Exploring Deep Learning for Video Understanding

by

Yikang Li

A Dissertation Presented in Partial Fulfillment
of the Requirement for the Degree
Doctor of Philosophy

Approved November 2020 by the
Graduate Supervisory Committee:

Baoxin Li, Chair
Lina Karam
Robert LiKamWa
Yezhou Yang

ARIZONA STATE UNIVERSITY

December 2020

ABSTRACT

Video analysis and understanding have obtained more and more attention in recent years. The research community also has devoted considerable effort and made progress in many related visual tasks, like video action/event recognition, thumbnail frame or video index retrieval, and zero-shot learning. The way to find good representative features of videos is an important objective for these visual tasks.

Thanks to the success of deep neural networks in recent vision tasks, it is natural to take the deep learning methods into consideration for better extraction of a global representation of the images and videos. In general, Convolutional Neural Network (CNN) is utilized for obtaining the spatial information, and Recurrent Neural Network (RNN) is leveraged for capturing the temporal information.

This dissertation provides a perspective of the challenging problems in different kinds of videos which may require different solutions. Therefore, several novel deep learning-based approaches of obtaining representative features are outlined for different visual tasks like zero-shot learning, video retrieval, and video event recognition in this dissertation. To better understand and obtained the video spatial and temporal information, Convolutional Neural Network and Recurrent Neural Network are jointly utilized in most approaches. And different experiments are conducted to present the importance and effectiveness of good representative features for obtaining a better knowledge of video clips in the computer vision field. This dissertation also concludes a discussion with possible future works of obtaining better representative features of more challenging video clips.

I dedicate this dissertation to my beloved family, my wife Yuxin Meng, and my newborn angel Leonard Meng Li.

ACKNOWLEDGEMENTS

First and foremost, I would like to thank Dr. Baoxin Li. He is dedicated, instructive, and erudite as a mentor and a professor. Only with his endless professional advises and help, I am able to complete my Ph.D study. He always gives me a lot of advises on my research direction and habit which would benefit a lot through all my life. He also tries to prevent me from getting into dilemma and point out a new direction when I am lost.

I would also like to thank Dr. Lina Karam, Dr. Robert LimKaWa, and Dr. Yezhou Yang for giving me a lot of helpful suggestions on my comprehensive proposal and the Ph.D dissertation and several heuristic advises on my future research. I also really appreciate their precious time to serve as my committee members.

Furthermore, I would like to thank all of the lab members in Dr. Baoxin Li's research group. I get a lot of inspirations from the discussions with them. They also help me a lot in both academic and daily life. Moreover, I learn a lot of life philosophy from them. These are my previous and current lab members: Dr. Xu Zhou, Dr. Lin Chen, Dr. Zhigang Tu, Dr. Jun Cao, Dr. Parag Chandakkar, Dr. Ragav Venkatesan, Dr. Yilin Wang, Dr. Qiongjie Tian, Steve Sheng-shung Hu, Vijetha Guttupalli, Tianshu Yu, Kevin Pak Lun Ding, Yuzhen Ding, Yaoxin Zhuo, Riti Paul, Nupur Thakur, Sachin Chhabra, Jiuxu Chen.

At the last, I give my appreciations to my friends and family. My friends: Mr. Shuai Zhao, Mrs. Hu Tao, Mr. Song, Mrs. Ke Zhao, Mr. Jeff, and Ms. Claire give me a lot of support and help in my daily life so that I can put more attention on my academic area. And my beloved family, my father Mr. Zhendong Li, my mother Mrs. Shipin Duan, my father-in-law Mr. Hanqing Meng, and my mother-in-law Mrs. Caixia Yu, supports my life unconditionally even from another continent far away. Most importantly, I would like to thank my wife, Yuxin Meng, to give me love, care and support for everything. Without their support and help, it is hard for me to complete my Ph.D study.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

Videos can be seen as a reflection of the real world. Data streams and visual information, which are essential in videos, will demonstrate the movement of objects, the interactions between humans, and the understanding of different scenes. Recently, with the rapid development of hardware/software equipment and the growth of demand for daily living, video-based communication media, like YouTube, Twitter, Facebook, and TikTok, are playing a more important role in recording people's daily life and sharing moods with friends. Those video clips data are key to understanding and analysis of human being's daily behaviors and human-computer interaction as well. In the past few decades, researchers have also devoted efforts to collecting informative video clips and analyzing corresponding internal knowledge. Especially, video-based event/activity recognition or related visual tasks make more contributions to the computer vision field.

The video-based event/activity recognition has brought about enormous and important challenges to computer vision as well. The computer vision research community has made progresses in many related tasks (e.g., action recognition Feichtenhofer *et al.* (2016); Simonyan and Zisserman (2014a); Veeriah *et al.* (2015); Wang *et al.* (2015); Donahue *et al.* (2015); Wang *et al.* (2016a); Bilen *et al.* (2016); Lan *et al.* (2015); Du *et al.* (2015), temporal localization Shou *et al.* (2016); Chao *et al.* (2018), video question answering Tapaswi *et al.* (2016); Antol *et al.* (2015), video summarization Gong *et al.* (2014); Lee *et al.* (2012); Zhang *et al.* (2016), just to name a few). And one fundamental and important learning process in these video-based visual tasks is to obtain a representative feature and capture the sequential context in the video clips. Compared with static images, capturing the information from video clips is more difficult since the sequence of video frames capturing the information along the temporal axis. The temporal attributes of such information may be the defining feature of the underlying actions/events, given that the appearance of the objects/scene can vary dramatically. For instance, the event “a person opening the trunk of a sedan” may involve a variety of cars of different models/makes/colors, with the person having different appearances as well, but the sequence of movements of the person interacting with the rear end of a sedan is typically distinctive. In this sense, the key of understanding the video lies not only on correctly detecting the objects, but also on identifying specific temporal interaction patterns. Therefore, the way to obtain a good representative feature which containing both spatial and temporal information is the core technique for video-based visual tasks.

In general, video representative features are obtained by encoding spatial (e.g. histogram of gradient) and temporal (e.g. trajectory or motion flow) information to a compact 2D feature with a much smaller size compared to the whole video clip size. This kind of compact 2D feature is much easier to be leveraged on the related visual tasks since the low dimension feature space requires less computational power while maintaining the video information. With the success of the deep neural networks on visual tasks in recent years, the deep-learning-based methods are more commonly used to capture the information from video clips. The representative features are extracted by utilizing the Convolutional Neural Network (**CNN**) or Recurrent Neural Network (**RNN**) or both. This dissertation discusses developing algorithms by leveraging both CNN and RNN to extract features from video clips for several visual tasks and the efficiency of those deep-learning-based approaches. Furthermore, a new learning strategy is proposed in this dissertation for spreading out the visual features in a bounded space for several specific visual tasks. The idea of this new learning strategy is essential to learn a more diverse feature space which is more suitable for understanding the spatial or semantic similarity of video features.

1.1 Overview of Video Representative Features

In recent decades, the rapid development of video technology in many related areas like social media results in a booming growth of video contents. The understanding and analysis of video contents like action or events become more and more important in several aspects from surveillance to personal daily life. To have a better knowledge of video content, the processing of videos at different levels plays roles and causes different challenges. The processing of low-level features requires robustness against errors, the processing of mid-level features demands scale-invariant representative information, and the processing of high-level needs the semantic knowledge of human activities. Usually, to obtain features at different levels require different approaches. But the essence of obtaining features of videos is laying in extracting spatial and temporal information from video frame sequences.

Generally, compared with static images, the temporal information provides extra data and knowledge along the time axis. Therefore, combining the temporal information along with the spatial data to form a new representative feature for video contents. The conventional techniques utilize hand-crafted descriptors which are derived from the approaches of image processing, such as Spatial-Temporal Interest Points Laptev and Lindeberg (2003), HOG/HOF Laptev *et al.* (2008) and Dense Trajectory Wang and Schmid (2013). The hand-crafted descriptors combine the gradients which will describe the spatial information along with optical flow which provides the temporal information. However, those hand-crafted features usually require high dimension to preserve both the spatial and temporal information. Therefore, the encoding techniques, such as bag-of-visual-words or fisher vector encoding, are leveraged to encode the high dimension and redundant visual features into low dimension and compact features to represent the video clips. Even though those hand-crafted features are robust to background clutter, illuminance changes, and video noise, they are lack discriminative capacity and flexibility of embedding the semantic information.

As a result, researchers are seeking a better and more flexible approach to extract spatial-temporal information. Thanks to the success of deep neural networks in visual tasks, utilizing deep-learning-based methods to represent video clips has become more and more popular in recent years. Compared with conventional hand-crafted descriptors, the deep-learning-based methods learn the spatial and temporal information depending on the objective functions. Moreover, the non-linear computation in the deep-learning-based methods makes the learned feature space more distinguishable for visual tasks like recognition and classification. This top-down processing is more capable of obtaining essential information since the object functions will lead the learnable parameters to be tuned to reach the aim of the specific tasks.

Specifically, the convolutional neural network (CNN) achieves a great improvement on image classification task Krizhevsky *et al.* (2012) in 2012. Thereupon, more and more CNN based methods are proposed to solve many challenging visual tasks. However, because CNN utilizes the learnable 2D spatial filters to fulfill the objective, CNN cannot be simply applied to obtain the video representative features. Thus, applying the motion information to CNN based methods draws more attention afterward. Ji *et al.* (2013) proposed a 3D CNN structure to capture the temporal information by expanding the convolutional filter along the time axis. Inspired by this paper, many other papers which are related to leveraging the 3D CNN structure to obtain the video representative features, like Tran *et al.* (2015) and Carreira and Zisserman (2017), are proposed and achieve a better performance in recognition and classification visual tasks.

Another kind of method of implementing optical information into 2D CNN is utilizing the 2D CNN to obtain the information from the preprocessed optical flow maps which can be viewed as 2D images containing motion information. Thus the 2D CNN can extract the motion feature with convolutional filters. Simonyan and Zisserman (2014a) introduces a two-stream structure to fusee both CNN appearance and optical flow features to form video representations. Feichtenhofer *et al.* (2016) inherits the method from Simonyan and Zisserman (2014a) and utilizes the feature fusion work to obtain a better video feature. The Chéron *et al.* (2015) also leverage the two-stream structure and learn the pose patch information to extract better video representations for action recognition.

Because of the capability of obtaining sequential data, the Recurrent Neural Network (RNN) is usually implemented in the language/text related tasks to obtain the order information of words. Therefore, the RNN can be also implemented for video feature extraction because the video clips can be viewed as sequences of frames. Usually, the spatial features are extracted from a pretrained CNN or hand-crafted descriptors for each frame to form a new feature vector sequence. Then the newly formed feature vector sequence is fed into an RNN to obtaining the sequential information. The extracted sequential information can be regarded as the motion feature involving the spatial information so that it can fulfill the visual tasks like action/event recognition or classification. Baccouche *et al.* (2010) proposed a method based on using the Long-Short Term Memory (LSTM), which is a kind of RNN algorithm, to encoding the hand-crafted descriptors for action recognition. Inspired by Baccouche *et al.* (2010), Donahue *et al.* (2015) is the first one to combine CNN features with RNN structure. This CNN-RNN based structure has the ability to be easily embedded into high-level semantic feature space as well since the RNN outputs the feature vector for each unit and it can be applied with captions or texts. Veeriah *et al.* (2015) also applies RNN with hand-crafted features HOG3D to do the action recognition. Different from those methods that applying 2D feature extraction to obtain the frame feature vectors, the Xu *et al.* (2019) utilizes a 3D CNN to obtain the feature for a block of frames and then implement the feature vectors to an RNN structure to extract video features.

With the rapid development of techniques in natural language processing, the techniques which can be used to obtain better sequential information can also be applied to video feature extraction. The attention-based methods like transformer Vaswani *et al.* (2017) and BERT Devlin *et al.* (2018) proposed an attention technique with convolution computation to deal with sequences. Those methods do not contain RNN structure since the disadvantages of RNN such as gradient vanishing problems have a bad influence on final performance when dealing with very long sequences. Following this trend, Girdhar *et al.* (2019) has already proposed a transformer-based video action recognition method and Wang *et al.* (2018) proposed a self-attention based method to boost the performance on action recognition tasks. The advantage of those attention-based methods is that they do not have the concern of gradient vanishing problem since most of them are based on CNN structure. The future work of video representative feature extraction should be focused on the direction of techniques that mitigate the gradient vanishing problem.

1.2 Goals and Motivation

The goal of this dissertation is to propose deep-learning-based approaches by utilizing both CNN and RNN to extract video representative features and demonstrate the efficiency and effectiveness of those extracted video features by several different visual tasks like action/event recognition, zero-shot-learning problem, semantic embedding, and video retrieval.

For those video related visual tasks, there are several challenging problems remains. In this dissertation, all the newly proposed algorithms are inspired by trying to solve those challenging problems by enhancing the capacity of obtaining more informative representative features of the same size. The motivations can be highlighted as below.

1. The recent rapid growth of action categories makes conducting video annotation an expensive, challenging, and time-consuming task. It is difficult and costly to collect a satisfactory amount of annotated spatial-temporal segments of videos. Building a bridge between human-level semantic knowledge and the deep-learning-based visual features provide a solution to mitigate those issues. A good video representative feature can connect with human-level semantic descriptions so that the information can be shared to enable unseen classes to be formed in terms of their semantic descriptions rather than expanding the training data size. Moreover, the good video representative feature sharing the semantic knowledge also provides a good understanding of actions/events that happened in the video clips.

2. With the ever-increasing amount of video data generated by surveillance systems, automatic capabilities for processing/analyzing surveillance videos have become imperative. However, an action/event may occur in only a short period of time and/or within a specific local region of the visual field of view in surveillance video data and thus obtaining a video feature for classifying the action/event is challenging in surveillance video clips.

3. Recently, more challenging video datasets which typically contain video clips with complex sub-activities in a much longer time period, have brought about new two challenges, complexity in content and excessive/varying length, to video recognition. Moreover, occasionally only the event-level labels are available during the training stage which means the lack of detailed information of sub-activities. Those challenges

of long and complex videos make finding good representative features more difficult. Derived from characteristics of long and complex videos, another problem, different sampling rates for testing video clips, also raised attention.

4. Even though the extracted video features can be good representatives for different video clips, the diversity of the extracted features can be low because of the classification-like or the label-driven or the supervised learning object function of visual tasks will force the features from the same category converge to one feature point in the feature space. The lack of diversity in the feature space can cause problems when several visual tasks like retrieval, hashing, or semantic comparison need to be carried out.

1.3 Contribution

The contribution of this dissertation is described as follow.

1. A more detailed literature review is proposed for different deep-learning-based methods of obtaining the video representative features on different visual tasks. Several benchmark video datasets are also presented in this dissertation.

2. Two two-stream deep learning structures are proposed in this dissertation. For the zero-shot learning problem, the dissertation also provides a view of the domain adaption method which makes zero-shot learning performs better. The proposed two-stream algorithms can embed human-level semantic knowledge into video representative features thus reducing the gap between high-level and low-level features.

3. New incorporation of CNN and RNN structure is proposed in this dissertation. The new deep-learning-based structure can extract spatial and temporal information simultaneously rather than feeding the CNN features into RNN based recognition models. Compared with conventional supervised learning approaches, the newly proposed algorithm is trained weakly-supervised.

4. An RNN based deep learning model is proposed for event-level recognition in this dissertation. This newly proposed approach not only recognizes events in very long and complex videos with only event-level available during the training stage but also maintaining the recognition performance with different sampling rates of testing video clips.

5. A new strategy of utilizing the Determinantal Point Process (DPP) with an end-to-end deep neural network is proposed to spread out the learned visual features in a bounded feature space. A simple but effective algorithm generating a valid sub-gradient of Determinantal Point Process (DPP) in the case where the Determinantal Point Process matrix is not invertible is proposed in this dissertation so that the DPP term can be easily incorporated with deep neural networks and be trained end-to-end for increasing the diversity of obtained feature space.

1.4 Outline

This dissertation is structured as follow.

Chapter 2 propose the two-stream structure deep-learning-based methods. This proposed structure mainly focuses on embedding the human-level knowledge into low-level video features. And two visual tasks, zero-shot learning and video retrieval, are conducted for demonstrating the efficiency and effectiveness of newly proposed deep structure.

Chapter 3 describes a new deep-learning-based approach which combining the CNN and RNN together. This newly proposed combination of CNN and RNN is designed to obtain the spatial and temporal information simultaneously for surveillance video clips. The action recognition and localization visual tasks are implemented in this chapter to show the capability of the proposed method with a weakly-supervised learning process.

Chapter 4 provides a thought on how to recognize events with very long and complex video clips. There are three challenges proposed in this chapter. 1. Only event-level label available (i.e. no sub-activities knowledge is accessible) during the training stage. 2. Performing the event recognition without fine-tuning the feature extraction part (i.e. the deep CNN part before RNN part). 3. Maintaining the performance when the sampling rates of testing video clips are different from the training video samples.

Chapter 5 proposes a simple but effective method to ensemble the Determinantal Point Process (DPP) term with neural network to learn a bounded feature space in which the feature vectors are spread out. Several different visual tasks experiments on different datasets are conducted to present the effectiveness of the proposed algorithm in diversify the feature vectors and the promising performance of practical learning problems. One thing need to be emphasized is that there are only image related datasets utilized in this chapter since the computation burden is low for images compared with videos with the same amount of labeled data. Moreover, the demonstration of the proposed algorithm should be clear enough with images in the spatial domain for the visual tasks.

Chapter 6 concludes this dissertation and makes a summary of the contributions proposed in this dissertation. This chapter also shows the possible future work of obtaining a better video representative features for more challenging visual tasks. Some more challenging problems or tasks are provided in this chapter and analysis of those tasks should help outlining more directions of extracting video representative features.

Chapter 2

DEEP TWO-STREAM STRUCTURE BASED VIDEO FEATURE EXTRACTION

As mentioned above in **the Chapter 2 Introduction** part, the two-stream deep learning structure successfully conduct the another CNN based network to acquire motion information from optical flow maps which are warped based on the theory from Brox *et al.* (2004). Simonyan and Zisserman (2014a) is the first to propose the two-stream structure and the performance on the two benchmark action recognition short video datasets UCF101 Soomro *et al.* (2012) and HMDB51 Kuehne *et al.* (2011) presents the feasibility and the improvement of this deep two-stream network on action recognition visual task.

Therefore, inspired by the idea of two-stream structure, two different deep neural networks are proposed in this dissertation and each of them demonstrates a better capability of extracting the video information by conducting the networks on zero-shot learning problem and semantic video retrieval task.

2.1 Recognizing Unseen Actions in a Domain-adapted Embedding Space

With the sustaining bloom of multimedia data, Zero-shot Learning (ZSL) techniques have attracted much attention in recent years for its ability to train learning models that can handle “unseen” categories. Existing ZSL algorithms mainly take advantages of attribute-based semantic space and only focus on static image data. Besides, most ZSL studies merely consider the semantic embedded labels and fail to address domain shift problem. In this dissertation, a deep two-output model is proposed for video ZSL and action recognition tasks by computing both spatial and temporal features from video contents through distinct Convolutional Neural Networks (CNNs) and training a Multi-layer Perceptron (MLP) upon extracted features to map videos to semantic embedding word vectors. Moreover, this dissertation introduces a domain adaptation strategy named "ConSSEV" – by combining outputs from two distinct output layers of the proposed MLP to improve the results of zero-shot learning. The experiments on UCF101 dataset demonstrate the purposed model has more advantages associated with more complex video embedding schemes, and outperforms the state-of-the-art zero-shot learning techniques.

2.1.1 Introduction

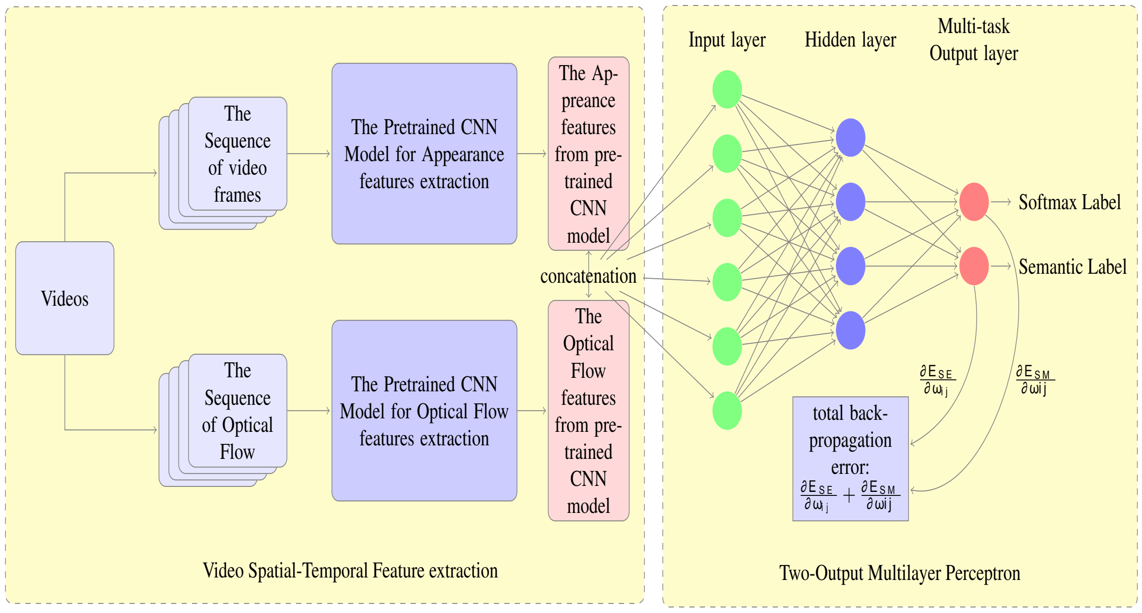
Video-based action recognition has many applications Wang and Li (2009). Recent rapid growth of action categories makes conducting video annotation an expensive, challenging and time consuming task. While conventional classifiers require sufficient training data to achieve acceptable results on action recognition tasks, it is difficult and costly to collect satisfactory amount of annotated spatial-temporal segments of videos. The zero-shot learning (ZSL) algorithm provide a solution to mitigate those issues by connecting human-level semantic descriptions of the action with low-level visual features and allowing different categories to share their information –thus enable new categories to be built in terms of their human descriptions rather than extending the size of the training visual-level data. Three keys are rather important in ZSL algorithm – selecting of visual descriptors, constructing human-level semantic descriptors and the mapping function to map visual to semantic space.

Most existing ZSL algorithms are realised by building human-level attribute model to bridge the visual features and their corresponding semantic space. New categories are then classified in terms of their attributes Lampert *et al.* (2014); Cheng *et al.* (2013). However, it is rather difficult to obtain reliable attribute-based representation for objects, especially for actions Gan *et al.* (2015), and this kind of semantic attribute-based ZSL classifiers suffer from lacking distributed representation of each attribute words.

