Differential Recurrent Neural Networks for Action Recognition

Vivek Veeriah, Naifan Zhuang, Guo-Jun Qi*
University of Central Florida
4000 Central Florida Blvd, Orlando, FL 32816
{vivekveeriah,zhuangnaifan}@knights.ucf.edu, guojun.qi@ucf.edu

Abstract

The long short-term memory (LSTM) neural network is capable of processing complex sequential information since it utilizes special gating schemes for learning representations from long input sequences. It has the potential to model any sequential time-series data, where the current hidden state has to be considered in the context of the past hidden states. This property makes LSTM an ideal choice to learn the complex dynamics of various actions. Unfortunately, the conventional LSTMs do not consider the impact of spatio-temporal dynamics corresponding to the given salient motion patterns, when they gate the information that ought to be memorized through time. To address this problem, we propose a differential gating scheme for the LSTM neural network, which emphasizes on the change in information gain caused by the salient motions between the successive frames. This change in information gain is quantified by Derivative of States (DoS), and thus the proposed LSTM model is termed as differential Recurrent Neural Network (dRNN). We demonstrate the effectiveness of the proposed model by automatically recognizing actions from the real-world 2D and 3D human action datasets. Our study is one of the first works towards demonstrating the potential of learning complex time-series representations via high-order derivatives of states.

1. Introduction

Recently, Recurrent Neural Networks (RNNs) [28], especially Long Short-Term Memory (LSTM) model [14], have gained significant attention in solving many challenging problems involving time-series data, such as action recognition [12, 8, 13], multilingual machine translation [30, 4], multimodal translation between videos and sentences [34], robot control [21] and time-series alignment [32]. In these applications, learning an appropriate representation of sequences is an important step in achieving artificial intelligence.

Compared with many existing spatio-temporal features [17, 29, 5, 24] from the time-series data, RNNs use either a hidden layer [28] or a memory cell [14] to learn the time-evolving states which models the underlying dynamics of the input sequence. For example, [2, 8] have used LSTMs to model the video sequences to learn their long short-term dynamics. In contrast to the conventional RNN, the major component of LSTM is the memory cell which is modulated by three gates - input, output and forget gates. These gates determine the amount of dynamic information entering/leaving the memory cell. The memory cell has a set of internal states, which store the information obtained over time. In this context, these internal states constitute a representation of an input sequence learned over time.

In many recent works, the LSTMs have shown tremendous potential in action recognition tasks [2, 13, 8]. The existing LSTM model represents a video by integrating over time all the available information from each frame. However, we observed that for an action recognition task, not all frames contain salient spatio-temporal information which are discriminative to different classes of actions. Many frames contain non-salient motions which are irrelevant to the performed action.

This inspired us to develop a new family of LSTM model that automatically learns the dynamic saliency of the actions performed. The conventional LSTM fails to capture the salient dynamic patterns, since the gate units do not explicitly consider whether a frame contains salient motion information when they modulate the input and output of the memory cells. Thus the model is insensitive to the dynamic evolution of the hidden states given the input video sequences. To address this problem, we propose the differential RNN (dRNN) model that learns these salient spatio-temporal representations of actions.

Specifically, dRNN models the dynamics of actions by computing different-orders of Derivative of State (DoS) that are sensitive to the spatio-temporal structure of input sequence. In other words, depending on the DoS, the gate units can learn the appropriate information that should be

[*] Corresponding author

required to model the dynamic evolution of actions. To train the dRNN model, we use truncated Back Propagation algorithm to prevent the exploding or diminishing errors through time [14]. In particular, we follow the rule that the errors propagated through the connections to those DoS nodes would be truncated once they leave the current memory cell.

Finally, we demonstrate that the dRNNs can achieve the state-of-the-art performance on both 2D and 3D action recognition datasets. Specifically, dRNNs outperform the existing LSTM model on these action recognition tasks, consistently achieving the better performance with the same input sequences. On the other hand, when compared with the other algorithms based on special assumptions about spatio-temporal structure of actions, the proposed general-purpose dRNN model can still reach competitive performance.

The remainder of this paper is organized as follows. In the next section 2, we review several related work to the action recognition problem. The background and details of RNNs and LSTMs are reviewed in section 3. Section 4 presents the proposed differential RNNs (dRNNs). The experimental results are presented in section 5. Finally, we conclude and discuss the future work related to dRNNs in section 6.

2. Related Work

Action recognition has been a long-standing research problem in computer vision and pattern recognition community, which aims to enable a computer to automatically understand the activities performed by people interacting with the surrounding environment and with each other [23]. This is a challenging problem due to the huge intra-class variance of actions performed by different actors at various speeds, in diverse environments (e.g., camera angles, lighting conditions, and cluttered background).

To address this problem, many robust spatio-temporal representations have been constructed. For example, HOG3D [17] uses the histogram of 3D gradient orientations to represent the motion structure over the frame sequences; 3D-SIFT [29] extends the popular SIFT descriptor to characterize the scale-invariant spatio-temporal structure for 3D video volume; actionlet ensemble [35] utilizes a robust approach to model the discriminative features from 3D positions of the tracked joints captured by depth cameras.

