KeyMemoryRNN: A Flexible Prediction Framework for Spatiotemporal Prediction Networks

SHENGCHUN WANG, XIANG LIN, AND HUIJIE ZHU
1Department of Software Engineering, Hunan Normal University, Changsha 410081, China
2Science and Technology on Near-Surface Detection Laboratory, Wuxi 214035, China

Corresponding author: Huijie Zhu (zhuhuijiehao@163.com)

This work was supported in part by the Key Research and Development Program of Hunan Province (2019SK2161) and the Key Research and Development Program of Hunan Province (2016SK2017), in part by the Foundation of Science and Technology on Near-Surface Detection Laboratory (6142414190203).

ABSTRACT Most previous recurrent neural networks for spatiotemporal prediction have difficulty in learning the long-term spatiotemporal correlations and capturing skip-frame correlations. The reason is that the recurrent neural networks update the memory states only using information from the previous time step node and the networks tend to suffer from gradient propagation difficulties. We propose a new framework, KeyMemoryRNN, which has two contributions. The first is that we propose the KeyTranslate Module to extract the most effective historical memory state named keyword state, and we propose the KeyMemory-LSTM which uses the keyword state to update the hidden state to capture the skip-frame correlation. In particular, KeyMemoryLSTM has two training stages. In the second stage, KeyMemoryLSTM adaptively skips the update of sometime step nodes to build a shorter memory information flow to alleviate the difficulty of gradient propagation to learn the long-term spatiotemporal correlations. The second is that both KeyTranslate Module and KeyMemoryLSTM are flexible additional modules, so we can apply them to most RNN-based prediction networks to build KeyMemoryRNN with different base network. The KeyMemoryRNN achieves the state-of-the-art on three spatiotemporal prediction tasks, and we provide ablation studies and memory analysis to verify the effectiveness of KeyMemoryRNN.

INDEX TERMS Recurrent neural networks, spatiotemporal prediction, prediction network.

I. INTRODUCTION

In recent years, deep learning has achieved great success in the fields of computer vision and natural language processing, especially in supervised learning. Compared with supervised learning, unsupervised learning has made few achievements. However, in the field of deep learning and computer vision, more and more people begin to pay attention to spatiotemporal prediction learning in unsupervised learning. Spatiotemporal prediction learning is to learn the spatiotemporal features from unlabeled spatiotemporal sequence data in an unsupervised manner and using them for subsequent tasks, it is similar to time series prediction [1], [2], except that the data used in spatiotemporal prediction is the data with spatial dimensions. In most cases, the subsequent tasks are to predict future sequence data. Spatiotemporal sequence data correspond to the sequence data with spatiotemporal correlations, such as precipitation radar echo data, traffic flows data or other video data. In a broad sense, any data with spatiotemporal dimension can be used as spatiotemporal sequence data. Such data is almost everywhere in daily life and scientific research work.

Because of the universal existence of spatiotemporal sequence data, spatiotemporal prediction learning is applied in many fields. For example, precipitation prediction [3] in the field of meteorology, traffic flows prediction [4] in the field of transportation, behavior recognition and prediction [5] in the field of automatic driving are extremely important prediction tasks. Although spatiotemporal prediction tasks have important applications in various fields, due to the characteristics of prediction tasks, video data is more complex than picture data and unlabeled data lacks semantic labels to drive learning, which makes spatiotemporal prediction tasks more difficult and complex than labeled...
image tasks. Therefore, how to maximize the information of correlations between existing spatiotemporal sequence data and learn spatiotemporal semantic information is important factors of the design of spatiotemporal prediction network. Because unsupervised learning manner without semantic labels is very similar to human learning and thinking, we try to analyze the human prediction manner to assist the cognitive defects of the spatiotemporal prediction network, so as to perform the difficult spatiotemporal prediction tasks more effectively.

Humans have the ability to predict the future within a certain range, which is the basis for humans to avoid danger and cooperate with others. To give a simple example, when a person is going to pass an intersection, he will first observe the situation on the road at this time to determine whether it is safe to pass the road. The situation on the road over a period is a spatiotemporal sequence data. We can simply divide the observed information into spatial information in the scene and temporal information of scene changes. Because the subjects of these changes are based on spatial images, researchers have tried to use Convolutional Neural Networks [6], which are widely used in the field of computer vision, to perform prediction tasks. For example, Oh et al. defined a CNN-based autoencoder Network for Atari game prediction [7], however, because the convolution calculation is limited by its receptive field, both the CNN prediction Network for single-frame or multi-frame input mainly focus on its spatial representation, but lack the ability to model the correlations between frames. Therefore, 3D convolution begins to use in predicting the future frames of the video sequence [8], [9], but the 3D convolution only optimized the time consistency compared with the 2D convolution but did not model the temporal sequence information. In order to effectively model spatial and temporal sequence information, ConvLSTM [3] adopted the underlying architecture of a mixture of convolutional and recurrent layers as the network architecture for spatiotemporal prediction tasks, this architecture has become the mainstream architecture in spatiotemporal prediction tasks. Such a structure is stacked with several base blocks, also known as conventional stacked structure. At present, advanced spatiotemporal prediction networks such as PredRNN [10], PredRNN++ [11], MIM [12], Conv-TT-LSTM [13] etc. are all based on this conventional stacked structure. Although this type of architecture can use convolution to model spatial representation and recurrent layers to model temporal change, ConvLSTM [3] or other advanced variant networks [10], [11], [12], [18], [20], [23] all have the inherent disadvantages of recurrent neural networks. The first is the recurrent neural network only uses the information from the previous time step node to update the memory state without building a connection between historical information and current state, which makes the recurrent neural network unable to directly capture the skip-frame correlation. The second is the vanishing gradient problem [14] of recurrent neural network. The more time step nodes the gradient passes through during BPTT [15], the more serious the gradient propagation difficulties will be, which makes it difficult for the network to learn long-term spatiotemporal correlations via gradient descent. Due to the existence of these disadvantages, the network will have difficulty in fitting in some situations in the prediction task, this is the dilemma of the spatiotemporal prediction learning. For example, when the moving subject leaves the field of view or blocked, that is, when the short-term spatial information cannot restore the complete spatial feature of the moving subject, it is necessary to build a connection between the current frame and the historical frame containing the complete spatial feature of the moving subject. Taking the previously mentioned person passing through the intersection as an example, when the approaching vehicle blocks the distant vehicle, the estimation of the motion of distant vehicles can only use the memory before occlusion, this is the so-called need to capture skip-frame correlation. And when dealing with complex spatiotemporal dynamics, long-term correlations plays a key role, such as long-term memory information is needed to model long-term trends and long-term motion context is needed to guide short-term changes and reduce prediction errors by reducing incorrect possible searching space. Similarly, taking people passing through the intersection as an example, short-term information can be used to estimate the speed of vehicles, while acceleration or other regular speed changes need to be estimated through memory and observation of long-term frames, which is the so-called need to learn long-term spatiotemporal correlation.

