Video Prediction Recalling Long-term Motion Context via Memory Alignment Learning

Sangmin Lee1 Hak Gu Kim2 Dae Hwi Choi1 Hyung-II Kim1,3 Yong Man Ro1∗
1 Image and Video Systems Lab, KAIST, South Korea
2 EPFL, Switzerland
3 ETRI, South Korea
{sangmin.lee,sinjh1796,ymro}@kaist.ac.kr hakgu.kim@epfl.ch hikim@etri.re.kr

Abstract

Our work addresses long-term motion context issues for predicting future frames. To predict the future precisely, it is required to capture which long-term motion context (e.g., walking or running) the input motion (e.g., leg movement) belongs to. The bottlenecks arising when dealing with the long-term motion context are: (i) how to predict the long-term motion context naturally matching input sequences with limited dynamics, (ii) how to predict the long-term motion context with high-dimensionality (e.g., complex motion). To address the issues, we propose novel motion context-aware video prediction. To solve the bottleneck (i), we introduce a long-term motion context memory (LMC-Memory) with memory alignment learning. The proposed memory alignment learning enables to store long-term motion contexts into the memory and to match them with sequences including limited dynamics. As a result, the long-term context can be recalled from the limited input sequence. In addition, to resolve the bottleneck (ii), we propose memory query decomposition to store local motion context (i.e., low-dimensional dynamics) and recall the suitable local context for each local part of the input individually. It enables to boost the alignment effects of the memory. Experimental results show that the proposed method outperforms other sophisticated RNN-based methods, especially in long-term condition. Further, we validate the effectiveness of the proposed network designs by conducting ablation studies and memory feature analysis. The source code of this work is available†.

1. Introduction

Video prediction in computer vision is to estimate upcoming future frames at pixel-level from given previous frames. Since predicting the future is an important base-

∗Corresponding author
†https://github.com/sangmin-git/LMC-Memory
term motion context the input motion belongs to. For example, in order to predict the future of leg movement, we need to know such partial leg movement belongs to either walking or running (i.e., long-term motion context). The bottlenecks arising when dealing with long-term motion context are as follows: (i) how to predict the long-term motion context naturally matching input sequences with limited dynamics, (ii) how to predict the long-term motion context with high-dimensionality.

In this paper, we propose novel motion context-aware video prediction to address the aforementioned issues. To solve the bottleneck (i), we introduce a long-term motion context memory (LMC-Memory) with memory alignment learning. Contrary to the internal memory cells of the RNNs, the LMC-Memory externally exists with its own parameters to preserve various long-term motion contexts of training data, which are not limited to the current input. Memory alignment learning is proposed to effectively store long-term motion contexts into the LMC-Memory and recall them even with inputs having limited dynamics. Memory alignment learning contains two training phases to align long-term and short-term motions: (Phase 1) storing long-term motion context from long-term sequences into the memory, (Phase 2) matching input short-term sequences with the stored long-term motion contexts in the memory. As a result, the long-term motion context (e.g., long-term walking dynamics) can be recalled from the input short-term sequence alone (e.g., short-term walking clip).

Furthermore, to resolve the bottleneck (ii), we propose decomposition of a memory query that is used to store and recall the motion context. Even if various motion contexts of training data are stored in the LMC-Memory, it is difficult to capture the motion context that is exactly matched with the input. This is because motions of video sequences have high-dimensionality (e.g., complex motion with local motion components). The dimensionality indicates the number of pixels in a video sequence. Since each motion is slightly different from one another in a global manner even for the same category, the proposed memory query decomposition is useful in that it enables to store local context (i.e., low-dimensional dynamics) and recall the suitable local context for each local part of the input individually. It can boost the alignment effects between the input and the stored long-term motion context in the LMC-Memory.

The major contributions of the paper are as follows.

