Learning Generator Networks for Dynamic Patterns

Tian Han
University of California, Los Angeles
hantian@ucla.edu

Lu Yang
University of California, Los Angeles
yanglv@ucla.edu

Jiawen Wu
University of Southern California
jiawenwu@usc.edu

Xianglei Xing
Harbin Engineering University
xingxl@hrbeu.edu.cn

Ying Nian Wu
University of California, Los Angeles
ywu@stat.ucla.edu

Abstract

We address the problem of learning dynamic patterns from unlabeled video sequences, either in the form of generating new video sequences, or recovering incomplete video sequences. This problem is challenging because the appearances and motions in the video sequences can be very complex. We propose to use the alternating back-propagation algorithm to learn the generator network with the spatial-temporal convolutional architecture. The proposed method is efficient and flexible. It can not only generate realistic video sequences, but can also recover the incomplete video sequences in the testing stage or even in the learning stage. The proposed algorithm can be further improved by using learned initialization which is useful for the recovery tasks. Further, the proposed algorithm can naturally help to learn the shared representation between different modalities. Our experiments show that our method is competitive with the existing state of the art methods both qualitatively and quantitatively.

1. Introduction

Understanding dynamic patterns in video sequences is one of the most fundamental problems in computer vision and artificial intelligence. For many real-world applications, e.g., simulation, forecasting, etc., an effective model of dynamic patterns is needed. However, learning generative models of dynamic patterns is challenging because the appearances and motions in video sequences can be very complex.

We aim to address the problem of learning dynamic patterns from unlabeled video sequences. A model for dynamic patterns must capture the underlying spatial and temporal patterns in the video sequences. It should have the ability to synthesize new video sequences, as well as recover the incomplete sequences. To achieve this goal, we capitalize on the recent advances in unsupervised learning algorithms for generative image models [7, 17, 25, 23, 9, 15, 26, 21]. Specifically, we propose to generalize the alternating back-propagation algorithm recently proposed by Han et al. [9], to learn the spatial-temporal generator network [31].

The generator network, in general, can be viewed as a non-linear generalization of the factor analysis model [9]. Based on this observation, it is natural to generalize the alternating back-propagation algorithm [9] to learn dynamic patterns. The algorithm iterates the following two steps: (1) inferential back-propagation, which infers the latent factors by Langevin sampling from the posterior distribution of the latent factors, and (2) learning back-propagation, which updates the network parameters given the inferred latent factors. It has been shown in [9] that such an iterative algorithm can be used for image modeling. However, the effectiveness of such an algorithm for modeling dynamic patterns remains unexplored.

In this paper, we generalize the alternating back-propagation algorithm to learn the different types of spatial-temporal generator networks [31]. We also implement and provide strong baseline models, i.e., spatial-temporal variational auto-encoder and spatial-temporal wake sleep model that generalize the conventional variational auto-encoder [26, 17, 18], and wake sleep model [12] to their spatial-temporal (3D) domains. These inference-based models are well suited for synthesizing dynamic patterns as well as recovering incomplete patterns. We further extend

our framework to multimodal learning by sharing latent representations among different modalities. Our experiments demonstrate the effectiveness of our approach in both synthesis and recovery/prediction tasks.

2. Related work and contributions

Our work is closely related to studies of video modeling. However, previous works have mostly focused on recognition or classification tasks [14, 6, 30, 32]. In our work, we are interested in generating and completing dynamic patterns, rather than detecting and retrieving features from them.

There are two major classes of approaches to modeling and generating dynamic patterns.

The first class of methods is based on state space models for dynamic patterns. They explicitly model the inter-frame transitions and can be viewed as generative models conditioned on the past frames. Notably, Doretto et al. [5] proposed an auto-regressive transition model together with linear frame-wise dimension reduction for dynamic textures. Using the deep learning methods, variants of RNN/LSTM have been used to model the transitions of dynamic patterns [24, 27]. These approaches can be used for pattern synthesis [33, 5], and future generation in videos [19, 27, 37]. These models are casual, meaning that they require the starting frame. Moreover, random noises may be accumulated over time, so that they can only generate realistic patterns for a relatively short time span. Besides, it can be difficult for such models to recover incomplete dynamic patterns and combine with other modalities.

The second class of methods is based on spatial-temporal models. Our method belongs to this class. The spatial-temporal models seek to capture the dynamic patterns directly by spatial-temporal kernels or filters. Most notably, Vondrick et al. [31] used the generative adversarial network to train the spatial-temporal generator network on unlabeled videos and demonstrated realistic temporal semantics on video generation task. Xie [34] trained the spatial-temporal energy model and also showed sharp generation results on dynamic textures. However, the latent factors in their models are not inferred in the adversarial training and energy-based training, and the lack of inference makes it hard to deal with the pattern completion task in the learning stage. Moreover, it is generally not easy to learn shared representation when other modalities are involved.

In the literature of dynamic patterns, there have been considerable past efforts on modeling dynamic textures. See [3, 29] for recent literature reviews. Dynamic textures are stochastic processes that exhibit stationary regularity in the temporal domain. A major class of models on dynamic textures focus on the state space auto-regressive model for synthesis [5, 9, 38, 4] and classification [2, 36]. However, the spatial-temporal approach to such problems has not been thoroughly explored. In this paper, we explore such an approach. Meanwhile, our method can also be used to model other dynamic patterns, e.g., simple actions, as shown in the experiment section.

We leverage the inference-based approach recently proposed by [9], and apply it to the modeling of dynamic patterns. The main differences with [9] are (1) they only considered frame-wise spatial generator model where only qualitative synthesis results are presented. While we propose and analyze three types of video configurations by utilizing different temporal and spatial stationarities, and perform thorough evaluations and comparisons over different video tasks. (2) They only learn the representation from single data modal, where we further extend it to tackle multimodal shared representation learning [10]. (3) They didn't consider the video recovery tasks, while in our work, we not only evaluate our method on such tasks, but also propose to learn the initialization to further improve the recovery accuracy and efficiency.

