applied sciences

MDPI

Article

Sequence-to-Sequence Video Prediction by Learning Hierarchical Representations

Kun Fan, Chungin Joung and Seungjun Baek *

Department of Computer Science and Engineering, Korea University, Seoul 02841, Korea; tankpig@korea.ac.kr (K.F.); whooznext@korea.ac.kr (C.J.)

* Correspondence: sjbaek@korea.ac.kr

Received: 20 October 2020; Accepted: 20 November 2020; Published: 23 November 2020

Abstract: Video prediction which maps a sequence of past video frames into realistic future video frames is a challenging task because it is difficult to generate realistic frames and model the coherent relationship between consecutive video frames. In this paper, we propose a hierarchical sequence-to-sequence prediction approach to address this challenge. We present an end-to-end trainable architecture in which the frame generator automatically encodes input frames into different levels of latent Convolutional Neural Network (CNN) features, and then recursively generates future frames conditioned on the estimated hierarchical CNN features and previous prediction. Our design is intended to automatically learn hierarchical representations of video and their temporal dynamics. Convolutional Long Short-Term Memory (ConvLSTM) is used in combination with skip connections so as to separately capture the sequential structures of multiple levels of hierarchy of features. We adopt Scheduled Sampling for training our recurrent network in order to facilitate convergence and to produce high-quality sequence predictions. We evaluate our method on the Bouncing Balls, Moving MNIST, and KTH human action dataset, and report favorable results as compared to existing methods.

Keywords: convolutional neural network; hierarchical features; long short-term memory; recurrent neural network; video prediction

1. Introduction

The goal of video prediction is to predict future frames given past observations. Being able to model and predict the future is essential to many applications of machine learning and computer vision, such as human pose estimation and recognition [1], pedestrian detection and tracking [2], weather forecasting [3]. In recent years, a number of video prediction methods have been proposed [4–7], driven by the success of convolutional neural networks (CNN) [8–12], recurrent neural networks (RNN) [13], and generative adversarial networks (GAN) [14]. However, video prediction remains a challenging task in computer vision, because of not only the uncertain nature of future events, but also the unpredictability of spatio-temporal dynamics as well as the high dimensionality.

Recent approaches for video prediction focus on separating representations that exhibit different temporal dynamics. For example, DRNet [15] and MCnet [16] propose to decompose video frames into stationary and time-varying parts, and use RNNs to model the dynamics of time-varying components. Other recent works [17–19] consider extracting high-level features from input frames, then predict the temporal evolution of such high-level features, and use generative models to reconstruct future frames. However, such latent representations to be decomposed are often chosen manually. The aforementioned methods require the estimation of high-level structures which needs domain-specific knowledge or relies on heuristics, e.g., joint locations in human poses [17] or temporal stationarity of frames [15,16]. Thus, it would be desirable if (i) hierarchical features can be *automatically*

learned from the input video; (ii) we can learn multiple, possibly more than two, levels of hierarchical features in a systematic way so as to fully exploit information in the video.

In this paper, we propose an end-to-end trainable neural network for video prediction called Hierarchical Recurrent Predictive Auto-Encoder (HRPAE). We consider hierarchical cascaded/multi-layer CNNs encoder-decoder incepted by RNN network architecture. Features output from different CNN layers will exhibit different temporal dynamics. Thus, we propose to use multiple Convolutional LSTMs (ConvLSTMs) [3] to separately model sequential structures from multiple levels of hierarchy of features. The key idea is to automatically and separately capture temporal dynamics of hierarchical features without any prior information on the input video. To facilitate the generation of realistic frames, we propose a recurrent variation of the skip-connection architectures inspired by U-Net [20] or Hourglass networks [21]. Specifically, hierarchical features output at each encoder layer skip the "bottleneck" formed at encoder-decoder network, and are fed to ConvLSTMs which combine hidden variables from previous frames, and propagate the updated/predicted feature maps to the decoder layer, which we call skip-and-update. The proposed architecture enables the encoder, decoder CNNs and ConvLSTMs to share useful spatio-temporal information at multiple levels of hierarchy, which helps producing sharp output frames.

Through experiments we show that HRPAE is able to obtain high quality results on the Bouncing Balls [22–24], the Moving MNIST (MMNIST) [4] and the KTH human action dataset [25]. With the Bouncing Balls dataset, experiments show that HRPAE is capable of modeling complex dynamics associated with the physical states of balls with a high accuracy. The normalized error in the ball positions are shown to be within 2% of the frame size with cosine similarity in the velocity of balls exceeding 0.97 over 10 frames of predictions. Experimental results on the MMNIST show that HRPAE is able to generate sharp predictions, and outperforms existing schemes in Mean Square Error (MSE), and achieves high Peak Signal-to-Noise Ratio (PSNR) [5] and Structural Similarity Index (SSIM) [26]. The experiments on MMNIST and KTH human action dataset show that our model better captures the details of digit shapes and human motions, as well as produces more accurate video predictions as compared to previous methods.

2. Related Work

A number of video prediction approaches [6,7,19,27–33] have been proposed in recent years. The work of [4] introduced the sequence to sequence LSTM model originated from language modeling [34,35] in order to predict video frames. Shi et al. [3] extended the LSTM to ConvLSTM by introducing the convolutional operation in the recurrent transitions to better capture the spatiotemporal correlations for precipitation nowcasting. Finn et al. [36] found transforming pixels from the previous frames is able to better capture object motion and produce more accurate video predictions and developed an unsupervised action-conditioned video prediction model that estimates the distribution over pixel motion from previous frames. Lotter et al. [37] extended the ConvLSTM to build neural networks using predictive coding to predict future frames. MCnet [16] and DRnet [15] proposed decomposition-based approaches such that, each frame is decomposed into content and motion, and then is fed into separate encoders. Other methods [38–40] consider modelling the individual dynamics of decomposed objects, and propose to decompose frames into those of separated objects, and to model the motion of each object. The work of [17,18] proposed to use high-level features (e.g., human joint landmarks) to model the dynamics of motion, while the method by [19] focuses on predicting high-level features. PredRNN++ [41] proposed Causal LSTMs with cascaded dual memories to model short-term video. Pan et al. [42] proposed to generate video frames from a single semantic map. Wang et al. [43] developed a point-to-point network which generates intermediate frames given the start and end frames.

