Fully Context-Aware Video Prediction

Anonymous Author(s)
Affiliation
Address
email

Abstract

This paper proposes a new neural network design for unsupervised learning through video prediction. Current video prediction models based on convolutional networks, recurrent networks, and their combinations often result in blurry predictions. Recent work has attempted to address this issue with techniques like separation of background and foreground modeling, motion flow learning, or adversarial training. We highlight that a contributing factor for this problem is the failure of current architectures to fully capture relevant past information for accurately predicting the future. To address this shortcoming we introduce a fully context-aware architecture, which captures the entire available past context for each pixel using Parallel Multi-Dimensional LSTM units and aggregates it using context blending blocks. Our model is simple, efficient and directly applicable to high resolution video frames. It yields state-of-the-art performance for next step prediction on three challenging real-world video datasets: Human 3.6M, Caltech Pedestrian, and UCF-101 and produces sharp predictions of high visual quality.

1 Introduction

Learning from raw videos to predict future video frames without labels has recently become an important direction of research. In this problem setting, a model is trained to predict future frames conditioned on the past and learns a representation that captures the information (the appearance and the motion of objects) in a video without external supervision. Such a representation can then be transferred to other video analysis tasks such as action recognition [1] or utilized for model based reinforcement learning [2].

Learning such predictive models for natural videos is a challenging problem due to various resolutions, object occlusion, camera movement, dynamic scene and light changes between frames. In particular, videos often include large and rapid motion which is hard to capture. To make modeling easier, explicit motion priors [3, 4] are often incorporated into deep neural networks.

Srivastava et al. [1] introduced Long Short-Term Memory (LSTM) [5] based encoder-decoder models for the task of video prediction. This model was able to model temporal dependencies but did not take spatial regularities of the data into account, making it very difficult to generate precise predictions. One of the popular methods, Convolutional LSTM (ConvLSTM) or Parallel Multi-dimensional LSTM (PyraMiD-LSTM) [6, 7] extended the fully-connected LSTM structure by incorporating convolutions in LSTM operations to better capture spatio-temporal correlations. This design better matches the requirements of a good model architecture for video prediction, and has been adopted in recent work [8, 9]. However, these and various other models [10, 11] which directly predict pixel values of future frames still often suffer from blurry predictions. Several possible methods to address this problem have been suggested [12, 13, 8, 9, 14].

Imprecise predictions (resulting in blurry outputs) from these models are a manifestation of model uncertainty, which can arise from two main sources: 1) the future may be ambiguous given the past,

Submitted to 31st Conference on Neural Information Processing Systems (NIPS 2017). Do not distribute.

and 2) the model fails to sufficiently capture relevant past information which might reduce uncertainty. In this paper, we attempt to address the second issue in models for video prediction.

We first show that current models based on ConvLSTM/PyraMiD-LSTM do not take the entire spatio-temporal context from past frames into account, leading to increased uncertainty about the future. We then propose a new model for video prediction based on PyraMiD-LSTM, which integrates the full available context of each pixel in all possible directions (left, right, top, bottom, and depth directions). This context-aware predictive model is rather simple compared to other recent approaches and produces sharper predictions without using novel loss functions, motion or background priors. It outperforms state-of-the-art approaches significantly in a variety of challenging video prediction scenarios: car driving, human motion, and diverse human actions. Quantitative improvements on metrics are accompanied by results of high visual quality showing sharp future predictions without suffering from blurriness or other motion artifacts.

2 Related Work

Network Architectures: Published approaches for video analysis exploit different amounts of spatio-temporal information in different ways depending on model architecture. One common strategy is to use models based on *3D Convolutional Neural Networks (CNNs)* that use convolutions across temporal and spatial dimensions to model all local correlations [3, 15] for supervised learning. Similar architectures have been used for video prediction to directly generate the RGB values of pixels in videos [10, 12, 16]. An inherent limitation of these models is that convolutions take only short-range dependencies into account due to the limited size of the kernels.

This short-range context is not enough to capture large and variable motion between frames, and therefore such networks need a large stack of convolutional layers to include a wider range of contexts. Since the context is always limited by the architecture, loss of relevant context can lead to uncertainty and often produces blurry results. One idea, applicable to situations with mostly static background, is to explicitly model moving foreground objects separately from the static background using a two-stream architecture to improve the precision of the predictions Simonyan et al. [3], Vondrick et al. [16], and Finn et al. [9].