The contributions of our paper are as follows:

- Proposing and analyzing three types of video models based on generator network, and generalizing the alternating back-propagation algorithm to learn such models.
- Improving the proposed algorithm by using the learned initialization for video recovery tasks which renders more accurate results.
- Conducting extensive experiments to show that our approach is competitive with existing spatial-temporal generative models both qualitatively and quantitatively. We further show that it can be naturally extended to deal with multimodal shared representation learning.

3. Model and algorithm

3.1. Spatial-temporal generator model

Let $\mathbf{I}(x, y, t)$ be an image sequence, where $(x, y) \in \mathcal{S}$ indexes the pixel in the spatial domain, and $t \in \mathcal{T}$ indexes the frame in the temporal domain. We assume that $\mathbf{I}$ is generated by a generator network. At the top of the network is a layer of latent factors $\mathbf{Z} = (Z_k, k = 1, ..., d)$. The model is a non-linear generalization of factor analysis:

$$\begin{aligned} &\mathbf{I} = G(\mathbf{Z}; W) + \epsilon, \\ &Z_k \sim \mathrm{N}(0, 1),\ \epsilon(x, y, t) \sim \mathrm{N}(0, \sigma^2), \end{aligned} \tag{1}$$

and all the Z_k and $\epsilon(x, y, t)$ are independent. $G(\mathbf{Z}; W)$ is a non-linear transformation parametrized by a top-down convolutional neural network that consists of multiple layers of deconvolution by spatial-temporal kernels, ReLU non-linearity, and up-sampling. W consists of all the weight

and bias parameters of the spatial-temporal kernels of the network.

For modeling image sequence $\mathbf{I}(x, y, t)$, we consider the following three versions of the generator model:

(1) The dynamic pattern in $\mathbf{I}(x, y, t)$ is stationary in both the spatial and temporal domains, such as water waves, rain, snow etc. In that case, we assume $\mathbf{Z} = \mathbf{Z}(x, y, t, m)$, where $(x, y) \in \mathcal{S}^{(0)}$, $t \in \mathcal{T}^{(0)}$, and $m = 1, ..., M$, where $\mathcal{S}^{(0)}$ and $\mathcal{T}^{(0)}$ are sub-sampled spatial and temporal domains, i.e., the index k in $\mathbf{Z} = (Z_k, k = 1, ..., d)$ takes the form $k = (x, y, t, m)$, so that at the top-layer $\mathbf{Z}$ consists of M Gaussian white noise image sequences. We assume that the kernel functions are all convolutional, so that $G(\mathbf{Z}; W)$ is stationary in both spatial and temporal domains after L layers of deconvolution and up-sampling. The up-sampling expands $(\mathcal{S}^{(0)}, \mathcal{T}^{(0)})$ at the top layer to $(\mathcal{S}, \mathcal{T})$ at the bottom layer.

(2) The dynamic pattern in $\mathbf{I}(x, y, t)$ is stationary in the temporal domain, but non-stationary in the spatial domain, such as dynamic textures that exhibit stochastic repetitiveness in the temporal domain. In that case, we assume $\mathbf{Z} = \mathbf{Z}(m, t)$, where $t \in \mathcal{T}^{(0)}$, and $m = 1, ..., M$. This model corresponds to the model in (1) where $\mathcal{S}^{(0)}$ is 1×1. The kernel functions at the top layer are convolutional in the temporal domain, but are fully connected in the spatial domain, so that $G(\mathbf{Z}; W)$ is stationary in the temporal domain but non-stationary in the spatial domain.

(3) The dynamic pattern in $\mathbf{I}(x, y, t)$ is non-stationary in both spatial and temporal domains, such as actions and movements. In that case, we assume $\mathbf{Z} = \mathbf{Z}(m)$, $m = 1, ..., M$. The model corresponds to the model in (1) where $\mathcal{S}^{(0)}$ is a single pixel, and $\mathcal{T}^{(0)}$ is a single frame. The kernel functions at the top layer are fully connected in both the temporal and spatial domains.

Models in (1) and (2) can be learned from a single training sequence. Models in (1) can generate image sequences with both the spatial and temporal ranges expanded. Models in (2) can generate image sequences with expanded temporal range. Models in (3) need to be learned from multiple sequences because of the lack of stochastic repetitiveness in either the spatial or temporal domain.

3.2. Alternating back-propagation

We now briefly review the alternating back-propagation algorithm [9]. The generator model can be learned from a training set of image sequences $\{\mathbf{I}_i, i = 1, ..., n\}$ by maximum likelihood. The basic idea of the learning algorithm is to perform gradient descent on the reconstruction error $\sum_{i=1}^{n} \|\mathbf{I}_i - G(\mathbf{Z}_i; W)\|^2$ alternatively over $\{\mathbf{Z}_i\}$ and W. Because $\mathbf{Z}_i$ is a random vector, it can also be sampled from its posterior distribution.

Specifically, we can write the model as $\mathbf{Z} \sim p(\mathbf{Z})$ and $[\mathbf{I}|\mathbf{Z}; W] \sim p(\mathbf{I}|\mathbf{Z}; W)$. The joint distribution $p(\mathbf{I}, \mathbf{Z}; W)$ is such that

$$\begin{aligned} &\log p(\mathbf{I}, \mathbf{Z}; W) = \log [p(\mathbf{Z}) p(\mathbf{I}|Z; W)] \\ &= -\frac{1}{2\sigma^2} \|Y - G(\mathbf{Z}; W)\|^2 - \frac{1}{2} \|\mathbf{Z}\|^2 + \text{const}, \end{aligned} \tag{2}$$

where the constant term has nothing to do with $\mathbf{I}$, $\mathbf{Z}$, W.