Besides the methods mentioned above, the GAN framework has been increasingly used in video generation or prediction [44–47]. The work of [5] applied the adversarial training method to a multi-scale architecture to obtain sharp frames for prediction. A dual-motion GAN framework

is proposed in [48] by fusing the future-flow and future-frame. The work of [49] combined GAN framework and variational autoencoder to get the stochastic adversarial video prediction model. In this paper, we present a novel hierarchical architecture for video prediction which exploits the hierarchical structures of spatiotemporal features.

3. Methods

Given a sequence of input video frames of length T, our goal is to learn a function that maps the input frames into K realistic future video frames. We denote the ground truth video frame at time t by $\mathbf{x}_t$, and denote the sequence of input frames by $\mathbf{x}_{1:T}$. The output frame at time t is denoted by $\hat{\mathbf{x}}_t$ for $i = 2, \cdots, T + K$. We would like to generate predictions $\hat{\mathbf{x}}_{T+1:T+K}$ which well-approximates the ground-truth future frames $\mathbf{x}_{(T+1):(T+K)}$.

3.1. Model

Overview. The proposed architecture makes use of the hierarchy of representations obtained from cascaded/multi-layer CNNs. Features output from different layers of cascaded CNNs capture visual information at different levels: say in object detection, high level features may represent object identities like cars, whereas mid level (resp. low level) features may represent shapes like circles (resp. edges). We posit that features at different levels tend to exhibit different temporal dynamics. For example, features representing a walking person as a whole will change slowly over time as compared to those of the motions of the person's limbs. Separating and capturing features at multiple levels of hierarchy is useful, e.g., capturing high-level features makes the prediction task simpler, whereas capturing low-level features helps generating realistic frames. We thus propose to use multiple ConvLSTMs, each of which is dedicated to model the temporal dynamics of each of multiple levels of CNN features. Our goal is to automatically capture sequential structures of low-, mid-, and high-level features with ConvLSTMs, and is to generate future frames by combining hierarchical features estimated by the ConvLSTMs. HRPAE consists of three modules: an Encoder CNN, ConvLSTMs and a Decoder CNN. These modules are used recurrently over the time horizon. An overview of 4-level HRPAE is shown in Figure 1.

Hierarchical Recurrent Predictive Auto-Encoder. The HRPAE architecture consists of three parts: a multi-layer Encoder CNN (EncCNN) with pooling layers, four ConvLSTMs and a multi-layer Decoder CNN (DecCNN) with up-sampling convolutional layers as shown in Figure 1. The EncCNN is used to extract hierarchical CNN features of different levels from a frame. Four parallel ConvLSTMs are used to model the dynamics of hierarchical (e.g., low-, mid-low-, mid-high- and high-level) features of the input sequence and DecCNN is used to generate predictions from the estimated hierarchical features. During training, our model computes context features by first obtaining different levels of features from EncCNN. Next, each of the ConvLSTMs receives the corresponding level of features to compute the hidden states given the previous hidden states. DecCNN combines the outputs from ConvLSTMs of corresponding level and the previous layer, and sequentially propagates the context information. The last layer of decoder network is CNN with 1×1 convolutional layer to produce output consistent with the frame size. The output from the last layer is fed back into the EncCNN to generate the next frame. The equations representing the model in Figure 1 are given as follows:

$$
\begin{aligned}
&(F_t^1, F_t^2, F_t^3, F_t^4) = \text{EncCNN}(\mathbf{x}_t), \\
&(H_t^m, C_t^m) = \text{ConvLSTM}_m(F_t^m, C_{t-1}^m),\ m = 1, 2, 3, 4, \\
&\hat{\mathbf{x}}_{t+1} = \text{DecCNN}(H_t^1, H_t^2, H_t^3, H_t^4),
\end{aligned}
$$

where F_t^1 to F_t^4 represent from the low- to high-level CNN features of the t-th frame $\mathbf{x}_t$ extracted by EncCNN. H_t^m and C_t^m represent the hidden and the cell states of level-m ConvLSTMs at t-th frame. Our model can be expanded in terms of the number of levels, by adding the corresponding CNN layers and ConvLSTMs to the model.

Following the VGG structure [9], each block of the EncCNN consists of two successive 3×3 convolutional layers with Batch Normalization [50] and leaky ReLU activation function. The max pooling layer is placed at the end of each block. The DecCNN can be viewed as mirrored version of EncCNN; the difference is that we use bilinear interpolation to upsample the CNN feature maps by a factor of 2. Finally, a 1×1 convolutional layer is used as the last block of the DecCNN. The ConvLSTMs are equivalent to standard convLSTM except that we add Layer Normalization [51] after each convolutional layer. The motivation is that Layer Normalization is effective at stabilizing the hidden state dynamics in recurrent networks, and thus accelerates the training time.