An alternative approach to the attribute-based method is the Semantic Embedding Space (SES) Mikolov *et al.* (2013b,a). SES is trained by a skip-grim or continuous bag-of-words neural network to map the text words into a word vector space – therefore enable new categories be simply annotated by the similarity and distribution of existing text-string vectors and avoid non-scaleably growth of attribute lists as the emergence of new categories Among all SES models, *Word2Vec* model is considered to be the most efficienct model in maintaining semantic meanings while keeping low model complexity Frome *et al.* (2013); Xu *et al.* (2015a); Norouzi *et al.* (2013).

Although semantic embedding space has demonstrated significant advantages, most ZSL studies only focus on static images semantic embedding since it is particularly difficult to extract reliable feature descriptors which cover both seen and unseen action categories from videos to train the mapping function. Moreover, the presence of amount of neighbour vectors surrounding the mapped vectors in semantic space has been proven to be a challenge for word-based vectors Dinu and Baroni (2014) (i.e., "*hubness*" problem or domain shift problem).

In this section, a deep two-output model is proposed for video ZSL and action recognition purposes by taking advantages of both "soft" SES labels and conventional"hard" binary labels to train a multi-layer perceptron that map CNN visual features to their corresponding semantic meanings. A new strategy called "Convex combination of Similar Semantic Embedding Vectors" (ConSSEV) is also implemented to deal with the domain shift problem. The purposed model not only outperforms state-of-the-art Xu *et al.* (2015a) method on UCF101 Soomro *et al.* (2012) video action dataset on zero-shot learning but also achieve comparative high accuracy with the conventional supervised action recognition classifier on action recognition task.



Videos
The Sequence of video frames
The Sequence of Optical Flow
The Pretrained CNN Model for Appearance features extraction
The Pretrained CNN Model for Optical Flow features extraction
The Appreance features from pre-trained CNN model
The Optical Flow features from pre-trained CNN model
concatenation
Video Spatial-Temporal Feature extraction
Input layer
Hidden layer
Multi-task Output layer
Softmax Label
Semantic Label
$\frac{\partial E_{SE}}{\partial \omega_{ij}}$
$\frac{\partial E_{SM}}{\partial \omega ij}$
total back-propagation error: $\frac{\partial E_{SE}}{\partial \omega_{ij}} + \frac{\partial E_{SM}}{\partial \omega ij}$
Two-Output Multilayer Perceptron

Figure 2.1: The Framework of the Deep Two-output Model for Video ZSL and Action Recognition Purposes.

2.1.2 Methodology

The overall framework of the proposed model is illustrated in Fig. 2.1. This proposed structure is constructed with multi-task learning and two objective functions can be viewed as a compensation for each us to tuning the learnable parameters more capable of obtaining features containing semantic meaning.

In the proposed approach, firstly, the frames and optical flow Brox *et al.* (2004) are extracted from video contents and then pass them into two different pre-trained CNN models which can be obtained by utilizing Chéron *et al.* (2015). The appearance features and optical flow features are collected from the second last fully connected layer (i.e., "fc7") from each CNN. Next both spatial and temporal features are aggregated and concatenated to represent one action by using a sliding window strategy. Then, a two output layer Multi-Layer Perceptron (MLP) is trained by backpropagating errors from semantic labels and fine-tuned on softmax hard binary labels to serve as a mapping function from visual to semantic space. Finally, zero-shot learning (ZSL) is performed by mapping visual features to semantic space vectors through the proposed model. The Convex combination of Similar Semantic Embedding Vectors (ConSSEV) is implemented as a domain shift method as well to boost the performance of the proposed zero-shot learning algorithm.

Visual Feature Extraction

Considering the success of Pose-based Convolution Neural Network Chéron *et al.* (2015) on recognizing human-pose and action, the video representative features are extracted in a similar way. Videos are sampled to RGB frames to represent appearances and the optical flows are computed to represent motions Brox *et al.* (2004). To describe both appearance features f^t_{app} and motion features f^t_{of}, two different pre-trained CNNs on RGB and optical flow frames are utilized respectively and the output of the second last fully-connected layer Chatfield *et al.* (2014) with dimension $k = 4096$ is served as the extracted descriptors for video clips. For RGB frames the publicly available "VGG-f" pre-trained network Chatfield *et al.* (2014) is used while the pre-trained temporal network from Gkioxari and Malik (2015) is applied on optical flows maps. **A Sliding Window Strategy**: a sliding window with size T and step size S on both f^t_{app} and f^t_{of} is applied. Features extracted from the CNNs within the same window are *avg* aggregated to obtain fixed-length video descriptors d^i over T frames

$$d^i_{app} = \frac{1}{T} \sum_{t=t_0}^{t_o+T-1} f^t_{app}(i) \tag{2.1}$$

$$d^i_{of} = \frac{1}{T} \sum_{t=t_0}^{t_o+T-1} f^t_{of}(i) \tag{2.2}$$

$$d^i = [d^i_{app}, d^i_{of}]. \tag{2.3}$$

Finally d^i is the extracted visual features for the i^{st} class with dimension $k = 8192$.

Semantic Embedding Space

The semantic embedding space is constructed with the help of the *word2vec* neural network Mikolov *et al.* (2013a) Mikolov *et al.* (2013b) in this dissertation because of its reliable mapping function between word corpses and mathematical meaning expressions. Through this *word2vec* networks, semantic labels $\{y^i\}_{i=1\dots n}$ are assigned to 500-D vectors $Z^i = g(y^i)$ and are divided by the amount of unique words in $Z^i = \frac{1}{N}\sum_{j=1}^{N} g(y^i_j)$ for normalization purpose.

Mapping Function

Given visual features d^i and semantic embedded space labels Z^i, the goal of the zero-shot learning problem now is to build a projection model: $Z^* = M(d^i)$ that can best map each video to a vector in the corresponding semantic embedding space. Inspired by the idea of Venkatesan and Li (2016), A two-output Multi-Layer Perceptron (MLP) is trained to achieve this goal.

Both the semantic embedding space soft labels Z^i and classification hard binary labels y^i are applied on training the MLP. Two distinct loss functions are calculated and both errors are backpropagated. For semantic soft label loss, a hinge rank loss function (similar to Frome *et al.* (2013)) is utilized to measure the similarity of the semantic output Z^* and semantic labels Z. The hinge rank loss function is defined as:

$$L_{se} = -\sum_{j \neq i} \max[0, m - Z^i \cdot Z^* + Z^j \cdot Z^*] \tag{2.4}$$

where m represents the margin value.

For classification hard binary label loss, the softmax loss function is leveraged because of its robustness for multi-class classification. The probability and loss, of which the softmax layer input Z^* is classified to the j^{th} class (i.e., $\bar{y} = j|Z^*$), are defined as:

$$P(\bar{y} = j|Z^*) = \frac{e^{Z^{*T} w_j}}{\sum_{k=1}^{K} e^{Z^{*T} w_k}} \tag{2.5}$$

$$loss_{sm} = -\sum_{j} 1\{\bar{y} = j\} \log(P(\bar{y} = j|Z^*)). \tag{2.6}$$

Here $1\{\bar{y} = j\}$ equals to 1 when predicted label $\bar{y}$ equals to target label, otherwise it equals to 0.

Both softmax and semantic outputs leverage visual and semantic similarity to train MLP since two outputs share information in input and hidden layer. Moreover, the summation of softmax and semantic loss backpropagate to learn weights in each layer, so that each loss can be serve as a compensation and fine-tuning method for the other one

Zero-shot Learning and Conssev Strategy

Zero-Shot learning is then performed on a completely disjoint dataset from training set. Utilizing the proposed deep-multi output model and *word2vec*, it is possible to project both "unseen" visual contents d^i_{test} and their corresponding word labels y^i_{test} to semantic embedded space vectors (Z^*_{test} and Z^i_{test})). Since both vectors are normalized, a simple cosine similarity is performed to match projected visual contents and labels

$$\bar{y} = \arg\max cos(Z^*_{test}, Z^i_{test}). \tag{2.7}$$

ConSSEV strategy: Due to the disjointedness of train and test set in ZSL, the trained mapping function M may not be the best fit in the case of mapping test set and thus biases the similarity measurement Frome *et al.* (2013); Xu *et al.* (2015a). In this dissertation, a self-adaptive domain shift method is introduced by utilizing both semantic and softmax outputs in the proposed model to adjust both semantic output Z^*_{test} and label prototype Z^i_{test}.

For semantic output, the top K training labels that have the highest similarities with test semantic label vector $\{Z^*_k\}_{k=1...K}$ as that in Eq. (2.7) are first found. Then a new semantic vector $\bar{Z}^*$ is formmed by weighted sum of all k vectors with their corresponding softmax output $P(\bar{y} = k|Z^*_k)$. The adaptive semantic output vector performs better since it penalized "ambiguous feature results" by weighting smaller softmax probabilities ("confidences").

$$\bar{Z}^* = \frac{1}{K}\sum_{k=1}^{K} P(\bar{y} = k|Z^*_k) \cdot Z^*_k \tag{2.8}$$

For label prototype, the same $self - training$ techniques as that in Xu *et al.* (2015a) on the adaptive data vector obtained by Eq. (2.8) is performed as well. The new convex combination of semantic output $\bar{Z}^*$ and test label prototype $\bar{Z}^i_{test}$ are more directly comparable by using Eq. (2.7).

Conventional Video Recognition

The proposed deep two-stream structure with two-output model can also serve as a conventional video recognition classifier by supervised training the mapping function M with all categories. Then each test visual feature d^i_{test} is mapped to a vector Z^* in semantic space through $Z^* = M(d^i_{test})$ and matching them with all projected labels Z^i through Eq. (2.7).

2.1.3 Experiments and Evaluation

Dataset

The proposed model is trained and tested on one of the most popular video action benchmark dataset – UCF101 Soomro *et al.* (2012) which contains 13320 videos from 101 cation categories (e.g."Apply Eye Make","Basketball Dunk", "Brest Stroke", and etc). Videos in each action category are grouped into 25 groups where each group shares some common features, such as background, viewpoints, objects, and etc.

For evaluating ZSL, the same evaluation protocol as in Xu *et al.* (2015a) is applied in this dissertation – 30 independent splits for UCF101 dataset with each split contains a completely disjoint 51 categories for training purpose and 50 for testing purpose. For conventional action recognition, on the other hand, the standard splits ("Three Train/Test Splits") for UCF101 dataset is utilized.

Experiment Setting

Semantic Embedding Space: *Word2Vec* Mikolov *et al.* (2013a),Mikolov *et al.* (2013b) method is leveraged to embed the text labels. A skip-gram text model is trained on a corpus containing 5.4 billion words which are extracted from Wikipedia. Dimension of word vector is set to 500-D to trade-off training complexity and maintaining semantic meanings Frome *et al.* (2013); Xu *et al.* (2015a).

Visual Feature Extraction: To further decrease model complexity, only one frame is sampled for each three consecutive frames from video and the corresponding optical flow maps Brox *et al.* (2004) are computed upon them. Two distinct CNNs are applied to extract features – both contain 5 convolutional and 3 fully-connected layers. The appearance and optical flow features are extracted from the second last fully-connected layer (i.e.,"fc7") which dimension for each feature is 4096. Then the sliding window strategy is utilized to aggregate and concatenate the appearance and optical flow feature vectors into one 8192-D vector.

Mapping function training: A Multi-Layer Peceptron (MLP) is trained with those aggregated visual features for each video clip. Target labels for softmax output are so-called "hard binary labels" (i.e., "1" and "0" for belonging to the specific class or not, respectively), and target labels for semantic embedding output are the 500-D semantic space word vectors. The number of hidden nodes is set to be 1000, learning rate to be 0.001, the momentum to be 0.9 and margin value for hinge rank loss function to be 0.9 based on the result of cross-validation. Moreover, for each splits, five iterations are implemented of all training features with random training order.

Results Comparison: For zero-shot learning, the following methods are implemented as comparisons on the same split data of UCF101: (1) Random Guess: the method randomly guesses one label from the unseen labels. (2) Attribute Based-Indirect Attribute Prediction (IAP) Lampert *et al.* (2014): the method selects the unseen label by the video representation attributes. (3) Convex Combination of Semantic Embeddings (ConSE) Norouzi *et al.* (2013): the method uses the conventional neural network classifier outputs (i.e. softmax probabilities) to weight the training labels and combine the top K embedded labels to denote a new semantic embedding word vectors. (4) Dense Trajectories Based Regression Model with Nearest Neighbour (DTRM+NN) Xu *et al.* (2015a): the model is trained a SVM classifier with RBF kernel on the dense trajectory descriptors Wang and Schmid (2013). This method is treated as the baseline. (5) The proposed deep two-stream strucutre with two-output model with Nearest Neighbour searching: finding the nearest neighbour in terms of maximal cosine similarity. (6) DTRM+NN+ST: Apply Self-training domain shift method with DTRM. (6) The proposed deep two-stream strucutre with deep two-output model with ConSSEV domain shift approach: Apply ConSSEV domain

Table 2.1: Zero-Shot Learning Performance

Method	**Accuracy ± Variation**
Random Guess	2.0
IAP Lampert *et al.* (2014)	12.8 ± 2.0
ConSE Norouzi *et al.* (2013)	10.5 ± 2.0
DTRM + NN Xu *et al.* (2015a)	10.9 ± 1.5
Proposed + NN	$\mathbf{11.3 \pm 2.1}$
DTRM + NN + ST Xu *et al.* (2015a)	15.8 ± 2.3
Proposed + NN + ConSSEV	$\mathbf{26.8 \pm 4.4}$

shift strategy on the proposed model.

For video recognition, the proposed model is validated with the following: (1) Dense Trajectories Wang and Schmid (2013). (2) Binary SVM classifier with RBF kernel (DTRM) Xu *et al.* (2015a). (3) The proposed model Semantic output. (4) The proposed model Softmax output.

Evaluations

The results of **zero-shot learning** are presented in Tab 2.1. All listed methods are significantly better than random guess which shows successful ZSL. Without applying any kinds of domain shifting techniques and only consider the Nearest Neighbour (NN), tje proposed deep two-output model achieves a slightly better performance than existing state-of-the-art semantic-based ZSL (DTRM) – suggesting visual contents are effectively mapped to semantic space vectors that are near to its human-level semantic

Table 2.2: Conventional Action Recognition Performance

Method	**Accuracy**
Dense Trajectories Wang and Schmid (2013)	**75.1**
DTRM Xu *et al.* (2015a)	73.7
Proposed (Semantic)	74.1
Proposed (Softmax)	72.7

meanings. Although the proposed model fails to demonstrate better performance than attribute-based model Lampert *et al.* (2014) when evaluating by NN, it does not suffer from lack of attributes and costly attribute annotation. By applying the developed ConSSEV domain shift strategy, the proposed model significantly outperforms other domain shift counterpart (DTRM+NN+ST).

Overall, the proposed zero-shot learning technique based on MLP has a great performance among the existing state-of-the-art ZSL methods and the ConSSEV domain shift strategy between test and train categories proves to be a significant performance boost on ZSL technique.

The performance of the proposed model conducting **conventional action recognition** task is listed in Tab 2.2. The final results reveal that the proposed approach performs comparatively with the baseline method including Dense Trajectories Wang and Schmid (2013) and Binary SVM classifier with RBF kernel Xu *et al.* (2015a). Thus demonstrating the capability of the proposed deep two-output model on addressing conventional action recognition tasks. The Dense Trajectory Wang and Schmid (2013) in the Tab 2.2 shows the performance is slightly better than the proposed approach. This may due to the sliding window strategy that is used to extract video features. Even though motion information of optical flow is utilized in the proposed model, the averaged features within a sliding window will lose some temporal information of video sequences compared with HOF and MBH features applied in Dense Trajectories Wang and Schmid (2013).

2.1.4 Conclusion

A deep two-stream structure with two-output model to realise the zero-shot learning paradigm on video recognition is proposed in this section 2.1. A domain shift technique, Convex Combination of Similar Semantic Embedding Vectors (ConSSEV), which proves to provide a significant improvement in terms of zero-shot learning accuracy by utilizing the known semantic space to express the unknown semantic space, is purposed. This section shows that the proposed zero-shot learning model with ConSSEV strategy greatly outperforms baseline zero-shot video action recognition techniques. And the deep two-stream structure is presented to be a good method to extract video representative features and it is feasible to combine the semantic features with low-level visual features.

2.2 *Video2Vec*: Two-stream Recurrent Neural Networks for Learning Semantic Spatio-temporal Embeddings of Video Clips

This dissertation proposes a RNN based two-stream structure to learn a better semantic spatio-temporal embeddings for videos to support high-level video analysis. The first step of the proposed embedding method employs a deep architecture consisting of two channels of Convolutional Neural Networks (capturing appearance and local motion) followed by their corresponding Gated Recurrent Unit encoders for capturing longer-term temporal structure of the CNN features. The resultant spatio-temporal representation (a vector) is used to learn a mapping via a Multi-Layer Perceptron to the *word2vec* semantic embedding space, leading to a semantic interpretation of the video vector that supports high-level analysis. This section demonstrates the usefulness and effectiveness of this new video representation by conducting experiments on action recognition, zero-shot video classification, and “word-to-video” retrieval, using the UCF-101 dataset.

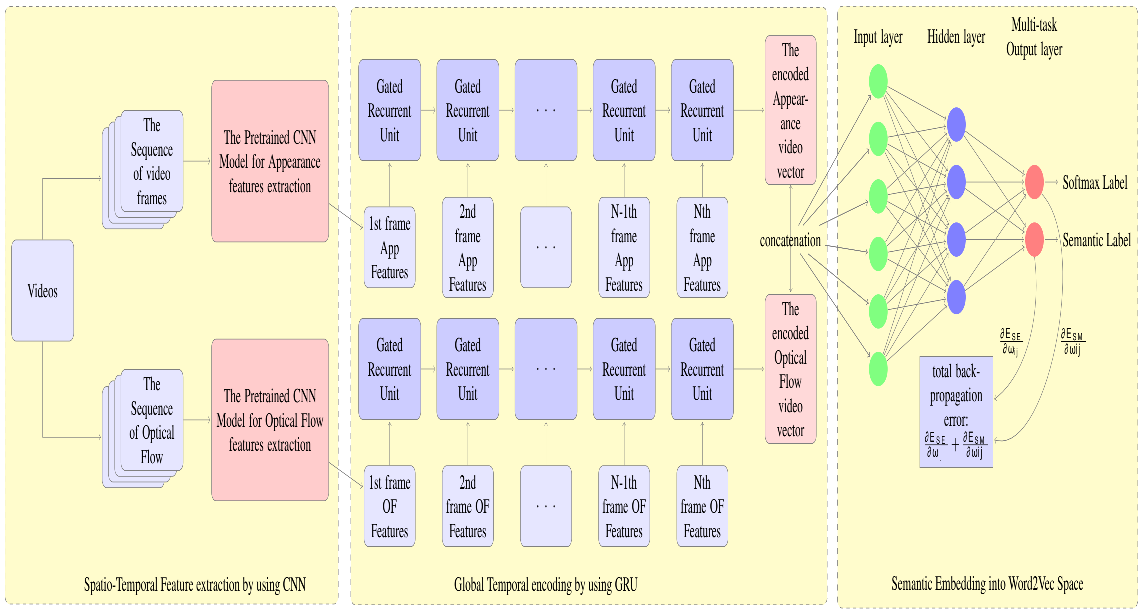
2.2.1 Introduction

Many computer vision applications involve general scene understanding based on videos. Examples include video-based action/event recognition, vision for human-computer interaction, and video surveillance, etc. One fundamental task in these applications is to establish certain mapping from a raw video input to some high-level semantic labels such as action or event categories, or gesture-based commands, etc. Typically, raw video data would first be processed for feature extraction before any technique for establishing the above mapping is applied. The quality of the features, or more generally, the representation of the videos, plays an important role and can have significant impacts on subsequent analysis tasks.

Some well-known video features include HOG3D Kläser *et al.* (2008), STIP Laptev and Lindeberg (2003), and Dense Trajectories Wang and Schmid (2013), all having been widely used in video-based analysis. These, what has been mentioned in Chapter 1, are often called ”hand-crafted” features, since they were deliberately designed based on reasonable considerations. In contrast, techniques relying on deep neural networks for directly learning features from videos have been seen in recent decades. For instance, Simonyan and Zisserman (2014a) proposed a CNN-based architecture was proposed to fuse appearance and optical flow features to form a (frame-level) video descriptor. Typically, features learned from such approaches are based on the output of the last Rectified Linear Unit (ReLU) layer in a CNN that contains many hidden layers acting as progressive feature extractors. Other than using optical flow as part of the input (hence capturing some local motion), the CNN-based approaches like Simonyan and Zisserman (2014a) and the proposed method mentioned in section 2.1 do not have the capacity to model global temporal evolution of the video/features, which may be the key to obtain a better higher-level semantic analysis. A naive approach of averaging frame-level representations to form a global representation would not solve the problem as the temporal structure is no longer maintained.

In this section, the dissertation aims at learning proper vector representations for videos so as to support a set of common semantic analysis tasks. A deep architecture is employed as the basic building blocks for their demonstrated performance. In order to capture the temporal structure of an underlying video, one may utilize Recurrent Neural Networks (RNNs) on top of frame-level CNNs. Two well-known alternatives are the Long-Short Term Memory (LSTM) Hochreiter and Schmidhuber (1997) and Gated Recurrent Unit (GRU) Cho *et al.* (2014), which mitigate gradient vanishing by implementing "gate units" that decide what to "forget" and what to "memorize". In this proposeed approach, GRUs are chosen to further encode frame-level CNN features, since they have comparative performance to LSTM while requiring less computation cost, which is an important consideration for video clips.

Furthermore, recognizing that the learned vector representations, although rich in spatio-temporal information, still lack a semantic organization or clustering that would directly facilitate a higher-level analysis task like action labeling, an additional mapping from the learned vector representations of the videos to the *word2vec* Semantic Embedding Space (SES) Mikolov *et al.* (2013a) is proposed to be learned. This learning task relies on labels of the videos in a training set (data-driven) *and* the SES learned from Wikipedia documents with more than 1 billion words (prior knowledge on semantic meanings of the labels). Hence the final mapping effectively leads to an



Videos
The Sequence of video frames
The Sequence of Optical Flow
The Pretrained CNN Model for Appearance features extraction
The Pretrained CNN Model for Optical Flow features extraction
Spatio-Temporal Feature extraction by using CNN
Gated Recurrent Unit
1st frame App Features
2nd frame App Features
N-1th frame App Features
Nth frame App Features
1st frame OF Features
2nd frame OF Features
N-1th frame OF Features
Nth frame OF Features
Global Temporal encoding by using GRU
The encoded Appearance video vector
concatenation
The encoded Optical Flow video vector
Input layer
Hidden layer
Multi-task Output layer
Softmax Label
Semantic Label
total back-propagation error:
Semantic Embedding into Word2Vec Space

Figure 2.2: Overall Framework of the Proposed Architecture for Semantic Spatio-Temporal Video Embedding.

embedding for the videos into the SES, enabling the utilization of the word (label) semantics for any higher-level analysis task. The overall framework is illustrated in Fig. 2.2. In the proposed deep structure, two GRUs are conducted followed by a two-stream CNN structure which obtains the appearance and motion information for each frame. A MLP which is similar to the proposed structure in the section 2.1 is applied with two loss functions. The details of the proposed structure will be further elaborated in the following sub-sections.