The model can be learned by taking gradient of the observed data log-likelihood $L(W)$ which can be obtained by integrating out $\mathbf{Z}$ [9]:

$$\begin{aligned} \frac{\partial}{\partial W} L(W) &= \frac{\partial}{\partial W} \log p(\mathbf{I}; W) \\ &= \mathbf{E}_{p(\mathbf{Z}|\mathbf{I}, W)} \left[\frac{\partial}{\partial W} \log p(\mathbf{I}, \mathbf{Z}; W) \right], \end{aligned} \tag{3}$$

where $p(\mathbf{Z}|\mathbf{I}; W) = p(\mathbf{I}, \mathbf{Z}; W)/p(\mathbf{I}; W) \propto p(\mathbf{I}, \mathbf{Z}; W)$ is the posterior distribution of $\mathbf{Z}$ given $\mathbf{I}$. The algorithm iterates between the so called inferential step and learning step.

For the inferential step, Langevin dynamics is used to approximate the expectation with respect to $p(\mathbf{Z}|\mathbf{I}; W)$ which iterates

$$\begin{aligned} \mathbf{Z}_{\tau+1} &= \mathbf{Z}_\tau + \delta \mathcal{E}_\tau \\ &- \frac{\delta^2}{2} \frac{\partial}{\partial \mathbf{Z}} \left[\frac{1}{2\sigma^2} \|\mathbf{I} - G(\mathbf{Z}_\tau; W)\|^2 \right], \end{aligned} \tag{4}$$

where τ indexes the step for the Langevin sampling, δ is the step size, and $\mathcal{E}_\tau \sim \mathrm{N}(0, I_d)$, i.e., the components of $\mathcal{E}$ are independent standard normal random variables. After the inferential step, posterior sampled Z can be obtained for each image sequence.

For the learning step, the inferred Zs are used to approximate $\partial \log p(\mathbf{I}_i; W)/\partial W$, and the spatial-temporal generator can be learned by stochastic gradient descent according to:

$$\begin{aligned} L'(W) &\approx \sum_{i=1}^{n} \frac{\partial}{\partial W} \log p(\mathbf{I}_i, \mathbf{Z}_i; W) \\ &= \sum_{i=1}^{n} \frac{1}{\sigma^2} (\mathbf{I}_i - G(\mathbf{Z}_i; W)) \frac{\partial}{\partial W} G(\mathbf{Z}_i; W), \end{aligned} \tag{5}$$

Both the inferential step and the learning step are based on back-propagations of the error $\mathbf{I}_i - G(\mathbf{Z}_i; W)$.

3.3. Initializing inferential back-propagation

To initialize $\mathbf{Z}_i$ in the Langevin dynamics of the inferential back-propagation, we can simply draw $\mathbf{Z}_i$ from its prior distribution in the first epoch. In each subsequent epoch, we initialize $\mathbf{Z}_i$ from its current value. This is the approach used in the original alternating back-propagation algorithm [9]. But this approach may be ineffective when dealing with incomplete data, since the Langevin may need

Algorithm 1 Alternating back-propagation

Require:

(1) training image sequences $\{\mathbf{I}_i, i = 1, ..., n\}$

(2) number of Langevin iterations l and learning iterations T

Ensure:

(1) learned generator network parameters W

(2) inferred latent factors $\{\mathbf{Z}_i, i = 1, ..., n\}$

(3) (Optional) learned initialization network parameters V

1: Let $t \leftarrow 0$, initialize W.
2: Initialize $\mathbf{Z}_i$ (or $\mathbf{Z}_i = f(\mathbf{I}_i, V_0)$), for $i = 1, ..., n$.
3: **repeat**
4: **Inferential step**: For each i, run l steps of Langevin dynamics to sample $\mathbf{Z}_i \sim p(\mathbf{Z}_i|\mathbf{I}_i, W)$ by starting from the current $\mathbf{Z}_i$ (or $\mathbf{Z}_i = f(\mathbf{I}_i; V)$), each step follows equation (4).
5: **Learning step**: Update $W \leftarrow W + \eta_t L'(W)$, where $L'(W)$ is computed according to equation (5), with learning rate η_t.
(Optional) Update $V \leftarrow V + \epsilon_t R'(V)$ where $R(V)$ is the reconstruction error based on generated $\tilde{\mathbf{I}}_i$ (see Section 3.3), with learning rate ϵ_t.
6: Let $t \leftarrow t + 1$
7: **until** $t = T$

a long run to get mixed when only partial information is available. Therefore, the informative starting point is needed for the Langevin dynamics. We propose to learn an initialization network $\hat{\mathbf{Z}} = f(\mathbf{I}; V)$, so that for each $\mathbf{I}_i$, we initialize the Langevin dynamics from $\hat{\mathbf{Z}}_i = f(\mathbf{I}_i; V)$. We can learn $f(\mathbf{I}; V)$ from the data generated by the current model. Specifically, we generate $(\tilde{\mathbf{Z}}_i, \tilde{\mathbf{I}}_i, i = 1, ..., \tilde{n}) \sim p(\mathbf{Z})p(\mathbf{I}|\mathbf{Z}; W_t)$. Then we update V by gradient descent that minimizes $R(V) = \sum_{i=1}^{\tilde{n}} \|\tilde{\mathbf{Z}}_i - f(\tilde{\mathbf{I}}_i; V)\|^2$. This method is related to the sleep phase of the wake-sleep algorithm. It is different from the wake-sleep algorithm [12] in that we do not use $\hat{\mathbf{Z}}_i$ as the inferred value. It only serves to initialize the Langevin dynamics that samples from the posterior distribution. Algorithm 1 describes the the learning and sampling algorithm.