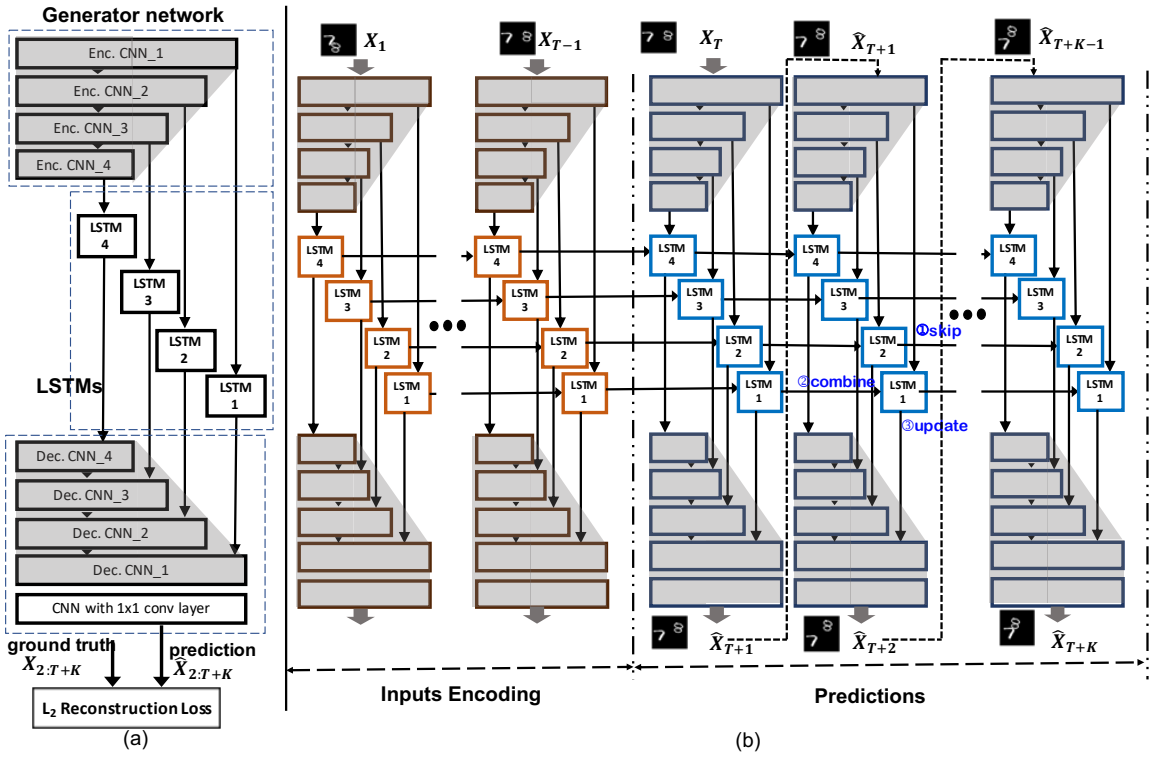

Figure 1. (**a**) Overview of the 4-level Hierarchical Recurrent Predictive Auto-Encoder (HRPAE) architecture. (**b**) Network unrolled over time horizon.

Skip-and-Update. The HRPAE is inspired by skip connections such as "U-Net" structure [20] (in Figure 1a, the "U"-shape structure is rotated clockwise, and appears as "C"-shape). U-Net consists of multi-layer CNNs with skip connections between the same levels of encoding and decoding layers. Such connections alleviate the problem of "information bottleneck" [52] in encoder-decoder networks. That is, skip connections enable sharing of low level information between encoding and decoding layers, which is essential in reconstructing details at the target images in image translation tasks. However, the difference in our network is that skip connections are intercepted by ConvLSTMs. The features of multiple levels of hierarchy from the current frame passed by encoding layer are combined with the latent variables of the previous frame; as a result, the ConvLSTMs output *updated* information, which is essentially a *predicted* feature map, to the decoding layer. Such sharing (updated) information between encoder and decoder layers are useful for prediction, e.g., for preserving the continuity of motion when the next frame is highly correlated with the current one. The skip-and-update in the prediction network is depicted in Figure 1b. In summary, the main purpose of skip-and-update is (i) to skip the encoder bottleneck to retain low- and mid-level information, (ii) to update such information capturing the sequential structure of each hierarchy, and propagate it to the decoder.

3.2. Training

Scheduled Sampling. RNNs are difficult to train, because early mistakes in the sequence prediction can be accumulated over the remaining prediction process. The issue is studied in Scheduled Sampling [53] which proposes to *randomly* sample either the current prediction or the ground-truth as input to the model during training. This makes the model robust to prediction mistakes during inference, as the model is gradually trained to correct such mistakes [36].

We use Scheduled Sampling to train HRPAE model as follows. We initialize the sampling probability to 0, and gradually increase the probability by constant step δ after each iteration during training. In the prediction process of Figure 1b, we randomly select either the ground-truth or generated video frames according to the probability, and feed it as input to HRPAE. As a result, HRPAE gradually learns how to recover from early mistakes in predictions with the increasing sampling probability, and is able to generate robust predictions during testing.

Loss function. We not only want to generate realistic frames, but also wish the generated frames to be continuous and capture the motion of objects fluently. To that end, we minimize the Mean Squared Error (MSE) reconstruction loss denoted by L_2 to enforce the generated frames to be close to the ground truth frames in the L_2 sense:

$$L_2(G) = \frac{1}{2} \sum_{i=2}^{T+K} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|^2 \tag{1}$$

where we sum up the pixel-wise L_2 frame loss over the predicted frames. Experimental results show that by training HRPAE with respect to Equation (1), our model can generate coherent video frames without an additional loss regularizer. The experimental results show that, by using the MSE loss, our model is able to generate sharp and coherent frames.

4. Experiments

In our experiments, we use 3×3 convolutional kernel, Batch Normalization, and ConvLSTMs. The generator network G is trained to predict $K = 10$ future frames conditioned on the first $T = 10$ input frames. We compare HRPAE with state-of-the-art baselines for video prediction: DRnet, MCnet and PredRNN++ by using either the codes released by the authors or re-implementations. We use the MSE, PSNR (PSNR computes the peak signal-to-noise ratio between the ground-truth and corresponding predicted video frame. The higher the PSNR, the better the quality of the predicted video frame) and SSIM (SSIM measures the structure similarity index between the ground-truth and corresponding predicted video frame from three visual impacts: luminance, contrast and structure, similar to PSNR, higher SSIM scores represent better predicted image quality) [26] as the evaluation metric.

We consider three datasets in order to demonstrate the capability of HRPAE in automatically capturing dynamics of features at multiple levels of hierarchy, but to varying degrees depending on the dataset:

- **Bouncing Balls dataset** contains videos of 4 balls randomly bouncing off the walls and the other balls. The emphasis is mostly on capturing the complex dynamics of *high*-level features in an interactive enviroment, e.g., the center positions and velocity of the balls.
- **Moving MNIST (MMNIST) dataset** contains 2 hand-written digits moving independently. The focus is on tracking the dynamics of both *high*-level (digit position) and *mid*- to *low*-level features (digit shapes).
- **KTH action dataset** contains several classes of human actions. For predicting future frames, one needs to capture high-level features such as the person's positions; but the focus is largely on *low*-level features representing details of body parts.

4.1. Bouncing Balls

The Bouncing Balls dataset simulates 4 balls bouncing within the frame. It is challenging to predict future frames of bouncing balls because of the underlying physical dynamics and interactions, e.g., the balls can bounce off walls, and may collide with each other and change directions. The purpose of this experiment is to evaluate the capacity of HRPAE of predicting object trajectories in an interactive environment.