Although these descriptors have achieved remarkable success, they are usually engineered to model a specific spatio-temporal structure in an ad-hoc fashion. Recently, the huge success of deep networks in image classification [18] and speech recognition [11] has inspired many researchers to apply the deep neural networks, such as 3D Convolutional Neural Networks (3DCNNs) [3] and Recurrent Neural Networks (RNNs) [2, 8], to action recognition. In particular, [3] developed a 3D convolutional neural network that extends the conventional CNN by taking space-time volume as input. On the contrary, [2, 8] used LSTMs to represent the video sequences directly, and modeled the dynamic evolution of the action states via a sequence of memory cells.

Meanwhile, the existing approaches combine deep neural networks with spatio-temporal descriptors, achieving competitive performance. For example, in [3], a LSTM model takes a sequence of Harris3D and 3DCNN descriptors extracted from each frame as input, and the result on KTH dataset has shown the state-of-the-art performance [3].

3. Background

In this section, we briefly review the recurrent neural network as well as its variant, long short-term memory model. Readers who are familiar with them might skip to the next section directly.

3.1. Recurrent Neural Networks

Traditional recurrent neural networks (RNNs) [28] model the dynamics of an input sequence of frames $\{\mathbf{x}_t \in \mathbb{R}^n | t = 1, \cdots, T\}$ through a sequence of hidden states $\{\mathbf{s}_t \in \mathbb{R}^m | t = 1, \cdots, T\}$ thereby learning the spatio-temporal structure of the input sequence. For example, a classical RNN model uses the following recurrent equation

$$\mathbf{s}_t = \tanh(\mathbf{W}_{ss}\mathbf{s}_{t-1} + \mathbf{W}_{sx}\mathbf{x}_t + \mathbf{b}_s) \quad (1)$$

to model the hidden state $\mathbf{s}_t$ at time t by combining the information from the current input $\mathbf{x}_t$ and the past hidden state $\mathbf{s}_{t-1}$, where the hyperbolic tangent $\tanh(\cdot)$ is an activation function with range $[-1, 1]$, $\mathbf{W}_{sx}$ and $\mathbf{W}_{ss}$ are two mapping matrices to the hidden state, and $\mathbf{b}_s$ is the bias vector.

The hidden state can be mapped to an output sequence $\{\mathbf{z}_t \in \mathbb{R}^k | t = 1, \cdots, T\}$ as

$$\mathbf{z}_t = \tanh(\mathbf{W}_{zs}\mathbf{s}_t + \mathbf{b}_z) \quad (2)$$

where each $\mathbf{z}_t$ represents an 1-of-k encoding of the confidence scores on k classes of actions. Then, this output vector can be transformed to a vector of probabilities $\mathbf{y}_t$ by softmax function as

$$y_{t,c} = \frac{\exp(z_{t,c})}{\sum\limits_{l=1}^{k} \exp(z_{t,l})}, \quad (3)$$

with each entry $y_{t,c}$ being the probability of frame t belonging to class $c \in \{1, \cdots, k\}$.

3.2. Long Short-Term Memory

The above classical RNN is limited in learning the long-term representation of video sequences, due to the exponential decay in retaining the context information of video frames [14]. To overcome this limitation, Long Short-Term Memory (LSTM) [14], a variant of RNN, has been designed to learn the long-range dependency between the output label and the input frame, which has achieved competitive performance on action recognition task [2][3].

In particular, LSTMs are composed of a sequence of memory cells, each containing an internal state $\mathbf{s}_t$ storing the memory of the input sequence up to time t. To store the memory with respect to a context in long period of time, three types of gate units are incorporated into LSTMs to control what information would enter and leave the memory cell over time [14]. These gate units are activated by a nonlinear function of input/output sequences as well as internal states, making them powerful enough to model the dynamically changing context given that the human actions evolve at various time scales.

Formally, a LSTM cell has the following gates:

1. The input gate $\mathbf{i}_t$ controls the degree to which the input information would enter the memory cell to influence its internal state $\mathbf{s}_t$ at time t. The activation of this gate has the following recurrent form

$$\mathbf{i}_t = \sigma(\mathbf{W}_{is}\mathbf{s}_{t-1} + \mathbf{W}_{iz}\mathbf{z}_{t-1} + \mathbf{W}_{ix}\mathbf{x}_t + \mathbf{b}_i)$$

where the sigmoid $\sigma(\cdot)$ is an activation function with the range $[0, 1]$, with 0 meaning the gate is closed and 1 meaning the gate is completely open; $\mathbf{W}_{i*}$ are the mapping matrices and $\mathbf{b}_i$ is the bias vector.

2. The forget gate $\mathbf{f}_t$ modulates the previous state $\mathbf{s}_{t-1}$ to control its contribution to the current state (c.f. Eq(4)). It is defined as

$$\mathbf{f}_t = \sigma(\mathbf{W}_{fs}\mathbf{s}_{t-1} + \mathbf{W}_{fz}\mathbf{z}_{t-1} + \mathbf{W}_{fx}\mathbf{x}_t + \mathbf{b}_f)$$

with the mapping matrices $\mathbf{W}_{f*}$ and the bias vector $\mathbf{b}_f$.

