PredCNN: Predictive Learning with Cascade Convolutions

Ziru Xu[†], Yunbo Wang[†], Mingsheng Long[*], and Jianmin Wang
KLiss, MOE; School of Software, Tsinghua University, China
National Engineering Laboratory for Big Data Software
Beijing Key Laboratory for Industrial Big Data System and Application
{xzr16,wangyb15}@mails.tsinghua.edu.cn {mingsheng,jimwang}@tsinghua.edu.cn

Abstract

Predicting future frames in videos remains an unsolved but challenging problem. Mainstream recurrent models suffer from huge memory usage and computation cost, while convolutional models are unable to effectively capture the temporal dependencies between consecutive video frames. To tackle this problem, we introduce a entirely CNN-based architecture, PredCNN, that models the dependencies between the next frame and the sequential video inputs. Inspired by the core idea of recurrent models that previous states have more transition operations than future states, we design a cascade multiplicative unit (CMU) that provides relatively more operations for previous video frames. This newly proposed unit enables PredCNN to predict future spatiotemporal data without any recurrent chain structures, which eases gradient propagation and enables a fully paralleled optimization. We show that PredCNN outperforms the state-of-the-art recurrent models for video prediction on the standard Moving MNIST dataset and two challenging crowd flow prediction datasets, and achieves a faster training speed and lower memory footprint.

1 Introduction

Video prediction has recently become an important topic in spatiotemporal learning, for the reason that it has broad application prospects in weather nowcasting [Wang *et al.*, 2017; Shi *et al.*, 2015], traffic flow prediction [Zhang *et al.*, 2017], air pollution forecasting and so forth. An accurate prediction of future spatiotemporal data depends on whether the prediction system is able to extract and utilize the relevant information between the previous video sequence and future frames. However, finding these relationships among the high dimensional spatiotemporal data is non-trivial due to the complexity and ambiguity inherent in videos. It always needs to deal with object overlapping, shape deformations and scale variations, which make prediction of future video frames a far more challenging task than the traditional time-series autoregression.

Therefore, convolutional neural networks (CNNs) have been introduced to solve the problem of video prediction [Zhang *et al.*, 2017; Mathieu *et al.*, 2016]. Deep CNNs come with great modeling capacity, enabling these models to capture the spatial correlations in consecutive video frames effectively. However, due to the inherent limitations of standard convolutional structures, these CNN-based methods treat all previous frames equally and are unable to model the temporal dynamics existing in a continuous Markov process. A relative simple transformation from the historical data to future frames is learned, while temporal coherency and long-term dependencies are hard to be preserved. Moreover, CNN-based methods create representations for fixed length videos.

Compared with CNN-based methods, deep recurrent video prediction models focus, to a great extent, on modeling temporal dependencies and show a stronger power in making sequence to sequence predictions. However, due to the inherent long-term gradient flow of recurrent neural networks (RNNs) [Rumelhart *et al.*, 1988; Werbos, 1990], these deep recurrent models suffer from two main bottlenecks. The first one is the well-known gradient vanishing problem: though RNNs make sequential outputs, it is hard for them to capture long-term dependencies across distant frames. That is to say, any changes in previous frames at the first few time steps can barely make an influence on the last few outputs. Another weakness is the high computation cost and memory usage caused by hidden states being updated repeatedly over time with full resolution.

In this paper, we propose PredCNN, an entirely convolutional video prediction model, built upon causal convolution structures in WaveNet [Van Den Oord *et al.*, 2016a]. Inspired by the core idea of recurrent models that capture the temporal dependency with a transition between states, we design a cascade multiplicative unit (CMU) that mimics the state-to-state transition by applying additional gate-convolution operations to the former state than the current state. Furthermore, unlike other CNN-based video prediction models, PredCNN creates hierarchical representations over video frames, where nearby input frames interact at lower layers while distant frames interact at higher layers. This hierarchical architecture stacks several convolutional units on top of each other, thus can easily enlarge the temporal receptive field and allow a more flexible length for the inputs. Compared with the chain structures in deep recurrent models, PredCNN provides a shorter path to capture long-term relationships between distant frames, thus

[†]Equal contribution
[*]Corresponding author: M. Long (mingsheng@tsinghua.edu.cn)

alleviates the gradient vanishing problem. The output at any time step does not depend on the computations of all previous activities, allowing parallelization over all frames. We evaluate our model on the standard Moving MNIST dataset and two challenging crowd flow datasets and show that PredCNN can yield a competitive prediction accuracy and efficiency over other state-of-the-art methods.

2 Related Work

Many recurrent neural network models have been designed for predictive learning of spatiotemporal data based on modeling historical frames. [Ranzato *et al.*, 2014] defined a recurrent architecture for spatial and temporal correlation discovery enlightened by language modeling technique [Mikolov *et al.*, 2010]. [Srivastava *et al.*, 2015] applied the LSTM framework for sequence to sequence learning. [Shi *et al.*, 2015] extended the LSTM cell by changing normal state transitions to convolution operations so as to capture high-level visual representations. This Convolutional LSTM (ConvLSTM) model has become a primary structure in this area. [Finn *et al.*, 2016] developed an action-conditioned model that captures pixel motions to predict the next frame. [Lotter *et al.*, 2017] presented a deep predictive coding network (PredNet) where each ConvLSTM layer outputs local predictions and only passes deviations to the following layers. Inspired by two-stream ConvNets [Simonyan and Zisserman, 2014] for video action recognition, [Patraucean *et al.*, 2016] and [Villegas *et al.*, 2017] introduced optical flow into recurrent models to describe motion changes. [Kalchbrenner *et al.*, 2017] proposed a probabilistic model, Video Pixel Network (VPN), that makes predictions by estimating the discrete joint distribution of the raw pixel values in a video and generating frames with PixelCNNs [van den Oord *et al.*, 2016b]. [Wang *et al.*, 2017] put up a PredRNN architecture with spatiotemporal LSTM unit (ST-LSTM) to extract and memorize spatial and temporal representations simultaneously. Limited by the chain structure of RNNs, these recurrent models suffer from both computation burden and parallelization difficulty.