Implementation. Following the settings in [23,40], we generated 10,000 training and 1000 testing sequences with 128×128 frame size where each sequence has 30 frames. The batch size is set to 8, and the default HRPAE (as shown in Figure 1) is trained for 10^5 iterations (about 80 epochs) by randomly sampling 20 frames. The rate of Scheduled Sampling is gradually increased to 1 during the first 50,000 iterations. We trained PredRNN++ under the same experimental settings, whereas the first layer of PredRNN++ is expanded to 64 channels; as compared to MMNIST, we doubled the patch size in order to maintain the width and height of input frames, because the frame height and width are 128 pixels.

Results. We present qualitative results with the Bouncing Ball dataset in Figure 2. We observe that HRPAE can successfully predict how balls interact with their environment, e.g., colliding with other balls and bouncing off walls. As shown in Figure 2, HRPAE correctly predicts that the balls will collide, but will not be merged/entangled, unlike the digits in MMNIST dataset. Furthermore, HRPAE is able to predict complex interactions such as the balls bouncing off the wall and then colliding with another ball as shown in the 6th row of Figure 2.

Next we present quantitative results with the Bouncing Ball dataset. As shown in Table 1, HRPAE shows the superior performance in the overall MSE and PSNR per frame; this is achieved with the model size less than one-third of that of PredRNN++.

Table 1. Results with the Bouncing Balls dataset.

Methods	MSE	PSNR	Parameters
PredRNN++	27.22	27.87	31,565 K
HRPAE	20.25	29.36	9391 K

Next, we evaluate the prediction accuracy, measured in terms of ball center positions. HRPAE only outputs predicted frames, but does not explicitly compute the ball positions. Thus, we perform image processing such as binary erosion [54] to extract ball center positions, denoted by c_t at time step t, from the predicted frames. In Figure 3a we show the average error in the ball center positions in the L_2 distance. The results show HRPAE is able to predict the ball positions with high accuracy. The positional error is observed to be the within 2% of the frame size on average across 10 frames (less than 2 pixels).

Following the experimental settings in [23,40], the normalized velocity, defined as $v_t = c_{t+1} - c_{t-1}$, of the balls, and the differences in L_2 norm between the ground-truth and the extracted ball velocities are shown in Figure 3b. The cosine similarity between the ground truth and predicted velocities is presented in Figure 3c. We observe that HRPAE outperforms PredRNN++ method with the smaller model size, e.g., the average cosine similarity of our method is 0.973, while that of PredRNN++ is 0.962. The results indicate that HRPAE excels at capturing high-level dynamics of physical states associated with the motion of the balls in a complex interactive environment.

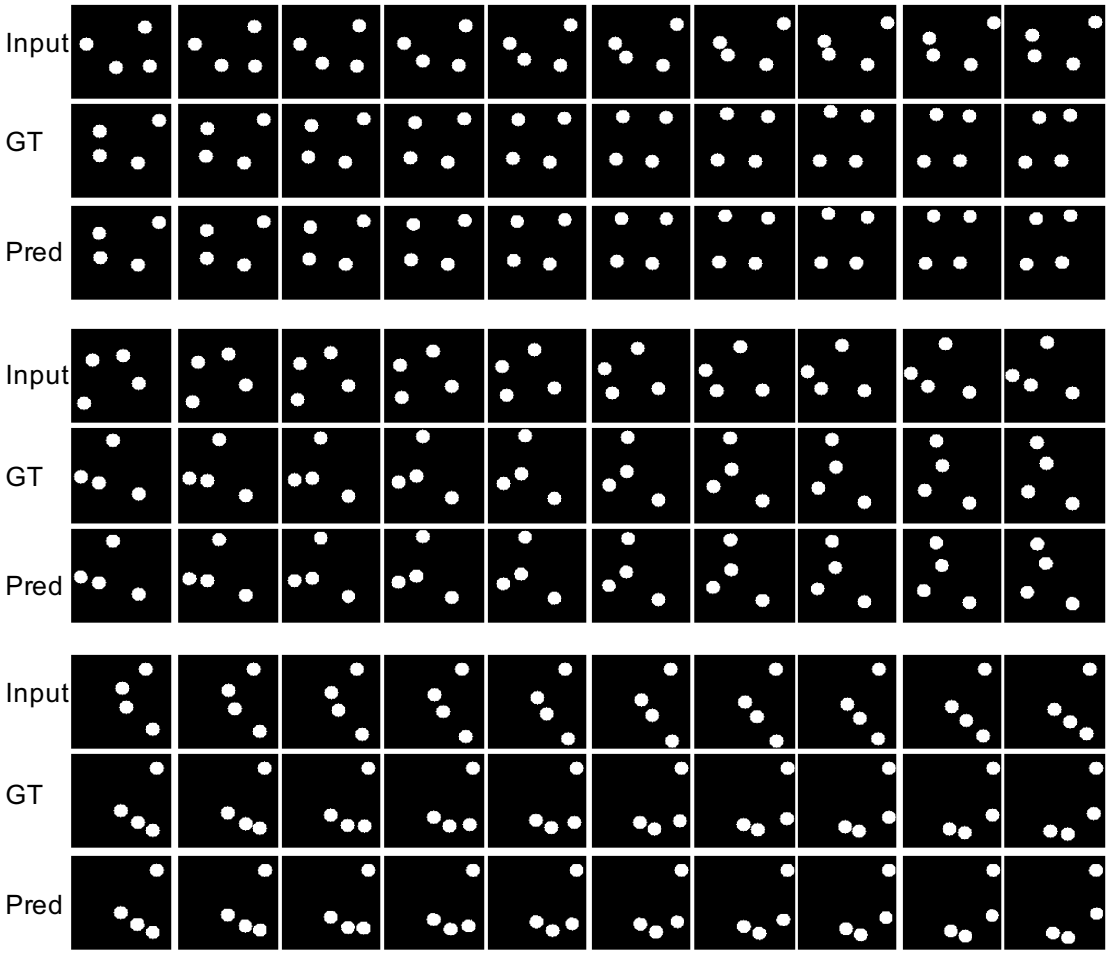

Figure 2. Qualitative results with Bouncing Balls dataset.

