SME-Net: Sparse Motion Estimation for Parametric Video Prediction through Reinforcement Learning

Yung-Han Ho Chuan-Yuan Cho Wen-Hsiao Peng Guo-Lun Jin
Department of Computer Science, National Chiao Tung University, Taiwan
{hectorho0409.cs04g@, fountaintin.cs05g@, wpeng@cs., jingjing963.iie07g@}nctu.edu.tw

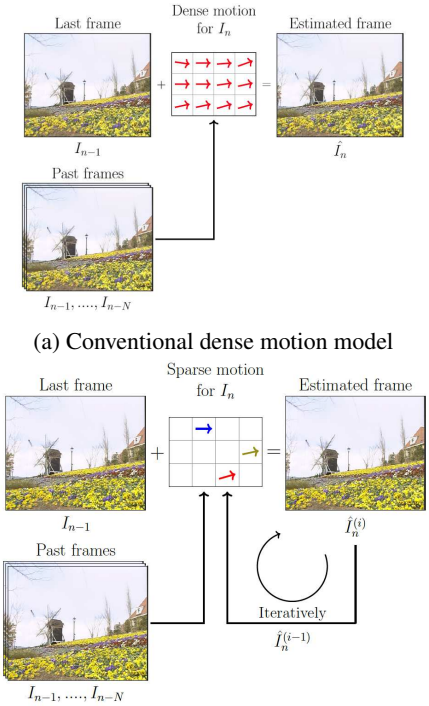
Abstract

This paper leverages a classic prediction technique, known as parametric overlapped block motion compensation (POBMC), in a reinforcement learning framework for video prediction. Learning-based prediction methods with explicit motion models often suffer from having to estimate large numbers of motion parameters with artificial regularization. Inspired by the success of sparse motion-based prediction for video compression, we propose a parametric video prediction on a sparse motion field composed of few critical pixels and their motion vectors. The prediction is achieved by gradually refining the estimate of a future frame in iterative, discrete steps. Along the way, the identification of critical pixels and their motion estimation are addressed by two neural networks trained under a reinforcement learning setting. Our model achieves the state-of-the-art performance on CaltchPed, UCF101 and CIF datasets in one-step and multi-step prediction tests. It shows good generalization results and is able to learn well on small training data.

1. Introduction

Video prediction is one challenging computer vision task. It is to predict future frames by observing a sequence of past frames. The task is complicated by the needs to address the variety of natural videos in both their motion dynamics and texture appearance.

There are several prior arts on learning-based video prediction. One class of approaches [2, 9, 12, 13, 22] use generative models to synthesize future frames directly. These models often involve using Long Short-Term Memory (LSTM) networks to capture motion dynamics from past frames to assist with the generation of future frames by convolutional neural networks (CNN). Typical examples are MCNet [22], PredNet [12] and BeyondMSE [13], where BeyondMSE [13] additionally introduces a multi-scale, pyramid generation process. One common problem with these methods is blurry synthesis results because raw pixel values often follow a multi-modal distribution that calls for subtle training objectives. This downside can be worsened by multi-step prediction when future frames have to be predicted recursively.



(a) Conventional dense motion model

(b) Proposed sparse motion model

Figure 1: Comparison of video prediction on (a) a dense and (b) a sparse motion fields.

Instead of generating video frames directly, another class of methods model the motion dynamics explicitly by estimating a dense motion field [5, 10, 11, 16, 17] that connects pixel values of a future frame to those in the most recent past frame, as depicted in Fig. 1(a). This dense motion field can be specified in terms of optical flows/motion vectors [11, 8], pixel-adaptive kernels [5, 15], or the combination of both [17]. Since the motion parameters have to be estimated for every target pixel, these methods demand complex models with artificial regularization.

Inspired by video compression, where video prediction is often carried out efficiently on a sparse motion field, we develop a SME-Net that leverages a classic prediction technique, known as *parametric overlapped block motion compensation* (POBMC) [3], in a reinforcement learning (RL) framework for video prediction. As illustrated in Fig. 1(b), it gradually refines the estimate of a future frame in iterative, discrete steps based on performing POBMC [3] on a sparse motion field composed of few critical pixels and their estimated motion vectors. The identification of critical pixels and the estimation of their motion vectors are addressed by two neural works trained under an RL setting.

Experimental results show that our model achieves the-state-of-the-art performance on several common datasets in one-step and multi-step prediction tests, in terms of both objective and subjective quality. Moreover, our model has a size that is at least one order of magnitude smaller than most of the competing methods. It also shows good generalization results and is able to learn well on small training data. To the best of our knowledge, this the first attempt that leverages a classic prediction technique in a learning framework for video prediction.