Although these RNN-based models enable video predictions at multiple time frames, in many realistic spatiotemporal forecasting applications, making an accurate prediction of the next frame is far more important than predicting into a distant future. Taking traffic flow prediction as an example, commonly used traffic flow images of an urban area are collected every half an hour [Zhang *et al.*, 2017] and do not change rapidly. Therefore, the state-of-the-art approaches [Zhang *et al.*, 2016; 2017] particularly designed for this next-frame prediction task have attempted ConvNets, deep residual structures [He *et al.*, 2016] and parametric-matrix-based fusion mechanisms [Zheng, 2015], to enhance their modeling capability of the spatial appearance in the near future. However, these methods deal with the temporal cues by simply feeding sequential input frames into different CNN channels, thus, cannot capture the temporal relationships or maintain the motion coherency to some required extent. Different from these CNN architectures, our proposed PredCNN model exploits a novel gated cascade convolutional structure to capture temporal dependencies underlying video frames in a logical way.

3 Preliminaries

3.1 Multiplicative Unit

The multiplicative unit (MU) [Kalchbrenner *et al.*, 2017] is a non-recurrent convolutional structure whose neuron connectivity is quite similar as LSTMs [Hochreiter and Schmidhuber, 1997]. But differently, it takes input $\mathbf{x}$ while discards the hidden representation of all previous states, and thus cannot capture the spatial and temporal representations simultaneously. There are no recurrent connections in the multiplicative unit, and the computation at each time step does not rely on any previous activities. Its key equations are as follows:

$$\begin{aligned}
&\mathbf{g}_1 = \sigma(\mathbf{W}_1 * \mathbf{x} + \mathbf{b}_1) \\
&\mathbf{g}_2 = \sigma(\mathbf{W}_2 * \mathbf{x} + \mathbf{b}_2) \\
&\mathbf{g}_3 = \sigma(\mathbf{W}_3 * \mathbf{x} + \mathbf{b}_3) \\
&\mathbf{u} = \tanh(\mathbf{W}_4 * \mathbf{x} + \mathbf{b}_4) \\
&\mathsf{MU}(\mathbf{x}; \mathbf{W}) = \mathbf{g}_1 \odot \tanh(\mathbf{g}_2 \odot \mathbf{x} + \mathbf{g}_3 \odot \mathbf{u}),
\end{aligned} \tag{1}$$

where σ is the sigmoid activation function, $*$ denotes the convolution operation and $\odot$ is the element-wise multiplication. As an analogy to LSTM unit, $\mathbf{g}_1$, $\mathbf{g}_2$ and $\mathbf{g}_3$ resemble the output gate, the forget gate and the input gate respectively, and $\mathbf{u}$ plays the role of the input modulation gate. $\mathbf{W}_1 \sim \mathbf{W}_4$ and $\mathbf{b}_1 \sim \mathbf{b}_4$ represent the weights and biases of the corresponding convolutional gates, while $\mathbf{W}$ denotes all gate parameters. The multiplicative unit is employed as an effective building block in Video Pixel Networks [Kalchbrenner *et al.*, 2017], which endows this model with powerful modeling capability of spatial representations without recurrent chain structures.

3.2 Residual Multiplicative Block

However, similar to other LSTM-style structures, the MUs are likely to suffer from gradient vanishing when stacked several times in a deep network. Thus, a residual multiplicative block (RMB) has been proposed to ease gradient propagation by [Kalchbrenner *et al.*, 2017]. RMB consists of two MUs that are connected end-to-end, one convolutional layer at the beginning and another at the end, and a residual connection that directly links the input of first convolution layer to the output of last convolution layer. Its equations are as follows:

$$\begin{aligned}
&\mathbf{h}_1 = \mathbf{W}_1 * \mathbf{x} + \mathbf{b}_1 \\
&\mathbf{h}_2 = \mathsf{MU}(\mathbf{h}_1; \mathbf{W}_2) \\
&\mathbf{h}_3 = \mathsf{MU}(\mathbf{h}_2; \mathbf{W}_3) \\
&\mathbf{h}_4 = \mathbf{W}_4 * \mathbf{h}_3 + \mathbf{b}_4 \\
&\mathsf{RMB}(\mathbf{x}; \mathbf{W}) = \mathbf{x} + \mathbf{h}_4,
\end{aligned} \tag{2}$$

where $\mathbf{W}_1$ and $\mathbf{W}_4$ are 1×1 convolutions that transform the channel number of RMB's internal feature maps. To accelerate computation, RMB first squeezes the size of the feature maps before feeding them into the two stacked multiplicative units, then restores the channel number to that of the inputs.

RMB outperforms individual MU by stabling the gradient change rate and alleviating the gradient vanishing problem. Also, it provides more expressive spatial representations by making the gated convolutional structures deep. But similar to MU, RMB also does not rely on the activities at previous time steps. It cannot deal with spatiotemporal motions and cannot be directly applied to spatiotemporal prediction tasks.