What is described above is the dilemma of spatiotemporal prediction networks, in order to solve this dilemma, we observe the process of human beings to capture information and recall, then we found that in the process of humans’ rapid capture of information and efficient memory, keyword information plays an important role. When reading publications, people first get an overview of the content of the full text from the keywords; when learning a dance, people tend to memorize a few key action frames instead of memorizing the actions of each frame. We illustrate the different effects of using full information or key information in the deep network through an example of clapping action. The clapping action denoted by full video frames as shown in Fig. 1 (a). When this group of picture information is used in the deep RNN-based prediction networks, due to the long-term dilemma mentioned earlier, the recurrent unit relies on short-term information to update the memory state, that is, the memory state only pays attention to the short-term spatiotemporal changes (such as a small swing of the arm) but failed to model long-term spatiotemporal correlation to capture long-term motion context (such as clapping). The long-term motion context limits the possibility of arms swing change (for example, it limits the possibility of raising the arm continuously), which reducing incorrect possible searching space. If the guidance of the long-term motion context is lost, the various possibilities of arm swing changes are taken as the mean value in the calculation of the distance loss, which causes the errors and the blur of the prediction picture.
And the clapping action denoted by 10 video frames as shown in Fig. 1 (b). When this group of picture information is used in the deep prediction networks, since there are fewer time step nodes in recursions, it is less affected by the vanishing gradient problem to retain the complete long-term motion context as much as possible in the recurrent process and effectively provide long-term information guidance. In other words, keyword information of the same length has higher efficiency and longer memory time than full information.

Another work of keyword is to quickly recall similar scenes and related memories, for example, when reciting speech words, people often suddenly forget what has been recited, at this time, only one keyword prompt is needed instead of repeating a whole paragraph, so that the man can recall the relevant content. In neural networks, this may provide an idea that we can recall the required information by establishing a connection with a keyword frame instead of a connection with multiple consecutive frames. Inspired by these, we integrated these ways of efficiently recalling via the keyword into the RNN-based network, to proposing a new KeyMemoryLSTM framework. KeyMemoryRNN consists of three parts: base recurrent unit, KeyTranslate Module, KeyMemoryLSTM.

II. RELATED WORK

A. DETERMINISTIC SPATIOTEMPORAL PREDICTION

Recurrent neural networks are widely used in spatiotemporal prediction tasks. Ranzato builds an RNN model to perform prediction of future video frames and interpolation of intermediate frames for visual words obtained by clustering image blocks [16], but the model can only predict the previous frame. Srivastava introduced the end-to-end LSTM encoder-decoder model into the video field to perform the tasks of predicting sequence and reconstructing sequence at the same time [17]. However, the fully connected LSTM is difficult to reconstruct the prediction space. In order to capture the spatiotemporal changes in the sequence at the same time, Shi et al. used convolution operations to replace the full connection operations for the hidden state update function [3]. The proposal of ConvLSTM became an important cornerstone work in the field of spatiotemporal prediction. After that, many RNN-based networks based on ConvLSTM appeared. For example, Shi et al. introduced learnable convolution and proposed that the TrajGRU model [18] follows the idea of optical flow to capture movement. Lotter et al. proposed that the predictive neural network PredNet [19] performs implicit modeling by extracting the features of objects, and predicts future frames via feature learning, although it performs poorly in prediction, it has achieved good results in extracting feature information. In order to transfer the top-level abstract information to the bottom layer of the next frame prediction, PredRNN [10] introduces a new spatiotemporal recurrent unit Spatio-Temporal LSTM (ST-LSTM). The memory state of the ST-LSTM unit is transferred in a zigzag way between recurrent units. PredRNN++ [11] increases the depth of state transitions via cascaded CausalLSTM recurrent units and enhances the network’s...
Transformer work in the field of vision [26], [27], [28] has proved its effectiveness. In these works, the image is split into multiple patches and the patches are linearly mapped to linear embedding. Timesformer [29] extends this mechanism to the spatiotemporal dimension, treating each frame of the video as a patch of the image and uses a similar linear mapping method to map spatiotemporal features into linear embedding, just like embedding NLP word. Inspired by this, in our work, we use a similar method to map each historical memory state into a linear embedding, and use the self-attention mechanism to calculate the correlation between the historical memory state and the current prediction process, and use the historical memory state which most relevant to the current prediction process as keyword state for subsequent calculations.

### III. METHOD

#### A. DILEMMA OF SPATIOTEMPORAL PREDICTION

The spatiotemporal convolutional recurrent network as shown in Fig. 2 (Left), in the spatiotemporal information flow, different types of states are transferred and updated in time transitions between stacked blocks, which makes the path of gradient propagation longer as increase of time step nodes. This causes difficulties in gradient backpropagation.

The update equations of ConvLSTM for the th time step of the th layer are as follows:

\[
\begin{align*}
    i_t &= \sigma \left( W_{ci} X_{t-1} + W_{hi} H_{t-1} + W_{ci} \odot C_{t-1} + b_i \right) \\
    f_t &= \sigma \left( W_{cf} X_{t-1} + W_{hf} H_{t-1} + W_{cf} \odot C_{t-1} + b_f \right) \\
    g_t &= \tanh \left( W_{cg} X_{t-1} + W_{hg} H_{t-1} + b_g \right) \\
    C_t &= f_t \odot C_{t-1} + i_t \odot g_t \\
    o_t &= \sigma \left( W_{co} X_{t-1} + W_{ho} H_{t-1} + W_{co} \odot C_t + b_o \right) \\
    H_t &= o_t \odot \tanh \left( C_t \right)
\end{align*}
\]

\( \sigma \) is the sigmoid function, \( W \) denotes the convolution kernel, \( \odot \) is the convolution operation, \( \odot \) is the Hadamard product. \( X_{t-1}, H_{t-1}, C_{t-1} \) is input state, hidden state and memory state respectively, and \( i_t, f_t, g_t, o_t \) are input gate, input modulation gate, forget gate and output gate respectively, the subscript \( t \) correspond to the th time step.

