2020

Differential Recurrent Neural Networks for Human Activity Recognition

Naifan Zhuang

University of Central Florida

Part of the Computer Sciences Commons

Find similar works at: https://stars.library.ucf.edu/etd2020

University of Central Florida Libraries http://library.ucf.edu

This Doctoral Dissertation (Open Access) is brought to you for free and open access by STARS. It has been accepted for inclusion in Electronic Theses and Dissertations, 2020- by an authorized administrator of STARS. For more information, please contact STARS@ucf.edu.

STARS Citation


Electronic Theses and Dissertations, 2020-. 163.

https://stars.library.ucf.edu/etd2020/163
DIFFRENTIAL RECURRENT NEURAL NETWORKS FOR HUMAN ACTIVITY RECOGNITION

by

NAIFAN ZHUANG
B.E. Northwestern Polytechnical University, 2012
M.S. University of Florida, 2014

A dissertation submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy
in the Department of Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Spring Term
2020

Major Professor: Kien A. Hua
ABSTRACT

Human activity recognition has been an active research area in recent years. The difficulty of this problem lies in the complex dynamical motion patterns embedded through the sequential frames. The Long Short-Term Memory (LSTM) recurrent 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 various time-series data, where the current hidden state has to be considered in the context of the past hidden states. 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 the change in information gain caused by the salient motions between the successive video frames. This change in information gain is quantified by Derivative of States (DoS), and thus the proposed LSTM model is termed differential Recurrent Neural Network (dRNN). Based on the energy profiling of DoS, we further propose to employ the State Energy Profile (SEP) to search for salient dRNN states and construct more informative representations. To better understand the scene and human appearance information, the dRNN model is extended by connecting Convolutional Neural Networks (CNN) and stacked dRNNs into an end-to-end model. Lastly, the dissertation continues to discuss and compare the combined and the individual orders of DoS used within the dRNN. We propose to control the LSTM gates via individual order of DoS and stack multiple levels of LSTM cells in increasing orders of state derivatives. To this end, we have introduced a new family of LSTMs, expanding the applications of LSTMs and advancing the performances of the state-of-the-art methods.
I would like to dedicate this dissertation to my love, Yawen and my parents, Hongjun and Shoumei, for always being there and supporting me through all the years.
ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my academic advisor, Dr. Kien A. Hua for his continued guidance, strong support, and valuable advice throughout my Ph.D. study. He introduces me to the world of research and teaches me scientific thinking, writing and presentation skills. He has always been supportive of my choice of research problems and been available whenever I need his advice. This dissertation would not have been possible without his guidance.

I would also like to thank my dissertation committee members, Dr. Hassan Foroosh, Dr. Haiyan N. Hu and Dr. Chung-Ching Wang, for their efforts in serving in my dissertation committee and providing valuable guidance and suggestions on my dissertation.

I would like to especially thank Dr. Guo-Jun Qi for his advice and help on our coauthored publications.

I would like to thank all the past and present DSG members, Jun Ye, Kai Li, Kutalmis Akpinar, Omar Nakhila, Sansiri Tarnpradab, Yusuph Turgun, Fereshte Jafariakinabad, Liheng Zhang, Bin Wang, Sanyi Zhang, for being good friends and making my Ph.D. study such an enjoyable and memorable experience.

Finally, I am especially grateful to my love, Yawen, and my parents, Hongjun and Shoumei, for offering unwavering love and wholehearted support. Their love and support have always been so important for my Ph.D. study and my life.
# TABLE OF CONTENTS

LIST OF FIGURES ..................................................... xi

LIST OF TABLES ..................................................... xiv

CHAPTER 1: INTRODUCTION ........................................... 1

  PROBLEM STATEMENT ............................................. 1

  MOTIVATION ....................................................... 2

    DIFFERENTIAL RECURRENT NEURAL NETWORKS ................. 3

    CONVOLUTIONAL DIFFERENTIAL RECURRENT NEURAL NETWORKS . 6

    DEEP DIFFERENTIAL RECURRENT NEURAL NETWORKS .......... 8

  DISSERTATION ORGANIZATION .................................. 10

CHAPTER 2: LITERATURE REVIEW ................................ 12

  ACTION RECOGNITION .......................................... 12

  GROUP ACTIVITY RECOGNITION .................................. 13

  CROWD ANALYSIS ................................................ 14

  LONG SHORT-TERM MEMORY ..................................... 16
<table>
<thead>
<tr>
<th>CHAPTER 3: BACKGROUND ON RECURRENT NEURAL NETWORKS</th>
<th>18</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECURRENT NEURAL NETWORKS</td>
<td>18</td>
</tr>
<tr>
<td>LONG SHORT-TERM MEMORY</td>
<td>19</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CHAPTER 4: DIFFERENTIAL RECURRENT NEURAL NETWORKS</th>
<th>21</th>
</tr>
</thead>
<tbody>
<tr>
<td>OVERVIEW</td>
<td>22</td>
</tr>
<tr>
<td>DERIVATIVE OF STATES</td>
<td>25</td>
</tr>
<tr>
<td>STATE ENERGY PROFILE</td>
<td>28</td>
</tr>
<tr>
<td>LEARNING ALGORITHM</td>
<td>30</td>
</tr>
<tr>
<td>EXPERIMENTS AND RESULTS</td>
<td>32</td>
</tr>
<tr>
<td>DATASETS</td>
<td>33</td>
</tr>
<tr>
<td>FEATURE EXTRACTION</td>
<td>34</td>
</tr>
<tr>
<td>ARCHITECTURE AND TRAINING</td>
<td>35</td>
</tr>
<tr>
<td>RESULTS ON KTH DATASET</td>
<td>36</td>
</tr>
<tr>
<td>RESULTS ON MSR ACTION3D DATASET</td>
<td>40</td>
</tr>
<tr>
<td>RESULTS ON NUS-HGA DATASET</td>
<td>42</td>
</tr>
<tr>
<td>RESULTS ON VIOLENT-FLOWS DATASET</td>
<td>43</td>
</tr>
<tr>
<td>SEP VS. OTHER POOLING TECHNIQUES</td>
<td>44</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

Figure 1.1: Various human activities from the UCF101 dataset. ......................... 2

Figure 1.2: The top part shows a sequence of successive frames of "Running" behavior from the KTH dataset. The middle part plots State Energy Profile (SEP) for different orders of DoS. The solid dots denote the local maxima of SEP. The corresponding frames of local maxima are shown in the bottom part. Mean pooling is performed over hidden states at local maxima landmarks and the hidden state at the last time-step to generate a comprehensive sequence representation. ................................................................. 5

Figure 1.3: The energy curves of the 0th-, 1st-, and 2nd-orders of DoS over an example of sequence for the activity "RunInGroup". ......................... 9

Figure 4.1: Architecture of dRNN model at time-step $t$. Best viewed in color. ........ 25

Figure 4.2: Confusion Matrices on the KTH-1 dataset obtained by the 2$^{nd}$-order dRNN model using LHS (left) and SEP (right). ......................... 37

Figure 4.3: Frame-by-frame prediction of action category over time with LSTM and SEP dRNNs. Best viewed in color. ......................... 38

Figure 4.4: Confusion Matrix on the MSR Action3D dataset obtained by the 2$^{nd}$-order dRNN model using SEP. ......................... 41

Figure 4.5: Confusion Matrices of 2$^{nd}$-order dRNN models using different pooling methods on the NUS-HGA dataset. ......................... 46
Figure 5.1: An illustration of CDRNN. It takes crowd image sequences as input and directly outputs predicted crowd class. The model consists of convolutional (red), pooling (green), mixed inception blocks (blue), fully-connected (cyan), DLSTM (yellow) and softmax (brown) layers. Best viewed in color.

Figure 5.2: An illustration of LSTM architecture at time $t$.

Figure 5.3: Training loss vs. epochs on the Violent-Flows dataset.

Figure 5.4: Confusion Matrices of CDRNN using different pooling strategies on the NUS-HGA dataset. CDRNN architecture with 3 stacked LSTM and FC1 is adopted.

Figure 5.5: Confusion Matrices of CDRNN using different pooling methods on the BEHAVE dataset. CDRNN architecture with 3 stacked LSTM and FC1 is adopted.

Figure 5.6: An illustration of a variant of CDRNN using VGG-16 as the convolutional part. The model consists of convolutional (red), pooling (green), fully-connected (blue), DLSTM (yellow) and softmax (brown) layers. Best viewed in color.

Figure 6.1: The energy curves of the 0th-, 1st-, and 2nd-orders of DoS over an example of sequence for the activity "RunInGroup".

Figure 6.2: Architectures of dRNN cell (left) and nth-order (layer $n + 1$) of $d^2$RNN cell (right) at time $t$. Best viewed in color.

Figure 6.3: An illustration of our framework for video-level activity recognition. Best viewed in color.

Figure 6.4: Training loss curves of different layers of $d^2$RNN over number of epochs on the NUS-HGA dataset.
# LIST OF TABLES

4.1 Architectures of the proposed model used on the datasets. .......................... 36

4.2 Performance comparison of LSTM models on the KTH-1 and KTH-2 datasets. 36

4.3 Cross-validation accuracy over five trials obtained by the other compared algorithms on KTH-1 and KTH-2 datasets. ......................................................... 39

4.4 Performance comparison of LSTM models on the MSR Action3D dataset. ... 40

4.5 Performances of the other compared methods on MSR Action3D dataset. ... 40

4.6 Performance comparison of LSTM models on the NUS-HGA dataset. ............ 42

4.7 Performances of the other compared methods on NUS-HGA dataset. MF indicates motion feature fusion and MAF indicates motion and appearance feature fusion. ................................................................. 42

4.8 Performance comparison of LSTM models on the Violent-Flows dataset. ....... 44

4.9 Performances of the other compared methods on Violent-Flows dataset. ....... 44

4.10 Performance comparison of SEP against different pooling strategies on the KTH-1 and NUS-HGA datasets. All the results are generated by 2nd-order dRNN. ................................................................. 45

4.11 Run-time efficiency on the datasets. ......................................................... 48

5.1 List of Crowd Video Classes for the CUHK Crowd dataset. ......................... 61
5.2 Performance comparison with existing methods on the NUS-HGA dataset. . . 63
5.3 Performance comparison of variants of CDRNN architectures on the NUS-
HGA dataset. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 64
5.4 Different Classes in BEHAVE dataset. . . . . . . . . . . . . . . . . . . . . 65
5.5 Performance comparison with existing methods on the BEHAVE dataset. . . 66
5.6 Performance comparison of variants of CDRNN architectures on the BE-
HAVE Crowd dataset. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 66
5.7 Performance comparison with existing methods on the Violent-Flows dataset.
We use GoogleNet Inception V3 as the convolutional part, three stacked LSTM/DLSTMs,
and mean-pooling for the video-level representation. . . . . . . . . . . . . . 68
5.8 Performance comparison of variants of CDRNN architectures on the Violent-
Flows dataset. IV3 indicates Inception V3. . . . . . . . . . . . . . . . . . . 69
5.9 Performance comparison with existing methods on the CUHK Crowd dataset.
We use GoogleNet Inception V3 as the convolutional part, three stacked LSTM/DLSTMs,
and mean-pooling for the video-level representation. . . . . . . . . . . . . . 70
5.10 Performance comparison of variants of CDRNN architectures on the CUHK
Crowd dataset. IV3 indicates Inception V3. . . . . . . . . . . . . . . . . . . 71
5.11 Evaluation of fully automatic recognition. . . . . . . . . . . . . . . . . . . 72
5.12 Evaluation of model generalization capability. . . . . . . . . . . . . . . . . 72
6.1 Architectures of the $d^2$RNN model used on the NUS-HGA and Violent-Flows datasets. .................................................. 86

6.2 Performance comparison of $d^2$RNN vs. traditional methods on the NUS-HGA dataset. MF indicates motion feature fusion and MAF indicates motion and appearance feature fusion. ........................................ 92

6.3 Performance comparison of deep models on the NUS-HGA dataset. .......... 93

6.4 Performance comparison of $d^2$RNN vs. traditional methods on the Violent-Flows dataset. ................................................................. 96

6.5 Performance comparison of deep models on the Violent-Flows dataset. ....... 96

6.6 Performance comparison of deep models on the UCF101 dataset. ............ 98
CHAPTER 1: INTRODUCTION

PROBLEM STATEMENT

Human activity recognition has been attracting increasing attention in the field of computer vision and artificial intelligence. Practical applications, such as video surveillance and public security, are inevitably based on the detection and recognition of human activities. Despite the importance of this problem, it remains unsolved due to the complex dynamical motion patterns embedded through the sequential data. To better approach human activity recognition, four sub-problems [2] have been proposed by researchers: (i) individual human action recognition, (ii) interaction recognition, (iii) group behavior analysis, and (iv) crowd activity recognition.

Individual action studies the behavior of a single person, and interaction recognition analyzes two people’s movements and interactions. Both sub-problems are relatively simple and are the most studied sub-problems of human activities. Since these activities only involve one or two persons, the individual movements of each person may be easily used to distinguish different actions.

While individual human action recognition is the fundamental basis of the activity recognition problem, understanding the group and crowd behavior tends to be more critical since they directly affect public safety. Group behavior analysis, or group activity recognition, analyzes activities of more than two but usually less than ten people. Since group behavior is usually performed by visually separable people, traditional solutions rely on human detection and tracking.

Crow activity recognition studies a large number of people and tries to understand the context. With the increase of the world population, crowd phenomena are growing more rapidly than ever before. Understanding crowd scenes, especially abnormal crowd behaviors, is becoming increasingly urgent and important. As crowd analysis generally involves a large number of people, it is
difficult to track each of the individuals and recognize their detailed actions to understand the overall activity of the crowd. Instead of tracking each person, which might cause unnecessary noise, holistic motion flows can be used to model the crowd as a whole.

Figure 1.1 shows an example of various human activities from the UCF101 dataset.

![Figure 1.1: Various human activities from the UCF101 dataset.](image)

**MOTIVATION**

To understand the complex spatio-temporal dynamics in human activities, we propose the following three models, e.g., differential recurrent neural networks, convolutional differential neural networks, and deep differential recurrent neural networks.
DIFFERENTIAL RECURRENT NEURAL NETWORKS

Recently, Recurrent Neural Networks (RNNs) [3], especially Long Short-Term Memory (LSTM) model [4], have gained significant attention in solving many challenging problems involving time-series data, such as human activity recognition [5, 6, 7], multilingual machine translation [8, 9], multimodal translation [10], and robot control [11]. In these applications, learning an appropriate representation of sequences is an important step in achieving artificial intelligence.

Compared with existing spatio-temporal features [12, 13, 14, 15] from the time-series data, RNNs use either a hidden layer [16] or a memory cell [4] to learn the time-evolving states which models the underlying dynamics of the input sequence. For example, [6] and [17] 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, forget, and output gates. These gates determine the amount of dynamic information entering or 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 activity recognition tasks [6, 7, 17]. The existing LSTM models represent a video by integrating over time all the available information from each frame. However, we observed that for a human activity recognition task, not all frames contain salient spatio-temporal information which are discriminative to different classes of activities. 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 human activities. 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 required to model the dynamic evolution of actions.

In our early work of dRNN [18], we used hidden state at the last time-step to model the entire video sequence, which we call the Last-Hidden-State (LHS) method. In a very long sequence, the information learned from previous time steps decays gradually over time. Consequently, the hidden state at the last time-step tends to be insufficient to model the whole sequence. In the meantime, we observed that each order of DoS energy could align a different level of motion saliency, as seen in Figure 1.2. We are motivated to address the above problem by digging into DoS to search for discriminative hidden states over different time-steps.

Based on the observation of natural alignment of DoS energy and motion saliency, we further propose to use State Energy Profile (SEP) to generate more discriminative and informative video representation. While DoS models the salient spatio-temporal representations of human activities, the motion energy intensity can be approximately estimated by the L2-norm of DoS. After plotting the energy curve of DoS over the time-steps, we can detect the local maxima landmarks of SEP. As shown in Figure 1.2, these landmarks indicate strong motion-intensity at the corresponding time-steps and are more likely to provide discriminative information for recognizing human activities. We then construct the video sequence representation based on the hidden states at those landmarks in addition to the hidden state at the last time-step. To train the dRNN SEP model, we use truncated
Back Propagation algorithm to prevent the exploding or diminishing errors through time [4]. 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.

Figure 1.2: The top part shows a sequence of successive frames of "Running" behavior from the KTH dataset. The middle part plots State Energy Profile (SEP) for different orders of DoS. The solid dots denote the local maxima of SEP. The corresponding frames of local maxima are shown in the bottom part. Mean pooling is performed over hidden states at local maxima landmarks and the hidden state at the last time-step to generate a comprehensive sequence representation.

To explore the potential of dRNN comprehensively, we have involved different orders of DoS to detect and capture the various levels of dynamical motion patterns for the dRNN model. The insight beneath is as follows. When modeling a moving object in a video, the 1st-order of DoS captures the velocity while the 2nd-order captures its acceleration. If we set the DoS to be 0th-order, which resembles the traditional LSTM, it should model the locality information. Until now, we can see that the relationship between LSTM and dRNN. With a 0th-order of DoS, LSTM is a special case of dRNN. Higher-order dRNN captures not only locality information, but also velocity and acceleration information. Thus, we have introduced a new family of LSTMs; and LSTM is a special form of dRNN.
The effectiveness of dRNN is demonstrated on its application to human activity recognition. We show that dRNN can achieve state-of-the-art performance on both 2D and 3D single-person action recognition datasets. Extensive experimental studies were further performed on human group and crowd activity datasets to show the generalization ability of dRNN. Specifically, dRNNs outperform the existing LSTM models on these activity recognition tasks, consistently achieving better performance with the input sequences. Armed with SEP, dRNNs further enhance the experimental performances. 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.

To this end, we propose to stack multiple levels of LSTM cells with individual and increasing orders of DoS. The proposed model has the following advantages. With the individual order of DoS, each layer of LSTM cell captures a certain level of salient spatio-temporal information. With stacked architecture, our model progressively builds up the ability of LSTM gates to detect salient dynamic patterns with deeper memory layers modeling higher orders of DoS. The proposed model is thus termed deep differential Recurrent Neural Network ($d^2$RNN). The $d^2$RNN differs from conventional stacked LSTMs in that stacked LSTMs use homogeneous LSTM layers while $d^2$RNN uses heterogeneous ones. In this way, $d^2$RNN is not only capable of modeling more complex dynamical patterns, but also enables a hierarchy of DoS saliency in deep layers to model the spatio-temporal dynamics over time.

**CONVOLUTIONAL DIFFERENTIAL RECURRENT NEURAL NETWORKS**

For the human activity recognition problem, motion dynamics might not be sufficient to understand the complex settings. In many cases, the background scene and human appearance information can provide necessary cues to distinguish different categories of activities. This often holds valid for
group activity recognition and crowd analysis, when visual ambiguities and occlusions occur frequently in those scenarios. In the last decade, researchers from the computer vision community have shown much interest in developing automated crowd scene understanding systems. Video analysis for uncrowded scenes usually involves object detection, object tracking, and behavior recognition. Such solutions, however, are not suitable for crowded scenes; and special considerations must be taken into account. As a crucial basis, appropriate feature representation for crowded scenes is necessary. In terms of representation level, previous crowd features can be divided into the following three categories [19]: flow-based features, local spatio-temporal features, and trajectory/tracklet features. However, the above methods still lack the ability to fully understand the background scene and human appearance information.