3.4. Recover incomplete sequences

The alternating back-propagation learning method can be used to recover the incomplete image sequences in either the testing stage or the training stage.

(1) Pattern completion in the testing stage. Suppose we learn the generator model $G(\mathbf{Z}; W)$ from complete training image sequences $\{\mathbf{I}_i\}$. For a testing sequence $\mathbf{I}$, suppose some frames or pixels are occluded. We can recover $\mathbf{I}$ using the inferential step described in the above subsection. The only difference is in the computation of $\|\mathbf{I} - G(\mathbf{Z}; W)\|^2$ in equation (4), where, instead of summing over all the pixels in the image sequence, we only sum over the observed pixels. After inferring $\mathbf{Z}$ by the Langevin dynamics, we can then feed it to the learned generator model to recovery $\mathbf{I}$ as $G(\mathbf{Z}; W)$.

(2) Pattern completion in the training stage. Suppose we want to learn a generator model $G(\mathbf{Z}; W)$ from incomplete training image sequences $\{\mathbf{I}_i\}$, where some frames or pixels of each $\mathbf{I}_i$ are occluded. We can still learn the model with a minor change in the learning algorithm. Again the difference is in the computation of $\|\mathbf{I}_i - G(\mathbf{Z}_i; W)\|^2$ in equation (4) in the inference step and in equation (5) in the learning step, where we only sum over the observed pixels. At the end of the learning algorithm, we obtain $\{\mathbf{Z}_i\}$ and W, so that the training image $\mathbf{I}_i$ can be recovered by $G(\mathbf{Z}_i; W)$.

3.5. Multimodal shared representation

The alternating back-propagation algorithm can also be used to learn a shared representation when multiple modalities are presented.

Suppose we have image sequences $\{\mathbf{I}_i\}$ and corresponding audio sequences $\{\mathbf{A}_i\}$. In order to learn the common representation, we can build one spatial-temporal generator $G(\mathbf{Z}; W_I)$ for video and one temporal generator $G(\mathbf{Z}; W_A)$ for audio. These generators should share the same latent factor Z. This task can be naturally fitted into the current algorithm, by adding an extra reconstruction $\|\mathbf{A} - G(\mathbf{Z}; W_A)\|^2$ for the inferential step to drive the representation Z to contain information from both image sequences and audio sequences. During testing, we can use the learned generator for one modality (e.g., audio) to infer the common Z, then feed it to the learned generator of the other modality (e.g., video) to make a prediction.

4. Experiments

We train the spatial-temporal generator network on a wide range of dynamic patterns randomly selected from the DynTex [22] database, action database [8], and from the Internet. Representative ones that contain various spatial-temporal features are chosen for our experiments. Each video sequence from the selected category is partitioned into segments of 32 frames, and such segments are used for training and testing. The pixel values are scaled to be within the range of [-1, 1]. For the generator network, we adopt the architecture that is similar to the generator network of [31] with different top-layer structures for different models. All the deconvolution layers are followed by the batch normalization [13] and the ReLU non-linearity except the last layer where tanh is used. The detailed descriptions of our model structures will be given later. We fix the standard deviation of noise vector $\sigma = 1$ for qualitative experiments and $\sigma = 0.5$ for quantitative experiments.