It can be seen from (1) that there are multiple calculations of gate and state in the transition process of each memory state, and the complicated calculation further exacerbates the vanishing gradient problem of the network. In the gradient descent learning process of the network, the goal is to find a set of optimal transfer function weights so that an optimal prediction result can be obtained for each input sequence information. However, due to the above-mentioned vanishing gradient problem, the gradient decays with the recursion of time steps. Therefore, for the gradient update calculation of a certain time step, the loss of large time step span has less influence on it. In other words, the network tends to be more optimize short-term forecast results, which makes it difficult for the network to learn the correlation between long-term states via gradient descent.

### B. VISUAL TRANSFORMER

Since Transformer [25] was proposed and effectively applied to the field of natural language processing, many works began to transfer Transformer to various fields. For example,
Figure 2. Stacked structure that uses ConvLSTM as the based block. Left: Conventional stacked structure, the green arrows correspond to spatiotemporal information flow. (The hidden state transferred between layers corresponds to the input state in (1)) Right: Stacked structure with KeyMemoryRNN framework, external modules KeyTranslate Module and KeyMemoryRNN are added to the structure, the orange arrows correspond to key memory information flow.

On the other hand, from the (1) we can see that each memory state transition only uses the hidden state of the current time step node and the previous time step node. It means that the memory state can only directly capture the short-term information, but difficult to directly capture the long-term correlation. Due to the vanishing gradient problem, the memory state is also difficult to capture long-term correlation information indirectly via learning. Therefore, the long-term memory state originally designed to store long-term memory information is difficult to effectively store long-term memory.

When solving simple linear dynamics, the prediction process can only rely on short-term information, but obviously most of the prediction tasks are not simple linear dynamics. For example, if occlusion and overlap occur in prediction process, the original spatial information needs to be restored from the historical memory state and when modeling complex spatiotemporal dynamics, motion context information based on long-term memory correlation is also very important, however, the memory information stored in the memory state is more short-term spatiotemporal information rather than long-term spatiotemporal information.

The short-term spatiotemporal information can be understood as local small changes as the swing of the arms in Fig. 1, the long-term motion context can be understood as global long-term changes as the clapping action. The motion context information is actually used as a constraint condition to limit the possibility of actions, just as in a clapping action, the palm will always open and close, rather than open endlessly. The compression of possibility space is very important for prediction task because the prediction task is a multi-possibility search space task and the future is multi-possibility. For the same amount of information, the deterministic network system will obtain its average results under various possibilities, resulting in error and blurring in the prediction results.

In order to solve the dilemma of spatiotemporal prediction network, we propose a new prediction framework KeyMemoryRNN as shown in in Fig. 2 (Right). Next, we will introduce the overall structure of KeyMemoryRNN and the specific structures of KeyTranslate Module and KeyMemoryLSTM in detail.

B. KEY MEMORY RNN

The structure of KeyMemoryRNN is shown in Fig. 2 (Right). Block is a base recurrent unit (such as ConvLSTM [3], ST-LSTM [10]), and we take ConvLSTM as an example. In front part of the network is a conventional stacked structure with base recurrent units. Since there is less historical memory information in the front part, the transitions of memory state do not require the use of historical states, in order to better distinguish the different states with/without historical memory information participating in the calculation, the stacked blocks of the front part of the network individually share a set of weights. We call this area a memory buffer area.

The overall equations of the memory buffer area for the $t$th time step of the $l$th layer are as follows:

$$H_{l}^{t}, C_{l}^{t} = \text{Block}_l \left( X_{l}^{t-1}, H_{l}^{t-1}, C_{l}^{t-1} \right)$$ (2)
We call the latter part of the network that prediction area. In order to store long-term memory more effectively, the prediction area adds two new modules to the conventional stacked structure (as shown by the red dashed box in the Fig. 2 (Right)). The dashed box in the upper is the KeyTranslate Module and the dashed box in the lower part is KeyMemoryLSTM.

The overall equations of the prediction area for the $l$th time step of the $l$th layer are as follows:

$$H^l_t, C^l_t = \text{Block}_2 \left( X^{l-1}_t, H^{l-1}_t, C^{l-1}_t \right)$$

$$E^l_t = \text{KeyTranslate} \left( C^{l-1}_{t-n-2d-2}, C^{l-1}_t \right)$$

$$U^l_t, T^l_{t-1} = \text{KeyMemoryLSTM} \left( E^l_t, U^l_{t-1}, C^l_{t-1}, H^l_{t-1} \right)$$

$$H^l_t, C^l_t = \text{Block}_2 \left( X^{l-1}_t, H^l_{t-1}, T^l_{t-1} \right) \quad (3)$$

$L$ is the total number of layers of the stacked block. Except for $L$th layer, the state transition method of other layers is the same as that of memory buffer area (but do not share the weight with the memory buffer area). The modules about the same as that of memory buffer area (but do not share weights too).

$E^l_t$ is the keyword state, which is extracted from the historical memory state by the KeyTranslate Module via a series of self-attention operations. $U^l_t$ is the key memory state, and $T^l_{t-1}$ is the mixed memory state. KeyMemoryLSTM uses the keyword state $E^l_t$ to update its own key memory state $U^l_t$ while outputting the mixed memory state $T^l_{t-1}$ for subsequent calculation of the base recurrent unit. We will give the specific equations of KeyTranslate Module and KeyMemoryLSTM in section C.

Taking into account the diversity of spatiotemporal prediction tasks and the differences between specific tasks, there are different adaptation models in different spatiotemporal prediction tasks. At the same time, due to the rapid development of the spatiotemporal field, new RNN-based variant networks are constantly being designed. Therefore, the KeyMemoryRNN has a low degree of coupling with the base recurrent unit. There is no need to change the calculation method of the original base recurrent unit in the code. Logically, KeyMemoryRNN only helps the upper current memory state to establish an interaction with the historical memory state, so as to enhance the long-term memory ability of the network without changing the state transition logic of the recurrent unit. Therefore, the KeyMemoryRNN framework can be adapted to combine a variety of prediction networks, such as ConvLSTM [3], PredRNN [10], PredRNN++ [11], MIM [12], and even other new RNN-Based networks that may be designed in the future. KeyMemoryRNN framework combines appropriate prediction networks in different tasks and uses KeyMemoryRNN to strengthen long-term memory ability to learn long-term correlation and skip-frame correlation on its prediction benchmark, so as to achieve better prediction perform.