The disadvantages of the above existing crowd representation methods motivate us to explore a new representation for crowded scenes, which is simple yet can still maintain the raw information of the source video as much as possible. We are inspired by the success of convolutional neural networks [20] to explore the use of raw input data for crowd scenes. We further propose an end-to-end deep architecture, Convolutional Differential Recurrent Neural Networks (CDRNN), for group activity recognition and crowd scene understanding. CDRNN consists of convolutional neural networks (CNN) and stacked levels of differential Recurrent Neural Networks (dRNN). Different from traditional non-end-to-end solutions which separate the steps of feature extraction and parameter learning, we utilize a unified deep model to optimize the parameters of CNN and RNN hand in hand. It thus has the potential of generating a more harmonious model. The proposed architecture takes sequential raw image data as input, and does not rely on tracklet or trajectory detection. It thus has clear advantages over the traditional flow-based and trajectory-based methods, especially in challenging crowd scenarios of high density and low mobility. Taking advantage of CNN and RNN, CDRNN can effectively analyze the crowd semantics, where CNN is good at modeling the semantic crowd scene semantics information and dRNN models the motion temporal
information.

The proposed CDRNN has the following advantages. Firstly, when dealing with highly dense crowd scenes, trajectory/tracklet methods tend to perform poorly. CDRNN has no such problem because it does not rely on trajectory detection. Secondly, flow-based and trajectory-based methods assume crowd mobility when extracting the flow and trajectory representations. The convolutional neural network layers in the proposed architecture can model the scene semantics and do not require motion information from the crowd. Thirdly, the increasing orders of DoS progressively build up the ability of LSTM gates to detect different levels of salient dynamical patterns. Lastly, different from existing LSTM-based crowd scene solutions which only use stacked LSTMs and claim “deep in time”, CDRNN models the spatial and temporal information in a unified architecture and achieves “deep in space and time”.

We extensively evaluate the performance of the proposed deep architecture on three public crowd understanding datasets, Violent-Flows [21], CUHK Crowd [22] and NUS-HGA [23]. Experimental results show that the proposed technique significantly outperforms the conventional flow-based and trajectory/tracklet-based methods by a great margin. We also show that our CDRNN model can outperform the LSTM-based methods by achieving “deep in both space and time”.

**DEEP DIFFERENTIAL RECURRENT NEURAL NETWORKS**

Our proposed dRNN model analyzes the dynamics of actions by computing different orders of Derivative of State (DoS). DoS models the change in information gain caused by the salient motions between the successive frames using higher orders of internal state derivatives. Intuitively, 1st-order DoS represents the velocity of change of internal state memory while 2nd-order DoS represents the acceleration of memory state change. This reveals that the conventional LSTM, whose internal cell is simply 0th-order DoS, only captures the locality of information change.
Despite the above-mentioned advantages, dRNN is formulated in the fashion that the gates are modulated by the weighted combinations of several orders of DoS. While an individual order of DoS is able to model a certain degree of dynamical structures, the sum of all the orders of DoS could distort the detected salient motion patterns. To support the above observation, Fig. 6.1 illustrates the energy curves of the 0th-, 1st-, and 2nd-orders of DoS over an example of a sequence for the activity "RunInGroup". The local maxima indicate high energy landmarks corresponding to the salient motion frames at different levels. Indeed, each order of DoS enables the LSTM unit to model the dynamics of local saliency at a certain level, e.g. 0th-, 1st-, and 2nd-orders of DoS captures locality, velocity, and acceleration information change, respectively. The weighted sum of different orders of DoS, however, may risk misaligning salient motion and result in distorted motion patterns. To further confirm the above claim, a preliminary experimental study was conducted. The study demonstrates that a simple ensemble model of individual orders of DoS outperforms the conventional same-order weighted-sum dRNN, thus reveals the suboptimality of combining different orders DoS within LSTM cell. The above analysis inspires us to question the internal structure of conventional dRNN, and reconsider to use the individual orders of state derivatives to control the LSTM gates.
Although we have separate different orders of DoS and use the individual to control the LSTM gates, the problem left is how to simultaneously utilize, preserve and enhance based on a different order of state derivatives. As is generally accepted, RNNs are inherently deep in time because the current hidden state is a function of all previous hidden states. By questioning whether RNNs could also benefit from depth in space, just as feed-forward layers which are stacked in conventional deep networks, Graves et al. [24] introduced Deep Recurrent Neural Networks, also known as stacked LSTMs. Stacked LSTMs have shown superiority over the traditional LSTM in modeling complex sequences and have been used in various types of applications. Inspired by Deep Recurrent Neural Network, we are motivated to explore whether the stacked deep layers in space could naturally reveal the saliency of motion dynamics over time, thus avoiding the misaligned DoS in different orders.

We demonstrate the performance of $d^2$RNN on three publicly available human activity datasets: NUS-HGA [23], Violent-Flows [21], and UCF101 [25]. Specifically, $d^2$RNN outperforms the existing LSTM, dRNN, and stacked LSTM models, consistently achieving better performance in detecting human activities in sequences. In addition, we compared with the other non-LSTM algorithms, where $d^2$RNN model also reached competitive performance.

**DISSERTATION ORGANIZATION**

The rest of this dissertation is organized as follows:

Chapter 2 presents the literature review, which consists of four parts: action recognition, group activity recognition, and crowd activity recognition.

Chapter 3 gives a background review of the traditional recurrent neural networks, as well as long short-term memory networks.
Chapter 4 presents our first proposed model, differential recurrent neural networks. It gives great details of the concepts newly proposed, such as Derivative of States (DoS) and State Energy Profile (SEP).

Chapter 5 presents the convolutional differential recurrent neural networks and studies its performance for group activity recognition and crowd analysis.

Chapter 6 provides our thoughts and discussion of combined or individual orders of Derivative of States, and then presents the deep differential recurrent neural networks.

Chapter 7 is the conclusion and future work.
CHAPTER 2: LITERATURE REVIEW

In this chapter, we first discuss the related works of human activity recognition. Since human activity recognition includes sub-problems individual human action, group activity recognition, and crowd analysis, we present the literature review to each of the sub-problem. Then, we present the related works for Recurrent Neural Networks (RNN), especially Long Short-Term Memory (LSTM). The detailed mathematical background review of RNN and LSTM can be found in the next chapter.

ACTION RECOGNITION

Action recognition has been a long-standing research problem in computer vision and pattern recognition community. 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 [26] uses the histogram of 3D gradient orientations to represent the motion structure over the frame sequences; 3D-SIFT [13] extends the popular SIFT descriptor to characterize the scale-invariant spatio-temporal structure for 3D video volume; actionlet ensemble [27] utilizes a robust approach to model the discriminative features from 3D positions of 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 [20] and speech recognition [28] has inspired many researchers to
apply the deep neural networks, such as 3D Convolutional Neural Networks (3DCNNs) [29] and Recurrent Neural Networks (RNNs) [6, 17], to action recognition. In particular, Baccouche et al. [29] developed a 3DCNN that extends the conventional CNN by taking space-time volume as input. On the contrary, [6] and [17] 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 [29], an LSTM model takes a sequence of Harris3D and 3DCNN descriptors extracted from each frame as input, and the results on KTH dataset have shown the state-of-the-art performance [29].

GROUP ACTIVITY RECOGNITION

Human group activity recognition has many practical applications in video surveillance and public security. Compared to individual human action recognition, group activity recognition is more challenging as it involves more participants and thus contains more complex semantics.

Most existing approaches of group activity recognition are based on motion trajectories of group participants. Ni et al. [23] applied motion trajectory segments as inputs and used digital filters’ frequency responses to represent the motion information. Zhou et al. [30] analyzed interactions between two individuals using the Granger Causality Test. Yin et al. [31] first clustered each person into sub-groups and then constructed a network and extracted a histogram feature for their analysis. Zhang et al. [32] used the K-means algorithm to find sub-groups and then regarded group activities as a combination of its sub-groups. Zhu et al. [33] considered motion trajectory as a dynamic system and used the Markov stationary distribution to acquire local appearance features as a descriptor of group action. A statistical representation to encode the causal relationships of couples of trajectories based on Bayesian Networks was proposed by Dore et al. [34] for interaction.
behavior analysis. Chu et al. [35] designed an algorithm based on using a heat-map to model the trajectories as a series of heat sources to create a heat map for representing group actions.

Cheng et al. [1] and Cho et al. [2] achieved state-of-the-art performances on the NUS-HGA and BEHAVE benchmark datasets respectively. Cheng et al. [1] proposed a layered model of human group action and represented activity patterns with both motion and appearance information. Their performance on NUS-HGA achieved an accuracy of 96.20%. Cho et al. [2] addressed the problem by using group interaction zones to detect meaningful groups to handle noisy information. They achieved the state-of-the-art performance of 93.74% on the BEHAVE dataset. Nevertheless, both of the methods used pre-determined human bounding boxes or trajectory information as input to their system: Cheng et al. [1] used tracking tools, which requires manual annotation, to acquire human trajectories for the NUS-HGA dataset. Cho et al. [2] directly used the accurate bounding box information provided by the BEHAVE dataset.

CROWD ANALYSIS

Due to the high-density and low-resolution characteristics of crowd scene videos, feature representation for crowd analysis is a crucial basis for this problem. Previous crowd analysis methods can be divided into three categories according to the feature representation level: flow-based features, local spatio-temporal features and trajectory/tracklet features.

Crowd scenes often present a highly sense scenario, which makes it very difficult to track each individual in the videos. Flow-based crowd analysis techniques extract densely features on the pixel level, avoiding the issue of tracking each object. For flow-based features, the rationale is as follows: as specific actions of individuals may be relatively random, but the overall dynamics of the crowd can still be convincing. Several works [36, 37] use optical flow to compute pixel-wise
instantaneous motion between consecutive frames and apply to crowd motion detection. Based on the Lagrangian framework of fluid dynamics [38], particle flow was introduced to handle abnormal crowd behavior detection [39, 40]. To acquire an accurate representation of the crowd motion flow, Mehran et.al. [41] introduced a streakline to compute the motion field for crowd scene analysis. Flow-based methods achieved success in addressing dense and complex crowd flows by avoiding tracking at the macroscopic level. However, flow-based features ignore the scene information and tend to fail in crowd videos with less mobility.

Some extremely crowded scenes, though similar in density, are less structured due to the high variability of individual movements. In this case, the motion within each local area may be non-uniform and flow-based features, such as optical flow, would not provide enough information. One solution is to exploit the dense local motion patterns created by the subjects and model their spatio-temporal relationships. The related methods use spatio-temporal gradients [42, 43], and histogram functions [44, 45]. Kratz et.al. [42, 43] use the distribution of spatio-temporal gradients as the base representation to detect unusual activities. Motion histograms can be considered as one kind of motion information defined in local regions. Jodoin et.al. [44] proposed a feature called orientation distribution function, which has advantage in computation for the upcoming motion pattern learning. Cong et.al. [45] proposed a novel feature descriptor called multi-scale histogram of optical flow. It preserves both motion information and spatial contextual information, and performs well on abnormal event detection. The downside of local spatio-temporal features is since these methods analyze local features of crowd dynamics and are sub-optimal for complex crowd behaviors with long-range dependency.

Compared with the above two types of feature representations, trajectories(tracklet) is more semantic and seems to be more popular in the recent computer vision research community. Since the density of the crowd increases and the scene clutter becomes severe, traditional object detection and tracking can hardly be performed accurately. A new motion feature called tracklet has been
proposed. As a fragment of a trajectory obtained by the tracker within a short period, tracklets terminate when ambiguities occur. Tracklets thus have been used to complete trajectories for tracking. Several tracklet based approaches [46, 47, 48] were proposed to learn semantic regions and clustering trajectories. Recent methods for crowd scene understanding mostly analyze crowd activities based on motion features extracted from trajectories/tracklets of objects [22, 21, 49, 50, 51]. Marsden et al. [52] studied scene-level holistic features using tracklets to solve real-time crowd behavior anomaly detection problems. Su et al. [49] used tracklet-based features and explored Coherent LSTM to model the nonlinear characteristics and spatio-temporal motion patterns in crowd behaviors. These two methods hold state-of-the-art performances for the Violent-Flows and CUHK Crowd datasets, respectively. The trajectory/tracklet feature contains more semantic information, but the accuracy of trajectories/tracklets dictates the performance of crowd scene analysis. In extremely crowded areas, tracking algorithms could fail and generate inaccurate trajectories.

LONG SHORT-TERM MEMORY

Due to the exponential decay, traditional RNNs [3] are limited in learning long-term sequences. Hochreiter et al. [4] designed Long Short-Term Memory (LSTM) to exploit the long-range dependency. As LSTM shows superiority in modeling time-series data, it is widely used for various kinds of sequential processing tasks and several LSTM variants were proposed to improve the architecture of standard LSTM. S-LSTM [53] is an LSTM network with tree structures. The hierarchical structure of S-LSTM aims to mitigate the gradient vanishing problem and model more complicated input than LSTM. Stacked LSTM [24] borrows the idea of depth in ANNs and stacking hidden layers with LSTM cells in space to increase the network capacity. Bidirectional LSTM [16] captures both future and past context of the input sequence. Multidimensional LSTM (MDLSTM) [54] uses interconnection from previous state of cell to extend the memory of LSTM along
every $N$-dimension. The MDLSTM receives inputs in an $N$-dimensional arrangement, thus can model multidimensional sequences. MDLSTM model becomes unstable with the growth of the grid size and LSTM depth in space. Grid LSTM [55] provides a solution by altering the computation of output memory vectors. Even though the above variants of LSTMs show superiority in some aspects, none of them explicitly analyzes the spatio-temporal dynamics corresponding to the information gain of internal memory states. This differs our work from all the above variants of LSTMs.
CHAPTER 3: BACKGROUND ON RECURRENT NEURAL NETWORKS

In this section, we review in detail 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.

RECURRENT NEURAL NETWORKS

Traditional Recurrent Neural Networks (RNNs) [3] model the dynamics of an input sequence of frames \( \{x_t \in \mathbb{R}^m | t = 1, 2, ..., T \} \) through a sequence of hidden states \( \{h_t \in \mathbb{R}^n | t = 1, 2, ..., T \} \), thereby learning the spatio-temporal structure of the input sequence. For instance, a classical RNN model uses the following recurrent equation

\[
    h_t = \tanh(W_{hh}h_{t-1} + W_{hx}x_t + b_h),
\]

(3.1)

to model the hidden state \( h_t \) at time \( t \) by combining the information from the current input \( x_t \) and the past hidden state \( h_{t-1} \). The hyperbolic tangent \( \tanh(\cdot) \) in the above equation is an activation function with range \([-1, 1]\). \( W_{hh} \) and \( W_{hx} \) are two mapping matrices to the hidden states, and \( b_h \) is the bias vector.

The hidden states will then be mapped to an output sequence \( \{y_t \in \mathbb{R}^k | t = 1, 2, ..., T \} \) as

\[
    y_t = \tanh(W_{yh}h_t + b_y),
\]

(3.2)

where each \( y_t \) represents a 1-of-\( k \) encoding of the confidence scores on \( k \) classes of human activities. This output can then be transformed to a vector of probabilities \( p_t \) by the softmax function as
$p_{t,c} = \frac{\exp(y_{t,c})}{\sum_{l=1}^{k} \exp(y_{t,l})}$, \hspace{1cm} (3.3)

where each entry $p_{t,c}$ is the probability of frame $t$ belonging to class $c \in \{1, ..., k\}$.

LONG SHORT-TERM MEMORY

Due to the exponential decay in retaining the context information from video frames, the aforementioned classical RNNs are limited in learning the long-term representation of sequences. To overcome this limitation, Long Short-Term Memory (LSTM) [4], a variant of RNN, has been designed to exploit and find the long-range dependency between the input sequences and output labels.

Specifically, LSTM consists of a sequence of memory cells, each containing an internal state $s_t$ to store the memory of the input sequence up to time $t$. In order to retain the memory with respect to a context in a long sequence, three types of gate units are incorporated into LSTMs to regulate what information enters and leaves the memory cells over time. These gate units are activated by a nonlinear function of input/output sequences as well as internal states, which makes LSTM powerful enough to model dynamically changing contexts.

Formally, an LSTM cell has the following gates:

(i) Input gate $i_t$ regulates the degree to which the input information would enter the memory cell to affect the internal state $s_t$ at time $t$. The activation of the gate has the following recurrent form:

$$i_t = \sigma(W_{is}s_{t-1} + W_{ih}h_{t-1} + W_{ix}x_t + b_i),$$

where the sigmoid $\sigma(\cdot)$ is an activation function in the range $[0,1]$, with 0 meaning the gate is
closed and 1 meaning the gate is completely open; \( W_{fs} \) are the mapping matrices and \( b_f \) is the bias vector.

(ii) Forget gate \( f_t \) modulates the previous state \( s_{t-1} \) to control its contribution to the current state. It is defined as
\[
f_t = \sigma(W_{fs}s_{t-1} + W_{fh}h_{t-1} + W_{fx}x_t + b_f),
\]
with the mapping matrices \( W_{fs} \) and the bias vector \( b_f \).

With the input and forget gate units, the internal state \( s_t \) of each memory cell can be updated as below:
\[
s_t = f_t \otimes s_{t-1} + i_t \otimes \tanh(W_{sh}h_{t-1} + W_{sx}x_t + b_s), \tag{3.4}
\]
where \( \otimes \) stands for element-wise product.

(iii) Output gate \( o_t \) gates the information output from a memory cell which would influence the future states of LSTM cells. It is defined as
\[
o_t = \sigma(W_{os}s_t + W_{oh}h_{t-1} + W_{ox}x_t + b_o).
\]

Then the hidden state of a memory cell is output as
\[
h_t = o_t \otimes \tanh(s_t). \tag{3.5}
\]

In brief, LSTM proceeds by iteratively applying Eq. (3.4) and Eq. (3.5) to update the internal state \( s_t \) and the hidden state \( h_t \). In the process, the input gate, forget gate, and output gate play an important role in controlling the information entering and leaving the memory cell. More details about LSTMs can be found in [4].
CHAPTER 4: DIFFERENTIAL RECURRENT NEURAL NETWORKS

Parts of this chapter have been presented in IEEE ICPR [56] (Copyright ©2016 IEEE).

The Long Short-Term Memory (LSTM) recurrent 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 present in long sequences. 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 the change in information gain caused by the salient motions between the successive video 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). In addition, the original work used the hidden state at the last time-step to model the entire video sequence. Based on the energy profiling of DoS, we further propose to employ the State Energy Profile (SEP) to search for salient dRNN states and construct more informative representations. The effectiveness of the proposed model was demonstrated by automatically recognizing human actions from the real-world 2D and 3D single-person action datasets. Extensive experimental studies were further performed on human group and crowd activity datasets to show the generalization ability of dRNN. We point out that LSTM is a special form of dRNN. As a result, we have introduced a new family of LSTMs. Our study is one of the first works towards demonstrating the potential of learning complex time-series representations via high-order derivatives of states.
OVERVIEW

Recurrent Neural Networks (RNNs) [3], especially Long Short-Term Memory (LSTM) model [4], have gained significant attention in solving many challenging problems involving time-series data, such as human activity recognition [5, 6, 7], multilingual machine translation [8, 9], multimodal translation [10], and robot control [11]. In these applications, learning an appropriate representation of sequences is an important step in achieving artificial intelligence.

Compared with existing spatio-temporal features [12, 13, 14, 15] from the time-series data, RNNs use either a hidden layer [16] or a memory cell [4] to learn the time-evolving states which models the underlying dynamics of the input sequence. For example, [6] and [17] 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, forget, and output gates. These gates determine the amount of dynamic information entering or 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 activity recognition tasks [6, 7, 17]. The existing LSTM models represent a video by integrating over time all the available information from each frame. However, we observed that for a human activity recognition task, not all frames contain salient spatio-temporal information which are discriminative to different classes of activities. 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 human activities. 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 required to model the dynamic evolution of actions.

In our prior work of dRNN [18], we used hidden state at the last time-step to model the entire video sequence, which we call the Last-Hidden-State (LHS) method. In a very long sequence, the information learned from previous time steps decays gradually over time. Consequently, the hidden state at the last time-step tends to be insufficient to model the whole sequence. In the meantime, we observed that each order of DoS energy could align a different level of motion saliency, as seen in Fig. 1.2.. We are motivated to address the above problem by digging into DoS to search for discriminative hidden states over different time-steps.