Incorporating recurrent neural networks can address the issue of limited context, and this is idea behind the use of *ConvLSTM* based models which replace the internal transformations of an LSTM cell with convolutions. Xingjian et al. [6] proposed this design for precipitation nowcasting; the motivation being that the convolutional operation would model spatial dependencies, while LSTM connectivity would offer increased temporal context. Since then it has been utilized for incorporating long term temporal context in video prediction models [8, 9]. Specifically, Finn et al. [9] use it in a model that learns to predict pixel motions instead of values, and Lotter et al. [8] develop a model inspired by the predictive coding that learns to better predict future frames by incorporating previous prediction errors. While these models improve prediction quality over past work, they still retain blurriness in generated frames.

Learning strategies: Mathieu et al. [12] attributed the issue of blurry predictions to the Mean Squared Error (MSE) loss function used by several other methods. Since MSE assumes that the data comes from a Gaussian distribution, it trains the model to produce the mean of possible outcomes when the posterior distribution is multimodal. To reduce the ambiguity in prediction, they proposed the use of $\mathcal{L}_1$ & Gradient Difference Loss (GDL) loss functions, together with adversarial training based on Generative Adversarial Networks [17]. Vondrick et al. [16] also used adversarial training, but developed a two-stream architecture to generate short videos conditioned on single input frames.

Context aggregation using MD-LSTMs: A general and principled approach to incorporate long term context in multiple dimensions is Multidimensional LSTM (MD-LSTM; [18]), a specialization of DAG-RNNs [19], which has been applied for various supervised tasks such as handwriting recognition [20], language translation [21], image classification [22], and segmentation [23]. This architecture uses MD-LSTM cells to sequentially connect the input elements between multiple dimensions, thus combining temporal and non-temporal dependencies. It contains two MD-LSTM modules per dimension to combine context from all possible directions.

In principle, the MD-LSTM can be applied to any high-dimensional domain problem (including video prediction) to model all relevant dependencies in the data compactly. However, the fully sequential nature of the model makes it unsuitable for parallelization and thus impractical for higher

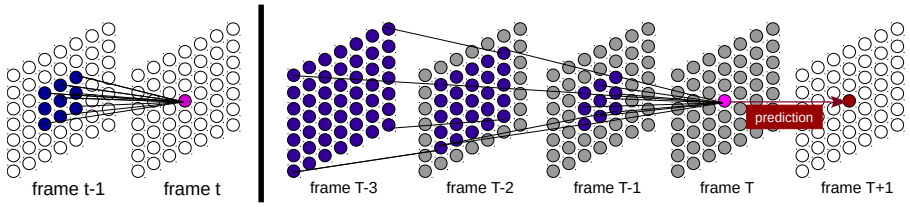

Figure 1: (left) The Convolutional LSTM (ConvLSTM) structure between two frames. (right) The information flow in ConvLSTM over time. This architecture introduces the blind areas (marked as a gray color) which cannot be used to predict the pixel value at time $T + 1$. The closer time frame has larger blind areas.

dimensional data such as videos. The *PyraMiD-LSTM* [7] addressed this issue by re-arranging the recurrent connection topology of each MD-LSTM module from cuboid to pyramidal and enabled an application to volumetric segmentation. The change in topology reduces the total number of modules needed and allows parallelization of computation inside each module. In section 4, we will utilize similar ideas to design an improved model for video prediction. We note here that mathematically, one of the parallelized MD-LSTM modules in PyraMiD-LSTM is equivalent to a typical ConvLSTM which has LSTM connections along the temporal dimension.

3 Missing Contexts in Other Network Architectures

As mentioned earlier, blurry predictions can result from a video prediction model if it does not adequately capture all relevant information in the past video frames which can be used to reduce uncertainty. Figure 1 shows the recurrent connections of a pixel at time t with a 3×3 convolution between two frames (left) and the information flow of a ConvLSTM predicting the pixel at time $T + 1$ (right). The covering context grows progressively over time (depth), but there are also blind areas which cannot be used for prediction. In fact, as can be seen in Figure 1 (right, marked in gray color), frames in the recent past have larger blind areas. Due to this structural issue, the network is unable to capture the entire available context, and is likely to miss important spatio-temporal dependencies leading to increased ambiguity in the predictions. The prediction will eventually fail when the object appearance or motion in videos changes dramatically within a few frames.

One possible way to address limited context, widely used in CNNs for image analysis, is to expand context by stacking multiple layers. However, stacking layers still limits the available context to a maximum as dictated by the network architecture, and the number of additional parameters required to gain sufficient context can be very large for high resolution videos. Another technique that can help is using a multi-scale architecture, but fixed scale factors may not generalize to all possible objects, their positions and motions. In the next section, we introduce a solution for removing these blind areas by combining the full contexts from all directions. Our approach does not require more parameters for higher resolution videos or a priori specification of scale factors, and results in precise predictions without any artifacts.