With the input/forget gate units, the internal state $\mathbf{s}_t$ of each memory cell can be updated below:

$$\mathbf{s}_t = \mathbf{f}_t \odot \mathbf{s}_{t-1} + \mathbf{i}_t \odot \mathbf{s}_{t-\frac{1}{2}} \tag{4}$$

where we define the pre-state $\mathbf{s}_{t-\frac{1}{2}}$ as

$$\mathbf{s}_{t-\frac{1}{2}} = \tanh(\mathbf{W}_{sz}\mathbf{z}_{t-1} + \mathbf{W}_{sx}\mathbf{x}_t + \mathbf{b}_s).$$

The pre-state can be considered as an intermediate state between two consecutive frames, aggregating the information from the last output $\mathbf{z}_{t-1}$ and the current input $\mathbf{x}_t$. Then it is combined with the gated information from the previous state $\mathbf{s}_{t-1}$ to update the current state $\mathbf{s}_t$ as in Eq. (4).

3. The output gate $\mathbf{o}_t$:

$$\mathbf{o}_t = \sigma(\mathbf{W}_{os}\mathbf{s}_t + \mathbf{W}_{oz}\mathbf{z}_{t-1} + \mathbf{W}_{ox}\mathbf{x}_t + \mathbf{b}_o).$$

It gates the information output from a memory cell which would influence the future states of LSTM cells. Then the output of a memory cell can be expressed as

$$\mathbf{z}_t = \mathbf{o}_t \odot \tanh(\mathbf{W}_{zs}\mathbf{s}_t + \mathbf{b}_z) \tag{5}$$

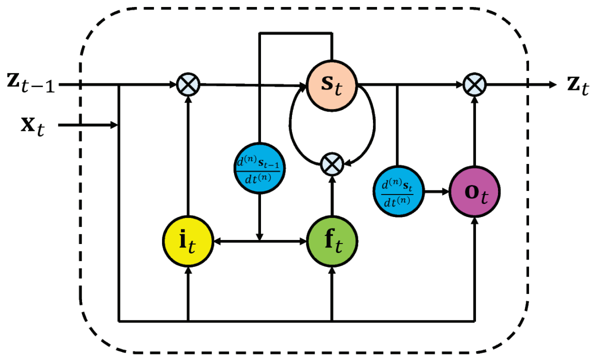


Figure 1. Architecture of the proposed dRNN model at time t. In the memory cell, the input gate $\mathbf{i}_t$ and the forget gate $\mathbf{f}_t$ are controlled by DoS $\frac{d^{(n)}\mathbf{s}_{t-1}}{dt^{(n)}}$ at $t-1$, and the output gate $\mathbf{o}_t$ is controlled by the DoS $\frac{d^{(n)}\mathbf{s}_t}{dt^{(n)}}$ at t.

where $\odot$ stands for element-wise product.

In brief, LSTM proceeds by iteratively applying Eq. (4) and Eq. (5) to update the state $\mathbf{s}_t$ and output $\mathbf{z}_t$. In this process, the forget gate, output gate and input gate play a critical role in controlling the information entering and leaving the memory cell. More details about LSTMs can be found in [14].

4. Differential Recurrent Neural Networks

For an action recognition task, not all video frames contain salient patterns to discriminate between different classes of actions. Many spatio-temporal descriptors, such as 3D-SIFT [29] and HoGHoF [19], have been proposed to localize and encode the salient spatio-temporal points. They detect and encode the spatio-temporal points related to salient motions of the objects in video frames, revealing the important dynamics of actions.

In this paper, we develop a novel LSTM model to automatically learn the dynamics of actions, by detecting and integrating the salient spatio-temporal sequences. The conventional LSTMs might fail to capture these salient dynamic patterns, because the gate units do not *explicitly* consider the impact of dynamic structures present in input sequences. This makes the model inadequate to learn the evolution of action states. To address this problem, we propose a differential RNN (dRNN) model to learn and integrate the dynamics of actions.

The proposed dRNN model is based on the observation that the internal state of each memory cell contains the accumulated information about the spatio-temporal structure, i.e., it is a long short-term representation of an input sequence. So the Derivative of States (DoS) $\frac{d\mathbf{s}_t}{dt}$ quantifies the change of information at each time t. In other words, a large magnitude of DoS is an indicator of a salient spatio-temporal structure containing the informative dynamics caused by an abrupt change of action state. In this case, the gate units should allow more information to enter the memory cell to update its internal state. Otherwise, when the magnitude of DoS is small, the incoming information should be gated out of the memory cell so the internal state would not be affected by the current input. Therefore, DoS should be used as a factor to gate the information flow into and out of the internal state of memory cell over time.

Moreover, we can involve higher-orders of DoS $\{\frac{d^n\mathbf{s}_t}{dt^n}|n \geq 2\}$ to detect and capture the higher-order dynamic patterns for the dRNN model. For example, when modeling a moving object in a video, the first-order DoS captures the velocity while the second-order captures its acceleration. These different orders of DoS will enable dRNN to better represent the dynamic evolution of action states.