Based on the observation of natural alignment of DoS energy and motion saliency, we now further propose to use State Energy Profile (SEP) to generate more discriminative and informative video representation. While DoS models the salient spatio-temporal representations of human activities, the motion energy intensity can be approximately estimated by the L2-norm of DoS. After plotting the energy curve of DoS over the time-steps, we can detect the local maxima landmarks of SEP. As shown in Fig. 1.2., these landmarks indicate strong motion-intensity at the corresponding time-steps and are more likely to provide discriminative information for recognizing human activities. We then construct the video sequence representation based on the hidden states at those landmarks in addition to the hidden state at the last time-step. To train the dRNN SEP model, we use the
truncated Back Propagation algorithm to prevent the exploding or diminishing errors through time [4]. 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.

To explore the potential of dRNN comprehensively, we have involved different orders of DoS to detect and capture the various levels of dynamical motion patterns for the dRNN model. The insight beneath is as follows. When modeling a moving object in a video, the 1st-order of DoS captures the velocity while the 2nd-order captures its acceleration. If we set the DoS to be 0th-order, which resembles the traditional LSTM, it should model the locality information. Until now, we can clearly see that the relationship between LSTM and dRNN. With a 0th-order of DoS, LSTM is a special case of dRNN. Higher-order dRNN captures not only locality information, but also velocity and acceleration information. Thus, we have introduced a new family of LSTMs; and LSTM is a special form of dRNN.

The effectiveness of dRNN is demonstrated in its application to human activity recognition. We show that dRNN can achieve state-of-the-art performance on both 2D and 3D single-person action recognition datasets. Extensive experimental studies were further performed on human group and crowd activity datasets to show the generalization ability of dRNN. Specifically, dRNNs outperform the existing LSTM models on these activity recognition tasks, consistently achieving better performance with the input sequences. Armed with SEP, dRNNs further enhance the experimental performances. 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.
For the task of activity recognition, not all video frames contain salient patterns to discriminate between different classes of activities. Many spatio-temporal descriptors, such as 3D-SIFT [13] and HoGHoF [57], 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 propose a novel LSTM model to automatically learn the dynamics of human activities, by detecting and integrating the salient spatio-temporal sequences. Although the traditional LSTM neural network is capable of processing complex sequential information, it might...
fail to capture the salient motion patterns because the gate units do not explicitly consider the impact of dynamic structures present in input sequences. This makes the conventional LSTM model inadequate to learn the evolution of human activity states.

As the internal state of each memory cell contains the accumulated information of the spatio-temporal structure, the Derivative of States (DoS) \( \frac{ds_t}{dt} \) quantifies the change of information at each time \( t \). In other words, a large magnitude of DoS indicates that a salient spatio-temporal structure containing the informative dynamics is caused by an abrupt change of human activity state. In this case, the gate units should allow more information to enter the memory cell to update its internal state. On the other hand, when the magnitude of DoS is small, the incoming information should be gated out of the memory cell so that the internal state would not be affected by the current input. Therefore, DoS can be used as a factor to gate the information flow into and out of the internal state of the memory cell over time.

Moreover, we can involve higher-orders of DoS to detect and capture the higher-order dynamic patterns for the dRNN model. For example, when modeling a moving object in a video, the \( 1^{st} \)-order of DoS captures the velocity and the \( 2^{nd} \)-order captures its acceleration. These different orders of DoS will enable dRNN to better represent the dynamic evolution of human activities in videos.

Fig. 4.1 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 \( N \):

\[
i_t = \sigma \left( \sum_{n=0}^{N} W_{id}^{(n)} \frac{d^{(n)} s_{t-1}}{dt} + W_{ih} h_{t-1} + W_{ix} x_t + b_i \right),
\]

\[
f_t = \sigma \left( \sum_{n=0}^{N} W_{fd}^{(n)} \frac{d^{(n)} s_{t-1}}{dt} + W_{fh} h_{t-1} + W_{fx} x_t + b_f \right),
\]

\[
n_t = \sigma \left( \sum_{n=0}^{N} W_{nd}^{(n)} \frac{d^{(n)} s_{t-1}}{dt} + W_{nh} h_{t-1} + W_{nx} x_t + b_n \right),
\]

\[
o_t = \sigma \left( \sum_{n=0}^{N} W_{od}^{(n)} \frac{d^{(n)} s_{t-1}}{dt} + W_{oh} h_{t-1} + W_{ox} x_t + b_o \right),
\]

\[
h_t = \tanh \left( \sum_{n=0}^{N} W_{hid}^{(n)} \frac{d^{(n)} s_{t-1}}{dt} + W_{h} h_{t-1} + W_{x} x_t + b_h \right).
\]
\[ o_t = \sigma \left( \sum_{n=0}^{N} W_{od}^{(n)} \frac{d^{(n)} s_t}{dt^{(n)}} + W_{oh} h_{t-1} + W_{ox} x_t + b_o \right), \]  

(4.3)

where \( \frac{d^{(n)} s_t}{dt^{(n)}} \) is the \( n^{th} \)-order DoS. Until now, it can be clearly seen that when \( N = 0 \), the dRNN model resembles the conventional LSTM. Therefore, LSTM is a special form of dRNN.

It is worth pointing out that we do not use the derivative of input as a measurement of salient dynamics to control the gate units. The derivative of input would amplify the unwanted noises which are often contained in the input sequence. In addition, this derivative of input only represents the local dynamic saliency, in contrast to the long short-term change in the information gained over time. For example, a similar movement which has occurred several frames ago and been stored by LSTM, could be treated as a novel salient motion using a derivative of inputs. On the contrary, DoS does not have this problem because the internal state \( s_t \) has long-term memory of the past motion pattern and would not treat the same motion as a salient pattern if it has previously occurred.

Since dRNN is defined in the discrete-time domain, the 1\( ^{st} \)-order derivative \( \frac{ds_t}{dt} \), as the velocity of information change, can be discretized as the difference of states:

\[ v_t \triangleq \frac{ds_t}{dt} = s_t - s_{t-1}, \]  

(4.4)

for simplicity [58].

Similarly, we consider the 2\( ^{nd} \)-order of DoS as the acceleration of information change. It can be discretized as:

\[ a_t \triangleq \frac{d^2 s_t}{dt^2} = v_t - v_{t-1} = s_t - 2s_{t-1} + s_{t-2}. \]  

(4.5)

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

In our prior work, we used the hidden state at the last time-step to represent the entire sequence. Even though dRNN is good at modeling long input sequential data, the hidden state at the last time-step still might not be enough to summarize the complex dynamic evolution of human activities. While dRNN processes the input frames sequentially, hidden state information learned from previous frames decays gradually over a very long sequence. It tends to be suboptimal to only use the hidden state from the last time-step for video representation.

As mentioned above, DoS represents the information gain of internal states between consecutive video frames. The L2-norm of DoS can approximately estimate the motion energy intensity of human activities. We name the above-estimated energy intensity over all time steps as State Energy Profile (SEP). Thus, SEP can be used to locate hidden states with salient information. Specifically, a salient spatio-temporal frame could yield a large magnitude of DoS, which corresponds to a local maximum in SEP. Therefore, in terms of SEP, we can determine whether information gain is strong at certain time-steps and thus local maxima of SEP indicate the most informative hidden states. Moreover, we involve different orders of SEP to detect different levels of dynamic patterns. Formally we compute the SEP, denoted as $E^n_t$, in terms of $n^{th}$-order DoS as follows:

$$E^n_t = \| \frac{d^{(n)}s_t}{dt^{(n)}} \|_2, (n = 0, 1, ..., N).$$  \hspace{1cm} (4.6)

In this paper, we consider up to $2^{nd}$-order of SEP. Again, since the dRNN model is defined in the discrete-time domain, SEP $E^n_t$ with different orders are discretized as below:

$$E^0_t = \| \frac{d^{(0)}s_t}{dt^{(0)}} \|_2 = \|s_t\|_2,$$ \hspace{1cm} (4.7)
In order to locate maxima landmarks, we plot SEP curves on different orders. Note that since LSTMs have long-term memory and controllable virtues, SEP curves are smoothed without the need for filtering. This would not be observed if the derivative of input is applied instead of DoS. Based on the SEP curves, we then detect local maxima landmarks of SEP for different orders. Again, those local maxima correspond to time-steps of salient and strong motion patterns. To be more specific, local maxima landmarks of $E_1^t$ correspond to the time-steps with high velocity and maxima landmarks of $E_2^t$ correspond to the time-steps with high acceleration. As seen in Fig. 1.2., the action of Running exhibits the most informative pattern when the person in the video is of high speed and/or high acceleration. These moments correspond to the local maxima landmarks of SEP.

Based on the above observation, we construct the set of hidden states corresponding to the local maxima of SEP as $\{h_{m_1}^0, h_{m_2}^0, \ldots, h_{m_u}^0\}, n = 0, 1, \ldots, N$. Here $N$ is the highest order of DoS, and $u$ is the number of local maxima landmarks for order $n$. Since LSTM cell aggregates over all the time-steps, the hidden state at the last time-step contains the overall information of the video sequence.

We then form the Candidate Set of hidden states by adding the hidden state at the last time-step to the above set. Candidate Set is then denoted as $\{h_{m_1}^n, h_{m_2}^n, \ldots, h_{m_u}^n, h_T\}$, here $T$ denotes the last time-step. To suppress the unwanted noise, mean pooling is then performed over the hidden states in the above Candidate Set to create the final representation for the entire sequence:

$$h_\tau = \mu(\{h_{m_1}^n, h_{m_2}^n, \ldots, h_{m_u}^n, h_T\}),$$

(4.10)
where $\mu$ denotes mean pooling, $n = 0, 1, ..., N$.

To support our motivation for learning LSTM representations based on the SEP method, we illustrate example frames of Running activity from the KTH dataset with SEP signal and local maxima landmarks in Fig. 1.2. It shows that local maxima of SEP correspond to most intense motion frames, where the most salient and informative hidden states are located. In this specific case, the maxima of the SEP signal correspond to the moments when the person in the video runs at high speed or acceleration. Interestingly, some human activities such as Walking and Running, exhibit regular motion periodicity [59]. SEP also increases the possibility of finding aligned video frames.

To better understand the potential of the SEP pooling strategy, we discuss the relationship between SEP and other pooling methods. There are currently two frequently used pooling methods. Mean pooling, by averaging all time-step hidden states, statistically summarizes the information collected from all previous frames and thus has a proper representation of the states. Due to the smoothing nature of mean pooling, the salient human activity information acquired by the neural network could be lost in the process. Max pooling is better at selecting salient signals and often generates more discriminative representation, but it is subject to noise. The proposed SEP pooling strategy, tailored for the dRNN model, combines the advantages of mean pooling and max pooling while minimizing the drawbacks.

LEARNING ALGORITHM

With the above recurrent equations, for a human activity video sequence including $T$ time-step features, the SEP dRNN model proceeds in the following order at time step $t$:

- Compute input gate activation $i_t$ and forget gate activation $f_t$ by Eq. (4.1) and Eq. (4.2).
• Update state $s_t$ with $i_t$ and $f_t$ by Eq. (3.4).

• Compute discretized DoS \( \left\{ \frac{d^{(n)}s_{t-1}}{dt^{(n)}} \right| n = 0, 1, \ldots, N \} \) up to \( N^{th} \)-order at time \( t \), e.g. Eq. (4.4) and Eq. (4.5).

• Compute output gate $o_t$ by Eq. (4.3).

• Output $h_t$ gated by $o_t$ from memory cell by Eq. (3.5).

• (For frame level prediction) Output the label probability $p_t$ by applying Eq. (3.2) and softmax Eq. (3.3) to $h_t$.

• (For sequence level prediction) Compute discretized SEP \( \left\{ \mathcal{E}_t^n \right| n = 0, 1, \ldots, N \} \) using Eq. (4.7), Eq. (4.8), and Eq. (4.9).

To learn the model parameters of dRNN, we define a loss function to measure the deviation between the target class $c_t$ and $p_t$ at time $t$:

$$\ell(p_t, c_t) = -\log p_{t,c_t}.$$  

If an individual level $c_t$ is given to each frame $t$ in the sequence, we can minimize the cumulative loss over the sequence:

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

For an activity recognition task, the label of activity is often given at the video level, so we mainly consider sequence level prediction. After completing the above recurrent steps for a video sequence, video representation using SEP $h_\tau$ is then computed by Eq. (4.10). The sequence level class probability $p$ is generated by computing the output of dRNN with Eq. (3.2) and applying the softmax function with Eq. (3.3). For a given training label $c$, the dRNNs can be trained by
minimizing the loss function below, i.e.

$$\ell(p, c) = -\log p_c.$$  

The loss function can be minimized by Back Propagation Through Time (BPTT) [4], which unfolds an LSTM model over several time-steps and then runs the back-propagation algorithm to train the model. To prevent back-propagated errors from decaying or exploding exponentially, we use the truncated BPTT according to [4] to learn the model parameters. Specifically, in our model, errors are not allowed to re-enter the memory cell once they leave it through the DoS nodes.

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

$$\frac{\partial i_t}{\partial v_{t-1}} = 0, \frac{\partial f_t}{\partial v_{t-1}} = 0, \frac{\partial o_t}{\partial v_t} = 0,$$

and

$$\frac{\partial i_t}{\partial a_{t-1}} = 0, \frac{\partial f_t}{\partial a_{t-1}} = 0, \frac{\partial o_t}{\partial a_t} = 0,$$

where $\approx$ stands for the truncated derivatives. The details about the implementation of the truncated BPTT can be found in [4].

**EXPERIMENTS AND RESULTS**

We compare the performance of the proposed method with the state-of-the-art LSTM and non-LSTM methods present in the existing literature on human activity datasets.
DATASETS

The proposed method is evaluated on individual human action recognition datasets: KTH and MSR Action3D datasets, as well as group and crowd activity recognition datasets: NUS-HGA and Violent-Flow datasets.

KTH dataset. We choose KTH dataset [60] because 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: indoors, outdoors, outdoors with scale variation, and outdoors with different clothes. The sequences are captured over a 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 the KTH-1 dataset and one second for the 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 [61] 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 dataset. We follow a similar experiment setting from [27], 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 [61] because all actions are evaluated together and the chance of confusion is much higher.
**NUS-HGA dataset.** We choose the NUS-HGA dataset [23] as it is a well-collected benchmark dataset for evaluating group activity recognition techniques. The NUS-HGA dataset includes 476 video clips covering six group activity classes: Fight, Gather, Ignore, RunInGroup, StandTalk, and WalkInGroup. Each instance involves 4-8 persons. The sequences are captured over different backgrounds with a static camera recording of 25 frames per second. Each video clip has a resolution of $720 \times 576$ and lasts around 10 seconds. We follow the experiment setting from [1] and evaluate our method via five-fold cross-validation.

**Violent-Flows dataset.** The Violent-Flows (VF) dataset [21] is a real-world video footage of crowd violence, along with standard benchmark protocols designed for violent/non-violent classification. The Violent-Flows dataset includes 246 real-world videos downloaded from YouTube. The shortest clip duration is 1.04 seconds, the longest clip is 6.52 seconds, and the average length of the video is 3.6 seconds. We follow the standard 5-fold cross-validation protocol in [21].

**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 [12]. The size of the descriptor is 1,000 per cell of grid, and there are 58 such cells in each frame, yielding a 58,000-dimensional feature vector per frame. We apply PCA to reduce the dimension to 2000, retaining 90% of energy among the principal components, to construct a compact input into the dRNN model.

For the 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 skele-
ton 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 [62]. 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 the 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 center 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.

Same as the KTH dataset, we choose the HOG3D feature for NUS-HGA and Violent-Flows datasets and similar procedures of feature extraction are performed on these two datasets. After PCA is applied, NUS-HGA and Violent-Flows have a feature dimension of 300 and 500, respectively.

ARCHITECTURE AND TRAINING

The architecture of the dRNN models trained on the above datasets is shown in Table 4.1. For the sake of fair comparison, we adopt the same architecture for both orders of dRNN models. We can see that the number of memory cell units is smaller than the input units on all datasets. This can be interpreted as follows. The sequence of a human activity video often forms a continuous pattern embedded in a low-dimensional manifold of the input space. Thus, a lower-dimension state space is sufficient to capture the dynamics of such patterns. The number of output units corresponds to
the number of classes in the datasets.

Table 4.1: Architectures of the proposed model used on the datasets.

<table>
<thead>
<tr>
<th></th>
<th>KTH</th>
<th>MSR</th>
<th>NUS</th>
<th>VF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Units</td>
<td>2000</td>
<td>583</td>
<td>300</td>
<td>500</td>
</tr>
<tr>
<td>State Units</td>
<td>1500</td>
<td>400</td>
<td>200</td>
<td>400</td>
</tr>
<tr>
<td>Output Units</td>
<td>6</td>
<td>20</td>
<td>6</td>
<td>2</td>
</tr>
</tbody>
</table>

During training, the learning rate of the BPTT algorithm is set to 0.0001. The objective loss continuously decreases over 50 epochs. Usually, after 40 epochs, the dRNN model begins to converge.

RESULTS ON KTH DATASET

There are several different evaluation protocols used on the KTH dataset in literature. This can result in as large as 9% difference in performance across different experimental protocols as reported in [63]. For a fair comparison, we follow the cross-validation protocol [29], 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 trials.

Table 4.2: Performance comparison of LSTM models on the KTH-1 and KTH-2 datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>LSTM Model</th>
<th>LHS</th>
<th>SEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>KTH-1</td>
<td>conventional LSTM + HOF [7]</td>
<td>87.78</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>conventional LSTM + HOG3D</td>
<td>89.93</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>1&lt;sup&gt;st&lt;/sup&gt;-order dRNN + HOG3D</td>
<td>93.28</td>
<td>93.94</td>
</tr>
<tr>
<td></td>
<td>2&lt;sup&gt;nd&lt;/sup&gt;-order dRNN + HOG3D</td>
<td>93.96</td>
<td>94.78</td>
</tr>
<tr>
<td>KTH-2</td>
<td>conventional LSTM + HOF [29]</td>
<td>87.78</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>conventional LSTM + HOG3D</td>
<td>87.32</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>1&lt;sup&gt;st&lt;/sup&gt;-order dRNN + HOG3D</td>
<td>91.98</td>
<td>92.73</td>
</tr>
<tr>
<td></td>
<td>2&lt;sup&gt;nd&lt;/sup&gt;-order dRNN + HOG3D</td>
<td>92.12</td>
<td>92.93</td>
</tr>
</tbody>
</table>

First, we compare the dRNN model with the conventional LSTM model in Table 4.2. Here we report the cross-validation accuracy on both KTH-1 and KTH-2 datasets. In addition, Fig. 4.2
shows the confusion matrix obtained by the 2\textsuperscript{nd}-order dRNN model on the KTH-1 dataset. This confusion matrix is computed by averaging over five trials in the above cross-validation protocol. The performance of the conventional LSTM has been reported in literature [7, 29]. 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.

Figure 4.2: Confusion Matrices on the KTH-1 dataset obtained by the 2\textsuperscript{nd}-order dRNN model using LHS (left) and SEP (right).

For the dRNN model, we report the accuracy with up to the 2\textsuperscript{nd}-order of DoS. Table 4.2 shows that with the same HOG3D feature, the proposed dRNN models outperform the conventional LSTM model, which demonstrates the effectiveness of dRNN. Utilizing DoS, dRNN explicitly models the change in information gain caused by the salient motions between the successive frames, thus can benefit the recognition process. For both the first and second orders of dRNNs, SEP consistently achieves better performance for KTH-1 and KTH-2 datasets. Based on DoS, SEP selected the most discriminative hidden states over all time steps, thus can generate more comprehensive represen-
tation than the hidden state at the last time-step. In the meanwhile, we can see that the $2^{nd}$-order dRNN yields a better accuracy than its $1^{st}$-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 even 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 problems, making the performance stop improving or even begin to degenerate after the order of DoS reaches a certain number. Therefore, for most practical applications, the first two orders of dRNN should be sufficient.