To illustrate the usefulness of such a learned embedding, the proposed approach is evaluated by using three video analysis tasks as case studies: action recognition, zero-shot learning for action classification, and semantic video retrieval. All experiments are based on the commonly-used UCF-101 dataset Soomro *et al.* (2012), for its diversity of contents (and hence challenges for an analysis algorithm) as well as the ready availability of results from many baselines (and hence making it easy to assess the significance of any performance gains).

2.2.2 Related Work

Conventional action recognition tasks utilize *hand-crafted* descriptors such as STIP Laptev and Lindeberg (2003), HOG3D Kläser *et al.* (2008), and dense features Wang and Schmid (2013) to capture spatio-temporal information. Such features have found wide applications in the literature. However, in general hand-crafted descriptors lack semantic and discriminative capacity and thus cannot effectively represent higher-level information Wang *et al.* (2015). In recent years, many deep-learning approaches have been applied to video feature extraction. The Simonyan and Zisserman (2014a) introduced a method to fuse both CNN appearance and optical flow features while the Wang *et al.* (2015) proposed descriptors that pool both low-level hand-crafted features and CNN feature to represent videos.

Despite of the fusion process, the above deep-learning approaches and the proposed method in section 2.1 still do not fully leverage temporal information of the given video. To overcome this limitation, several more recent approaches Srivastava *et al.* (2015); Donahue *et al.* (2015); Ng *et al.* (2015) attempted to encode the video by LSTM. The LSTM can be considered as a gated RNN, which is capable of discovering the implicit temporal structure of the input sequences while avoiding the gradient vanishing problem. In the above literature, representing videos by LSTM was shown to have some advantages when modeling complex temporal dynamics and competitive results on tasks like action recognition have been reported. Unfortunately, all of the above LSTM-based methods encode videos without considering semantic meanings of the representation. As a result, the learned representation does not directly support high-level semantic tasks such as semantic video retrieval (retrieving videos by word descriptions never used in training) and zero-shot video classification (recognizing unseen video categories).

Associating video representations with semantics has been studied in various contexts including content-based video retrieval Li *et al.* (2016); Gitte *et al.* (2014); Veltkamp *et al.* (2013); Snoek and Worring (2009). In Venugopalan *et al.* (2015, 2014); Donahue *et al.* (2015), attempts have been made to generate semantic label sequences from video inputs. However, these efforts do not seem to explicitly associate videos with semantic meaning derived directly from the semantic labels of the videos (but rather relying on external dictionaries). Lacking is a learned embedding of videos that may directly lead to semantic interpretation of a novel video, which may or may not have any textual labels. Such an embedding could lead to a new presentation that supports high-level semantic analysis. The proposed approach in this Chapter attempts to achieve this by learning the mapping between spatio-temporal representations and the label vectors in the *word2vec* semantic space. And compared with the method mentioned in the section 2.1, the proposed method in this section tries to capture a hierarchical temporal information by conducting a two-stream RNN structure.

The proposed two-stream RNN based semantic embedding will easily support zero-shot learning for video classification (to be further illustrated in following subsection). Most existing zero-shot learning techniques focus on static images and many rely on attributed-based representations Lampert *et al.* (2014). In practice, it is difficult to obtain sufficient amount of data for training comprehensive attribute-based representations for a large number of categories. The proposed newly video representation is advantageous on this regard since the basic embedding space derives its semantics from general human knowledge base (e.g., meanings of words learned from Wikipedia documents), and only the last stage of mapping requires video labels to train.

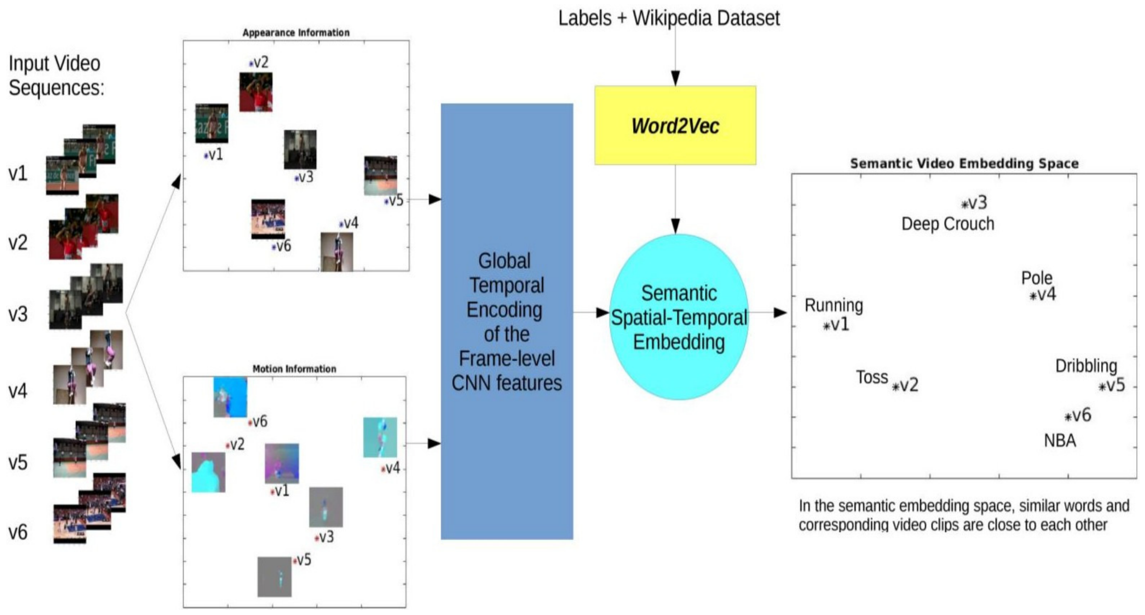

Labels + Wikipedia Dataset
Input Video Sequences:
v1
v2
v3
v4
v5
v6
Appearance Information
Motion Information
Global Temporal Encoding of the Frame-level CNN features
Word2Vec
Semantic Spatial-Temporal Embedding
Semantic Video Embedding Space
Deep Crouch
Pole
Running
Toss
Dribbling
NBA
In the semantic embedding space, similar words and corresponding video clips are close to each other

Figure 2.3: Illustrating the Working of the Proposed Two-stream RNN Model.

2.2.3 Methodology

Given a dataset of video clips with corresponding semantic labels, the goal of the proposed method in this section is to learn a fixed-length vector representation for each video so that the representation captures spatio-temporal information of the underlying video *as well as* the semantics contained in the labels. The proposed approach achieves this goal by a deep architecture demonstrated in Fig. 2.2. The approach consists of three major learning steps. The first step extracts spatial and (local) temporal features using two CNN channels. The second step encodes (more global) temporal structures of the video (in terms of learned CNN features) by GRUs. The third step learns a mapping from the encoded spatio-temporal video representations to a *word2vec* semantic embedding space by an MLP. The proposed architecture is an end-to-end learning model that can be trained by back-propagation with a loss computed by summing the results of the hinge rank loss function and the softmax loss function at the output layer of the model. These steps are elaborated as below.

An illustration of the proposed method is shown in Fig. 2.3. As it is known that video clips carry similar semantic meanings can vary greatly in terms of spatio-temporal features (e.g., videos v6 "NBA" and v5 "Dribbling" are far from each other when only temporal encoding is performed). The global temporal structure of these features is encoded by the two-stream GRUs structure. A learned mapping further embed the GRU-encoded spatio-temporal feature into a semantic embedding space, where diverse videos sharing the similar semantics cluster together (e.g., "NBA" and "Dribbling" videos are projected to similar coordinate after embedding).

CNN-based Spatio-temporal Feature Sequences

With a collection of video clips V where each clip $v \subset V$ contains a sequence of frames with a specific order $\{f_1, ..., f_n\}$ and label l_v. As described in the section 2.1, the RGB frames are first pulled out to represent spatial information and compute the optical flows maps from the frames to represent (local) motion information. Both the optical flows maps and the RGB frames are processed at 10fps, and the optic flows maps are computed by using the implementation described in Brox *et al.* (2004) and the section 2.1.

As the same as the way of extracting frame features mentioned in section 2.1, two pre-trained CNN models are then used to extract appearance f_{v_app} and optical flow features f_{v_of} respectively. The "VGG-f" Simonyan and Zisserman (2014b) pre-trained model on the ImageNet ILSVRC-2012 dataset Deng *et al.* (2009) is responsible for extracting appearance features while the pre-trained networks implemented by Gkioxari and Malik (2015) is used to extract optical flow features. The f_{v_app} and f_{v_of} are collected from the last Rectified Linear Unit (ReLU) layer from each pre-trained model.

GRU-based Temporal Encoding

Given the spatio-temporal feature sequences f_{v_app} and f_{v_of}, each of them is encoded independently with two variable-length GRUs. The choice of encoding separately is based on the hypodissertation that f_{v_app} and f_{v_of} contain different and/or complementary types of information of the video at different space-time scale and thus they should not be pooled together at the frame level.

As shown in Fig. 2.4, the GRU memorizes a state variable h. The state variable can be either updating or remaining the same, depending on the value of the update gate z. The reset gate r controls the influence of previous input sequence toward the current input.the Gated Recurrent Unit (GRU) uses the reset gate r^n and the update gate z^n (both gates take values between zero and one) to memorize and forget sequence states, thus mitigating the gradient vanishing problem while maintaining the power to discover temporal relationship. Given an input sequence $x = (x_1, ..., x_N)$,

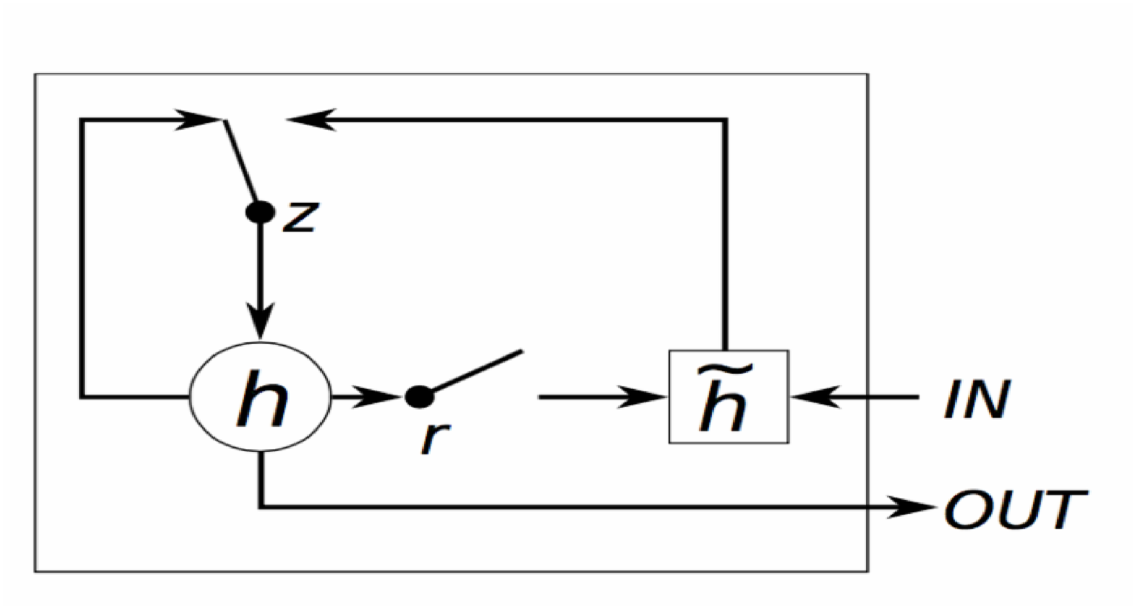


Figure 2.4: A Standard Gated Recurrent Unit Cho *et al.* (2014).

.

the GRU encoder feeds forward the input by iterating the following equation from $n = 1$ to N:

$$r^n = \sigma(W_r x_n + U_r h^{n-1}) \tag{2.9}$$

$$z^n = \sigma(W_z x_n + U_z h^{n-1}) \tag{2.10}$$

$$\tilde{h}^n = tanh(W x_n + U(r^n \bullet h^{n-1}) \tag{2.11}$$

$$h^n = (1 - z^n) \bullet h^{n-1} + z^n \bullet \tilde{h}^n \tag{2.12}$$

where h^n is the model state at Step n, $\tilde{h}^n$ is the proposed state update at Step n, $\sigma()$ denotes the logistic sigmoid function, $\bullet$ denotes element-wise product and W_r, W_z, U_R, U_z denote hidden variables Cho *et al.* (2014).

The GRU architecture utilized in this section is shown in the middle part of Fig. 2.2, where each GRU contains 1024 hidden units. Various numbers of units are experimented and chose 1024 to best trade off complexity and performance. Upon the GRU outputs, a mean-pooling layer is implemented to obtain a fixed-length representation. The outputs of the GRU encoder are the appearance video representation E_v^{app} and the optical flow video representation E_v^{of}, both having 1024 dimensions.

Finally, a simple concatenation is performed to combine the appearance and optical flow representations at the video level: $E_v = [E_v^{app}; E_v^{of}]$. This results in a temporally-encoded representation E_v of 2048 dimensions.

2.2.4 Experiments and Performances

The proposed video representation architecture is evaluated on UCF-101 dataset Soomro *et al.* (2012) (13320 video clips from 101 categories) by conducting three visual tasks: video action recognition, video zero-shot learning, and semantic video retrieval. All three tasks are solved by learning video representations using the same

Table 2.3: 3-Fold Recognition Accuracy on UCF-101 Dataset

Algorithm	**Accuracy (%)**
Dense Trajectory (Wang *et al*) Wang and Schmid (2013)	75.1
LRCN (Donahue *et al*) Donahue *et al.* (2015)	82.9
Composite LSTM model (Srivastava *et al*) Srivastava *et al.* (2015)	84.3
Two-stream Convolution Net (Simonyan *et al*) Simonyan and Zisserman (2014a)	88.0
Deep LSTM with 30 Frame (Ng *et al*) Ng *et al.* (2015)	88.6
TDDs (Wang *et al*) Wang *et al.* (2015)	**91.5**
The Proposed method (softmax)	86.9

architecture. Dense optical flow maps and RGB frames are extracted at 10fps. Each GRU unit contains 1024 hidden units. The MLP has 2048, 1200 and 500 neurons in its input, hidden and output layers respectively. Learning rate for the whole end-to-end structure is initialized as 0.0001 for the first 15 epochs and then reduced by half for every 15 epochs. The batch size is set to 30 videos per batch. The margin for hinge rank loss computation, however, varies between tasks: 0.4 for zero-shot classification and 0.55 for video recognition and semantic video retrieval.

Table 2.4: Zero-Shot Learning Accuracy on UCF-101 Dataset

Algorithm	**Accuracy (%)**
ConSE (Norouzi *et al*) Norouzi *et al.* (2013)	10.5
SES (Xu *et al*) Xu *et al.* (2015a)	10.9
IAP (Lampert *et al*) Lampert *et al.* (2014)	12.8
DAP(Lampert *et al*) Lampert *et al.* (2014)	14.3
The Proposed Method	**14.7**

Video Action Recognition

For video action recognition, the three train-test split rule in Soomro *et al.* (2012) is conducted to train and test the proposed video representation architecture. When testing, test videos are categorized to the trained label that has maximum probability based on the softmax layer output.

The performance of proposed method is compared with Wang and Schmid (2013) and Donahue *et al.* (2015); Srivastava *et al.* (2015); Ng *et al.* (2015); Simonyan and Zisserman (2014a); Wang *et al.* (2015) as shown in Table 2.3. Compared with Simonyan and Zisserman (2014a); Wang *et al.* (2015), the newly video representative features do not require late classifier-based fusion or pooling method. Compared with Ng *et al.* (2015), the new video representation does not train multi-layer LSTMs and thus is more time efficient. It can be assumed that applying late fusion and multi-layer GRUs can potentially lead to further improvements. However, this sub-section mainly focuses on demonstrating that the proposed video representation can handle much high-level task.

Video Zero-Shot Learning

For video zero-shot classification, 10 training and testing experiments are performed. For each experiment, 101 categories are randomly splited into two sub-datasets: training dataset with 51 categories and testing dataset with the rest 50 categories. During the testing stage, test videos are categorized to the "unseen" test label that has maximum cosine similarity (nearest neighbor) to the obtained temporal and semantic embedded representation.

The results of the proposed method are compared with Xu *et al.* (2015a); Norouzi *et al.* (2013); Lampert *et al.* (2014) as shown in Table 2.4. The proposed representation outperforms Xu *et al.* (2015a); Norouzi *et al.* (2013); Lampert *et al.* (2014) by around 2% and 4% and slightly outperforms the state-of-the-art attribute-based model DAP Lampert *et al.* (2014) by around 0.5%. The superior performance of the proposed video representation indicates the effectiveness of the proposed semantic embedding technique that encodes semantics as well as spatio-temporal information.

Semantic Video Retrieval

To further demonstrate that the proposed representation can perform "semantic association" of videos, the dissertation challenges it with the semantic video retrieval task. For this task, the first train-test split rule in Soomro *et al.* (2012) is applied to separate the UCF-101 dataset. A word pool which contains 40 words were created manually to serve as query words. Query words are never seen in training dataset and the retrieve results are the retrieved video clips denoted by the corresponding categories in UCF-101 dataset. When testing, all test videos are fed forward through the trained architecture firstly and obtain the corresponding semantic embedding repre-

Table 2.5: Query Word Pool and the Corresponding Retrieval Results

Query Labels	**Top10 Retrieve Results**	**Query Labels**	**Top10 Retrieve Results**
NBA	Basketball Dunk (10)	Extreme	Rock Climbing Indoor (5), Uneven Bars (2), Soccer Juggling (2), Pole Vault (1)
Orchestra	Playing Cello (9), Playing Piano (1)	Tide	Cliff Diving (4), Surfing (2), Throw Discus (2), Sky Diving (1), Rafting (1)
Army	Military Parade (10)	India	Paying Tabla (4), Playing Sitar (2), Head Massage (1), Cricket Shot (1), Mixing (1)
Music	Playing Sitar (9), Playing Piano (1)	Celebrate	Military Parade (6), Long Jump (1), Band Marching (1), Ice Dancing (1), Blowing Candles (1)
Computer	Typing (10)	Home-run	Baseball Pitch (5), Basketball Dunk (3), Field Hockey Penalty (1), Frisbee Catch (1)
Park	Biking (9), Golf Swing (1)	Boat	Kayaking (4), Rafting (2), Rowing (2), Cliff Diving (1), Push Ups (1)
Summit	Cliff Diving (7), Skiing (2), Rope Climbing (1)	Toy	Yo-yo (4), Nun chucks (4), Pull Ups (1), Juggling Ball (1)
School	Skate Boarding (10)	Snow	Skiing (2), Ice Dancing (2), Cricket Bowling (1), Pole Vault (1), Blowing Candles (1), Blow Dry Hair (1), Rafting (1), Sky Diving (1)
Park	Biking (9), Golf Swing (1)	Acrobatics	Juggling Balls (5), Soccer Juggling (5)
Water	kayaking (10)	Ocean	Cliff Diving (4), Sky Diving (3), Kayaking (2), Rafting (1)
FIFA	Soccer Penalty (8), Soccer Juggling (2)	Hurl	Throw Discus (2), Mopping Floor (2), Baby Crawling (1), Javelin Throw (1), Cricket Shot (1), Blowing Candles (1), Pull Ups (1)
Club	Golf Swing (8), Soccer Juggling (2)	Hiking	Biking (5), Kayaking (4), Rafting (1)
Nature	Tai Chi (7), Hammering (2), Walking with Dog (1)	Swim	Diving (5), kayaking (3), Cricket Bowling (1), Sky Diving (1)
Beethoven	Playing Cello (8), Playing Voilin (2)	Jogging	Biking (5), Skate Boarding (2), Soccer Juggling (1), Skiing (1), Ice Dancing (1)
Classical	Playing Cello (7), Playing Voilin (3)	Foam	Blowing Candles (7), Pull Ups (1), Rope Climbing (1), Juggling Balls (1)
Yankees	Baseball Pitch (10)	Hip-hop	Trampoline Jumping (6), Swing (4)
Duel	Boxing Punching Bag (8), Punch(2)	Scramble	Pull Ups (6), Trampoline Jumping (2), Rope Climbing (1), Cricket Shot (1)
Lifting	Body Weight Squats (4), Rope Climbing (4), Pull Ups (2)	Mat	Rope Climbing (4), Pommel Horse (3), Trampoline Jumping (2), Javelin Throw (1)
Martial	Fencing (3), Archery (3), Boxing Punching Bag (3), Balance Beam (1)	Parachuting	Diving (6), Cricket Bowling (2), Hand Stand Walking (1), Sky Diving (1)
Tumbling	Trampoline Jumping (8), Throw Discus (1), Frisbee Catch (1)	Hunting	Horse Riding (3), Kayaking (3), Nun chucks (3), Frisbee Catch (1)

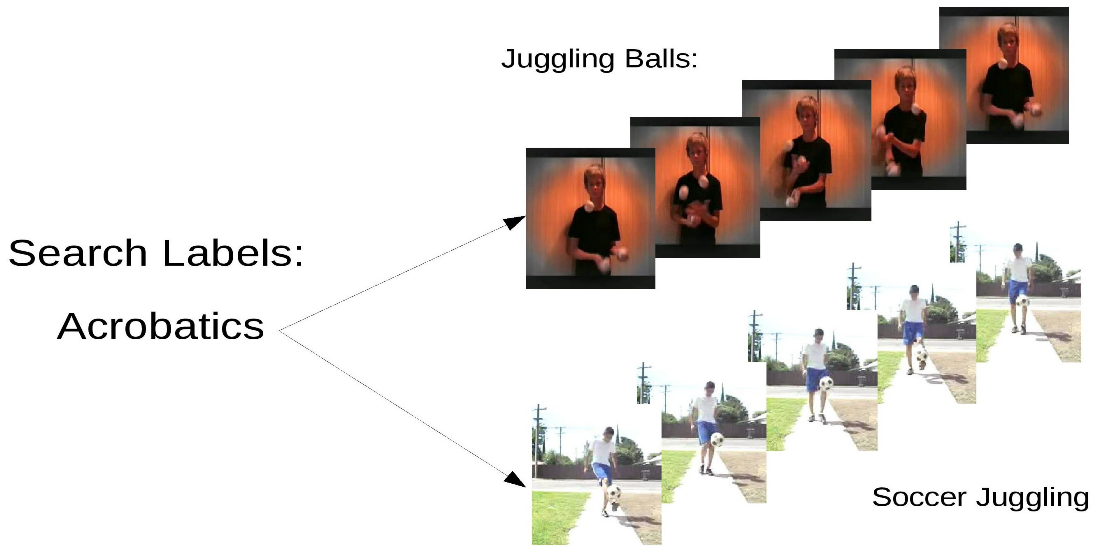


Figure 2.5: Example Result of the Proposed Architecture Performing the Semantic Video Retrieval Task.

sentation. Then, for each query word, the proposed architecture retrieves top 10 test video clips that have maximum cosine similarity (nearest neighbor) to the *word2vec* transformed query word. As an illustration of semantic video retrieval demonstrated in Fig. 2.5, by inputting the query word "Acrobatics" which is not in the original labels pool, the proposed architecture retrieves "Juggling Balls" and "Soccer Juggling" video clips, which are available in the original dataset. Note that, for a query which contains multiple words, the same averaging method as described previously is performed.