Figure 1 illustrates the architecture of the proposed dRNN model. Formally, we have the following recurrent equations to control the gate units with the DoS up to order N:

$$\mathbf{i}_t = \sigma(\sum_{n=0}^{N} \mathbf{W}_{id}^{(n)} \frac{d^{(n)}\mathbf{s}_{t-1}}{dt^{(n)}} + \mathbf{W}_{iz}\mathbf{z}_{t-1} + \mathbf{W}_{ix}\mathbf{x}_t + \mathbf{b}_i) \quad (6)$$

$$\mathbf{f}_t = \sigma(\sum_{n=0}^{N} \mathbf{W}_{fd}^{(n)} \frac{d^{(n)}\mathbf{s}_{t-1}}{dt^{(n)}} + \mathbf{W}_{fz}\mathbf{z}_{t-1} + \mathbf{W}_{fx}\mathbf{x}_t + \mathbf{b}_f) \quad (7)$$

$$\mathbf{o}_t = \sigma(\sum_{n=0}^{N} \mathbf{W}_{od}^{(n)} \frac{d^{(n)}\mathbf{s}_{t}}{dt^{(n)}} + \mathbf{W}_{oz}\mathbf{z}_{t-1} + \mathbf{W}_{ox}\mathbf{x}_t + \mathbf{b}_o) \quad (8)$$

where $\frac{d^{(n)}\mathbf{s}_{t-1}}{dt^{(n)}}$ is the n-order DoS, and $W_{*d}^{(n)}$ are the corresponding mapping matrices.

Finally, it is worth pointing out that we do not use the derivative of inputs as a measurement of salient dynamics to control the gate units. The derivative of inputs would amplify the unwanted noises which are often contained in the input sequence. This derivative of inputs only represent the local dynamic saliency, in contrast to the long short-term change in the information gained over time. For example, a motion may have been performed several frames ago. Using derivative of inputs would treat it as a novel salient motion, even though it has already been stored by LSTM. On the contrary, DoS does not have this problem, because the internal state $\mathbf{s}_t$ has long-term memory of the past motion pattern, even though the same motion had previously occurred.

4.1. Discretized Model

Since the model is defined in the discrete-time domain, the first-order derivative $\frac{d\mathbf{s}_t}{dt}$, as the velocity of information change, can be discretized as the difference of states

$$\mathbf{v}_t \triangleq \frac{d\mathbf{s}_t}{dt} \doteq \mathbf{s}_t - \mathbf{s}_{t-1} \quad (9)$$

for its simplicity [9].

Similarly, we can consider the second order of DoS as the acceleration of information change can be discretized as

$$\mathbf{a}_t \triangleq \frac{d^2\mathbf{s}_t}{dt^2} \doteq \mathbf{v}_t - \mathbf{v}_{t-1} = \mathbf{s}_t - 2\mathbf{s}_{t-1} + \mathbf{s}_{t-2} \quad (10)$$

In this paper, we only consider the first two orders of DoS. Higher orders can be derived in a similar way.

With the above recurrent equations, at time step t, the dRNN model proceeds in the following order.

- Compute input gate activation $\mathbf{i}_t$ and forget gate activation $\mathbf{f}_t$ by Eq. (6) and Eq. (7);
- Update state $\mathbf{s}_t$ with $\mathbf{i}_t$ and $\mathbf{f}_t$ by Eq. (4);
- Compute discretized DoS $\{\frac{d^{(n)}\mathbf{s}_t}{dt^{(n)}}|n = 1, \cdots, N\}$ up to order N at time t, e.g. Eq. (9) and Eq. (10);
- Compute output gate $\mathbf{o}_t$ by Eq. (8);
- Output $\mathbf{z}_t$ gated by $\mathbf{o}_t$ from memory cell by Eq. (5);
- (Optional) Output the label $\mathbf{y}_t$ by applying the softmax to $\mathbf{z}_t$ by Eq. (3).

Now it is obvious that this model is termed differential RNNs (dRNNs) because of the central role of derivatives of states in detecting and capturing the salient spatio-temporal structures.

4.2. Learning Algorithm

To learn the model parameters of dRNNs, we define a loss function to measure the deviation between the target class c_t and $\mathbf{y}_t$ at time t:

$$\ell(\mathbf{y}_t, c_t) = -\log y_{t,c_t}.$$

For an action recognition task, the label of action is often given at the video level. Since LSTMs have the ability to memorize the content of an entire sequence, the last memory cell of LSTMs ought to contain all the necessary information for action recognition. Thus, for a sequence of length T, and a given training label c, the dRNNs can be trained by minimizing the loss at time T, i.e., $\ell(\mathbf{y}_T, c) = -\log y_{T,c}$.

Otherwise, if an individual label c_t is given to each frame t in the sequence, we can minimize the cumulative loss over the sequence:

$$\sum_{t=1}^{T} \ell(\mathbf{y}_t, c_t).$$

Both types of loss functions can be minimized by Back Propagation Through Time (BPTT) [6], which unfolds a dRNN model over several time steps and then runs the back propagation algorithm to train the model. To prevent the back-propagated errors from decaying or exploding exponentially, LSTMs usually use truncated BPTT [14]. The idea is rather simple: once the back-propagated error leaves the memory cell or gates, it will not be allowed to enter the memory cell again. In the proposed dRNNs, we also use the truncated errors to learn the model parameters. In particular, we do not allow the errors to re-enter the memory cell once they leave it through the DoS nodes $\mathbf{v}_t$ and $\mathbf{a}_t$.

