Learning to Generate Time-Lapse Videos Using Multi-Stage Dynamic Generative Adversarial Networks

Wei Xiong[†‡*] Wenhan Luo[†] Lin Ma[†] Wei Liu[†] Jiebo Luo[‡†]
[†]Tencent AI Lab [‡]University of Rochester
[‡]{wxiong5,jluo}@cs.rochester.edu [†]{whluo.china,forest.linma}@gmail.com [†]wliu@ee.columbia.edu

Abstract

Taking a photo outside, can we predict the immediate future, e.g., how would the cloud move in the sky? We address this problem by presenting a generative adversarial network (GAN) based two-stage approach to generating realistic time-lapse videos of high resolution. Given the first frame, our model learns to generate long-term future frames. The first stage generates videos of realistic contents for each frame. The second stage refines the generated video from the first stage by enforcing it to be closer to real videos with regard to motion dynamics. To further encourage vivid motion in the final generated video, Gram matrix is employed to model the motion more precisely. We build a large scale time-lapse dataset, and test our approach on this new dataset. Using our model, we are able to generate realistic videos of up to 128×128 *resolution for 32 frames. Quantitative and qualitative experiment results demonstrate the superiority of our model over the state-of-the-art models.*

1. Introduction

Humans can often estimate fairly well what will happen in the immediate future given the current scene. However, for vision systems, predicting the future states is still a challenging task. The problem of future prediction or video synthesis has drawn more and more attention in recent years since it is critical for various kinds of applications, such as action recognition [22], video understanding [31], and video captioning [35]. The goal of video prediction in this paper is to generate realistic, long-term, and high-quality future frames given one starting frame. Achieving such a goal is difficult, as it is challenging to model the multi-modality and uncertainty in generating both the content and motion in future frames.

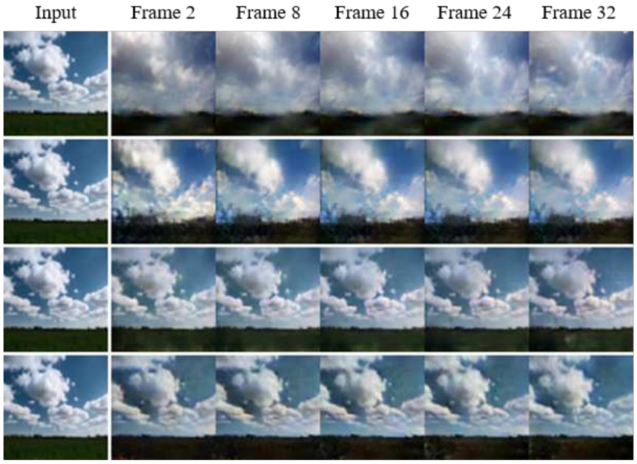

Figure 1. From top to bottom: example frames of generated videos by VGAN [28], RNN-GAN [37], the first stage of our model, and the second stage of our model, respectively. The contents generated by our model (the third and fourth rows) are visually more realistic. The left column is the input starting frame.

In terms of content generation, the main problem is to define what to learn. Generating future on the basis of only one static image encounters inherent uncertainty of the future, which has been illustrated in [29]. Since there can be multiple possibilities for reasonable future scenes following the first frame, the objective function is difficult to define. Generating future frames by simply learning to reconstruct the real video can lead to unrealistic results [28, 16]. Several models including [27] and [28] are proposed to address this problem based on generative adversarial networks [5]. For example, 3D convolution is incorporated in an adversarial network to model the transformation from an image to a video in [28]. Their model produces plausible futures given the first frame. However, the generated video tends to be blurry and lose content details, which degrades the reality of generated videos. A possible cause is that the vanilla encoder-decoder structure in the generator fails to preserve all the indispensable details of the content.

Regarding motion transformation, the main challenge is to drive the given frame to transform realistically over time.

[*] This work was primarily done while Wei Xiong was a Research Intern with Tencent AI Lab.

Some prior work has investigated this problem. Zhou and Berg [37] use an RNN to model the temporal transformations. They are able to generate a few types of motion patterns, but not realistic enough. The reason may be that, each future frame is based on the state of previous frames, so the error accumulates and the motion distorts over time. The information loss and error accumulation during the sequence generation hinder the success of future prediction.

The performance of the prior models indicates that it is nontrivial to generate videos with both realistic contents in each frame and vivid motion dynamics across frames with a single model at the same time. One reason may be that the representation capacity of a single model is limited in satisfying two objectives that may contradict each other. To this end, we divide the modeling of video generation into content and motion modeling, and propose a Multi-stage Dynamic Generative Adversarial Network (MD-GAN) model to produce realistic future videos. There are two stages in our approach. The first stage aims at generating future frames with content details as realistic as possible given an input frame. The second stage specifically deals with motion modeling, *i.e.*, to make the movement of objects between adjacent frames more vivid, while keeping the content realistic.

To be more specific, we develop a generative adversarial network called Base-Net to generate contents in the first stage. Both the generator and the discriminator are composed of 3D convolutions and deconvolutions to model temporal and spatial patterns. The adversarial loss of this stage encourages the generator to produce videos of similar distributions to real ones. In order to preserve more content details, we use a 3D U-net [21] like architecture in the generator instead of the vanilla encoder-decoder structure. Skip connections [6] are used to link the corresponding feature maps in the encoder and decoder so that the decoder can reuse features in the encoder, thus reducing the information loss. In this way, the model can generate better content details in each future frame, which are visually more pleasing than those produced by the vanilla encoder-decoder architecture such as the model in [28].