The remainder of this paper is organized as follows. Section 2 briefly reviews related work. Section 3 details the proposed method, with Section 4 elaborating on our network architectures. Section 5 describes the training procedure. Section 6 evaluates our method against several state-of-the-art baselines. Section 7 concludes this work.

2. Related Work

Our proposed method aims to perform parametric video prediction on a sparse motion field. We thus address prior works with explicit motion models that are most relevant to our task and scheme. We shall also introduce briefly parametric overlapped block motion compensation [3], a classic parametric video prediction technique that underpins our framework.

Dense Motion Models. Video prediction that bases pixel generation of a future frame on explicit motion models remains the state-of-the-art solution. This class of prediction methods often estimate a dense motion field that connects pixel values of a future frame to those in past frames. Typical examples are the flow-based (also known as vector-based) models, such as the deep voxel flow (DVF) [11] and the spatial transformation (SPT) [7]. The DVF [11], when operated in extrapolation mode, learns to predict optical flows between a future frame and a past frame, while the SPT [7] learns the backward content transformation from the future to the past through a 6-parameter affine model. The dense motion field can also be represented in the form of a pixel-adaptive convolution kernel that convolves with a past frame to synthesize a future frame. One such approach is the dynamic neural advection (DNA) [5]. The kernel-based representation typically shows better synthesis quality than the flow-based representation because the convolution kernels can be learned to compensate for fine motion and suppress noise inherent in video frames. They however have difficulty modeling large motions due to the usually small kernel size. More recently, the SDC-Net [17] proposes to learn a motion vector and a convolution kernel for each target pixel to gain the merits of both representations. These dense motion models however face one common problem that there are an excessive number of optical flows and/or kernels to estimate, which calls for complex models and additional regularization especially because the estimation must rely only on casual information.

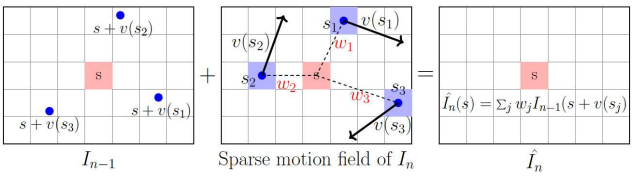


Figure 2: Illustration of POBMC [3].

Sparse Motion Models. Prediction of a video frame based on a sparse motion field where only a handful of target pixels have their motion vectors is prevalent in video compression. This arises as a trade-off between prediction efficiency and overhead needed to communicate motion vectors to the decoder. Over the years, the compression community has developed very efficient prediction techniques that can work on a sparse motion field. One example is the parametric overlapped block motion compensation (POBMC) [3]. To produce a prediction $\hat{I}_n(s)$ for a target pixel $s = (s_x, s_y) \in I_n$ in a future frame I_n at time instance n, *e.g.* the red pixel in Fig. 2, it computes a weighted sum $\hat{I}_n(s) = \sum_{j=1}^{K} w_j I_{n-1}(s + v(s_j))$ of several hypotheses, each being a motion-compensated signal $I_{n-1}(s + v(s_j))$ from the most recent past frame I_{n-1} using the motion vector $v(s_j) = (v_x(s_j), v_y(s_j)) \in R^2$ associated with one of the surrounding critical pixels $\{s_j\}_{j=1}^{K} \in I_n$, *i.e.* the blue pixels in Fig. 2. Under mild conditions, the optimal weight w_j^* are computed in closed-form to be inversely proportional to the Euclidean distance $r(s, s_j)$ between the target pixel s and the surrounding critical pixels s_j [3]:

$$w_j^* = \frac{r(s, s_j)^{-\alpha}}{\sum_{i=1}^{K} r(s, s_i)^{-\alpha}},\ j = 1, 2, \ldots, K \qquad (1)$$

where α is a hyper-parameter.

In a sense, POBMC [3] is a more efficient combination of the kernel-based and flow-based methods. Like the kernel-based method, it adapts the convolution kernel $\{w_j^*\}_{j=1}^{K}$ to every target pixel s (see Eq. (1)). However, its kernel has a highly variable support, which is determined by the motion vectors $\{v(s_j)\}_{j=1}^{K}$ of critical pixels $\{s_j\}_{j=1}^{K}$. Also, like the flow-based method, POBMC [3] needs to estimate motion vectors. This is however done with respect to few critical pixels only. Given these and many other

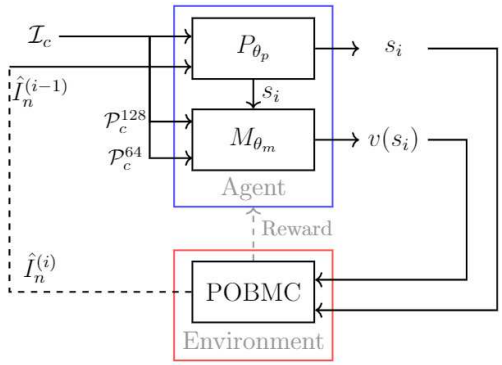


Figure 3: Overall architecture of our model.

good properties of POBMC [3], it forms the basis of our prediction method. To the best of our knowledge, this the first attempt that leverages a classic prediction technique in a learning framework for video prediction.