The query word pool and the corresponding retrieval results are shown in Table 2.5 and the number in the brackets denoted how many retrieved video clips belong to this category in the top10 hit list. It shows that the proposed model is capable of capturing both visual contents and their semantics. One specific example is the retrieved results using the query word "NBA". All retrieved videos belongs to the

"Basketball Dunk" category in the UCF-101 dataset.

2.2.5 Conclusion

In this section, a two-stream deep structure with two-channel RNN encoders that learns semantic spatio-temporal embbeddings for videos is proposed. Gated Recurrent Units are utilized to temporally encode deep CNN features while an MLP is trained to further embed the learned spatio-temporal features into a semantic space given by *word2vec*. The proposed video representation architecture is evaluated on the UCF-101 dataset for action recognition, zero-shot classification and semantic video retrieval. The experimental results suggest that the proposed approach is able to compute video representations useful and effective for semantic visual analysis tasks. In particular, the semantic video retrieval example demonstrates that the proposed approach can support retrieving semantically meaningful videos simply based on a word query.

2.3 Summary

In this chapter, two deep two-stream structure based models are proposed. The performance on several visual tasks, like zero-shot learning, video action recognition, and semantic video retrieval, demonstrate that the two-stream structure is capable of obtaining the temporal information for deep features based on the optical flow maps. However, the disadvantage of the two-stream structure is obvious as well. The simple fusion or combination of two feature vectors which are obtained from two channels separately will mix the information acquired from the deep neural network, since the feature vector is highly condensed from sequences of frames to a 4k dimension vector. Therefore, the following chapter will introduce a deep model which will obtain feature maps instead of feature vectors.

Chapter 3

INCORPORATION OF CNN AND RNN FOR SURVEILLANCE VIDEO REPRESENTATIVE FEATURE EXTRACTION

As described in **Chapter 1**, with the rapid growth of different video data, the demands of obtaining representative features for more complex video clips increase exponentially. One kind of the complex video data based on the surveillance system draws more and more attention in recent years since those surveillance video data has close relationship with people daily life. In surveillance video analysis, action/event localization and recognition are two critical capabilities, which have been largely addressed separately in the literature. In this section, an approach is proposed to simultaneously localize and recognize visual events from raw surveillance videos, employing an end-to-end learning strategy. The proposed approach formulates the task as weakly-supervised sequential semantic segmentation, in which a specific convolutional RNN is utilized to capture not only the appearance and the motion information but also their temporal evolution patterns. The proposed approach is tested on the VIRAT 2.0 dataset. The experimental results, in comparison with relevant existing state-of-the-art, suggest that the proposed approach is promising in extracting video representative features and delivering a practical solution.

3.1 Introduction

Video-based surveillance systems have become a key technology in maintaining law and order. Current technological and political trends are accelerating deployment of video-based surveillance systems in our society. With ever-increasing amount of video data generated by such systems, automatic capabilities for processing and analyzing surveillance videos have become imperative. Among others, key automatic capabilities include the detection of the occurrence of any event/action of interest, including both where and when an event/action happened (spatio-temporal localization) and what has happened (event/action recognition). As such, action and event recognition in surveillance videos has received significant attention in recent years.

As describes in **Chapter 1**, unlike static images, a sequence of video frames captures the interactions among objects along the temporal axis. The temporal attributes of such interactions may be the defining feature of the underlying actions, given that the appearance of the objects/scene can vary dramatically. For example, the event "a person opening the trunk of a sedan" may involve a variety of cars of different makes/models/colors, with the person having different appearances as well, but the sequence of movements of the person interacting with the rear end of a sedan is typically distinctive. In this sense, the key of recognizing events lies not only on correctly detecting the objects, but also on identifying specific temporal interaction patterns. Following such intuitions, different from some works mainly focusing on appearance of frames Chang *et al.* (2017); Tran *et al.* (2015), a more effective line of works was proposed seeking to incorporate interactions via motion information Simonyan and Zisserman (2014a); Feichtenhofer *et al.* (2016); Wang *et al.* (2016b), which is introduced in the **Chapter 3** as the deep two-stream framework. A typical two-stream recognition framework considers both appearance and motion information, and categorizes a video clip into a specific class. This procedure amounts to answering "what" happens in a video.

However, in raw surveillance videos, an event may occur in only a short period of time and/or within a small local region of the visual field of view, and thus the aforementioned methods became less effective as the task is no longer to simply classify a short video clip capturing an underlying action/event. To handle such situations, a parallel series of algorithms are devised to localize events spatially Tran *et al.* (2014); Ke *et al.* (2005) or temporally Tang *et al.* (2012); Yeung *et al.* (2016). The spatial and temporal "localizers" attempt to answer the question "where" and "when" an event happens in a long video, respectively. It is worth noting that, in many situations (like many simultaneous actions/events occurring in a video), the task of event localization may inevitably be dependent of event recognition. Nevertheless, this task has not be fully addressed, despite of its practical importance.

In this chapter, a novel and unified framework that can simultaneously localize (both spatially and temporally) and recognize events in a long surveillance video is proposesd. Given an input video clip, the proposed model seeks to answer "when, where and what happens" simultaneously, through an end-to-end learning and inference framework without requiring explicit preprocessing steps like separate object detection and motion-based segmentation. To the best of current knowledge, this is the first attempt to address such tasks in surveillance video using an end-to-end learning approach. To achieve this, an inspiration is drawn from 2D semantic segmentation and consider the task as one of sequential weakly-supervised semantic segmentation over a sequence of frames. The ground-truth spatial-temporal bounding boxes is treated as weak labeling and an end-to-end model to localize and recognize events is trained correspondingly. The proposed approach employs a convolutional RNN structure (ConvRNN) Xingjian *et al.* (2015) to capture both the order information of frames and the spatio-temporal features. The proposed model can handle arbitrary video length without the need of clipping out the event region, and thus is suitable for real-world applications. The approach is evaluated by using the VIRAT dataset and obtained promising results.

3.2 Related Work

As introduced in the **Chapter 3**, the two-stream or multi-stream deep neural networks have been used to obtain state-of-the-art performance in a few related tasks like video classification Karpathy *et al.* (2014); Wang and Ji (2017); Simonyan and Zisserman (2014a); Wang *et al.* (2016b) and action recognition Wang *et al.* (2016a); Hu *et al.* (2016). Most multi-stream deep neural networks involve fully connected layers to fuse and extract effective representation (a feature vector) Wang and Ji (2017); Wang *et al.* (2016a); Simonyan and Zisserman (2014a). Such features are widely utilized in a large variety of video-based task (in particular, video-based event localization and recognition).

Even though event localization and recognition in surveillance video share some common sub-tasks like extracting spatial-temporal features, these two tasks are distinguished from each other. In general, event localization focuses on locating an event within a frame (spatial) Tran *et al.* (2014); Ke *et al.* (2005) and/or on the time axis (temporal) Xu *et al.* (2015b); Tang *et al.* (2012); Lai *et al.* (2014); Yeung *et al.* (2016). Event recognition, on the other hand, can be viewed as a classification problem, where higher-level semantic features are often employed Wang and Ji (2017); Piergiovanni and Ryoo (2018); Fernando *et al.* (2015).

Most localization algorithms seek to localize some given events. The authors of Tran *et al.* (2014) proposed an algorithm to find the optimal spatial-temporal path along with a 3D event detection volume by using a pre-detected score. Lai *et al.* (2014) segments the video into multiple instances and learns to predict on instance with only video-level label. Recently, reinforcement learning is also applied to predict the beginning and ending time of an event Yeung *et al.* (2016). On the other hand, typical event recognition algorithms need to know the approximate spatial and temporal position of the events. The authors of Wang and Ji (2017) utilized low-level STIP Laptev (2005) and GIST features Oliva and Torralba (2001) to represent video context, which helped to improve recognition. However, the training and testing sets are cropped from the raw videos. Fernando *et al.* (2015) devised a ranking scheme to represent the video with more robust features, but the event still has to occupy most of the frames spatially.

Some efforts have been devoted to solve segmentation in 3D volume by taking into account the temporal information Siam *et al.* (2017); Wang *et al.* (2016a); Fayyaz *et al.* (2016) which is similar to the proposed method. The main difference is that, while the aforementioned methods only focus on object segmentation, the proposed model seeks to find the specific interaction patterns between objects. Additionally, the proposed method does not need the training samples to be pixel-level, which alleviates the burden of training data acquisition. It should also be noted that, the proposed approach does not contain any fully connected layers and only utilizes the feature maps instead of feature vector. This procedure has its counterpart in 2D semantic segmentation Badrinarayanan *et al.* (2017); Long *et al.* (2015), where feature maps proved sufficient to capture the context information and are suitable for the following manipulations like deconvolution and unpooling. The proposed model takes advantage of both the multi-stream neural network structure and fully convolutional neural networks to fulfill the event localization and recognition task simultaneously in a unified learning framework.

Compared with existing works separately focusing on event localization and recognition, the proposed framework only utilizes the multi-class classification loss as the objective function and predict heatmaps for different event labels. The proposed method leverages the channel of optical flow maps, which go through a processing pipeline similar to the appearance channel, so that the model can potentially learn higher-order motion patterns and their interactions. Thus the proposed two-stream convolutional RNN based deep neural network is well-suited for addressing simultaneous event localization and recognition.

3.3 Methodology

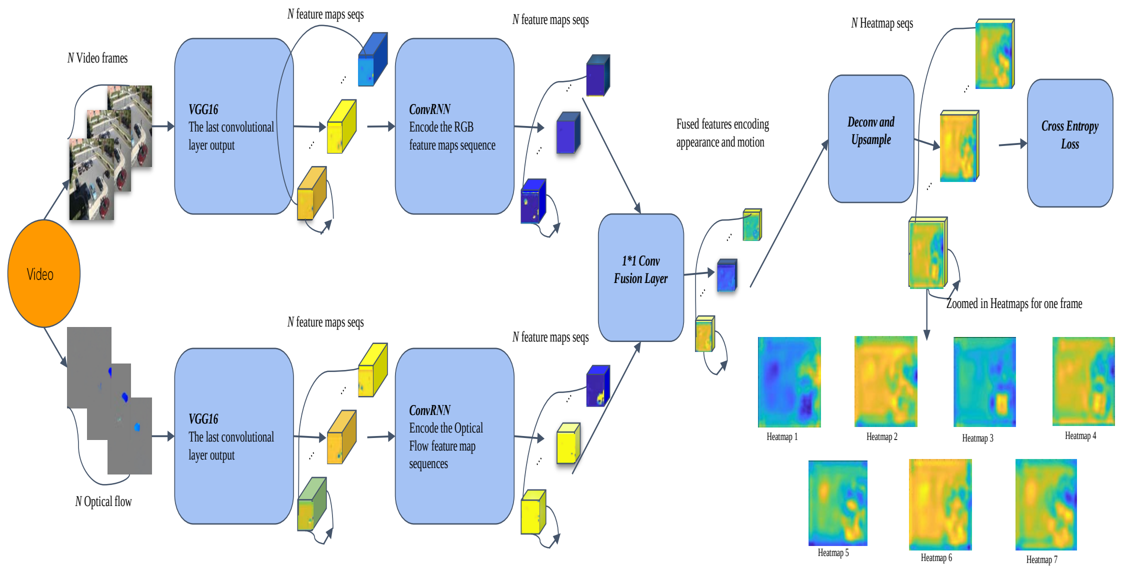


Figure 3.1: Overview Structure of the Proposed Framework.

3.3.1 Overview of the Proposed Approach

An overall structure of the proposed model can be found in Fig. 3.1. The illustration is based on a subset of the VIRAT dataset which contains 6 events from parking lot surveillance videos. The 7 heatmaps correspond to the confidence map of each event in addition to an extra “background” event. Assume that a video sequence consists of frame $\{X_i\}$, where $i \in \{1, ..., N\}$ is the index of the frame. The corresponding optical flow map is $\{Y_i\}$. The main flow of the proposed algorithm starts from taking as input the video frames and optical flow maps, and then feeding them into two

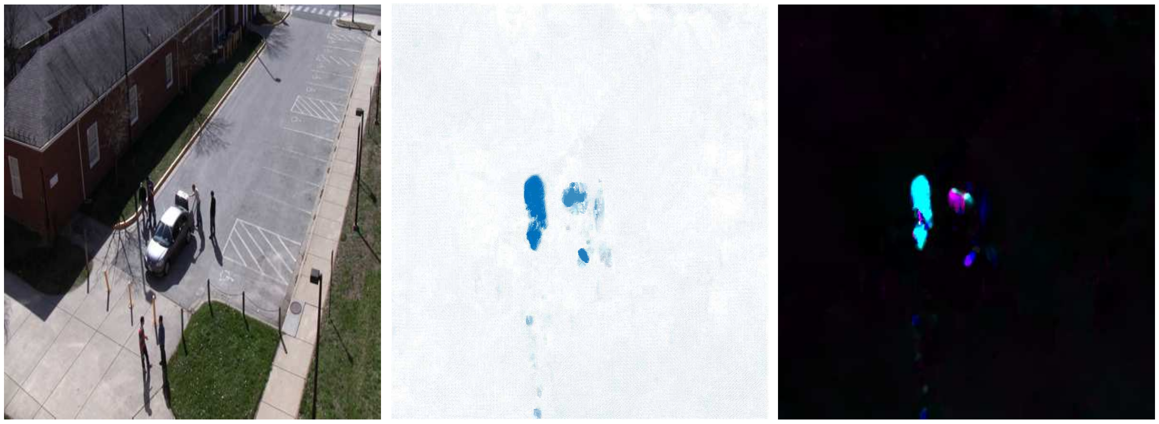


Figure 3.2: Optical Flow Map as an Image.

separate pretrained VGG16 networks Simonyan and Zisserman (2014b). The output features of VGGs are then fed into two ConvRNNs to capture the temporal patterns. The ConvRNNs can output patterns with the identical dimension as the input feature map. With such a property, the patterns can be decoded into heatmaps in a semantic segmentation fashion by using deconvolution and upsampling layers, where each channel of the heatmaps corresponds to the confidence falling into a specific event class.

3.3.2 Appearance and Motion Features

It is natural to extract appearance features from frames using pre-trained CNN architecture (i.e. VGG16). The consideration behind employing VGG to deal with optical flow is explained in the following. The two channels of optical flow correspond to the horizontal and the vertical movements. In order to convert the flow to optical flow map, the map can be viewed as a two-channel flow vector field and an extra channel of the flow magnitude. An example can be found in Fig. 3.2. In the figure, the left one is the original frame from a video clip. The middle one depicts the optical flow in vector field. The right one is the optical flow image obtained by adding an additional magnitude channel. It can be observed that, human can easily identify the motion pattern from such flow images. In the proposed approach, a CNN structure is employed to capture and extract motion information from such inputs. It is possible to employ pre-trained VGG, as the motion fields from a real surveillance video depicts properties like smoothness, similar to a natural image. In the setting of the proposed method, the last output before the first fully-connected layer of VGG is taken as appearance and motion features.

3.3.3 Recurrent Convolutional Neural Network

As the joint localization and recognition is considered as a 3D sequential semantic segmentation problem, the dimension of the extracted temporal feature after feeding the RNN structure should be consistent, yielding the requirement for the following deconvolution and upsampling procedures. To this end, a specific type of incorporation of CNN and RNN called ConvRNN is employed Xingjian *et al.* (2015) to calculate the temporal features. Suppose the feature map at time t is f_t, a ConvRNN in LSTM fashion is calculated as:

$$r_t = \sigma(W_{fr} \otimes f_t + W_{hr} \otimes h_{t-1} + W_{cr} \circ c_{t-1} + b_r) \tag{3.1}$$

$$z_t = \sigma(W_{fz} \otimes f_t + W_{hz} \otimes h_{t-1} + W_{cz} \circ c_{t-1} + b_z) \tag{3.2}$$

$$c_t = z_t \circ c_{t-1} + r_t \circ \tanh(W_{fc} \otimes f_t + W_{hc} \otimes h_{t-1} + b_c) \tag{3.3}$$

$$o_t = \sigma(W_{fo} \otimes f_t + W_{ho} \otimes h_{t-1} + W_{co} \circ c_t + b_o) \tag{3.4}$$

$$h_t = o_t \circ \tanh(c_t) \tag{3.5}$$

where $\otimes$ and $\circ$ denote the convolution operator and Hadamard's product, respectively. σ is the activation function and W is the kernel. A variant of ConvRNN is to employ a Gated Reccurent Unit (GRU) Chung *et al.* (2014). The mathematical details is described here. Intuitively, for each position in the feature map at time t, the corresponding output is affected by a neighborhood around the same position from time $t-1$. When the motion at this position is faster, a kernel with a larger size can be utilized. Conversely, the smaller size of kernel size is sufficient to capture the slow motion. This is a reasonable setting since the speed of common objects in surveillance videos should have a limit. An advantage of ConvRNN structure is that the number of the parameters can be significantly reduced as only convolutional kernels are involved. Thus the training of the proposed model can be efficient. ConvRNN can be viewed as a generalized version of traditional RNN, since if the kernel size is extended to the feature map size, the layer involving W becomes fully connected.

3.3.4 Feature Fusion

In the feature fusion layer, the outputs of two ConvRNNs are fused into one to be a suitable feature shape for the following manipulations. To this end, one 1×1 fully-convolutional layer is conducted. For the output of the ConvRNNs $c^{(I)} \in \mathbb{R}^{m \times m \times p}$ and $c^{(O)} \in \mathbb{R}^{m \times m \times p}$ ($c^{(I)}$ and $c^{(O)}$ are the feature maps corresponding to the appearance and optical flow channels, respectively), firstly they are concatenated into one feature map $c^{(F)} \in \mathbb{R}^{m \times m \times 2p}$, then an 1×1 kernel is applied on it. All 1×1 kernels are learnable during the training phase, and serve as the function of "feature fusion" to find the optimal way of combining $2p$ features.

3.3.5 Deconvolution and Upsampling

The output of aforementioned fusion layer at time t is c_t, which is with the identical first two dimensions as input feature map $c^{(I)}$ and $c^{(O)}$ (typically this is a 3D matrix). For algorithms handling the task of 2D semantic segmentation Badrinarayanan *et al.* (2017); Long *et al.* (2015), a general framework is an encoder consisting of convolutional and pooling layers followed by a decoder with deconvolutional and unpooling layers. Considering the dimensional benefit, the output c_t can also be decoded into a heatmap, of which the pixel level label reveals the event category. This is based on the intuition that the ConvRNN output c_t encodes both the frame-level information (from CNNs) and temporal information (from RNNs). Assuming there are k different events in total, another knid of background event $k+1$ is further added corresponding to the class of no event happening. Thus the output of the semantic decoder would be with size $a \times b \times (k+1)$, where $a \times b$ is the dimension of the frame, in other words, the size of each heatmap of the output is the same as the each input size. To accelerate the calculation, the unpooling procedure is replaced with a simple bilinear upsampling strategy. A basic decoding unit with a deconvlution followed by a bilinear sampling is constructed. In the setting of the proposed algorithm, there are 4 such units concatenated in the decoder. Denote the output heatmap $H_t \in \mathbb{R}^{a \times b \times (k+1)}$.

3.3.6 Loss Function and Training

In the training phase, the cross-entropy is utilized to evaluate the loss between the calculated heatmap H_t and the ground-truth label. In the setting of this proposed method, the ground-truth labels are 3D bounding boxes lying in the frame volume. While the projection of a box on the temporal axis defines an interval showing when an event starts and ends, the projection on the frames corresponds to the spatial location. Then the elements within the 3D bounding box are made to be the corresponding label value. For any elements outside the bounding box, they are assigned the value $k+1$, which is the "background" containing no meaningful events. Denote such volume with ground-truth label $\hat{H}$. Then the loss function becomes:

$$\mathbb{L} = \sum_{a,b,t} H_{a,b,t} \log(\frac{1}{\hat{H}_{a,b,t}}) + \lambda Z^2 \tag{3.6}$$

where Z is the parameter of the proposed model. Note that, unlike object detection which is a regression in terms of bounding boxes, the proposed algorithm is a pixel-wise classification problem. The bounding box is treated as weakly supervised label, since it is difficult to define the exact boundary of an event and its constituent objects, and the bounding box only gives a coarse approximation.

3.3.7 Prediction and Testing

When processing a new video, given heatmap H_t, a soft-max function is applied to obtain the confidence $L_{t;abl}$ for event l at each pixel position (a, b):

$$L_{t;abl} = \frac{\exp(H_{t;a,b,l})}{\sum_{j=1}^{k+1} \exp(H_{t;a,b,j})} \tag{3.7}$$

Then the dominant label is the one with the maximal confident $L_{t;ab} = \max_l L_{t;abl}$ at position (a, b). Thus the prediction at time t is L_t. By stacking all the time point t, a 3D matrix which encodes the event label and the (spatial and temporal) location information can be obtained.

3.4 Experiments and Results

3.4.1 Dataset and Evaluation Criteria

All the experiments were based on the **VIRAT 2.0** dataset which consists of over 200 video clips summing up to 8 hours. Only 6 events with human-vehicle interactions captured from parking lots are considered in this dissertation: *Loading a Vehicle* (LAV), *Unloading a Vehicle* (UAV), *Opening a Trunk* (OAT), *Closing a Trunk* (CAT), *Getting into a Vehicle* (GIV) and *Getting out of a Vehicle* (GOV).

All videos are downsampled to a smaller resolution 224×224. Then the events are temporally cropped and extended by 20 frames on both directions and downsampled the frame rate to 5 FPS.

Considering the difference with traditional methods, the following three different criteria are defined to evaluate.

1) For the event recognition task, the proposed model can predict pixel-wise label for each frame. Therefore a majority voting process is defined to obtain the video-level label for each sample. First, the dominant pixel label are found, which is the most among all the predicted labels, as the frame-level label. Second, the number of frame-level labels of different events are counted and the most frame-level label are found as the video-level label. Last, the new video-level labels are utilized to compute the event recognition accuracy.

2) For the spatial localization task, since the proposed algorithm does not generate bounding boxes, a simple evaluation metric called percentage of Hit frames is developed. One frame can be called a "Hit" if the predicted high-confidence region is within the ground truth bounding box. Then the percentage of Hit frames in an event is counted. The average Hit percentage of the dataset can represent the performance of the spatial localization task.

