2022 26th International Conference on Pattern Recognition (ICPR)
August 21-25, 2022, Montréal, Québec, Canada

# VPTR: Efficient Transformers for Video Prediction

Xi Ye
LITIV Laboratory, Polytechnique Montréal
Montréal, Canada
Email: xi.ye@polymtl.ca

Guillaume-Alexandre Bilodeau
LITIV Laboratory, Polytechnique Montréal
Montréal, Canada
Email: gabilodeau@polymtl.ca

2022 26th International Conference on Pattern Recognition (ICPR) | 978-1-6654-9062-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICPR56361.2022.9956707

***Abstract*—In this paper, we propose a new Transformer block for video future frames prediction based on an efficient local spatial-temporal separation attention mechanism. Based on this new Transformer block, a fully autoregressive video future frames prediction Transformer is proposed. In addition, a non-autoregressive video prediction Transformer is also proposed to increase the inference speed and reduce the accumulated inference errors of its autoregressive counterpart. In order to avoid the prediction of very similar future frames, a contrastive feature loss is applied to maximize the mutual information between predicted and ground-truth future frame features. This work is the first that makes a formal comparison of the two types of attention-based video future frames prediction models over different scenarios. The proposed models reach a performance competitive with more complex state-of-the-art models. The source code is available at *https://github.com/XiYe20/VPTR*.**

## I. Introduction

Video future frames prediction (VFFP) is applied to many research areas, for instance, intelligent agents [1], [2], autonomous vehicles [3], model-based reinforcement learning [4]. More recently, it has drawn a lot of attention since it is naturally a good self-supervised learning task [5], [6].

In this paper, we focus on the most common video prediction task, i.e. predicting $N$ future frames given $L$ past frames, with $L$ and $N$ greater than 1. For training a deep learning VFFP model, we can formalize the task to be $\arg\max_\theta p(\hat{x}_{L+N}, ..., \hat{x}_{L+1}|x_L, ..., x_1; \theta)$, where $\hat{x}_t$ and $x_t$ denote the predicted future frames and input past frames respectively, $\theta$ denotes the model parameters.

Even though many deep learning-based VFFP models have been proposed, some challenges still remain to be solved. Almost all the state-of-the-art (SOTA) VFFP models are based on ConvLSTMs, i.e. convolutional short-term memory networks, which are efficient and powerful. Nevertheless, they suffer from some inherent problems of recurrent neural networks (RNNs), such as slow training and inference speed, error accumulation during inference, gradient vanishing, and predicted frames quality degradation. Researchers keep improving the performance by developing more and more sophisticated ConvLSTM-based models. For instance, by integrating custom motion-aware units into ConvLSTM [7], or building complex memory modules to store the motion context [8].

Inspired by the great success of Transformers in NLP, more and more researchers are starting to adapt Transformers for various computer vision tasks [9], [10], [11], [12], including few recent works for VFFP [13], [14], [15]. However, it is computational expensive to apply Transformer to high dimensional visual features. We still need further research about more efficient visual Transformers, especially for videos. *Therefore, we propose a novel efficient Transformer block with smaller complexity, and we developed a new video prediction Transformer (VPTR) based on it.*

Among the Transformers-based VFFP models [13], [14], [15] that we mentioned earlier, some of them are autoregressive models while some others are non-autoregressive models, and they are based on different attention mechanisms, e.g. a custom convolution multi-head attention (MHA) [13] and standard dot-product MHA [14], [15]. There is no formal comparison of the two typical approaches (autoregressive vs non-autoregressive) to use Transformer-based VFFP models so far. Thus, we developed an fully autoregressive VPTR (VPTR-FAR) and a non-autoregressive VPTR (VPTR-NAR). The two VPTR variants share the same attention mechanism and same number of Transformer block layers, which guarantees a fair comparison between the two approaches.

Our main contributions are summarized as:

1) We proposed a new efficient Transformer block for spatio-temporal feature learning by combining spatial local attention and temporal attention in two steps. The new Transformer block successfully reduces the complexity of a standard Transformer block with respect to same input spatio-temporal feature size, specifically, from $\mathcal{O}((THW)^2)$ to $\mathcal{O}(\frac{H^2W^2}{P^2} + T^2)$.

2) Two VPTR models, VPTR-NAR and VPTR-FAR, were developed. We show that the proposed simple attention-based VPTRs are capable of reaching and outperforming more complex SOTA ConvLSTM-based VFFP models.

3) A formal comparison of two VPTR variants was conducted. The results show that VPTR-NAR has a faster inference speed and smaller accumulation of errors during inference, but it is more difficult to train. We solved the training problem of VPTR-NAR by employing a contrastive feature loss which maximizes the mutual information of predicted and ground-truth future frame features.

4) We found that given the same number of Transformer block layers, VPTR-FAR has a worse generalization performance due to the accumulated inference errors, which are introduced by the discrepancy between train and test behaviors. We also found that recurrent inference over pixel space introduces less accumulation errors than recurrent inference over latent space in the case of VPTR-FAR.

## II. Related Work

Almost all the SOTA deep learning-based VFFP models are ConvLSTM-based autoencoders, where the encoder extracts the representations of past frames, and then the decoder generates future frame pixels based on those representations [16], [17], [18], [7], [8]. In general, the SOTA models rely on complex ConvLSTM models that integrates attention mechanism or memory augmented modules. For example, LMC-Memory model [8] stores the long-term motion context by a novel memory alignment learning, and the motion information is recalled during test to facilitate the long-term prediction. Zhang et al. [7] proposed a attention-based motion-aware unit to increase the temporal receptive field of RNNs.