3. Proposed Method

In this section, we begin with an overview of our sparse motion estimation-based (SME-based) video prediction for the one-step case, followed by its extension to multi-step prediction.

3.1. One-step Prediction

Fig. 3 presents an overall architecture of our SME-based parametric video prediction. The task is to synthesize one single future frame I_n based on N observed past frames $I_{n-1}, I_{n-2}, \ldots, I_{n-N}$, which we refer collectively to as the *context frames* and denote by $\mathcal{I}_c = \{I_{n-l}\}_{l=1}^{N}$. Unlike the flow-based or kernel-based prediction, which typically performs *pixel-based frame synthesis* in one single forward pass on a *dense motion field*, ours gives an estimate $\hat{I}_n$ of the future frame I_n in K iterative, discrete steps based on performing *parametric frame synthesis* on a *sparse motion field* composed of only K pairs $\{(s_j, v(s_j))\}_{j=1}^{K}$ of critical pixels $s_j \in I_n$ and their estimated motion vectors $v(s_j)$.

The entire process begins with the SME to determine those critical pixels $\{(s_j, v(s_j))\}_{j=1}^{K}$, followed by parametric frame synthesis. Specifically, in the i-th iteration, the SME unfolds as follows:

Positioning Network $P(\cdot;\theta_p)$ parameterized by θ_p takes as inputs the context frames $\mathcal{I}_c$ and the estimate $\hat{I}_n^{(i-1)}$ of the future frame $I_n(s)$ obtained in the previous iteration, and outputs a multinominal distribution over the location of the i-th critical pixel s_i. That is,

$$s_i \sim P(s|\mathcal{I}_c, \hat{I}_n^{(i-1)}; \theta_p). \tag{2}$$

Multi-scale Motion Estimation Network $M(\cdot;\theta_m)$ parameterized by θ_m takes as inputs a newly estimated critical pixel s_i drawn from $P(\cdot;\theta_p)$ and the image patches $\mathcal{P}_c^{64}(s_i), \mathcal{P}_c^{128}(s_i)$ of sizes 64×64 and 128×128 cropped from the context frames $\mathcal{I}_c$ at s_i and outputs an estimated motion vector $v(s_i)$ for the critical pixel s_i. That is,

$$v(s_i) = M(\mathcal{P}_c^{64}(s_i), \mathcal{P}_c^{128}(s_i); \theta_m). \tag{3}$$

Also produced in the i-th iteration of the SME is an estimate $\hat{I}_n^{(i)}$ of the future frame I_n given by the parametric frame synthesis:

Parametric Frame Synthesis performs POBMC [3] with respect to the most recent past frame I_{n-1} by taking into account all the critical pixels that have been obtained up to the i-th iteration. In symbols, we have

$$\hat{I}_n^{(i)}(s) = \sum_{j \in \mathcal{N}(s)} w_j^* I_{n-1}(s + v(s_j)), \forall s \in I_n, \tag{4}$$

where in estimating the value $\hat{I}_n^{(i)}(s)$ of a target pixel at $s \in I_n$, we have tacitly limit the use of critical pixels to only the two nearest ones $s_j \in \mathcal{N}(s)$ to s in Euclidean distance. This design choice is motivated by the general observation that multi-hypothesis prediction schemes, such as POBMC [3], may lead to blurry results if the prediction of a target pixel involves a large number of hypotheses. Note that in this work, the total number of critical pixels K is variable and can be as high as 100.

After K iterations, $\hat{I}_n^{(K)}$ forms our final prediction of the target frame I_n. It is expected to approximate closely I_n in mean squared error sense by training the two networks to minimize

$$\mathcal{L}(\theta_p, \theta_m) = E[\|I_n - \hat{I}_n^{(K)}\|_2^2]. \tag{5}$$

3.2. Multi-step Prediction

For the prediction of multiple future frames at time instances starting from n to $n + M - 1$, where M specifies the time span in which the prediction is to be made, the single frame (one-step) prediction algorithm described previously can be applied recursively. For example, given the last N context frames, $I_{n-1}, I_{n-2}, \ldots, I_{n-N}$, we first produce an estimate $\hat{I}_n$ of the next future frame I_n. This newly predicted frame along with the last $N - 1$ context frames, i.e. $\hat{I}_n, I_{n-1}, \ldots, I_{n-N+1}$ will then form the new basis for the prediction of the next consecutive future frame I_{n+1}. To be precise, its estimate $\hat{I}_{n+1}$ will be synthesized based on $\hat{I}_n$ by observing the critical pixels derived from $\hat{I}_n, I_{n-1}, \ldots, I_{n-N+1}$. In essence, our multi-step prediction is built from a sliding window-based single frame prediction.