3) For the temporal localization task, the capability of the proposed model in predicting the starting and ending frames within each video clip should be evaluated and the missing prediction during each event would be found as well. Therefore, this kind of evaluation is treated as the retrieval problem where precision and recall are key factors to evaluate the performance of the proposed algorithm. F measure (or F score) which combine the information of precision and recall is utilized to evaluate tthe proposed model's performance on temporal localization.

3.4.2 Implementation Details

As shown in Fig. 3.1, the pre-trained VGG16 network Simonyan and Zisserman (2014b) is utilized to extract the feature maps from the last convolutional layer to represent both RGB and Optical Flow maps. The number of feature maps for each frame is 512. The output dimension of ConvRNNs is set to 128 and the size of 1×1 convolutional fusion layer is $1 \times 1 \times 128$. There are four deconvolution and upsampling layers in the proposed model. The number of output channels of each deconvolutional layer is $\{128, 64, 32, 7\}$ and the last 7 represents the number of events + background. The bilinear upsampling layer is used after each deconvolutional layer to enlarge the output size as twice as its original size.

For the training stage, the learning rate is initialized as 10^{-4} and the Adam algorithm is utilized to update weights. The total number of epochs is set to 600 and the learning rate drop by a factor of 0.5 if the training loss does not change for 20 epochs. The coding of the system was based on the PyTorch Deep Learning Toolbox [1] and the experiments were run on a server with one single Tesla K40C GPU and 160G RAM.

In the experiments, three different structures in terms of feature fusion are implemented. A model merely based on RGB frames and two-stream structures with a simple feature fusion method, which simply connects two streams of feature maps along the channel axis instead of utilizing the fusion 1×1 convolutional layer is built for comparison. The ConvRNN is further equipped with LSTM and GRU. And Wang and Ji (2017) is selected for comparison.

[1]https://pytorch.org/

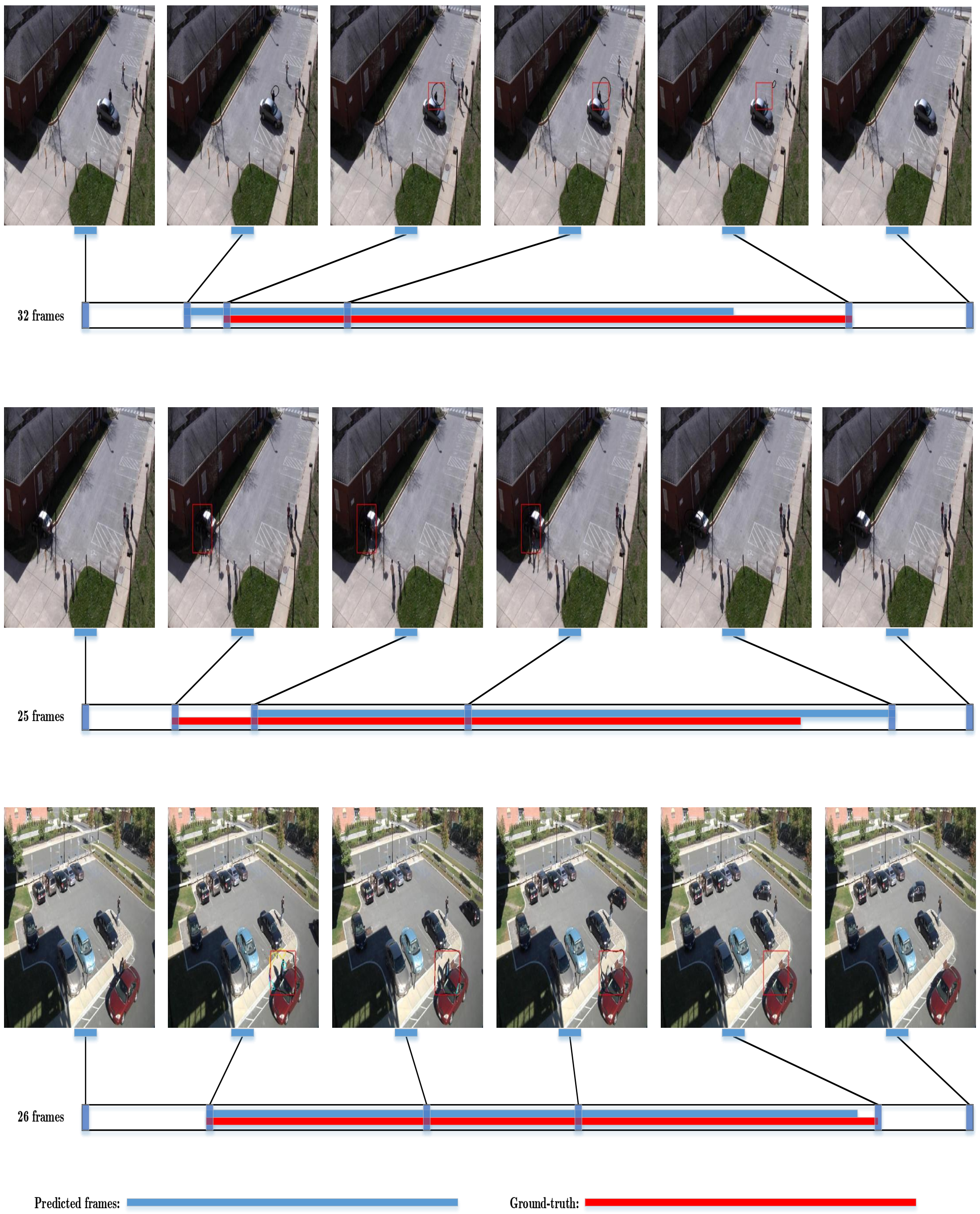


Figure 3.3: Event Localization and Recognition Results on VIRAT 2.0 Dataset.

Table 3.1: Performance Comparison on VIRAT dataset.

Task	Tmp Loc	Spt Loc	Recog
224 one-stream **LSTM**	0.6815/ 0.7670	69.06% / 80.97%	45.18% / 69.80%
224 two-stream **LSTM**	**0.7647**/ **0.7810**	**80.97**% / **83.19**%	**72.73**% / 73.71%
224 two-stream **LSTM** no 1×1	0.7511 0.7721	79.98% / 82.19%	70.68% / **74.95**%
224 two-stream **GRU**	0.7517/ 0.7751	80.28% / 80.53%	66.34% / 70.34%
Hierarchical ContextWang and Ji (2017)	–	–	66.45%/–

3.4.3 Results

The quantitative results based on the evaluation metric are demonstrated in Table 3.1. In the table, The left and the right scores report the performance on testing and training samples, respectively. The best results are in bold. The event recognition task is compared with Wang and Ji (2017) since they reported state-of-the-art performance on the VIRAT dataset with the same 6 events. Though the metric is different, the proposed method is evaluated more stringently since Wang and Ji (2017) assumes a segmented video (with event bounding boxes given) while the proposed method deals with a raw video. Hence the results suggest that the proposed model performs competitively. Lacking a suitable baseline in the literature for both spatial and temporal segmentation for events in surveillance videos, the evaluation of these tasks were only based different settings of the proposed model, as reported in the table. Sample results of temporal and spatial localization in different scenes are shown in Fig. 3.3. Overall, it is fair to state that the proposed model was able to deliver state-of-the-art performance on event recognition while predicting the temporal and spatial locations of the events with reasonable accuracy in surveillance video data.

3.5 Conclusion

An approach is proposed to fulfil simultaneous event localization and recognition tasks in surveillance video, employing an end-to-end learning framework. The learning process is only weakly supervised in that the supervision comes from merely the event bounding boxes (e.g., no requirement for object IDs or elaborate object contours). In the testing/evaluation stage, there is no demands for preprocessing steps like object detection or motion segmentation, which are commonly required in many existing approaches. Hence the proposed approach has the potential of providing a more practical solution. Experimental results demonstrated that the proposed method was able to deliver superior performance in comparison with a few competitions, which supports the claim in the beginning of this chapter.

Moreover, the proposed algorithm presents the capability of capturing spatial and temporal information simultaneously which should be more practical in understanding and analysis of the people's daily life video data. The fusion layer, which utilizes the 1×1 convolutional layer to fuse the feature maps from two channels, demonstrates the effectiveness and usefulness compared with simple fusion method described in **Chapter 3**.

Chapter 4

A NEWLY PROPOSED RNN STRUCTURE FOR VERY LONG AND COMPLEX VIDEO REPRESENTATIVE FEATURE EXTRACTION

As introduced in above-mentioned chapters, many successful approaches have been proposed for recognizing events in short and homogeneous videos, but doing so with long and complex videos remains a challenge. One particular reason is that events in long and complex videos can consist of multiple heterogeneous sub-activities (in terms of rhythms, activity variants, composition order, etc.) within quite a long period. This fact brings about two main difficulties: the excessive and varying length and the complex dynamic and rhythm of video clips. To address this, a new RNN structure called Rhythmic RNN (RhyRNN) is proposed. This newly proposed RhyRNN is capable of handling long video sequences (up to 3,000 frames) as well as capturing rhythms at different scales. Two novel modules: diversity-driven pooling (DivPool) and bilinear reweighting (BR), which consistently and hierarchically abstract higher-level information, are proposed as well. The behavior and the performance of RhyRNN is studied and shown empirically to present the proposed method can work well even when *only event-level labels are available* in the training stage (compared to algorithms requiring sub-activity labels for recognition), and thus is more practical when the sub-activity labels are missing or difficult to obtain. Extensive experiments on several public datasets demonstrate that, even *without fine-tuning the feature backbones*, the proposed method can achieve promising performance for long and complex videos that contain multiple sub-activities.

4.1 Introduction

As mentioned in the above-mentioned chapter, the video-based event/activity recognition has brought about enormous and important challenges to computer vision in recent years. The research community has devoted considerable effort and made progresses in many related tasks (e.g., action recognition Feichtenhofer *et al.* (2016); Simonyan and Zisserman (2014a); Veeriah *et al.* (2015); Wang *et al.* (2015); Donahue *et al.* (2015); Wang *et al.* (2016a); Bilen *et al.* (2016); Lan *et al.* (2015); Du *et al.* (2015), temporal localization Shou *et al.* (2016); Chao *et al.* (2018), video question answering Tapaswi *et al.* (2016); Antol *et al.* (2015), video summarization Gong *et al.* (2014); Lee *et al.* (2012); Zhang *et al.* (2016), to name a few). By learning more representative features and capturing stronger sequential context, deep-learning-based approaches have delivered the state-of-the-art results on several datasets of short video clips (e.g. UCF101 Soomro *et al.* (2012), KTH Schuldt *et al.* (2004), HMDB51 Kuehne *et al.* (2011)). Recently, more challenging datasets (e.g. VIRAT Oh *et al.* (2011), Charades Sigurdsson *et al.* (2016) and Breakfast Kuehne *et al.* (2014)), which typically contain video clips with complex and/or multiple sub-activities in a much longer time period, have brought about new challenges to video event recognition. To address these, some event recognition algorithms were proposed Duan *et al.* (2012); Tran and Davis (2008); Jiang *et al.* (2013); Wang and Ji (2017); Xu *et al.* (2015b); Hussein *et al.* (2019a); Wu *et al.* (2019); Piergiovanni and Ryoo (2018); Feichtenhofer *et al.* (2019); Xu *et al.* (2019); Fernando *et al.* (2020), taking into account either long-time dependency or the activity variation to some extent. In this chapter, a specific RNN structure is investigated to understand long and complex videos.

For clarity of discussion, a distinction is made between activity and event. Consider one example for each. "Jogging", which belongs to activity in the defined context, exhibits relatively fixed or homogeneous visual pattern and temporal dynamic (repetitive motion in this case). In contrast, "Cooking spaghetti", which is categorized as an event, is composed of multiple sub-activities (e.g., "bringing out condiment", "boiling spaghetti", etc.) that can occur in different rhythms, orders or visual appearances, resulting in much more complex scene dynamics for an algorithm to capture. Furthermore, some events can occur over a significantly longer time period than activities. In general, events in long videos brings about two challenges to video-based recognition: complexity in content and excessive and varying length, making it challenging to adapt a traditional activity recognition model designed for much simpler videos.

Another important yet barely investigated issue in video-based recognition is, how to identify video events when *only event-level labels are available* for training a model. This arises frequently due to lack of detailed labeling information that is difficult and/or costly to obtain for long videos. Though some previous methods incorporate sub-activity labels to enhance event-level recognition Kuehne *et al.* (2014, 2016); Hussein *et al.* (2019a,b), such fine-grained labels are not always available in practice because of the aforementioned reason. In general, the event label describing a long video is highly abstract in nature, and it may imply a lot of latent contexts.

In this chapter, a progress towards long and complex video event recognition (with or without sub-activity labels) is made by conducting RNN based structure. And a way to perform video-based recognition when only event-level labels are available is further studied. To this end, the Rhythmic RNN (**RhyRNN**) which dynamically captures the multi-level contexts, as well as a diversity-driven sequential pooling (**DivPool**) and a Bilinear Re-weighting (**BR**) mechanism is proposed. This work has the following contributions: 1) the RhyRNN is newly introduced which can ease the gradient back-propagation for long and complex sequences. Thee RhyRNN also allows to capture latent video context at different levels; 2) The DivPool and BR strategies are developed in this chapter, which further enable multi-level feature aggregation (analogous to pooling in CNNs) with varying sequence length; 3) The property and behavior of all the proposed modules are studied analytically and empirically; 4) The proposed method delivered superior or competitive performance in long video datasets compared to the state-of-the-art algorithms even without fine-tuning feature backbones.

4.2 Related Work

Short activity recognition

Some early video datasets (e.g., KTH Schuldt *et al.* (2004) and UCF101 Soomro *et al.* (2012)) typically contain activity/action-level video clips, which are homogeneous in content without too complex temporal dynamics. A conventional trial for activity recognition employed 2D CNN features to perform recognition Karpathy *et al.* (2014), while some variants incorporate complementary frame-level motion features Simonyan and Zisserman (2014a); Bilen *et al.* (2016, 2017). The main drawback of such a line of works is that the temporal patterns cannot be well learned since neither short nor long range dependencies are explicitly taken into account. 3D CNNs are natural extension from 2D by introducing one additional kernel dimension on the time axis Tran *et al.* (2015); Carreira and Zisserman (2017); Wang *et al.* (2018), but with excessive parameters. To alleviate this, several works were proposed to decouple the 3D kernel into combinations of lower dimension (e.g., Chollet (2017); Tran *et al.* (2018); Xie *et al.* (2018)). Another line of works in parallel to CNNs employs RNNs Du *et al.* (2015); Lan *et al.* (2015); Sharma *et al.* (2015); Donahue *et al.* (2015). RNNs can handle varying length of videos compared with CNNs, but suffer from gradient vanishing/explosion issue especially when the sequence is too long.

Complex event recognition

Datasets consisting of long and complex videos bring about new challenges Oh *et al.* (2011); Kuehne *et al.* (2014); Sigurdsson *et al.* (2016). Extending CNNs for long-range video recognition has become an aroused research interest recently. To capture more complex temporal patterns in long videos, Sigurdsson *et al.* (2017) stacks a CRF on top of CNN output. Under some specific sampling procedure, TSN Wang *et al.* (2016a) and TRN Zhou *et al.* (2018) model the video-level representation by considering inter and intra video relations, respectively. Non-local networks Wang *et al.* (2018) built upon 3D CNN can range up to 128 time steps, hence is capable of handling more complex dynamics. Timeception Hussein *et al.* (2019a) can further capture the dependencies up to 1024 frames by designing multi-scale convolutional kernels. In parallel to CNNs, RNNs are also investigated to tackle long and varying video length with complex context. Yeung *et al.* (2018) considers dense labeling in complex videos, where the expensive part is to densely label the training data. Wang *et al.* (2016c) proposed a hierarchical RNN to capture temporal visual attention. Both Lan *et al.* (2015) and Du *et al.* (2015) devise hierarchical RNN structures to obtain multi-level representation, which proved effective in understanding video content. In Sharma *et al.* (2015), soft attention is computed spatially and temporally via deep RNNs, which helps the model to focus selectively on more meaningful parts of a video.

RNNs

LSTM Hochreiter and Schmidhuber (1997) and GRU Cho *et al.* (2014) are successively proposed to address the gradient vanishing/exploding issue by introducing the gating mechanism against standard RNNs. There is a series of further developments following this strategy Gers *et al.* (1999); Campos *et al.* (2017); Kanuparthi *et al.* (2019). Skip-RNN Campos *et al.* (2017) learns to keep the hidden state intact at some steps once “Skip” is emitted. H-detach Kanuparthi *et al.* (2019) detaches the gradient flow of LSTM structure at an arbitrary time step under a Bernoulli distribution. Some other efforts focused on the variants of standard RNNs without using gating. Multiplicative Integration Wu *et al.* (2016) couples the operations on inputs and hidden states. In this fashion, the vanishing gradient is likely to be correlated by the input sequence. Unitary-RNN Arjovsky *et al.* (2016) allows smoother gradient flow by constraining RNNs to have a unitary transition matrix. Very recently, IndRNN Li *et al.* (2018) was proposed, which enforces the neurons in each RNN unit to be independent. By doing so, IndRNN can handle long sequences and achieved state-of-the-art performance on multiple benchmarks.

4.3 Methodology

4.3.1 Algorithm Overview

The overview framework of the proposed method consisting of three modules is illustrated in Fig. 4.1, RhyRNN, DivPool and BR (a recognition module). BR refers to the GRU equipped with bilinear re-weighting in the setting of the proposed method. The model takes visual features as input and feeds them sequentially to RhyRNN. RhyRNN outputs embedded features with the same length as the sequence. Using a diversity score, DivPool is then applied to select the most informative features as inputs to the following recognition module. For the final recognition stage, a GRU equipped with BR module is employed. The output of GRU at the final timestamp will be fed to a two-layer fully connected network at last. Each part will be described in details in the following sections. The proposed approach is motivated by the following considerations. First, the proposed approach should be capable of handling long sequences. To this end, it is necessary to design a specific RNN structure which eases the gradient flow under this setting. Second, since complex events contain latent

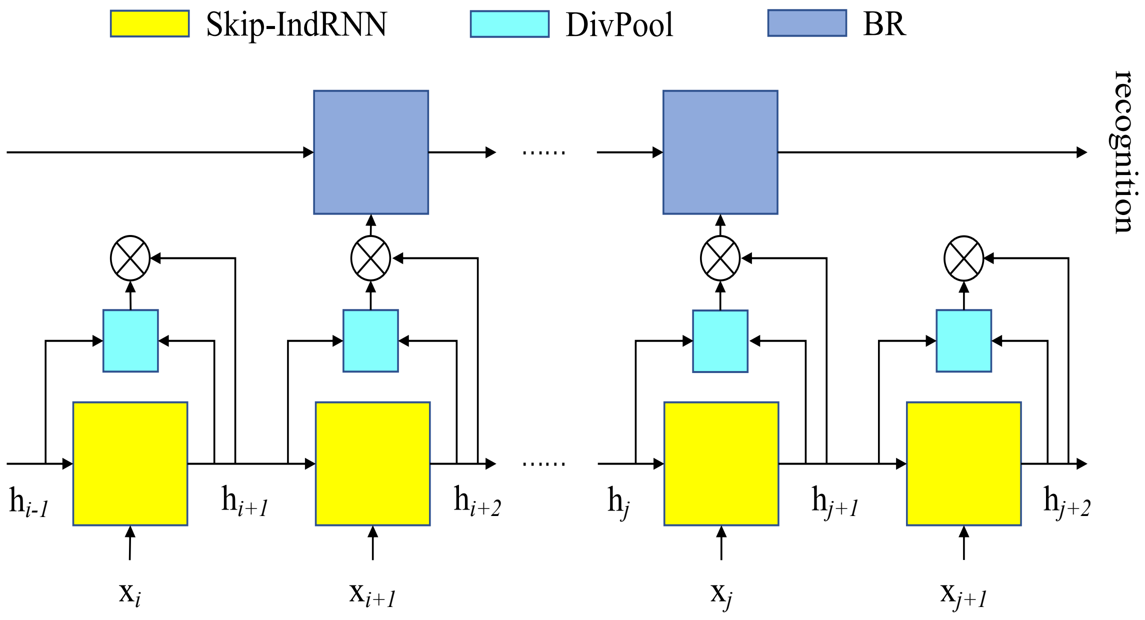


Figure 4.1: Overview Framework of the Proposed Approach.

contexts in different scales, the proposed approach needs to capture such multi-level dynamics. A hierarchical model, in this case, can be a good choice as done in a large body of relevant literature.

4.3.2 Rhythmic Recurrent Neural Network

One essential part of the proposed algorithm is to deploy an architecture that is capable of handling a long and complex sequence. In this chapter, the Rhythmic Recurrent Neural Network, RhyRNN structure is proposed, which is inspired in part by IndRNN Li *et al.* (2018) and Skip-RNN Campos *et al.* (2017), and is much more powerful than both (see Sec. 4.4). IndRNN enforces each neuron operating on the hidden state to be independent, and the update rule of IndRNN reads:

$$\mathbf{h}_t = \sigma\left(\mathbf{W}\mathbf{x}_t + \mathbf{u} \odot \mathbf{h}_{t-1} + \mathbf{b}\right) \tag{4.1}$$

where $\odot$ is the element-wise product and $\sigma(\cdot)$ is the activation function (ReLU function in Li *et al.* (2018)). $\mathbf{h}_t$ corresponds to the hidden state at time t. It has been shown that, by enforcing the neuron independence (no matrix multiplication), back-propagation upon Eq. (4.1) becomes more stable and manageable. IndRNN has delivered good performance for very long sequences.

While IndRNN alleviates gradient vanishing by replacing matrix multiplication with scalar multiplication, the RhyRNN is proposed to further *shorten the longest path of the computational graph* of IndRNN independently for each neuron, through introducing a *skip* operator. This idea is similar to Campos *et al.* (2017) that implements skip operation on conventional RNN, which can be viewed as a Bernoulli distribution sampler on *UPDATE* or *COPY* operations at each timestamp t (an analogous idea appeared in h-detach Kanuparthi *et al.* (2019) which is applied on LSTM). The proposed RhyRNN differs from Skip-RNN in such a way that, unlike Campos *et al.* (2017) where *UPDATE* and *COPY* operations are computed on a whole hidden state $\mathbf{h}_t$ by a matrix multiplication, the proposed RhyRNN structure decides the choice of *UPDATE* or *COPY* operation by using Hardmard's product, which further makes the decision *independent of each neuron.* The mathematical formula of RhyRNN can be written as follows:

$$\mathbf{s}_t = f_{binarize}(\mathbf{o}_t) \tag{4.2}$$

$$\mathbf{h}_t = \mathbf{s}_t \odot \tilde{\mathbf{h}}_t + (\mathbf{1} - \mathbf{s}_t) \odot \mathbf{h}_{t-1} \tag{4.3}$$

$$\Delta \mathbf{o}_t = \zeta(\mathbf{w}_p \odot \mathbf{h}_t + \mathbf{b}_p) \tag{4.4}$$

$$\mathbf{o}_{t+1} = \mathbf{s}_t \odot \Delta \mathbf{o}_t + (\mathbf{1} - \mathbf{s}_t) \odot (\mathbf{o}_t + \min(\Delta \mathbf{o}_t, \mathbf{1} - \mathbf{o}_t)) \tag{4.5}$$

where $\zeta(\cdot)$ is the sigmoid activation function and $f_{binarize}$ is the step function: $f_{binarize} : [0,1]^n \rightarrow \{0,1\}^n$, which binarizes each input element. $\mathbf{w}_p$ is the weight vector that can be learned to obtain the incremental value $\Delta \mathbf{o}_t$. $\tilde{\mathbf{h}}_t$ is obtained by Eq (4.1) (replacing $\mathbf{h}_t$ with $\tilde{\mathbf{h}}_t$) and $\mathbf{h}_{t-1}$ is the hidden state from the previous timestamp.