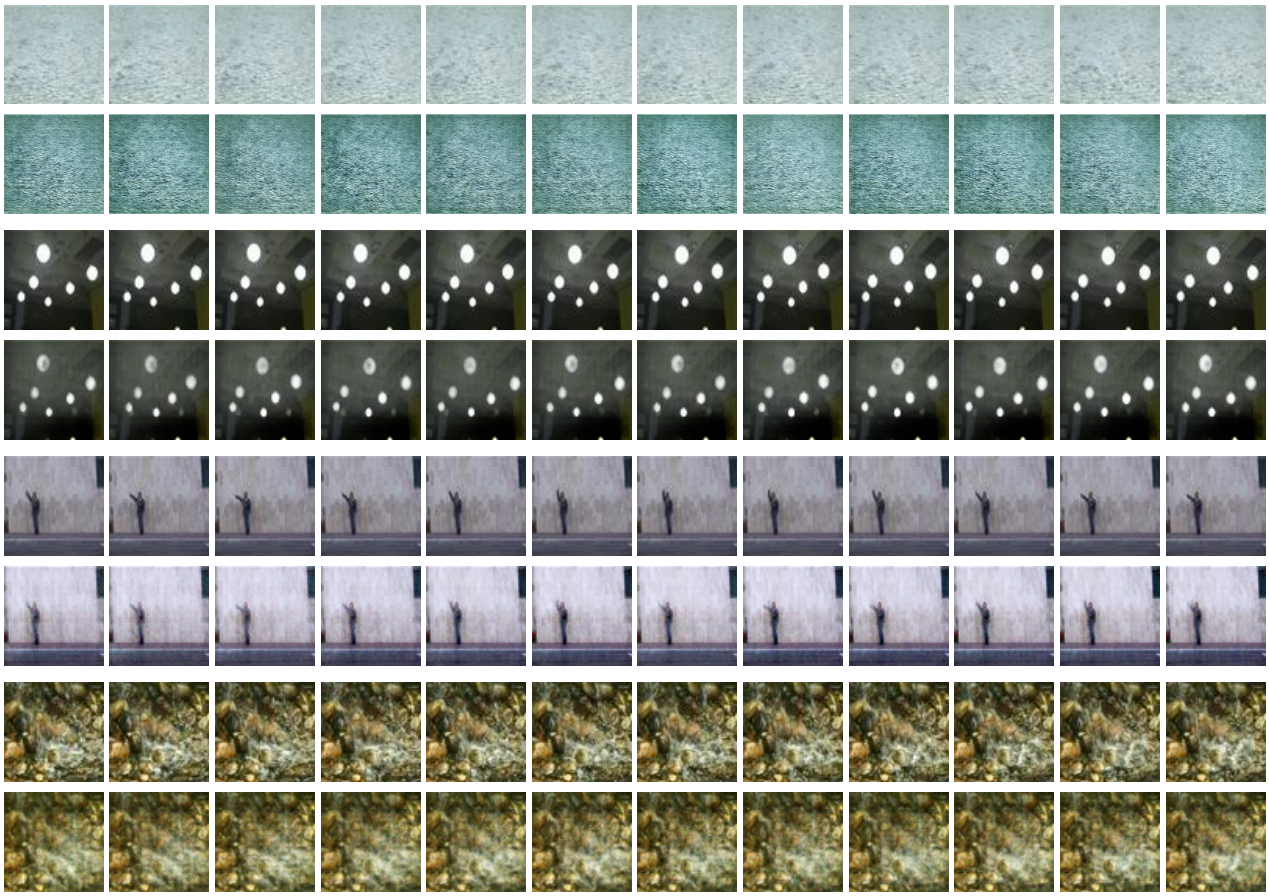

Figure 1. Synthesized dynamic sequences using three versions of the generator model. For each category, the first row displays frames of the training sequence, and the second row displays the frames of the generated sequence. The first category shows the synthesized sequence generated by model-1 with the original length. Category 2, category 3 display the synthesized sequences generated by model-2 with doubled length. The last category displays the generated sequence by model-3.

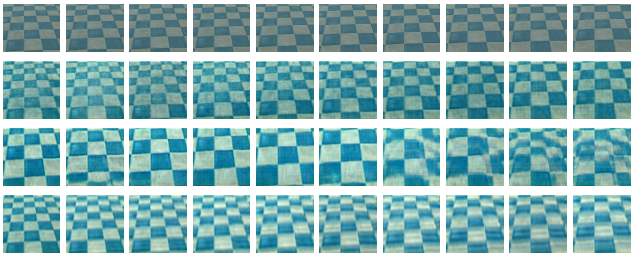

Figure 2. The comparison results for moving grid sequence. First row: 8 frames of the training sequence. Second row: 8 frames of the generated sequence using our method. Third row: 8 frames of the generated sequence using [9]. Fourth row: 8 frames of generated sequence using [5].

We use $l = 20$ steps of Langevin dynamics in each learning iteration, and the Langevin step size is set to 0.1. We use the Adam [16] optimizer with a fixed learning rate of 0.0002 and momentum term of 0.5. The algorithm is terminated after 500 iterations. More results can be found on the project page: `https://hthth0801.github.io/UCLA_files/wacv2019_project/main.html`

4.1. Qualitative experiments

Our method can be used to generate different dynamic patterns. Once the spatial-temporal generator network is learned, we randomly sample the latent factors from $\mathrm{N}(0, I)$, and then use the learned generator to transform the latent factors to the synthesized video sequence.

We perform two sets of qualitative experiments to visualize the learned spatial-temporal generator networks and compare with other popular video models in the literature.

Dynamic pattern synthesis. Most of the dynamic patterns exhibit some regularities in either the spatial or temporal domain. By utilizing various convolutional structures, we can synthesize realistic patterns even with a single training video. Specifically, we experiment with three model structures corresponding to the three versions of the generator models described in section 3.1,

(1) Model-1 corresponds to the model that is stationary

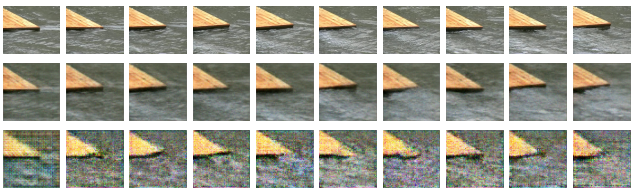

Figure 3. First row: bark water training sequence. Second row: frames from the generated sequences using our method. Third row: frames from the generated sequences using 3D GAN [31].

in both the spatial and temporal domains. We learn a 4-layer ConvNet, where there are 256, 128, 64 and 3 filters with the size of $4 \times 4 \times 4$ and a stride of 2 in 4 layers respectively. We use $8 \times 8 \times 4 \times 2$ latent factors to generate dynamic patterns.

(2) Model-2 corresponds to the model that is stationary in the temporal domain, but non-stationary in the spatial domain. We learn a 6-layer ConvNet, where the first layer has 512 filters with a size of $4 \times 4 \times 1$ and a stride of 1. The second layer has 64 filters with a size of $7 \times 7 \times 7$ and a stride of 1. There are 256, 128, 64 and 3 filters with a size of $4 \times 4 \times 4$ and a stride of 2 from layer 3 to layer 6. We use $1 \times 1 \times 8 \times 20$ latent factors to generate the dynamic patterns of the double length.

(3) Model-3 corresponds to the model that is non-stationary in both the spatial and temporal domains. We learn a 6-layer ConvNet, where the first layer has 512 filters with the size of $4 \times 4 \times 1$ and stride of 1. The second layer has 384 filters with the size of $7 \times 7 \times 7$ and stride of 1. There are 256, 128, 64 and 3 filters with the size of $4 \times 4 \times 4$ and stride of 2 from layer 3 to layer 6. We use $1 \times 1 \times 1 \times 20$ latent factors to generate dynamic patterns.

We use 64 frames for training in model-1 and model-2, and show various synthesis results in Figure 1.

Comparison. We compare our method with state space auto-regressive models [9, 5] for dynamic textures. Since their methods are based on frame-wise modeling, we set the latent dimension to 20 for each frame as suggested in their papers. Figure 2 shows one comparison result. It can be seen that frame-wise modeling can accumulate noises, rendering less sharp synthesis results as time evolves.