- We introduce novel motion context-aware video prediction to solve the inherent problem of the RNN-based methods in capturing long-term motion context. We address the arising long-term motion context issues in the video prediction.
- We propose the LMC-Memory with memory alignment learning to address storing and recalling long-term motion contexts. Through the learning, it is possible to recall long-term motion context corresponding to an input sequence even with limited dynamics.
- To address the high-dimensionality of motions, we decompose memory query to separate an overall motion into local motions with low-dimensional dynamics. It makes it possible to recall suitable local motion context for each local part of the input individually.

2. Related Work

2.1. Video Prediction

In video prediction, errors for predicting future frames can be divided into two factors [36]. The first one is about systematic errors due to the lack of modeling capacity for deterministic changes. The second one is related to modeling the intrinsic uncertainty of the future. There have been several works to address the second factor [1, 6, 19, 42]. These methods utilized stochastic modeling to generate plausible multiple futures. Contrary to these, our paper addresses the video prediction focusing on the first factor.

Recently, deep learning-based video prediction methods have been proposed to deal with the first factor. They considered the problems leading to prediction difficulty such as capturing high-dimensionality and long-term dynamics in video data [2, 7, 8, 13, 15, 16, 21, 24, 29, 31–37, 39, 41, 43, 44]. Finn et al. [7] incorporated appearance information in the previous frames with the predicted pixel motion information for long-range video prediction. Villegas et al. [34] introduced a hierarchical prediction model that generates the future image from the predicted high-level structure. A predictive recurrent neural network (PredRNN) model was presented by Wang et al. [37] to model and memorize both spatial and temporal representations simultaneously. Wang et al. [35] further extended this model, named PredRNN++ to solve the vanishing gradient problem in deep-in-time prediction by building adaptive learning between long-term and short-term frame relation. Recently, eidetic 3D LSTM (E3D-LSTM) [36] was proposed to integrate 3D convolutions into the RNNs for effectively addressing memories across long-term periods. Jin et al. [15] introduced spatial-temporal multi-frequency analysis for high-fidelity video prediction with temporal-consistency. Su et al. [29] proposed convolutional tensor-train decomposition to learn long-term spatio-temporal correlations. However, these works still have a limitation in encoding long-term dynamics in that they mainly rely on the input sequence to find frame relations. Therefore, it is difficult to capture the long-term motion context for predicting the future from the input sequence with limited dynamics.

2.2. Memory Network

Memory augmented networks have recently been introduced for solving various problems in computer vision tasks

3055
Motion Matching Feature \( E_{\text{spa}} \) \( \{f_{i_{\text{spa}}} \}_{i=t-n+1}^{t} \) Motion Matching Feature \( Z^{\text{MM}} \) LMC-Memory Future Sequence \( S_{t+1:t+K}^{\text{mem}} = \{I_{i}^{t+1:t+K} \} \) Input Sequence \( S_{t-n+1:t} = \{I_{i}^{t-n+1:t} \} \) Difference Frames \( I_{t-n+2} - I_{t-n+1} \) \( I_{t-n+1} - I_{t-n} \) Recall In \( Z^{\text{MM}} \) Recall Out \( Z^{\text{MM}} \) Spatial Encoder \( \mathcal{G}_{\text{LSTM}} \) Frame Decoder \( \mathcal{D} \) Figure 1: Overall framework with the proposed LMC-memory for video prediction at testing phase. The lower path is for recalling long-term motion context from the external memory, named LMC-Memory. The upper path is for predicting future frames with recurrent manner considering the recalled long-term motion context.

First, in the lower path of Figure 1, the differences between the consecutive frames (i.e., difference frames) are used as inputs of motion matching encoder \( E_{\text{MM}} \). Then, a motion matching feature \( Z^{\text{MM}} \) is extracted to recall the motion context feature \( F^{\text{mem}} \) from the external memory, named LMC-Memory. This LMC-Memory contains various long-term motion contexts of training data. Thus, \( F^{\text{mem}} \) from the memory can be considered as long-term information corresponding to the input sequence \( S_{t-n+1:t} \) (described in Section 3.2 in detail). It is then embedded in the upper part \( P \). This long-term motion context embedding contains 2D-DeConvs to match the spatial size with the upper part, which results in \( F^{\text{mem}} \).