Remark. There are two advantages of utilizing Hardmard's product in computing the gate value $\mathbf{s}_t$. Firstly, it keeps the independence of each neuron in IndRNN intact, which allows each neuron to have a distinct strategy of choosing *UPDATE/COPY* operations and thus being capable of capturing the varying context in different scales. This advantage will be demonstrated in Section 4.4. Secondly, the computation of the gradient of the RhyRNN is easier and more stable compared to either IndRNN or Skip-RNN, since the lengths of gradient path for different neurons can be shortened due to the skip operator, and the absence of matrix multiplication will yield more tractable gradient flow.

To enable the intra-neuron interaction, multiple layers of RhyRNNs are stacked and a matrix multiplication $\mathbf{W}_l$ is applied between layers to aggregate the global information. Specifically, assuming $\mathbf{h}_{t,l}$ to be the input to RhyRNN ($\mathbf{h}_{t,0} = \mathbf{x}_t$) at layer l and time t, the hidden state will be:

$$\mathbf{h}_{t,l} = \sigma(\mathbf{W}_l \mathbf{h}_{t,l-1} + \mathbf{u}_l \odot \mathbf{h}_{t-1,l} + \mathbf{b}_l) \tag{4.6}$$

Skip regularization

To limit the computational budget, a regularization term that controls the frequency of *UPDATE*s similar to Campos *et al.* (2017) is introduced. This term is written as:

$$\mathcal{L} = \lambda \sum_{t,k} \mathbf{s}_{t,k} \tag{4.7}$$

where $\mathbf{s}_{t,k}$ refers to the kth binary neuron decision (on *COPY* or *UPDATE* and the λ is a parameter which called cost per sample, see Eq. (4.2). In general, this term sums up the number of *UPDATE*s on all neurons at every time step and this term forces the network converges with fewer *UPDATE* decisions.

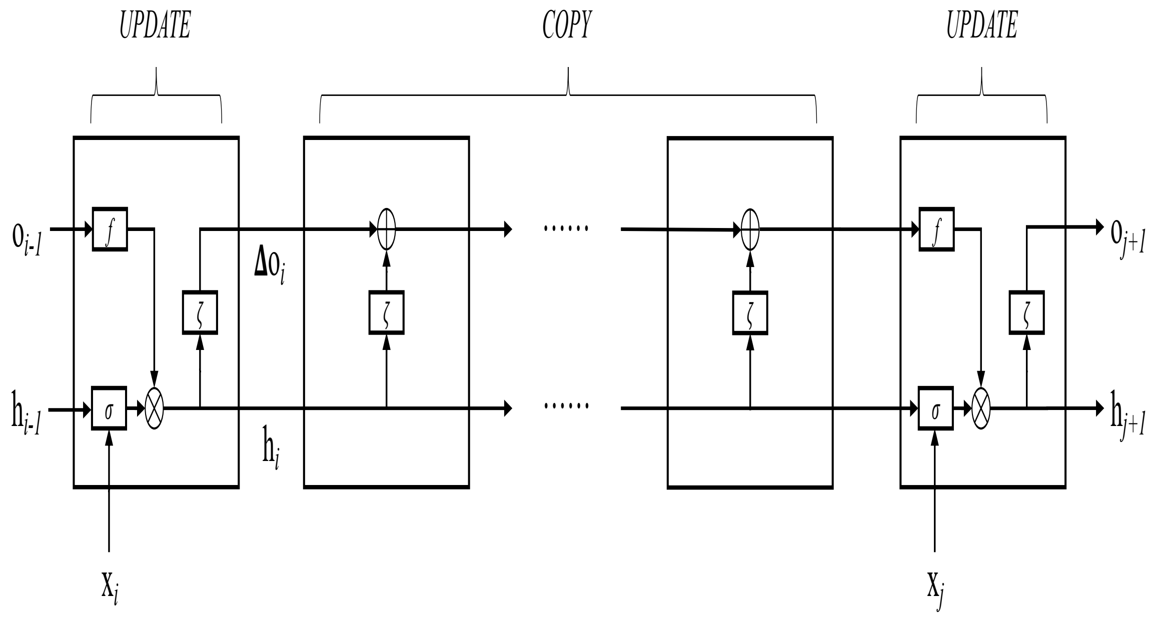


Figure 4.2: A Basic Unit in RhyRNN.

Gradient analysis

A strategy is employed to approximate the gradient of the step function $f_{binarize}$ as in Campos *et al.* (2017):

$$\partial f_{binaries}(x)/\partial x = 1 \tag{4.8}$$

In other words, Eq. (4.2) and Eq. (4.8) are implemented in forward-pass and backward-pass for the network, respectively. Following such a setting, the gradient during the backward pass by taking an example is shown in Fig. 4.2. In all the following analyses, the bias $\mathbf{b}$ and $\mathbf{b}_p$ are all discarded for simplicity. In Fig. 4.2, it shows that the *UPDATE* is emitted at time i and j and all the resting operations in between are *COPY*s. This segment can be viewed as a basic unit since any forward pass of RhyRNN can be separated into such segments (with varying numbers of *COPY*s). $\otimes$ and $\oplus$ correspond to element-wise product and plus, respectively. A sequence can be divided into several consecutive basic units. Then the gradient back-propagated can be written as the product of multiple gradients of such units. Since all operations between i and j are $COPY$s, the hidden state $\mathbf{h}_i$ will directly pass until j, thus in the forward pass there is:

$$\mathbf{h}_{j+1} = \sigma(\mathbf{W}\mathbf{x}_j + \mathbf{u} \odot \mathbf{h}_i) \tag{4.9}$$

In Eq. (4.9) a term $f_{binarize}(\mathbf{o}_j)$ is omitted since it equals 1 (see Fig. 4.2 at time j). However, this term will participate in the backward pass according to the gradient defined in Eq. (4.8). The Eq. (4.9) can be expanded as follows:

$$\mathbf{h}_{j+1} = \underbrace{f_{binarize}\left(\sum^{j-i} \zeta\left(\mathbf{w}_p \odot \mathbf{h}_i\right)\right)}_{=\mathbf{1}\text{,for the basic unit}} \odot \sigma(\mathbf{W}\mathbf{x}_j + \mathbf{u} \odot \mathbf{h}_i) \tag{4.10}$$

Given Eq. (4.10) and after a series of mathematical manipulations, the gradient at time i can be obtained by taking into account Eq. (4.8):

$$\nabla J_i = \frac{\partial \mathbf{h}_{j+1}}{\partial \mathbf{h}_i} = \underbrace{\mathbf{u} \odot \sigma' + \sigma \odot \sum^{j-i} \mathbf{w}_p \odot \mathbf{w}_p \odot (\mathbf{1} - \mathbf{w}_p \odot \mathbf{h}_i) \odot \mathbf{h}_i} \tag{4.11}$$

where the term within the underbrace is the basic unit for any such segment and σ' is the gradient of function σ. Thus, one can calculate the gradient at any time k (where at k there is an $UPDATE$ emitted) by calculating the element-wise product:

$$\left.\frac{\partial J}{\partial \mathbf{h}_k}\right|_{k=UPDATE} = \underline{\prod}_{l=UPDATE, l>k} \nabla J_l \tag{4.12}$$

where $\underline{\prod}$ is the element-wise product.

Two facts involved in this gradient chain rule is notified: 1) there is only scalar multiplication involved in the unit (and element-wise product of multiple such units) which is more tractable than matrix multiplication; 2) hidden states $\mathbf{h}_i$s directly participate in the back-propagation, which can correlate and thus stabilize the gradient from vanishing/exploding as discussed in Multiplicative Integration Wu *et al.* (2016). In this sense, RhyRNN has a gradient behavior benefiting from both IndRNN and Multiplicative Integration. More details on related analysis can be found in Li *et al.* (2018) and Wu *et al.* (2016).

4.3.3 Diversity-driven Pooling Layer

Though the proposed RhyRNN can capture the context at different scales to some extent, it still cannot fully utilize the intrinsically hierarchical context of long videos. In this section, a temporal pooling strategy that explicitly selects most contributing hidden states within a sequence is proposed.

The pooling stage is essential in CNNs, which aggregates low-level representations into high-level ones. A series of works also focused on temporal pooling where the objective is to hierarchically shorten and abstract the temporal representations Girdhar and Ramanan (2017); Fernando *et al.* (2016); Xu *et al.* (2017); Nguyen *et al.* (2018). In this chapter, a simple yet efficient method termed as diversity-driven sequential pooling (**DivPool**) by mostly diversifying the capacity of the pooled representations is proposed. The method is based on the observation that, since a video is always highly redundant, an effective pooled representation should ignore the slight difference across frames and concentrate on the most significant changes. Thus, the proposed pooling method performs selection to maximally diversify the hidden states (features). To this end, the dissimilarity by cosine distance between $\mathbf{h}_t$ and its previous state $\mathbf{h}_{t-1}$ is firstly calculated as:

$$a_t = 1 - \frac{\mathbf{h}_t \mathbf{h}_{t-1}^T}{\|\mathbf{h}_t\| \|\mathbf{h}_{t-1}\| + \epsilon} \tag{4.13}$$

where $\epsilon > 0$. Then all a_ts are sorted in descending order and the $\alpha\%$ most dissimilar states are selected as the pooled features. Note that this procedure works in an incremental fashion and thus a pairwise distance calculation on all states is not necessary, yielding high efficiency in implementation.

The DivPool layer has no learning parameters and thus is similar to max-pooling or average-pooling in CNNs. Yet it differs from max-pooling or average-pooling since it performs pooling globally on all features. Besides, it generates the pooling cue in an incremental fashion which is adopted in some effective sequential pooling strategies Girdhar and Ramanan (2017); Fernando *et al.* (2016). The only overhead of performing DivPool is on sorting a_t, which is typically $\mathcal{O}(n \log n)$ and can be efficient in practice.

Back-propagation

DivPool dynamically generates links across RhyRNN layers in the computational graph. The generated links will be effective during a forward-backward round for the network computation. The back-propagated gradients will only follow the general RNN's update path together with the current effective links. Fig. 4.3 schematically shows an example. In the Fig. 4.3, the black arrow indicates the flow in forward pass. Orange and blue arrows correspond to the shortest and the longest backward path from state $\mathbf{h}_{1,0}$ to the loss during backward pass, respectively. The "backward 1" path is shorter than "backward 2" path due to this hierarchy. From the Fig. 4.3, it can be observed that DivPool can greatly shorten the shortest computational path, which is dominant (compared to other longer paths) in propagating gradients to avoid the vanishing issue. For example, assuming the pooled ratio is 0.5 with q RhyRNN layers and the sequence length is n, the length of the shortest path becomes $\mathcal{O}(q + \log(n))$.

4.3.4 Bilinear Reweighting for Recognition

With DivPool, the redundancy and the complexity of the input video sequence has been reduced. In the following stage, to better incorporate the dependency of the long-range selected hidden states, the bilinear reweighting (BR) mechanism is designed to capture the temporal relation among the pooled hidden states.

In the proposed model, a simplified bilinear reweighting (BR) strategy is employed to learn and enhance the temporal correlation of patterns within the pooled hidden states (features). The BR module is applied to the output sequence of DivPool and to

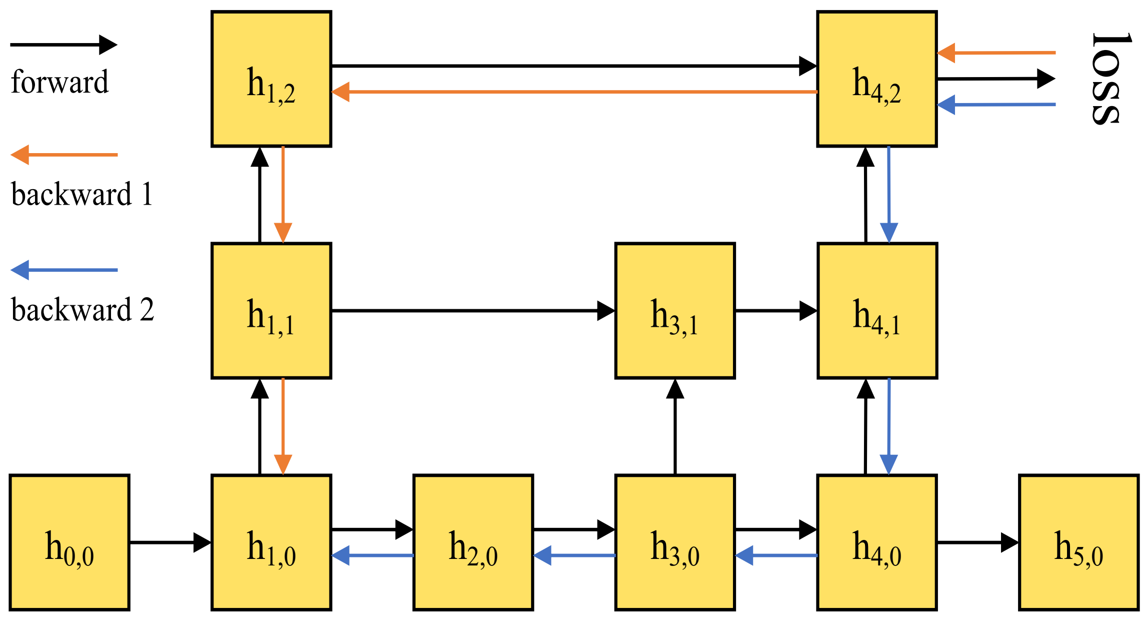


Figure 4.3: Schematic Diagram of Hierarchical Architecture with DivPool.

embed the hidden states to a new feature re-weighted by the pairwise affinity scores. BR module is inspired by bilinear attention Kim *et al.* (2018) but follows the metric properties. Assuming that the selected features from DivPool form a feature matrix $\mathbf{V}$, BR rule can be written as:

$$\begin{aligned} \mathbf{S} &= \mathbf{V}^\top \mathrm{norm}(\mathbf{M})\mathbf{V} \\ \hat{\mathbf{V}} &= \mathbf{V} \circ \mathrm{softmax}\left(\mathrm{proj}\left(\mathrm{norm}(\mathbf{S})\right)\right) \end{aligned} \tag{4.14}$$

where $\mathbf{M} = \mathbf{P}\mathbf{P}^\top$ is a symmetric semi-definite matrix and $\mathbf{P}$ is the parameter to be learnt. This decomposition is to reduce the number of weights to be learned. The output $\hat{\mathbf{V}}$ is the re-weighted feature matrix. norm$(\cdot)$ performs column-wise L^2-normalization and proj$(\cdot)$ projects a square matrix into a vector by summarizing the elements per-column. norm$(\cdot)$ performs twice to avoid the magnitude of the final affinity being too large (which may result in almost a one-hot reweighting vector). Note BR (Eq. (4.14)) differs from Bilinear Pooling in Kim *et al.* (2018) by introducing a column-wise *projection* operator. In this sense, only the magnitude of the input is adjusted, rather than replacing the input by a sum of all other inputs. The intuition is that, since the input $\mathbf{V}$ carries temporal information, a summing schema may violate or mix up this intrinsic (e.g. ordering information).

The output sequence $\hat{\mathbf{V}}$ of BR module is then fed to a standard GRU Cho *et al.* (2014). The output of the GRU module at the final time step is utilized as the video-level feature and two fully connected layers are appended following GRU to conduct recognition for different datasets.

4.4 Experiment

4.4.1 Datasets and Reference Methods

Experiments are conducted on Breakfast Kuehne *et al.* (2014), VIRAT 2.0 surveillance video Oh *et al.* (2011) and Charades Sigurdsson *et al.* (2016). All the experiments were done on a computer equipped with a *single* GTX Titan Xp GPU with 12GB memory.

Breakfast dataset

Kuehne *et al.* (2014) comprises of 10 breakfast preparation related events that are performed by 52 different individuals in 18 different kitchen scenes. The total number of video clips is 1989. The overall video duration is about 77 hours and the average length of each video is about 140 seconds. Events in the Breakfast dataset are very complicated since each event contains several sub-activities, indicating much higher intra-class variations. The dataset is splited into training and testing by following the “s1” split in Kuehne *et al.* (2014).

VIRAT 2.0 surveillance video dataset

Oh *et al.* (2011) includes about 8 hours of high-resolution surveillance videos (i.e. 1080p or 720p) with 12 kinds of events from 11 different scenes. In the experiment, only the 6 types of person-vehicle interaction events that occur on the parking lot scene are considered. The input video sequence only contains the event area that is cropped based on the ground truth bounding box. The training and testing video samples are randomly selected by following the ratio of 7:3. As such, the training multiple rounds are conducted and the average performance is reported.

Charades dataset

Sigurdsson *et al.* (2016) is a multi-label action video benchmark with 157 classes in total. Each video is around 30 seconds and contains 6 singleton actions on average. The experiment is conducted by following the same training/testing split in Hussein *et al.* (2019a) which contains 7.8k and 1.8k videos in each. The mean average precision (mAP) is reported on two challenging tasks: multi-label action recognition and temporal localization.

Reference methods

Several existing algorithms are employed for comparison. **C3D** Tran *et al.* (2015), **TSN** Wang *et al.* (2016a) and **TRN** Zhou *et al.* (2018) are implemented in a simple version with only spatial (RGB) features (without optical-flow). **Two-stream** Simonyan and Zisserman (2014a) and **Temporal Fields** Sigurdsson *et al.* (2017) utilize both RGB and optical flow features. **IDT** Wang and Schmid (2013) alters to employ action trajectories. For **C3D**, it is trained from scratch on all datasets and preprocess frames by following the Tran *et al.* (2015). For TSN, the frame feature is extracted from a pre-trained VGG16 which won't be fine-tuned in the training stage. Only the segmental consensus part of TSN are trained. The plain **IndRNN** Li *et al.* (2018) is also compared for the Breakfast dataset by stacking 6 layers of IndRNN cells and setting the dropout ratio to 0.5 for each other layer. **3D-ResNet** Hara *et al.* (2018) is employed as both a peer method and a backbone. Timeception Hussein *et al.* (2019a) (**TC**) is compared since the authors claimed that Timeception is a strong baseline for complex video and can capture long range dependency. (Supervised) **SuperEvents** Piergiovanni and Ryoo (2018) and (weakly-supervised) **ActGraph** Rashid *et al.* (2020) are selected for comparison on Charades. For the tests of the methods with CNN backbone on Breakfast and Charades, the same frame sampling procedure as in Hussein *et al.* (2019a) is conducted.

4.4.2 Implementation Details

ResNet101 He *et al.* (2016) pre-trained on ImageNet and I3D Carreira and Zisserman (2017) pre-trained on Kinetics400 are employed to extract features for all frame-wise based algorithms. For event recognition and multi-label action recognition/localization tasks, *Cross-Entropy* and *Binary Cross-Entropy* are applied as loss functions, respectively. For the proposed method, either ResNet101 or I3D backbone is NOT fine-tuned on three selected datasets during the training stage due to GPU resource limitation, different from some prior works Hussein *et al.* (2019a); Wang *et al.* (2018); Feichtenhofer *et al.* (2019); Wu *et al.* (2019) which update the CNN backbones. The features are obtained from the last pooling layer of ResNet101 and I3D, yielding 2048-d and 1024-d, respectively. The output (as well as all hidden state) dimension of RhyRNN is 256 and the dimension of the output of BR is 128. Furthermore, there are two fully connected layers with 100 and the number of classes (e.g. 10 for event recognition on Breakfast) neurons following BR.

For 3D-ResNet50 and Timeception Hussein *et al.* (2019a) models, the 3D video segments with size of $1 \times 7 \times 7$ ($time \times height \times width$) are extracted from a 3D ResNet-50 model which is pre-trained on Kinetics-400 Kay *et al.* (2017). This experiment is implemented by following the settings in Hussein *et al.* (2019a) and 64 uniformly sampled video segments are collected, while each segment contains 8 successive frames.

All the proposed and baseline algorithms are implemented with PyTorch Paszke *et al.* (2017) toolbox. The proposed model is trained with 100 epochs and the Adam optimizer with the learning rate $1e-5$ is utilized. The pooling ratio for DivPool is set to 25% and the control parameter λ for the skip regularization is set to $1e-7$ empirically. For the Breakfast dataset, the training samples are subsampled every 5 frames. The first and last 10 frames are removed from the training samples since those frames are mostly redundant.

4.4.3 Breakfast Dataset

Experimental results (of event-level recognition and multi-label activity recognition) on this dataset are summarized in Tab. 4.1a [1] . In Table. 4.1a, the performances of (a) event recognition (in Acc) and multi-label classification (in mAP), (b) different settings of RhyRNNs are presented. "RhyRNN(2-layer)" is a model stacked with 2 layers of RhyRNNs concatenated with 2 fully connected layers. Blue color corresponds to singleton RNNs. In general, the proposed method (full setting) outperformed all the other peer methods in event-level recognition, with competitive performance against state-of-the-art on multi-label recognition. It also can be observed that the proposed RhyRNN (2-layer) has better performance compared to IndRNN (6-layer) or SkipRNN without introducing any other modules.

The behavior of the proposed model with multiple RhyRNN + DivPool settings (without BR) is further evaluated on Breakfast, as shown in Tab. 4.1b. Specifically, "4 RhyRNN + 1 DivPool" and "4 RhyRNN + 2 DivPool" settings are added, referring to the structure $\{4\times\text{RhyRNN}+\text{DivPool}\}$ and $\{2\times\text{RhyRNN}+\text{DivPool}+2\times\text{RhyRNN}+$

[1]The method TC Hussein *et al.* (2019a) is re-implemented following the same setting but did not obtain the performance reported in the original paper. Since there is a large gap between the re-implementation and their results, the best performance of Hussein *et al.* (2019a) is reported in the re-implementation.

Table 4.1: Results on Breakfast dataset.

(a) Event Recognition

Method	Feature	Acc (%)	mAP (%)
TSN Wang *et al.* (2016a)	2D	14.3	-
LRCN Donahue *et al.* (2015)	2D	13.3	-
C3D Tran *et al.* (2015)	3D	14.6	-
IndRNN(6-layer) Li *et al.* (2018)	2D	19.4	14.1
SkipRNN Campos *et al.* (2017)	2D	31.9	28.7
IndRNN(+DivPool+BR)	2D	42.7	40.8
SkipRNN(+DivPool+BR)	2D	40.2	-
3D-Res50 Hara *et al.* (2018)	3D	23.7	-
3D-Res50+TC Hussein *et al.* (2019a)	3D	40.3	41.2
RhyRNN(2-layer)	2D	35.8	30.5
RhyRNN(+DivPool+BR)	2D	**44.3**	**41.9**

(b) Different Settings

Setting	Acc
2 RhyRNN + 1 DivPool	43.7
4 RhyRNN + 1 DivPool	43.1
4 RhyRNN + 2 DivPool	41.4

Table 4.2: Results on VIRAT 2.0 dataset.