4 PredCNN

In this section, we present in details the predictive convolutional neural network (PredCNN), which is entirely built upon gated convolution structures. Initially, this architecture is enlightened by the key idea of modeling both spatial appearances and temporal variations sequentially. To model the temporal dependency in a well-specified way, we design a cascade multiplicative unit (CMU) that predicts future frames based on previous input frames, and guarantees that previous frames must take more operations than future frames. By stacking CMUs hierarchically into PredCNN, we can model the temporal dependencies like the recurrent chain structure.

4.1 Cascade Multiplicative Unit

Sequential modeling in most successful recurrent models has a key ingredient: it *explicitly* models the dependency between different time steps by conditioning the current state on the previous state. However, existing convolution-based units including the multiplicative unit (MU) do not support this key dependency mechanism. In this paper, we design a novel cascade multiplicative unit (CMU) based on gated convolutions to explicitly capture the sequential dependency between the adjacent frames. Our key motivations of CMU are two-folds:

- The current state depends only on previous states but not vice versa. We guarantee this sequentiality by a *cascade* convolution structure similar to the causal convolution in WaveNet [Van Den Oord *et al.*, 2016a], which computes the current state by convolving only on previous inputs.
- Recurrent networks model the temporal dependency by a *transition* from the previous state to the current state. We model the temporal dependency without using state-to-state transition—instead, we compute the hidden representation of the current state directly using the input frames of both previous and current time steps, and force the previous time step to take more gated convolutions.

The diagram of the proposed CMU is shown in Figure 1.

Specifically, we first apply MUs to extract the spatial features of each input frame and then add them element-wisely. Then a gate is used to control the flow of hidden representations to the next layer. As we have mentioned before, MU has LSTM-like structure and can capture a better representation of input frames rather than standard convolutions. The CMU accepts two consecutive inputs $\mathbf{F}_{t-1}^{l}$ and $\mathbf{F}_{t}^{l}$ and generates one output $\mathbf{F}_{t}^{l+1}$, where $\mathbf{F}_{t}^{l}$ denotes the representation at frame t of layer l. Due to a temporal gap, the two consecutive states cannot be added directly. Recurrent models bridge the temporal gap by applying a temporal transition from the previous state to the current state. Instead, we realize this by applying two MUs to the former frame $\mathbf{F}_{t-1}^{l}$ and one MU to the latter frame $\mathbf{F}_{t}^{l}$. By having such LSTM-like cascade structure, we explicitly guarantee sequential ordering and enhance cascade dependency between the two consecutive frames, which may generate more expressive representation $\mathbf{F}_{t}^{l+1}$ for them. The same as recurrent models, we apply weight-sharing to the two MUs in the previous state of CMU to reduce model complexity, which also improves the performance potentially

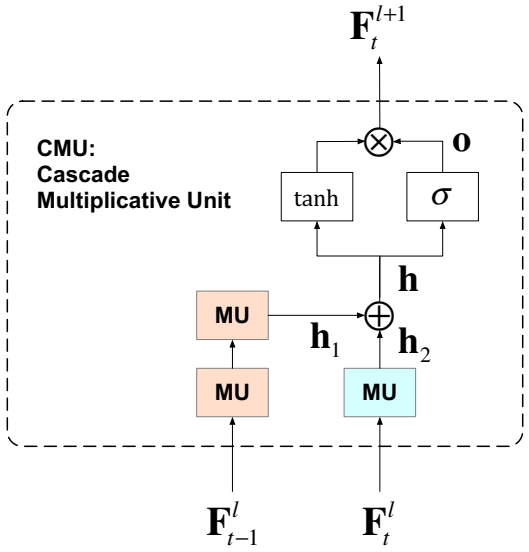

Figure 1: Schematic structure of cascade multiplicative unit (CMU), which accepts two inputs of previous and current states and generates an output for the current state. Colored blocks in the same color represent weight-sharing multiplicative units, while white blocks denote gated structures containing convolutions along with non-linear activation functions, and circles represent element-wise operations.

in the mean time. The key equations of CMU are as follows:

$$\begin{aligned}
\mathbf{h}_1 &= \mathsf{MU}(\mathsf{MU}(\mathbf{F}_{t-1}^{l}; \mathbf{W}_1); \mathbf{W}_1) \\
\mathbf{h}_2 &= \mathsf{MU}(\mathbf{F}_{t}^{l}; \mathbf{W}_2) \\
\mathbf{h} &= \mathbf{h}_1 + \mathbf{h}_2 \\
\mathbf{o} &= \sigma(\mathbf{W}_o * \mathbf{h} + \mathbf{b}_o) \\
\mathbf{F}_{t}^{l+1} &= \mathbf{o} \odot \tanh(\mathbf{W}_h * \mathbf{h} + \mathbf{b}_h),
\end{aligned} \tag{3}$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are parameters of orange MUs in the left branch and blue MU in the right branch respectively, σ is the sigmoid function, $*$ denotes the convolution operation and $\odot$ is the element-wise multiplication. $\mathbf{W}_o$, $\mathbf{W}_h$, $\mathbf{b}_o$ and $\mathbf{b}_h$ are the parameters of the gated convolutions. We adopt $\mathbf{F}_{t}^{l+1} = \mathsf{CMU}(\mathbf{F}_{t-1}^{l}, \mathbf{F}_{t}^{l}; \mathbf{W})$ to simplify the representations of above equations, where $\mathbf{W}$ represents all the learnable parameters.