4 Approach

Let $x_1^T = \{x_1, ..., x_T\}$ be a given input sequence of length T. $x_t \in \mathbb{R}^{H \times W \times C}$ is the t-th frame, where $t \in \{1, ..., T\}$, H is the height, W the width, and C the number of channels. For simplicity, assume $C = 1$. x_1^T is then a cuboid of pixels bounded by six planes. The task is to predict p future frame(s) in the sequence, $x_{t+1}^{t+p} = \{x_{t+1}, ..., x_{t+p}\}$ (next-frame prediction if $p = 1$). Therefore, our goal is to integrate information from the entire cuboid x_1^T into a representation at the plane where $t = T$, which can be used for predicting x_{t+1}^{t+p}. The proposed model includes two main components that learn dependencies along spatio-temporal dimensions in a single architecture. The first is the *Parallel MD-LSTM unit* (PMD unit) that connects dependencies to each pixel. The second is the *Context Blending Block* that combines the output of PMD units from multiple directions.

4.1 Parallel MD-LSTM Unit

A parallel MD-LSTM (PMD) unit is a variant of the MD-LSTM unit, proposed by Stollenga et al. [7], which is more amenable to parallelization. As mentioned in section 2, it implements a method of modeling dependencies in multiple dimensions. Although it is mathematically similar to ConvLSTM,

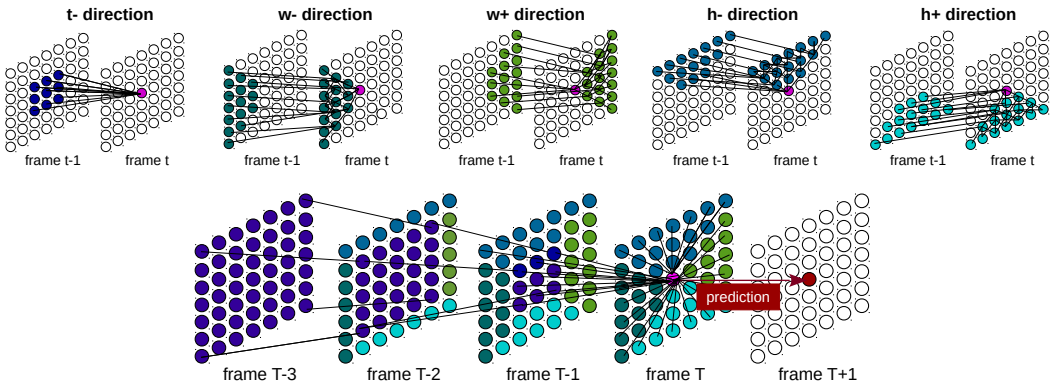

Figure 2: (top) The Parallel MD-LSTM (PMD) units between two frames for five directions: $(t-1)$, $(w-1)$, $(w+1)$, $(h-1)$, and $(h+1)$, where h, w, and t indicates the current position for height, width, and time dimensions. (down) The information flow using all directional PMD units. It covers all available contexts of the history frames in a single architecture.

we adopt this terminology to highlight that it is **not** necessary to limit convolutional operations to spatial dimensions and LSTM connectivity to the temporal dimension.

For any sequence of K two dimensional planes $x_1^K = \{x_1, ..., x_K\}$, the PMD unit computes the current cell and hidden state c_k, s_k using input, forget, output gates i_k, f_k, o_k, and the transformed cell $\tilde{c}_k$ given the cell and hidden state from the previous plane, c_{k-1}, s_{k-1}.

$$
\begin{aligned}
i_k &= \sigma(W_i * x_k + H_i * s_{k\text{-}1} + b_i), \\
f_k &= \sigma(W_f * x_k + H_f * s_{k\text{-}1} + b_f), \\
o_k &= \sigma(W_o * x_k + H_o * s_{k\text{-}1} + b_o) \\
\tilde{c}_k &= \tanh(W_{\tilde{c}} * x_k + H_{\tilde{c}} * s_{k\text{-}1} + b_{\tilde{c}}), \\
c_k &= f_k \odot c_{k\text{-}1} + i_k \odot \tilde{c}_k, \\
s_k &= o_k \odot \tanh(c_k).
\end{aligned}
\tag{1}
$$

Here $(*)$ is the convolution operation, and $(\odot)$ the element-wise multiplication. W and H are the weights for input-state and state-state. The size of weight matrices are dependent only on the kernel size and number of units. If the kernel size is larger, more local context is taken into account. The convolution operation provides an efficient computation on GPUs [1], as well as takes the surrounding local correlation into account.