The ConvLSTM-based models are flexible and efficient, but recurrent prediction is slow. Therefore, standard CNNs or 3D-CNNs are also used as the backbones of VFFP to generate multiple future frames simultaneously [19], [20], [21], [22]. Besides, the future prediction is by nature multimodal [23], i.e. stochastic. Some VFFP models aim to solve this problem based on VAEs, such as SV2P [24], SVG-LP [25], improved conditional VRNNs [26]. Stochasticity learning is challenging and thus most VFFP models ignore it. A detail survey of VFFP models can be found in [23].

Recently, Transformers were applied for VFFP. The ConvTransformer [13] model follows the architecture of DETR [9]. DETR follows a classical neural machine translation (NMT) Transformer architecture. It also inspired the development of our VPTR-NAR. Despite the similarities, our VPTR-NAR is different from ConvTransformer with respect to the fundamental attention mechanism. Specifically, ConvTransformer proposed a custom hybrid multi-head attention module based on convolution, but our VPTR-NAR uses the standard multi-head dot-product attention. Another more recent model (VideoGPT) [14] takes a 3D-CNN as backbone to encode video clips into spatial-temporal features, which are then flattened to be a sequence to train a standard Transformer with the GPT manner [27], [28]. VideoGPT shares a similar architecture and train/test behaviours as our VPTR-FAR. But VideoGPT performs the attention along the spatial and temporal dimensions jointly while our VPTR-FAR performs the attention along the spatial and temporal dimensions separately. More importantly, VideoGPT downsamples the time dimension of input videos by 3D-CNN and thus helps the temporal information modeling. In contrast, our VPTR models solely depend on attention for a full temporal information modeling, without downsampling. Another recent work NÜWA [15] shares a similar idea to VideoGPT.

**Efficient visual Transformers.** To reduce the computation cost for visual Transformers, some models reduce the flattened sequence length by different methods. ViT and the successive works [10], [29], [12] divided input features into local patches, either 2D or 3D, and then tokenize the local patch by concatenation or Pooling [30]. Some other models introduce sparse attention to reduce the complexity, e.g. restricting the attention over a local region [31], [32], [33], or decomposing the global attention into a series of axial-attention [34], [35], [12]. HRFormer [33] is an example of local region attention-based Transformers, which is designed for image classification and dense prediction.

Specifically, a HRFormer block is composed of a local-window multi-head self attention layer and a depth-wise convolution feed-forward network. The input feature maps $Z \in \mathbb{R}^{H\times W\times C}$ are firstly evenly divided into $P$ non-overlapping local patches, each patch is $Z_p \in \mathbb{R}^{\frac{H}{P}\times\frac{W}{P}\times C}$. Then a multi-head self attention is performed for each patch. Finally, the depth-wise convolution is used to exchange information among different local patches.

## III. The Proposed VPTR Models

### A. Overall framework of VPTR

Our overall VPTR framework is illustrated in Fig. 1. A CNN encoder shared by all the past frames extracts the visual features of each frame. Then a VPTR is taken to predict the visual features of each future frame based on the past frame features. The detail architectures of two different VPTR variants are described in the following subsections. In order to make the model architecture simple and easier to train, there is no skip connections between encoder and decoder.

### B. Encoder and decoder

We adapted the ResNet-based autoencoder from the Pix2Pix model [36]. The output feature channels of the encoder and input feature channels of the decoder are modified to be of size $d_{model}$ to match with the VPTR input and output size. The loss function to train the encoder and decoder is defined as follows,

$$\mathcal{L}_{rec} = \mathcal{L}_2(X,\hat{X}) + \mathcal{L}_{gdl}(X,\hat{X}) + \lambda_1 \arg\min_G \max_D \mathcal{L}_{GAN}(G,D), \quad (1)$$

where $\mathcal{L}_2$ denotes the MSE loss (Eq. 2) and $\mathcal{L}_{gdl}$ denotes image gradient difference loss [19] (Eq. 3), $X$ and $\hat{X}$ denote the original frames and reconstructed frames respectively, $x_i$ denotes a single frame, $\lambda_1$ and $\alpha$ are hyperparameters. $\mathcal{L}_{GAN}$ denotes the GAN loss (Eq. 4), where $D$ denotes a discriminator, which is not shown in Fig. 1, and the combination of the encoder and decoder is considered to be a generator $G$. We train $\mathcal{L}_{GAN}$ with the PatchGAN [36] manner.

$$\mathcal{L}_2(X,\hat{X}) = \sum_{i=1}^{n} \|x_i - \hat{x}_i\|_2^2 \quad (2)$$

$$\mathcal{L}_{gdl}(X,\hat{X}) = \sum_{i=1}^{n}\sum_{i,j} \Big|\big||x_{i,j} - x_{i-1,j}| - |\hat{x}_{i,j} - \hat{x}_{i-1,j}|\big|^{\alpha} + \big|\big||x_{i,j-1} - x_{i,j}| - |\hat{x}_{i,j-1} - \hat{x}_{i,j}|\big|^{\alpha} \quad (3)$$

$$\mathcal{L}_{GAN}(G,D) = \mathbb{E}_X[logD(X)] + \mathbb{E}_{\hat{X}}[log(1-D(G(X))] \quad (4)$$

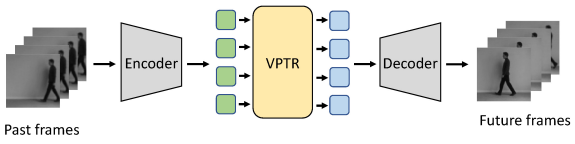


Fig. 1: Overall framework of VPTR. Green squares and blue squares denote the past frame features and future frames features respectively.

### C. VidHRFormer Block

We proposed a new Transformer block based on the HRFormer block [33] for video processing, which is named VidHRFormer block. The detail architecture of a VidHRFormer block is shown in the gray area of Fig. 2(a). Essentially, we integrate a temporal multi-head attention layer, together with some other necessary feed-forward and normalization layers, into the HRFormer block.