4.2 PredCNN Architecture

We enable spatiotemporal predictive learning with cascade convolutions by building the predictive convolutional neural network (PredCNN). The overall architecture consists of three parts: **(a) Encoder**, which extracts abstract hidden representation of each frame as input to CMUs; **(b) Hierarchical cascade structure**, which models the spatiotemporal dependencies using the proposed CMUs hierarchically as building blocks; **(c) Decoder**, which reconstructs the hidden representations of CMUs back to the pixel space and generates future frames. A schematic diagram of the PredCNN architecture is illustrated in Figure 2. Note that the encoder and the decoder consist of multiple gated convolutions (RMBs) for modeling spatial appearances.

Encoder and Decoder

We adopt RMB-based encoder and decoder to extract spatial features of input frames and reconstruct the pixel values of output frames. As mentioned before, RMB is a residual structure composed of non-recurrent multiplicative units, making it easy to explore expressive spatial representations and avoid the gradient vanishing trap even with a deep architecture. We stack l_e RMB blocks as the encoder and l_d RMB blocks as the decoder, where l_e and l_d are two hyperparameters. All frames share parameters in the encoder and the decoder.

To acquire a larger receptive field, we apply the dilated convolution [Chen *et al.*, 2016; Yu and Koltun, 2015] to the encoder RMBs as [Kalchbrenner *et al.*, 2017], where the dilation rate is doubled every layer up to the limit 8 and repeated, that is $[1, 2, 4, 8, 1, 2, 4, 8, \cdots]$. For accurate reconstruction, we do not apply the dilated convolutions to the RMB decoder.

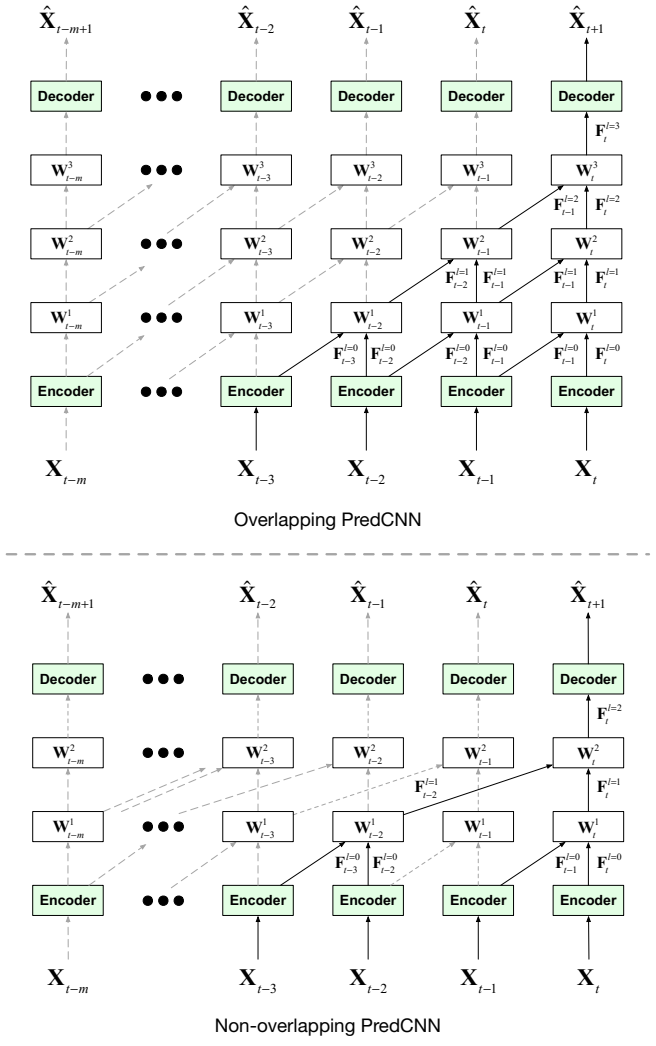

Figure 2: The PredCNN architecture, comprised of encoder and decoder to process spatial appearances, and hierarchical cascade multiplicative units (CMUs) to capture spatiotemporal dependencies. We design two variants of PredCNN: overlapping one with higher accuracy (top) and non-overlapping one with higher efficiency (bottom).

Hierarchical Cascade Structure

Besides the spatial appearances, the temporal dependencies between the next frame and the sequential inputs are the key to spatiotemporal prediction. Motivated by the big success of WaveNet [Van Den Oord *et al.*, 2016a] for generating natural audios, we propose a hierarchical cascade convolution structure using the proposed CMU unit as building blocks, to capture the temporal relationships over high-dimensional video frames. By taking the advantages of hierarchical CMUs, the temporal receptive field is enlarged greatly and the inputs sequence length can be more flexible. When predicting the next frame, only the input frames before the current time step can be used, guaranteeing the temporal ordering constraint.