4. Network Architecture

This section describes in detail the design of our positioning network and multi-scale motion estimation network.

4.1. Positioning Network

The positioning network $P(\cdot;\theta_p)$ is a 6-layer convolutional neural network, as shown in Fig. 4(a), that recurrently observes the temporal coherence between the context frames $\mathcal{I}_c$ and the most recent estimate $\hat{I}_n^{(i-1)}$ of the target frame I_n to output the location s_i of the next critical pixel. This is achieved by having the network learn to predict the evolution of video frames along the temporal dimension. To this end, we compute its inputs to be the frame differences formed by subtracting the context frame I_{n-N} at the earliest time instance from the subsequent context frames $I_{n-N+1}, I_{n-N+2}, \ldots, I_{n-1}$ and $\hat{I}_n^{(i-1)}$. The resulting frame differences have a spatial resolution of $W \times H$, where W, H denote their width and height, respectively.

Instead of giving a point estimate of the next critical pixel s_i, the positioning network outputs a 2-D map of dimension $W/4 \times H/4$, representing its probability distribution over the permissible spatial locations. The reason why the output is designed to be a probabilistic distribution is that we found the coordinate regression difficult to converge. In addition, the output resolution is only one sixteenth of the input's; thus, the s_i drawn from this distribution will be scaled up before its use for motion estimation and frame synthesis. Furthermore, a binary mask is placed before the softmax activation layer to mask out critical pixels that have been identified previously.

4.2. Multi-scale Motion Estimation Network

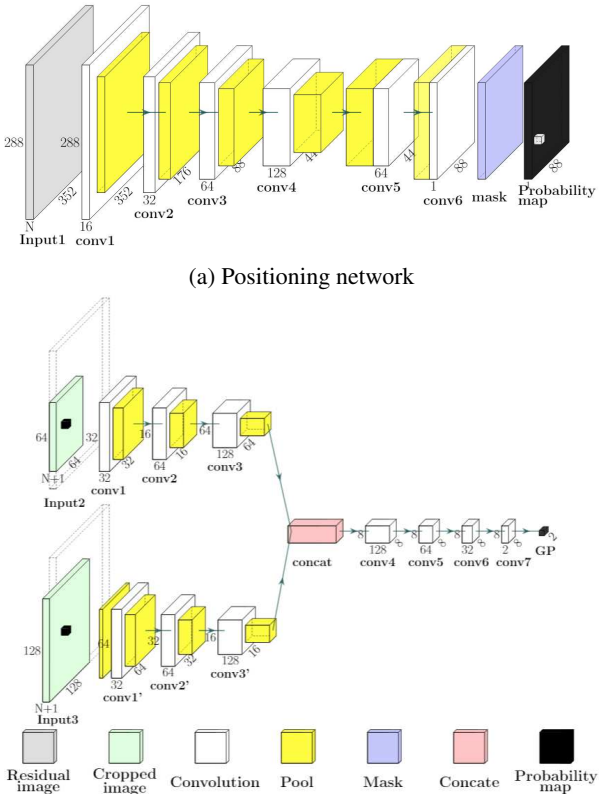
The multi-scale motion estimation network $M(\cdot;\theta_m)$ outputs an estimate $v(s_i)$ of the motion vector for a selected critical pixel s_i in the target frame I_n. This estimated motion vector establishes under the *constant intensity* assumption that the pixel value of I_n at s_i satisfies

$$I_n(s_i) = I_{n-1}(s_i + v(s_i)), \tag{6}$$

where bilinear interpolation is carried out to retrieve $I_{n-1}(s_i + v(s_i))$ when $s_i + v(s_i)$ is not on the sampling grid of I_{n-1}.

As shown in Fig. 4(b), The estimation of $v(s_i)$ involves extracting features of image patches cropped from context frames $\mathcal{I}_c$ at s_i. These image patches have two scales, 64×64 and 128×128, in order to capture both large and small motions. Before their features are extracted by the two separate convolutional networks of the same structure, they are resized to 64×64 by downsampling whenever applicable. The features extracted at different scales are then concatenated, processed by subsequent convolutional layers, and pooled globally to yield the final estimate $v(s_i)$.

It is important to note that our multi-scale motion estimation network performs motion estimation for individual critical pixels $\{s_i\}_{i=1}^K$ both locally and independently based on context frames $\mathcal{I}_c$ only. There is no regularization imposed on their estimated motion vectors $v(s_i)$. Moreover, in our current implementation, both the positioning and the motion estimation networks take only gray-scale inputs. Their outputs are however used for RGB predictions.