Formally, we assume the following truncated derivatives of gate activations:

$$\frac{\partial \mathbf{i}_t}{\partial \mathbf{v}_{t-1}} \doteq \mathbf{0}, \quad \frac{\partial \mathbf{f}_t}{\partial \mathbf{v}_{t-1}} \doteq \mathbf{0}, \quad \frac{\partial \mathbf{o}_t}{\partial \mathbf{v}_t} \doteq \mathbf{0}$$

and

$$\frac{\partial \mathbf{i}_t}{\partial \mathbf{a}_{t-1}} \doteq \mathbf{0}, \quad \frac{\partial \mathbf{f}_t}{\partial \mathbf{a}_{t-1}} \doteq \mathbf{0}, \quad \frac{\partial \mathbf{o}_t}{\partial \mathbf{a}_t} \doteq \mathbf{0}$$

where $\doteq$ stands for the truncated derivative. The details about the implementation of truncated BPTT can be found in [14].

5. Experiments and Results

We compare the performance of the proposed method with the state-of-the-art LSTM and non-LSTM methods present in existing literature on both 2D and 3D human action datasets.

5.1. Datasets

The proposed method is evaluated on the KTH 2D action recognition dataset, as well as MSR Action3D dataset.

KTH dataset. We choose KTH dataset [27] for it is a *de facto* benchmark for evaluating action recognition algorithms. This makes it possible to directly compare with the other algorithms. There are two KTH datasets - KTH-1 and KTH-2, which both consist of six action classes: walking, jogging, running, boxing, hand-waving and hand-clapping. The actions are performed several times by 25 subjects in four different scenarios: outdoors, outdoors with scale variation, outdoors with different clothes and indoors. The sequences are captured over homogeneous background with a static camera recording 25 frames per second. Each video has a resolution of 160×120, and lasts for about 4 seconds on KTH-1 dataset and for about a second for KTH-2 dataset. There are 599 videos in the KTH-1 dataset and $2,391$ video sequences in the KTH-2 dataset.

MSR Action3D dataset. The MSR Action3D dataset [20] consists of 567 depth map sequences performed by 10 subjects using a depth sensor similar to the Kinect device. The resolution of each video is 320×240 and there are 20 action classes where each subject performs each action two or three times. The actions are chosen in the context of gaming. They cover a variety of movements related to arms, legs, torso etc. This dataset has a lot of noise in the joint locations of the skeleton as well as high intra-class variations and inter-class similarities, making it a challenging dataset for evaluation among the existing 3D datasets. We follow a similar experiment setting from [35], where half of the subjects are used for training and the other half are used for testing. This setting is much more challenging than the subset one used in [20], because all actions are evaluated together and the chance of confusion is much higher.

5.2. Feature Extraction

We are using densely sampled HOG3D features to represent each frame of video sequences from the KTH dataset. Specifically, we uniformly divide the 3D video volumes into a dense grid, and extract the descriptors from each cell of the grid. The parameters for HOG3D are the same as the one used in [17]. We extract HOG3D features using the standard KTH optimized dense sampling parameters mentioned on the authors' webpage [1]. The size of the descriptor was 1000 per cell of grid, and there are 56 such cells in each frame, yielding a $56,000$ dimensional feature vector per frame. We apply PCA to reduce the dimension to 450, retaining 97% of energy among the principal components, to construct a compact input into the dRNN model.

For 3D action dataset, MSR Action3D, a depth sensor like Kinect provides an estimate of 3D joint coordinates of body skeleton, and the following features were extracted to represent MSR Action3D depth sequences – (1) Position: 3D coordinates of the 20 joints obtained from the skeleton map. These 3D coordinates were then concatenated resulting in a 60 dimensional feature per frame; (2) Angle: normalized pair-wise angles. The normalized pair-wise angles were obtained from 18 joints of the skeleton map. The two feet joints were not included. This resulted in a 136 dimensional feature vector per frame; (3) Offset: offset of the 3D joint positions between the current and the previous frame [38]. These offset features were also computed using the 18 joints from the skeleton map resulting in a 54 dimensional feature per frame; (4) Velocity: histogram of the velocity components obtained from point cloud. This feature was computed using the 18 joints as in the previous cases resulting in a 162 dimensional feature per frame; (5) Pairwise joint distances: The 3D coordinates obtained from the skeleton map were used to compute pairwise joint distances with the centre of the skeleton map resulting in a 60 dimensional feature vector per frame. For the following experiments, these five different features were concatenated to result in a 583 dimensional feature vector per frame.

5.3. Architecture and Training

The architectures of the dRNN models trained on the two datasets are shown in Table 1. For the sake of fair comparison, we adopt the same architecture for the dRNN models of both orders on two datasets. We can see that the number of memory cell units is smaller than the input units on both

Dataset	KTH	MSR Action3D
Input Units	450	583
Memory Cell State Units	300	400
Output Units	6	20

Table 1. Architecture of the dRNN model used on two datasets. Each row shows the number of units in each component. For the sake of fair comparison, we adopt the same architecture for the dRNN models of both orders on two datasets.

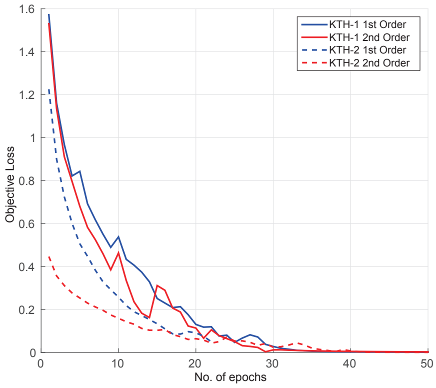


Figure 2. Objective loss curve over training epochs on the KTH dataset.