C. BUILDING BLOCK: KeyTranslate MODULE AND KeyMemoryLSTM

In this section, we will introduce in detail our thinking process of proposing KeyTranslate Module and KeyMemoryLSTM and the working process of these modules.

The network is connected from input to output via hidden state transitions between layers. Although the memory state in the prediction network is not directly transit between layers, the memory state is used to update the hidden state of each layer.

Since the state update from input to output is mainly carried out via the update of hidden state, the hidden state contains more current spatial information and change information of the previous time step node. Through the use of memory state, hidden state captures spatiotemporal information and change information in a certain period. The goal of network learning is that the memory state of each time step node can store all the spatiotemporal information and change information required for the prediction of the current time step node. However, through our analysis of the long-term difficulty of prediction network in section A, we can see that only part of the spatiotemporal information and short-term change information are saved in the memory state. Therefore, we hope to find a way to store the full-time spatiotemporal information as much as possible.

As analyzed in section A, the key of the difficulty to store long-term information in memory state is that the memory state updates at each time step node, and this update is a short-term perception range update that only depends on the state of the previous node and the current node. Therefore, suppose we can build a separate memory information flow in some way, and this separate memory information flow does not update at every time step, then this memory information flow will pass through fewer nodes, and the information flow with fewer nodes can overcome the difficulties of long-term memory and store more long-term memory. The idea that fewer node information flow can store more long-term memory is proved by experiments in EXPERIMENTS. However, it is worth considering that when the update of memory state is still the update with short-term perception range and only some memory information flow nodes are deleted fixedly, although the length of the memory information flow is shortened, such short memory information flow cannot guarantee the effectiveness of the information content contained therein. Because we cannot guarantee that the nodes we delete are redundant.

In order to solve the above problems, inspired by the human memory mode, we introduced the concept of keyword. As we described in INTRODUCTION, keyword is main focus of a segment. People can know the overall content by reading the keyword information during fast reading and memory. Therefore, if we divide the historical memory state sequence into multiple overlapping memory state segments, then extract
the keyword from each memory state segment and construct these keywords into a new information sequence, and finally delete the redundant duplicate keywords, so that we can build a short key memory information flow containing most of the effective information and the overall context information as described above.

Therefore, we proposed the KeyTranslate Module to extract keyword state. The detailed equations of KeyTranslate Module for the \( t \)th time step are as follows:

\[
\begin{align*}
Q^L_{t-n-2:t-2} &= \text{Map} \left( C^L_{t-n-2:t-2} \right) \\
K^L_{t-1} &= \text{Map} \left( C^L_{t-1} \right) \\
S^L_{t-n-2:t-2} &= K^L_{t-1} \cdot \left( Q^L_{t-n-2:t-2} \right) \\
N &= \text{Index} \left( \text{Max} \left( S^L_{t-n-2:t-2} \right) \right) \\
E^L_t &= C^L_N
\end{align*}
\]  

(4)

At \( t \)th time step, we cut out a sequence of memory states \( C^L_{t-n-2:t-2} \) which containing \( n \) nodes in the historical memory state via the sliding window with the size of \( n \). The memory state sequence is mapped into query vector \( Q^L_{t-n-2:t-2} \) after convolution, flattening and linear full connection processing by the Map method in Fig. 3. Then perform the same processing on the memory state \( C^L_{t-1} \) to get the key vector \( K^L_{t-1} \). (In fact, all the memory states via the Map method will generate two vectors \( Q^L_{t-1} \) and \( K^L_{t-1} \). As a more detailed description of Map, each historical memory state \( C^L \) which is \( R^{C \times H \times W} \) tensor is mapped to a \( R^{2 \times H \times W} \) tensor via \( 3 \times 3 \) convolution operation, and then split it into two \( R^{1 \times H \times W} \) tensor via the split operation. The \( R^{1 \times H \times W} \) tensor is finally mapped to the \( R^{1 \times (H \times W)} \) query vector and the key vector via the flattening and linear fully connected layers.) \( K^L_{t-1} \) is transpose of \( K^L_{t-1} \). The correlation score sequence of the memory state \( S^L_{t-n-2:t-2} \) is obtained via the dot product operation, its subscript corresponds to the subscript of memory state. The Max operation takes the maximum value in the sequence \( S^L_{t-n-2:t-2} \) and obtains the time subscript \( N \) via Index operation. And \( C^L_N \) is the keyword state of \( t \)th time step.

The method of embedding and self-attention calculation is inspired by the image Transformer. In the image transformer [26]–[28], the image is divided into patches, and the patches are mapping into embedding vectors via the trainable fully connected linear layer. These embedding vectors can be further calculated via the self-attention mechanism to obtain the expression of the whole image. It can be considered that each embedding vector denote the semantic information of the patch. We regard the memory state sequence as a whole, and the memory state of each time step node is equivalent to a patch in the whole. The embedding vector of the memory state denote the spatiotemporal semantic information of the memory state. The dot product of the key vector and the query vector is a calculation of self-attention mechanism; dot product results correspond to vector correlation. The historical memory state with the highest correlation score can be
In order to further integrate the keyword state into the base network, we propose the KeyMemoryLSTM. The working memory state needs to be restored. In order to reduce the number of time step nodes passed by the key memory information flow to retain the memory information for a longer time, we adopt a two-stage form for the update of the key memory state $U^L_t$.

In the first stage as shown by Fig. 4 (a), the state is updated at each time step in the prediction area as shown in (5), so that the network can better learn the complete extraction mode of keyword state.

When the training loss in the first stage no longer decreases, the network turn into the form of the second stage as shown by Fig. 4 (b).