We also compare our method with 3D generative adversarial model [31], i.e., 3D GAN. We use the code provided on their webpage. Figure 3 shows one comparison result.

4.2. Quantitative experiments

To evaluate our method quantitatively, we perform four sets of experiments. We first apply the model-2 structure on a single training sequence for the interpolation task. Then we compare our model, i.e., 3D ABP, with the current state of the art generative adversarial model [31], i.e., 3D GAN, for recovery tasks. In addition, we perform multimodal representation learning and compare the prediction errors with the baseline method. For fair comparison with [31], we adopt the model-3 structure which assumes non-stationarity in both the spatial and temporal domains. For recovery tasks, we further use learned initialization to improve the original model, i.e., LI-ABP, and we also provide strong baseline models, i.e., 3D variational auto-encoder (3D VAE), based on the original variational auto-encoder [17, 25], and 3D wake sleep models (3D WS) based on [12]. The 3D VAE and 3D WS share the same generator network structure as our models and 3D GAN [31]. For the inference network for 3D VAE, 3D WS and our model (LI-ABP), it has the "mirror" structure as its generator network, with convolution and LeakyReLU (with ratio 0.2) to replace the deconvolution and typical ReLU used in the generator. For all five models, we set the latent factor dimension to 100.

Table 1. Interpolation errors for various training videos with different occlusion masks.

Methods	Consecutive Block		Random Block	
	LHD	ours	LHD	ours
flag	0.1359	**0.0392**	0.1111	**0.0300**
waterfall	0.2720	**0.2558**	0.2666	**0.2393**
waterstone	0.1899	**0.1321**	0.1717	**0.1251**
light	0.0728	**0.0725**	0.0586	**0.0400**
elevator	0.1043	**0.0768**	0.0735	**0.0463**

Interpolation of time-stationary patterns. For video sequences with repetitive patterns in the temporal domain, our method can efficiently learn such patterns by assuming $\mathbf{Z} = \mathbf{Z}(m, t)$ even on a single training sequence. We evaluate our method by interpolation. We consider two types of missing patterns. The first type is the consecutive block, in which we entirely block 8 consecutive frames in the middle of the training sequence. The second type is the random block, in which we entirely block 3 consecutive frames in 3 randomly chosen positions in the training sequence. Each interpolation experiment is performed on only one video training sequence. Note that the current 3D GAN model [31] cannot easily do this, because the spatial-temporal non-stationary structure is used in their paper and it lacks the inference mechanism. We compare our approach with the widely used Laplacian Heat Diffusion method (LHD) where we extend the 2D Laplacian kernel to 3D version, and we use 500 sweeps for the video interpolation with learning rate 0.5. Table 1 shows the recovery errors for various kinds of dynamic patterns and different missing patterns. The recovery error is defined as the per-pixel absolute difference between the ground truth video and the recovered video on the occluded pixels. Figure 4 shows the interpolation results on the consecutive block pattern. It can be seen that our method can get sharper and more realistic results.

Video recovery. For this task, we split the training/testing video sequences by about 4:1 in proportion, and randomly block the testing videos using different occlusion patterns. For adversarial trained network [31], we ap-

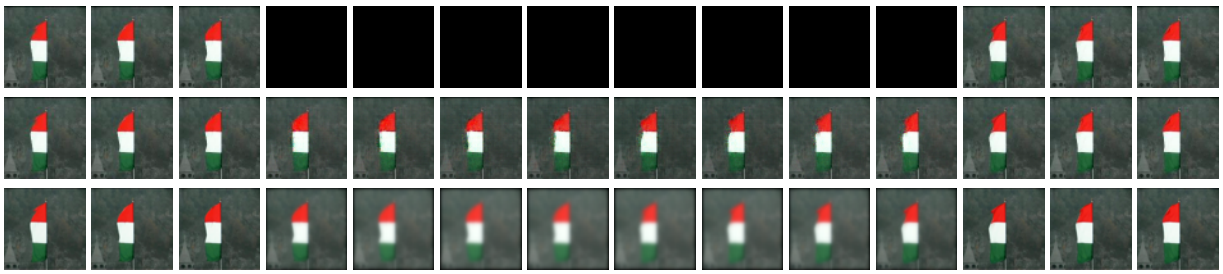

Figure 4. Interpolation results. The first row displays the occluded training sequence. The second row displays the interpolation results by our method. The third row shows the interpolation results by using Laplacian heat diffusion.

Videos		flag	elevator	grid	windmill	flower	traffic
M0.5	3D GAN [31]	0.3385	0.1679	0.3696	0.2151	0.1084	0.1697
	3D VAE [17]	0.1603	0.0485	0.2516	0.1170	0.0624	0.0943
	3D WS	0.2074	0.1345	0.2526	0.1341	0.0750	0.1142
	3D ABP	0.1370	0.0457	0.2106	0.1157	0.0662	0.0903
	LI-ABP	**0.1179**	**0.0456**	**0.1996**	**0.1143**	**0.0598**	**0.0898**
P20	3D GAN [31]	0.3380	0.1796	0.3542	0.2483	0.1160	0.1780
	3D VAE [17]	0.1511	0.0591	0.1981	0.1643	0.0686	0.1037
	3D WS	0.2003	0.1109	0.2221	0.1869	0.0783	0.1275
	3D ABP	0.1459	0.0562	0.2044	0.1621	0.0733	0.1001
	LI-ABP	**0.1286**	**0.0553**	**0.1914**	**0.1585**	**0.0655**	**0.0993**
B35	3D GAN [31]	0.3382	0.1803	0.4505	0.2626	0.1208	0.1740
	3D VAE [17]	0.1801	0.0591	0.2082	0.1851	0.0752	0.1012
	3D WS	0.2905	0.1060	0.2339	0.2462	0.0932	0.1237
	3D ABP	0.1778	0.0577	0.2159	0.1779	0.0831	0.0991
	LI-ABP	**0.1583**	**0.0570**	**0.2000**	**0.1758**	**0.0728**	**0.0965**

Table 2. Recovery errors for various incomplete testing videos with different occlusion masks.

ply the inferential mechanism powered by Langevin dynamic for direct comparison with our method. For 3D VAE and 3D WS, we use their own learned inference networks and iteratively impute the occluded pixels during inference stages [25].