The upper part of Figure 1 demonstrates long-term motion context-aware video prediction \( P \) scheme. In this path, the required motion context is refined through attention-based encoding to effectively embed it in predicting future frames. Each frame of the input sequence is independently fed to spatial encoder \( E_{\text{sp}} \) with 2D-Convs to extract appearance characteristics. The ConvLSTM [40] receives each extracted spatial feature \( f_{i}^{sp} = E_{\text{sp}}(I_{i}) \) as inputs in time step order. A cell state \( C_{t} \) and an output state \( H_{t} \) are obtained from recurrent processing of the ConvLSTM. Since \( C_{t} \) contains the information from the past to the present of the input sequence, we use \( C_{t} \) to refine \( F^{\text{mem}} \) for embedding the required motion context at the current step. \( C_{t} \) and \( F^{\text{mem}} \) are concatenated and pass through fully connected layers to make channel-wise attention \( A^{\text{mem}}_{t} \) for \( F^{\text{mem}} \). The channel-wise refined feature \( F^{\text{mem}} = A^{\text{mem}}_{t} \otimes F^{\text{mem}} \) and output state \( H_{t} \) from the ConvLSTM are concatenated to embed long-term context to the ConvLSTM output (i.e., spatio-temporal information of the input). The concatenated feature is fed to a frame decoder.
$D$ with 2D-DeConvs to generate corresponding next frame $\hat{I}_{t+1} = D([H_t; f^\text{mem}_t])$. The embedded motion context memory feature can provide the prior of long-term motion context for the current input sequence. Note that the generated next frame $\hat{I}_{t+1}$ enters $P$ as a new input to create the further future frame.

### 3.2. LMC-Memory with Alignment Learning

The long-term motion context memory, named LMC-Memory, is to provide the long-term motion context for current input sequences to predict future frames. To effectively recall the long-term motion context even for the input sequence with limited dynamics, we propose novel memory alignment learning. Figure 2 shows the training scheme of the LMC-Memory. The memory is trained alternately with two phases: (1) storing long-term motion context, (2) matching limited short-term sequence with the long-term context.

During the storing phase (Phase 1), we take a long-term sequence $S^\text{long}_n$ with length $N$ from the training data. After obtaining the difference frames, the motion context of the long-term sequence is extracted by a long-term motion context encoder $E_{\text{LMC}}$. We adopt typical motion extractor, C3D [30] with 3D-Convs for $E_{\text{LMC}}$. The resulted long-term motion context feature $Z^\text{LMC} = \{z^\text{LMC}_l\}_{l=1}^{w_h} \in \mathbb{R}^{w_h \times \text{c}}$ is divided into local parts to exploit decomposed dynamics. The local feature $z^\text{LMC}_l \in \mathbb{R}^c$ is used as a memory query individually.

The parameters of the LMC-Memory have a matrix form, $M = \{m_i\}_{i=1}^s \in \mathbb{R}^{s \times \text{c}}$ with $s$ slot size and $c$ channels. A row vector $m_i \in \mathbb{R}^c$ denotes a memory item of $M$. An addressing vector $a_t = \{a_{i,j}\}_{i=1}^s \in \mathbb{R}^s$ for query $z^\text{LMC}_l$ is used to address the location of the memory $M$. Each scalar value $a_{ij}$ of $a_t$ can be considered as an attention weight for the corresponding memory slot $m_i$. Memory addressing procedure can be formulated as

$$a_{ij} = \frac{\exp(d(z^\text{LMC}_l, m_i))}{\sum_{j=1}^s \exp(d(z^\text{LMC}_l, m_j))},$$

where $d(\cdot, \cdot)$ indicates cosine similarity function and $\exp(\cdot)/\sum \exp(\cdot)$ denotes softmax function. With $M$ and $a_t = \{a_{i,j}\}_{i=1}^s$, the memory outputs a local motion context memory feature $f^\text{mem}_l \in \mathbb{R}^c (l = 1, 2, ..., w \times h)$ for each location $l$ as follows