**Local spatial multi-head self-attention (MHSA).** Given a spatiotemporal feature map $Z \in \mathbb{R}^{N\times T\times H\times W\times d_{model}}$, we firstly reshape and evenly divide it into $P$ local patches $\{Z_1, Z_2, ..., Z_P\}$ along the $H$ and $W$ dimensions, where $Z_p \in \mathbb{R}^{(NT)\times K^2\times d_{model}}$, and each local patch is of size $K \times K$, with $P = \frac{HW}{K^2}$ patches in total. $MHSA(Z_p) = Concat[head(Z_p)_1, ..., head(Z_p)_h]$, where $head(Z_p)_i \in \mathbb{R}^{K^2\times \frac{d_{model}}{h}}$ is formulated as

$$head(Z_p)_i = softmax[\frac{((Z_p^Q W_i^Q)(Z_p^K W_i^K)}{\sqrt{d_{model}/h}}]Z_p W_i^V, \quad (5)$$

where $W_i^Q$, $W_i^K$, $W_i^V$ are linear projection matrices for the query, key and value of each head $i$ respectively, $Z_p^Q$ and $Z_p^K$ denote the key and query for attention. We may use a fixed absolute 2D positional encoding [37], or a relative positional encoding (RPE) [38] of the local patch to get $Z_p^Q$ and $Z_p^K$. We compared the two different positional encodings in the experiments. The complexity of local spatial MHSA is $\mathcal{O}(\frac{H^2W^2}{P^2})$.

**Convolutional feed-forward neural network (Conv FFN).** After the local spatial MHSA, $\{Z_1, Z_2, ..., Z_P\}$ are assembled back to be $Z \in \mathbb{R}^{(NT)\times H\times W\times d_{model}}$. The Conv FFN layer is composed of a $3 \times 3$ depth-wise convolution and two point-wise MLPs. Note that all the normalization layers in Conv FFN are layer normalization, instead of batch normalization used in the original HRFormer block.

**Temporal MHSA.** The local spatial MHSA and Conv-FFN are shared by every frame feature. A temporal MHSA is placed on top of them to model the temporal dependency between frames. We reshape the input feature map $Z \in \mathbb{R}^{(NT)\times H\times W\times d_{model}}$ to be $Z \in \mathbb{R}^{(NHW)\times T\times d_{model}}$. Temporal MHSA is a standard multi-head self-attention similar to the local spatial MHSA, except that there is no local patch division and it takes a fixed absolute 1D positional encoding of time. The complexity of temporal MHSA is $\mathcal{O}(T^2)$. The temporal MHSA is followed by a MLP feed-forward neural network as in a standard Transformer, and the output feature map is reshaped back to be $Z \in \mathbb{R}^{N\times T\times H\times W\times d_{model}}$ for the next layer of VidHRFormer block.

In summary, the proposed VidHRFormer block reduces the compute complexity from $\mathcal{O}((THW)^2)$ to be $\mathcal{O}(\frac{H^2W^2}{P^2} + T^2)$ by combining spatial local window attention and temporal attention in two steps. Based on the VidHRFormer, we develop two different VPTR models.

### D. VPTR-FAR

The fully autoregressive VPTR model is simply a stack of multiple VidHRFormer blocks. The architecture is shown in Fig. 2(a). Theoretically, given a well-trained CNN encoder and decoder, the VPTR-FAR parameterizes the following distribution:

$$p(x_1, ..., x_L, ..., x_{L+N}) = \prod_{t=1}^{L+N} p(x_t|x_{t-1}, ...x_1) \quad (6)$$

In other words, VPTR-FAR predicts the next frame conditioned on all previous frames, which is also the most common paradigm for most SOTA VFFP models. An attention mask is applied to the temporal MHSA module to impose the conditional dependency between the next frame and previous frames.

During training, we feed the ground-truth frames $\{x_1, ..., x_{L+N-1}\}$ into the encoder, which generates the feature sequence $\{z_1, ..., z_{L+N-1}\}$. VPTR-FAR then predicts the future feature sequence $\{\hat{z}_2, ..., \hat{z}_{L+N}\}$, which is then decoded by the decoder to generate frames $\{\hat{x}_2, ..., \hat{x}_{L+N}\}$. The training loss of VPTR-FAR is:

$$\mathcal{L}_{FAR} = \sum_{t=2}^{L+N} \mathcal{L}_2(x_t, \hat{x}_t) + \sum_{t=2}^{L+N} \mathcal{L}_{gdl}(x_t, \hat{x}_t) \quad (7)$$

During test, we firstly get the ground-truth past frames features $\{z_1, ..., z_L\}$. Then there are two different ways of recurrently predicting the future frames. The first method is recurrently generating all the future frame features only by the VPTR-FAR module, i.e. $\hat{z}_t = \mathcal{T}(z_1, ..., z_{t-1}), t \in [L+1, ...L+N]$, where $\mathcal{T}$ denotes the VPTR-FAR module. Then we get $\hat{x}_t = Dec(\hat{z}_t), t \in [L+1, ...L+N]$, where $Dec$ denotes the CNN frame decoder. The second prediction method introduces two additional steps. Particularly, $\hat{z}_t = Enc(Dec(\mathcal{T}(z_1, ..., z_{t-1}))), t \in [L+1, ...L+N]$, where $Enc$ denotes the CNN frame encoder. In short, we decode each future feature to be frame $\hat{x}_t$, and then encode the frame back into a latent feature before the prediction of next future frame feature. The second way significantly reduces the accumulated error during inference, and the reasons are analyzed in the experiments section.