The Base-Net can generate frames with concrete details, but may not be capable of modeling the motion transformations across frames. To generate future frames with vivid motion, the second stage MD-GAN takes the output of the first stage as input, and refines the temporal transformation with another generative adversarial network while preserving the realistic content details, which we call Refine-Net. We propose an adversarial ranking loss to train this network so as to encourage the generated video to be closer to the real one while being further away from the input video (from stage I) regarding motion. To this end, we introduce the Gram matrix [4] to model the dynamic transformations among consecutive frames. We present a few example frames generated by the conventional methods and our method in Fig. 1. The image frames generated by our model are sharper than the state-of-the-art and are visually almost as realistic as the real ones.

We build a large scale time-lapse video dataset called Sky Scene to evaluate the models for future prediction. Our dataset includes daytime, nightfall, starry sky, and aurora scenes. MD-GAN is trained on this dataset and predicts future frames given a static image of sky scene. We are able to produce 128×128 realistic videos, whose resolution is much higher than that of the state-of-the-art models. Unlike some prior work which generates merely one frame at a time, our model generates 32 future frames by a single pass, further preventing error accumulation and information loss.

Our key contributions are as follows:

1. We build a large scale time-lapse video dataset, which contains high-resolution dynamic videos of sky scenes.

2. We propose a Multi-stage Dynamic Generative Adversarial Network (MD-GAN), which can effectively capture the spatial and temporal transformations, thus generating realistic time-lapse future frames up to 128×128 resolution given only one starting frame.

3. We introduce the Gram matrix for motion modeling and propose an adversarial ranking loss to mimic motions of real-world videos, which refines motion dynamics of preliminary outputs in the first stage and forces the model to produce more realistic and higher-quality future frames.

2. Related Work

Generative Adversarial Networks. A generative adversarial network (GAN)[5, 1, 32, 30] is composed of a generator and a discriminator. The generator tries to fool the discriminator by producing samples similar to real ones, while the discriminator is trained to distinguish the generated samples from the real ones. GANs have been successfully applied to image generation. In the seminal paper [5], models trained on the MNIST dataset and the Toronto Face Database (TFD), respectively, generate images of digits and faces with high likelihood. Relying only on random noise, GAN cannot control the mode of the generated samples, thus conditional GAN [17] is proposed. Images of digits conditioned on class labels and captions conditioned on image features are generated. Many subsequent works are variants of conditional GAN, including image to image translation [9, 38], text to image translation [20] and super-resolution [13]. Our model is also a GAN conditioned on a starting image to generate a video.

Inspired by the coarse-to-fine strategy, multi-stack methods such as StackGAN [36], LAPGAN [2] have been proposed to first generate coarse images and then refine them to finer images. Our model also employs this strategy to stack GANs in two stages. However, instead of refining the pixel-level details in each frame, the second stage focuses on improving motion dynamics across frames.

Video Generation. Based on conditional VAE [12], Xue et al. [34] propose a cross convolutional network to model layered motion, which applies learned kernels to image features encoded in a multi-scale image encoder. The output difference image is added to the current frame to produce the next frame. [16] is one of the earliest work that adopts generative adversarial networks to produce future frames. It uses the adversarial loss and an image gradient difference loss instead of the standard Mean Square Error to avoid blurry results. In [28], a two-stream CNN, one for foreground and the other one for background, is proposed for video generation. Combining the dynamic foreground stream and the static background stream, the generated video looks real. In the follow-up work [29], Vondrick and Torralba formulate the future prediction task as transforming pixels in the past to future. Based on large scale unlabeled video data, a CNN model is trained with adversarial learning. Content and motion are decomposed and encoded separately by multi-scale residual blocks, and then combined and decoded to generate plausible videos on both the KTH and the Weizmann datasets [26]. A similar idea is presented in [25]. To generate long-term future frames, Villegas et al. [27] estimate high-level structure (human body pose), and learn a LSTM and an analogy-based encoder-decoder CNN to generate future frames based on the current frame and the estimated high-level structure.

The closest work to ours is [37], which also generates time-lapse videos. However, there are important differences between their work and ours. First, our method is based on 3D convolution while a recurrent neural network is employed in [37] to recursively generate future frames, which is prone to error accumulation. Second, as modeling motion is indispensable for video generation, we explicitly model motion by introducing the Gram matrix. Finally, we generate high-resolution (128×128) videos of dynamic scenes, while the generated videos in [37] are simple (usually with clean background) and of resolution 64×64.

3. Our Approach

3.1. Overview

The proposed MD-GAN takes a single RGB image as input and attempts to predict future frames that are as realistic as possible. This task is accomplished in two stages in a coarse-to-fine manner: 1) Content generation by Base-Net in Stage I. Given an input image $\mathbf{x}$, the model generates a video $\mathbf{Y}_1$ of T frames (including the starting frame, *i.e.*, the input image). The Base-Net ensures that each produced frame in $\mathbf{Y}_1$ looks like a real natural image. Besides, $\mathbf{Y}_1$ also serves as a coarse estimation of the ground-truth $\mathbf{Y}$ regarding motion. 2) Motion generation by Refine-Net in Stage II. The Refine-Net makes efforts to refine $\mathbf{Y}_1$ with vivid motion dynamics, and produces a more vivid video $\mathbf{Y}_2$ as the final prediction. The discriminator D_2 of the Refine-Net takes three inputs, the output video $\mathbf{Y}_1$ of the Base-Net, the fake video $\mathbf{Y}_2$ produced by the generator of the Refine-Net and the real video $\mathbf{Y}$. We define an adversarial ranking loss to encourage the final video $\mathbf{Y}_2$ to be closer to the real video and further away from video $\mathbf{Y}_1$. Note that on each stage, we follow the setting in Pix2Pix [9] and do not incorporate any random noise. The overall architecture of our model is plotted in Fig. 2.