$$f^\text{mem}_l = \sum_{i=1}^s a_{ij} m_i.$$  

Finally, a motion context memory feature $F^\text{mem} = \{f^\text{mem}_l\}_{l=1}^{w_h} \in \mathbb{R}^{w_h \times \text{c}}$ is obtained by positioning each local feature $f^\text{mem}_l$ as shown in Figure 2. As addressed in Section 3.1, $F^\text{mem}$ is embedded to motion context-aware video prediction $P$. During the training phase 1, the weights of the memory $M$ are updated through backpropagation as [9]. We train the networks to generate long-term future from long-term input so that long-term motion context can be stored in the memory at this phase.

At the matching phase (Phase 2), the model receives a short-term sequence $S^\text{short}_n$ with length $n$ (long-term length $N >$ short-term length $n$). The matching phase allows long-term information in the memory to be recalled by a limited short-term sequence. Similar to the long-term encoding process, the difference frames are utilized for motion encoding. Then, motion matching feature $Z^\text{MM}$ is extracted by a motion matching encoder $E_{\text{MM}}$. Same as
Algorithm 1 Memory Alignment Learning

1: Inputs: short-term sequence $S_{n}^{short} = S_{t-n+1:t}$, long-term sequence $S_{n}^{long} = S_{t-n+1:t–r–n+r+N}$ (random integer $r \sim U\{0, n\}$), and learning rate $\alpha$.
2: Initialize parameters of PHASE 1 networks ($\theta$), PHASE 2 networks ($\phi$), and LMC-memory ($M$).
3: for each iteration do
4:    (PHASE 1: Storing Phase)
5:    Get $Z_{LMC} = E_{LMC}(S_{n}^{long})$
6:    Get $F_{mem} = LMC-Memory(Z_{LMC})$
7:    for $i = 0, 1, ..., K-1$ do
8:        Get $I_{t+1} = P(S_{t-n+1:t}, S_{t+1:t+i}, F_{mem})$
9:    end for
10:   $L \leftarrow \mathcal{L}_{pred}(\hat{S}_{t+1:t+K}, S_{t+1:t+K})$
11:   Update $\theta \leftarrow \theta - \alpha \nabla_{\theta} L$
12: (PHASE 2: Matching Phase)
13:   Get $Z_{MM} = E_{MM}(S_{n}^{short})$
14:   Get $F_{mem} = LMC-Memory(Z_{MM})$
15:   for $i = 0, 1, ..., K-1$ do
16:       Get $I_{t+1} = P(S_{t-n+1:t}, S_{t+1:t+i}, F_{mem})$
17:   end for
18:   $L \leftarrow \mathcal{L}_{pred}(\hat{S}_{t+1:t+K}, S_{t+1:t+K})$
19:   Update $\phi \leftarrow \phi - \alpha \nabla_{\phi} L$
20: end for

$E_{LMC}$, $E_{MM}$ has the C3D [30] structure, but does not share parameters with $E_{LMC}$. The local feature $z_{i}^{MM}$ of $Z_{MM} = \{z_{i}^{MM}\}_{i=1}^{w\times h \times c}$ is used to recall the corresponding long-term motion context from the memory. The memory addressing procedures are the same as the first storing phase (Eq. 1 and 2). However, unlike the phase 1, the weights of the memory $M$ are not optimized and only used to recall the motion context during this matching phase. This is to preserve the stored long-term motion context in $M$. Except for the memory $M$, overall network weights are trained to predict the long-term future frames with the memory feature. Thus, it enables $E_{MM}$ to extract $Z_{MM}$ that properly recalls the corresponding long-term motion context in the given LMC-memory. Optimization is performed with the prediction framework $P$ (see Figure 2). Only the short-term sequence $S_{n}^{short} = S_{t-n+1:t}$ is fed as a main input of $P$. The memory path receives the long-term $S_{n}^{long}$ and short-term $S_{n}^{short}$ alternately. Two training phases are alternately performed in each iteration. In both phases, according to [29,35–37], we exploit a prediction loss function $\mathcal{L}_{pred}$ as follows