### E. VPTR-NAR

Inspired by the achitecture of DETR [37], a non-autoregressive variant is proposed to increase the inference speed and reduce inference accumulation error of autoregressive models. VPTR-NAR is illustrated in Fig. 2(b). It

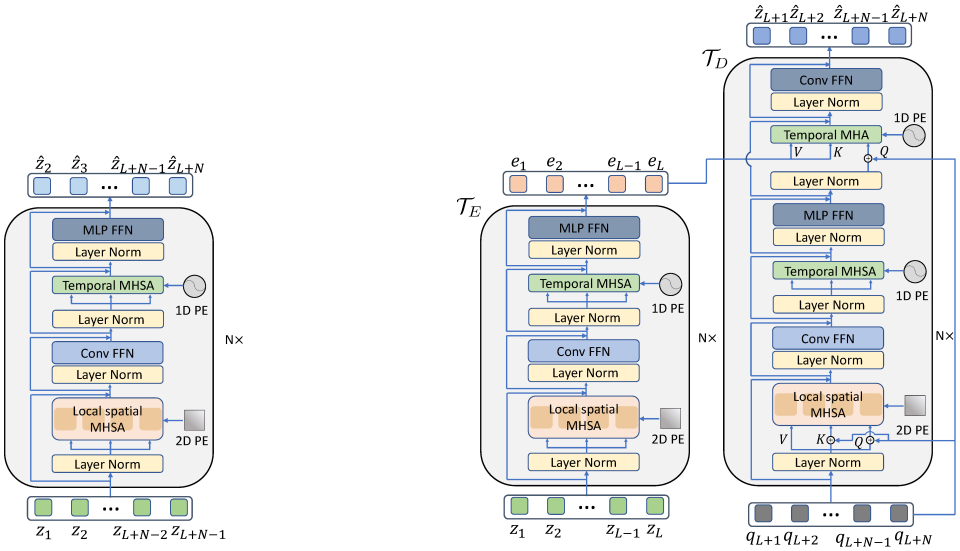


Fig. 2: (a) VPTR-FAR. The gray area indicates the proposed basic VidHRFormer block. A temporal attention mask is applied to the Temporal MHSA module for VPTR-FAR. (b)VPTR-NAR. The left part is the Transformer encoder and right part is the non-autoregressive Transformer decoder.

consists of a Transformer encoder and decoder, where the encoder $\mathcal{T}_E$ encodes all past frame features $z_t, t \in [1, L]$ to be $e_t, t \in [1, L]$, which are normally named as "memories" in NLP. The architecture of $\mathcal{T}_E$, left part of Fig. 2(b), is the same as the VPTR-FAR, except that there is no temporal attention mask for the temporal MHSA module.

The decoder $\mathcal{T}_D$ of VPTR-NAR, right part of Fig. 2(b), includes two more layers compared with $\mathcal{T}_E$. A temporal multi-head attention (MHA) layer and another output Conv FFN layer. The Temporal MHA layer is also called the encoder-decoder attention layer, which takes the memories as value and key, while the query is derived from the future frame query sequence $\{q_{L+1}, ..., q_{L+N}\}$, where $q_t \in \mathbb{R}^{H \times W \times C}, t \in [L+1, L+N]$. $\{q_{L+1}, ..., q_{L+N}\}$ is randomly initialized and updated during training. Note that there is no temporal attention mask for Temporal MHSA layer, since we do not need to impose conditional dependency between each future frame query. Theoretically, VPTR-NAR directly models the following conditional distribution:

$$p(x_{L+N}, ..., x_{L+1} | x_L, ..., x_1) \tag{8}$$

**Contrastive feature loss for VPTR-NAR.** We failed to train VPTR-NAR with a loss only composed by MSE and GDL, i.e., $\mathcal{L} = \sum_{t=L+1}^{L+N} \mathcal{L}_2(x_t, \hat{x}_t) + \mathcal{L}_{gdl}(x_t, \hat{x}_t)$, since it is easy to fall into some local minimums. Specifically, all the predicted future frames are somewhat similar to each other. A similar phenomenon is also observed in the non-autoregressive NMT models, where the Transformer decoder frequently generate repeated tokens [39]. To solve this problem, we impose another contrastive feature loss $\mathcal{L}_c$ [40] to maximize the mutual information between predicted future frame feature $\hat{z}_t$ and the future frame feature $z_t$ (ground-truth) generated by the CNN encoder, where $t \in [L+1, L+N]$. $\mathcal{L}_c$ is formulated as follows,

$$\mathcal{L}_c(z_t, \hat{z}_t) = \frac{1}{2} \sum_{s=1}^{S_l} l_c(\hat{v}_s, v_s, sg(\bar{v}_s)) + l_c(v_s, \hat{v}_s, sg(\hat{\bar{v}}_s)), \tag{9}$$

where $v_s \in \mathbb{R}^{d_{model}}$ denotes a feature vector at spatial location $s$ of $z_t$, $\bar{v}_s \in \mathbb{R}^{(S_l-1) \times d_{model}}$ denotes the collection of feature vectors at all other spatial locations of $z_t$. $S_l = H \times W$ is the total number of spatial locations in a feature map. $\hat{v}_s$ and $\hat{\bar{v}}_s$ of $\hat{z}_t$ are defined in the same way. $sg$ is the stop gradient operation and $l_c$ is the info-NCE based contrastive loss defined by

$$l_c(v, v^+, v^-) = \\ -log \frac{exp(s(v, v^+))}{exp(s(v, v^+)) + \sum_{m=1}^{M} exp(s(v, v^-))}. \tag{10}$$

Given a feature vector $v \in \mathbb{R}^{d_{model}}$, $v^+ \in \mathbb{R}^{d_{model}}$ is the spatially-corresponding ground-truth feature vector, and $v^- \in \mathbb{R}^{M \times d_{model}}$ denotes the $M$ other spatially different ground-truth feature vectors. $s(v1, v2)$ measures the feature dot-product similarity. Finally, the training loss function for VPTR-NAR is defined as

$$\mathcal{L}_{NAR} = \sum_{t=L+1}^{L+N} \mathcal{L}_2(x_t, \hat{x}_t) + \mathcal{L}_{gdl}(x_t, \hat{x}_t) + \lambda_2 \mathcal{L}_c(z_t, \hat{z}_t). \tag{11}$$

During test, VPTR-NAR predicts $N$ future frames simultaneously, instead of recurrently.