3.2. Stage I: Base-Net

As shown in Fig. 2, the Base-Net is a generative adversarial network composed of a generator G_1 and a discriminator D_1. Given an image $\mathbf{x} \in \mathbb{R}^{3\times H\times W}$ as a starting frame, we duplicate it T times, obtaining a static video $\mathbf{X} \in \mathbb{R}^{3\times T\times H\times W}$ [1]. By forwarding $\mathbf{X}$ through layers of 3D convolutions and 3D deconvolutions, the generator G_1 outputs a video $\mathbf{Y}_1 \in \mathbb{R}^{3\times T\times H\times W}$ of T frames, *i.e.*, $\mathbf{Y}_1 = G_1(\mathbf{X})$.

For generator G_1, we adopt an encoder-decoder architecture, which is also employed in [19] and [28]. However, such a vanilla encoder-decoder architecture encounters problems in generating decent results as the features from the encoder may not be fully exploited. Therefore, we utilize a 3D U-net like architecture [21] instead so that features in the encoder can be fully made use of to generate $\mathbf{Y}_1$. This U-net architecture is implemented by introducing skip connections between the feature maps of the encoder and the decoder, as shown in Fig. 2. The skip connections build information highways between the features in the bottom and top layers, so that features can be reused. In this way, the generated video is more likely to contain rich content details. This may seem like a simple modification, yet it plays a key role in improving the quality of videos.

The discriminator D_1 then takes video $\mathbf{Y}_1$ and the real video $\mathbf{Y}$ as input and tries to distinguish them. $\mathbf{x}$ is the first frame of $\mathbf{Y}$. D_1 shares the same architecture as the encoder part of G_1, except that the final layer is a single node with a sigmoid activation function.

To train our GAN-based model, the adversarial loss of the Base-Net is defined as:

$$\begin{aligned}\mathcal{L}_{adv} = \min_{G_1}\max_{D_1} & \mathbb{E}\left[\log D_1\left(\mathbf{Y}\right)\right] + \\ & \mathbb{E}\left[\log\left(1 - D_1\left(G_1\left(\mathbf{X}\right)\right)\right)\right] .\end{aligned} \tag{1}$$

Prior work based on conditional GAN discovers that combining the adversarial loss with an L_1 or L_2 loss [9] in the pixel space will benefit the performance. Hence, we define a content loss function as a complement to the adversarial loss, to further ensure that the content of the generated

[1] In the generator, we can also use a 2D CNN to encode an image, but we duplicate the input image to a video to better fit our 3D U-net like architecture of G_1.

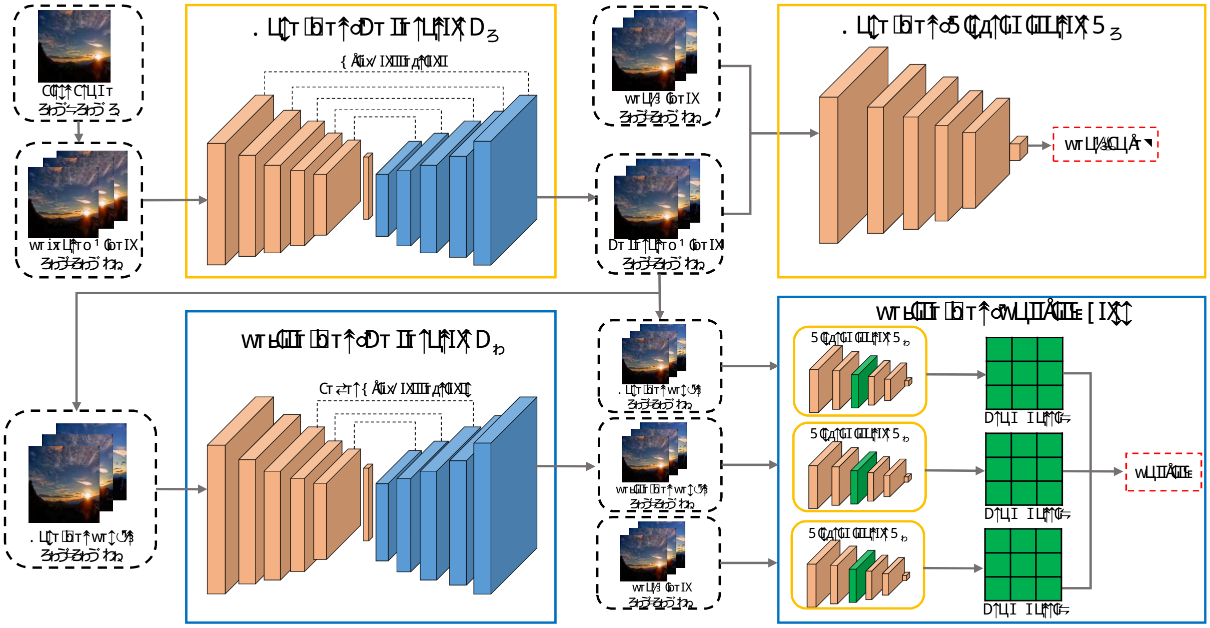

Figure 2. The overall architecture of our MD-GAN model. The input image is first duplicated to 32 frames as input to generator G_1 of the Base-Net, which produces a video $\mathbf{Y}_1$. Discriminator D_1 then distinguishes the real video $\mathbf{Y}$ from $\mathbf{Y}_1$. Following the Base-Net, the Refine-Net takes the generated video of G_1 as the input and produces a more realistic video $\mathbf{Y}_2$. Discriminator D_2 is updated with an adversarial ranking loss to push $\mathbf{Y}_2$ (the result of Refine-Net) closer to real videos.

video follows similar patterns to the content of real-world videos. As pointed out in [9], L_1 distance usually results in sharper outputs than those of L_2 distance. Recently, instead of measuring the similarity of images in the pixel space, perceptual loss [10] is introduced in some GAN-based approaches to model the distance between high-level feature representations. These features are extracted from a well-trained CNN model and previous experiments suggest they capture semantics of visual contents [13]. Although the perceptual loss performs well in combination with GANs [13, 14] on some tasks, it typically requires features to be extracted from a pretrained deep neural network, which is both time and space consuming. In addition, we observe in experiments that directly combining the adversarial loss and an L_1 loss that minimizes the distance between the generated video and the ground-truth video in the pixel space leads to satisfactory performance. Thus, we define our content loss as

$$\mathcal{L}_{con}\left(G_1\right) = \left\|\mathbf{Y} - G_1\left(\mathbf{X}\right)\right\|_1 . \tag{2}$$