$$\mathcal{L}_{pred} = \|\hat{S}_{t+1:t+K} - S_{t+1:t+K}\|_{2}^{2} + \|\hat{S}_{t+1:t+K} - S_{t+1:t+K}\|_{1},$$

where $\hat{S}_{t+1:t+K}$ denotes $K$ predicted future frames while $S_{t+1:t+K}$ denotes $K$ ground truth future frames. Note that the proposed method only takes short-term sequences at inference time as shown in Figure 1. Training procedure is further described in Algorithm 1.

4. Experiments

4.1. Datasets

To validate the proposed method, we utilize both synthetic and natural video datasets. We use a synthetic Moving-MNIST dataset [28] that is mainly used in video prediction. In addition, we use a KTH Action [26] and a Human 3.6M [12] datasets including natural videos with human action scenarios.

Moving-MNIST. The Moving-MNIST [28] contains the moving of two randomly sampled digits from the original MNIST dataset. Each digit moves in a random direction within a 64×64 size image with a gray scale. The constructed Moving-MNIST dataset consists of 10,000 sequences for training and 5,000 sequences for testing as [35].

KTH Action. KTH Action dataset [26] consists of 6 types of action videos for 25 subjects. It includes indoor, outdoor, scale variations, and different clothes. Each frame is resized to 128×128 with a gray scale. The videos of 1-16 subjects are used as the training set while the videos of 17-25 subjects are used as the test set. We follow the experimental setting [33] of video prediction for the KTH Action dataset.

Human 3.6M. The Human 3.6M [12] includes 17 human action scenarios with total 11 actors. It contains 4 different camera views. Each frame is resized to 64×64 with RGB color channels. Videos of subjects 1, 5, 6, 7, and 8 are used to train the model while videos of subjects 9 and 11 are used to test the model. We follow the experimental setting [34].

4.2. Implementation

The video frames are normalized to intensity of $[0, 1]$ and resized to 64 × 64 (MNIST and Human 3.6M) or 128 × 128 (KTH) as [33–35]. The proposed model is trained by Adam optimizer [17] with a learning rate of 0.0002. Memory slot size $s$ is fixed as 100 for all experiments. Input short-term sequence length $n$ is set as 10. Long-term sequence length $N$ is set as 30 (MNIST) or 40 (KTH and Human 3.6M). Our model is trained to predict corresponding $N$ future frames. We use 4-layer ConvLSTMs for frame prediction. The overall detailed network structures are described in the supplementary material.

4.3. Evaluation

We use MSE, PSNR, SSIM [38], and LPIPS [45] to measure the performances. MSE and PSNR are calculated by the pixel-wise difference between the actual frame and the predicted frame. We also evaluate the performance using SSIM that considers the structural similarity between
<table>
<thead>
<tr>
<th>Prediction Method</th>
<th>Performance (10 → 10)</th>
<th>Performance (10 → 30)</th>
<th>Computational Cost (10 → 30)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MSE</td>
<td>SSIM</td>
<td>LPIPS</td>
</tr>
<tr>
<td>TRAGRU [27]</td>
<td>106.9</td>
<td>0.713</td>
<td>-</td>
</tr>
<tr>
<td>CDNA [7]</td>
<td>97.4</td>
<td>0.721</td>
<td>-</td>
</tr>
<tr>
<td>VPN [15]</td>
<td>64.1</td>
<td>0.870</td>
<td>-</td>
</tr>
<tr>
<td>PredRNN [37]</td>
<td>56.8</td>
<td>0.867</td>
<td>-</td>
</tr>
<tr>
<td>PredRNN++ [35]</td>
<td>42.1</td>
<td>0.913</td>
<td>59.5</td>
</tr>
<tr>
<td>E3D-LSTM [36]</td>
<td>50.9</td>
<td>0.912</td>
<td>86.7</td>
</tr>
<tr>
<td>Conv-TT-LSTM [29]</td>
<td>53.0</td>
<td>0.915</td>
<td>40.5</td>
</tr>
<tr>
<td>Proposed Method</td>
<td>41.5</td>
<td>0.924</td>
<td>46.9</td>
</tr>
</tbody>
</table>

Table 1: Results on the Moving-MNIST. Higher SSIM values are better while lower MSE and LPIPS values are better. Red and Blue indicate the best and the second best, respectively. Ours outperforms the others especially in long-term condition.