(a) Positioning network

(b) Multi-scale motion estimation network

Figure 4: Architectures of (a) the positioning and (b) the motion estimation networks.

5. Training

For training, we consider the minimization of $\mathcal{L}(\theta_p, \theta_m)$ in Eq. (5) with respect to θ_p, θ_m a reinforcement learning problem. As illustrated in Fig. 3, we view collectively the $(s_i, v(s_i))$ to be output in the i-th iteration as an *action* taken by the *agent* consisting of the two networks $P(\cdot;\theta_p), M(\cdot;\theta_m)$. The *environment* with which the agent interacts implements POBMC [3] and outputs in each iteration i the predicted frame $\hat{I}_n^{(i)}$ based on Eq. (4) together with the context frames $\mathcal{I}_c$ as a *state* signal for the next iteration. The process then repeats K times. In this paper, we adopt a *delayed reward* mechanism; that is, the agent receives no immediate reward until the end of K iterations, at which we compute the reward for the entire action sequence to be the mean squared error between I_n and $\hat{I}_n^{(K)}$ to match our objective in Eq. (5).

We train the positioning and motion estimation networks jointly with the REINFORCE algorithm [20, 14]. The agent is made purely stochastic by treating the policy for motion estimation as a conditional Gaussian distribution with mean

given by Eq. (3). To facilitate exploration, we add an entropy loss to the training objective for the positioning network [20], and apply the variance reduction strategy [14].

For better learning and faster convergence, we pre-train both the positioning and motion estimation networks supervisely, with ground-truths produced by a greedy pixel selection algorithm and EpicFlow [18]. The former chooses critical pixels based on the prediction residual of POBMC [3]. In an iteration i, the pixel $s \in I_n$ showing the largest prediction residual, *i.e.* $s_i = \arg\max_s \|I_n(s) - \hat{I}_n^{(i-1)}(s)\|$, is chosen as the ground-truth for the critical pixel s_i, whose motion vector $v(s_i)$ is in turn derived from EpicFlow [18]. In the process, we assume having full knowledge of the target frame I_n. Another point to be noted is that the training is carried out for $K = 20$ only. But, at test time, the number of critical pixels can be as high as $K = 100$.

6. EXPERIMENTAL RESULTS

This section evaluates our method against several state-of-the-art algorithms on the test partition of CaltechPed [4], UCF101 [19] and common intermediate format (CIF) [1] datasets for both one-step and multi-step predictions. Since there is no standardized common test conditions for the video prediction task, Table 1 summarizes the training sets used by the competing methods in our tests together with their model sizes. The objective quality measurements include Mean Squared Error (MSE), Peak-Signal-to-Noise Ratio (PSNR), Structure Similarity Index (SSIM), Learned Perceptual Image Patch Similarity (LPIPS) [23] and Frechet Video Distance (FVD) [21], where lower (respectively, higher) values of MSE, LPIPS [23], and FVD [21] (respectively, PSNR and SSIM) indicate better quality. Both LPIPS [23] and FVD [21] employ CNN features for quality assessment and were shown to correlate more highly with subjective quality. For FVD [21], which requires to extract enough independent sub-sequences features from a long video, we only apply FVD [21] to CaltechPed [4]. Note that all the methods are trained for one-step prediction only. For tests with multi-step prediction, the sliding window mechanism in Section 3.2 is applied.

6.1. Comparison on CaltechPed

For this experiment, we train our model on KITTI [6] dataset by randomly selecting 10000 frames from City and Road sequences. Both training and test videos are re-sized to 352×288. We obtain the results of BeyondMSE [13] and MCNet [22] by running their test software with pre-trained weights (see Table 1) on the same re-sized videos. For the results of DVF [11], we train the model using their training software on 15k frames from KITTI [6] dataset and test it following the same protocol. For the remaining methods including PredNet [12], DualGAN [10], SDC [17], we simply report their results in the papers.

Table 1: Training datasets used by the competing methods in tests on CaltechPed [4], UCF101 [19] and CIF [1].

Method	CaltechPed	UCF101	CIF	#param
Ours	KITTI	UCF101	UCF101	1M
DVF [11]	KITTI	UCF101	UCF101	2.2M
BMSE [13]	Sport 1M	Sport 1M	Sport 1M	8.9M
MCNet [22]	Sport 1M	Sport 1M	Sport 1M	6.9M
DualGAN [10]	KITTI	UCF101	-	113M
PredNet [12]	KITTI	-	-	6.9M
SDC [17]	Battlefield-1	-	-	160M

Table 2: Comparison of next frame prediction on CaltechPed [4], UCF101 [19] and CIF sequences [1]. MSE results are in 1e-3 and #ctx indicates the number of context frames.