### F. Training strategy

The whole VFFP model training process is divided into two stages. For stage one, we ignore the VPTR module and only train the encoder and decoder as a normal autoencoder with the loss function in Eq. 1, which aims to reconstruct all the frames of the whole training set perfectly. During stage two, we only update parameters of the VPTR module while the well-trained encoder and decoder are fixed. VPTR-FAR and VPTR-NAR are trained with the loss function in Eq. 7 and Eq. 11 respectively. It is well-known that Transformers are hard to train, therefore we proposed this two-stage training strategy to ease the training. Besides, the two-stage training strategy is flexible and allows us to test different VPTR variants without repetitive training of the encoder and decoder. Experimental results show that a final joint finetuning of autoencoder and VPTR is not helpful.

## IV. Experiments

### A. Datasets and Metrics

We evaluate the proposed VPTR models over three datasets, KTH [41], MovingMNIST [42] and BAIR [43]. For KTH and Moving MNIST, VPTR models are trained to predict 10 future frames given 10 past frames. For BAIR dataset, VPTR models are trained to predict 10 future frames given 2 past frames. All datasets are trained with a resolution of $64 \times 64$. We process the KTH dataset as previous works [44], [17]. Random horizontal and vertical flip of each video clip are utilized as data augmentation. We use the MovingMNIST created by E3D-LSTM [45], which takes the same data augmentation method as KTH. There is no data augmentation for BAIR.

**Metrics.** Learned Perceptual Image Patch Similarity (LPIPS)[46] and Structural Similarity Index Measure (SSIM) are used to evaluate all the three datasets. Peak Signal-to-Noise Ratio (PSNR) is used to evaluate the KTH and BAIR dataset, and Mean Square Error (MSE) is used to evaluate the MovingMNIST dataset. All the LPIPS values are presented in $10^{-3}$ scale.

### B. Implementation

**Training stage one.** In Eq. 1, $\lambda_1 = 0.01$ for KTH and MovingMNIST, $\lambda_1 = 0$ for the BAIR dataset. The optimizer is Adam [47], with a learning rate of $2e^{-4}$. **Training stage two.** For the visual features of each frame, $H = 8, W = 8, d_{model} = 528$. $K = 4$ for the local spatial MHSA. The Transformer of VPTR-FAR includes 12 layers. For VPTR-NAR, the number of layers of $\mathcal{T}_E$ is 4, and the number of layers of $\mathcal{T}_D$ is 8. We take AdamW [48] with a learning rate of $1e^{-4}$ for the optimization of all Transformers. Gradient clipping is taken to stabilize the training. For the loss function of VPTR-NAR (Eq. 11), $\lambda_2 = 0.1$.

### C. Results

**Results on KTH.** The best results of the two VPTR variants are recorded in Table I. Following the evaluation protocol of previous works, we extend the prediction length to be 20 frames during test. Compared with the SOTA models, the proposed VPTR models reach competitive performances in terms of PSNR and SSIM. Notably, both two VPTR variants outperform the SOTAs in terms of LPIPS by a large margin. Some prediction examples are shown in Fig. 3. It shows that the predicted arm motion by VPTRs is more aligned with the ground-truth, which indicates that the VPTRs more successfully capture the cyclic hand waving movements that only depends on the past frames, in contrast to LMC-Memory that recalls some inaccurate motion from the memory bank.

TABLE I: Results on KTH and MovingMNIST. ↑: higher is better, ↓: lower is better. **Boldface**: best results.

| Methods | KTH $10 \rightarrow 20$ | | | MovingMNIST $10 \rightarrow 10$ | | |
|---|---|---|---|---|---|---|
| | PSNR↑ | SSIM↑ | LPIPS↓ | MSE↓ | SSIM↑ | LPIPS↓ |
| MCNET [44] | 25.95 | 0.804 | - | - | - | - |
| PredRNN++ [49] | 28.47 | 0.865 | 228.9 | 46.5 | 0.898 | 59.5 |
| E3D-LSTM [45] | 29.31 | 0.879 | - | **41.3** | 0.910 | - |
| STMFANet [17] | **29.85** | 0.893 | 118.1 | - | - | - |
| Conv-TT-LSTM [50] | 28.36 | **0.907** | 133.4 | 53.0 | 0.915 | **40.5** |
| LMC-Memory [8] | 28.61 | 0.894 | 133.3 | 41.5 | **0.924** | 46.9 |
| VPTR-NAR | 26.96 | 0.879 | 86.1 | 63.6 | 0.882 | 107.5 |
| VPTR-FAR | 26.13 | 0.859 | **79.6** | 107.2 | 0.844 | 157.8 |

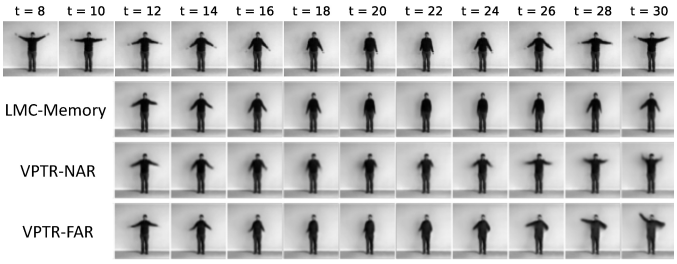


Fig. 3: Qualitative results on KTH dataset. The first row is ground-truth. For the past frames, $t \in [1, 10]$. For the future frames, $t \in [11, 30]$.