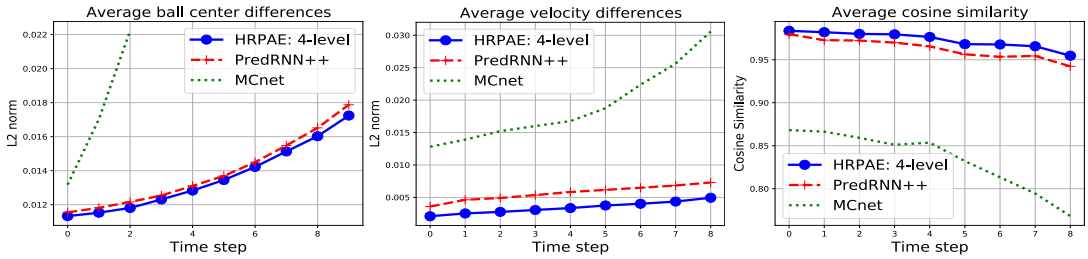

Figure 3. Experimental results on Bouncing Balls. This figure shows the ball center differences, velocity differences and cosine similarity.

4.2. Moving MNIST

The MMNIST dataset consists of two randomly moving hand-written digits in a gray-scale frame. The MMNIST contains 10,000 sequences of 20 frames. Each sequence shows a pair of digits bouncing around in a 64×64 frame. The digits can bounce off from the boundary, and may overlap with each other. We randomly split the dataset into a fixed 9000 training and validation set, and 1000 testing set. For all methods, we use the unmodified/original version of MMNIST dataset from [4], publicly available at [55] (meanwhile, in DRnet [15] and PredRNN++ [41], the authors synthesized their own Moving MNIST dataset for training and testing).

Implementation. The MMNIST image pixel values are re-scaled to the range $[0, 1]$. All compared models are trained to make 10 future predictions given the first 10 input frames. Our model is trained under similar training settings as the PredRNN++. We use the Adam optimizer [56] with a constant learning rate of 0.001 with the batch size 8. Our model is trained for 80,000 iterations (about 72 training epochs). During the first 50,000 iterations, the probability of Scheduled Sampling is gradually increased to 1.

Ablation Study. HRPAE consists of multiple levels of CNN blocks and convLSTMs. Our experiment contains the ablation study by varying the number of levels, and compare the following three variants.

- The default 4-level HRPAE as shown in Figure 1a with the output channel sizes given by 32, 64, 128 and 256.
- The 3-level HRPAE which contains the first 3 levels of CNN blocks and convLSTMs from the network in Figure 1a (e.g., uses EncCNN 1–3, LSTM 1–3, and DecCNN 1–3) with the output channel sizes given by 32, 64 and 128.
- The 2-level HRPAE which consists of the first 2 layers of CNN blocks and convLSTMs with output channel sizes given by 32 and 64.

Results. The experimental results are summarized in Table 2 which shows the comparison with the state-of-the-art models in terms of the MSE, PSNR and SSIM metrics. We observe that 4-level HRPAE outperforms DRNet, MCnet, PredRNN++ and gives the best performance. All three variants of our model show reasonable results on the testing MMNIST dataset. We find that variants with a higher number of hierarchical levels exhibit better performances. For instance, the averaged MSE loss decreases from 66 to 49 by increasing the number of levels from 2 to 4. The results demonstrate that it is important to exploit sequential structures of features at various levels of hierarchy. The frame-wise average losses over 10 prediction steps are shown in Figure 4. As shown in the figure in which we excluded the results for DRNet and MCnet results, 4-level HRPAE achieves the best results. Also 3-level HRPAE performs comparable to PredNet++. The boost in the performance is greater when the number of levels of HRPAE is increased from 2 to 3 than from 3 to 4. Thus, 3-level HRPAE achieves a good trade-off between the model size and performances; this aspect will be discussed in follow-up research.

Table 2. Results with the Moving MNIST dataset. The metrics are averaged over 10 predictions.

Method	MSE	PSNR	SSIM
DRNet [15]	207.9	13.2	0.42
MCnet [16]	172.2	13.9	0.66
PredRNN++ [41]	50.66	20.12	0.8984
2-level HRPAE	66.36	18.75	0.839
3-level HRPAE	52.91	19.99	0.891
4-level HRPAE	49.67	20.47	0.8987

In Figure 5, we present examples of the predicted testing frames for the discussion of qualitative results. As shown in the figure, even though the digits get entangled with each other, our model is able to make accurate predictions. For example, in the left-side example of Figure 5, the two digits of "2" are overlapped with each other in the beginning of the target future sequence. Our model is able to maintain the shape of digits and make accurate predictions of the locations. This proves our models because even a small one, e.g., 2-level HRPAE, is able to preserve the distance information between digits. Similar observations can be made for the right-side example.

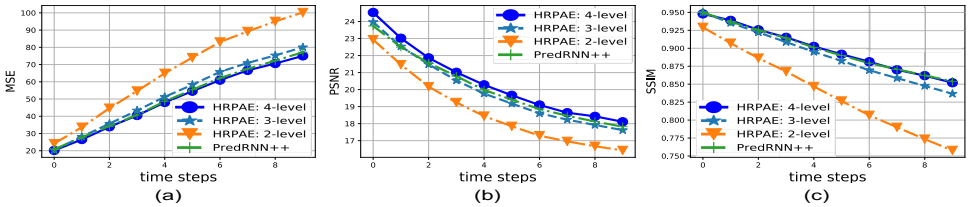

Figure 4. Frame-wise performances over the testing MMNIST dataset. (**a**–**c**) represent the averaged MSE, PSNR and SSIM evaluation metrics over the next 10 predicted frames, respectively.

Figure 5. Qualitative results on MMNIST. Our model is able to generates sharp frames during all the time steps, and accurately predicts the digit locations.

Resource Efficiency. Next we discuss the resource efficiency of models, for which we compare the model sizes. The number of trainable parameters is compared on Table 3. The 2nd to the 5th row show the total number of convolutional weights in corresponding LSTMs. The 6th row shows the total number of convolutional weights. Overall, our models use a smaller number of trainable weights when compared with PredRNN++. For instance, our model only has about 74 K convolutional weights in the LSTM-1. By contrast, PredRNN++ requires 5921 K convolutional weights in its first layer of Causal LSTM unit, as shown in the 1st row of Table 3. With only 15% of the weights of PredRNN++, 3-level HRPAE is able to achieve comparable performances, which proves the effectiveness of our model. Note that our training settings are identical to the default settings of PredRNN++. The efficiency is attributable to the model's capability of automatically learning hierarchical structures, e.g., low-level features for generating the details of digit shapes, and high-level features for tracking digit positions.