Method	CaltechPed			UCF101		CIF		
	MSE	SSIM	#ctx	PSNR	SSIM	PSNR	SSIM	#ctx
Ours	2.65	0.878	3	30.80	0.910	27.90	0.890	4
BeyondMSE [13]	2.82	0.875	4	26.7	0.820	27.54	0.899	4
MCNet [22]	2.57	0.878	3	30.29	0.913	28.34	0.905	4
DVF [11]	6.65	0.801	3	31.54	0.918	22.75	0.653	4
PredNet [12]	3.13	0.884	4	-	-	-	-	-
DualGAN [10]	2.41	0.899	4	-	-	-	-	-
SDC [17]	1.62	0.918	5	-	-	-	-	-
CopyLast	5.31	0.815	-	-	-	-	-	-

Table 3: Comparison of FVD [21] for multi-step prediction.

-	Ours	MCNet	Bmse	DVF
CaltechPed	132	148	846	819

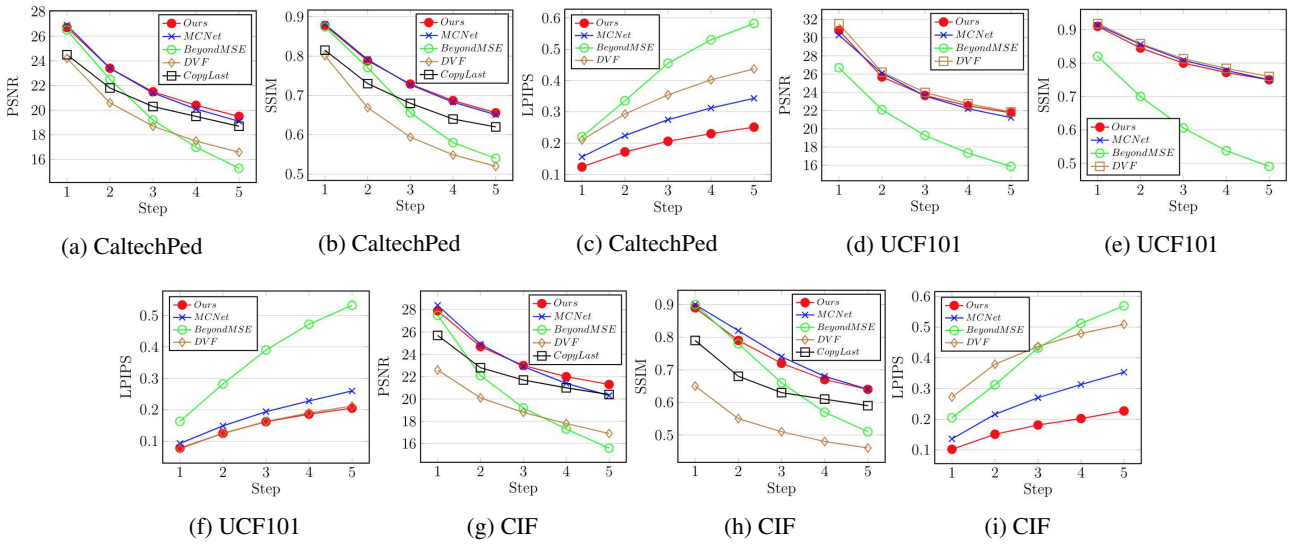
The left most section of Table 2 summarizes the objective quality comparison for one-step prediction. It is seen that with only 3 context frames and a model size almost one order of magnitude smaller, our model already performs comparably with MCNet [22] and BeyondMSE [13]. In this test, it outperforms DVF [11] significantly.

Fig. 5(a), 5(b), and 5(c) further report the PSNR, SSIM, and LPIPS [23] results, respectively, for the more challenging five-step prediction. In this task, our model achieves the best performance. Subjective quality comparison in Fig. 6 further confirms its superiority over the other competing methods. Our model is seen to be more robust to error propagation along the temporal dimension, producing clearer images at time instances further into the future than MCNet [22] and BeyondMSE [13], of which their images become increasingly blurred. It is also observed that DVF [11] is very sensitive to errors, rendering almost unrecognizable images at latter time instances. For FVD [21] measurement, we randomly pick three long sequences in CaltechPed [4]. In each of these three sequences, 256 features are extracted from mostly non-overlapping sub-sequences composed of 4 context and 9 predicted frames. The results are presented in Table 3, where our method is slightly better than MCNet [22] and outperforms the others significantly.

As for inference time comparison, our 20 critical pixels model requires around 0.43s compared with 0.03s of DVF [11], 0.06s of MCNet [22] and 1.13s of BeyondMSE [13].