The final objective of our Base-Net in Stage I is

$$\mathcal{L}_{stage1} = \mathcal{L}_{adv} + \mathcal{L}_{con} . \tag{3}$$

The adversarial training allows the Base-Net to produce videos with realistic content details. However, as the learning capacity of GAN is limited considering the uncertainty of the future, one single GAN model may not be able to capture the correct motion patterns in the real-world videos. As a consequence, the motion dynamics of the generated videos may not be realistic enough. To tackle this problem, we further process the output of Stage I by another GAN model called Refine-Net in Stage II, to compensate it for vivid motion dynamics, and generate more realistic videos.

3.3. Stage II: Refine-Net

Inputting video $\mathbf{Y}_1$ from Stage I, our Refine-Net improves the quality of the generated video $\mathbf{Y}_2$ regarding motion to fool human eyes in telling which one is real against the ground-truth video $\mathbf{Y}$.

Generator G_2 of the Refine-Net is similar to G_1 in the Base-Net. When training the model, we find it difficult to generate vivid motion while retaining realistic content details using skip connections. In other words, skip connections mainly contribute to content generation, but may not be helpful for motion generation. Thus, we remove a few skip connections from G_2, as illustrated in Fig. 2. The discriminator D_2 of the Refine-Net is also a CNN with 3D convolutions and shares the same structure as D_1 in the Base-Net.

We adopt the adversarial training to update G_2 and D_2. However, naively employing the vanilla adversarial loss can lead to an identity mapping since the input $\mathbf{Y}_1$ of G_2 is an optimal result of *i.e.* G_1, which has a very similar structure as G_2. As long as G_2 learns an identity mapping, the output $\mathbf{Y}_2$ would not be improved. To force the network to learn effective temporal transformations, we propose an adversarial ranking loss to drive the network to generate videos which are closer to real-world videos while further away from the input video ($\mathbf{Y}_1$ from Stage I). The ranking loss is defined as $\mathcal{L}_{rank}\left(\mathbf{Y}_1, \mathbf{Y}_2, \mathbf{Y}\right)$, which will be detailed later, with regard to the input $\mathbf{Y}_1$, output $\mathbf{Y}_2$ and the ground-truth video $\mathbf{Y}$. To construct such a ranking loss, we should take the advantage of effective features that can well represent the dynamics across frames. Based on such feature representations, distances between videos can be conveniently calculated.

We employ the Gram matrix [4] as the motion feature representation to assist G_2 to learn dynamics across video frames. Given an input video, we first extract features of the video with discriminator D_2. Then the Gram matrix is calculated across the frames using these features such that it incorporates rich temporal information.

Specifically, given an input video $\mathbf{Y}$, suppose that the output of the *l-th* convolutional layer in D_2 is $\mathbf{H}_{\mathbf{Y}}^{l} \in \mathbb{R}^{N \times C_l \times T_l \times H_l \times W_l}$, where (N, C_l, T_l, H_l, W_l) are the batch size, number of filters, length of the time dimension, height and width of the feature maps, respectively. We reshape $\mathbf{H}_{\mathbf{Y}}^{l}$ to $\hat{\mathbf{H}}_{\mathbf{Y}}^{l} \in \mathbb{R}^{N \times M_l \times S_l}$, where $M_l = C_l \times T_l$ and $S_l = H_l \times W_l$. Then we calculate the Gram matrix $g(\mathbf{Y}; l)$ of the *n-th* layer as follows:

$$g\left(\mathbf{Y}; l\right) = \frac{1}{M_l \times S_l} \sum\nolimits_{n=1}^{N} \hat{\mathbf{H}}_{\mathbf{Y}}^{l,n} \left(\hat{\mathbf{H}}_{\mathbf{Y}}^{l,n}\right)^{T} , \qquad (4)$$

where $\hat{\mathbf{H}}_{\mathbf{Y}}^{l,n}$ is the *n-th* sample of $\hat{\mathbf{H}}_{\mathbf{Y}}^{l}$. $g\left(\mathbf{Y}; l\right)$ calculates the covariance matrix between the intermediate features of discriminator D_2. Since the calculation incorporates information from different time steps, it can encode motion information of the given video $\mathbf{Y}$.

The Gram matrix has been successfully applied to synthesizing dynamic textures in previous works [3, 24], but our work differs from them in several aspects. First, we use the Gram matrix for video prediction, while the prior works use it for dynamic texture synthesis. Second, we directly calculate the Gram matrix of videos based on the features of discriminator D_2, which is updated in each iteration during training. In contrast, the prior works typically calculate it with a pre-trained VGG network [23], which is fixed during training. The motivation of such a different choice is that, as discriminator D_2 is closely related to the measurement of motion quality, it is reasonable to directly use features in D_2.