Method	original	S1	S2	S3
C3D Tran *et al.* (2015)	42.9	40.2	37.7	41.1
TSN Wang *et al.* (2016a)	52.4	52.1	51.6	51.9
IndRNN (6-layer) Li *et al.* (2018)	77.6	78.3	78.2	78.0
IndRNN (+DivPool+BR)	79.0	77.4	78.2	78.2
Ours (full setting)	**81.9**	**81.5**	**81.7**	**80.4**

DivPool}, respectively. It can be concluded that 2-layer RhyRNN (standard setting in all tests) has slightly better performance than other two.

Independent Skip strategy

To investigate the effectiveness of "Skip" operations in RhyRNN, the Skip operations of the first 10 neurons of the weights in the second RhyRNN layer on a breakfast video clip (with length 212) in the testing stage is visualized (in Fig. 4.4). In the figure, yellow and blue bars correspond to "*UPDATE*" and "*COPY*" operations, respectively. Vertical and horizontal axes refer to channel and frame, respectively. 10 neurons perform Skip with almost different rhythm to each other. It is seen that almost every neuron (each row) indeed holds a distinct and independent Skip rhythm as well. While some neurons emit *UPDATE*s with high frequency to capture the context in high temporal resolution (e.g., neuron 1, 4 and 10), other neurons learn lazier strategies.

4.4.4 Video and Image Retrieval and Analysis Tool 2.0 Dataset

The performance of the proposed method with full setting (RhyRNN + DivPool + BR) on this dataset is shown in Tab. 4.2. Specifically, the capacity of algorithms under *varying sampling rhythm* is compared to training rhythm. As shown in Tab. 4.2,

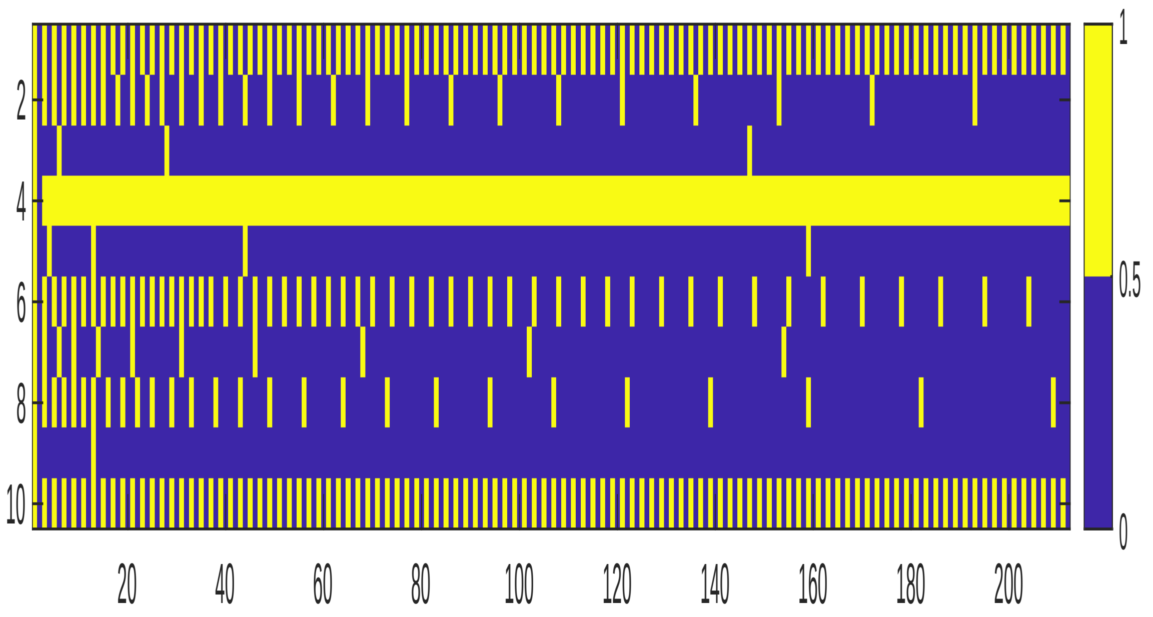


Figure 4.4: Visualization of “skip” Operation on the First 10 Channels/Neurons (out of 256) at the Second Layer of RhyRnn on a Video with 212 Frames from the Breakfast Dataset.

“original” indicates sampling each frame (and feature) with the same sampling rhythm at the training stage. The other three scenarios are designed with different combinations of sampling rates. To make the problem more challenging, first each testing video sequence is equally divided into three intervals and apply different sampling rates to each interval to form a new testing sequence. For scenario one (S1), the first and the third intervals are subsampled with every 2 and 5 frames respectively, while keeping the rhythm intact for the middle interval. In scenario two (S2), the first and third intervals are sampled every 5 and 2 frames, respectively (reverse way of S1). For the last scenario (S3), a half length of the testing frames is randomly sampled out. Since the randomness of the last scenario brings uncertainty, the well-trained model is tested 5 times and the average performance of this scenario is reported accordingly. It can be seen that the proposed model is quite stable under varying sampling rhythm.

4.4.5 Charades Dataset

For the Charades dataset, the I3D Carreira and Zisserman (2017) with 1024-D output feature pre-trained on Kinetics-400 is employed **without** inheriting any frame-level knowledge from or fine-tuning on the Charades dataset Carreira and Zisserman (2017); Wang *et al.* (2018); Hussein *et al.* (2019a). The frame stride is set to 8 for the I3D model and the size of the feature matrix for each video clip equals to Timelength$\times$ 1024 where the TimeLength = FrameLength/8. Two challenging tasks are evaluated in the experiment: multi-label (MLA) recognition and temporal localization.

The Tab. 4.3 presents the performance of the two tasks. In the table, for (a) recognition, “w/o BR” and “w/ BR” refer to the settings removing and keeping BR, respectively. For (b) localization, “S” and “W” refer to “supervised” and “weakly supervised”, respectively. *IndRNN in the table indicates IndRNN+DivPool+BR. The result of Skip-RNN is quoted from original paper Campos *et al.* (2017) where mAP is calculated per 100 frames instead of 25 frames. In the table, the ”TS” represents two-stream and ”TF” represents temporal fields.

Tab. 4.3a shows the MLA recognition performance of different algorithms on the Charades dataset. And the results demonstrate that the proposed algorithm has a competitive capacity compared to the state-of-the-art on capturing the temporal information of sub-actions in the complex videos.

Tab. 4.3b summarizes the results of temporal localization. To this end, the model parameters pre-trained on MLA recognition task is inherited but with removing the DivPool module [2] . Then the last 2 fully-connected layers is fine-tuned with BCE loss on MLA recognition task (only for the last time stamp) for 5 epochs. In the testing stage, the output of the RhyRNN at *each* time stamp is passed through the

[2]DivPool cannot work frame-wise since it selects only a portion of time stamps. Therefore DivPool from the full setting is removed.

Table 4.3: Result on Charades of Multi-Label Activity (MLA).

(a) MLA Recognition

Method	Modality	mAP(%)
C3D Tran *et al.* (2015)	RGB	10.9
TS Simonyan and Zisserman (2014a)	RGB+Flow	18.6
TS+LSTM Simonyan and Zisserman (2014a)	RGB+Flow	17.8
IndRNN* Li *et al.* (2018)	RGB	21.1
IDT Wang and Schmid (2013)	RGB+Flow	17.2
TF Sigurdsson *et al.* (2017)	RGB+Flow	22.4
TRN Zhou *et al.* (2018)	RGB	25.2
The Proposed(w/o BR)	RGB	24.6
The Proposed(w/ BR)	RGB	**25.4**

(b) MLA Localization

Model	Training	mAP(%)
LSTMPiergiovanni and Ryoo (2018)	S	10.4
Skip-RNNCampos *et al.* (2017)†	S	8.94
TS+LSTMPiergiovanni and Ryoo (2018)	S	18.2
SuperEventsPiergiovanni and Ryoo (2018)	S	**19.4**
TFSigurdsson *et al.* (2017)	S	12.8
I3DCarreira and Zisserman (2017)	S	17.2
ActGraphRashid *et al.* (2020)	W	15.8
The Proposed	W	**17.6**

last 2 fully-connected layers to produce score of action classes for each frame. Action class with the highest score is regarded as the predicted label for the current frame. Since during the training stage no frame-level label is provided, the proposed model is trained in a **weakly supervised** fashion, which is more challenging than fully supervised localization task (e.g. Piergiovanni and Ryoo (2018); Sigurdsson *et al.* (2017)). Surprisingly, it can be seen that the proposed model outperforms several fully supervised counterparts and a very recent weakly-supervised method ActGraph Rashid *et al.* (2020). This observation supports the claim that RhyRNN is capable of capturing temporal context at multiple levels.

4.4.6 Ablation Analysis and Model Size

Ablation analysis was conducted on the Breakfast dataset following the settings in Sec. 4.4.3. Results are summarized in Tab. 4.4. In the full setting, 2 layers of RhyRNNs are stacked followed by DivPool and BR. By turning off DivPool, all the output states of RhyRNN are simply fed to BR. On the other hand, the BR is removed and the pooled states are fed to a naive GRU once turning off BR. The capacity of the RhyRNN module is tested in capturing complex temporal information by turning off both DivPool and BR modules as well.

It can be observed that the full setting (RhyRNN + DivPool + BR) delivers the best performance among all. And the DivPool module plays a more important role in understanding the long and complex videos, compared with the BR module. Moreover, the RhyRNN module itself is able to acquire information from long and complex videos compared with some conventional algorithms presented in the Tab 4.1a. In general, the proposed modules in the framework consistently enhance the performance.

Tab. 4.5 summarizes the models sizes on different setting under event-level recog-

Table 4.4: Ablation Study

RhyRNN	DivPool	BR	Acc	mAP
√	√	√	44.3	41.9
√	√	×	43.7	40.5
√	×	√	40.7	35.2
√	×	×	35.8	-

Table 4.5: Model Size

Method	Model size
TC Hussein *et al.* (2019a)	15.8MB
The Proposed (Full setting)	23.2MB
The Proposed (RhyRNN+DivPool)	20.4MB

nition on the Breakfast dataset. The size of the proposed model with or without BR is around 20MB, comparative to the size of state-of-the-art method TC Hussein *et al.* (2019a), which claimed to be capable of reducing the model size significantly.

4.5 Conclusion

In this chapter, the task of recognizing events in long and complex videos is fully studied. Since there is critical distinction between traditional action recognition based on short clips and event recognition using long videos, simply adapting the methods for the former to the latter is ineffective. To address this, an end-to-end RNN framework is designed taking into account the latent context at multiple levels. Especially, three novel and essential parts were proposed: RhyRNN, DivPool and BR. By taking advantage of each, the proposed model can capture video context at different scales in an adaptive and hierarchical fashion. The property of the proposed model is investigated in this chapter and demonstrated its superiority through extensive experiments even without the need of fine-tuning the feature extraction backbones.

Chapter 5

INCORPORATING DETERMINANTAL POINT PROCESS WITH DEEP LEARNING MODELS FOR LEARNING A DIVERSE FEATURE SPACE

As mentioned in the Chapter 1 Introduction, many visual tasks, like hashing, semantic retrieval, graph knowledge construction and so on, require a diversity in the feature space. However, the classification-like, or the label information drive, or the supervised learning process tasks may force the training of the model of obtaining visual features converge into several points depended on the number of categories which should cause problem and performance decay for those visual tasks. Even though the retraining or transfer learning of pretrained model with different object functions could mitigate the influence of the lack of diversity, the costly progress of defining a suitable object function and the retraining of the model is still a problem for obtaining a more diverse and representative visual features. Therefore, in this chapter, a more effective and useful algorithm is proposed to ensemble a regularization term with the common neural networks for classification problem to spread out the obtained feature vectors in the space without requirement of new object functions.

As it is already been known that, the determinantal point processes (DPPs) is an effective tool to deliver diversity in multiple machine learning and computer vision tasks. Under the deep learning framework, DPP is typically optimized via approximation, which is not straightforward and has some conflicts with the diversity requirement. However, there have been no deep learning paradigms to optimize DPP *directly* since it involves matrix inversion that may result in computational instability. This fact greatly hinders the use of DPP on some specific objective functions where DPP would otherwise serve as a term to measure the feature diversity. In this chapter, a simple but effective algorithm is designed to optimize the DPP term directly through expressing with L-ensemble in the spectral domain over the gram matrix, which is more flexible than learning on parametric kernels. By further taking into account additional geometric constraints, the proposed algorithm seeks to generate valid sub-gradients of the DPP term in cases where the DPP gram matrix is not invertible (no gradients exist in this case). In this sense, the proposed algorithm can be easily incorporated with multiple deep learning tasks. In this chapter, several experiments on image related datasets show the effectiveness of the proposed algorithm, indicating promising performance for practical learning problems.

5.1 Introduction

Diversity is desired in multiple machine learning and computer vision tasks (e.g., image hashing (Chen *et al.*, 2017; Carreira-Perpinán and Raziperchikolaei, 2016), descriptor learning (Zhang *et al.*, 2017), metric learning (Mishchuk *et al.*, 2017) and video summarization (Sharghi *et al.*, 2018; Liu *et al.*, 2017)), in which sub-sampled points or learned features need to spread out through a specific bounded space. Originated from quantum physics, determinantal point processes (DPP) have shown its power in delivering such properties (Kulesza *et al.*, 2012; Kulesza and Taskar, 2011b). Compared with other diversity-oriented techniques (e.g., entropy (Zadeh *et al.*, 2017) and orthogonality (Zhang *et al.*, 2017)), DPP shows its superiority as it incorporates only one single metric and delivers genuine diversity on *any bounded space* (Kulesza *et al.*, 2012; Affandi *et al.*, 2013; Gillenwater *et al.*, 2012). Therefore, DPP has been utilized in a large body of diversity-oriented tasks.

In general, sample points from a DPP tend to distribute diversely within a bounded space $\mathcal{A}$ (Kulesza *et al.*, 2012). Given a positive semi-definite kernel function κ : $\mathcal{A} \times \mathcal{A} \to \mathbb{R}$, the probability of a discrete point set $\mathcal{X} \subset \mathcal{A}$ under a DPP with kernel function κ can be characterized as:

$$\mathcal{P}_{\kappa}(\mathcal{X}) \propto \det(\mathbf{L}_{\mathcal{X}}) \tag{5.1}$$

where $\mathbf{L}$ is a $|\mathcal{X}| \times |\mathcal{X}|$ matrix with entry $\mathbf{L}_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$ and $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$. $\mathbf{L}$ is called L-ensemble. Note that $\mathcal{A}$ is a continuous space, whereas $\mathcal{X}$ is finite. In the Hilbert space associated with κ, larger determinant implies larger spanned volume, thus the mapped points tend not to be similar or linearly dependent.

DPP can be viewed from two perspectives: sampling and learning. A comprehensive introduction to mathematical fundamentals of DPP for sampling from a discrete space can be found in Kulesza *et al.* (2012). Based on this, a line of works has been proposed (Kulesza and Taskar, 2011a; Kang, 2013; Hennig and Garnett, 2016). In this dissertation, the chapter 6 concentrates on demonstation of learning DPPs. In learning of DPP, the term $\det(\mathbf{L})$ is typically treated as a singleton diversity measurement and is extended to learning paradigms on continuous space (Chao *et al.*, 2015; Kulesza and Taskar, 2010; Affandi *et al.*, 2014). There are generally two lines of strategies to learn DPPs:

Approximation. This type of methods is to convert DPP into a simpler format which can ease and stabilize the computation. *low-rank approximation* proves powerful in easing the computational burden (Gartrell *et al.*, 2017), in which the gram matrix is factorized as $\mathbf{L} = \mathbf{B}\mathbf{B}^{\top}$ where $\mathbf{B} \in \Re^{n \times m}$ with $m \ll n$. This decomposition can also reduce the complexity which is originally a cubic time of $|\mathbf{L}|$. Kulesza and Taskar (2011b) explicitly expressed the kernel with $\kappa(\mathbf{x}, \mathbf{y}) = \sigma_1 \sigma_2 \delta(\mathbf{x})^{\top} \delta(\mathbf{y})$, where σ measures the intrinsic quality of the feature and $\delta(\cdot)$ is function mapping input $\mathbf{x}$ to a feature space. In this sense, the pairwise similarity is calculated in Euclidean feature space with cosine distance. Elfeki *et al.* (2019) suggest approximating a given distribution by approximating the eigenvalues of the corresponding DPP. As such, the computation can be eased and become stable. Following this, DPP is also applied on some visual tasks, such as video summarization (Sharghi *et al.*, 2018), ranking (Liu *et al.*, 2017) and image classification (Xie *et al.*, 2017). It can be noted that the approximation is not straightforward for DPP, thus cannot fully deliver the diversity property (e.g. resulting in rank-deficiency).

Direct optimization. While the aforementioned methods optimize DPP with specific approximation, a series of efforts also seek to optimize the DPP term *directly* (Gillenwater *et al.*, 2014; Mariet and Sra, 2015; Bardenet and Titsias, 2015). In this setting, the whole gram matrix $\mathbf{L}$ corresponding to the pairwise similarity among features is updated directly, which allows accommodating more flexible feature mapping functions rather than an approximation. Gillenwater *et al.* (2014) proposed an Expectation-Maximization algorithm to update marginal kernel DPP $\mathbf{K} = \mathbf{L}(\mathbf{L} + \mathbf{I})^{-1}$, together with a baseline K-Ascent derived from projected gradient ascent (Levitin and Polyak, 1966). Mariet and Sra (2015) extended DPP from a fixed-point perspective and Bardenet and Titsias (2015) proposed to optimize DPP upon a lower bound in variational inference fashion. A key problem of such line of works is that the computation is not differentiable, making it difficult to be used in deep learning frameworks.

To the best of current knowledge, there is no previous method incorporating DPP as a feature-level diversity metric in *deep learning*. A key difficulty in doing such method is that the calculation of the gradient of $\det(\mathbf{L})$ involves matrix inversion, which can be unstable and inaccurate in GPUs. Though K-Ascent seems to be a naive rule, it still needs explicit matrix inversion in the first step before the projection procedure. This fact greatly hinders the tight integration of DPP with deep networks. Some alternative methods seek to reach diversity under more constrained settings. For example, Zhang *et al.* (2017) resorted to a global pairwise orthogonality constraint in hyper-sphere and Zadeh *et al.* (2017) employed statistical moments to measure the diversity. However, compared with DPP, such measurements are unable to fully characterize diversity in an arbitrary bounded space.

In this dissertation, rather than providing more efficient DPP solvers, the way of delivering a feasible feature-level DPP integration under the deep learning framework is concentrated on. To this end, the spectral decomposition of DPP is revisited and a sub-gradient generation method which can be tightly integrated with deep learning is proposed in this chapter. The proposed method differs from either approximation or direct optimization by introducing a "differentiable direct optimization" procedure, thus can produce genuinely diverse features in continuous bounded space. The proposed algorithm is stable and scalable to the relatively large dataset with a specific mini-batch sampling strategy, which is verified by several experiments on various tasks.

The contribution of this chapter can be concluded as follows:

- The spectral sub-gradient is proposed for DPP, which overcomes the long-lasting optimization difficulty in deep learning frameworks.
- The proposed method can regularize the network to learn a smoother mapping.
- The newly proposed DPP regularizer can enhance the performance of specific diversity-driven tasks.

Notations: Bold lower case $\mathbf{x}$ and bold upper case $\mathbf{K}$ represent vector and matrix, respectively. $\det(\cdot)$ and $(\cdot)$ calculate the determinant and trace of a matrix, respectively. $\mathbf{A} \otimes \mathbf{B}$ is the element-wise product of matrices $\mathbf{A}$ and $\mathbf{B}$. $|\mathcal{X}|$ and $|\mathbf{x}|$ measure the cardinality of a finite set $\mathcal{X}$ and the L^2 length of a vector $\mathbf{x}$, respectively. $\langle \mathbf{x}, \mathbf{y} \rangle$ calculates the inner product of the two vectors. $\mathbf{x} = \mathrm{diag}(\mathbf{X})$ transforms a diagonal matrix $\mathbf{X}$ into its vector form $\mathbf{x}$, and vice versa. The "positive semi-definite" and "positive definite" are referred to PSD and PD, respectively. Denote $\Re$ the real numbers.

5.2 Background

5.2.1 Determinantal Point Process

L-ensemble expression of DPP requires $\mathbf{L}$ to be PSD, whereas kernel expression further constrains $\mathbf{K} < \mathbf{I}$ (each eigenvalue of $\mathbf{K}$ is less than 1). A conversion from $\mathbf{L}$ to $\mathbf{K}$ can thus be written as $\mathbf{K} = \mathbf{L}(\mathbf{L}+\mathbf{I})^{-1}$ following the truth $\sum_{\mathcal{X}} \det(\mathbf{L}_{\mathcal{X}}) = \det(\mathbf{L}+\mathbf{I})$, which is the marginal normalization constant given a specific $\mathbf{L}$. While there is always conversion from $\mathbf{L}$ to $\mathbf{K}$, the inverse may not exist (Kulesza *et al.*, 2012). In practice, one may construct L-ensemble first, then normalize it into a marginal kernel. This fact may give rise to the difficulty of deep networks. Since a conversion from $\mathbf{K}$ to $\mathbf{L}$ might not exist, the network needs carefully adjusting the gradients under specific constraints to ensure the updated $\mathbf{L}$ to be valid. As $\mathbf{L}$ and $\mathbf{K}$ share the same eigenvectors $\mathbf{v}_i$, a pair of $\mathbf{L}$ and $\mathbf{K}$ holds the relation:

$$\mathbf{K} = \sum_i \lambda_i \mathbf{v}_i \mathbf{v}_i^\top \quad \Longleftrightarrow \quad \mathbf{L} = \sum_i \frac{\lambda_i}{1-\lambda_i} \mathbf{v}_i \mathbf{v}_i^\top \tag{5.2}$$

where λ_i is the ith eigenvalue. It is seen that such conversion is not straightforward to be directly integrated with deep learning framework. Therefore, the ensemble $\mathbf{L}$ is optimized directly in this dissertation.

5.2.2 Gaussian Kernel

The Gaussian kernel is briefly introduces in this subsection, which works on Hilbert space with infinite dimension. Mercer's theorem Friedman *et al.* (2001) ensures the PSD properties when constructing new kernels with existing ones under a specific procedure. Such procedure is also employed in multiple kernel learning paradigms

(Affandi *et al.*, 2014; Kulesza and Taskar, 2011b; Chao *et al.*, 2015), which is out of the scope of this dissertation.