6.2. Comparison on UCF101

This experiment compares prediction performance of BeyondMSE [13], MCNet [22], and DVF [11] on 10% of



(a) CaltechPed (b) CaltechPed (c) CaltechPed (d) UCF101 (e) UCF101

(f) UCF101 (g) CIF (h) CIF (i) CIF

Figure 5: Five-step prediction results for our model, BeyondMSE [13], MCNet [22] and DVF [11] on CaltechPed [4] (a)(b)(c), UCF101 [19] (d)(e)(f) and CIF sequences [1] (g)(h)(i).

Figure 6: Subjective comparison of five-step prediction on CaltechPed [4]: left to right, $t = 1, 2, ..., 5$.

UCF101 test set [19]. For both training and test, we use original videos of size 320×240. Specifically, we train our model on 12000 randomly selected frames, while training DVF [11] on 240000 frames following the protocol in [11]. We report results of BeyondMSE [13] and MCNet [22] by running their test software with pre-trained weights. All the methods adopt 4 context frames for prediction.

From Table 2 (the middle section), we see that DVF [11] achieves the best PSNR and SSIM performance for one-step prediction, while ours ranks second performing similarly to MCNet [22], with compatible SSIM but slightly higher PSNR. Since UCF101 [19] has only a small percentage of moving images, we conjecture that DVF [11] may have been overfit to this dataset. This is partly corroborated

Figure 7: Subjective comparison of five-step prediction on UCF101 [19]: left to right, $t = 1, 2, ..., 5$.

Figure 8: Subjective comparison of five-step prediction on CIF Mobile [1]: left to right, $t = 1, 2, ..., 5$.

by the results on CIF dataset [1], where the same model shows poor generalization.

The multi-step prediction results in Fig. 5(d), 5(e), and 5(f) again validate the robustness of our model to error propagation. In this task, our objective quality performs closely with MCNet [22] and DVF [11], and outperforms

BeyondMSE [13] considerably. In terms of subjective quality (see Fig. 7), our model shows a clear advantage over MCNet [22] and BeyondMSE [13] by preserving more texture details.

6.3. Comparison on CIF

This experiment tests the generalization ability of different models by evaluating their prediction performance on CIF dataset [1], which contains a variety of videos with diverse motion characteristics often used for the development of video codecs. It is however rarely utilized for prediction tasks. That is the main reason behind our choice. In this test, we simply apply the models trained for the previous UCF101 task without any fine tuning.

We see from the right most section of Table 2 that in one-step prediction, MCNet [22] achieves slightly higher PSNR than ours and BeyondMSE [13], while all three methods show similar SSIM. It is however noticed that DVF [11] shows extremely poor generalization, even thought it is trained on the same UCF101 dataset [19] as ours is.

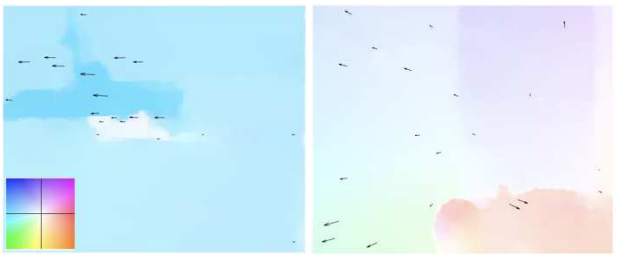
In multi-step prediction, as shown in Fig. 5(g), 5(h), and 5(i), our objective quality performs similarly to MCNet [22] while our subjective quality outperforms all the others significantly. Note however in Fig. 8 that the color distortion of MCNet [22] becomes more apparent as the prediction step increases.

6.4. Visualization of critical pixels

Fig. 9 visualizes the critical pixels and their motion vectors selected by our agent. The results are overlaid on top of the (ground-truth) optical flows given by EpicFlow [18], to verify the accuracy of the estimated motion vectors for critical pixels. We observe that the locations of critical pixels crucially depend on the spatial and motion characteristics of video frames. They are mostly concentrated in highly textured areas with large motion (see foreground objects in Fig. 9(a) and 9(b)), or with non-uniform motion (see background in Fig. 9(c) and 9(d)). It is also interesting to see that few critical pixels are placed in large motion areas with less texture variation, such as the road region at the bottom of Fig. 9d. In these areas, copying pixel values from the corresponding regions in the last frame still yields a good prediction result.

6.5. Generalization with variable critical pixels

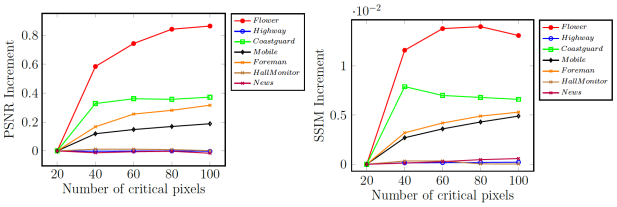
This experiment showcases the generalization of our model for a variable number of critical pixels. At training time, our model is learned with a maximum of 20 critical pixels. We then test its one-step prediction performance for a varying number of critical pixels ranging from 20 to 100. From Fig. 10, our model shows improving PSNR and SSIM when the number of critical pixels increases, especially on those fast motion sequences. Coastguard sequence is the only exception where the SSIM drops slightly with more critical pixels.