To make full use of the video representations, we adopt a variant of the contrastive loss introduced in [7] and [15] to compute the distance between videos. Our adversarial ranking loss with respect to features from the *l-th* layer is defined as:

$$\begin{aligned} &\mathcal{L}_{rank}\left(\mathbf{Y}_1, \mathbf{Y}_2, \mathbf{Y}; l\right) \\ &= -log \frac{e^{-\|g(\mathbf{Y}_2;l) - g(\mathbf{Y};l)\|_1}}{e^{-\|g(\mathbf{Y}_2;l) - g(\mathbf{Y};l)\|_1} + e^{-\|g(\mathbf{Y}_2;l) - g(\mathbf{Y}_1;l)\|_1}} . \end{aligned} \qquad (5)$$

We extract the features from multiple convolutional layers of the discriminator D_2 for the input $\mathbf{Y}_1$, output $\mathbf{Y}_2$ and ground-truth video $\mathbf{Y}$, and calculate their Gram matrices, respectively. The final adversarial ranking loss is:

$$\mathcal{L}_{rank}\left(\mathbf{Y}_1, \mathbf{Y}_2, \mathbf{Y}\right) = \sum_{l} \mathcal{L}_{rank}\left(\mathbf{Y}_1, \mathbf{Y}_2, \mathbf{Y}; l\right) . \qquad (6)$$

Similar to the objective in Stage I, we also incorporate the pixel-wise L_1 distance to capture low-level details. The overall objective for the Refine-Net is:

$$\mathcal{L}_{stage2} = \mathcal{L}_{adv} + \lambda \cdot \mathcal{L}_{rank} + \mathcal{L}_{con} . \qquad (7)$$

As shown in Algorithm 1, the generator and discriminator are trained alternatively. When training generator G_2 with discriminator D_2 fixed, we try to minimize the adversarial ranking loss $\mathcal{L}_{rank}\left(\mathbf{Y}_1, \mathbf{Y}_2, \mathbf{Y}\right)$, such that the distance between the generated $\mathbf{Y}_2$ and the ground-truth $\mathbf{Y}$ is encouraged to be smaller, while the distance between $\mathbf{Y}_2$ and $\mathbf{Y}_1$ is encouraged to be larger. By doing so, the distribution of videos generated by the Refine-Net is forced to be similar to that of the real ones, and the visual quality of videos from Stage I can be improved.

When training discriminator D_2 with generator G_2 fixed, on the contrary, we maximize the adversarial ranking loss $\mathcal{L}_{rank}\left(\mathbf{Y}_1, \mathbf{Y}_2, \mathbf{Y}\right)$. The insight behind is: if we update D_2 by always expecting that the distance between $\mathbf{Y}_2$ and $\mathbf{Y}$ is not small enough, then the generator G_2 is encouraged to produce $\mathbf{Y}_2$ that is closer to $\mathbf{Y}$ and further away from $\mathbf{Y}_1$ in the next iteration. By optimizing the ranking loss in such an adversarial manner, the Refine-Net is able to learn realistic dynamic patterns and yield vivid videos.

4. Experiments

4.1. Dataset

We build a relatively large-scale dataset of time-lapse videos from the Internet. We collect over 5,000 time-lapse videos from Youtube and manually cut these videos into short clips and select those containing dynamic sky scenes, such as the cloudy sky with moving clouds, and the starry sky with moving stars. Some of the clips may contain scenes that are dark or contain effects of quick zoom-in and zoom-out, thus are abandoned.

We split the set of selected video clips into a training set and a testing set. Note that all the video clips belonging to the same long video are in the same set to ensure that the testing video clips are disjoint from those in the training set. We then decompose the short video clips into frames, and generate clips by sequentially combining continuous 32 frames as a clip. There are no overlap between two consecutive clips. We collect 35,392 training video clips, and 2,815 testing video clips, each containing 32 frames. The original

Algorithm 1 The training procedure of the Refine-Net.

```
Set learning rates $\rho_d$ and $\rho_g$. Initialize the network parameters $\theta_d$ and $\theta_g$.
for number of iterations do
    Updating the discriminator $D_2$:
    Sample $N$ real video clips (a batch) $\{\mathbf{Y}^{(1)}, ... ,\mathbf{Y}^{(N)}\}$ from the training set.
    Obtain a batch of videos $\{\mathbf{Y}_1^{(1)}, ... ,\mathbf{Y}_1^{(N)}\}$ generated by the Base-Net.
    $\theta_d := \theta_d + \rho_d \nabla_{\theta_d} \frac{1}{N} \sum_{n=1}^{N} \left(\log D_2(\mathbf{Y}^{(n)}) + \log\left(1 - D_2(G_2(\mathbf{Y}_1^{(n)}))\right) + \lambda \cdot \mathcal{L}_{rank}\left(\mathbf{Y}_1^{(n)}, G_2(\mathbf{Y}_1^{(n)}), \mathbf{Y}^{(n)}\right)\right)$
    Updating the generator $G_2$:
    Sample $N$ new real video clips $\{\mathbf{Y}^{(1)}, ... ,\mathbf{Y}^{(N)}\}$ from the training set.
    Obtain a new batch of videos $\{\mathbf{Y}_1^{(1)}, ... ,\mathbf{Y}_1^{(N)}\}$ generated by the Base-Net .
    $\theta_g := \theta_g - \rho_g \nabla_{\theta_g} \frac{1}{N} \sum_{n=1}^{N} \left(\log\left(1 - D_2(G_2(\mathbf{Y}_1^{(n)}))\right) + \lambda \cdot \mathcal{L}_{rank}\left(\mathbf{Y}_1^{(n)}, G_2(\mathbf{Y}_1^{(n)}), \mathbf{Y}^{(n)}\right) + \mathcal{L}_{con}\right)$
end for
```

size of each frame is $3 \times 640 \times 360$, and we resize it into a square image of size 128×128. Before feeding the clips to the model, we normalize the color values to $[-1, 1]$. No other preprocessing is required.

Our dataset contains videos with both complex contents and diverse motion patterns. There are various types of scenes in the dataset, including daytime, nightfall, dawn, starry night and aurora. They exhibit different kinds of foregrounds (the sky), and colors. Unlike some previous time-lapse video datasets, *e.g.* [37], which contain relatively clean backgrounds, the backgrounds in our dataset show high-level diversity across videos. The scenes may contain trees, mountains, buildings and other static objects. It is also challenging to learn the diverse dynamic patterns within each type of scenes. The clouds in the blue sky may be of any arbitrary shape and move in any direction. In the starry night scene, the stars usually move fast along a curve in the dark sky.