**Results on MovingMNIST.** The right part of Table I shows the results on the MovingMNIST dataset. We observe that the SSIM of the VPTR variants is close to the SOTAs, but there are large gaps in terms of MSE and LPIPS, especially for VPTR-FAR. Qualitative examination shows that VPTRs make poor predictions for the overlapping characters.

**Results on BAIR.** Compared with KTH and MovingMNIST, BAIR is more challenging, because the robot arm motion is random and only two past frames are given for the prediction. From Table II, we find that VPTR-NAR outperforms STMFANet [17] in terms of SSIM and LPIPS. Our good performance can be attribute to the large model capacity of VPTRs and the large size of BAIR dataset. We note however that the predicted robot arm becomes blurry after the first few frames, due the deterministic nature of VPTRs. Our VPTRs could be extended to be stochastic models easily, and we expect that the stochastic version of VPTRs would achieve an even better performance on the BAIR dataset.

### D. Ablation Study

**RPE.** The VPTRs with fixed absolute positional encodings are taken as the base models, i.e. VPTR-NAR-BASE and

TABLE II: Results on BAIR. ↑: higher is better, ↓: lower is better. **Boldface**: best results.

| Methods | PSNR↑ | SSIM↑ | LPIPS↓ |
|---|---|---|---|
| SV2P [24] | 20.36 | 0.817 | 91.4 |
| SVG-LP [25] | 17.72 | 0.815 | 60.3 |
| Improved VRNN [26] | - | 0.822 | 55.0 |
| STMFANet [17] | **21.02** | 0.844 | 93.6 |
| VPTR-NAR | 19.40 | **0.852** | **53.9** |
| VPTR-FAR | 18.63 | 0.824 | 69.3 |

TABLE III: Ablation study on KTH and MovingMNIST. ↑: higher is better, ↓: lower is better. **Boldface**: best results.

| Methods | KTH $10 \rightarrow 20$ | | | MovingMNIST $10 \rightarrow 10$ | | |
|---|---|---|---|---|---|---|
| | PSNR↑ | SSIM↑ | LPIPS↓ | MSE↓ | SSIM↑ | LPIPS↓ |
| VPTR-NAR-BASE | 26.92 | **0.881** | 94.6 | 64.2 | 0.880 | 114.2 |
| VPTR-NAR-RPE | **26.96** | 0.879 | 86.1 | **63.6** | **0.882** | **107.5** |
| VPTR-NAR-FEDA | 26.25 | 0.872 | 101.1 | 68.0 | 0.872 | 128.7 |
| VPTR-FAR-BASE | 25.71 | 0.816 | **79.5** | 108.3 | 0.843 | 157.3 |
| VPTR-FAR-RPE | 26.13 | 0.859 | 79.6 | 107.2 | 0.844 | 157.8 |
| VPTR-FAR-RIL | 21.61 | 0.678 | 192.7 | 138.2 | 0.821 | 445.7 |

VPTR-FAR-BASE in Table III. To investigate the influence of relative positional encodings, we get VPTR-NAR-RPE and VPTR-FAR-RPE by substituting the 2D absolute positional encoding of all local spatial MHSA module with a learned 2D RPE. We argue that RPE is beneficial because both VPTR-FAR-RPE and VPTR-NAR-RPE outperform the base models with regard to most metrics on the two datasets.

**Spatial-temporal separation attention.** The separation of spatial and temporal attention reduces the complexity, but it also means that a feature at one location only attends to partial locations of the whole spatiotemporal space. To investigate the influence of the separated attention, we replace the encoder-decoder attention layers of VPTR-NAR with a full spatiotemporal attention, which has a complexity of $\mathcal{O}(\frac{H^2W^2T^2}{P^2})$. The increased computation cost is affordable as we only replace the encoder-decoder attention layers. Comparing the VPTR-NAR-FEDA with the base model, where FEDA denotes "full encoder-decoder attention", we find that FEDA is not beneficial. It indicates that the alternate stacking of multiple spatial and temporal attention layers is capable of propagating global information from past frames to future frames.

**Autoregressive inference methods.** As we have described in Section III-D, we can perform recurrently inference over latent space (RIL) or recurrently inference over pixel space (RIP) for VPTR-FAR. VPTR-FAR-BASE is evaluated by RIP. Even though RIL is little faster than RIP, VPTR-FAR-BASE outperforms VPTR-FAR-RIL by a large margin. Severe accumulation of errors is observed for VPTR-FAR-RIL.

We believe the reason is that VPTR-FAR receive only supervision from the pixel space during training. There is no direct constraints on the distance between the feature space predicted by the Transformer and the feature space generated by the CNN encoder. Furthermore, the latent space dimension of the autoencoder is greater than the pixel space dimension, which is a common case for VFFP, as we expect a good reconstruction visual quality. Therefore, recurrent inference only depending on the Transformer predictor would make the predicted features deviate from the ground-truth (learned by autoencoder during stage one) features quickly. But decoding the feature firstly and then encoding it back into latent space restrict the deviation to some degree.

### *E. Comparison of VPTR variants*

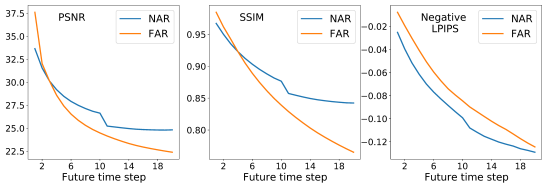


Fig. 4: Results of VPTR variants on KTH for increasing prediction steps.