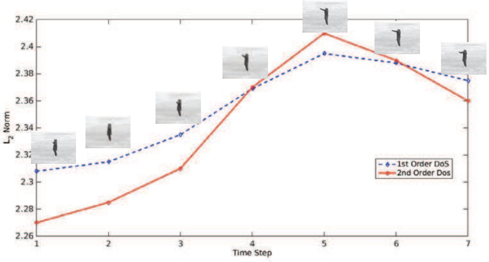


Figure 3. The curve of the 1st and 2nd orders of DoS over an example of sequence for the action "boxing." Note that the local maximum of DoS corresponds to the change from punching to relaxing.

datasets. This can be interpreted as follows. The sequence of an action video often forms a continuous trajectory embedded in a low-dimensional manifold of the input space. Thus, a lower-dimension state space suffices to capture the dynamics of such a trajectory.

We plot the learning curve for training the model on KTH dataset in Figure 2. The learning rate of BPTT algorithm is set to 0.0001. The figure shows that the objective loss continuously decreases over 50 epochs. Usually after 40 epochs, the training of dRNN model begins to converge.

5.4. Results on KTH Dataset

There are several different evaluation protocols used on KTH dataset in literature. This can result in as large as 9% differences in performance across different experiment protocols as reported in [10]. For the sake of fair comparison, we follow the cross-validation protocol [3], in which we randomly select 16 subjects to train the model, and test over the remaining 9 subjects. The performance is reported by the average across five such trails.

First, we compare the dRNN model with the conventional LSTM model in Table 2. Here we report the cross-validation accuracy on both KTH-1 and KTH-2 datasets. In addition, Figure 5 shows the confusion matrix obtained by the 2-order dRNN model on KTH-1 dataset. This confusion matrix is computed by averaging over five trials in the above cross-validation protocol. The performance of conventional LSTM has been reported in literature [13, 3]. We note that these reported accuracies often vary with different types of features. Thus, a fair comparison between different models can only be made with the same type of input feature.

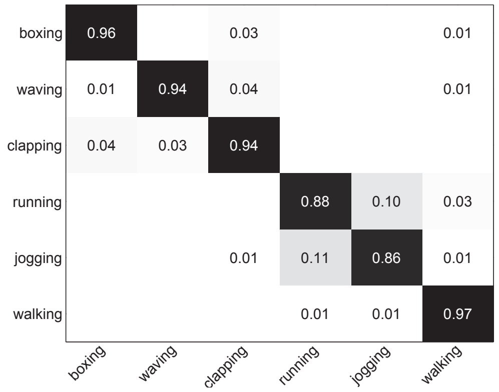
For the dRNN model, we report the accuracy with up to the 2-order of DoS. The table shows that with the same HOG3D feature, the proposed dRNN models outperform the conventional LSTM model, and the 2-order dRNN yields a better accuracy than its 1-order counterpart. Although higher-order of DoS might improve the accuracy further, we do not report the result since it becomes trivial to simply add more orders of DoS into dRNN, and the improved performance might not compensate for the increased computational cost. Moreover, with an increased order of DoS, more model parameters would have to be learned with the limited training examples. This tends to cause overfitting problem, making the performance stop improving or even begin to degenerate after the order of DoS reaches a certain number. Therefore, for most of practical applications, the first two orders of dRNN should be sufficient.

Baccouche et al. [3] reported an accuracy of 94.39% and 92.17% on KTH-1 and KTH-2 data sets, respectively. But it is worth noting that they used a combination of 3DCNN and LSTM, where 3DCNN plays the crucial role in reaching such performance. Actually, 3DCNN model alone can reach an accuracy of 91.04% and 89.40% on KTH-1 and KTH-2 data sets as reported in [3]. On the contrary, they reported that the LSTM with Harris3D feature only achieved 87.78% on KTH-2, as compared with 92.12% accuracy obtained by 2-order dRNN with HOG3D feature. In Table 2, under a fair comparison with the same feature, the dRNN models of both orders outperform their LSTM counterpart with the same HOG3D feature.

In Figure 3, to support our motivation of learning LSTM representations based on the dynamic change of states evolving over frames, we illustrate some example frames of "boxing" action versus the curve of L_2-norm of 1-order

LSTM	Clapping	Clapping	Clapping	Walking	Boxing	Boxing	Boxing
1st Order DoS	Clapping	Walking	Clapping	Boxing	Boxing	Boxing	Boxing
2nd Order DoS	Clapping	Clapping	Boxing	Boxing	Boxing	Boxing	Boxing

Figure 4. Frame-by-frame prediction of action category over time.



	boxing	waving	clapping	running	jogging	walking
boxing	0.96		0.03			0.01
waving	0.01	0.94	0.04			0.01
clapping	0.04	0.03	0.94			
running				0.88	0.10	0.03
jogging			0.01	0.11	0.86	0.01
walking				0.01	0.01	0.97

Figure 5. Confusion Matrix on the KTH-1 dataset obtained by the 2-Order dRNN model.

and 2-order DoS on KTH dataset. It shows the change from “punching” to “relaxing” at the local maximum of DoS, showing the ability of the dRNN model to capture the salient dynamics for the action. We also illustrate the predictions over time in Figure 4. From the result, we found that as time evolves, the proposed dRNNs are faster in learning the salient dynamics for predicting the correct action category than the LSTMs. Moreover, the 2nd order of DoS is better than the 1st order of DoS in learning the salient features.