Our dataset can be used for various tasks on learning dynamic patterns, including unconditional video generation [28], video prediction [27], video classification [11], and dynamic texture synthesis [3]. In this paper, we use it for video prediction.

4.2. Implementation Details

The Base-Net takes a $3 \times 128 \times 128$ starting image and generates 32 image frames of resolution 128×128, *i.e.*, $T = 32$. The Refine-Net takes the output video of the Base-Net as input, and generates a more realistic video with 128×128 resolution. The models in both stages are optimized with stochastic gradient descent. We use Adam as the optimizer with $\beta = 0.5$ and the momentum being 0.9. The learning rate is 0.0002 and fixed throughout the training procedure.

We use Batch Normalization [8] followed by Leaky ReLU [33] in all the 3D convolutional layers in both generators and discriminators, except for their first and last layers. For the deconvolutional layers, we use ReLU [18] instead of Leaky ReLU. We use Tanh as the activation function of the output layer of the generators. The Gram matrices are calculated using the features of the first and third convolutional layers (after the ReLU layer) of discriminator D_2. The weight of the adversarial ranking loss is set to 1 in all experiments, *i.e.*, $\lambda = 1$. The detailed configurations of G_1 are given in Table 1. In G_2, we remove the skip connections between "conv1" and "deconv6", "conv2" and "deconv5". We use the identity mapping as the skip connection [6].

4.3. Comparison with Existing Methods

We perform quantitative comparison between our model and the models presented in [28] and [37]. For notation convenience, we name these two models as VGAN [28] and RNN-GAN [37], respectively. For a fair comparison, we reproduce the results of their models exactly according to their papers and reference codes, except some adaption to match the same experimental setting as ours. The adaption includes that, all the methods produce 32 frames as the output. Note that, both VGAN and RNN-GAN generate videos of resolution 64×64, so we resize the videos produced by our model to resolution 64×64 for fairness.

Fig. 1 shows exemplar results by each method. The video frames generated by VGAN (the first row) and RNN-GAN (the second row) tend to be blurry, while our Base-Net (the third row) and Refine-Net (the fourth row) produce samples that are much more realistic, indicating that skip connections and the 3D U-net like architecture greatly benefit the content generation.

In order to perform a more direct comparison for each model on both content and motion generation, we compare them in pairs. For each two models, we randomly select 100 clips from the testing set and take their first frames as the input. Then we produce the future prediction as a video of 32 frames by the two models. We conduct 100 times of opinion tests from professional workers based on the outputs. Each time we show a worker two videos generated from the two models given the same input frame. The worker is required to give opinion about which one is more realistic. The two

Table 1. The architecture of the generators in both stages. The size of the input video is $3 \times 32 \times 128 \times 128$.

Layers	conv1	conv2	conv3	conv4	conv5	conv6	deconv1	deconv2	deconv3	deconv4	deconv5	deconv6
# Filters	32	64	128	256	512	512	512	256	128	64	32	3
Filter Size	(3, 4, 4)	(4, 4, 4)	(4, 4, 4)	(4, 4, 4)	(4, 4, 4)	(2, 4, 4)	(4, 4, 4)	(4, 4, 4)	(4, 4, 4)	(4, 4, 4)	(4, 4, 4)	(3, 4, 4)
Stride	(1, 2, 2)	(2, 2, 2)	(2, 2, 2)	(2, 2, 2)	(2, 2, 2)	(1, 1, 1)	(1, 1, 1)	(2, 2, 2)	(2, 2, 2)	(2, 2, 2)	(2, 2, 2)	(1, 2, 2)
Padding	(1, 1, 1)	(1, 1, 1)	(1, 1, 1)	(1, 1, 1)	(1, 1, 1)	(0, 0, 0)	(0, 0, 0)	(1, 1, 1)	(1, 1, 1)	(1, 1, 1)	(1, 1, 1)	(1, 1, 1)

Table 2. Quantitative comparison results of different models. We show pairs of videos to a few workers, and ask them "which is more realistic". We count their evaluation results, which are denoted as Preference Opinion Score (POS). The value range of POS can be $[0, 100]$. If the value is greater than 50 then it means that the former performs better than the latter.

"Which is more realistic?"	POS
Random Selection	50
Prefer Ours over VGAN	92
Prefer Ours over RNN-GAN	97
Prefer VGAN over Real	5
Prefer RNN-GAN over Real	1
Prefer Ours over Real	16

videos are shown in a random order to avoid the potential issue that the worker tends to always prefer a video on the left (or right) due to laziness. Five groups of comparison are conducted in total. Apart from the comparisons between ours and VGAN and RNN-GAN, respectively, we also conduct comparisons of ours, VGAN and RNN-GAN against real videos to evaluate the performance of these models.

Table 2 shows the quantitative comparison results. Our model outperforms VGAN [28] with regard to the Preference Opinion Score (POS). Qualitatively, videos generated by VGAN are usually not as sharp as ours. The following reasons are suspected to contribute to the superiority of our model. First, we adopt the U-net like structure instead of a vanilla encoder-decoder structure in VGAN. The connections between the encoder and the decoder bring more powerful representations, thus producing more concrete contents. Second, the Refine-Net makes further efforts to learn more vivid dynamic patterns. Our model also performs better than RNN-GAN [37]. One reason may be that RNN-GAN uses an RNN to sequentially generate image frames, so their results are prone to error accumulation. Our model employs 3D convolutions instead of RNN so that the state of the next frame does not heavily depend on the state of previous frames.

When comparing ours, VGAN and RNN-GAN with real videos, our model consistently achieves better POS than both VGAN and RNN-GAN, showing the superiority of our multi-stage model. Some results of our model are as decent as the real ones, or even perceived as more realistic than the real ones, suggesting that our model is able to generate realistic future scenes.