For a better visualization, we plotted the metrics curve of VPTR-NAR-BASE and VPTR-FAR-BASE with respect to the predicted future time steps in Fig. 4. For the first few predicted frames, VPTR-FAR achieves a better PSNR and SSIM than VPTR-NAR, but the values drop quickly due to the accumulated errors introduced by recurrent inference. It shows that VPTR-NAR has a smaller quality degradation in terms of PSNR and SSIM. For the last 10 steps of the LPIPS curve, VPTR-NAR also has a smaller slope than VPTR-FAR.

The accumulation of errors in VPTR-FAR is mainly due to the discrepancy between training and testing behaviors. Specifically, the previously predicted frames are used during inference instead of the ground-truth as during training, which leads to a worse generalization ability of VPTR-FAR given the same number of Transformer layers as VPTR-NAR. In contrast, there is no discrepancy the between training and testing behaviors of VPTR-NAR. However, it is more difficult for the VPTR-NAR to estimate the joint distribution directly, so an additional contrastive feature loss is required.

Another advantage of VPTR-NAR is the faster inference speed. For VPTR-NAR, predicting $N$ frames has a complexity of $\mathcal{O}(N^2)$, but the complexity for VPTR-FAR is $\mathcal{O}(\sum_{n=1}^{N} n^2)$. For simplicity, in this assessment, we ignored the spatial dimensions of features, computation cost of processing past frames, and supposed that the future frames length of inference is same as of the training. However, the model size of VPTR-NAR is larger because of the learned future frame queries.

## V. Conclusion

In this paper, we proposed an efficient VidHRFormer block for spatiotemporal representation learning, and two different VFFP models are developed based on it. We expect that the proposed VidHRFormer block could be used as a backbone for many other video processing tasks. We compared the performance of proposed VPTRs with SOTA models on various datasets, and we are competitive with more complex models. Finally, we analyzed the influence of different modules for two VPTR variants by a thorough ablation study, and we observed that VPTR-NAR achieves a better performance than VPTR-FAR.

## REFERENCES


[1] W. Liu, W. Luo, D. Lian, and S. Gao, "Future Frame Prediction for Anomaly Detection – A New Baseline," in *CVPR*, 2018, pp. 6536–6545.

[2] Y. Lu, K. M. Kumar, S. s. Nabavi, and Y. Wang, "Future Frame Prediction Using Convolutional VRNN for Anomaly Detection," in *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Sep. 2019, pp. 1–8.

[3] J.-A. Bolte, A. Bar, D. Lipinski, and T. Fingscheidt, "Towards Corner Case Detection for Autonomous Driving," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2019, pp. 438–445, iSSN: 2642-7214.

[4] F. Leibfried, N. Kushman, and K. Hofmann, "A Deep Learning Approach for Joint Video Frame and Reward Prediction in Atari Games," in *ICML 2017 Workshop on Principled Approaches to Deep Learning*, Nov. 2016.

[5] Y. Bengio, A. Courville, and P. Vincent, "Representation Learning: A Review and New Perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[6] X. Wang and A. Gupta, "Unsupervised Learning of Visual Representations Using Videos," in *ICCV*, 2015, pp. 2794–2802.

[7] Z. Chang, X. Zhang, S. Wang, S. Ma, Y. Ye, X. Xiang, and W. Gao, "MAU: A Motion-Aware Unit for Video Prediction and Beyond," in *NeurIPS*, May 2021.

[8] S. Lee, H. G. Kim, D. H. Choi, H.-I. Kim, and Y. M. Ro, "Video Prediction Recalling Long-Term Motion Context via Memory Alignment Learning," in *CVPR*, 2021.

[9] T. Meinhardt, A. Kirillov, L. Leal-Taixe, and C. Feichtenhofer, "TrackFormer: Multi-Object Tracking with Transformers," in *arXiv:2101.02702 [cs]*, Jan. 2021, arXiv: 2101.02702.

[10] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *ICLR*, 2021.

[11] P. Esser, R. Rombach, and B. Ommer, "Taming Transformers for High-Resolution Image Synthesis," in *arXiv:2012.09841 [cs]*, Feb. 2021, arXiv: 2012.09841.

[12] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid, "ViViT: A video vision transformer," in *ICCV*, 2021.

[13] Z. Liu, S. Luo, W. Li, J. Lu, Y. Wu, C. Li, and L. Yang, "ConvTransformer: A Convolutional Transformer Network for Video Frame Synthesis," in *arXiv:2011.10185 [cs]*, Nov. 2020, arXiv: 2011.10185. [Online]. Available: http://arxiv.org/abs/2011.10185

[14] W. Yan, Y. Zhang, P. Abbeel, and A. Srinivas, "VideoGPT: Video Generation using VQ-VAE and Transformers," in *arXiv:2104.10157 [cs]*, Sep. 2021, arXiv: 2104.10157.

[15] C. Wu, J. Liang, L. Ji, F. Yang, Y. Fang, D. Jiang, and N. Duan, "N\"UWA: Visual Synthesis Pre-training for Neural visUal World creAtion," *arXiv:2111.12417 [cs]*, Nov. 2021, arXiv: 2111.12417. [Online]. Available: http://arxiv.org/abs/2111.12417

[16] M. Chaabane, A. Trabelsi, N. Blanchard, and R. Beveridge, "Looking Ahead: Anticipating Pedestrians Crossing with Future Frames Prediction," in *WACV*, 2020, pp. 2286–2295.

[17] B. Jin, Y. Hu, Q. Tang, J. Niu, Z. Shi, Y. Han, and X. Li, "Exploring Spatial-Temporal Multi-Frequency Analysis for High-Fidelity and Temporal-Consistency Video Prediction," in *CVPR*, 2020, pp. 4554–4563.