4.2 Context Blending Block

This block captures the entire available context by combining the output of PMD units from all relevant directions. This is the critical difference from the traditional ConvLSTM: the conditioning directions are aligned not only with the time dimension but also with the spatial dimensions. As shown in section 3, using a ConvLSTM would be equivalent to running a PMD unit along the time dimension from $k = 1$ to $k = T$, which would only integrate information from a pyramid shaped region of the cuboid and ignore several blind areas. For this reason, it is necessary to use four additional PMD units, for which the conditioning directions are aligned with the spatial dimensions, as shown in Figure 2 (top). When the current processing pixel is at the location $l := (h, w, t)$, the connecting recurrent directions are defined as h-, h+, w-, w+, and t-. The five PMD unit outputs from these directions are h^d where $d \in D = \{h\text{-}, h\text{+}, w\text{-}, w\text{+}, t\text{-}\}$. We suggest two ways to combine the information from different directions.

Unrestrained blending (U-blending): this strategy is used in the traditional MD-LSTM [24, 23] and PyraMiD LSTM [7]. The computation is simply the summation of all directions before the regular fully-connected connections:

$$
m_l = f((\sum_{d \in D} s_l^d) \cdot W_l + b_l), \tag{2}
$$

[1] Unlike the traditional MD-LSTM structure, each pixel in the same plane is independent from the conditioning pixels (see Figure 2-top). Thus, all pixels in the plane can be processed concurrently.

where $W_l \in \mathbb{R}^{N1 \times N2}$ and $b_l \in \mathbb{R}^{N2}$ are a weight matrix and a bias. $N1$ is the number of PMD units, and $N2$ is the number of (blending) blocks. f is an activation function.

Weighted blending (W-blending): PMD units from all five directions output $s_l^{d1}, ...s_l^{d5}$, where $d1, ..d5$ are the index of the directions. This block then concatenates s from all directions:

$$S_l = \begin{bmatrix} s_l^{d1} \\ s_l^{d2} \\ \vdots \\ s_l^{d5} \end{bmatrix} \tag{3}$$

The vector S_l is then weighted as follows:

$$m_l = f(S_l \cdot W_l + b_l), \tag{4}$$

where $W_l \in \mathbb{R}^{(5 \times N1) \times N2}$ (5 is the number of directions). The main difference between the two blending approaches is whether the information for each pixel is equally important for all directions or needs to be rated and prioritized. In the latter case, this priority is learned while training.

The PMD computation for each direction (top) and the context mapping to past frames (down) are visualized in Figure 2. Compared to Figure 1, the architecture in Figure 2 covers all available context at every frame (no blind areas) in order to predict the pixel values at time $T + 1$.

4.3 Training

$\hat{x}_l = g(m_l)$ is an output of the top-most (output) layer, where g is an output activation function. The model minimizes the loss between the predicted pixel (model output) $\hat{x}_l$ and the target pixel x_{l+1}. Mathieu et al. [12] introduced several loss functions for better prediction. Here, $\mathcal{L}^p$ loss and the Image Gradient Difference Loss (GDL) are combined. Let y and $\hat{x}$ are the target and the predicted frame. The objective function is defined as follows:

$$\begin{aligned} \mathcal{L}(y, \hat{x}) &= \lambda_p \mathcal{L}_p(y, \hat{x}) + \lambda_{gdl} \mathcal{L}_{gdl}(y, \hat{x}) \\ \mathcal{L}_p(y, \hat{x}) &= ||y - \hat{x}||_p \\ \mathcal{L}_{gdl}(y, \hat{x}) &= \sum_{i,j} ||y_{i,j} - y_{i-1,j}| - |\hat{x}_{i,j} - \hat{x}_{i-1,j}|| + ||y_{i,j-1} - y_{i,j}| - |\hat{x}_{i,j-1} - \hat{x}_{i,j}||, \end{aligned} \tag{5}$$

where $|.|$ is the absolute value function, $\hat{x}_{i,j}$ and $y_{i,j}$ are the pixel elements from the frame $\hat{x}$ and y, respectively. λ_p and λ_{gdl} are the weights for each loss. In our experiments, λ_{gdl} is set to 1 when $p = 1$ and 0 when $p = 2$. λ_p is always set to 1.

We use ADAM [25] as the optimizer with an initial learning rate of $1e{-}3$. The learning rate is decayed every 5 epochs with the decay rate 0.99. Weights are initialized using the Xavier initializer [26] and the states of the LSTMs are initialized to zero (no prior).

5 Experiments

We evaluated the proposed approach on three real-world scenarios with distinct characteristics: human motion prediction, car-mounted camera video prediction, and human activity prediction.