Table 3. Quantitative comparison results of Stage I versus Stage II. The evaluation metric is the same as that in Table 2.

"Which is more realistic?"	POS
Random Selection	50
Prefer Stage II to Stage I	70
Prefer Stage II to Real	16
Prefer Stage I to Real	8

4.4. Comparison between Base-Net and Refine-Net

Although the Base-Net can generate videos of decent details and plausible motion, it fails to generate vivid dynamics. For instance, some of the results in the scene of cloudy daytime fail to exhibit apparent cloud movements. The Refine-Net makes attempts to compensate for the motion based on the result of Base-Net, while preserving the concrete content details. In this part, we evaluate the performance of Stage II versus Stage I in terms of both quantitative and qualitative results.

Quantitative Results. Given an identical starting frame as input, we generate two videos by the Base-Net in Stage I and the Refine-Net in Stage II separately. The comparison is carried out over 100 pairs of generated videos in a similar way to that in the previous section. Showing each pair of two videos, we ask the workers which one is more realistic. To check how effective our model is, we also compare the results of the Base-Net and Refine-Net with the ground-truth videos. The results shown in Table 3 reveal that the Refine-Net contributes significantly to the reality of the generated videos. When comparing the Refine-Net with the Base-Net, the advantage is about 40 (70 versus 30) in terms of the POS. Not surprisingly, the Refine-Net gains better POS than the Base-Net when comparing videos of these two models with the ground-truth videos.

Qualitative Results. As is shown in Fig. 1, although our Refine-Net mainly focuses on improving the motion quality, it still preserves fine content details which are visually almost as realistic as the frames produced by Base-Net. In addition to content comparison, we further compare the motion dynamics of the generated video by the two stages. We show four video clips generated by the Base-Net and the Refine-Net individually on the basis of the same starting frame in Fig. 3. Motions are indicated by red circles in the frames. Please note the differences between the next and previous frames. Results in Fig. 3 indicate that although the Base-Net can generate concrete object details, the con-

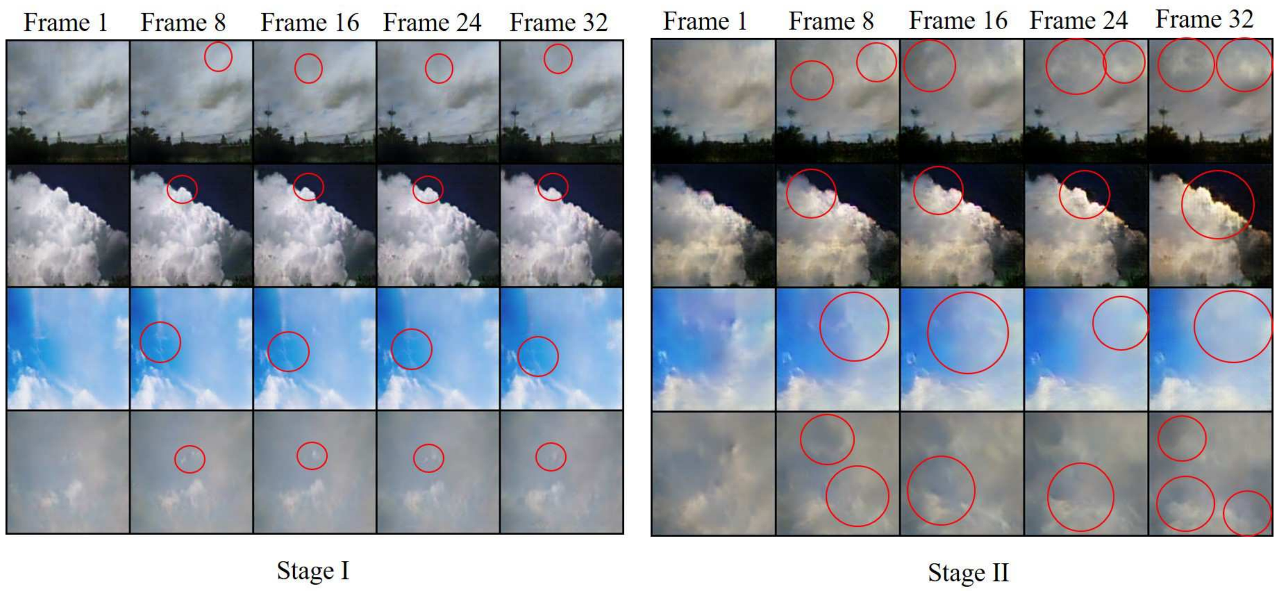

Figure 3. The generated video frames by Stage I (left) and Stage II (right) given the same starting frame. We show exemplar frames 1, 8, 16, 24, and 32. Red circles are used to indicate the locations and areas where obvious movements take place between adjacent frames. Larger and more circles are observed in the frames of Stage II, indicating that there are more vivid motions generated by the Refine-Net.

tent of the next frames seems to have no significant difference from the previous frames. While it does captures the motion patterns to some degree, like the color changes or some inconspicuous object movements, the Base-Net fails to generate vivid dynamic scene sequences. In contrast, the Refine-Net takes the output of the Base-Net to produce more realistic motion dynamics learned from the dataset. As a result, the scene sequences show more evident movements across adjacent frames.

4.5. Experiment on various video contexts

Although our model works on time-lapse video generation, it can be generalized to the prediction of other video scenes. To evaluate the robustness and effectiveness of our approach, we compare our model with both VGAN and RNN-GAN on the Beach and Golf datasets released by [28], which do not contain any time-lapse video. For each dataset, we use only 10% of them as training data, and the rest as testing data. For a fair comparison, all these models take a 64×64 starting frame as input. To this end, we adjust our model to take 64×64 resolution image and video by omitting the first convolutional layer of the generators and discriminators and preserving the rest parts. For each approach, we calculate the Mean Square Error (MSE), Peak Signal to Noise Ratio (PSNR) and Structural Similarity Index (SSIM) between 1000 randomly sampled pairs of generated video and the corresponding ground-truth video. Results shown in Tables 4 and 5 demonstrate the superiority of our MD-GAN model.