Specifically, Figure 2 gives an example of using the previous 4 frames $\mathbf{X}_{t-3} \sim \mathbf{X}_t$ to predict the next frame $\widehat{\mathbf{X}}_{t+1}$. We propose two structures of PredCNN for a trade-off of input sequence length and training time. Figure 2 (top) shows an *overlapping* structure, which has 3 layers (other than the encoder and decoder) since the input sequence length is 4. Suppose $\mathbf{F}_t^l$ is the hidden representation of the input frame at time step t of layer l, and the initial representation captured by the encoder is at layer 0. We have $\mathbf{F}_t^{l=3}$ as the final combination of temporal and spatial features, which will be passed to the decoder to generate the next frame $\widehat{\mathbf{X}}_{t+1}$. Each of the inner CMU units follows operation $\mathbf{F}_t^{l+1} = \mathsf{CMU}(\mathbf{F}_{t-1}^l, \mathbf{F}_t^l; \mathbf{W})$ as in Equation (3). Figure 2 (bottom) is a *non-overlapping* structure of PredCNN with each state used only once in each layer, thus has 2 layers (other than the encoder and decoder). We can verify that for a sequence of length T, the overlapping structure has $O(T^2)$ CMUs and the non-overlapping structure has only $O(T)$ CMUs. As a tradeoff, the overlapping one is more accurate and the non-overlapping one is more efficient.

Remarks: Different from the other CNN-based methods, our PredCNN creates hierarchical representations over video frames, where nearby input frames interact at lower layers while distant frames interact at higher layers. Compared with RNN-based chain structures, our PredCNN architecture enables a highly paralleled optimization during training process, where encoders and decoders for all frames can be trained in parallel. Future states in the hierarchical CMUs only depend on a limited number of hidden representations in the lower layers, rather than being trapped by the strong dependencies on computation of previous states in the same layer as recurrent models. This new hierarchical cascade convolution structure significantly reduces the computation cost and memory usage while successfully capturing the temporal coherence. PredCNN focuses more on capturing spatiotemporal relationship between the next frame and multiple video input frames, which is the main concern of many real applications, such as crowd flow prediction and air quality forecasting. Note that PredCNN can also predict multiple future frames by recursively feeding the predicted frames as inputs to future frames.

5 Experiments

We evaluate PredCNN with state-of-the-arts on three datasets, containing both realistic and synthetic images. Datasets and codes will be released at `https://github.com/thuml`.

TaxiBJ and BikeNYC [Zhang *et al.*, 2017] are two crowd flow prediction datasets, collected from GPS trajectory monitors in Beijing and New York respectively. We follow the same experiment settings as in [Zhang *et al.*, 2017], and predict the citywide crowd movements instead of limited blocks of several road segments. And most real spatiotemporal prediction applications exploit preprocessed gridded data rather than raw photographs, and require prediction accuracy rather than prediction frequency. Besides, we also apply our method to a commonly used video prediction dataset, Moving MNIST, to further validate the superiority of our method.

We train all of our models with $L2$ loss, using Adam optimizer [Kingma and Ba, 2015]. Unless otherwise specified, the starting learning rate of Adam is set to 10^{-4}, and the training process is stopped after 100 epochs with a batch size of 16. We choose the mean squared error (MSE/RMSE) and the mean absolute error (MAE) as two basic evaluation metrics. All experiments are implemented in Keras [Chollet and others, 2015] with TensorFlow [Abadi *et al.*, 2016] as back-ends.

Baselines We compare the proposed PredCNN with several baselines, including not only traditional and deep learning models particularly designed for crowd flow prediction, but also the state-of-the-art approaches originally applied to other video prediction tasks. Specific introductions are as follows:

- **ARIMA**: Auto-Regressive Integrated Moving Average, a well-known model predicting future time series data.
- **SARIMA**: Seasonal ARIMA for periodic time series.
- **VAR**: Vector Auto-Regressive, exploring the pairwise correlation of all flows and usually being used in multivariate time series analysis.
- **DeepST** [Zhang *et al.*, 2016]: a DNN-based spatiotemporal prediction model considering various temporal properties specified for crowd flow prediction task.
- **ST-ResNet** [Zhang *et al.*, 2017]: an updated version of DeepST with residual structure designed for crowd flow prediction. It further considers the closeness, period, trend as well as external factors all together.
- **VPN** [Kalchbrenner *et al.*, 2017]: a well-designed predictive model based on ConvLSTM. Due to the difficulty of reproducing the experimental results of the VPN, we compare our model with its suboptimal baseline version that uses RMBs instead of PixelCNN as its decoder.
- **PredNet** [Lotter *et al.*, 2017]: an efficient and effective method particularly designed for one-frame prediction.
- **PredRNN** [Wang *et al.*, 2017]: a state-of-the-art recurrent structure modeling spatial appearances and temporal variations simultaneously with dual memory LSTMs.

All compared models are evaluated on the same datasets as PredCNN. Experimental results are shown in later sections.

5.1 TaxiBJ

Settings and hyperparameters We first evaluate our models on the TaxiBJ dataset, where each frame is a 32×32 grid map collected from the taxi GPS in Beijing every half an hour, with crowd inflow and outflow being regarded as two frame channels. We rescale the pixel values of the input data from $[0, 1292]$ to $[-1, 1]$, treating them as continuous variables, and rescale the predicted pixel values back to the original range. The whole dataset contains $22,484$ piecewise consecutive frames in four time intervals (1st Jul. 2013 $\sim$ 30th Oct. 2013, 1st Mar. 2014 $\sim$ 30th Jun. 2014, 1st Mar. 2015 $\sim$ 30th Jun. 2015, 1st Nov. 2015 $\sim$ 10th Apr. 2016). To make quantitative results comparable, we follow [Zhang *et al.*, 2017] and split the whole TaxiBJ dataset into a training set of $19,788$ sequences and a test set of $1,344$ sequences.