(a) Coastguard (b) Mobile

(c) UCF101 (d) CaltechPed

Figure 9: Visualization of critical pixels ($K = 20$) and their motion vectors overlaid on top of the ground-truth optical flows given by EpicFlow [18].



(a) PSNR (b) SSIM

Figure 10: PSNR and SSIM increments on CIF sequences [1] for a varying number of critical pixels.

7. Conclusion

In this paper, we leverage POBMC in a reinforcement learning framework to enable video prediction on a sparse motion field. In this way, only few critical pixels and their motion vectors need to be estimated for prediction, reducing greatly the model complexity. The identification of critical pixels and their motion estimation are addressed by two simple neural works trained with reinforcement learning. Our model achieves the state-of-the-art prediction performance on several common datasets. Its advantage is most obvious in multi-step prediction, where it produces subjectively more pleasing results than the other competing methods. Due to its relatively smaller size, our model can generalize better when learned on small data. The automation of deciding a proper number of critical pixels for individual video frames remains an open issue.

References

[1] ASU Video Trace Library. YUV Video Sequences. Retrieved March 22, 2019, from the World Wide Web: `http://trace.eas.asu.edu/yuv/`, 2009.

[2] Wonmin Byeon, Qin Wang, Rupesh Kumar Srivastava, and Petros Koumoutsakos. ContextVP: Fully Context-Aware Video Prediction. In *Proc. European Conference on Computer Vision (ECCV)*, 2018.

[3] Yi-Wen Chen and Wen-Hsiao Peng. Parametric OBMC for Pixel-adaptive Temporal Prediction on Irregular Motion Sampling Grids. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(1):113–127, 2012.

[4] Piotr Dollár, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian Detection: A Benchmark. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[5] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised Learning for Physical Interaction through Video Prediction. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2016.

[6] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision Meets Robotics: The KITTI Dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[7] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial Transformer Networks. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2015.

[8] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik G. Learned-Miller, and Jan Kautz. Super Slomo: High Quality Estimation of Multiple Intermediate Frames for Video Interpolation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[9] Nal Kalchbrenner, Aäron van den Oord, Karen Simonyan, Ivo Danihelka, Oriol Vinyals, Alex Graves, and Koray Kavukcuoglu. Video Pixel Networks. In *Proc. International Conference on Machine Learning (ICML)*, 2017.

[10] Xiaodan Liang, Lisa Lee, Wei Dai, and Eric P Xing. Dual Motion GAN for Future-flow Embedded Video Prediction. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2017.

[11] Ziwei Liu, Raymond A Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video Frame Synthesis using Deep Voxel Flow. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2017.

[12] William Lotter, Gabriel Kreiman, and David Cox. Deep Predictive Coding Networks for Video Prediction and Unsupervised Learning. In *Proc. International Conference on Learning Representations (ICLR)*, 2017.

[13] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep Multi-scale Video Prediction beyond Mean Square Error. In *Proc. International Conference on Learning Representations (ICLR)*, 2016.

[14] Volodymyr Mnih, Nicolas Heess, Alex Graves, and Koray Kavukcuoglu. Recurrent Models of Visual Attention. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2014.

[15] Niklaus, Simon, Long Mai, and Feng Liu. Video Frame Interpolation via Adaptive Separable Convolution. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2017.

[16] Patraucean, Viorica, Ankur Handa, and Roberto Cipolla. Spatio-Temporal Video Autoencoder with Differentiable Memory. In *Proc. The International Conference on Learning Representations (ICLR)*, 2016.

[17] Fitsum A Reda, Guilin Liu, Kevin J Shih, Robert Kirby, Jon Barker, David Tarjan, Andrew Tao, and Bryan Catanzaro. SDC-Net: Video Prediction using Spatially-displaced Convolution. In *Proc. European Conference on Computer Vision (ECCV)*, 2018.

[18] Jerome Revaud, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. Epicflow: Edge-preserving Interpolation of Correspondences for Optical Flow. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[19] Soomro, Khurram, Amir Roshan Roshan, and M. Shah. A Dataset of 101 Human Action Classes from Videos in The Wild. In *UCF Center for Research in Computer Vision (UCF CRCV)*, 2012.

[20] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2000.

[21] Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018.

[22] Ruben Villegas, Jimei Yang, Seunghoon Hong, Xunyu Lin, and Honglak Lee. Decomposing Motion and Content for Natural Video Sequence Prediction. In *Proc. International Conference on Learning Representations (ICLR)*, 2017.

[23] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.