The detailed equations of the second stage are as follows:

$$d = \text{Expands}(\text{tanh}(a \bullet (\sum_{i=1}^{C \times H \times W} (E^L_{i-1}(i) - E^L_{i-1}(i))^2)))$$

$$i_t = \sigma(W_{ik} \ast U^L_{i-1} + W_{ic} \ast E^L_{i-1})$$

$$f_t = \sigma(W_{fk} \ast U^L_{i-1} + W_{fc} \ast E^L_{i-1} + b_f)$$

$$g_t = \tanh(W_{gk} \ast U^L_{i-1} + W_{gc} \ast E^L_{i-1})$$

$$U^L_t = f_t \odot U^L_{i-1} + i_t \odot g_t \quad (6)$$

The first formula in (6) measures the L2 distance loss between the current keyword state $E^L_t$ and the keyword state $E^L_{i-1}$ of the previous time step node and the keyword state $U^L_{i-1}$ of the previous time step node and the keyword state $U^L_{i-1}$.

To some extent, it can be imagined that when the memory states are relatively similar, the embedding vectors will be relatively similar, which means that the keyword states are likely to be the same, in other words, the change of keywords always occurs at the time step with drastic changes in memory state, so the information sequence composed of all keywords is a sequence of the overall action context in a larger receptive field observing, which is also the logical base for us to provide long-term motion context through keyword information. Moreover, when the prediction object is blocked and overlapped, it is often the time step that the memory state changes drastically, so it is also the time when the skip-frame memory state needs to be restored.

In order to further integrate the keyword state into the base network, we propose the KeyMemoryLSTM. The working process is divided into two parts, the first part is the update of the key memory state $U^L_{i-1}$, and its detailed equations for the $t^{th}$ time step are as follows:

$$i_t = \sigma(W_{ik} \ast U^L_{i-1} + W_{ic} \ast E^L_{i-1})$$

$$f_t = \sigma(W_{fk} \ast U^L_{i-1} + W_{fc} \ast E^L_{i-1} + b_f)$$

$$g_t = \tanh(W_{gk} \ast U^L_{i-1} + W_{gc} \ast E^L_{i-1})$$

$$U^L_t = f_t \odot U^L_{i-1} + i_t \odot g_t \quad (5)$$

The key memory state $U^L_{i-1}$ of the current node extracted via the KeyTranslate Module. In this part, the key memory state $U^L_t$ is updated to form a key memory information flow by using the keyword information. In order to reduce the number of time step nodes passed by the key memory information flow to generate mixed memory states.

The key memory state $U^L_{i-1}$ is updated using the key memory information flow to retain the memory information for a longer time, we adopt a two-stage form for the update of the key memory state $U^L_t$.

FIGURE 4. Specific structure of KeyMemoryLSTM. (a)The first part of KeyMemoryLSTM (first stage) which used to generate key memory information flow. (b)The first part of KeyMemoryLSTM (second stage). The red arrow part is the new structure of second stage, which is used to shorten the length of key memory information flow. (c)the second part of KeyMemoryLSTM which used to mix key memory information flow and conventional spatiotemporal information flow to generate mixed memory states.
to the gating of 0-1 via the amplification factor $a$ and tanh function. Expands is the operation of reshaping the gating $d$ to the same tensor dimension as $U_{t-1}^L$. Amplification factor $a$ is a fixed constant, usually 255. (The effect of this factor is to make the tanh function more saturated or zero.) The characteristic of tanh is that the output is 0 when the input is 0, and the output is 1 when the input is large. (Since the distance loss is always greater than 0, tanh only works in the area greater than 0.) Therefore, when the same keyword is extracted at $t^\text{th}$ time step and $t-1^\text{th}$ time step, $E_t^L$ and $E_{t-1}^L$ are equal, $d$ is equal to 0, then $U_t^L$ is completely equal to $U_{t-1}^L$, that is, there is no update in this time step, and the gradient can pass directly in back propagation. In other words, the key memory information flow deletes this time step node. When the distance between $E_t^L$ and $E_{t-1}^L$ is large, $d$ is completely saturated equal to 1 and the equation at this time is completely equivalent to the equation of one stage.

The second part of KeyMemoryLSTM deals with the generation of mixed memory states $T_{t-1}^L$. The detailed equations of this part are as follows:

$$
i_t' = \sigma \left( W_{fu} * U_t^L + W_{fc} * C_{t-1}^L + W_{fh} * H_{t-1}^L \right)
$$

$$
f_t' = \sigma \left( W_{fu} * U_t^L + W_{fc} * C_{t-1}^L + W_{fh} * H_{t-1}^L + b_f \right)
$$

$$
g_t' = \tanh \left( W_{gu} * U_t^L + W_{gc} * C_{t-1}^L + W_{gh} * H_{t-1}^L \right)
$$

$$
T_{t-1}^L = f_t' \odot C_{t-1}^L + i_t' \odot g_t'
$$

(7)

The calculation of mixed memory state $T_{t-1}^L$ generates gates using key memory state $U_t^L$, memory state $C_{t-1}^L$ and hidden state $H_{t-1}^L$ to control the ratio of memory state to key memory state in mixed memory state. We Intentionally did not use hidden state in the first part but used hidden state in the second part. Because there is more current spatial information in the hidden state, and we avoid the interference of the current spatial information on the key memory information flow and reduce the difference between the two stages of the first part. In the top block of the prediction area, $T_{t-1}^L$ replaces the original memory state $C_{t-1}^L$ as the input memory state of the block to provide additional long-term information in the calculation of the subsequent base recurrent unit as the (8).

$$
H_t^L, C_t^L = \text{Block2}\left(X_{t-1}^L, H_{t-1}^L, T_{t-1}^L\right)
$$

(8)

In general, through the keyword mechanism and the construction of key memory information flow, KeyMemoryRNN goes surpasses the past prediction methods, effectively retains long-term historical memory and establishes a way to capture skip-frame correlation, and uses historical information to guide prediction while maintaining the time consistency of the original memory state.

IV. EXPERIMENTS

In order to measure the performance of our proposed method, three spatiotemporal datasets are used in this experiment: Moving MNIST dataset, KTH Action dataset, Radar Echo dataset.

A. MOVING MNIST

Moving MNIST [17] is a synthetic dataset. By randomly selecting two digits in MNIST and randomly placing the digits in the $64 \times 64$ image, each digit is given a random initial speed size and direction, moves at a constant speed, and bounces when touching the image edge, resulting in a sequence of 20 frames in total, of which 10 frames are input frames and other 10 frames are prediction frames. The difficulty of prediction task of the moving MNIST lies in the frequent overlap. Since the starting position and speed of digits are given randomly, there is a great probability that digits will overlap in the process of moving. Especially when this overlap occurs in the prediction part, due to the drastic change of short-term information, the recovery of digital complete shape depends on the historical memory obtained by skip-frame connection. Therefore, the experimental results of moving MNIST dataset can show the performance of recover skip-frame memory information when the spatiotemporal process encounters drastic changes.