Baccouche et al. [29] reported an accuracy of 94.39% and 92.17% on KTH-1 and KTH-2 datasets, respectively. It is worth noting that they used a combination of 3DCNN and LSTM, where 3DCNN plays a crucial role in reaching such performance. Actually, the 3DCNN model alone can reach an accuracy of 91.04% and 89.40% on KTH-1 and KTH-2 datasets reported in [29]. On the contrary, they reported that the LSTM with Harris3D feature only achieved 87.78% on KTH-2, as compared with 92.93% accuracy obtained by $2^{nd}$-order dRNN with SEP using HOG3D feature. In Table 4.2., under a fair comparison with the same feature, the dRNN models of both orders outperform their LSTM counterpart with the same HOG3D feature.

![Figure 4.3: Frame-by-frame prediction of action category over time with LSTM and SEP dRNNs. Best viewed in color.](image-url)
To support our motivation for learning LSTM representations based on the dynamic change of states evolving over frames, we illustrate the predictions over time in Fig. 4.3. 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 DoS is better than 1st-order of DoS in learning the salient features.

Table 4.3: Cross-validation accuracy over five trials obtained by the other compared algorithms on KTH-1 and KTH-2 datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Rodrigues et al. [64]</td>
<td>81.50</td>
</tr>
<tr>
<td>KTH-1</td>
<td>Jhuang et al. [65]</td>
<td>91.70</td>
</tr>
<tr>
<td></td>
<td>Schidler et al. [66]</td>
<td>92.70</td>
</tr>
<tr>
<td></td>
<td>3DCNN [29]</td>
<td>91.04</td>
</tr>
<tr>
<td></td>
<td>3DCNN + LSTM [29]</td>
<td>94.39</td>
</tr>
<tr>
<td>KTH-2</td>
<td>Ji et al. [67]</td>
<td>90.20</td>
</tr>
<tr>
<td></td>
<td>Taylor et al. [68]</td>
<td>90.00</td>
</tr>
<tr>
<td></td>
<td>Laptev et al. [57]</td>
<td>91.80</td>
</tr>
<tr>
<td></td>
<td>Dollar et al. [69]</td>
<td>81.20</td>
</tr>
<tr>
<td></td>
<td>3DCNN [29]</td>
<td>89.40</td>
</tr>
<tr>
<td></td>
<td>3DCNN + LSTM [29]</td>
<td>92.17</td>
</tr>
</tbody>
</table>

We also show the performance of the other non-LSTM state-of-the-art approaches in Table 4.3. 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 that do not satisfy these assumptions. In contrast, the proposed dRNN model is a general-purpose model, not being tailored to a specific type of action sequences. This also makes it competent on 3D action recognition and even more challenging tasks such as group and crowd activity recognition, which we will show below.
RESULTS ON MSR ACTION3D DATASET

Table 4.4 and Table 4.5 compare the results on MSR Action3D dataset. Fig. 4.4 shows the confusion matrix by the 2\textsuperscript{nd}-order dRNN model with SEP. The results are obtained by following exactly the same experimental setting in [27], in which half of the actor subjects are used for training and the rest are used for testing. This is in contrast to another evaluation protocol in literature [61] which splits across 20 action classes into three subsets and performs the evaluation within each individual subset. This evaluation protocol is more challenging because it is evaluated over all 20 action classes with no common subjects in training and testing sets.

Table 4.4: Performance comparison of LSTM models on the MSR Action3D dataset.

<table>
<thead>
<tr>
<th>LSTM Model</th>
<th>LHS</th>
<th>SEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>conventional LSTM</td>
<td>87.78</td>
<td>89.12</td>
</tr>
<tr>
<td>1\textsuperscript{st}-order dRNN</td>
<td>91.40</td>
<td>92.74</td>
</tr>
<tr>
<td>2\textsuperscript{nd}-order dRNN</td>
<td>92.03</td>
<td>92.91</td>
</tr>
</tbody>
</table>

Table 4.5: Performances of the other compared methods on MSR Action3D dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actionlet Ensemble</td>
<td>88.20</td>
</tr>
<tr>
<td>HON3D [70]</td>
<td>88.89</td>
</tr>
<tr>
<td>DCSF [71]</td>
<td>89.30</td>
</tr>
<tr>
<td>Lie Group [72]</td>
<td>89.48</td>
</tr>
</tbody>
</table>

From the results, all the dRNN models outperform the conventional LSTM algorithm with the same feature. Also, dRNN models using SEP outperform dRNNs using LHS. By taking into consideration of not only the hidden state at the last time-step, which might have lost certain information from early frames due to exponential decay, dRNNs with SEP can generate more informative representation.

In the meanwhile, dRNN models perform competitively as compared with the other algorithms. We notice that the Super Normal Vector (SNV) model [73] has reported an accuracy of 93.09% on
Figure 4.4: Confusion Matrix on the MSR Action3D dataset obtained by the $2^{nd}$-order dRNN model using SEP.

the 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 the 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. In the following sections, we demonstrate the application of dRNN on even more challenging tasks of multi-person activity recognition.

RESULTS ON NUS-HGA DATASET

For the sake of fair comparison, we follow [1] and evaluate our method via a five-fold cross validation. We compare our method with conventional LSTM and previous group activity recognition methods in Table 4.6 and Table 4.7, respectively.

Table 4.6: Performance comparison of LSTM models on the NUS-HGA dataset.

<table>
<thead>
<tr>
<th>LSTM Model</th>
<th>LHS</th>
<th>SEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>conventional LSTM</td>
<td>93.48</td>
<td>94.71</td>
</tr>
<tr>
<td>1st-order dRNN</td>
<td>96.36</td>
<td>97.43</td>
</tr>
<tr>
<td>2nd-order dRNN</td>
<td>97.37</td>
<td>98.95</td>
</tr>
</tbody>
</table>

Table 4.7: Performances of the other compared methods on NUS-HGA dataset. MF indicates motion feature fusion and MAF indicates motion and appearance feature fusion.

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ni et al. [23]</td>
<td>73.50</td>
</tr>
<tr>
<td>Zhu et al. [33]</td>
<td>87.00</td>
</tr>
<tr>
<td>Cho et al. [2]</td>
<td>96.03</td>
</tr>
<tr>
<td>Cheng et al. [1] (MF)</td>
<td>93.20</td>
</tr>
<tr>
<td>Cheng et al. [1] (MAF)</td>
<td>96.20</td>
</tr>
</tbody>
</table>

As shown in Table 4.6, all the dRNN models outperform the conventional LSTM model. This again shows the effectiveness of explicitly analyzing dynamic structures between successive frames using DoS. Comparing between the dRNN models, 2nd-order dRNN achieves better performance...
than 1\textsuperscript{st}-order dRNN. As mentioned above, 2\textsuperscript{nd}-order dRNN analyzes not only the velocity, but also the acceleration information between internal states, which enlarge its ability for understanding more complex spatio-temporal dynamics. For the same order of dRNN, SEP outperforms LHS more than 1\%, which can be viewed as a large margin considering that the performance is already very high. This again demonstrates the effectiveness of SEP.

In addition, dRNN models generally achieve better performance than other non-LSTM methods. It is worth noting that 2\textsuperscript{nd}-order dRNN with SEP achieves almost 99\% recognition accuracy, outperforming [1] by 2.75\%. Traditional solutions for group activity algorithms need human supervision to acquire accurate human object trajectories from the videos. According to [1], they acquired human bounding boxes using existing tracking tools, which requires manual annotation for bounding box initialization. This constraint prevents their method from being used in automatic or real-time applications. On the other hand, dRNN models can outperform these traditional methods without human manual annotation, enabling much broader applications for group behavior analysis.

\textit{RESULTS ON VIOLENT-FLOWS DATASET}

To evaluate our method on the Violent-Flows dataset, we follow the standard 5-fold crossvalidation protocol in [21] and report the results in terms of mean accuracy in Table 4.8 and Table 4.9.

From Table 4.8, we can see that both the first and second orders of dRNN models outperform the traditional LSTM. This indicates that for even more complex scenarios such as crowd behavior, DoS can still model the dynamical structures present in the video sequences more effectively. 2\textsuperscript{nd}-order of dRNN consistently achieves slightly better performance than 1\textsuperscript{st}-order of dRNN. As mentioned before, high orders of dRNNs have more model parameters to be trained. For a relatively small dataset such as Violent-Flows, the overfitting problem tends to be more severe. On the other hand, Table 4.8 shows that the proposed SEP pooling strategy further boosts the perfor-
Table 4.8: Performance comparison of LSTM models on the Violent-Flows dataset.

<table>
<thead>
<tr>
<th>LSTM Model</th>
<th>LHS</th>
<th>SEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>conventional LSTM</td>
<td>83.78</td>
<td>84.92</td>
</tr>
<tr>
<td>1\textsuperscript{st}-order dRNN</td>
<td>86.37</td>
<td>87.83</td>
</tr>
<tr>
<td>2\textsuperscript{nd}-order dRNN</td>
<td>86.98</td>
<td>87.84</td>
</tr>
</tbody>
</table>

Table 4.9: Performances of the other compared methods on Violent-Flows dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Violent Flows [21]</td>
<td>81.30</td>
</tr>
<tr>
<td>Common Measure [74]</td>
<td>81.50</td>
</tr>
<tr>
<td>Hist of Tracklet [50]</td>
<td>82.30</td>
</tr>
<tr>
<td>Substantial Derivative [75]</td>
<td>85.43</td>
</tr>
<tr>
<td>Holistic Features [52]</td>
<td>85.53</td>
</tr>
</tbody>
</table>

mances, which again demonstrates the effectiveness of State Energy Profile over the Last Hidden State strategy. In addition, dRNN models outperform the other non-LSTM state-of-the-art methods too, as seen in Table 4.9.

**SEP VS. OTHER POOLING TECHNIQUES**

To show the advantage of the SEP pooling method, we compare the results of SEP against other pooling strategies, e.g. Last-Hidden-State (LHS), Mean-Pooling, Max-Pooling, on KTH-1 and NUS-HGA datasets in Table 4.10. For a fair comparison, all the results are acquired using 2\textsuperscript{nd}-order dRNN model. LHS, used in our prior work, simply employs the hidden state at the last time-step to optimize the parameters of dRNN neural networks. Mean pooling/max pooling indicates that mean pooling/max pooling is performed over all the hidden states to generate a representation of the input video. These three methods serve as baselines to compare with our proposed SEP pooling method.

The LHS method achieves the lowest performance compared to other methods. Since the information learned from previous time steps decays slowly over a very long sequence, the last time-step
hidden state is unable to summarize the most distinguishing motion patterns from successive states. Mean pooling of all hidden states achieves a slightly better performance than using only the last hidden states, as it statistically averages and summarizes the information collected from all states. In the process of smoothing, mean pooling might lose the salient information acquired by the neural network. Max pooling, on the contrary, is good at selecting discriminative and salient information from human behaviors. Because of this, max pooling over hidden states gets better results than LHS and mean pooling methods. On the other hand, max pooling is subject to motion noise. If a noisy and strong motion happens during the video sequence, max pooling could misclassify the human activity.

SEP achieves the best performance over other pooling strategies, outperforming dRNN using LHS by about 1% and 1.5% on KTH-1 and NUS-HGA datasets, respectively. SEP detects the hidden states corresponding to the most intense motions through the energy of DoS. On top of it, mean pooling is applied to suppress the unwanted noise. As a result, SEP can generate both discriminative and reliable representation of states.

Table 4.10: Performance comparison of SEP against different pooling strategies on the KTH-1 and NUS-HGA datasets. All the results are generated by 2nd-order dRNN.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>KTH-1</td>
<td>LHS</td>
<td>93.96</td>
</tr>
<tr>
<td></td>
<td>Mean-Pooling</td>
<td>94.11</td>
</tr>
<tr>
<td></td>
<td>Max-Pooling</td>
<td>94.29</td>
</tr>
<tr>
<td></td>
<td>SEP</td>
<td><strong>94.78</strong></td>
</tr>
<tr>
<td>NUS-HGA</td>
<td>LHS</td>
<td>97.37</td>
</tr>
<tr>
<td></td>
<td>Mean-Pooling</td>
<td>97.63</td>
</tr>
<tr>
<td></td>
<td>Max-Pooling</td>
<td>97.89</td>
</tr>
<tr>
<td></td>
<td>SEP</td>
<td><strong>98.95</strong></td>
</tr>
</tbody>
</table>

To better understand the superiority of State Energy Profile over other pooling methods, we show the confusion matrices of 2nd-order dRNNs using LHS, mean-pooling, max-pooling and SEP methods on the NUS-HGA dataset in Fig. 4.5. The matrices lead to several interesting discoveries.
The LHS method does not achieve a 100% recognition rate in any class. 3.85% of RunInGroup group behaviors are wrongly classified as WalkInGroup by the LHS method, while all other techniques fully identify RunInGroup behavior correctly. This indicates that the last time-step hidden state cannot sufficiently utilize the DoS, even though DoS better describes the motion pattern.
The mean-pooling method has a large error while dealing with the group behavior Ignore. This can be explained that mean pooling serves as noise filtering: the information of one person walking by and being ignored by others could be smoothed by mean pooling. The unique motion pattern of the person who is excluded from the group tends to be smoothed by the mean-pooling method, thus, the neural networks get confused by Ignore, WalkInGroup, Gather and Fight behaviors. Our SEP method mitigates the influence of this problem by choosing only hidden states corresponding to intensive and informative motions to generate the final representations.

Max pooling wrongly classifies StandTalk into Fight by a significant error rate of 4.41%. Max pooling can select the salient information over hidden states through all time-steps, but is subject to motion noise. When people stand and talk in a group, there might be interactions between them. Max pooling detects such motions and incorrectly identifies them as Fight. StandTalk behavior in the NUS-HGA dataset gives an example of one person raising arm when he is talking. Short and fast behavior could lead to misclassification. The SEP method performs mean pooling over the candidate hidden states, which largely decreases the chance of incorrect recognition by smoothing the noisy motions.

SEP recognizes the Ignore, RunInGroup and StandTalk behaviors perfectly. This demonstrates that SEP, analyzing the energy of DoS, integrates the virtues of mean pooling and max pooling and minimizes the disadvantages of the two techniques.

**RUN-TIME EFFICIENCY**

We performed our experiments on a personal computer with an Intel Core i7-6700K CPU, Nvidia GeForce GTX 1080 Ti GPU, and 32GB of RAM. From Table 4.11., we can see that the training convergence takes reasonable amount of time.
Adding up run time for feature extraction and testing per example, dRNN takes 0.414 second averagely to recognize an action on KTH dataset. For the NUS-HGA and BEHAVE datasets, it takes only 0.153 and 0.138 second to identify a group activity, respectively, which fully meets the requirement for real-time applications.

Table 4.11: Run-time efficiency on the datasets.

<table>
<thead>
<tr>
<th></th>
<th>KTH</th>
<th>MSR</th>
<th>NUS (s)</th>
<th>VF (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature extraction</td>
<td>168</td>
<td>-</td>
<td>62</td>
<td>82</td>
</tr>
<tr>
<td>Feature per example</td>
<td>0.28</td>
<td>-</td>
<td>0.13</td>
<td>0.13</td>
</tr>
<tr>
<td>Training convergence</td>
<td>10340</td>
<td>3430</td>
<td>2790</td>
<td>1090</td>
</tr>
<tr>
<td>Testing per example</td>
<td>0.134</td>
<td>0.083</td>
<td>0.023</td>
<td>0.081</td>
</tr>
</tbody>
</table>

SUMMARY

In this chapter, we present a new family of differential Recurrent Neural Networks (dRNNs) that extend the Long Short-Term Memory (LSTM) structure by modeling the dynamics of states evolving over time. The conventional LSTM is a special form and base model of the proposed dRNNs. 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 human activities. Based on the energy analysis of Derivative of States (DoS), we further introduce the SEP pooling strategy which can select the most salient hidden states and generate more discriminative representation for video sequences. Experimental results on human action, group activity, and crowd behavior datasets demonstrate that the dRNN model outperforms the conventional LSTM model. Armed with the SEP pooling strategy, the dRNN model can further enhance the performance. In the meantime, even in comparison with the other state-of-the-art approaches based on strong assumptions about the motion structure of actions being studies, the proposed general-purpose dRNN model still demonstrates much competitive performance on both single-person and multi-person activity problems.
CHAPTER 5: CONVOLUTIONAL DIFFERENTIAL RECURRENT NEURAL NETWORKS

Parts of this chapter have been presented in IEEE FG [76] (Copyright ©2017 IEEE), IEEE ISM [77] (Copyright ©2016 IEEE) and World Scientific IJSC [78] (Copyright ©2018 World Scientific).

With the growth of crowd phenomena in the real world, crowd scene understanding is becoming an important task in anomaly detection and public security. Visual ambiguities and occlusions, high density, low mobility, and scene semantics, however, make this problem a great challenge. In this paper, we propose an end-to-end deep architecture, Convolutional Differential Recurrent Neural Networks (CDRNN), for crowd scene understanding. CDRNN consists of VGG-16/GoogleNet Inception V3 convolutional neural networks (CNN) and stacked differential long short-term memory (DLSTM) networks. Different from traditional non-end-to-end solutions that separate the steps of feature extraction and parameter learning, CDRNN utilizes a unified deep model to optimize the parameters of CNN and RNN hand in hand. It thus has the potential of generating a more harmonious model. The proposed architecture takes sequential raw image data as input, and does not rely on tracklet or trajectory detection. It thus has clear advantages over the traditional flow-based and trajectory-based methods, especially in challenging crowd scenarios of high density and low mobility. Taking advantage of the semantic representation of CNN and the memory states of LSTM, CDRNN can effectively analyze both the crowd scene and motion information. Existing LSTM-based crowd scene solutions explore deep temporal information and are claimed to be “deep in time”. CDRNN, however, models the spatial and temporal information in a unified architecture and achieves “deep in space and time”. Extensive performance studies on the Violent-Flows and CUHK Crowd datasets show that the proposed technique significantly outperforms state-of-the-art methods.
With the increase of the world population and various human activities, crowd phenomena are growing more rapidly than ever before. To ensure public security and safety, understanding crowd scenes, especially abnormal crowd behaviors and emotions, is becoming increasingly urgent and important [19, 79]. Although human observers are able to monitor behavior patterns and detect unusual crowd activities in the surveillance area, the wide use of video surveillance in the last decade has led to huge amounts of video data that are beyond the capability of human observers [52]. In addition, psychophysical research suggests that humans’ ability to monitor simultaneous signals deteriorates after long-term monitoring because extremely crowded scenes exhibit excessive numbers of individuals and their activities [19]. These issues make humans poor-performing observers of crowd interactions and anomaly events.

In the last decade, researchers from the computer vision community have shown much interest in developing automated crowd scene understanding systems. Video analysis for uncrowded scenes usually involves object detection, object tracking, and behavior recognition. Such solutions, how-
ever, are not suitable for crowded scenes; and special considerations must be taken into account. As a crucial basis, appropriate feature representation for crowded scenes is necessary. In terms of representation level, previous crowd features can be divided into the following three categories [19]: flow-based features, local spatio-temporal features, and trajectory/tracklet features.

Flow-based features are extracted densely on the pixel level and are suitable for highly dense crowded scenes when tracking each person in the videos is impracticable. Several flow-based features have been presented in recent years [80, 39, 41, 81]. Their methods achieved success in addressing dense and complex crowd flows by avoiding tracking at the macroscopic level. However, flow-based features ignore the scene information and tend to fail in crowd videos with less mobility.

Local spatio-temporal features exploit the dense local motion features created by the subjects and model their spatio-temporal relationships to represent the underlying intrinsic structure formed in the video. Some related works use histogram functions [44, 45], and spatio-temporal gradients [42, 43]. Local spatio-temporal features, however, analyze local features of crowd dynamics and are sub-optimal for complex crowd behaviors with long-range dependency.