We experiment on three types of occlusions: (1) salt and pepper mask (M) which covers roughly 50% (M0.5) of the pixels per-frame. (2) 2D patch occlusion (P) where we randomly place a 20×20 (P20) mask on each frame. (3) 3D block occlusion (B) where we randomly place a 35×35×20 block in the whole video. For the inference stage on occluded testing videos, we run the inference step for 200 iterations with step size 0.05 for Langevin dynamic. Table 2 shows recovery errors for various kinds of dynamic patterns and different occluding masks. It can be seen that our proposed ABP based methods show competitive or superior performance over baseline models, in particular, ABP with learned initialization outperform all models. ABP with leaned initialization further boost the performance over 3D ABP since the initializer learned can give us a more informative starting point of Langevin dynamics in the testing stage than one from simple Gaussian distribution.

Learning from incomplete videos. We also evaluate our methods on a much more difficult task: learning directly from incomplete videos while completing them during the learning stage. This task is generally infeasible for models that do not have an inference mechanism, because such models cannot easily borrow strength from other training videos. Our methods and the baseline methods can be easily adapted to this task with a simple modification in which we only consider the observed pixels for our objective function. To the best of our knowledge, there is no prior work on this task using video generative models, so we only compare our methods with the implemented 3D VAE and 3D WS as our baselines.

We again experiment on three types of occlusions for this task, including salt and pepper noise, 2D patch occlusion and 3D block occlusion as described above. Table 3 displays the recovery errors on various dynamic patterns for 3 different occlusion types. It can be seen that the inference-based methods are well suited for this task, and our methods are competitive and outperform baseline methods in terms of recovery errors. Compared to 3D ABP, if we use learned initialization, it tends to give better recovery results in most cases, the reason is that the initialization network can provide sensible starting point of latent factors transition, there-

Experiments		elevator	windmill	flag1	flamingo	flag2
M0.5	3D VAE [17]	0.0421	0.0442	0.0296	0.0478	0.0775
	3D WS	0.0423	0.0451	0.0300	0.0483	0.0787
	3D ABP	0.0415	**0.0412**	0.0283	0.0471	0.0772
	LI-ABP	**0.0410**	0.0418	**0.0279**	**0.0467**	**0.0739**
P20	3D VAE [17]	0.0422	0.0464	0.0375	0.0467	0.9746
	3D WS	0.0422	0.0479	0.0397	0.0457	0.0955
	3D ABP	**0.0415**	0.0439	**0.0363**	**0.0441**	0.0947
	LI-ABP	0.0416	**0.0434**	0.0369	**0.0441**	**0.0927**
B25X25X15	3D VAE [17]	0.0525	0.0773	0.0400	0.0603	0.1601
	3D WS	0.0487	0.0758	0.0384	0.0606	0.1591
	3D ABP	0.0476	0.0769	0.0389	**0.0589**	0.1592
	LI-ABP	**0.0463**	**0.0751**	**0.0370**	**0.0589**	**0.1582**

Table 3. Recover error for the various incomplete training videos based on different occlusion masks.

Experiments		Execuse	Goodby	Hello	How	Nice	Seeyou	Sorry	Thank	Time	Welcome
$V \rightarrow A$	CCA	0.1779	0.1828	0.1878	0.1894	0.1870	0.1889	0.1943	0.1842	0.1818	0.1821
	ours	**0.0179**	**0.0243**	**0.0199**	**0.0379**	**0.0210**	**0.0184**	**0.0306**	**0.0164**	**0.0301**	**0.0348**
$A \rightarrow V$	CCA	0.4269	0.4468	0.4326	0.6225	0.4549	0.4811	0.5945	0.4472	0.6044	0.5892
	ours	**0.1771**	**0.1860**	**0.1758**	**0.1735**	**0.1791**	**0.1747**	**0.1788**	**0.1713**	**0.1846**	**0.1743**

Table 4. Average prediction errors on missing modality for different phrases.

fore would not easy to trap in local modes compared to 3D ABP which updates the latent factors based on their current values. Note that we only ran 10 steps of Langevin in LI-ABP which is more efficient.

Multimodal shared representation learning. We further evaluate the shared representation learned for both video and audio modalities. Most of the existing methods [20, 28, 1], used spectrogram for audio representation, and require labels for both modalities, so they could hardly generate raw signals. In this paper, we directly use *raw* audio and video sequences as inputs and focus on the generation of the missing modality. Note that this task is in general infeasible for models without inference mechanism and not easy for variational methods which need elaborately designing the proper approximate shared posteriors. One feasible and well-known approach for this task is canonical correlation analysis (CCA) [11, 35] which finds the *linear* transformations of audio and video data to form a shared representation. Our model can be naturally seen as a *non-linear* version of CCA where two generator nets are trained to find non-linear transformations and latent factors Z are shared across both modalities during inference.