1) DATASETS AND SETUP

In order to compare with the most advanced network, our dataset setting adopts the same setting as the SOTA network on the Moving MNIST dataset Crevnet [22]: the digits of the training set and the test set are sampled in the mutually exclusive subset of Moving MNIST dataset to avoid data leakage, and evaluated with a fixed test set of 5000 sequences. All our models are trained on the Adam optimizer with an initial learning rate of $1 \times 10^{-4}$ and use reverse scheduled samplings [30] to stitch discrepancy in the different time step prediction processes and use the L2 loss as the training loss. The training batch is 8 and the iteration number is 150000 iterations. The convolution kernel size of KeyMemoryLSTM is $5 \times 5$, and the number of state channels in the network is 128.

2) MAIN RESULT

We compare KeyMemoryRNN with eight benchmark networks: ConvLstn [3], TrajGRU [18], PredRNN [10], PredRNN-V2 [30], CausallSTM [11], MIM [12], PhydNet [21], CrevNet [22]. CrevNet is state-of-the-art on Moving MNIST dataset. In order to measure the performance of our model, we used the mean square error (MSE) to estimate the absolute pixel-wise errors, the frame-wise structural similarity index measure (SSIM) to evaluate the structural similarity within the spatial neighborhood. All MSE and SSIM result of the benchmark networks use the data from the original paper. Lower MSE or higher SSIM correspond to better prediction performance. As shown in Table 1, in the 10 to 10 Moving MNIST test, we built the KeyMemoryRNN based on ST-LSTM and CausallSTM respectively. The KeyMemoryRNN based on CausallSTM surpasses the SOTA network CrevNet and achieves the best result.
3) QUALITATIVE COMPARISON

The Fig. 5 shows an example of the difficult prediction frames. In the prediction part, the digit 4 and digit 6 overlap continuously. Before the two digits overlap, most models can accurately predict the simple linear movement of the digits, but when overlapping occurs, other models lose the original shape of digit 6 and the prediction frames becomes more and more blurred in the prediction process, while our model still maintains the original shape of digit 6 and digit 4 and keeps the prediction frames clear. Because the prediction results of most models are based on adjacent spatiotemporal information and memory, while our model has excellent ability of restore skip-frame memory information.

4) ABLATION STUDY

In order to analyze the contribution of each module, we conducted ablation study of multiple modules. The KeyTranslate Module refers to directly using the keyword state extracted by the KeyTranslate Module for the state update of the base recurrent unit. The results are summarized in the Table 2. The table shows that after the keyword state is directly introduced through the KeyTranslate Module, although a skip-frame connection is built, this method does not uniformly model the keyword state and memory state, which negatively affects the time consistency, so only a small improvement is achieved. By adding KeyMemoryLSTM unit to uniformly model keyword state and memory state to form key memory information flow, this method maintains time consistency and greatly improves

![CausalLSTM+KeyMemoryRNN](image)

![CausalLSTM](image)

FIGURE 6. Information flow in CausalLSTM with/without KeyMemoryRNN. Key-M Flow correspond to key memory information flow; Base flow correspond to conventional spatiotemporal information flow. The green arrow corresponds to the skip-frame connection paths of multiple time steps. The red arrow corresponds to the complete spatial information transmission path related to the prediction frame in the red box.

In the KeyMemoryRNN framework, the frame information of 14th time step skips the time of 15th to 16th time step and directly participates in the prediction of 17th time step.

4) ABLATION STUDY

In order to analyze the contribution of each module, we conducted ablation study of multiple modules. The KeyTranslate Module refers to directly using the keyword state extracted by the KeyTranslate Module for the state update of the base recurrent unit. The results are summarized in the Table 2. After the keyword state is directly introduced through the KeyTranslate Module, although a skip-frame connection is built, this method does not uniformly model the keyword state and memory state, which negatively affects the time consistency, so only a small improvement is achieved. By adding KeyMemoryLSTM unit to uniformly model keyword state and memory state to form key memory information flow, this method maintains time consistency and greatly improves

---

TABLE 1. Prediction results on the moving MNIST test set. The data in the upper column comes from the original paper, and the data in the lower column comes from our experiment.

| Model                      | MSE  | SSIM |
|----------------------------|------|------|
| ConvLSTM [3]               | 103.3| 0.707|
| TrajGRU [18]               | 106.9| 0.713|
| PredRNN [10]               | 56.8 | 0.867|
| PredRNN-V2 [30]            | 48.4 | 0.891|
| MIM [12]                   | 44.2 | 0.910|
| PhyDNet+ConvLSTM [21]      | 24.4 | 0.947|
| CrevNet+ST-LSTM [22]       | 22.3 | 0.949|
| CausalLSTM [11]            | 28.1 | 0.927|
| ST-LSTM+KeyMemoryRNN       | 23.2 | 0.940|
| CausalLSTM=KeyMemoryRNN (ours) | 20.2 | 0.951|
TABLE 2: Ablation study on the moving MNIST test set: analyze the effect of KeyTranslate module and two-stage KeyMemoryLSTM, correspond to percentage increase relative to base module.

| Model                  | MSE | △MSE | SSIM | △SSIM |
|------------------------|-----|------|------|-------|
| CausalLSTM (BaseModel) | 28.1| -    | 0.927| -     |
| +KeyTranslate Module   | 26.6| 5.3% | 0.930| 0.3%  |
| +One Stage KeyMemoryLSTM | 21.6| 23.1%| 0.946| 2.0%  |
| +Second Stage KeyMemoryLSTM | 20.2| 28.1%| 0.951| 2.6%  |

TABLE 4: Ablation study: Ablation experiment on the moving MNIST test set for the mix of hidden state, memory state and historical state.

| Model                  | MSE | △MSE | SSIM | △SSIM |
|------------------------|-----|------|------|-------|
| BaseModel              | 28.1| -    | 0.927| -     |
| w H,w/o C              | 30.4| -7.5%| 0.918| -0.9% |
| w H,w C                | 27.3| 2.8% | 0.919| -0.8% |
| w/o H,w C (ours)       | 20.2| 28.1%| 0.951| 2.6%  |

The effect. Through the function of second stage KeyMemoryLSTM, the key memory information flow is shortened to retain long-term information. However, because the moving MNIST prediction task benefits more from the restored of skip-frame information than the use of long-term memory, so it only got a small improvement.