In our experiments, we initially use the previous 4 frames to predict the next crowd flow image. By recursively taking generated images as inputs, our model is able to make predictions further into the future. PredCNN has 4 RMBs in the encoder and 6 RMBs in the decoder. The number of channels in CMUs is 64. Note that a more sophisticated architecture of the encoder and decoder might result in better performance but would also bring in more time cost and memory usage.

Table 1: Quantitative results of different methods on TaxiBJ.

Model	RMSE	MAE
ARIMA	22.78	-
SARIMA	26.88	-
VAR	22.88	-
DeepST [Zhang *et al.*, 2016]	18.18	-
ST-ResNet [Zhang *et al.*, 2017]	16.59	9.52
VPN [Kalchbrenner *et al.*, 2017]	16.75	9.62
PredNet [Lotter *et al.*, 2017]	16.68	9.67
PredRNN [Wang *et al.*, 2017]	16.34	9.62
PredCNN (1 Conv2D / 1 Conv2D)	15.90	9.71
PredCNN (1 MU / 1 MU)	15.68	9.47
PredCNN (2 untying MUs / 1 MU)	15.63	9.43
PredCNN (CMU, non-overlapping)	<u>15.23</u>	<u>9.26</u>
PredCNN (CMU, overlapping)	**15.17**	**9.22**

Results As shown in Table 1, PredCNN makes a great improvement in prediction accuracy. Its RMSE decreases from 16.34 (former best) to 15.17. To assess the effectiveness of the proposed cascade multiplicative unit (CMU), we perform ablation experiments on four PredCNN variant structures. By replacing the multiplicative units in the PredCNN architecture with ordinary convolutions (1 Conv2D / 1 Conv2D), the RMSE increases from 15.17 to 15.90. By removing one of the weight-shared MUs in the left branch of CMUs (1 MU / 1 MU), the RMSE increases from 15.17 to 15.68. By untying the two MUs in the left branch (2 untying MUs / 1 MU), the RMSE increases from 15.17 to 15.63. These results validate our design of CMU as an accurate convolutional surrogate to the recurrent unit like LSTM. Two structures of PredCNN (overlapping vs. non-overlapping) will be compared later.

We visualize the corresponding frame-wise prediction results in Figure 3. We observe that ST-ResNet, the previous state-of-the-art based on convolutions, tends to underestimate the pixel values, while recurrent models, especially

PredRNN, are likely to over-estimate them. The proposed PredCNN model usually gives more accurate predictions. For instance, the hot areas in yellow denote high crowd flow intensities, where possibly the traffic jam happens. For these regions, the prediction by ST-ResNet at time $t+1$ is obviously smaller than the ground truth, indicating a relatively high false negative rate. On the opposite, PredRNN has a high false positive rate as it often predicts higher crowd flow values, which is especially obvious at time steps $t+3$ and $t+4$. By contrast, our PredCNN model balances the F-score and predicts more accurate hot areas of the crowd flow.

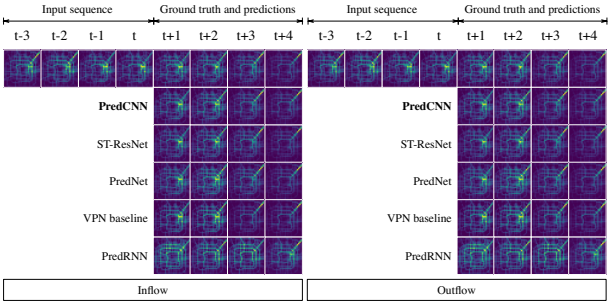

Figure 3: Samples of TaxiBJ inflow and outflow predictions. We predict the next 4 frames each with its previous 4 real frames.

Furthermore, we compare PredCNN's training time (until convergence) and memory usage with those of the baseline models (see Table 2). Our model takes less memory footprint and yields a higher convergence speed than the state-of-the-art recurrent methods. It is worth noting that ST-ResNet shares similar characteristics, which strongly proves the computation efficiency of the CNN-based models. PredCNN would thoroughly show its strength of highly paralleled optimization when dealing with longer spatiotemporal sequences.

Table 2: Training time and memory usage on TaxiBJ.

Model	Training Time (min)	Memory (MB)
ST-ResNet	258.5	1627
VPN	683.7	2369
PredNet	325.3	1601
PredRNN	340.6	2497
PredCNN (overlapping)	197.0	**1345**
PredCNN (non-overlapping)	**137.4**	**1058**

Table 1 and Table 2 show the comparison of two PredCNN structures (overlapping and non-overlapping). They have tiny quantitative differences on RMSE and memory usage but huge distinction on training time. With acceptable RMSE loss, we can substantially decrease the training time. This advantage of the non-overlapping structure is more prominent for longer input sequences in the following section 5.3.

Besides one-frame prediction, we recursively apply our PredCNN model trained with 4 inputs and 1 target to frame-by-frame prediction task and compare it with the other video prediction methods (see Table 3). At inference time, we only show our model 4 input frames and make it resemble the recurrent models to predict the next 4 frames in a rolling style. PredNet, as a strong competitor for one-frame prediction, performs worse in this scenario. Evidently, PredCNN achieves a better performance than all the others with lower RMSEs.

Table 3: Frame-wise RMSE of different methods on TaxiBJ.