Table 4. Experiment results on the Beach dataset in terms of MSE, PSNR and SSIM (arrows indicating direction of better performance). The best performance values are shown in bold.

Model	MSE↓	PSNR ↑	SSIM ↑
VGAN [28]	0.0958	11.5586	0.6035
RNN-GAN [37]	0.1849	7.7988	0.5143
MD-GAN Stage II (Ours)	**0.0422**	**16.1951**	**0.8019**

Table 5. Experiment results on the Golf dataset.

Model	MSE↓	PSNR ↑	SSIM ↑
VGAN [28]	0.1188	9.9648	0.5133
RNN-GAN [37]	0.2333	7.7583	0.4306
MD-GAN Stage II (Ours)	**0.0681**	**13.7870**	**0.7085**

5. Conclusions

We propose the MD-GAN model which can generate realistic time-lapse videos of resolution as high as 128×128 in a coarse-to-fine manner. In the first stage, our model generates sharp content details and rough motion dynamics by Base-Net with a 3D U-net like network as the generator. In the second stage, Refine-Net improves the motion quality with an adversarial ranking loss which incorporates the Gram matrix to effectively model the motion patterns. Experiments show that our model outperforms the state-of-the-art models and can generate videos which are visually as realistic as the real-world videos in many cases.

6. Acknowledgement

This work is supported in part by New York State through the Goergen Institute for Data Science, as well as the corporate sponsors Snap Inc. and Cheetah Mobile.

References

[1] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223, 2017.

[2] E. Denton, S. Chintala, A. Szlam, and R. Fergus. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances In Neural Information Processing Systems (NIPS)*, pages 1486–1494, 2015.

[3] C. M. Funke, L. A. Gatys, A. S. Ecker, and M. Bethge. Synthesising dynamic textures using convolutional neural networks. *arXiv preprint arXiv:1702.07006*, 2017.

[4] L. Gatys, A. S. Ecker, and M. Bethge. Texture synthesis using convolutional neural networks. In *Advances In Neural Information Processing Systems (NIPS)*, pages 262–270, 2015.

[5] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances In Neural Information Processing Systems (NIPS)*, pages 2672–2680, 2014.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision (ECCV)*, pages 630–645. Springer, 2016.

[7] E. Hoffer and N. Ailon. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*, pages 84–92. Springer, 2015.

[8] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, pages 448–456, 2015.

[9] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[10] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer, 2016.

[11] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1725–1732, 2014.

[12] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *International Conference on Learning Representations (ICLR)*, 2014.

[13] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4681–4690, 2017.

[14] J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, and S. Yan. Perceptual generative adversarial networks for small object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[15] X. Liang, H. Zhang, and E. P. Xing. Generative semantic manipulation with contrasting gan. *arXiv preprint arXiv:1708.00315*, 2017.

[16] M. Mathieu, C. Couprie, and Y. LeCun. Deep multi-scale video prediction beyond mean square error. *arXiv preprint arXiv:1511.05440*, 2015.

[17] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[18] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning (ICML)*, pages 807–814, 2010.

[19] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *International Conference on Learning Representations (ICLR)*, 2016.

[20] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. *International Conference on Learning Representations (ICLR)*, 2016.

[21] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer, 2015.

[22] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014.

[23] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations (ICLR)*, 2015.

[24] M. Tesfaldet, M. A. Brubaker, and K. G. Derpanis. Two-stream convolutional networks for dynamic texture synthesis. *arXiv preprint arXiv:1706.06982*, 2017.

[25] S. Tulyakov, M.-Y. Liu, X. Yang, and J. Kautz. Mocogan: Decomposing motion and content for video generation. *arXiv preprint arXiv:1707.04993*, 2017.

[26] R. Villegas, J. Yang, S. Hong, X. Lin, and H. Lee. Decomposing motion and content for natural video sequence prediction. *International Conference on Machine Learning (ICML)*, 1(2):7, 2017.

[27] R. Villegas, J. Yang, Y. Zou, S. Sohn, X. Lin, and H. Lee. Learning to generate long-term future via hierarchical prediction. *International Conference on Machine Learning (ICML)*, 2017.

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

[29] C. Vondrick and A. Torralba. Generating the future with adversarial transformers. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[30] B. Wu, W. Chen, P. Sun, W. Liu, B. Ghanem, and S. Lyu. Tagging like humans: Diverse and distinct image annotation. 2018.

[31] Z. Wu, Y. Fu, Y.-G. Jiang, and L. Sigal. Harnessing object and scene semantics for large-scale video understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3112–3121, 2016.

[32] S. Xiao, M. Farajtabar, X. Ye, J. Yan, L. Song, and H. Zha. Wasserstein learning of deep generative point process models. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3247–3257, 2017.

[33] B. Xu, N. Wang, T. Chen, and M. Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.

[34] T. Xue, J. Wu, K. L. Bouman, and W. T. Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *Advances In Neural Information Processing Systems (NIPS)*, 2016.

[35] H. Yu, J. Wang, Z. Huang, Y. Yang, and W. Xu. Video paragraph captioning using hierarchical recurrent neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4584–4593, 2016.

[36] H. Zhang, T. Xu, H. Li, S. Zhang, X. Huang, X. Wang, and D. Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. *International Conference on Computer Vision (ICCV)*, 2017.

[37] Y. Zhou and T. L. Berg. Learning temporal transformations from time-lapse videos. In *European Conference on Computer Vision (ECCV)*, pages 262–277. Springer, 2016.

[38] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *International Conference on Computer Vision (ICCV)*, 2017.