To analyze the effect of different KeyMemoryRNN architectures, we set up comparative experiments to compare these module architectures at different layers. The results are summarized in the Table 3. The experimental results show that only attaching relevant modules to the top of the network can achieve the best prediction results, because the memory state at the top of network is closer to the prediction results, and its semantic information is more suitable for building long-term movement information flow.

TABLE 3: Comparative experiments on the moving MNIST test set: compare the effect of the KeyMemoryRNN additional modules in different layers. The number of total layers correspond to the number of layers added with additional modules, layer correspond to the layer where add the additional modules.

| The Number of Total Layers | Layer | MSE | SSIM |
|---------------------------|-------|-----|------|
| 0 (BaseModel)             |       | 28.1| 0.927|
| 1 (ours)                  | 4     | 20.2| 0.951|
|                           | 3     | 23.7| 0.936|
|                           | 2     | 25.4| 0.932|
|                           | 1     | 29.2| 0.913|
|                           | 1,2   | 25.7| 0.927|
|                           | 2     | 22.6| 0.941|
|                           | 1,2,3,4| 23.6| 0.933|

We conducted an experiment to mix the hidden state with the historical hidden state via an equivalent mechanism. The results are summarized in the Table 4. The hidden state contains more current frame information, the method of mixing the hidden state with the historical hidden state leads to the loss of current frame information to a certain extent, resulting in negative effects.

B. KTH ACTION

Kth Action dataset [31] is a human actions dataset, which contains six type of human actions: walking, jogging, running, boxing, waving and clapping. There are 25 subjects in 4 different scenarios which include indoor, outdoor, scale variations and different clothes for each action. The video clip is captured with a static camera with a frame rate of 25 FPS, and the average length of the video clip is four seconds. Different from the simple linear movement of moving digits, there are many high-order variations and periodic variations in human action that depend on long-term memory. At the same time, because the dataset contains a variety of action, and the patterns of change of different action are quite different, so the prediction frames depends on the guidance of long-term motion context information. Therefore, the experiment results of KTH Action dataset can evaluate the storage ability of the network for long-term memory and global information (such as long-term motion context).

1) DATASETS AND SETUP

In order to make a fair comparison with the previous models, we adopted the same experimental settings as the previous models and each frame was adjusted to a gray image with a size of 128 × 128. In order to avoid data leakage, the videos of 1 to 16 subjects were used as the training set and the videos of 17 to 25 subjects were used as the test set. Each experimental sequence has a total of 30 frames, of which 10 frames are input frames and 20 frames are prediction frames. All our models were trained with L2 loss on the Adam optimizer at an initial learning rate of 1 × 10⁻⁴, with 16 batchsize and 200,000 iterations, and use reverse scheduled samplings [30] to stitch discrepancy in the different time step prediction processes. Layer Normalize [32] strategy was used to normalize hidden layer.

2) MAIN RESULT

We compare KeyMemoryLSTM with seven benchmark networks and E3D-LSTM [20] is the SOTA on KTH Action dataset. We use Peak Signal to Noise Ratio (PSNR) and SSIM as metrics to evaluate the prediction result. Higher PSNR and SSIM denote better prediction results. The prediction results are shown in Table 5.

3) QUALITATIVE COMPARISON

The prediction examples of KTH Action test set are shown in the Fig. 7. Most of the prediction networks lose the original outline of the human and the motion track deviates from the real situation at the end of the prediction. For example, in example 1, people’s arms are constantly raised or disappeared by other models. Due to the existence of key memory information flow, our model provides long-term trend information and long-term motion context information for the prediction process. Therefore, even in the later stage of prediction, the prediction results of our model are still clear, and the track of human motion is close to the real situation.
TABLE 5. Prediction results on the KTH action test set. The data in the upper column comes from the original paper, and the data in the lower column comes from our experiment. (Because the calculation method of metrics used in the original paper of Conv-TT-LSTM is different from that used in other papers, the experimental results of Conv-TT-LSTM use the experimental results given in PredRNN-V2.).

| Model                          | PSNR | SSIM  |
|-------------------------------|------|-------|
| ConvLSTM [3]                  | 23.58| 0.712 |
| TrajGRU [18]                  | 26.97| 0.790 |
| PredRNN-V2 [30]               | 28.37| 0.838 |
| PredRNN++ [11]                | 28.47| 0.865 |
| E3D-LSTM [20]                 | 29.31| 0.879 |
| Conv-TT-LSTM [13]             | 27.62| 0.815 |
| ST-LSTM [10]                  | 27.98| 0.844 |
| ST-LSTM+KeyMemoryRNN          | 30.79| 0.875 |

FIGURE 7. Two prediction examples on the KTH action test set.

4) MEMORY ANALYSIS
We believe that the reason why KeyMemoryRNN can get better prediction results than other models is that the key memory information flow built by KeyMemoryLSTM stores more long-term memory information than other models, so we performed a memory analysis test to confirm our conjecture. We train a model with a conventional stacked structure with ST-LSTM [10] unit, and a model with ST-LSTM unit combined with the KeyMemoryRNN framework. We freeze the trained model as a network that cannot continue training and only provides memory output, take the 20 frames sequence as the input of the trained model and output the final memory state. (For key memory RNN, output its memory state and key memory state.) The output is used as the initial memory state or key memory state of the new model, the input of the new model is set to 0, and the training of the new model takes the input of the trained model as the target to calculate the L2 loss. Because the trained model weight is frozen and the input of the new model is 0, the new model can only restore the information of the whole sequence from the output memory state of the trained model. Therefore, in this experiment, the restored result can denote the storage ability of the model for long-term memory. The results of the memory analysis test are shown in the Fig. 8. The restored results of the conventional stacked structure are gradually blurred and deviate from the correct motion trajectory, while the restored results of the KeyMemoryRNN framework can retain the basic outline of human and clearly show the correct motion trajectory. At the same time, we set up a control group model with conventional stacking structure. We sample the original input sequence and take an input frame every other time step node. Therefore, the memory transitions of the control group model only pass through half of the time step nodes of the conventional stacked structure model. It can be seen from the experimental results that the less nodes the memory transition passes through, the more long-term memory can be stored in the memory flow.