Most recent methods for crowd scene understanding mostly analyze crowd activities based on motion features extracted from trajectories/tracklets of objects [22, 49, 50, 46, 47, 48, 51]. The trajectory/tracklet feature contains more semantic information, but the accuracy of trajectories/tracklets dictates the performance of crowd scene analysis. In extremely crowded areas, tracking algorithms could fail and generate inaccurate trajectories.

The disadvantages of the above existing crowd representation methods motivate us to explore a new representation for crowded scenes, which is simple yet can still maintain the raw information of the source video as much as possible. We are inspired by the success of convolutional neural networks [20] to explore the use of raw input data for crowd scenes.
Human crowds exhibit complex temporal dynamics and psychological characteristics [82]. To model such complex dynamics, Long Short-Term Memory (LSTM) [4] was proposed to learn the dynamical evolution of a long sequence. LSTM possesses the potential to model various sequential 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 human activities [18]. Alexandre et al. [83] viewed the human trajectory prediction in crowded spaces as a sequence generation task and used an LSTM model to learn general human movement and predict their future trajectories. Su et al. [49] explored Coherent LSTM to model the nonlinear characteristics and spatio-temporal motion patterns in crowd behaviors. Using two stacked LSTMs, their model is better at learning deep temporal information and they claimed their model to be “deep in time”. In this paper, we aim to construct a unified deep model exploring spatial and temporal information concurrently for crowd scene understanding and investigate the possibility of achieving “deep in space and time”.

To address the challenge of video data processing, we have introduced the dynamic temporal quantization [84] and differential long-short term memory [56] in our early work to achieve a fixed-length representation of video data with varied length. However, these two methods are not designed for crowd scene analysis. In addition, neither of them is an end-to-end solution and does not fully explore the capability of spatial and temporal representation in deep neural networks. In this paper, we propose a novel end-to-end Convolutional Differential Recurrent Neural Networks (CDRNN) network for crowd scene understanding. The architecture connects VGG-16/GoogleNet Inception V3 [85] convolutional neural networks and stacked Differential Long Short-Term Memory [18]. The deep neural network directly takes the sequential raw image data as input, and outputs the predicted crowd scene label. It differs from the three existing categories of feature representations by directly using the raw image sequences. The proposed technique has the following advantages over existing methods.
Firstly, when dealing with highly dense crowd scenes, trajectory/tracklet methods tend to perform poorly. CDRNN has no such problem because it does not rely on trajectory detection. Secondly, flow-based and trajectory-based methods assume crowd mobility when extracting the flow and trajectory representations. The convolutional neural network layers in the proposed CDRNN can model the scene semantics and perform reasonable analysis and do not require motion information from the crowd. Thirdly, different from existing LSTM-based crowd scene solutions which are “deep in time”, CDRNN models the spatial and temporal information in a unified architecture and achieves “deep in space and time”.

We extensively evaluate the performance of the proposed deep architecture on two public group and crowd understanding datasets, NUS-HGA [23], BEHAVE [86], Violent-Flows [21] and CUHK Crowd [22]. Experimental results show that the proposed technique significantly outperforms the conventional flow-based and trajectory or tracklet-based methods by a great margin. We also show that our CDRNN model can outperform the LSTM-based methods by achieving “deep in both space and time”.

THE PROPOSED MODEL

In this section, we elaborate upon our proposed Convolutional Differential Recurrent Neural Networks (CDRNN) framework for crowd scene analysis. CDRNN connects GoogleNet Inception V3 [85] and stacked DLSTMs [4] into an end-to-end model. Fig. 5.1 shows the diagram of the proposed CDRNN architecture. As shown in Fig. 5.1, CDRNN takes the sequential raw RGB image data as input and outputs the predicted crowd scene class. The model is composed of first few convolutional and max-pooling layers, ten mixed Inception blocks with the last block Mixed 9 containing two identical inception blocks, one average-pooling layer, dropout and fully-connected layer, three stacked LSTMs and lastly, a softmax layer. In CDRNN, each frame of a crowd video is
first processed by VGG-16/GoogleNet Inception V3 convolutional neural network (CNN) to generate frame-level representations, which are then allowed to flow between time-steps using stacked DLSTMs. By doing so, CDRNN possesses the potential of analyzing the spatial and temporal information in a unified model and achieves “deep in space and time”.

CONVOLUTIONAL NEURAL NETWORKS

In the proposed ConvDLSTM model, we adopt VGG-16 and GoogleNet Inception V3 as the prototype of the convolutional neural network part. The Inception micro-architecture was first introduced by Szegedy et al. [87]. The goal of the inception module is to act as a multi-level feature extractor by computing $1 \times 1$, $3 \times 3$, and $5 \times 5$ convolutions within the same module of the network. The output of these filters are then stacked along the channel dimension before being fed into the next layer in the network. The Inception V3 architecture generally includes the following disciplines. It reduces the number of convolutions to maximum $3 \times 3$ blocks. In addition, it increases the general depth of the networks. Lastly, Inception V3 uses the width increase technique at each layer to improve feature combination. The Inception V3 architecture further boosts ImageNet classification accuracy. In the meantime, it keeps the number of parameters for the whole architecture smaller, enabling the high efficiency for training and testing the network.

GoogleNet Inception V3 originally consists of first few regular convolutional and max-pooling layers, ten mixed Inception blocks and one average-pooling, dropout and fully-connected layer. For our proposed ConvDLSTM model, we remove the top fully-connected layer FC-1000 since it corresponds to the 1000 ImageNet class label probabilities, which do not specifically correlate with our crowd scene analysis tasks. In Fig. 5.1, due to page limit, we use one blue block to demonstrate the same inception blocks. For more details for the Inception V3 architecture, please refer to [85].
For notational simplicity, we refer to the modified GoogleNet Inception V3 network as \( x = V(m) \), which takes the raw RGB data of an image \( m \) as input and produces a 2048-dimension representation. Denote \( \{m_1, m_2, \ldots, m_T\} \) as a crowd image sequence of length \( T \), where \( m_t \) indicates the image frame at time \( t \). The sequential image data for a crowd video are passed through the Inception V3 network frame by frame to produce \( \{x_1, x_2, \ldots, x_T\} \), which serve as the input sequence to stacked DLSTMs. At time \( t \), modified Inception V3 takes the input image \( m_t \) and computes \( x_t \) via:

\[
x_t = V(m_t).
\]

(5.1)

Note that the parameters of the convolutional neural networks are shared across all time-steps. This is one of the key differences from the recurrent neural networks.

**DIFFERENTIAL LONG SHORT-TERM MEMORY**

Due to exponential decay, traditional RNNs are limited in learning long-term sequences. Hochreiter et al. [4] designed Long Short-Term Memory (LSTM) to exploit long-range dependency. According to a recent study, the Derivative of States (DoS) in differential long short-term memory (DLSTM) [8] can explicitly model spatio-temporal structure and better learn salient patterns within. Replacing internal state with the DoS in the gate units in LSTM, the DLSTM has the following updated equations:

(i) Input gate \( i_t \) regulates how much input information would enter the memory cell to affect its internal state \( s_t \) at time-step \( t \). The activation of the input gate has the following recurrent form:

\[
i_t = \sigma(W_{id} \frac{ds_t}{dt} + W_{ih} h_{t-1} + W_{ix} x_t + b_i),
\]

(5.2)

where \( \sigma(\cdot) \) stands for a sigmoid activation function in the range \([0, 1]\).
(ii) Forget gate $f_t$ modulates the contribution of the previous state $s_{t-1}$ to the current state. It is defined by the following equation as:

$$f_t = \sigma(W_{fd}\frac{ds_{t-1}}{dt} + W_{fh}h_{t-1} + W_{fx}x_t + b_f). \tag{5.3}$$

The internal state $s_t$ of each memory cell can be updated using the input and forget gate units, which is shown in the equation below:

$$s_t = f_t \odot s_{t-1} + i_t \odot \tilde{s}_t, \tag{5.4}$$

where $\odot$ stands for element-wise product. Pre-state $\tilde{s}_t$ is defined as:

$$\tilde{s}_t = tanh(W_{sh}h_{t-1} + W_{sx}x_t + b_s).$$

(iii) Output gate $o_t$ gates the information output from a memory cell and it affects the future states of DLSTM cells. The output gate can be expressed as:

$$o_t = \sigma(W_{od}\frac{ds_t}{dt} + W_{oh}h_{t-1} + W_{ox}x_t + b_o). \tag{5.5}$$

Then the hidden state of a memory cell is output as:

$$h_t = o_t \odot tanh(W_{hs}s_t + b_h). \tag{5.6}$$

Fig. 5.2 gives an illustration of the architecture of DLSTM at time-step $t$. 
Figure 5.2: An illustration of LSTM architecture at time $t$.

**STACKED DLSTMS**

As shown in Fig. 5.1, to investigate the temporal information in deep architectures, we adopt stacked DLSTMs up to three layers for ConvDLSTM. The advantage of stacked DLSTMs over a single layer of DLSTM is as follows: for a single layer of DLSTM, the input of DLSTM is the output of CNN, which contains only the spatial information; for deeper layers of DLSTMs in stack DLSTM setup, they take the output of previous DLSTMs as input sequences. Such inputs as each time-step contain both spatial and temporal information. In other words, the hidden states from the previous layer of LSTM serve as the input sequence to the next layer, higher LSTM layers can capture abstract concepts in the sequences, which helps the whole system to better interpret the complex scene semantics and crowd dynamics.

For the last layer of DLSTMs, we consider mean-pooling, for the hidden states to generate a video-
level representation. The last time-step hidden state was frequently used to denote a sequence level representation, e.g. [18]. As LSTMs process the input frames sequentially, hidden state information learned from previous frames decays gradually over a very long sequence. For crowd scene videos with a large number of frames, it tends to be sub-optimal to only use the last time-step hidden state to learn the parameters of the neural networks. Max pooling is better at selecting salient signals and often generates a more discriminative representation. The problem of max pooling is it tends to be affected by motion noise. Mean pooling averages all time-step hidden states, and statistically summarizes the information collected from all previous frames and thus has a more stable representation of the sequences.

Using the mean pooling method, we acquire $h_{\tau}$ from last layer of stacked DLSTMs as a video-level representation of a crowd video. $h_{\tau}$ denotes a 1-of-$k$ encoding of the confidence scores on $k$ classes of crowd scenes. The confidence scores are then transformed into a vector of probabilities $p$ by the softmax function:

$$p_c = \frac{\exp(h_{T,c})}{\sum_{m=1}^{k} \exp(h_{T,m})},$$

(5.7)

where each entry $p_c$ is the probability of input crowd video belonging to class $c \in \{1,2,...,k\}$.

**CDRNN AT TIME-STEP T AND LEARNING STRATEGY**

Given a crowd image sequence $\{m_1,m_2,...,m_T\}$, CDRNN proceeds as shown in Algorithm 1 at time step $t$. After $T$ time steps, $h_{\tau}$ for the last layer of stacked DLSTMs is computed with mean pooling method. Given the video-level class $c$ of this crowd scene, compute crowd scene label probability $p_c$ by applying the softmax function Eq.(3.3). CDRNN can then be trained by minimizing the loss function below, i.e.

$$\ell(p, c) = -\log p_c.$$
The loss function can be minimized by Back Propagation Through Time (BPTT) [88], which unfolds an LSTM model over several time steps and then runs the backpropagation algorithm to train the model. To prevent back-propagated errors from decaying or exploding exponentially, we use truncated BPTT according to Hochreiter et al. [4] to learn the model parameters.

**Algorithm 1 CDRNN at time step t**

1: Given image frame \( m_t \), compute \( x_t \) via Eq.(5.1)
2: Compute input gate activation \( i_t \) and forget gate activation \( f_t \) by Eq. (5.2) and Eq. (5.3)
3: Update state \( s_t \) with \( i_t \) and \( f_t \) by Eq. (5.4)
4: Compute output gate \( o_t \) by Eq. (5.5)
5: Output \( h_t \) gated by \( o_t \) from memory cell by Eq. (5.6)
6: If there exists a deeper layer of LSTM, set \( x_t = h_t \) for the following stacked LSTM and repeat steps 2 - 6

**RATIONALE BEHIND DEEP IN SPACE AND TIME**

CDRNN optimizes the information flow of the spatial and temporal crowd scene dynamics in a unified model thus achieves “deep in space and time”. The CNN layers in CDRNN can model the scene information of crowd videos. LSTM can take advantage of the semantic CNN representation and analyze the long-term temporal dependency of crowd dynamics. Different from traditional non-end-to-end solutions which separate the steps of feature extraction and parameter learning, CDRNN utilizes a unified deep model to optimize the parameters of CNN and LSTM hand in hand. It thus has the potential of generating a more harmonious model. The advantage of stacked LSTMs over a single-layer of LSTM is also intuitive. For stacked LSTMs, the first layer of stacked LSTMs takes the output of CNN. Such input is the same as single-layer LSTM and contains only the spatial information. Deeper layers of stacked LSTMs, however, take the output of previous LSTM as input sequences. Different from single-layer of LSTM, such inputs at each time step contain both spatial and temporal information thus are more comprehensive for understanding complex spatio-temporal structures. To conclude, CDRNN is better at learning crowd scenes than
non-end-to-end structures. In addition, stacked LSTMs possess superiority over a single layer of LSTM. We will demonstrate these claims by experimental results.

EXPERIMENTS

In this section, we extensively evaluate the performances of the proposed method for group activity and crowd scene understanding on four public video datasets: NUS-HGA, BEHAVE, Violent-Flows, and CUHK Crowd.

The NUS-HGA dataset includes 476 video clips covering six group activity classes: Fight, Gather, Ignore, RunInGroup, StandTalk and WalkInGroup. Each instance involves 4-8 persons. The sequences are captured over different backgrounds with a static camera recording of 25 frames per second. Each video clip has a resolution of 720 × 576 and lasts around 10 seconds. The BEHAVE dataset consists of 7 long video sequences ranging from half an hour to an hour with a uniform resolution of 640 × 480. As each sequence includes different classes of group activities, researchers have been manually selecting video clips as group activity instances from the sequences [1]. This dataset provides 10 group activity classes. Six group activity classes, Approach, Fighting, In-Group, RunTogether, Split and WalkTogether, are used for evaluation since the rest only contain a few short sequences [2]. We select 216 video clips from the BEHAVE dataset covering the above six group activities for evaluation. This dataset itself provides bounding box information for each of the group members. The bounding box information is used as input by existing group behavior analysis methods.

The Violent-Flows dataset [21] is a real-world video footage of crowd violence, along with standard benchmark protocols designed for violent and non-violent classification. The Violent-Flows dataset includes 246 real-world videos downloaded from YouTube. The shortest clip duration is
1.04 seconds, the longest clip is 6.52 seconds, and the average length of the videos is 3.6 seconds, with the shortest/longest 1.04/6.52 seconds. The Violent-Flow dataset is designed for five-fold cross-validation. Specifically, the video set is split into five sets: half the videos in each set portray are violent crowd behavior and half non-violent behavior. Five tests are performed: in each test, four sets are used for training and the fifth set is used for testing. The CUHK Crowd dataset [22] consists of 474 crowd videos from over 200 crowded scenes, which were collected from many different environments, e.g. streets, shopping malls, airports, and parks. The videos are manually annotated into 8 different classes and the human trajectories are provided by the dataset. Details of 8 different classes of the CUHK Crowd dataset are shown in Table 5.1.

Table 5.1: List of Crowd Video Classes for the CUHK Crowd dataset.

<table>
<thead>
<tr>
<th>Class Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Highly mixed pedestrian walking</td>
</tr>
<tr>
<td>2</td>
<td>Crowd walking following a mainstream and well organized</td>
</tr>
<tr>
<td>3</td>
<td>Crowd walking following a mainstream but poorly organized</td>
</tr>
<tr>
<td>4</td>
<td>Crowd merge</td>
</tr>
<tr>
<td>5</td>
<td>Crowd split</td>
</tr>
<tr>
<td>6</td>
<td>Crowd crossing in opposite directions</td>
</tr>
<tr>
<td>7</td>
<td>Intervened escalator traffic</td>
</tr>
<tr>
<td>8</td>
<td>Smooth escalator traffic</td>
</tr>
</tbody>
</table>

EXPERIMENT SETTING AND TRAINING STRATEGY

To achieve better performance, the CDRNN architecture is initialized with pre-trained parameters. We initialize GoogleNet Inception V3 in CDRNN with parameters [85] trained on ImageNet in the ILSVRC-2014 competition. To initialize the parameters of stacked DLSTMs, we first freeze the Inception V3 weights in CDRNN and then train the model on unaugmented crowd videos. The number of memory cells for each DLSTM layer is determined as 1024 determined by cross-validation. RMSprop adaptive learning optimizer $1_e^{-3}$ is employed for

pre-training the stacked DLSTMs. The original dataset is used for this step.

Figure 5.3: Training loss vs. epochs on the Violent-Flows dataset.

Since the CDRNN architecture is large and complex, it suffers from a high chance of overfitting. We thus perform data augmentation for both two datasets to increase the diversity of the training sequences. Random rotation, shear, zoom, horizontal flip, width, height, and channel shifts are conducted on training instances. Given a crowd video sequence, the same augmentation is applied to all frames. The datasets Violent-Flows and CUHK Crowd are enlarged to 15 and 20 times of original sizes, respectively. All videos are resized to $224 \times 224$.

Data augmentation is performed to increase the diversity of the training sequences. Random rotation, shear, zoom, width and height shift, channel shift and horizontal flip are performed on training instances. Given a crowd video sequence, the same augmentation is applied to all frames. For the Violent-Flow dataset, training data are augmented to 5 times the original size. For the CUHK Crowd dataset, classes 1, 4, 5 and 7 have fewer examples and these classes are augmented to 15 times the original size while the remaining classes are augmented to 5 times the original size. All videos are resized to $299 \times 299$ pixels.
To train CDRNN, we first initialize the weighting parameters as mentioned above and then unfreeze the weights in Mixed Inception Block 9, fully-connected layer and stacked DLSTM layers. The network is trained on augmented crowd image sequences using the stochastic gradient descent method with mini-batch and a learning rate $1e^{-4}$. Fig. 5.3 shows an example of the training loss through the iteration epochs on the Violent-Flows dataset. The implementation is done with Keras [89].

**EXPERIMENTS ON THE NUS-HGA DATASET**

To evaluate our method on the NUS-HGA dataset, we follow the standard 5-fold cross-validation protocol in [1] and report the results in terms of mean accuracy in Table 5.2. It can be seen that our method achieves over 99% accuracy, outperforming the state-of-the-art by more than 3%. This is impressive because existing solutions need human supervision to acquire accurate human object trajectories from the videos. According to [1], they acquire human bounding boxes for the NUS-HGA dataset using existing tracking tools, which require manual annotation for bounding-box initializations. This limits such methods from being used in many real-world applications.

Table 5.2: Performance comparison with existing methods on the NUS-HGA dataset.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zhu et al. [33]</td>
<td>87.00</td>
</tr>
<tr>
<td>Cheng et al. [1] (MF)</td>
<td>93.20</td>
</tr>
<tr>
<td>Cho et al. [2]</td>
<td>96.03</td>
</tr>
<tr>
<td>Cheng et al. [1] (MAF)</td>
<td>96.20</td>
</tr>
<tr>
<td>ConvLSTM</td>
<td><strong>99.25</strong></td>
</tr>
</tbody>
</table>

Table 5.3 summarizes the performances of variants of CDRNN architectures on the NUS-HGA dataset. To validate the advantages of the end-to-end CDRNN model, we compare its performance with the solution, which uses pre-trained VGG-16 features as the input to the LSTMs. Results show that CDRNN achieves higher performance than the non-end-to-end structures containing
only LSTMs. Table 5.3 also shows that a larger number of stacked LSTMs achieves better performance. This can be explained as follows. For single-layer LSTM, the input is the output of VGG-16, which contains only the spatial information. For deeper layers of stacked LSTMs, the input is the output of previous LSTM, which contains both spatial and temporal representations. Such a high-level comprehensive representation contributes to stacked LSTMs’ better ability to understand complex group behaviors. We consider another variant in CDRNN architecture that preserves VGG-16 until the FC1/FC2 layer. The results from FC1 are mostly slightly better than those from the FC2. One of the possible explanations is, when including FC2 layer, the architecture tends to be overly-complex and suffers from overfitting.