Figure 3: Qualitative results with given 10 frames on the Moving-MNIST. The other results are obtained from official sources.

Compared to other complex RNN-based methods, we adopt simple ConvLSTMs. Further, memory feature $F_{mem}$ is extracted only once at the beginning, which is advantageous in computational cost. Figure 3 shows examples of frames predicted by the proposed method and other video prediction methods. As shown in the figure, the predicted frames of the proposed method show convincingly similar results to the ground truth. However, the prediction results by other methods show that they lose the trajectories or the shape of digits, especially in long-term condition.

Results on Moving-MNIST. Table 1 shows the performance comparisons with the state-of-the-art methods on the Moving-MNIST. The left part of the table shows the experimental results of 10 frames prediction with the input 10 frames. The right part of the table shows the experimental results for 30 frames prediction with 10 input frames. The proposed method outperforms the other state-of-the-art methods. In particular, our method far surpasses the others in predicting 30 frames in terms of the LPIPS metric. In addition, the proposed method shows much better results on the computational cost compared to the other methods. Compared to other complex RNN-based methods, we adopt simple ConvLSTMs. Further, memory feature $F_{mem}$ is extracted only once at the beginning, which is advantageous...
<table>
<thead>
<tr>
<th>Prediction Method</th>
<th>Performance (10 → 20)</th>
<th>Performance (10 → 40)</th>
<th>Computational Cost (10 → 40)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR</td>
<td>SSIM</td>
<td>LPIPS</td>
</tr>
<tr>
<td>MCNET [33]</td>
<td>25.95</td>
<td>0.804</td>
<td>-</td>
</tr>
<tr>
<td>FRNN [22]</td>
<td>26.12</td>
<td>0.771</td>
<td>-</td>
</tr>
<tr>
<td>PredRNN [37]</td>
<td>27.55</td>
<td>0.839</td>
<td>-</td>
</tr>
<tr>
<td>PredRNN++ [35]</td>
<td>28.62</td>
<td>0.888</td>
<td>228.9</td>
</tr>
<tr>
<td>E3D-LSTM [36]</td>
<td>27.92</td>
<td>0.893</td>
<td>298.4</td>
</tr>
<tr>
<td>Conv-TT-LSTM [29]</td>
<td>28.36</td>
<td>0.907</td>
<td>133.4</td>
</tr>
<tr>
<td>Proposed Method</td>
<td>28.61</td>
<td>0.894</td>
<td>133.3</td>
</tr>
</tbody>
</table>

Table 2: Results on the KTH. Higher values are better for PSNR and SSIM while lower values are better for LPIPS. Red and Blue indicate the best and the second best, respectively. Ours outperforms the others especially in long-term condition.

![Figure 4: Qualitative results with given 10 frames on the KTH Action. The other results are obtained from official sources.](image)

<table>
<thead>
<tr>
<th>Prediction Method</th>
<th>Performance (10 → 40)</th>
<th>Performance (10 → 40, Last 10)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR</td>
<td>SSIM</td>
</tr>
<tr>
<td>PredRNN++ [35]</td>
<td>23.23</td>
<td>0.876</td>
</tr>
<tr>
<td>E3D-LSTM [36]</td>
<td>22.33</td>
<td>0.850</td>
</tr>
<tr>
<td>Proposed Method</td>
<td>24.97</td>
<td>0.919</td>
</tr>
</tbody>
</table>

Table 3: Results on the Human 3.6M. Higher PSNR and SSIM are better while lower LPIPS is better.