The first scenario is to predict human motions using the Human 3.6M dataset [27]. These videos have static backgrounds and are recorded with a fixed camera. Therefore, the model can focus on the appearance and the motion of humans. The second scenario is to model a real-world driving environment, with potential applications in autonomous driving. Videos are taken from cameras mounted on moving vehicles, and include inconsistent camera movement but with (relatively) similar background, for instance, sky, road, or others cars appear in many of the scenes; but sometimes there are unique backgrounds like a car parking space. To model such videos, a model needs to learn the camera movements and not only motion of foreground objects like the previous scenario. Moreover, training and testing videos are taken from the different cities: KITTI dataset [28] (train) from Germany and CalTech Pedestrian dataset [29] (test) from the US. Such a setup ensures that the model does not memorize particular backgrounds and has to use past information for predictions. The last scenario is to capture people performing a diverse range of activities (applying makeup, playing sports etc.) using the UCF-101 dataset [30]. These videos, collected from YouTube (https://www.youtube.com/), feature varied backgrounds as well as rapid camera movement.

All input pixel values are normalized to the range $[0, 1]$. For the human motion and car-mounted videos, the models are trained to use ten frames as input to predict the next frame. For the human activity videos, the input consists four frames (for fair comparison to previous work). Quantitative evaluation on the test sets is performed based on mean Peak Signal-to-Noise Ratio (PSNR) and the Structural Similarity Index Measure (SSIM) [31][2]. These commonly used numerical measures are known to be not fully representative of human vision. Therefore, we highly recommend looking at the visual results in Figure 3 and the website `https://sites.google.com/view/contextvp`.

Network architecture: Several video prediction models based on the proposed architecture were tested, with variable number of layers (one, three, or four) and types of context blending (unrestrained or weighted). When using a single hidden layer, 64 PMD units from each of the five directions are directly connected to the output layer. The three and four layer models include PMD units followed by context blending blocks. For the three layer model, the number of hidden units are 16 (PMD1) - 20 (Blending1) - 32 (PMD2) - 45 (Blending2) - 64 (PMD3) - 80 (Blending3). For the four layer model, the number of hidden units are 32 (PMD1) - 32 (Blending1) - 64 (PMD2) - 64 (Blending2) - 32 (PMD3) - 32 (Blending3) - 64 (skip connection with Blending1) - 64 (PMD4) - 64 (Blending4) - 128 (skip connection with Blending2). The output layer always uses the sigmoid activation function which outputs values in the range $(0, 1)$. Each PMD unit has convolutional kernels of size 3×3 (for human motion and car-mounted camera videos) or 5×5 (for human action prediction). The activation function f in both Equation 2 and Equation 4 are set to linear. Use of ReLU [32] or layer normalization [33] in the blending blocks does not affect the performance in our experiments. We report the results from $\mathcal{L}_1$ ($p = 1$ in Equation 5) with the GDL loss for our experiments. Finn et al. [9] found that $\mathcal{L}_1$ with the GDL loss function performs better than $\mathcal{L}_2$ but the performance in our case was very similar.

Human Motion Prediction: We first evaluate our model on Human3.6M dataset [34]. The dataset includes seven human subjects (three females and four males). Five subjects are used for training and the other two for validation and testing. The videos are subsampled to 10 fps and downsampled to 64×64 resolution. These and others details of the experimental setup are same as Finn et al. [9].

Note that the model of Finn et al. [9] learns a mask to reconstruct the fixed background easily (copying pixels directly from the previous frame) since videos in this dataset contain a static background and camera with small human movements. This masking scheme is suitable for this scenario but is restricted to such conditions. In contrast, our model learns the entire structure of appearance and motion (as well as their correlations) without such domain specific properties.

Table 1: Evaluation of Next-Frame Predictions on the Human3.6M dataset. The model is trained on ten frames and predicts the next frame. The results are averaged over test videos. Higher values of PSNR and SSIM indicate better results. L indicates the number of layers.

Method	PSNR	SSIM
Copy-Last-Frame	32	-
BeyondMSE [12]	26.7	-
PredNet [8]	38.9	-
DNA [9]	42.1	0.992
Ours (1-L)	38.1	0.990
Ours (3-L)	41.2	0.992
Ours (4-L, U-blending)	42.3	0.994
Ours (4-L, W-blending)	**44.8**	**0.996**

Table 1 shows the comparison of the prediction results with the state-of-the-art approach (DNA) Finn et al. [9] as well as PredNet [8], BeyondMSE [12], and Copy-Last-Frame. Copy-Last-Frame is included to ensure that our model does not simply copy the last frame. Our approach consistently outperforms all other approaches, and notable improvements are obtained by increasing the number of layers. Single layer networks already outperform BeyondMSE which is built on 3D-CNN, and the three layer networks outperform PredNet which uses ConvLSTM. Finally, the four layer networks outperform the best performing approach on this dataset (DNA) which also uses ConvLSTM and explicitly models a background mask and motion separately. All compared approaches have between five to seven layers. These results show that our approach can efficiently capture the internal representation using simpler networks. Figure 3 (row 1) shows sample visual prediction results.