Table 5.3: Performance comparison of variants of CDRNN architectures on the NUS-HGA dataset.

<table>
<thead>
<tr>
<th>Methods</th>
<th>FC1</th>
<th>FC2</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG16 feature + 1 stacked LSTM</td>
<td>94.45</td>
<td>94.03</td>
</tr>
<tr>
<td>VGG16 feature + 2 stacked LSTM</td>
<td>95.07</td>
<td>94.32</td>
</tr>
<tr>
<td>VGG16 feature + 3 stacked LSTM</td>
<td>95.15</td>
<td>95.04</td>
</tr>
<tr>
<td>CDRNN (1 stacked LSTM)</td>
<td>97.34</td>
<td>97.36</td>
</tr>
<tr>
<td>CDRNN (2 stacked LSTM)</td>
<td>99.17</td>
<td>98.83</td>
</tr>
<tr>
<td>CDRNN (3 stacked LSTM)</td>
<td>99.25</td>
<td>99.02</td>
</tr>
</tbody>
</table>

Figure 5.4 shows the confusion matrices of CDRNN using different pooling strategies. As CDRNN proceeds through the input frame sequentially, hidden state information learned from previous frames decays gradually. It is thus sub-optimal to only use the hidden state from the last time step. Max-pooling performs better than last-hidden-state because it is good at selecting salient signals and generates a more discriminative representation. However, max-pooling is subject to motion noise. For instance, it wrongly classifies StandTalk as Fight by a large error rate of 4.41%. StandTalk is a group state with less mobility, but there might be interactions, such as shaking hands or patting on shoulders, during the process. Such short and fast behavior could be detected as salient motions by max-pooling and lead to misclassification. Mean-pooling over the temporal sequences allows the network to better cope with noisy motion and performs better than either of the above two methods.
Figure 5.4: Confusion Matrices of CDRNN using different pooling strategies on the NUS-HGA dataset. CDRNN architecture with 3 stacked LSTM and FC1 is adopted.

**EXPERIMENTS ON THE BEHAVE DATASET**

Table 5.4: Different Classes in BEHAVE dataset.

<table>
<thead>
<tr>
<th>Class Index</th>
<th>Class Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Highly mixed pedestrian walking</td>
</tr>
<tr>
<td>2</td>
<td>Crowd walking following a mainstream and well organized</td>
</tr>
<tr>
<td>3</td>
<td>Crowd walking following a mainstream and poorly organized</td>
</tr>
<tr>
<td>4</td>
<td>Crowd merge</td>
</tr>
<tr>
<td>5</td>
<td>Crowd split</td>
</tr>
<tr>
<td>6</td>
<td>Crowd crossing in opposite directions</td>
</tr>
<tr>
<td>7</td>
<td>Intervened escalator traffic</td>
</tr>
<tr>
<td>8</td>
<td>Smooth escalator traffic</td>
</tr>
</tbody>
</table>

We also evaluate our model on the BEHAVE dataset and report the results in Table 5.5. CDRNN outperforms the state-of-the-art method by 1%. According to [2], they used the bounding boxes provided by the dataset, which were manually-labeled. Instead of using manually pre-labeled bounding boxes, CDRNN automatically identifies group activities without sacrificing performance, enabling a wide range of new applications. The performance of CDRNN again demonstrates its superior ability in understanding group behaviors by achieving “deep in space and time”.

Table 5.6 summarizes the performance of variants of the CDRNN architectures on the BEHAVE
dataset. Firstly, similar to the NUS-HGA dataset, CDRNN outperforms the non-end-to-end so-

olution. Secondly, deeper stacked LSTMs acquire higher classification accuracy. Thirdly, FC1

achieves generally better performance than FC2. Removing one fully-connected layer helps de-
crease the model complexity and reduce the chance of overfitting. The above results are consistent

with the results of the NUS-HGA dataset.

Table 5.5: Performance comparison with existing methods on the BEHAVE dataset.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Munch et al. [90]</td>
<td>66.25</td>
</tr>
<tr>
<td>Zhang et al. [32]</td>
<td>81.50</td>
</tr>
<tr>
<td>Cheng et al. [1]</td>
<td>92.93</td>
</tr>
<tr>
<td>Cho et al. [2]</td>
<td>93.74</td>
</tr>
<tr>
<td>CDRNN</td>
<td>94.63</td>
</tr>
</tbody>
</table>

Table 5.6: Performance comparison of variants of CDRNN architectures on the BEHAVE Crowd dataset.

<table>
<thead>
<tr>
<th>Methods</th>
<th>FC1</th>
<th>FC2</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG16 feature + 1 stacked LSTM</td>
<td>91.35</td>
<td>91.06</td>
</tr>
<tr>
<td>VGG16 feature + 2 stacked LSTM</td>
<td>91.68</td>
<td>92.24</td>
</tr>
<tr>
<td>VGG16 feature + 3 stacked LSTM</td>
<td>92.53</td>
<td>92.14</td>
</tr>
<tr>
<td>CDRNN (1 stacked LSTM)</td>
<td>92.66</td>
<td>92.07</td>
</tr>
<tr>
<td>CDRNN (2 stacked LSTM)</td>
<td>94.45</td>
<td>94.13</td>
</tr>
<tr>
<td>CDRNN (3 stacked LSTM)</td>
<td>94.63</td>
<td>94.18</td>
</tr>
</tbody>
</table>

Figure 5.4 shows the confusion matrices of CDRNN using different pooling methods. The results
are also consistent with that of the NUS-HGA dataset. The mean-pooling method performs best
among the three methods. Although max-pooling detects salient motion information and performs
slightly better than last-hidden-state, it suffers from noisy data. It misclassified InGroup behavior
as Fight with a large error rate of 6.58%. One possible explanation that is interactions such as hugs
and gestures during the InGroup state could be detected as salient motion and lead to misclassifi-
cation for max-pooling.
EXPERIMENTS ON THE VIOLENT-FLOWS DATASET

To evaluate our method on the Violent-Flows dataset, we follow the standard 5-fold cross-validation protocol in [21] and report the results in terms of mean accuracy in Table 5.7. It can be seen that our method achieves over 93% accuracy, outperforming state-of-the-art by more than 8%. Note that since Violent-Flows was released in 2012, several methods have been proposed to address the crowd understanding problem on this dataset, but only 4% performance improvement has been made over these years. Such an increase in performance achieved from our model demonstrates the effectiveness of CDRNN.

We also compare in Table 5.7 the performances of CDRNN and CDRNN. In CDRNN, we use traditional LSTMs instead of DLSTMs. Since the Derivative of States in DLSTM can explicitly model the information gain between successive frames, it is reasonable that CDRNN outperforms ConvLSTM. ConvLSTM indicates that CNN and traditional LSTM are connected.

Table 5.8 summarizes the performances of variants of CDRNN architectures on the Violent-Flows
dataset. To validate the advantage of the end-to-end CDRNN model, we compare its performance with the solution using pre-trained CNN features as the input to DLSTMs. Results show that CDRNN achieves higher performance than the non-end-to-end deep structures containing only DLSTMs. The end-to-end solution employs a unified scheme to optimize the parameters of CNN and RNN hand in hand, thus generates a more harmonious model.

Table 5.7: Performance comparison with existing methods on the Violent-Flows dataset. We use GoogleNet Inception V3 as the convolutional part, three stacked LSTM/DLSTMs, and mean-pooling for the video-level representation.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Violent Flows [21]</td>
<td>81.30</td>
</tr>
<tr>
<td>Common Measure [74]</td>
<td>81.50</td>
</tr>
<tr>
<td>Hist of Tracklet [50]</td>
<td>82.30</td>
</tr>
<tr>
<td>Substantial Derivative [75]</td>
<td>85.43</td>
</tr>
<tr>
<td>Holistic Features [52]</td>
<td>85.53</td>
</tr>
<tr>
<td>ConvLSTM</td>
<td>91.34</td>
</tr>
<tr>
<td>CDRNN</td>
<td><strong>93.59</strong></td>
</tr>
</tbody>
</table>

Table 5.8 also shows that larger number of stacked DLSTM achieves better performance. This can be explained as follows. For single-layer DLSTM, the input is the output of corresponding convolutional neural networks. Such input in certain time-step contains only spatial information. For deeper layers of stacked DLSTM, the input is the output of previous DLSTM, which in each time-step contains both spatial and temporal representations. Such high-level comprehensive representation contributes to stacked DLSTMs’ better ability to understand complex crowd behaviors. We notice that the margin between two and three stacked DLSTMs is very small. The reason is with the increase of DLSTM layers, CDRNN has a higher chance of overfitting. The choice of two or three stacked DLSTMs depends on the trade-off of the selection of higher accuracy or faster detection.

We consider another variant in CDRNN architecture that uses VGG-16 [91] instead of Inception V3 as the convolutional part, shown in Fig 5.6. We can see that CDRNN with Inception V3
architecture constantly achieves higher performance than its variant using VGG architecture. Since
the Inception blocks have a lower number of parameters, training and testing for CDRNN using
Inception V3 are also faster.

Figure 5.6: An illustration of a variant of CDRNN using VGG-16 as the convolutional part. The
model consists of convolutional (red), pooling (green), fully-connected (blue), DLSTM (yellow)
and softmax (brown) layers. Best viewed in color.

Table 5.8: Performance comparison of variants of CDRNN architectures on the Violent-Flows
dataset. IV3 indicates Inception V3.

<table>
<thead>
<tr>
<th>Methods</th>
<th>IV3</th>
<th>VGG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-End-to-End CNN feature + 1 stacked DLSTM</td>
<td>86.34</td>
<td>84.75</td>
</tr>
<tr>
<td>Non-End-to-End CNN feature + 2 stacked DLSTM</td>
<td>86.98</td>
<td>84.15</td>
</tr>
<tr>
<td>Non-End-to-End CNN feature + 3 stacked DLSTM</td>
<td>87.03</td>
<td>85.18</td>
</tr>
<tr>
<td>End-to-End CDRNN (1 stacked DLSTM)</td>
<td>91.85</td>
<td>89.86</td>
</tr>
<tr>
<td>End-to-End CDRNN (2 stacked DLSTM)</td>
<td>93.50</td>
<td>91.09</td>
</tr>
<tr>
<td>End-to-End CDRNN (3 stacked DLSTM)</td>
<td>93.59</td>
<td>91.13</td>
</tr>
</tbody>
</table>

**EXPERIMENTS ON THE CUHK CROWD DATASET**

We also evaluate our model on the CUHK Crowd dataset and report the results in Table 5.9.
CDRNN outperforms Collective Transition [22] and Coherent LSTM [49] for 10% and 4%, re-
spectively. These two methods take human trajectories as input, which were provided by the
dataset after manual correction. Considering that these two methods use “ground truth” human
trajectories as input, CDRNN’s performance is impressive as it does not require trajectory clues. Comparing the performances of Coherent LSTM and CDRNN, we can see that CDRNN has a better ability in understanding crowd dynamics by achieving “deep in space and time”.

Table 5.9 also shows that CDRNN achieves slightly lower performance than CDRNN. The conventional LSTMs do not consider the impact of spatio-temporal dynamics corresponding to the give salient motion patterns, when they gate the information that ought to be memorized through time. The weakness is addressed by the Derivative of States in DLSTMs.

Table 5.9: Performance comparison with existing methods on the CUHK Crowd dataset. We use GoogleNet Inception V3 as the convolutional part, three stacked LSTM/DLSTMs, and mean-pooling for the video-level representation.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collective Transition [22]</td>
<td>70.00</td>
</tr>
<tr>
<td>Un-coherent LSTM [49]</td>
<td>73.82</td>
</tr>
<tr>
<td>Coherent LSTM [49]</td>
<td>76.50</td>
</tr>
<tr>
<td>ConvLSTM</td>
<td>78.65</td>
</tr>
<tr>
<td>ConvDLSTM</td>
<td><strong>80.33</strong></td>
</tr>
</tbody>
</table>

Table 5.10 summarizes the performance of variants of CDRNN architectures on the CUHK Crowd dataset. Firstly, similar to the Violent-Flows dataset, CDRNN outperforms the non-end-to-end solution. Non-end-to-end solutions separate the steps of feature extraction and parameter learning. CDRNN unifies these two processes and thus generates better performances. Secondly, deeper stacked LSTMs acquire higher classification accuracy. Deeper layers of stacked DLSTM take the output of the previous DLSTM layer, which contains spatio-temporal information in each step. Such information is more comprehensive in understanding complex crowd dynamics. Thirdly, CDRNN using VGG-16 achieves lower performance than the one using GoogleNet Inception V3. VGG-16 architecture. The above results are consistent with the results on the Violent-Flows dataset.
Table 5.10: Performance comparison of variants of CDRNN architectures on the CUHK Crowd dataset. IV3 indicates Inception V3.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Methods</th>
<th>IV3</th>
<th>VGG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-End-to-End</td>
<td>CNN feature + 1 stacked DLSTM</td>
<td>73.35</td>
<td>71.06</td>
</tr>
<tr>
<td></td>
<td>CNN feature + 2 stacked DLSTM</td>
<td>73.68</td>
<td>72.94</td>
</tr>
<tr>
<td></td>
<td>CNN feature + 3 stacked DLSTM</td>
<td>75.83</td>
<td>73.43</td>
</tr>
<tr>
<td>End-to-End</td>
<td>CDRNN (1 stacked DLSTM)</td>
<td>78.32</td>
<td>76.15</td>
</tr>
<tr>
<td></td>
<td>CDRNN (2 stacked DLSTM)</td>
<td>79.68</td>
<td>77.48</td>
</tr>
<tr>
<td></td>
<td>CDRNN (3 stacked DLSTM)</td>
<td>80.33</td>
<td>78.03</td>
</tr>
</tbody>
</table>

**EVALUATION OF FULLY AUTOMATIC RECOGNITION**

To demonstrate CDRNN’s effectiveness in automatically understanding group behaviors, we implement state-of-the-art methods, Cheng et al. [1] and Cho et al. [2]. These two methods, using no human supervision, are compared with CDRNN on both datasets. We use gKLT tracker [92] to generate the human trajectories, which then serve as inputs to Cheng et al. [1] and Cho et al. [2]. During trajectory generation, we perform no manual initialization or correction to ensure fully automatic group activity recognition.

As shown in Table 5.11, the performances of Cheng et al. [1] and Cho et al. [2] drop dramatically without human supervision. With no initializations for human bounding boxes on the NUS-HGA dataset, the performances of Cheng et al. [1] and Cho et al. [2] drop 7%. For the BEHAVE dataset, without manually labeled human bounding boxes, state-of-the-art methods again perform unsatisfactorily, underperforming by more than 9%. The above results show that the detection and tracking accuracy dictates the performance of existing methods. Without human supervision, existing methods perform undesirably and are not suitable for automatic group behavior understanding. CDRNN, however, requires no trajectory detection and takes scene semantics into consideration, thus possessing a better potential for automatically analyzing group activities.
Table 5.11: Evaluation of fully automatic recognition.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>NUS-HGA</th>
<th>BEHAVE</th>
</tr>
</thead>
<tbody>
<tr>
<td>With Human Supervision</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Cheng et al. [1]</td>
<td>96.20</td>
<td>89.51</td>
</tr>
<tr>
<td>Cho et al. [2]</td>
<td>96.03</td>
<td>88.86</td>
</tr>
<tr>
<td>Our method</td>
<td>–</td>
<td>99.25</td>
</tr>
</tbody>
</table>

EVALUATION OF MODEL GENERALIZATION CAPABILITY

To evaluate CDRNN’s capability of understanding various crowd scene scenarios, we carefully implement the Holistic Features [52] and Coherent LSTM [49] methods and compare them with CDRNN on both the above two datasets. For Coherent LSTM, we use the same gKLT tracker [92] as the CUHK Crowd dataset to generate the human trajectories for the Violent-Flows dataset.

As shown in Table 5.12, our CDRNN outperforms the two methods on both two datasets. Holistic Features [52] performs uncomparably on the CUHK Crowd dataset because it uses simple hand-crafted features of only four dimensions, which are not optimized for different crowd scenarios. Although Coherent LSTM [49] works decently on the CUHK dataset, it performs unsatisfactorily on the Violent-Flows dataset. In the Violent-Flows dataset, crowd density is much higher. This could result in generating inaccurate human trajectories, which then serve as inputs for Coherent LSTM. In addition, there exists less crowd motion flow in Violent-Flows. As a trajectory-based method, Coherent LSTM makes no use of scene information and tends to be less effective. CDRNN, however, requires no trajectory detection and takes into consideration scene semantics, thus possesses better generalization capability for crowd scene understanding.

Table 5.12: Evaluation of model generalization capability.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Violent-Flows</th>
<th>CUHK Crowd</th>
</tr>
</thead>
<tbody>
<tr>
<td>Holistic Features [52]</td>
<td>85.53</td>
<td>70.75</td>
</tr>
<tr>
<td>Coherent LSTM [49]</td>
<td>84.23</td>
<td>76.65</td>
</tr>
<tr>
<td>Our method</td>
<td>93.59</td>
<td>80.33</td>
</tr>
</tbody>
</table>
SUMMARY

In this chapter, we propose an end-to-end deep architecture Convolutional Differential Recurrent Neural Networks (CDRNN) for group behavior and crowd scene understanding. Our model consists of convolutional neural networks and stacked long short-term memory recurrent neural networks. CDRNN directly takes the raw image sequences as the input and does not require additional handcrafted flow-based or trajectory-based feature representation and works with crowds of high density and low mobility. Performance studies on two public crowd datasets have shown that the proposed technique significantly outperforms state-of-the-art methods.
CHAPTER 6: DEEP DIFFERENTIAL RECURRENT NEURAL NETWORKS

Parts of this chapter have been presented in ACM TOMM [93] (Copyright ©2019 ACM).

Due to the special gating schemes of Long Short-Term Memory (LSTM), LSTMs have shown a greater potential to process complex sequential information than the traditional Recurrent Neural Network (RNN). The conventional LSTM, however, fails to take into consideration the impact of salient spatio-temporal dynamics present in the sequential input data. This problem was first addressed by the differential Recurrent Neural Network (dRNN), which uses a differential gating scheme known as Derivative of States (DoS). DoS uses higher orders of internal state derivatives to analyze the change in information gain originated from the salient motions between the successive frames. The weighted combination of several orders of DoS is then used to modulate the gates in dRNN. While each individual order of DoS is good at modeling a certain level of salient spatio-temporal sequences, the sum of all the orders of DoS could distort the detected motion patterns. To address this problem, we propose to control the LSTM gates via individual orders of DoS. To fully utilize the different orders of DoS, we further propose to stack multiple levels of LSTM cells in increasing order of state derivatives. The proposed model progressively builds up the ability of the LSTM gates to detect salient dynamical patterns in deeper stacked layers modeling higher orders of DoS, and thus the proposed LSTM model is termed deep differential Recurrent Neural Network ($d^2$RNN). The effectiveness of the proposed model is demonstrated on three publicly available human activity datasets: NUS-HGA, Violent-Flows, and UCF101. The proposed model outperforms both LSTM and non-LSTM based state-of-the-art algorithms.
OVERVIEW

Recent years have witnessed a revival of Long Short-Term Memory (LSTM) [4], thanks to the special gating mechanism that controls access to memory cells. The superior capability of LSTM has been shown in a wide range of problems such as machine translation [8, 9], speech recognition [24], and multi-modal translation [10]. Compared with many existing spatio-temporal features [12, 13] from the time-series data, LSTM uses either a hidden layer [16] or a memory cell [4] to learn the time-evolving states which model the underlying dynamics of the input sequences. 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.