Model	Frame 1	Frame 2	Frame 3	Frame 4
ST-ResNet	16.75	19.56	21.46	22.91
VPN	17.42	20.50	22.58	24.26
PredNet	27.55	254.68	255.54	255.47
PredRNN	16.08	19.51	20.66	22.69
PredCNN	**15.17**	**17.35**	**19.04**	**20.59**

5.2 BikeNYC

Settings and hyperparameters BikeNYC is another trajectory dataset taken from the New York City bicycle system in 2014, 1st Apr. $\sim$ 30th Sept., containing riding durations, starting and ending locations and times. The frames are collected once an hour, shown as a grid map of 8×16. Following the experimental settings in [Zhang *et al.*, 2017], we divide these consecutive frames into sequences and then split them into a training set of $4,148$ items and a test set of 240 items. Similar to our experiments on TaxiBJ, we transform the original data values from $[0, 267]$ to $[-1, 1]$ and use the previous 4 frames to predict the next frame.

To avoid over-fitting, the number of RMBs in the encoder and the decoder are reduced to 2 and 3. We keep each CMU with 64 channels to remain PredCNN's spatiotemporal modeling capability. Different from TaxiBJ, we assess all of the models with the image-level RMSE averaged on the number of bike stations ($N = 81$) for this dataset, which is defined as: $\sqrt{||\mathbf{X}_{t+1} - \widehat{\mathbf{X}}_{t+1}||^2/N}$, where $\mathbf{X}_{t+1}$ and $\widehat{\mathbf{X}}_{t+1}$ denote the ground truth frame and the predicted frame respectively.

Table 4: Evaluations of different methods on BikeNYC.

Model	RMSE	MAE
ARIMA	10.07	-
SARIMA	10.56	-
VAR	9.92	-
DeepST [Zhang *et al.*, 2016]	7.43	-
ST-ResNet [Zhang *et al.*, 2017]	6.37	2.95
VPN [Kalchbrenner *et al.*, 2017]	6.17	3.68
PredNet [Lotter *et al.*, 2017]	7.45	3.71
PredRNN [Wang *et al.*, 2017]	5.99	4.89
PredCNN (1 Conv2D / 1 Conv2D)	5.94	3.27
PredCNN (1 MU / 1 MU)	5.87	2.97
PredCNN (2 untying MUs / 1 MU)	5.77	2.89
PredCNN (CMU, non-overlapping)	5.65	2.87
PredCNN (CMU, overlapping)	**5.61**	**2.82**

Table 5: Performances of different methods on Moving MNIST.

Model	MSE				Sharpness	Training Time (sec./iteration)	Memory (MB)
	Frame 1	Frame 2	Frame 4	Frame 8			
VPN [Kalchbrenner *et al.*, 2017]	30.20	43.12	**62.74**	**89.24**	210.23	1.91	10603
ST-ResNet [Zhang *et al.*, 2017]	43.97	71.39	118.51	153.83	170.18	0.74	1627
PredCNN (1 MU / 1 MU)	28.48	41.87	72.20	117.45	251.41	**0.40**	**833**
PredCNN (2 untying MUs / 1 MU)	28.25	41.37	71.75	115.66	251.09	0.42	858
PredCNN (CMU, non-overlapping)	27.89	41.03	70.50	115.07	**251.74**	**0.40**	845
PredCNN (CMU, overlapping)	**26.36**	**39.24**	67.99	114.97	249.91	0.61	1345

Results Table 4 indicates that our PredCNN model outperforms other approaches on the BikeNYC dataset, even earning a better result than state-of-the-art video prediction architectures such as VPN , PredNet and PredRNN. The RMSE reduction from 5.99 (former best) to 5.61 can also clearly indicates the effectiveness of our architecture. As an ablation study, we decline RMSE from 5.94 to 5.87 by upgrading a simple convolution (1 Conv2D / 1 Conv2D) to the multiplicative unit (1 MU / 1 MU), to 5.77 by applying two consecutive multiplicative unit to the left branch (2 untying MUs / 1 MU), and to 5.61 by adding a shared weight multiplicative unit to the left branch of the cascade multiplicative unit (CMU, overlapping). Even trained with the $L2$ loss, our PredCNN model generates accurate frames with the lowest MAE value 2.82.

5.3 Moving MNIST

Settings and hyperparameters We follow the experiment setting on Moving MNIST as [Wang *et al.*, 2017]. Each sequence contains 20 consecutive frames, 10 for the input and 10 for the prediction, and each frame has 2 handwritten digits bouncing inside a 64×64 grid of image. We use $10,000$ sequences for training and $5,000$ sequences for testing.

To verify our model for capturing longer sequence features, for instance, predicting the following 10 frames recursively with 10 real input frames on Moving MNIST, we use 4 RMBs for encoder, 6 RMBs for decoder and CMUs with 64 channels for PredCNN. We use Adam optimizer with a starting learning rate of 10^{-4}, 8 video sequences per batch and the training process stopped after approximately $200,000$ iterations.

Results We compare PredCNN with a RNN-based model, VPN [Kalchbrenner *et al.*, 2017], and a CNN-based model, ST-ResNet [Zhang *et al.*, 2017]. From Table 5 we can see that PredCNN outperforms ST-ResNet in 10 frames prediction in all metrics, indicating that our model can capture better temporal dependencies across frames than previous convolution models. Compared with VPN, our model yields lower error in the first 3 frame prediction, higher average sharpness in 10 frames, as well as less training time and memory usage. Figure 4 shows the frame-wise prediction results of our model and two competitors. It is clear that our model is favourable for near-frame prediction, such as the first 3 frame prediction.