Table 3. Comparison of the number of convolutional weights of LSTMs at different levels, bias excluded.

	PredRNN++	3-Level	4-Level
LSTM 1	5921 K	74 K	74 K
LSTM 2	3359 K	295 K	295 K
LSTM 3	2261 K	1180 K	1180 K
LSTM 4	2261 K	0	4719 K
Total	15,441 K	2313 K	9391 K
Ratio	1	0.15	0.61

4.3. KTH Action Dataset

The KTH dataset consists of 600 videos of 25 humans performing 6 types of actions (boxing, hand-clapping, hand-waving, jogging, running, and walking) in 4 scenarios. Each video has the duration of 4 seconds on average.

Implementation. We use the same settings of MCnet [16] as follows. We use person 1–16 for training and 17–25 for testing. We gather 1525 video clips of various length for training and 2819 video clips (30 frames) for testing.

Then each frame is resized to a resolution of 120×120 where the pixel value is re-scaled to the range of $[-1, 1]$. We train our network and baselines by feeding the first 10 frames as input, and making the models to predict the next 10 frames. The batch size is set to 8 where the training is done for 2×10^5 iterations with Adam optimizer with constant learning rate of 0.001. The rate of Scheduled Sampling is gradually increased to 1 during the first 10^5 iterations. At testing time, the prediction horizon is extended to 20 future frames. For fair comparison, we have trained MCnet, PredRNN++ and HRPAE under the same experimental settings.

Results. In this section, we present quantitative results comparing HRPAE with baselines. Table 4 shows the overall prediction results compared with PredRNN++ and MCnet in terms of the PSNR and SSIM. We observe that 4-level HRPAE outperforms PredRNN++ and MCnet. Figure 6 shows the frame-wise losses over 20 prediction frames, which shows our model performs consistently better than the baseline models at all the time steps. Figure 7 presents qualitative prediction examples, which shows that our model is able to make accurate predictions about the human motion trajectories, and generates sharp video frames.

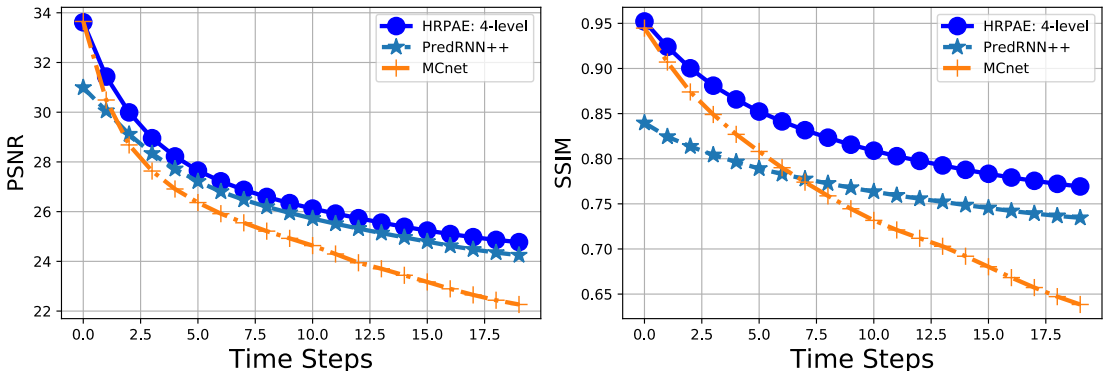

Figure 6. Frame-wise performance on the KTH dataset.

Table 4. Results with the KTH human action dataset. The metrics are averaged over 20 predicted frames.

Methods	PSNR	SSIM
MCnet	25.439	0.756
PredRNN++	26.400	0.772
HRPAE: 4-level	27.029	0.828