[2]Mean Squared Error (MSE) is also reported for the car-mounted camera video prediction to compare with PredNet [8].

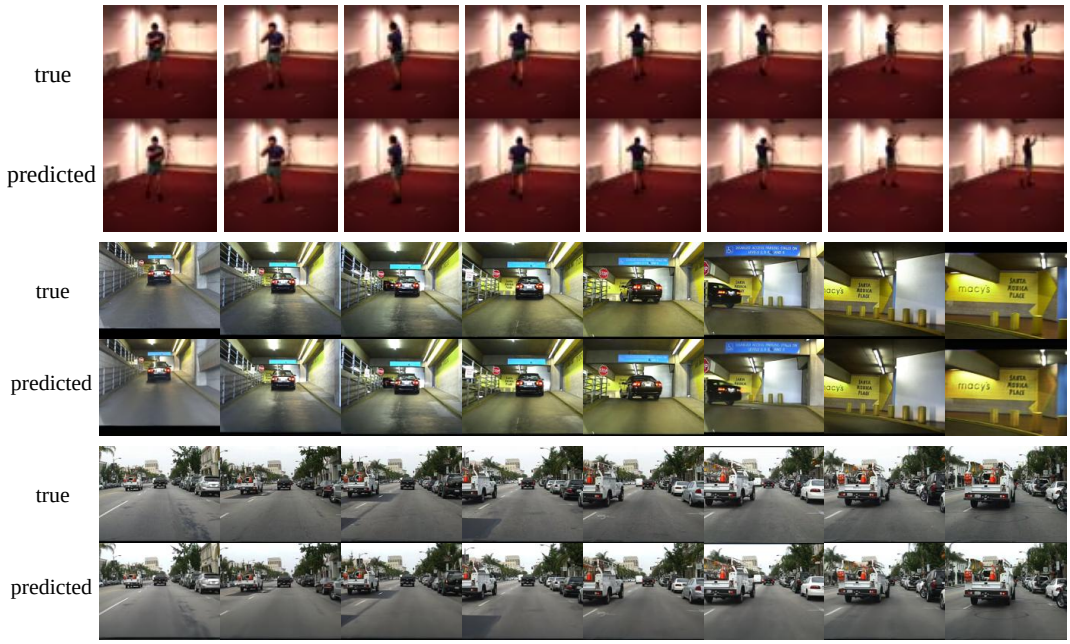

Figure 3: Next frame prediction for human motion (row 1) and car-cam video (row 2-3) sequences. The first rows are the true frames and the second rows are predictions. For car-cam video sequences, the model was trained on the KITTI dataset and the prediction results are from the CalTech Pedestrian dataset. More examples of the prediction results are online: `https://sites.google.com/view/contextvp`

Car-mounted Camera Video Prediction: As mentioned earlier, the model is trained on the KITTI dataset [28] (Germany) and tested on the CalTech Pedestrian dataset [29] (USA) Every ten frames from "city", "Residential", and "Road" videos are sampled for training resulting in ≈41K frames. Frames from both datasets are center-cropped and down-sampled to 128×160 pixels. We used the exact data preparation as PredNet [8] for direct comparison.

The car-mounted camera videos are taken from moving vehicles and consist of a wide range of motions. Compared to Human3.6M dataset, which has static background and small motion flow, this dataset has diverse and large motion of cars (in different scales) and also has a camera movement. To understand such videos, a model is required to learn not only small movement of pedestrians, but also relatively large motion of surrounding vehicles and backgrounds.

Table 2: Evaluation of Next frame prediction on the CalTech Pedestrian dataset (trained on the KITTI dataset). The model is trained on ten frames and predicts the next frame. The results are averaged over test videos. Higher values of PSNR and SSIM indicate better results and the lower for MSE. L indicates the number of layers.

Method	MSE ($\times 10^{-3}$)	PSNR	SSIM
Copy-Last-Frame	7.95	23.3	0.779
PredNet [8]	2.42	27.6	0.905
Ours (3-L, W-blending)	2.18	27.9	0.914
Ours (4-L, W-blending)	**2.04**	**28.4**	**0.919**

We compare our approach with the Copy-Last-Frame baseline and the PredNet which is the current state-of-the-art model for this dataset. Note that the scores provided in Lotter et al. [8] are averaged over nine frames (time step 2-10), but ours are computed only on the last predicted frame. We therefore re-calculated the scores of PredNet using their trained network. As shown in Table 2, our four layer model outperforms the state-of-the-art on all metrics.Some samples of the prediction results on CalTech Pedestrian dataset are provided in Figure 3 (row 2-3). Samples are selected to show the results from diverse driving scenarios: moving into the indoor parking space, (row 2) and the front car changing the lane (row 3). Our model is able to adapt predictions to the current scene and make sharp predictions without any artifacts.