FIGURE 8. A memory analysis test example on the KTH action test set, the display interval is 4. (a) The input sequence. (b) The sequence recovered through the conventional stacking structure network with 1 input interval. (c) The sequence recovered through the conventional stacking structure network with 2 input interval. (d) The sequence recovered through KeyMemoryRNN.

5) ABLATION STUDY
We conducted an ablation experiment to prove the contribution of each module of KeyMemoryRNN in the long-term prediction task. The experimental results are shown in the Table 6. It can be seen that the second stage KeyMemoryLSTM greatly improves the results of long-term prediction because it produces shorter information flow.

TABLE 6. Ablation study on the KTH action test set: analyze the effect of KeyTranslate module and two-stage KeyMemoryLSTM.

| Model                  | PSNR | ΔPSNR | SSIM  | ΔSSIM |
|------------------------|------|-------|-------|-------|
| ST-LSTM(BaseModel)     | 27.98| -     | 0.844 | -     |
| +KeyTranslate Module   | 28.42| 1.5%  | 0.849 | 0.5%  |
| +One Stage KeyMemoryLSTM| 28.56| 2.0%  | 0.855 | 0.7%  |
| +Second Stage KeyMemoryLSTM | 30.79| 10.0% | 0.875 | 3.6%  |
C. RADAR ECHO

Radar echo data is important for precipitation forecasting. The accuracy of radar echo extrapolation prediction is of great significance to meteorological work. This benchmark radar echo dataset uses the radar echo data collected by Hunan Meteorological Bureau. The gray images of $64 \times 64 \times 1$ are collected every 6 minutes, and 40 consecutive radar echo images are taken as a sequence, of which 20 frames are input frames and 20 frames are prediction frames. Because radar echo will move, generate or dissipate due to complex atmospheric physics, long-term trend information and semantic information guidance based on long-term memory is very useful for radar echo prediction.

1) DATASETS AND SETUP

To prevent data leakage, we divide the dataset into 24382 training sequences and 3420 test sequences under the condition that there is no intersection between the training set and the test set. About 5% of the data in the test sequence are radar echo data from different regions from the training set data to test the generalization ability of the model.

2) MAIN RESULT

We compare KeyMemoryRNN with three benchmark networks: PredRNN [10], MIM [12], MotionRNN [23]. MIM [12] and MotionRNN [23] are networks specifically used for radar echo extrapolation tasks. In particular, MotionRNN is the latest radar echo extrapolation network of CVPR in 2021. We use evaluation metrics that were used by previous methods: SSIM and Critical Success Index (CSI). CSI is a commonly used metrics in meteorology, which can indicates the overall accuracy of forecasting. It is defined as (9).

$$CSI = \frac{Hits}{Hits + Misses + Falsealarms}$$ (9)

Hits correspond to the true positive, misses correspond to the false positive, and falsealarms correspond to the false negative. Higher CSI indicates higher forecasting accuracy. For the radar echo intensity, we convert the pixel value to dBZ, and calculate the critical success index (CSI) with 30 dBZ, 40 dBZ, and 50 dBZ as thresholds respectively. The forecasting results are shown in the Table 7, KeyMemoryRNN achieves the best results in SSIM and CSI of three thresholds, which shows that our model base on MotionRNN can play an excellent forecasting results for precipitation forecasting of different precipitation intensities.

3) QUALITATIVE COMPARISON

As shown in the Fig. 9, the deformation of the echo area prediction by the PredRNN [10] and MIM [10] deviates from the change of the echo area in the real situation. Although the coverage of the echo area prediction by MotionRNN [23] is roughly close to the real situation, there are still deficiencies in detail and clarity. After using the KeyMemoryRNN framework, due to the motion context and long-term information guidance, it makes the prediction results richer in details, realizes better prediction for small areas and high echo intensity, and improves the overall clarity to a certain extent.

V. CONCLUSION

We propose a prediction framework that can be flexibly applied to RNN based prediction networks. It generates additional effective memory information flow through a series of self-attention mechanisms and calculation of new recurrent units, so as to enhance the ability of the network to use non-adjacent information. Experimental results on multiple datasets show that our prediction framework can effectively optimize the prediction results of prediction networks. At the same time, we explored the impact of the number of information flow nodes on the efficiency of network use of this information, trying to explain the role of different states in the network.

REFERENCES

[1] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. W. Cottrell, “A dual-stage attention-based recurrent neural network for time series prediction,” in Proc. 26th Int. Joint Conf. Artif. Intell., Aug. 2017, pp. 2627–2633.
[2] J. Hu and W. Zheng, “A deep learning model to effectively capture mutation information in multivariate time series prediction,” Knowl.-Based Syst., vol. 203, Sep. 2020, Art. no. 106139.
[3] X. Shi, Z. Chen, H. Wang, D. Yeung, W. Wong, and W. Woo, “Convolutional LSTM network: A machine learning approach for precipitation now-casting,” in Proc.: Neural Inf. Process. Syst. (NIPS), 2015, pp. 802–810.
[4] J. Zhang, T. Zheng, and D. Qi, “Deep spatio-temporal residual networks for citywide crowd flows prediction,” in Proc. AAAI, 2017, pp. 1655–1661.
SHENGCHUN WANG received the B.S. degree in computer software from Tianjin University, Tianjin, China, and the M.S. and Ph.D. degrees in operations research and control from Hunan Normal University, Changsha, China. From 2001 to 2011, he was a Lecturer with the Computer Teaching Department, Hunan Normal University. From 2011 to 2021, he was a Professor with the Software Engineering Department, College of Information Science and Engineering, Hunan Normal University. He is the author of two books, more than 20 articles, and more than ten inventions. His research interests include artificial intelligence, deep learning, and computer vision.

XIANG LIN received the B.S. degree in electrical engineering and automation from Huqiao University, Xiamen, China, in 2015, where he is currently pursuing the M.S. degree from Hunan Normal University. His research interests include spatiotemporal prediction, computer vision, and deep learning.

HUIJIE ZHU received the B.S. degree in automobile service engineering from Wuhan University of Technology, Wuhan, China, in 2009, and the Ph.D. degree from PLA University of Science and Technology, Nanjing, China, in 2015. After graduation, he was an Engineer with the Science and Technology on Near-Surface Detection Laboratory, Wuxi, China, until now. His current research interests include deep learning, machine vision, and application in control and meteorology.