LSTMs have shown tremendous potential in activity recognition tasks [6, 17, 7]. The existing LSTM model represents a video by integrating all the available information from each frame over time. It was pointed out in [18] that for an activity recognition task, not all frames contain salient spatio-temporal information which is equally discriminative to different classes of activities. Many frames contain non-salient motions which are irrelevant to the performed actions. When the gate units in LSTM modulate the input and output of the memory cells, they do not explicitly consider whether a frame contains salient motion information. Because of this, LSTM is insensitive to the dynamical evolution of the hidden states given the input video sequences. In other words, it cannot capture the salient dynamic patterns embedded in the actions.

dRNN addresses this problem and models the dynamics of actions by computing different orders of Derivative of State (DoS). DoS models the change in information gain caused by the salient motions between the successive frames using higher orders of internal state derivatives. Intuitively,
1st-order DoS represents the velocity of change of internal state memory while 2nd-order DoS represents the acceleration of memory state change. This reveals that the conventional LSTM, whose internal cell is simply 0th-order DoS, only captures the locality of information change.

Figure 6.1: The energy curves of the 0th-, 1st-, and 2nd-orders of DoS over an example of sequence for the activity "RunInGroup".

Despite the above-mentioned advantages, dRNN is formulated in the fashion that the gates are modulated by the weighted combinations of several orders of DoS. While an individual order of DoS is able to model a certain degree of dynamical structures, the sum of all the orders of DoS could distort the detected salient motion patterns. To support the above observation, Fig. 6.1 illustrates the energy curves of the 0th-, 1st-, and 2nd-orders of DoS over an example of a sequence for the activity "RunInGroup". The local maxima indicate high energy landmarks corresponding to the salient motion frames at different levels. Indeed, each order of DoS enables the LSTM unit to model the dynamics of local saliency at a certain level, e.g. 0th-, 1st-, and 2nd-orders of DoS captures locality, velocity, and acceleration information change, respectively. The weighted sum of different orders of DoS, however, may risk misaligning salient motion and result in distorted motion patterns. To further confirm the above claim, a preliminary experimental study was conducted and the results were shown in Section 4. This study demonstrates that a simple ensemble model of individual orders of DoS outperforms the conventional same-order weighted-sum dRNN,
thus reveals the suboptimality of combining different orders DoS within LSTM cell. The above analysis inspires us to question the internal structure of conventional dRNN, and reconsider to use the individual orders of state derivatives to control the LSTM gates.

As is generally accepted, RNNs are inherently deep in time because the current hidden state is a function of all previous hidden states. By questioning whether RNNs could also benefit from depth in space, just as feed-forward layers which are stacked in conventional deep networks, Graves et al. [24] introduced Deep Recurrent Neural Networks, also known as stacked LSTMs. Stacked LSTMs have shown superiority over the traditional LSTM in modeling complex sequences and have been used in various types of applications. Inspired by Deep Recurrent Neural Network, we are motivated to explore whether the stacked deep layers in space could naturally reveal the saliency of motion dynamics over time, thus avoiding the misaligned DoS in different orders.

To this end, we propose to stack multiple levels of LSTM cells with individual and increasing orders of DoS. The proposed model has the following advantages. With the individual order of DoS, each layer of LSTM cell captures a certain level of salient spatio-temporal information. With stacked architecture, our model progressively builds up the ability of LSTM gates to detect salient dynamic patterns with deeper memory layers modeling higher orders of DoS. The proposed model is thus termed deep differential Recurrent Neural Network ($d^2$RNN). The $d^2$RNN differs from conventional stacked LSTMs in that stacked LSTMs use homogeneous LSTM layers while $d^2$RNN uses heterogeneous ones. In this way, $d^2$RNN is not only capable of modeling more complex dynamical patterns, but also enables a hierarchy of DoS saliency in deep layers to model the spatio-temporal dynamics over time.

We demonstrate the performance of $d^2$RNN on three publicly available human activity datasets: NUS-HGA [23], Violent-Flows [21], and UCF101 [25]. Specifically, $d^2$RNN outperforms the existing LSTM, dRNN, and stacked LSTM models, consistently achieving better performance in
detecting human activities in sequences. In addition, we compared with the other non-LSTM algorithms, where \( d^2 \text{RNN} \) model also reached competitive performance.

![Figure 6.2: Architectures of dRNN cell (left) and nth-order (layer \( n + 1 \)) of \( d^2 \text{RNN} \) cell (right) at time \( t \). Best viewed in color.](image)

**THE PROPOSED MODEL**

Given an activity recognition task, not all video frames contain salient patterns to discriminate between different classes of activities. dRNN tries to detect and integrate the salient spatio-temporal sequences via the state derivative. As the internal state contains the memory of the previous input sequences, the state derivative explicitly models the change in information gain and considers the impact of dynamic structures. Thus, the state derivative tends to be effective in recognizing actions.

As mentioned in [94], LSTM gates serve as the most crucial elements of LSTM. dRNN formulates the input, forget, and output gates using the combination of different orders of DoS. To be more specific, 0th-order DoS, which is the same as conventional LSTM internal cell, models the locality of memory change; 1st-order DoS denotes the velocity of change in information gain; and 2nd-order DoS describes the acceleration of memory change, etc. While each individual order of DoS is effective in capturing a certain level of salient spatio-temporal sequences, the sum of all the orders
of DoS could distort the detected salient motion patterns and result in less effective modulation of those gates.

In this paper, we propose to modulate the LSTM gates via individual orders of DoS. Inspired by [24], we stack multiple levels of LSTM cells with increasing orders of DoS. To be more specific, layer 1 of $d^2$RNN uses 0th-order DoS, which resembles the conventional LSTM cell; layer 2 uses LSTM cell with 1st-order DoS; and layer 3 uses LSTM cell with 2nd-order DoS, etc. Since we are integrating the ideas of DoS from dRNN and deep stacked layers from deep RNN, our proposed model is termed deep differential Recurrent Neural Network ($d^2$RNN). Within each layer of $d^2$RNN, our model learns the change in information gain with individual order of DoS. With deeper layers of $d^2$RNN cell, our model learns higher-degree and more complex dynamical patterns.

Fig. 6.2 illustrates the LSTM unit in layer $(n + 1)$ of the proposed $d^2$RNN model. Hollow lines indicate the information flow of $s_{t-1}$. Formally, we have the following recurrent equations to control the LSTM gates in layer $(n + 1)$ of $d^2$RNN.

(i) Input gate:

$$i_t = \sigma(W_{id}^{(n)} \frac{d^{(n)} s_{t-1}}{dt^{(n)}} + W_{ih} h_{t-1} + W_{ix} x_t + b_i),$$  \hspace{1cm} (6.1)$$

(ii) Forget gate:

$$f_t = \sigma(W_{fd}^{(n)} \frac{d^{(n)} s_{t-1}}{dt^{(n)}} + W_{fh} h_{t-1} + W_{fx} x_t + b_f),$$  \hspace{1cm} (6.2)$$

(iii) Output gate:

$$o_t = \sigma(W_{od}^{(n)} \frac{d^{(n)} s_{t}}{dt^{(n)}} + W_{oh} h_{t} + W_{ox} x_t + b_o),$$  \hspace{1cm} (6.3)$$
DISCRETIZATION

In this section, we aim to discretize DoS since \( d^2 \text{RNN} \) is defined in the discrete-time domain. The 1st-order derivative \( \frac{ds_t}{dt} \), which is the velocity of information change, can be discretized as the difference of states:

\[
v_t \triangleq \frac{ds_t}{dt} = s_t - s_{t-1},
\]

for simplicity. More details can be found in [58].

Similarly, we consider the 2nd-order of DoS as the acceleration of information change. It can be discretized as:

\[
a_t \triangleq \frac{d^2 s_t}{dt^2} = v_t - v_{t-1} = s_t - 2s_{t-1} + s_{t-2}.
\]

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

Figure 6.3: An illustration of our framework for video-level activity recognition. Best viewed in color.
With the above recurrent equations, the $d^2$RNN model proceeds with the following procedures starting in layer 1 ($n = 0$) at time step $t$:

- Compute input gate activation $i_t$ and forget gate activation $f_t$ by Eq. (6.1) and Eq. (6.2);
- Update state $s_t$ with $i_t$ and $f_t$ by Eq. (5.4);
- Compute discretized DoS $\frac{d^{(n)}s_t}{dt^{(n)}}$;
- Compute output gate $o_t$ by Eq. (6.3);
- Output $h_t$ gated by $o_t$ from memory cell by Eq. (5.6);
- If there exists a deeper layer in $d^2$RNN, set $n = n + 1$ and $x_t = h_t$ for the following layer and repeat the above steps;

For a frame-by-frame prediction task, we output the label $p_t$ by applying the softmax to $h_t$ using Eq. (3.2) and (5.7). To learn the model parameters of $d^2$RNN, we define a loss function to measure the deviation between the target class $c_t$ and $p_t$ at time $t$:

$$\ell(p_t, c_t) = -\log p_t, c_t.$$ 

Then, we can minimize the cumulative loss over the sequence:

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

For an activity recognition task, the label of activity 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 recognizing the activity. The sequence level class probability $p$ is generated by computing the output of $d^2$RNN with Eq. (3.2) and applying the softmax function with Eq. (5.7). For a given training label $c$, the $d^2$RNN can be trained by minimizing the loss function below, i.e.

$$\ell(p, c) = -\log p_c.$$  

The loss function can be minimized by Back Propagation Through Time (BPTT) [88], which unfolds an LSTM model over several time steps and then runs the back-propagation algorithm to train the model. To prevent back-propagated errors from decaying or exploding exponentially, we use truncated BPTT according to Hochreiter et al. [4] to learn the model parameters. Specifically, in our model, errors are not allowed to re-enter the memory cell once they leave it through the DoS nodes.

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

$$\frac{\partial i_t}{\partial v_{t-1}} \doteq 0, \quad \frac{\partial f_t}{\partial v_{t-1}} \doteq 0, \quad \frac{\partial o_t}{\partial v_{t}} \doteq 0,$$

and

$$\frac{\partial i_t}{\partial a_{t-1}} \doteq 0, \quad \frac{\partial f_t}{\partial a_{t-1}} \doteq 0, \quad \frac{\partial o_t}{\partial a_{t}} \doteq 0,$$

where $\doteq$ stands for the truncated derivatives.

**COMPLEXITY ANALYSIS**

In the LSTM models, there are an input layer, a recurrent LSTM layer, and an output layer. We denote the number of input units by $m_i$, the number of memory cells by $m_s$, and the number of
output units by $m_o$. Here we use $M$ to denote the total number of parameters in LSTM network. The computational complexity of learning LSTM models per weight and per time step with the stochastic gradient descent (SGD) optimization technique is $O(1)$ [95]. Therefore, the learning computational complexity per time step is $O(M)$. In the following analysis, we ignore the computation for biases. In the meantime, we do not take into account the sigmoid activations, hyperbolic tangent functions or element-wise products, since they are all trivial factors.

We now start with the complexity analysis of the standard LSTM. As discussed in previous sections, the conventional LSTM is a special form of dRNN or $d^2$RNN, in which the order $N = 0$. For input, forget, and output gate computations, we can calculate the number of weighting parameters according to Eq. (6.1)(6.2)(6.3), where each includes $m_s \times m_s + m_s \times m_s + m_s \times m_i$ weights, ignoring the biases. Here note that the peephole connection introduces additional parameters of $m_s \times m_s$. To update the internal state, we use Eq. (5.4) where the number of weights equals to $m_s \times m_s + m_s \times m_i$. The calculation of the hidden state in Eq. (5.6) involves only element-wise product and hyperbolic tangent functions, which will be ignored. Finally, Eq. (3.2) introduces $m_o \times m_s$ weighting parameters. To sum up, the total number of parameters for the conventional LSTM is,

$$M_{LSTM} = m_s \times m_s \times 7 + m_s \times m_i \times 4 + m_s \times m_o.$$ 

For the original dRNN, the difference from the conventional LSTM lies only in the calculations of input, forget, and output gates. Within each gate computation, differential orders $N$ adds on $m_s \times m_s \times N$ weighting parameters. Thus, the total number of parameters for $N$-th order dRNN is,

$$M_{dRNN}^N = m_s \times m_s \times (7 + 3 \times N) + m_s \times m_i \times 4 + m_s \times m_o.$$ 

Now we analyze the computational complexity for the $d^2$RNN model. When $N = 0$, $d^2$RNN shares
the same number of parameters as LSTM and 0th-order dRNN,

\[ M_{d^2RNN}^0 = m_s \times m_s \times 7 + m_s \times m_i \times 4 + m_s \times m_o. \]

When \( N > 0 \), we compute the number of parameters as follows,

\[ M_{d^2RNN}^N = M_{d^2RNN}^{N-1} + \Delta M \]

Since \( x_t = h_t \) for deeper layers, we have \( m_i = m_s \),

\[ \Delta M = m_s \times m_s \times 11 + m_s \times m_o. \]

Finally, we have the total number of weighting parameters for \( N \)th-order \( d^2RNN \),

\[ M_{d^2RNN}^N = m_s \times m_s \times (7 + 11 \times N) + m_s \times m_i \times 4 + m_s \times m_o \times (1 + N). \]

From the above analysis, the learning time for an LSTM network with a relatively small dimension of inputs is dominated by the \( m_s \times m_s \) factor. For the tasks requiring a large number of memory cells to store temporal contextual information, learning LSTM models tends to be computationally expensive.

**EXPERIMENTAL RESULTS**

In this section, we first talk about experimental setup, including datasets, feature extraction, and network architecture. We then compare the performance of the proposed method with state-of-the-art LSTM and non-LSTM methods present in the existing literature on human activity datasets.
The proposed method is evaluated on three publicly available human activity datasets: NUS-HGA [23], Violent-Flow [21], and UCF101 [25].

We choose the NUS-HGA dataset as it is a well-collected benchmark dataset for evaluating activity recognition techniques. The NUS-HGA dataset includes 476 video clips covering six group activity classes: Fight, Gather, Ignore, RunInGroup, StandTalk and, WalkInGroup. Each instance involves 4-8 persons. The sequences are captured over different backgrounds with a static camera recording 25 frames per second. Each video clip has a resolution of $720 \times 576$ and lasts around 10 seconds.

The Violent-Flows (VF) dataset is a real-world video footage of crowd violence, along with standard benchmark protocols designed for violent/non-violent classification. The Violent-Flows dataset includes 246 real-world videos downloaded from YouTube. The shortest clip duration is 1.04 seconds, the longest slip is 6.52 seconds, and the average length is 3.6 seconds.

UCF101 is an action recognition data set of realistic action videos, collected from YouTube, having 101 action categories. This data set is an extension of the UCF50 data set which has 50 action categories. With 13320 videos from 101 action categories, UCF101 gives the largest diversity in terms of actions and with the presence of large variations in camera motion, object appearance and pose, object scale, viewpoint, cluttered background, illumination conditions, etc, it is the most challenging data set to date. As most of the available action recognition data sets are not realistic and are staged by actors, UCF101 aims to encourage further research into action recognition by learning and exploring new realistic action categories. The videos in 101 action categories are grouped into 25 groups, where each group can consist of 4-7 videos of an action. The videos from the same group may share some common features, such as similar backgrounds, similar
viewpoints, etc.

In the following experiments, we are mainly using densely sampled HOG3D features [12] to represent each frame of video sequences for the NUS-HGA and Violent-Flows datasets; while deep CNN features such as ResNet and SENet are mainly used for the UCF101 dataset. Details for extracting the above deep features can be found on Keras [89]. For HOG3D features, 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 kept the same as the one used in [12]. The standard dense sampling parameters for extracting HOG3D features can be found on the author’s webpage. All the videos are resized into 160 × 120. The size of the descriptor is 300 per cell of grid, and there are 58 such cells in each frame, yielding a 17,400-dimensional feature vector per frame. To construct a compact input into $d^2$RNN model, Principal Component Analysis (PCA) is then applied to reduce the feature dimension. After PCA dimension reduction, NUS-HGA has a feature dimension of 300, and Violent-Flows has a feature dimension of 500. Both of them are retaining 90% of energy among the principal components. For the sake of fair comparison, we use the same features as input for other LSTM models too.

**ARCHITECTURE AND TRAINING**

Table 6.1: Architectures of the $d^2$RNN model used on the NUS-HGA and Violent-Flows datasets.

<table>
<thead>
<tr>
<th></th>
<th>NUS-HGA</th>
<th>Violent-Flows</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Units</td>
<td>300</td>
<td>500</td>
</tr>
<tr>
<td>State Units</td>
<td>200</td>
<td>300</td>
</tr>
<tr>
<td>Output Units</td>
<td>6</td>
<td>2</td>
</tr>
</tbody>
</table>

The network architectures of the $d^2$RNN models trained on the above datasets are shown in Table 6.1. We keep the state units the same for all the LSTM layers in $d^2$RNN. For the sake of fair comparison, we adopt the same architecture for stacked LSTMs models. For dRNN model, we
keep the same number of state units as stacked LSTMs and $d^2$RNN. We can see that the number of memory cell units is smaller than the input units on both datasets. This can be interpreted as follows. The sequence of a human activity video often forms a continuous pattern embedded in a low-dimensional manifold of the input space. Thus, a lower-dimension state space is sufficient to capture the dynamics of such patterns. The number of output units corresponds to the number of classes in the datasets.

We plot the learning curves for training the $d^2$RNN models on the NUS-HGA dataset in Fig. 6.4. $(n+1)$-layer $d^2$RNN refers to the model of $(n+1)$ layers with DoS starting from 0th-order to nth-order. The learning rate of the BPTT algorithm is set to 0.0001. The figure shows that the objective loss continuously decreases over 50 epochs. Deep layers of $d^2$RNN converge faster due to larger model complexity.

RESULTS ON THE NUS-HGA DATASET

There are several different evaluation protocols used on the NUS-HGA dataset in the literature, which can lead to fairly large differences in performance across different experimental protocols.

To evaluate the performances of the proposed $d^2$RNN vs. other LSTM models, we perform five-fold cross-validation. This experiment set-up increases the challenge for the task compared to the one using Monte-Carlo cross-validation. For the NUS-HGA dataset, the activity videos are produced by chopping longer sequences into shorter ones. Due to random sampling, Monte-Carlo cross validation makes the task easier by putting similar video instances to both training and testing sets. Five-fold cross-validation, however, dramatically increases the difficulty since training and testing examples usually have high in-class variations regarding background, view-angle, lighting, and activity participants.
In order to better understand the disadvantage of the combination of different orders of DoS in dRNN model, we perform the following experimental study. First, we construct a variant of LSTM using individual orders of DoS and then treat them as separate models. We call these models "1st-order LSTM" and "2nd-order LSTM". Then we use the AdaBoost algorithm [96] to ensemble the above models, which we call Ensemble RNN (eRNN). To be more specific, 1st-order eRNN ensembles 0th- and 1st-order LSTMs; 2nd-order eRNN ensembles 0th-, 1st-, and 2nd-order LSTMs. By doing so, we also intend to study whether each individual order of DoS is good at modeling a certain level of motion saliency. All the above models use the same HOG3D feature presented above.
Figure 6.5: Frame-by-frame prediction of activity category over time. Best viewed in color.

Figure 6.6: Evaluation of individual vs. combination of DoS.

In Fig. 6.6, the leftmost three bars are performances of conventional LSTM, 1st-order LSTM, and 2nd-order LSTM. LSTMs with higher orders of individual DoS gain slightly better performances. On the other hand, their ensemble models, which are eRNNs, achieve substantially better results. This shows that each individual order of DoS indeed can detect a certain level of motion saliency and contributes to their ensemble model. While comparing the same order of dRNN with eRNN,
we find out that eRNN consistently achieves higher results. This demonstrates that it is suboptimal to combine different orders of DoS within the LSTM gates and the sum of all the orders of DoS would distort the detected motion patterns.

Figure 6.7: Comparison of the proposed $d^2$RNN model with other LSTM models on the NUS-HGA dataset.