![Figure 5: Qualitative results with given 10 frames on the Human 3.6M. Results of the others are from official sources.](image)

actually starts from the middle. As shown in the figure, the other methods fail to capture the detailed leg movement, especially in long-term condition. Whereas, our predicted frames are very similar to the ground truth frames. The proposed method maintains a clear shape of the legs while following the long-term trajectories even in such a challenging condition (i.e., limited dynamics).

**Results on Human 3.6M.** Table 3 shows the performance comparisons with the other methods on the Human 3.6M. The left part of the table shows the experimental results for predicting 40 frames with given 10 input frames. The right part of the table indicates performances for the last 10 frames among the future 40 frames. The proposed method outperforms other state-of-the-art video prediction methods both in predicting 40 future frames and predicting the last 10 frames. Figure 5 shows qualitative results for the long-term prediction on the Human 3.6M. The proposed method captures direction changing in the long-term while the other methods show the disappearance of a person at the corner. Compared to the others, the proposed method properly captures the long-term motion context with redirection.

3060
Table 4: Effects of the network designs on the performance and the computational cost. Performance evaluations are conducted on KTH Action dataset.

<table>
<thead>
<tr>
<th>Prediction Method</th>
<th>Performance</th>
<th>Computational Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR</td>
<td>SSIM</td>
</tr>
<tr>
<td>Model w/o LMC-Memory</td>
<td>25.29</td>
<td>0.851</td>
</tr>
<tr>
<td>Model w/ LMC-Memory (Non-local Motion Context)</td>
<td>25.57</td>
<td>0.854</td>
</tr>
<tr>
<td>Model w/ LMC-Memory (Local Motion Context)</td>
<td>26.21</td>
<td>0.862</td>
</tr>
</tbody>
</table>

4.4. Ablation Study

We analyze the effects of network designs by ablating them as shown in Table 4. In detail, we investigate the effectiveness of the LMC-Memory (i.e., memory alignment learning) and the local motion context (i.e., memory query decomposition). The baseline, 'Model w/o LMC-Memory' consists of the spatial encoder, the ConvLSTMs, and the frame decoder. The second one, 'Model w/ LMC-Memory (Non-local motion context)' contains LMC-Memory but it does not adopt memory query decomposition to use local motion context as memory queries. This model uses a globally pooled motion context feature as a query. The last one indicates our final proposed model of this paper. As shown in the table, each component contributes to the performances in predicting the last 10 among 40 frames. The final model outperforms the other models, especially in terms of perceptual metric LPIPS. These results show that locally manipulated query boosts the effects of the memory since it is more accessible to store and recall the motion context with low-dimensional dynamics. Further, the additional computational cost to use the LMC-Memory is marginal.

Figure 6 shows qualitative results for different network designs on the KTH Action dataset.

4.5. Memory Addressing

We analyze the memory addressing for different sequences. Figure 7 shows the cosine similarity values between addressing vectors from long-term and short-term sequences including different subjects. The areas addressed in the memory are more comparable (similarity between addressing vectors is high) in the case of the same action scenario than in the case of the different actions. It shows that the long-term and short-term features that belongs to the similar action are convincingly aligned in the memory.

5. Conclusion

The objective of the proposed work is to predict future frames being aware of the long-term motion context. To this end, we propose the LMC-Memory with the alignment learning scheme to effectively store abundant long-term contexts of training data and recall suitable motion context even from limited inputs. In addition, we utilize memory query decomposition to separate overall motion into low-dimensional dynamics. It enables to cope with the high-dimensionality in terms of utilizing motion contexts in the memory. As a result, the proposed method outperforms the state-of-the-art methods with sophisticated RNNs. In particular, it is significantly noticeable in long-term condition. Further, the effectiveness of the proposed method is analyzed in both quantitative and qualitative ways.

Acknowledgement. This work was partly supported by the IITP grant (No. 2020-0-00004) and BK 21 Plus project.
References