[18] Y. Wang, J. Wu, M. Long, and J. B. Tenenbaum, "Probabilistic Video Prediction From Noisy Data With a Posterior Confidence," in *CVPR*, Jun. 2020.

[19] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," in *ICLR*, 2016.

[20] B. Chen, W. Wang, and J. Wang, "Video Imagination from a Single Image with Transformation Generation," in *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*, ser. Thematic Workshops '17. New York, NY, USA: Association for Computing Machinery, Oct. 2017, pp. 358–366. [Online]. Available: https://doi.org/10.1145/3126686.3126737

[21] C. Vondrick, H. Pirsiavash, and A. Torralba, "Generating videos with scene dynamics," in *NIPS*, 2016.

[22] Y. Wu, R. Gao, J. Park, and Q. Chen, "Future Video Synthesis With Object Motion Prediction," in *CVPR*, 2020.

[23] S. Oprea, P. Martinez-Gonzalez, A. Garcia-Garcia, J. A. Castro-Vargas, S. Orts-Escolano, J. Garcia-Rodriguez, and A. Argyros, "A review on deep learning techniques for video prediction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 8, 2020.

[24] M. Babaeizadeh, C. Finn, D. Erhan, R. Campbell, and S. Levine, "Stochastic variational video prediction," in *ICLR*, 2018.

[25] E. Denton and R. Fergus, "Stochastic Video Generation with a Learned Prior," in *International Conference on Machine Learning*. PMLR, Jul. 2018, pp. 1174–1183, iSSN: 2640-3498.

[26] L. Castrejon, N. Ballas, and A. Courville, "Improved Conditional VRNNs for Video Prediction," in *ICCV*, Oct. 2019, pp. 7607–7616, iSSN: 2380-7504.

[27] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018.

[28] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[29] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions," in *arXiv:2102.12122 [cs]*, Feb. 2021, arXiv: 2102.12122. [Online]. Available: http://arxiv.org/abs/2102.12122

[30] H. Fan, B. Xiong, K. Mangalam, Y. Li, Z. Yan, J. Malik, and C. Feichtenhofer, "Multiscale vision transformers," in *ICCV*, 2021, pp. 6824–6835.

[31] P. Zhang, X. Dai, J. Yang, B. Xiao, L. Yuan, L. Zhang, and J. Gao, "Multi-Scale Vision Longformer: A New Vision Transformer for High-Resolution Image Encoding," in *ICCV*, 2021.

[32] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *ICCV*, 2021.

[33] Y. Yuan, R. Fu, L. Huang, W. Lin, C. Zhang, X. Chen, and J. Wang, "HRFormer: High-resolution transformer for dense prediction," in *NeurIPS*, 2021.

[34] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, "CCNet: Criss-Cross Attention for Semantic Segmentation," 2019, pp. 603–612.

[35] H. Wang, Y. Zhu, B. Green, H. Adam, A. Yuille, and L.-C. Chen, "Axial-DeepLab: Stand-Alone Axial-Attention for Panoptic Segmentation," in *Computer Vision – ECCV 2020*, ser. Lecture Notes in Computer Science, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Cham: Springer International Publishing, 2020, pp. 108–126.

[36] P. Isola, J. Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *CVPR*, 2017.

[37] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-End Object Detection with Transformers," in *ECCV*. Springer International Publishing, 2020, pp. 213–229.

[38] P. Shaw, J. Uszkoreit, and A. Vaswani, "Self-attention with relative position representations," in *NAACL*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 464–468.

[39] Y. Wang, F. Tian, D. He, T. Qin, C. Zhai, and T.-Y. Liu, "Non-autoregressive machine translation with auxiliary regularization," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, 2019, pp. 5377–5384, number: 01.

[40] A. Andonian, T. Park, B. Russell, P. Isola, J.-Y. Zhu, and R. Zhang, "Contrastive feature loss for image prediction," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 1934–1943.

[41] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: a local SVM approach," in *ICPR*, vol. 3, Aug. 2004, pp. 32–36 Vol.3, iSSN: 1051-4651.

[42] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised Learning of Video Representations using LSTMs." PMLR, Jun. 2015, pp. 843–852.

[43] F. Ebert, C. Finn, A. X. Lee, and S. Levine, "Self-supervised visual planning with temporal skip connections," in *1st annual conference on robot learning, CoRL 2017, mountain view, california, USA, november 13-15, 2017, proceedings*, ser. Proceedings of machine learning research, vol. 78. PMLR, 2017, pp. 344–356. [Online]. Available: http://proceedings.mlr.press/v78/frederik-ebert17a.html

[44] R. Villegas, J. Yang, S. Hong, X. Lin, and H. Lee, "Decomposing motion and content for natural video sequence prediction," in *ICLR*, 2017.

[45] Y. Wang, L. Jiang, M.-H. Yang, L.-J. Li, M. Long, and L. Fei-Fei, "Eidetic 3D LSTM: A Model for Video Prediction and Beyond," in *ICLR*, Sep. 2018.

[46] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *CVPR*, 2018, pp. 586–595.

[47] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015, pp. 1–15.
[48] I. Loshchilov and F. Hutter, "Decoupled Weight Decay Regularization," Sep. 2018.
[49] Y. Wang, Z. Gao, M. Long, J. Wang, and P. S. Yu, "PredRNN++: Towards A Resolution of the Deep-in-Time Dilemma in Spatiotemporal Predictive Learning." PMLR, Jul. 2018, pp. 5123–5132.
[50] J. Su, W. Byeon, J. Kossaifi, F. Huang, J. Kautz, and A. Anandkumar, "Convolutional Tensor-Train LSTM for Spatio-temporal Learning," in *NeurIPS*, 2020.