seen from the Table 3 that our method consistently outperforms the present state-of-the-art methods even with a lesser number of layers. This also proves that our model is stable and maintains the quality of generated videos with a lower number of parameters. As expected, the performance of the network improves when more layers are used.

5.6. Longer Sequence Generation

One of the advantages of using a recurrent generator over 3D-CNN is that we can synthesize a longer sequence during inference time. This is useful when we have limited computational resources for training. Thus, we analyze the performance of our method for the generation of longer video sequences, both quantitatively in Table 4 and qualitatively in Figure 4. We used the model trained on sequences of 16 frames to generate videos for 24 and 32 time steps. We further compare it with MoCoGAN which uses the recurrent component to generate the motion noise vector and G3AN [42] which is a fully convolutional GAN. In MoCoGAN and G3AN a consistent discontinuity pattern is observed around 15th to 17th frame on Weizmann as well as MUG datasets (see rows 1,3 and 2,4 in Figure 4). However, we did not encounter such a pattern for our model. This shows that using a recurrent network for modelling motion only in latent space is not sufficient. More results for longer videos are available in SM.
5.7. Applications to Conditional Video Generation

We show that our recurrent model can easily be extended to other conditional video generation tasks. In particular, we trained a slightly modified version of RV-GAN for video generation conditioned on class label (also referred to as class conditional video generation) and text-to-video generation as described below. For both of these tasks, the experiments are performed on Weizmann Action dataset.

**Class Conditional Video Generation:** The task is to take a class label and random noise vector as input and generate a fake video which corresponds to the given class label. We concatenate one hot vector embedding of action label with input noise along the channel dimension and pass it through the generator. To condition both image and video discriminator, a repetition of one hot vector is concatenated as the fourth channel (after 3 RGB channels) in each frame of the video. The results can be visualized in Figure 5.

**Text to Video Generation:** Here, the conditioning factor for video generation is a caption (text). Since the text description of a video belongs to an entirely different modality, hence it is crucial to have sufficient number of data mapping from text to video. Text gives marginal information to create a video, which makes it a challenging problem. We chose to use an encoder-decoder model for this task. Pre-trained skip-thought [26] embedding is used to encode the caption and RV-GAN generator is used as a decoder. We choose five action categories of Weizmann action dataset namely: running, walking, side-walking, skipping and jumping because these classes have maximal motion along with an associated direction of movement. Some of the generated examples are shown in Figure 6. We can notice that the results are semantically correct but there is scope of improvement in the quality of the video which is hindered due to unavailability of the large size dataset.

6. Conclusion

In this work, we present a novel recurrent framework for video synthesis. We extend ConvLSTM to build our novel TC-LSTM. Ablation analysis confirms that TC-LSTM is a stable LSTM unit for building complex generative architectures for spatio-temporal data. By incorporating a hybrid generator-discriminator architecture with adversarial learning, our framework is able to achieve results superior to the state-of-the-art methods on benchmark datasets. We believe that our work will open avenues for exploring even better recurrent architectures for the unconditional video generation task.

<table>
<thead>
<tr>
<th>Method</th>
<th>WZM</th>
<th>MUG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>24 fr</td>
<td>32 fr</td>
</tr>
<tr>
<td></td>
<td>FID↑</td>
<td>FID↑</td>
</tr>
<tr>
<td>G3AN [42]</td>
<td>97.89</td>
<td>118.84</td>
</tr>
<tr>
<td>MoCoGAN [39]</td>
<td>101.05</td>
<td>103.25</td>
</tr>
<tr>
<td>Ours</td>
<td><strong>72.53</strong></td>
<td><strong>77.07</strong></td>
</tr>
</tbody>
</table>

Table 4. Quantitative comparison with SOTA methods for longer video generation at inference time. fr represents frame.
References


2032


[32] Luca Simonetto. Generating spiking time series with generative adversarial networks: an application on banking transactions. MS thesis - Univ. of Amsterdam, 2018. 2


[34] Yu Tian, Jian Ren, Menglei Chai, Kyle Olszewski, Xi Peng, Dimitris N. Metaxas, and S. Tulyakov. A good image generator is what you need for high-resolution video synthesis. ArXiv, abs/2104.14683, 2021. 2


[38] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. Mocogan: Decomposing motion and content for video generation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1526–1535, 2018. 1, 2, 5, 6, 8