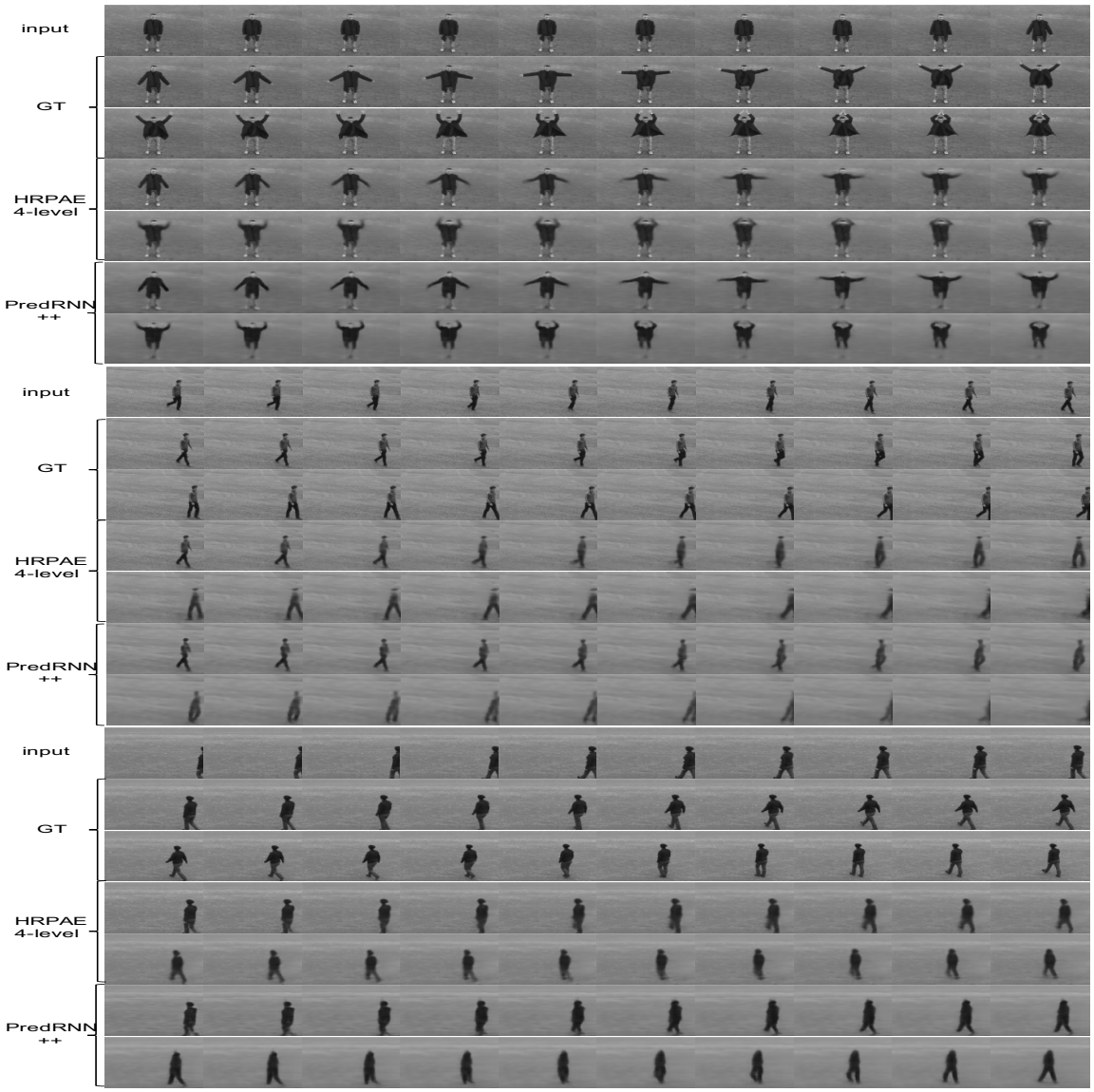

Figure 7. Qualitative KTH prediction examples.

5. Conclusions and Future Work

In this paper, we propose a Hierarchical Recurrent Predictive Auto-Encoder (HRPAE) for video prediction. The idea is to automatically separate different levels of CNN representations, and to use ConvLSTMs to model the temporal dynamics at each level of hierarchical features combined with skip connections for sharing feature information between encoders and decoders. Experimental results on MMNIST, Bouncing Balls and KTH human action datasets show that HPRAE is able to generate highly accurate predictions in terms of high-level structures such as locations, velocity, interaction of objects, as well as low-level details such as pixel values. Our future work includes improving the performance of our model by incorporating the GAN framework for generating realistic frames, and enhancing our model for processing higher resolution videos.

Author Contributions: Conceptualization, K.F.; methodology, K.F.; software, K.F.; validation, K.F. and C.J.; formal analysis, K.F.; investigation, K.F.; resources, K.F. and C.J.; data curation, K.F.; writing–original draft preparation, K.F.; writing–review and editing, S.B.; visualization, K.F. and S.B.; supervision, S.B.; project administration, S.B.; funding acquisition, S.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ICT Creative Consilience program(IITP-2020-0-01819) supervised by the IITP(Institute for Information & communications Technology Planning & Evaluation), and also by National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP) (No. 2018R1A2B6007130).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, C.; Zhang, Z.; Lee, W.S.; Lee, G.H. Convolutional Sequence to Sequence Model for Human Dynamics. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018.
2. Foxlin, E. Pedestrian tracking with shoe-mounted inertial sensors. *IEEE Comput. Graph. Appl.* **2005**, *25*, 38–46. [PubMed]
3. Shi, X.; Chen, Z.; Wang, H.; Yeung, D.Y.; Wong, W.K.; Chun Woo, W. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 802–810.
4. Srivastava, N.; Mansimov, E.; Salakhudinov, R. Unsupervised learning of video representations using LSTMs. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 843–852.
5. Mathieu, M.; Couprie, C.; LeCun, Y. Deep multi-scale video prediction beyond mean square error. In Proceedings of the International Conference on Learning Representations, San Juan, PR, USA, 2–4 May 2016.
6. Oh, J.; Guo, X.; Lee, H.; Lewis, R.L.; Singh, S. Action-conditional video prediction using deep networks in atari games. *Adv. Neural Inf. Process. Syst.* **2015**, 28, 2863–2871.
7. Kalchbrenner, N.; van den Oord, A.; Simonyan, K.; Danihelka, I.; Vinyals, O.; Graves, A.; Kavukcuoglu, K. Video pixel networks. In Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; Volume 70, pp. 1771–1779.
8. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. *Commun. ACM* **2012**, *60*, 84–90.
9. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. In Proceedings of the International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015.
10. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
11. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
12. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
13. Schuster, M.; Paliwal, K.K. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.* **1997**, *45*, 2673–2681.
14. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. *Adv. Neural Inf. Process. Syst.* **2014**, 2, 2672–2680.
15. Denton, E.L. Unsupervised learning of disentangled representations from video. *Adv. Neural Inf. Process. Syst.* **2017**, 1, 4414–4423.
16. Villegas, R.; Yang, J.; Hong, S.; Lin, X.; Lee, H. Decomposing motion and content for natural video sequence prediction. In Proceedings of the International Conference on Learning Representations, Toulon, France, 24–26 April 2017.
17. Villegas, R.; Yang, J.; Zou, Y.; Sohn, S.; Lin, X.; Lee, H. Learning to generate long-term future via hierarchical prediction. In Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; Volume 70, pp. 3560–3569.