We also show the performance of the other non-LSTM state-of-the-art approaches in Table 3. Many of these compared algorithms focus on the action recognition problem, relying on the special assumptions about the spatio-temporal structure of actions. They might not be applicable to model the other type of sequences which do not satisfy these assumptions. In contrast, the proposed dRNN model is a general-purpose model, not being tailored to specific type of action sequences. This also makes it competent on 3D action recognition task as we will show below.

5.5. Results on MSR Action3D Dataset

Table 4 compares the results on MSR Action3D dataset, and Figure 6 shows the confusion matrix by the 2-order dRNN model. The results are obtained by following exactly the same experimental setting in [35], in which half of actor subjects are used for training and the rest are used for testing. This is in contrast to another evaluation protocol in literature [20] that splits across 20 action classes into three subsets and performs the evaluation within each individual subset. The evaluation protocol we adopt is more challenging because it is evaluated over all 20 action classes with no common subjects in training and test sets.

From the results, the dRNN models of both orders outperform the conventional LSTM algorithm with the same feature. Also, both dRNN models perform competitively

Dataset	Method	Accuracy
KTH-1	LSTM + HOF [13]	90.7
	LSTM + HOG3D	89.93
	1-order dRNN + HOG3D	**93.28**
	2-order dRNN + HOG3D	**93.96**
KTH-2	LSTM + Harris3D [3]	87.78
	LSTM + HOG3D	87.32
	1-order dRNN + HOG3D	**91.98**
	2-order dRNN + HOG3D	**92.12**

Table 2. Cross-validation accuracy over five trails obtained by the proposed dRNN model in comparison with the conventional LSTM model on KTH-1 and KTH-2 data sets.

Dataset	Method	Accuracy
KTH-1	Rodriguez et al. [25]	81.50
	Jhuang et al.[15]	91.70
	Schindler et al. [26]	92.70
	3DCNN [3]	91.04
KTH-2	Ji et al. [16]	90.20
	Taylor et al. [31]	90.0
	Laptev et al. [19]	91.80
	Dollar et al. [7]	81.20
	3DCNN [3]	89.40

Table 3. Cross-validation accuracy over five trials obtained by the other compared algorithms on KTH-1 and KTH-2 datasets.

Method	Accuracy
Actionlet Ensemble [35]	88.20
HON4D [22]	88.89
DCSF [36]	89.3
Lie Group [33]	89.48
LSTM	87.78
1-order dRNN	**91.40**
2-order dRNN	**92.03**

Table 4. Comparison of the dRNN model with the other algorithms on MSR Action3D dataset.

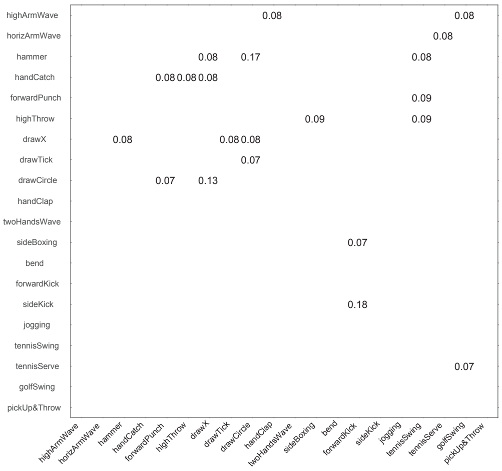


Figure 6. Confusion Matrix on the MSR Action3D dataset by the 2-Order dRNN model

as compared with the other algorithms. We notice that the Super Normal Vector (SNV) model [37] has reported an accuracy of 93.09% on MSR Action3D dataset. However, this model is based on a special assumption about the 3D geometric structure of the surfaces of depth image sequences. Thus, this approach is a very special model for solving 3D action recognition problem. This is contrary to dRNN as a general model without any specific assumptions on the dynamic structure of the video sequences.

In brief, through the experiments on both 2D and 3D human action datasets, we show the competitive performance of dRNN compared with both LSTM and non-LSTM models. This demonstrates its wide applicability in representing and modeling the dynamics of both 2D and 3D action sequences, irrespective of any assumptions on the structure of video sequences.

6. Conclusion and Future Work

In this paper, we present a new family of differential Recurrent Neural Networks (dRNNs) that extend Long Short-Term Memory (LSTM) structure by modeling the dynamics of states evolving over time. The new structure is better at learning the salient spatio-temporal structure. Its gate units are controlled by the different orders of derivatives of states, making the dRNN model more adequate for the representation of the long short-term dynamics of actions. Experiment results on both 2D and 3D human action datasets demonstrate the dRNN model outperforms the conventional LST-M model. Even in comparison with the other state-of-the-art approaches based on strong assumptions about the motion structure of actions being studied, the general-purpose dRNN model still demonstrates much competitive performance on both 2D and 3D datasets. In the future work, we will test dRNN in combination with more sophisticated input feature sequences to explore the specific motion structure of actions.