Table 5 shows the comparison of the two PredCNN structures. Compared with non-overlapping structure, overlapping PredCNN has better performance as well as longer training time and more memory usage. The overlapping structure may become extremely deep especially when the input sequence is long. We can trade off the two structures according to the sequence lengths for acceptable accuracy and training cost.

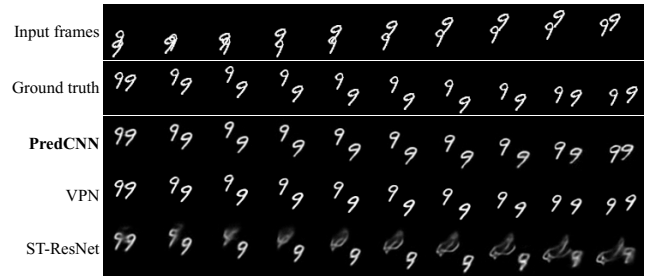

Figure 4: Visualization of predictions on Moving MNIST. We use 10 real frames as input and predict the next 10 frames recursively. We visualize the result of PredCNN using non-overlapping structure.

6 Conclusions and Future Work

This paper presents PredCNN, a convolutional architecture for spatiotemporal predictive learning. Entirely composed of convolutions, our model eases gradient propagation and reduces computation and memory burdens. Furthermore, we introduce a cascade multiplicative unit that applies more operations to previous frames, which explicitly captures the temporal dependency. A hierarchical cascade structure is proposed to capture the temporal dependencies between the next frame and input frames by stacking cascade multiplicative units. The proposed PredCNN model achieves the state-of-the-art performance on the standard Moving MNIST dataset and two challenging traffic crowd flow prediction datasets.

Video prediction is still an extremely challenging problem due to the inherent uncertainty of the future. To deal with this uncertainty, the adversarial training strategy [Goodfellow *et al.*, 2014] has been adopted to deep models. It remains unclear whether adversarial training would be useful to enhance the quality of predictions on the gridded spatiotemporal data like crowd flows. We believe integrating predictive networks with adversarial training is a promising research direction.

Acknowledgements

This work was supported by the National Key R&D Program of China (No. 2017YFC1502003), the National Natural Science Foundation of China (No. 61772299, 71690231, 61502265), and Tsinghua TNList Laboratory Key Project.

References

[Abadi *et al.*, 2016] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[Chen *et al.*, 2016] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv preprint arXiv:1606.00915*, 2016.

[Chollet and others, 2015] François Chollet et al. Keras. `https://github.com/keras-team/keras`, 2015.

[Finn *et al.*, 2016] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In *NIPS*, 2016.

[Goodfellow *et al.*, 2014] Ian J. Goodfellow, Jean Pougetabadie, Mehdi Mirza, Bing Xu, David Wardefarley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *NIPS*, 3:2672–2680, 2014.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Kalchbrenner *et al.*, 2017] Nal Kalchbrenner, Aaron van den Oord, Karen Simonyan, Ivo Danihelka, Oriol Vinyals, Alex Graves, and Koray Kavukcuoglu. Video pixel networks. In *ICML*, 2017.

[Kingma and Ba, 2015] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[Lotter *et al.*, 2017] William Lotter, Gabriel Kreiman, and David Cox. Deep predictive coding networks for video prediction and unsupervised learning. In *International Conference on Learning Representations (ICLR)*, 2017.

[Mathieu *et al.*, 2016] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. In *ICLR*, 2016.

[Mikolov *et al.*, 2010] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[Patraucean *et al.*, 2016] Viorica Patraucean, Ankur Handa, and Roberto Cipolla. Spatio-temporal video autoencoder with differentiable memory. In *ICLR Workshop*, 2016.

[Ranzato *et al.*, 2014] MarcAurelio Ranzato, Arthur Szlam, Joan Bruna, Michael Mathieu, Ronan Collobert, and Sumit Chopra. Video (language) modeling: a baseline for generative models of natural videos. *arXiv preprint arXiv:1412.6604*, 2014.

[Rumelhart *et al.*, 1988] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

[Shi *et al.*, 2015] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *NIPS*, pages 802–810, 2015.

[Simonyan and Zisserman, 2014] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, pages 568–576, 2014.

[Srivastava *et al.*, 2015] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhutdinov. Unsupervised learning of video representations using lstms. In *ICML*, 2015.

[Van Den Oord *et al.*, 2016a] Aaron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[van den Oord *et al.*, 2016b] Aaron van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. In *NIPS*, pages 4790–4798, 2016.

[Villegas *et al.*, 2017] Ruben Villegas, Jimei Yang, Seunghoon Hong, Xunyu Lin, and Honglak Lee. Decomposing motion and content for natural video sequence prediction. In *International Conference on Learning Representations (ICLR)*, 2017.

[Wang *et al.*, 2017] Yunbo Wang, Mingsheng Long, Jianmin Wang, Zhifeng Gao, and S Yu Philip. Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms. In *Advances in Neural Information Processing Systems*, pages 879–888, 2017.

[Werbos, 1990] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[Yu and Koltun, 2015] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[Zhang *et al.*, 2016] Junbo Zhang, Yu Zheng, Dekang Qi, Ruiyuan Li, and Xiuwen Yi. Dnn-based prediction model for spatio-temporal data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 92. ACM, 2016.

[Zhang *et al.*, 2017] Junbo Zhang, Yu Zheng, and Dekang Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *AAAI*, pages 1655–1661, 2017.

[Zheng, 2015] Yu Zheng. Methodologies for cross-domain data fusion: An overview. *IEEE transactions on big data*, 1(1):16–34, 2015.