We experiment on OuluVS dataset [39] which includes audio and pre-extracted mouth ROI sequences for 20 persons speaking 10 phrases 5 times. We circularly add or delete mouth images to make each video clip contains 32 frames, each of size [64, 64]. We also fix the length of audios to be $48,000$ which equals the sample rate, and only keep the first sound channel for simplicity. We use Model-3 spatial-temporal generator structure for videos, and 4-layer 1D ConvNet for audios. For audio net, there are 512, 256, 128, 1 filters for each deconvolution layer, and the filter size is 20×1 of stride 10 for all layers. We also utilize batch normalization and ReLU non-linearity except the last layer where tanh is used. For training, we randomly choose 10 persons, each saying 10 phrases 5 times, and use corresponding audios and videos to learn our model and CCA. For testing, we randomly select 3 new persons to: (1) predict the mouth movement based on their audios only ($A \rightarrow V$), (2) predict the audio track based on their mouth movements only ($V \rightarrow A$). We run 200 steps of Langevin dynamics with stepsize 0.1 for the inference of our model. Table 4 shows the mean absolute per-pixel/signal prediction error on 10 different phrases. A lower error indicates a more powerful model for preserving the common information between two modalities.

5. Conclusion

This paper proposes to learn different types of generator networks from video sequences using the alternating back-propagation method. We show that our method can learn realistic models of dynamic patterns, recover incomplete video sequences. We further propose to use learned initialization for better recovery and we show the proposed method can help to learn shared representation over multiple modalities.

Acknowledgment

The work is supported by DARPA XAI project N66001-17-2-4029; ARO project W911NF1810296; ONR MURI project N00014-16-1-2007; and a Hikvision gift to UCLA.

References

[1] T. Baltrušaitis, C. Ahuja, and L.-P. Morency. Multimodal machine learning: A survey and taxonomy. *arXiv preprint arXiv:1705.09406*, 2017.

[2] A. B. Chan and N. Vasconcelos. Modeling, clustering, and segmenting video with mixtures of dynamic textures. *IEEE transactions on pattern analysis and machine intelligence*, 30(5):909–926, 2008.

[3] D. Chetverikov and R. Péteri. A brief survey of dynamic texture description and recognition. In *Computer Recognition Systems*, pages 17–26. Springer, 2005.

[4] R. Costantini, L. Sbaiz, and S. Susstrunk. Higher order svd analysis for dynamic texture synthesis. *IEEE Transactions on Image Processing*, 17(1):42–52, 2008.

[5] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto. Dynamic textures. *International Journal of Computer Vision*, 51(2):91–109, 2003.

[6] K. Fragkiadaki, S. Levine, P. Felsen, and J. Malik. Recurrent network models for human dynamics. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4346–4354, 2015.

[7] 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*, pages 2672–2680, 2014.

[8] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2247–2253, December 2007.

[9] T. Han, Y. Lu, S.-C. Zhu, and Y. N. Wu. Alternating Back-Propagation for generator network. In *AAAI*, 2017.

[10] T. Han, X. Xing, and Y. Wu. Learning multi-view generator network for shared representation. *preprint*, 2018.

[11] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural computation*, 16(12):2639–2664, 2004.

[12] G. E. Hinton, P. Dayan, B. J. Frey, and R. M. Neal. The" wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161, 1995.

[13] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[14] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2013.

[15] G. Karol, I. Danihelka, A. Graves, D. Rezende, and D. Wierstra. Draw: a recurrent neural network for image generation. ICML, 2015.

[16] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[17] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *ICLR*, 2014.

[18] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. In *Proceedings of the 33rd International Conference on Machine Learning (icml 2016)*, 2016.

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

[20] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng. Multimodal deep learning. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 689–696, 2011.

[21] A. V. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1747–1756, 2016.

[22] R. Péteri, S. Fazekas, and M. J. Huiskes. DynTex : a Comprehensive Database of Dynamic Textures. *Pattern Recognition Letters*, doi: 10.1016/j.patrec.2010.05.009. http://projects.cwi.nl/dyntex/.

[23] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

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

[25] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In T. Jebara and E. P. Xing, editors, *ICML*, pages 1278–1286. JMLR Workshop and Conference Proceedings, 2014.

[26] T. Salimans, D. Kingma, and M. Welling. Markov chain monte carlo and variational inference: Bridging the gap. In *Proceedings of The 32nd International Conference on Machine Learning*, pages 1218–1226, 2015.

[27] N. Srivastava, E. Mansimov, and R. Salakhutdinov. Unsupervised learning of video representations using lstms. In *ICML*, pages 843–852, 2015.

[28] N. Srivastava and R. R. Salakhutdinov. Multimodal learning with deep boltzmann machines. In *Advances in neural information processing systems*, pages 2222–2230, 2012.

[29] D. Tiwari and V. Tyagi. A novel scheme based on local binary pattern for dynamic texture recognition. *Computer Vision and Image Understanding*, 150:58–65, 2016.

[30] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4489–4497, 2015.

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

[32] X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2794–2802, 2015.

[33] Y. Wang and S.-C. Zhu. A generative method for textured motion: Analysis and synthesis. In *European Conference on Computer Vision*, pages 583–598. Springer, 2002.

[34] J. Xie, S.-C. Zhu, and Y. N. Wu. Synthesizing dynamic patterns by spatial-temporal generative convnet. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7093–7101, 2017.

[35] X. Xing, K. Wang, T. Yan, and Z. Lv. Complete canonical correlation analysis with application to multi-view gait recognition. *Pattern Recognition*, 50:107–117, 2016.

[36] Y. Xu, Y. Quan, H. Ling, and H. Ji. Dynamic texture classification using dynamic fractal analysis. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1219–1226. IEEE, 2011.

[37] T. Xue, J. Wu, K. Bouman, and B. Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2016.

[38] X. You, W. Guo, S. Yu, K. Li, J. C. Príncipe, and D. Tao. Kernel learning for dynamic texture synthesis. *IEEE Transactions on Image Processing*, 25(10):4782–4795, 2016.

[39] G. Zhao, M. Barnard, and M. Pietikainen. Lipreading with local spatiotemporal descriptors. *IEEE Transactions on Multimedia*, 11(7):1254–1265, 2009.