Figure 4: Examples of failed prediction for car-cam video sequences. The first rows are the true frames and the second rows are predictions. $T-1$ and $T-2$ are the last two input frames, and T is the current output frame. As can be seen in the predicted frame at time T, some details of objects which were not appeared in the last frames, are missed.

Human Action Prediction: The last evaluation is on the UCF-101 Dataset [30]. Although many videos in this dataset contain small movements between frames, they contain much more diversity in objects, backgrounds and camera motions compared to previous datasets. Our experimental setup follows that of Mathieu et al. [12]. About 500,000 training videos are selected from the UCF-101 training set, and 10% of UCF-101 test set is used for testing (378 videos). All frames are resized to 256×256. Note that Mathieu et al. [12] used randomly selected sequences of 32×32 patches from the Sports-1M dataset [36] for training because the motion between frames in the UCF-101 dataset are too small to capture easily. Our model however, is directly trained on UCF-101 subsequences of length four with the original resolution. Motion masks generated using Epicflow [37] provided by Mathieu et al. [12] are used to compute PSNR and SSIM to focus evaluation on regions with significant motion.

Table 3: Evaluation of Next-Frame Predictions on the UCF-101 dataset. The model is trained on four frames and predicts the next frame. The results are averaged over test videos. Higher values of PSNR and SSIM indicate better results. L indicates the number of layers.

Method	PSNR	SSIM
EpicFlow [35]	31.6	**0.93**
BeyondMSE [12]	32	0.92
Ours (3-L, W-blending)	32.2	0.90
Ours (4-L, W-blending)	**33.4**	0.91

Table 3 presents the quantitative evaluation on the UCF-101 dataset. Our best model, a four layered network with weighted context blending, again outperforms BeyondMSE and predictions generated using the outputs of EpicFlow. These results indicate that this model can capture all relevant spatial-temporal information and use it to make sharp predictions in very diverse settings without requiring adversarial training.

Failure Cases: Figure 4 shows some examples of failing cases in prediction. Although most of objects and background in the predicted frame at time $T+1$ look sharp, some object (marked as a red circle) appears to be blurry. This particular object hardly appears in the last (input) frames before prediction (see true frames at time $T-1$, T, and $T+1$ in Figure 4), so the input does not have sufficient information to predict details of the object (see the predicted frames at time $T+1$ in Figure 4). Thus, this failure case is caused by future ambiguity mentioned in section 1.

6 Conclusion and Future Directions

This paper identified the issue to missing context in current video prediction models, which is a contributor to uncertain predictions about the future and leads to generation of blurry frames. To address this issue, we adapted ideas from recent work in volumetric segmentation of biomedical data to develop a prediction architecture that can capture all the relevant context efficiently. The resulting model outperformed existing approaches for video prediction in a variety of scenarios, demonstrating the importance of fully context-aware models.

Our model did not incorporate the use of other orthogonal ideas to improve video prediction, such as use of multiple scales, explicit background/motion flow modeling, or novel training strategies. Since these ideas have been previously explored for models with incomplete context, a promising future direction is to evaluate their influence on fully context-aware models. Finally, an important line of investigation is transfer learning for various supervised or reinforcement learning tasks.

References

[1] N. Srivastava, E. Mansimov, and R. Salakhudinov. "Unsupervised learning of video representations using lstms". In: *International Conference on Machine Learning*. 2015, pp. 843–852.

[2] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

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

[4] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. "Beyond short snippets: Deep networks for video classification". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 4694–4702.

[5] S. Hochreiter and J. Schmidhuber. *Long Short-Term Memory*. Tech. rep. FKI-207-95. Revised 1996 (see www.idsia.ch/˜juergen, www7.informatik.tu-muenchen.de/˜hochreit). Fakultät für Informatik, Technische Universität München, 1995.

[6] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. "Convolutional LSTM network: A machine learning approach for precipitation nowcasting". In: *Advances in Neural Information Processing Systems*. 2015, pp. 802–810.

[7] M. F. Stollenga, W. Byeon, M. Liwicki, and J. Schmidhuber. "Parallel multi-dimensional LSTM, with application to fast biomedical volumetric image segmentation". In: *Advances in Neural Information Processing Systems*. 2015, pp. 2998–3006.