Secondly, we show the performance comparison of our proposed model vs. dRNN and stacked LSTMs in Fig. 6.7. For the dRNN and $d^2$RNN models, we report the accuracy up to the 2nd-order of DoS. For stacked LSTMs, we report the accuracy up to 3 layers, which is the same highest layer as $d^2$RNN. All the LSTM models use the same HOG3D feature presented above. The conventional LSTM yields the lowest performance compared to other LSTM models. This is because conventional LSTM uses neither DoS nor deep layers to capture motion saliency presented in given video frames.
Generally, higher orders of DoS generate better performance for both dRNN and $d^2$RNN. However, the performance increase for dRNN is smaller than $d^2$RNN. This can be explained that for dRNN, the weighted combination of all the orders of DoS distorts the detected motion patterns. $d^2$RNN, on the other hand, uses individual orders of DoS on each layer, and can preserve and align the salient dynamic structures.

It can also be seen that $d^2$RNN outperforms stacked LSTMs given the same number of deep layers, which demonstrates the advantage of heterogeneous LSTM layers used in $d^2$RNN over the homogeneous ones in stacked LSTMs. To be more specific, the higher orders of DoS from $d^2$RNN detects salient spatio-temporal structures which cannot be captured by conventional LSTM layers used in stacked LSTMs. To this end, the above analysis shows that $d^2$RNN not only is capable of modeling more complex dynamical patterns, but also enables a hierarchy of DoS saliency in deep layers to model the spatio-temporal dynamics over time.

Although deeper layers or higher orders of $d^2$RNN might improve the accuracy further, we do not report the result since it becomes trivial by simply adding more deep layers with higher-order DoS. The improved performance, however, might not compensate for the increased computational cost. Moreover, with an increased number of deep layers modeling higher orders of DoS, a larger number of model parameters would have to be learned with the limited training examples. This tends to cause overfitting problems, making the performance stop improving or even begin to degenerate. Therefore, for most practical applications, the 3-layer setup for $d^2$RNN should be sufficient. More applications of deep architectures of RNNs can be found in [97, 8].

In order to compare $d^2$RNN with other non-LSTM state-of-the-art methods, we follow [1] and evaluate our method via Monte-Carlo Cross-Validation. To be more specific, we randomly select 80% of the examples from each class of the dataset to form the training sets, and then assign the rest to the test set. The performance is reported by the average accuracy across five such trials.
Table 6.2: Performance comparison of $d^2$RNN vs. traditional methods on the NUS-HGA dataset. MF indicates motion feature fusion and MAF indicates motion and appearance feature fusion.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ni et al. [23]</td>
<td>73.50</td>
</tr>
<tr>
<td>Zhu et al. [33]</td>
<td>87.00</td>
</tr>
<tr>
<td>Cho et al. [2]</td>
<td>96.03</td>
</tr>
<tr>
<td>Cheng et al. [1] (MF)</td>
<td>93.20</td>
</tr>
<tr>
<td>Cheng et al. [1] (MAF)</td>
<td>96.20</td>
</tr>
<tr>
<td>HOG3D + 3-layer $d^2$RNN</td>
<td><strong>98.24</strong></td>
</tr>
</tbody>
</table>

We compare $d^2$RNN model with other non-LSTM state-of-the-art algorithms in Table 6.2 and Table 6.3. In addition, Fig. 6.8 shows the confusion matrices obtained by [1] and our proposed method.

From Table 6.2, we can see that $d^2$RNN model generally achieves better performance than traditional non-LSTM methods. Those traditional solutions for group activity algorithms need human supervision to acquire accurate human object trajectories from the videos. According to [1], they acquired human bounding boxes using existing tracking tools, which requires manual annotation for bounding box initialization. This constraint prevents their method from being used in automatic or real-time applications. On the contrary, $d^2$RNN models can outperform these traditional methods without the aid of manual operation, enabling broader applications for group behavior analysis. In addition, since traditional models rely on such special assumptions, they might not be applicable to other types of sequences which do not satisfy these assumptions. In contrast, the proposed $d^2$RNN model is a general-purpose model, not being tailored to any specific type of sequences, holding the potential for other sequence-related applications.

Zhuang [76] reported an accuracy of 99.25% on NUS-HGA dataset. It is worth noting that they used a combination of Deep VGG network [91] with stacked LSTMs, where the Deep VGG network plays a crucial role in reaching such performance. More importantly, to avoid overfitting for the complex model, they applied data augmentation on the training set to increase the diversity of
training sequences. While in our experiments, no data augmentation technique is used.

Table 6.3: Performance comparison of deep models on the NUS-HGA dataset.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HOG3D + 3-layer (d^2)RNN</td>
<td>98.24</td>
</tr>
<tr>
<td>ResNet-50 [98]</td>
<td>98.03</td>
</tr>
<tr>
<td>SENet-154 [99]</td>
<td>98.83</td>
</tr>
<tr>
<td>ResNet-50 + 3-layer (d^2)RNN</td>
<td><strong>98.92</strong></td>
</tr>
<tr>
<td>SENet-154 + 3-layer (d^2)RNN</td>
<td><strong>99.13</strong></td>
</tr>
</tbody>
</table>

In Table 6.3, we compare the performances of deep models on the NUS-HGA dataset. It can be seen that \(d^2\)RNN can achieve very competitive performance using handcrafted HOG3D feature, compared with recent CNN deep methods ResNet-50 and SENet-154. While using ResNet or SENet feature, \(d^2\)RNN can further improve the state-of-the-art performance.

![Figure 6.8: Confusion Matrices obtained by Cheng et al. [1] (left) and 3-layer \(d^2\)RNN (right) on the NUS-HGA dataset.](image-url)
RESULTS ON THE VIOLENT-FLOWS DATASET

To evaluate our method on the Violent-Flows dataset, we follow the standard 5-fold cross-validation protocol in [21] and report the results in terms of mean accuracy.

Figure 6.9: Comparison of the proposed $d^2$RNN model with other LSTM models on the Violent-Flows dataset.

Fig. 6.9 compares the results of LSTM models on the Violent-Flows dataset. For the dRNN models, we report the accuracy up to the 2nd-order of DoS. For stacked LSTMs and $d^2$RNN, we report the accuracy up to 3 layers.

From the results, dRNN, stacked LSTMs, and the proposed $d^2$RNN outperform the conventional LSTM algorithm with the same HOG3D feature. $d^2$RNN outperforms both dRNN and stacked LSTMs, which demonstrates the effectiveness of learning intrinsic dynamical patterns present in
video sequences with both deeper layers of LSTMs and higher-orders of DoS. While deeper layers enable greater model complexity, higher-orders of DoS strengthen the model’s ability to detect salient spatio-temporal structures. In the meantime, the stacked deep layers in space naturally reveal the salient dynamics over time, and decrease the chance of misaligned DoS in different orders. dRNN model suffers from the distortion of salient motion patterns due to the combination of different orders DoS. On the other hand, stacked LSTMs ignore the salient spatio-temporal structures, by simply stacking homogeneous layers onto the model.

It is worth pointing out that for stacked LSTMs and $d^2$RNN, the 3-layer architectures do not improve the performance much compared to their corresponding 2-layer models on the Violent-Flows dataset. For stacked LSTMs, the performance is even slightly decreasing. The reason is that with deep layers in the architecture, the model is getting high in complexity and has a larger chance of overfitting.

We show the confusion matrices of 2nd-order dRNN, 3-layer stacked LSTMs, and 3-layer $d^2$RNN on the Violent-Flows dataset in Fig. 6.10. From Fig. 6.10, we can see that $d^2$RNN can effectively detect the violent scenes, which demonstrates the superiority of using individual orders of DoS. Stacked LSTMs, without using DoS, perform less desirably in recognizing violent activities. This is probably due to the lack of strength in detecting motion saliency.

In Table 6.4, we compare 3-layer $d^2$RNN with the traditional methods on the Violent-Flows dataset. $d^2$RNN model outperforms the traditional non-LSTM methods with handcrafted HOG3D feature, which again demonstrates our model’s effectiveness in learning more complex dynamical patterns via deep-stacked layers and detecting spatio-temporal saliency via a hierarchy of DoS.

In Table 6.5, we compare the performances of deep models on the Violent-Flows dataset. Since Violent-Flow dataset contains much more scene semantics than motion information, it enhances the CNN’s strength for scene modeling while diluting the advantage of RNN for analyzing motion.
Figure 6.10: Confusion Matrices obtained by 2nd-order dRNN (left), 3-layer stacked LSTMs (middle), and 3-layer $d^2$RNN (right) on the Violent-Flows dataset.

Table 6.4: Performance comparison of $d^2$RNN vs. traditional methods on the Violent-Flows dataset.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Violent Flows [21]</td>
<td>81.30</td>
</tr>
<tr>
<td>Common Measure [74]</td>
<td>81.50</td>
</tr>
<tr>
<td>Hist of Tracklet [50]</td>
<td>82.30</td>
</tr>
<tr>
<td>Substantial Derivative [75]</td>
<td>85.43</td>
</tr>
<tr>
<td>Holistic Features [52]</td>
<td>85.53</td>
</tr>
<tr>
<td>HOG3D + 3-layer $d^2$RNN</td>
<td><strong>86.58</strong></td>
</tr>
</tbody>
</table>

dynamics. Thus it is understandable that $d^2$RNN with HOG3D feature cannot beat the performance of deep features such as ResNet and SENet. However, when armed with deep CNN features, $d^2$RNN can further boost performances of the state-of-the-art deep models.

Table 6.5: Performance comparison of deep models on the Violent-Flows dataset.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HOG3D + 3-layer $d^2$RNN</td>
<td>86.58</td>
</tr>
<tr>
<td>ResNet-50 [98]</td>
<td>91.93</td>
</tr>
<tr>
<td>SENet-154 [99]</td>
<td>92.28</td>
</tr>
<tr>
<td>ResNet-50 + 3-layer $d^2$RNN</td>
<td><strong>92.93</strong></td>
</tr>
<tr>
<td>SENet-154 + 3-layer $d^2$RNN</td>
<td><strong>93.52</strong></td>
</tr>
</tbody>
</table>
RESULTS ON THE UCF101 DATASET

In this section, we show the experimental results of our proposed \(d^2\)RNN model on the more challenging dataset UCF101.

![Comparison of LSTM models on UCF101 dataset](image)

Figure 6.11: Comparison of the proposed \(d^2\)RNN model with other LSTM models on the UCF101 dataset.

We show the performance comparison of \(d^2\)RNN model against dRNN and stacked LSTMs in Fig. 6.11. Similar to the previous experiments, we report the accuracy up to the 2nd-order of DoS for the dRNN and \(d^2\)RNN models. For stacked LSTMs, we report the accuracy up to 3 layers, which is the same highest layer as \(d^2\)RNN. All the LSTM models use the same SENet CNN feature. From the figure, it can be seen that the results are consistent with the above results from previous datasets. The conventional LSTM yields the lowest performance compared to other LSTM models. This can be explained that conventional LSTM uses neither DoS or deep layers to capture motion saliency presented in given video frames, thus results in poorer performance.
As expected, higher orders of DoS generate better performance for both dRNN and \(d^2\)RNN since higher orders of DoS incorporate more information change into the network. We could notice that the performance increase for \(d^2\)RNN is larger than dRNN. This again justifies our previous discussion that the weighted combination of all the orders of DoS distorts the detected motion patterns. \(d^2\)RNN uses individual orders of DoS on each layer, which help preserve and align the salient dynamic patterns.

Comparing the results of \(d^2\)RNN against stacked LSTMs, we find out that \(d^2\)RNN consistently outperforms stacked LSTMs given the same number of deep layers. This can be explained that the higher orders of DoS from \(d^2\)RNN detects salient spatio-temporal structures that cannot be captured by conventional LSTM layers used in stacked LSTMs. The results again strengthen our claim of the advantages of heterogeneous LSTM layers used in \(d^2\)RNN over the homogeneous ones in stacked LSTMs. To sum up, the above results and analysis demonstrate the superiority of \(d^2\)RNN over conventional LSTM, dRNN, and stacked LSTM.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet-50 [98]</td>
<td>88.7</td>
</tr>
<tr>
<td>SENet-154 [99]</td>
<td>89.2</td>
</tr>
<tr>
<td>ResNet-50 + 3-layer (d^2)RNN</td>
<td>\textbf{89.8}</td>
</tr>
<tr>
<td>SENet-154 + 3-layer (d^2)RNN</td>
<td>\textbf{90.3}</td>
</tr>
<tr>
<td>Two-stream CNN [100]</td>
<td>88.0</td>
</tr>
<tr>
<td>TDD [101]</td>
<td>90.3</td>
</tr>
<tr>
<td>C3D [102]</td>
<td>82.3</td>
</tr>
<tr>
<td>P3D [103]</td>
<td>88.6</td>
</tr>
<tr>
<td>Two-stream I3D [104]</td>
<td>\textbf{98.0}</td>
</tr>
</tbody>
</table>

In Table 6.6, we compare 3-layer \(d^2\)RNN with recent deep models on the UCF101 dataset. It can be found out that \(d^2\)RNN improves the performances of deep models such as ResNet and SENet. \(d^2\)RNN with SENet can outperform or compete with and a number of recent deep models, such as two-stream CNN [100], TDD[101], C3D[102], and P3D[103]. Two-steam I3D, on the other
hand, utilizes the large-scale Kinetics dataset [104] for pre-training and sophisticated two-steam 3D architectures, achieved the best accuracy.

**SUMMARY**

In this chapter, we present a novel LSTM model deep differential Recurrent Neural Network ($d^2$RNN), which integrates stacked LSTMs and Derivative of State (DoS). Instead of simply stacking homogeneous LSTM layers, $d^2$RNN stacks multiple levels of LSTM cells with individual and increasing orders of DoS. Our model inherits the strength of stacked LSTMs to model more complex dynamical patterns than the conventional LSTM. Besides, it further enables the ability to detect salient spatio-temporal structures via the hierarchy of DoS. On the other hand, $d^2$RNN differs substantially from dRNN. Instead of using the combination of different orders of DoS, which has been shown suboptimal, our model modulates the LSTM gates with individual orders DoS and mitigates the problem of information distortion. We demonstrate our model’s superiority on human activity datasets by showing that $d^2$RNN outperforms LSTM, dRNN, and stacked LSTMs. Even in comparison with the other state-of-the-art methods based on strong assumptions about the motion structure of activities being studied, the general-purpose $d^2$RNN model still demonstrates competitive performance. The proposed model can achieve competitive performances compared to state-of-the-art deep models with handcrafted features. Combined with CNN deep model, our model can further boost the performances for deep learning architectures, such as ResNet and SENet. In future work, we will explore the potential of $d^2$RNN in broader applications, such as speech recognition, music synthesis, online handwriting recognition, video captioning, and gesture recognition.
Human activity recognition remains an important and unsolved problem in artificial intelligence and computer vision research area. Practical applications, such as video surveillance and public security, have to rely on the detection and classification of human activities. However, due to the complex dynamical motion patterns, lighting change, and occlusions, it could take years of collective research efforts to fully solve this problem. In this dissertation, we discuss the three sub-problems of human activity recognition, e.g., individual human recognition, group behavior analysis, and crowd activity recognition and propose a new family of LSTM to handle the above problems.

The Long Short-Term Memory (LSTM) recurrent neural networks have strength in modeling complex sequential data. Thanks to LSTM’s special gating schemes, it can learn the representations from long input sequences. Thus, LSTMs have been utilized by many pieces of researches to approach the human activity recognition problem. However, when gating the information that should be memorized through time, the conventional LSTMs do not consider the impact of spatio-temporal dynamics corresponding to the given motion patterns. In other words, the saliency of the motion pattern is not explicitly modeled by the traditional LSTMs. This key observation serves as the fundamental basis of the dissertation. Below, we summarize the three proposed models and their major contributions.

Firstly, we introduce a new family of LSTMs, differential Recurrent Neural Networks (dRNN). dRNNs extends LSTM’s structure by modeling the dynamics of states evolving over time. Considering the different orders of Derivative of States (DoS) for dRNN, the conventional LSTM is,
in fact, a special form and base model of the proposed dRNNs as it utilizes a single order DoS. The new structure of dRNN 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 effective for the representation of the long short-term dynamics of human activities. Based on the energy analysis of Derivative of States (DoS), we further introduce the SEP pooling strategy which can select the most salient hidden states and generate more discriminative representation for video sequences. Experimental results on human action, group activity, and crowd behavior datasets demonstrate that the dRNN model outperforms the conventional LSTM model. Armed with the SEP pooling strategy, the dRNN model can further enhance the performance. In the meantime, even in comparison with the other state-of-the-art approaches based on strong assumptions about the motion structure of actions being studies, the proposed general-purpose dRNN model still demonstrates much competitive performance on both single-person and multi-person activity problems.

In group activity recognition and crowd analysis, visual ambiguity and occlusion occur more frequently. To better understand the scene semantics and human appearance, we propose an end-to-end deep architecture Convolutional Differential Recurrent Neural Networks (CDRNN) for group behavior crowd scene understanding. Our model consists of convolutional neural networks and stacked layers of differential recurrent neural networks. CDRNN directly takes the raw image sequences as the input and does not require additional handcrafted flow-based or trajectory-based feature representation. It also works with crowds of high density and low mobility. Performance studies on three public crowd datasets have shown that the proposed technique significantly outperforms state-of-the-art methods.

Lastly, we present a novel LSTM model deep differential Recurrent Neural Network ($d^2$RNN), which integrates stacked LSTMs and Derivative of State (DoS). Instead of simply stacking homogeneous LSTM layers, $d^2$RNN stacks multiple levels of LSTM cells with individual and increasing
orders of DoS. Our model inherits the strength of stacked LSTMs to model more complex dynamical patterns than the conventional LSTM. In addition, it further enables the ability to detect salient spatio-temporal structures via the hierarchy of DoS. On the other hand, $d^2$RNN differs substantially from dRNN. Instead of using the combination of different orders of DoS, which has been shown suboptimal, our model modulates the LSTM gates with individual orders DoS and mitigates the problem of information distortion. We demonstrate our model’s superiority on human activity datasets by showing that $d^2$RNN outperforms LSTM, dRNN, and stacked LSTMs. Even in comparison with the other state-of-the-art methods based on strong assumptions about the motion structure of activities being studied, the general-purpose $d^2$RNN model still demonstrates competitive performance. The proposed model can achieve competitive performances compared to state-of-the-art deep models with handcrafted features. Combined with CNN deep model, our model can further boost the performances for deep learning architectures, such as ResNet and SENet.

FUTURE WORK

In this dissertation, we have shown the success of employing the new family of differential Recurrent Neural Networks (dRNN) for modeling human activity in video and 3D human activity datasets. Nevertheless, this is not the end of the research, many other interesting questions are still open and demand further investigation. We summarize the future work as follows.

The first promising direction is to explore the potential of dRNN and $d^2$RNN in broader applications. As demonstrated in the dissertation, the Derivative of States (DoS) is sensitive to the dynamical saliency in the sequential data. For any application involving the need to detect such saliencies, such as speech recognition, music synthesis, online handwriting recognition, and gesture recognition, differential Recurrent Neural Networks could achieve better performance than traditional
LSTM. When combined with the most recent Convolutional Neural Networks (CNN), it would be promising that dRNN and $d^2$RNN can further improve the state-of-the-art performances.

In this dissertation, we model the dynamical saliency from sequential data using Derivative of States (DoS). DoS can represent the change in the information gain inside the internal LSTM memory. Intuitively, it could work for human activity recognition and potential broader application as mention above involving detecting sequential saliency. The extensive experiments showed in this dissertation have demonstrated our thoughts. Now, we would like to question if the opposite of DoS, e.g., Integral of States (IoS) could work well for certain applications. By comparison, IoS can model the accumulated information change within the LSTM cell. It would be promising if IoS is utilized in stock price prediction. The reason is as follows. Give the sequential data from stock price prediction, in each time step we are provided with the stock price change over the previous time step. IoS naturally models the accumulated price change and eventually represents the overall gain or loss within a certain period of time. Paired with reinforcement learning techniques, we can further extend the stock price predictor into an automatic stock trading system. This could be an interesting topic as it has both substantial research and financial values.
LIST OF REFERENCES


115