18. Walker, J.; Marino, K.; Gupta, A.; Hebert, M. The pose knows: Video forecasting by generating pose futures. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 3332–3341.
19. Wichers, N.; Villegas, R.; Erhan, D.; Lee, H. Hierarchical Long-term Video Prediction without Supervision. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018.
20. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Munich, Germany, 5–9 October 2015; pp. 234–241.
21. Newell, A.; Yang, K.; Deng, J. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision*; Springer: Amsterdam, The Netherlands, 2016; pp. 483–499.
22. Battaglia, P.; Pascanu, R.; Lai, M.; Rezende, D.J. Interaction networks for learning about objects, relations and physics. *Adv. Neural Inf. Process. Syst.* **2016**, 1, 4502–4510.
23. Chang, M.B.; Ullman, T.; Torralba, A.; Tenenbaum, J.B. A Compositional Object-Based Approach to Learning Physical Dynamics. In Proceedings of the International Conference on Learning Representations, Toulon, France, 24–26 April 2017.
24. Fragkiadaki, K.; Agrawal, P.; Levine, S.; Malik, J. Learning visual predictive models of physics for playing billiards. In Proceedings of the International Conference on Learning Representations, San Juan, PR, USA, 2–4 May 2016.
25. Schüldt, C.; Laptev, I.; Caputo, B. Recognizing human actions: A local SVM approach. In Proceedings of the 17th International Conference on Pattern Recognition (ICPR), St. Petersburg, Russian, 18–23 October 2004; Volume 3, pp. 32–36.
26. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612.
27. Wang, Y.; Jiang, L.; Yang, M.H.; Li, L.J.; Long, M.; Fei-Fei, L. Eidetic 3D LSTM: A Model for Video Prediction and Beyond. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
28. Xue, T.; Wu, J.; Bouman, K.; Freeman, B. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. *Adv. Neural Inf. Process. Syst.* 2016, 1, 91–99.
29. Babaeizadeh, M.; Finn, C.; Erhan, D.; Campbell, R.H.; Levine, S. Stochastic Variational Video Prediction. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
30. Wang, Y.; Long, M.; Wang, J.; Gao, Z.; Yu, P.S. PredRNN: Recurrent Neural Networks for Predictive Learning using Spatiotemporal LSTMs. *Adv. Neural Inf. Process. Syst.* **2017**, 1, 879–888.
31. Oliu, M.; Selva, J.; Escalera, S. Folded Recurrent Neural Networks for Future Video Prediction. In *ECCV*; Springer: Munich, Germany, 2018.
32. Lakhal, M.I.; Lanz, O.; Cavallaro, A. View-LSTM: Novel-View Video Synthesis Through View Decomposition. In Proceedings of the 2019 ICCV, Seoul, Korea, 27 October–2 November 2019; pp. 7576–7586.
33. Gao, H.; Xu, H.; Cai, Q.Z.; Wang, R.; Yu, F.; Darrell, T. Disentangling Propagation and Generation for Video Prediction. In Proceedings of the 2019 ICCV, Seoul, Korea, 27 October–2 November 2019; pp. 9005–9014.
34. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to Sequence Learning with Neural Networks. *Adv. Neural Inf. Process. Syst.* **2014**, 2, 3104–3112.
35. Gers, F.A.; Schraudolph, N.N.; Schmidhuber, J. Learning Precise Timing with LSTM Recurrent Networks. *J. Mach. Learn. Res.* **2002**, *3*, 115–143.
36. Finn, C.; Goodfellow, I.; Levine, S. Unsupervised learning for physical interaction through video prediction. *Adv. Neural Inf. Process. Syst.* **2016**, 1, 64–72.
37. Lotter, W.; Kreiman, G.; Cox, D. Deep predictive coding networks for video prediction and unsupervised learning. In Proceedings of the International Conference on Learning Representations, Toulon, France, 24–26 April 2017.
38. Eslami, S.A.; Heess, N.; Weber, T.; Tassa, Y.; Szepesvari, D.; Hinton, G.E. Attend, infer, repeat: Fast scene understanding with generative models. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 3225–3233.
39. Kosiorek, A.; Kim, H.; Teh, Y.W.; Posner, I. Sequential attend, infer, repeat: Generative modelling of moving objects. *Adv. Neural Inf. Process. Syst.* **2018**, 1, 8615–8625.

40. Hsieh, J.T.; Liu, B.; Huang, D.A.; Fei-Fei, L.F.; Niebles, J.C. Learning to decompose and disentangle representations for video prediction. *Adv. Neural Inf. Process. Syst.* **2018**, 1, 515–524.
41. Wang, Y.; Gao, Z.; Long, M.; Wang, J.; Yu, P.S. PredRNN++: Towards A Resolution of the Deep-in-Time Dilemma in Spatiotemporal Predictive Learning. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018.
42. Pan, J.; Wang, C.; Jia, X.; Shao, J.; Sheng, L.; Yan, J.; Wang, X. Video Generation From Single Semantic Label Map. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019.
43. Wang, T.H.; Cheng, Y.C.; Lin, C.H.; Chen, H.T.; Sun, M. Point-to-Point Video Generation. In Proceedings of the IEEE International Conference on Computer Vision, Seoul, South Korea, 27 October–2 November 2019.
44. Vondrick, C.; Pirsiavash, H.; Torralba, A. Generating videos with scene dynamics. *Adv. Neural Inf. Process. Syst.* **2016**, 1, 613–621.
45. Tulyakov, S.; Liu, M.Y.; Yang, X.; Kautz, J. MoCoGAN: Decomposing Motion and Content for Video Generation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 1526–1535.
46. Denton, E.; Fergus, R. Stochastic video generation with a learned prior. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018.
47. Ohnishi, K.; Yamamoto, S.; Ushiku, Y.; Harada, T. Hierarchical video generation from orthogonal information: Optical flow and texture. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.
48. Liang, X.; Lee, L.; Dai, W.; Xing, E.P. Dual Motion GAN for Future-Flow Embedded Video Prediction. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 1762–1770.
49. Lee, A.X.; Zhang, R.; Ebert, F.; Abbeel, P.; Finn, C.; Levine, S. Stochastic Adversarial Video Prediction. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
50. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015.
51. Ba, J.; Kiros, J.R.; Hinton, G.E. Layer Normalization. *arXiv* **2016**, arXiv:abs/1607.06450.
52. Isola, P.; Zhu, J.Y.; Zhou, T.; Efros, A.A. Image-to-Image Translation with Conditional Adversarial Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 5967–5976.
53. Bengio, S.; Vinyals, O.; Jaitly, N.; Shazeer, N. Scheduled sampling for sequence prediction with recurrent neural networks. *Adv. Neural Inf. Process. Syst.* **2015**, 1, 1171–1179.
54. Haralick, R.M.; Sternberg, S.R.; Zhuang, X. Image analysis using mathematical morphology. *IEEE Trans. Pattern Anal. Mach. Intell.* **1987**, *4*, 532–550.
55. Srivastava, N.; Mansimov, E.; Salakhudinov, R. Moving MNIST Download Link. 2015. Available online: http://www.cs.toronto.edu/~nitish/unsupervised_video/ (accessed on 6 May 2019).
56. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. In Proceedings of the International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015.