[8] W. Lotter, G. Kreiman, and D. Cox. "Deep predictive coding networks for video prediction and unsupervised learning". In: *arXiv preprint arXiv:1605.08104* (2016).

[9] C. Finn, I. Goodfellow, and S. Levine. "Unsupervised learning for physical interaction through video prediction". In: *Advances In Neural Information Processing Systems*. 2016, pp. 64–72.

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

[11] J. Oh, X. Guo, H. Lee, R. L. Lewis, and S. Singh. "Action-conditional video prediction using deep networks in atari games". In: *Advances in Neural Information Processing Systems*. 2015, pp. 2863–2871.

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

[13] Z. Liu, R. Yeh, X. Tang, Y. Liu, and A. Agarwala. "Video Frame Synthesis using Deep Voxel Flow". In: *arXiv preprint arXiv:1702.02463* (2017).

[14] N. Kalchbrenner, A. v. d. Oord, K. Simonyan, I. Danihelka, O. Vinyals, A. Graves, and K. Kavukcuoglu. "Video pixel networks". In: *arXiv preprint arXiv:1610.00527* (2016).

[15] D. Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri. "C3D: generic features for video analysis". In: *CoRR, abs/1412.0767* 2 (2014), p. 7.

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

[17] 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*. 2014, pp. 2672–2680.

[18] A. Graves, S. Fernández, and J. Schmidhuber. "Multi-dimensional Recurrent Neural Networks". In: *Artificial Neural Networks – ICANN 2007: 17th International Conference, Porto, Portugal, September 9-13, 2007, Proceedings, Part I*. Ed. by J. M. de Sá, L. A. Alexandre, W. Duch, and D. Mandic. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 549–558. ISBN: 978-3-540-74690-4.

[19] P. Baldi and G. Pollastri. "The principled design of large-scale recursive neural network architectures–dag-rnns and the protein structure prediction problem". In: *Journal of Machine Learning Research* 4.Sep (2003), pp. 575–602.

[20] A. Graves and J. Schmidhuber. "Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks". In: *NIPS*. 2009.

[21] N. Kalchbrenner, I. Danihelka, and A. Graves. "Grid long short-term memory". In: *arXiv preprint arXiv:1507.01526* (2015).

[22] W. Byeon, M. Liwicki, and T. M. Breuel. "Texture classification using 2d lstm networks". In: *Pattern Recognition (ICPR), 2014 22nd International Conference on*. IEEE. 2014, pp. 1144–1149.

[23] W. Byeon, T. M. Breuel, F. Raue, and M. Liwicki. "Scene Labeling With LSTM Recurrent Neural Networks". In: *CVPR*. 2015.

[24] A. Graves, S. Fernández, and J. Schmidhuber. "Multi-dimensional recurrent neural networks". In: *Proceedings of the 17th International Conference on Artificial Neural Networks*. 2007.

[25] D. Kingma and J. Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[26] X. Glorot and Y. Bengio. "Understanding the difficulty of training deep feedforward neural networks." In: *Aistats*. Vol. 9. 2010, pp. 249–256.

[27] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. "Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments". In: *IEEE transactions on pattern analysis and machine intelligence* 36.7 (2014), pp. 1325–1339.

[28] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. "Vision meets robotics: The KITTI dataset". In: *The International Journal of Robotics Research* 32.11 (2013), pp. 1231–1237.

[29] P. Dollár, C. Wojek, B. Schiele, and P. Perona. "Pedestrian detection: A benchmark". In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE. 2009, pp. 304–311.

[30] K. Soomro, A. R. Zamir, and M. Shah. "UCF101: A dataset of 101 human actions classes from videos in the wild". In: *arXiv preprint arXiv:1212.0402* (2012).

[31] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. "Image quality assessment: from error visibility to structural similarity". In: *IEEE transactions on image processing* 13.4 (2004), pp. 600–612.

[32] V. Nair and G. E. Hinton. "Rectified linear units improve restricted boltzmann machines". In: *Proceedings of the 27th international conference on machine learning (ICML-10)*. 2010, pp. 807–814.

[33] J. L. Ba, J. R. Kiros, and G. E. Hinton. "Layer normalization". In: *arXiv preprint arXiv:1607.06450* (2016).

[34] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. "Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.7 (2014), pp. 1325–1339.

[35] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. "Epicflow: Edge-preserving interpolation of correspondences for optical flow". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 1164–1172.

[36] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. "Large-scale Video Classification with Convolutional Neural Networks". In: *CVPR*. 2014.

[37] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. "EpicFlow: Edge-Preserving Interpolation of Correspondences for Optical Flow". In: *Computer Vision and Pattern Recognition*. 2015.