References

[1] http://lear.inrialpes.fr/people/klaeser/software_3d_video_descriptor. 5

[2] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt. Action classification in soccer videos with long short-term memory recurrent neural networks. In *Artificial Neural Networks–ICANN 2010*, pages 154–159. Springer, 2010. 1, 2, 3

[3] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt. Sequential deep learning for human action recognition. In *Human Behavior Understanding*, pages 29–39. Springer, 2011. 2, 3, 6, 7

[4] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. 1

[5] S. Chang, W. Han, J. Tang, G.-J. Qi, C. C. Aggarwal, and T. S. Huang. Heterogeneous network embedding via deep architectures. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 119–128. ACM, 2015. 1

[6] M. P. Cuéllar, M. Delgado, and M. Pegalajar. An application of non-linear programming to train recurrent neural networks in time series prediction problems. In *Enterprise Information Systems VII*, pages 95–102. Springer, 2006. 4

[7] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on*, pages 65–72. IEEE, 2005. 7

[8] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. *arXiv preprint arXiv:1411.4389*, 2014. 1, 2

[9] J. F. Epperson. *An Introduction to Numerical Methods and Analysis*. Wiley, 2nd edition, October 2013. 4

[10] Z. Gao, M.-Y. Chen, A. G. Hauptmann, and A. Cai. Comparing evaluation protocols on the kth dataset. In *Human Behavior Understanding*, pages 88–100. Springer, 2010. 6

[11] A. Graves and N. Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1764–1772, 2014. 2

[12] K. Gregor, I. Danihelka, A. Graves, and D. Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015. 1

[13] A. Grushin, D. D. Monner, J. A. Reggia, and A. Mishra. Robust human action recognition via long short-term memory. In *Neural Networks (IJCNN), The 2013 International Joint Conference on*, pages 1–8. IEEE, 2013. 1, 6, 7

[14] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 1, 2, 3, 5

[15] H. Jhuang, T. Serre, L. Wolf, and T. Poggio. A biologically inspired system for action recognition. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. Ieee, 2007. 7

[16] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(1):221–231, 2013. 7

[17] A. Klaser, M. Marszałek, and C. Schmid. A spatio-temporal descriptor based on 3d-gradients. In *BMVC 2008-19th British Machine Vision Conference*, pages 275–1. British Machine Vision Association, 2008. 1, 2, 5

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 2

[19] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008. 3, 7

[20] W. Li, Z. Zhang, and Z. Liu. Action recognition based on a bag of 3d points. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 9–14. IEEE, 2010. 5, 7

[21] H. Mayer, F. Gomez, D. Wierstra, I. Nagy, A. Knoll, and J. Schmidhuber. A system for robotic heart surgery that learns to tie knots using recurrent neural networks. *Advanced Robotics*, 22(13-14):1521–1537, 2008. 1

[22] O. Oreifej and Z. Liu. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 716–723. IEEE, 2013. 8

[23] R. Poppe. A survey on vision-based human action recognition. *Image and vision computing*, 28(6):976–990, 2010. 2

[24] G.-J. Qi, C. Aggarwal, Q. Tian, H. Ji, and T. S. Huang. Exploring context and content links in social media: A latent space method. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(5):850–862, 2012. 1

[25] M. Rodriguez, J. Ahmed, and M. Shah. Action mach: a spatio-temporal maximum average correlation height filter for action recognition. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008. 7

[26] K. Schindler and L. Van Gool. Action snippets: How many frames does human action recognition require? In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008. 7

[27] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local svm approach. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 32–36. IEEE, 2004. 5

[28] M. Schuster and K. K. Paliwal. Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on*, 45(11):2673–2681, 1997. 1, 2

[29] P. Scovanner, S. Ali, and M. Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th international conference on Multimedia*, pages 357–360. ACM, 2007. 1, 2, 3

[30] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112, 2014. 1

[31] G. W. Taylor, R. Fergus, Y. LeCun, and C. Bregler. Convolutional learning of spatio-temporal features. In *Computer Vision–ECCV 2010*, pages 140–153. Springer, 2010. 7

[32] V. Veeriah, R. Durvasula, and G.-J. Qi. Deep learning architecture with dynamically programmed layers for brain connectome prediction. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1205–1214. ACM, 2015. 1

[33] R. Vemulapalli, F. Arrate, and R. Chellappa. Human action recognition by representing 3d skeletons as points in a lie group. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 588–595. IEEE, 2014. 8

[34] S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. Mooney, and K. Saenko. Translating videos to natural language using deep recurrent neural networks. *arXiv preprint arXiv:1412.4729*, 2014. 1

[35] J. Wang, Z. Liu, Y. Wu, and J. Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1290–1297. IEEE, 2012. 2, 5, 7, 8

[36] L. Xia and J. Aggarwal. Spatio-temporal depth cuboid similarity feature for activity recognition using depth camera. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2834–2841. IEEE, 2013. 8

[37] X. Yang and Y. Tian. Super normal vector for activity recognition using depth sequences. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 804–811. IEEE, 2014. 8

[38] Y. Zhu, W. Chen, and G. Guo. Fusing spatiotemporal features and joints for 3d action recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2013 IEEE Conference on*, pages 486–491. IEEE, 2013. 5