A Gaussian kernel is defined as $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-|\mathbf{x}_i - \mathbf{x}_j|^2/\sigma^2\right)$, where σ is a controlling parameter. Thus an L-ensemble matrix becomes $\mathbf{L}_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$. According to the definition, $\mathbf{L}_{ii} = 1$ and for any element in the matrix can be described in an interval as $\mathbf{L}_{ij} \in (0, 1]$. With Gaussian kernel, a nice property $0 \leq \det(\mathbf{L}) \leq 1$ can be inferred. This can be easily verified by applying geometric inequality to the eigenvalues of $\mathbf{L}$. Although not tight, this property shows that the determinant value with Gaussian kernel is bounded. This fact inspires one version of the proposed algorithm detailed in the next section. Throughout this chapter, all of the discussion is based on the Gaussian kernel unless specified.

Note: Since the similarity between each point pairs are measured in a continuous space, the normalization of L2 norm (or Euclidean Distance) should be a good measurement for the similarity metric. Compared with the L2 norm, L1 norm (or the Manhattan Distance) is not so suitable for measuring the continuous feature space. Therefore, it is natural to use the Gaussian Kernel (a normalization function of L2 norm distance) other than Laplacian Kernel (a normalization function of L1 norm distance). However, the kernel itself only defines the similarity metric for feature space. Therefore, for different situations, the Gaussian kernel (or the similarity metric) can be replaced by any other kernels. But in this chapter, the Gaussian Kernel is only considered since the data is more suitable for L2 norm distance.

5.3 Method

Given vectorized inputs $\mathbf{I}_i \in \mathbb{R}^h$ where $i = 1, ..., n$, the goal of this proposed method is to learn a map f such that the features $\mathbf{x}_i = f(\mathbf{I}_i)$ can spread out within a bounded feature space $\mathbf{x}_i \in \mathcal{S}$. Hereafter the *space* is referred to an Euclidean bounded space (e.g., $[-1,1]^d$) without loss of generality. Given any loss function J, the chain rule of gradient involving DPP is written as:

$$\Delta J = \frac{\partial J}{\partial \det(\mathbf{L})} \frac{\partial \det(\mathbf{L})}{\partial \mathbf{L}} \frac{\partial \mathbf{L}}{\partial \mathbf{X}} \tag{5.3}$$

where $\mathbf{X}$ refer to the features before DPP layer. While calculating $\partial J / \partial \det(\mathbf{L})$ and $\partial \mathbf{L} / \partial \mathbf{X}$ is straightforward, the main difficulty lies on the calculation of $\partial \det(\mathbf{L}) / \partial \mathbf{L}$. The discussion of the calculation of this term is conducted under two case: 1) When the inversion $\mathbf{L}^{-1}$ can be stably obtained, the derivation of the gradient of DPP $\det(\mathbf{L})$ is shown on Sec 5.3.1; 2). When $\mathbf{L}$ is not invertible or $\mathbf{L}^{-1}$ is difficult to calculate, the procedure to handle the case by generating valid sub-gradient is given in details in Sec 5.3.2. Since the objective of this chapter is to diverse features, $\det(\mathbf{L})$ will serve as a (partial) objective term to be directly maximized.

5.3.1 Derivation of Gradient

With kernel κ, a DPP regularization term seeks to maximize the possibility of a feature configuration $\mathbf{x}_i$, $i = 1, ..., n$. As this possibility is proportional to $\det(\mathbf{L})$, the objective is $\max \det(\mathbf{L})$. This can become a regularization term where diversity is required. Thus with a general loss function L_G, the aim is to solve $\min L_G - \lambda_1 \det(\mathbf{L})$, with the controlling parameter $\lambda_1 \geq 0$. For the time being, the kernel matrix $\mathbf{L}$ is supposed to be invertible (the case when $\mathbf{L}$ is not invertible will be discussed in the next section), hence $\mathbf{L}^{-1}$ exists. Without loss of generality, the gradient of the determinant equipped with Gaussian kernel is discussed in this dissertation. For other kernels the derivation is analogous. For a Gaussian Kernel, $\mathbf{L}_{ij}$ can be further factorized as:

$$\mathbf{L}_{ij} = \exp\left(-\frac{\sum_l (\mathbf{x}_{il} - \mathbf{x}_{jl})^2}{\sigma^2}\right) \tag{5.4}$$

where $\mathbf{x}_{ij}$ is the jth dimension of feature $\mathbf{x}_i$. Using chain rule, the derivative of $\det(\mathbf{L})$ w.r.t. $\mathbf{x}_{il}$ can be written as:

$$\frac{\partial \det(\mathbf{L})}{\partial \mathbf{x}_{il}} = \det(\mathbf{L}) \left(\mathbf{L}^{-1} \frac{\partial \mathbf{L}}{\partial \mathbf{x}_{il}} \right) \tag{5.5}$$

where on the ijth position of $\frac{\partial \mathbf{L}}{\partial \mathbf{x}_{il}}$ the corresponding element is:

$$\left(\frac{\partial \mathbf{L}}{\partial \mathbf{x}_{il}} \right)_{ij} = \exp\left(-\frac{|\mathbf{x}_i - \mathbf{x}_j|^2}{\sigma^2} \right) \left(-\frac{2(\mathbf{x}_{il} - \mathbf{x}_{jl})}{\sigma^2} \right) \tag{5.6}$$

Eq (5.6) can be more compactly expressed as:

$$\frac{\partial \mathbf{L}}{\partial \mathbf{x}_{il}} = \mathbf{L} \otimes \mathbf{M}^{(il)} \tag{5.7}$$

where $\mathbf{M}^{(il)}$ is such a matrix that, except for the ith column and row, all resting elements are 0s. Besides, the ijth and jith elements of $\mathbf{M}^{(il)}$ are both $-\frac{2(\mathbf{x}_{il} - \mathbf{x}_{jl})}{\sigma^2}$. In summary, Eq (5.5) can be simplified as:

$$\frac{\partial \det(\mathbf{L})}{\partial \mathbf{x}_{il}} = \det(\mathbf{L}) \left(\mathbf{L}^{-1} \left(\mathbf{L} \otimes \mathbf{M}^{(il)} \right) \right) \tag{5.8}$$

To ease the computation and fully utilize the chain rule in deep learning architecture, the DPP loss is decomposed into two layers, and the corresponding gradient product can be expressed as:

$$\left(\frac{\partial \det(\mathbf{L})}{\partial \mathbf{L}} \right) \cdot \left(\frac{\partial \mathbf{L}}{\partial \mathbf{x}} \right) \tag{5.9}$$

While the existing package can be utilized to obtain $\partial \mathbf{L}/\partial \mathbf{x}$ reliably, the way to stably calculate $\partial \det(\mathbf{L})/\partial \mathbf{L}$ becomes essential. The detail will be presented in the next section once the term is hard to calculate.

5.3.2 Proper Spectral Sub-gradient for Back-propagation

The calculation of the gradient $\partial \det \mathbf{L}/\partial \mathbf{L}$ involves computing the inverse matrix $\mathbf{L}^{-1}$. However, the kernel matrix $\mathbf{L}$ is not always invertible. This situation happens

iff there exists at least a pair of features $\mathbf{x}_i$ and $\mathbf{x}_j$ such that $\mathbf{x}_i = \mathbf{x}_j$. In this case, there exist two identical columns/rows of $\mathbf{L}$ and the 0 eigenvalue results in the non-invertibility. This phenomenon is sometimes caused by ReLU function, which will map different input values onto an identical one. Even when all features are distinct, the numerical precision (typically on float number in GPU) may also lead to failure. It is occasionally observed that GPU calculation of $\mathbf{L}^{-1}$ reports error even no eigenvalue is 0. One may imagine a naive replacement of matrix inverse with the pseudo-inverse, which can be applied on singular matrices. However, pseudo-inverse will keep the zero eigenvalues intact (still rank-deficiency), and the back-propagated gradient will play no part to increase the determinant value (both 0 before and after updates).

To address this, the objective of DPP $\max \det(\mathbf{L})$ is firstly diverged to be considered. Since DPP term seeks to maximize the determinant, for a configuration $\mathbf{L}^{(t)}$ at iteration t with $\det(\mathbf{L}^{(t)}) = 0$, any sufficiently small η sufficing $\det(\mathbf{L}^{(t+1)}) > 0$ with $\eta = \mathbf{L}^{(t+1)} - \mathbf{L}^{(t)}$ can be a valid ascending direction. Thus the following definition is given:

Definition 5.3.1. *Proper Sub-gradient: For a PSD matrix* $\mathbf{L}$ *such that* $\det(\mathbf{L}) = 0$*,* $\hat{\mathbf{L}}$ *is called its proper sub-gradient if* $\hat{\mathbf{L}}$ *is a sub-gradient and* $\det(\mathbf{L} + \alpha\hat{\mathbf{L}}) > 0$ *for sufficiently small* $\alpha > 0$*.*

It is shown that if a proper sub-gradient $\hat{\mathbf{L}}$ can be found at $\det(\mathbf{L}) = 0$, back-propagation procedure in deep learning can consequently perform calculation using $\hat{\mathbf{L}}$. To obtain such $\hat{\mathbf{L}}$, it should be noted firstly that $\mathbf{L}$ can be eigen-decomposed as following since it is symmetric and PSD:

$$\mathbf{L} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top \tag{5.10}$$

where $\mathbf{U}$ is the orthogonal eigenvector matrix and $\mathbf{\Lambda}$'s diagonal elements are the corresponding eigenvalues. As $\mathbf{L}$ has zero eigenvalues, the rank of $\mathbf{\Lambda}$ is lower than the dimension of $\mathbf{L}$. All eigenvalue are sorted into descending order to $\mathbf{k} = (\sigma_1, ..., \sigma_q, 0, ..., 0)$, where $q < n$. Then a simple yet effective amplification procedure by amplifying any eigenvalue smaller than Δ to Δ is employed. The amplified eigenvalues are now $\bar{\mathbf{k}} = (\sigma_1, ..., \sigma_s, \Delta, ..., \Delta)$, where $s \leq q$. Let the diagonalized amplified eigenvalue matrix be $\bar{\mathbf{\Lambda}}$ (w.r.t. $\mathbf{k}$), then the modified matrix with small positive determinant can be written as:

$$\bar{\mathbf{L}} = \mathbf{U}\bar{\mathbf{\Lambda}}\mathbf{U}^\top \tag{5.11}$$

Now that $\det(\bar{\mathbf{L}}) = \prod_{i=1}^{q} \sigma_i \prod_{j=q+1}^{n} \Delta > 0$. For any $\epsilon > 0$, a sufficiently small Δ such that $\det(\bar{\mathbf{L}}) < \epsilon$ can be chosen wisely. Thus the continuity of this procedure is guaranteed. The difference $\hat{\mathbf{L}} = \bar{\mathbf{L}} - \mathbf{L}$ can be viewed as a proper ascending direction w.r.t. $\mathbf{L}$, as by adding $\hat{\mathbf{L}}$, $\det(\mathbf{L} + \hat{\mathbf{L}})$ becomes above 0 as well as arbitrarily small. It is trivial to prove that $\hat{\mathbf{L}}$ is a sub-gradient on a neighbor of $\mathbf{L}$, thus $\hat{\mathbf{L}}$ is also a proper sub-gradient sufficing Definition 5.3.1. This procedure is summarized in Algorithm 1 and is termed as **DPPSG**. Intuitively, once encountering an identical or too close feature pair $\mathbf{x}_i$ and $\mathbf{x}_j$, this procedure tries to enhance the diversity by separating them apart from each other.

Algorithm 1 DPPSG

```
**Input:** $\mathbf{K}$, tolerance $\Delta$; **Output:** $\bar{\mathbf{K}}$
$\mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top \leftarrow \mathbf{K}$
$(\sigma_1, ..., \sigma_n) \leftarrow \mathrm{diag}(\mathbf{K})$
for $i$ in $\{1, ..., n\}$ do
    if $\sigma_i < \Delta$ then
        $\sigma_i \leftarrow \Delta$
    end if
end for
$\hat{\mathbf{\Lambda}} \leftarrow \mathrm{diag}(\sigma_1, ..., \sigma_n)$
$\hat{\mathbf{K}} \leftarrow \mathbf{U}\hat{\mathbf{\Lambda}}\mathbf{U}^\top$
$\bar{\mathbf{K}} \leftarrow \hat{\mathbf{K}} - \mathbf{K}$
```

Inspired by geometric inequality, an improved version of the algorithm taking into account the property of Gaussian kernel is provided in this dissertation as well. First it is easy to show that the function $\prod_i \sigma_i$ is concave in the feasible set $\sum_i \sigma_i = n$ (diagonal of Gaussian gram matrices are 1s, thus trace is n) and the maximal objective is reached out **iff**$\sigma_i = 1$. Therefore, any point $\mathbf{b} = (1-\theta)(\sigma_1, ..., \sigma_n) + \theta(1, ..., 1)$ will increase the objective $\prod_i \sigma_i$. By letting θ being a small enough value, the proper sub-gradient becomes $\mathbf{U}\mathrm{diag}(\mathbf{b} - \sigma)\mathbf{U}^\top$, where $\sigma = (\sigma_1, ..., \sigma_n)$. This version of update differs from **DPPSG** as it generates sub-gradients under geometric constraints. The method is summarized in Algorithm 2 and is termed as **DPPSG***.

Algorithm 2 DPPSG*

```
**Input:** $\mathbf{K}$, tolerance $\theta$; **Output:** $\bar{\mathbf{K}}$
$\mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top \leftarrow \mathbf{K}$
$(\sigma_1, ..., \sigma_n) \leftarrow \text{diag}(\mathbf{K})$
$\mathbf{b} = (1-\theta)(\sigma_1, ..., \sigma_n) - \theta(1, ..., 1)$
$\hat{\boldsymbol{\Lambda}} \leftarrow \text{diag}(\mathbf{b})$
$\hat{\mathbf{K}} \leftarrow \mathbf{U}\hat{\boldsymbol{\Lambda}}\mathbf{U}^\top$
$\bar{\mathbf{K}} \leftarrow \hat{\mathbf{K}} - \mathbf{K}$
```

During implementation, the irregularity of $\mathbf{L}$ is examined to determine whether to adopt a normal back-propagation (in Sec 5.3.1) or sub-gradient (in Sec 5.3.2). This can be done by verifying if the determinant value in the forward pass is less than a pre-defined small enough value β. This proper sub-gradient based back-propagation method can be used to integrate to deep learning framework with other objectives involving matrix determinant. It should be emphasized that the proposed method is different from the line of gradient-projection based methods, such as K-Ascent. While projection-based methods calculate the true gradient then project it back to a feasible set, the proposed methods *generate proper sub-gradient directly.* Without explicitly computing matrix inversion, sub-gradients, in this case, is more feasible for deep learning framework.

Mini-batch sampling a balanced sampling strategy is employed for each mini-batch. Assuming the batch size is n and there are c classes in total, in each mini-batch the distribution of samples generally follows the whole training sample distribution on c classes. This strategy is considered to utilize the intrinsic diversity of the original data. Besides, mini-batch sampling can constrain the overhead of DPP computation depending only on the batch size, which can be viewed as a constant in practice.

5.3.3 Bounding the Features with Wasserstein GAN

Practically, the features are always required to lie in a bounded space. This is essential in some applications as a bounded space is more controllable. Especially, sometimes one may demand that the features should suffice to a pre-defined distribution $\mathcal{P}$. This bounding requirement is crucial to the objective of DPP since maximizing determinant tends to draw feature points infinitely apart from each other. A naive method to achieve this is to truncate the features or using barrier functions. However, these methods will result in irregularly dense distribution on the learned feature space boundary. To overcome this issue, the Wasserstein GAN (WGAN) Arjovsky *et al.* (2017) is conducted to enforce the features mapped to a specific distribution $\mathcal{P}$. The definition of WGAN are referred to Arjovsky *et al.* (2017) and the pseudo-algorithm can be structed as shown in Algorithm 3 which is originally from Arjovsky *et al.* (2017),

To this end, m [1] points $\bar{\mathbf{x}}_i$ are randomly sampled from the distribution $\mathcal{P}$ under balanced sampling, which are treated as positive samples. The generator $f(\cdot)$ takes a feature as input and outputs the corresponding embedding. Denote the discriminator $h(\cdot)$ (which is also the mapping from input to feature). Then the WGAN loss for discriminator is:

$$L_W = \mathbb{E}_{\bar{\mathbf{x}} \sim \mathcal{P}} \left[h(\bar{\mathbf{x}}) \right] - \mathbb{E}_{\mathbf{I} \sim p(\mathbf{I})} \left[h(f(\mathbf{I})) \right] \tag{5.12}$$

[1]the m is chosen as the batch size for simplicity. One can change this value for the own purpose.

Algorithm 3 WGAN. All the default values are in the original paper is = 0.00005, $c = 0.01$, $m = 64$, $n_{critic} = 5$

```
Require: $\alpha$, the learning rate. $c$, the clipping parameter. $m$, the batch size. $n_{critic}$, the number of iterations of the critic per generator iteration.
Require: $w_0$, initial critic parameters. $\theta_0$, initial generator's parameters.
while $\theta$ has not converged do
    for $t_0, ..., n_{critic}$ do
        Sample $\{x^{(i)}\}_{i=1}^{m} \sim \mathbb{P}_r$ a batch from the real data.
        Sample $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$ a batch of prior samples.
        $g_w \leftarrow \nabla_w[\frac{1}{m}\sum_{i=1}^{m} f_w(x^{(i)}) - \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))]$
        $w \leftarrow w + \cdot RMSProp(w, g_w)$
        $w \leftarrow clip(w, -c, c)$
    end for
    Sample $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$ a batch of prior samples.
    $g_\theta \leftarrow -\nabla_\theta \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)})$
    $\theta \leftarrow \theta - \alpha \cdot RMSProp(\theta, g_\theta)$
end while
```

According to the Arjovsky *et al.* (2017), the generator loss $L_C = -\mathbb{E}[h(f(\mathbf{I}))]$ is incorporated into general loss L_G and obtain

$$L = L_G - \lambda_1 L_D + \lambda_2 L_C, \tag{5.13}$$

where $\lambda_1 > 0$ and $\lambda_2 > 0$ are controlling parameters and L_D is the DPP term. In general, the second and third losses serve as regularization. While the DPP term L_D makes the points spread out over the whole space, the WGAN term L_W enforce the points to be under a distribution $\mathcal{P}$. These two terms are set to negative as the aim is to maximize them. In the implementation, L_W and L are trained alternatively.

5.4 Experiments and Settings

In this section, two experiments are conducted. One is about metric learning and image hashing on MNIST and CIFAR to verify the effectiveness of the proposed method, while another is for local descriptor retrieval task based on HardNet (Mishchuk *et al.*, 2017). There is no video related visual dataset is utilized for the experiments since the image related dataset is conducted quickly in training and testing stage in the mean time the feature extraction procedure is similar to video feature extraction. Since the demonstration of the effectiveness of the proposed methods on visual tasks is the main purpose in this dissertation, a more clear and simple image classification and retrieval tasks are only considered in the experiments.

5.4.1 Verification Test

MNIST This simple dataset is suitable to reveal the geometric properties of the features on various tasks. The image retrieval task equipped with contrastive loss $L_C = \sum_{\mathcal{L}(i)=\mathcal{L}(j)} (\mathbf{x}_i - \mathbf{x}_j)^2 + \alpha \sum_{\mathcal{L}(i)\neq\mathcal{L}(j)} \max(\mu - (\mathbf{x}_i - \mathbf{x}_j)^2, 0)$ under Gaussian DPP regularization is tested, where $\mathcal{L}(i)$ indicates the label of the ith feature and $\mathbf{x}_i$ is the learnt feature. A simple network structure is employed for MNIST. This network consists of 3 convolutional layers (Conv) followed by 2 fully connected layers (FC). Batch normalization (Ioffe and Szegedy, 2015) is applied on each layer. The number of filters of each Conv are 32, 32 and 64, respectively. The sizes of the filters are identically 5×5. For the first Conv, a maxpooling layer is applied with. For the other 2 Convs, the average pooling layer is adopted. The dimensions of the last FCs are 200 and 2 (for 2D visualization).

The detail of the structure of the employed backbone is {conv_1(5×5)+maxpool +conv_2(5×5)+avepool+conv_3(5×5)+avepool+fully_con1(200-d)+fully_con2(2-d)+

fully_con3(10-d)+contrastive_loss} . The DPP and WGAN regularization is added to the features at "fully_con2" layer, which is 2-dimensional thus better for visualization.

The parameters for the MNIST experiment are set as follows: $\alpha = 5$, $\lambda_1 = 10^3$, $\lambda_2 = 10^6$, margin $\mu = 0.8$, variance for Gaussian kernel $\sigma = 0.2$ and $\Delta = 10^{-7}$. During the training, the batch size is set to 200. In each iteration of DPP and WGAN training, $2,000$ adversarial points are sampled uniformily from the space $[-1, 1]^2$. The RMSprop is adopted and the learning rate is 10^{-4} for all tests. In the testing stage, $2,000$ points are sampled from $[-1, 1]^2$ and calculate the Wasserstein distance with all the testing samples. This procedure is conducted 10 times and the mean distance is reported.

Table 5.1: Retrieval Performance on MNIST.

	mAP-k(%)			
k	10	20	50	100
baseline	63.90	62.87	61.43	58.65
DPPSG	67.22	67.45	65.82	62.78
DPPSG*	67.94	68.73	66.32	62.75
DPPSG+WGAN	68.07	69.34	66.19	63.40
DPPSG*+WGAN	69.14	70.32	68.04	64.58

The performance can be found in Table 5.1 and the feature distribution is visualized in Fig. 5.1. In Fig. 5.1, left and right of each sub-image represents the training and testing samples, respectively. The (a) sub-image is the contrastive loss for metric learning; The (b) sub-image is the contrastive loss + DPP regularization; The (c) sub-image is contrastive loss + DPP regularization + WGAN regularization. For (c), the features are generally lying in the space $[-1, 1]^2$. From Table 5.1, it is observed that the performance on retrieval task can be enhanced by adding the DPP and WGAN regularization terms. It is shown that DPP term can enhance the retrieval performance by avoiding feature points from concentrating too much. In this sense, the learned map around the separating boundary can be much smoother. As retrieval task typically requires the existence of top-k inter-class samples rather than concentrating property, the DPP term is more preferable. In Fig. 5.1(c), it demonstrates that the feature points generally fall into the pre-defined space $[-1, 1]^2$. The utility of such space is high without sacrificing the retrieval performance. Typically, DPPSG* is slightly superior to DPPSG. Thus in the following testing, only the performance under DPPSG* setting (termed as DPP* for short) is reported.

CIFAR–10 image hashing The image hashing experiment is conducted on CIFAR–10 which seeks to produce binary code for images. To this end, the binary hashing code generation procedure in Lin *et al.* (2015) which is activated by a Sigmoid function is applied in this section. The number of neurons in the second last layer equals to the number of bits of the hashing codes. It is anticipated that DPP regularization can enhance the utility in binary code space since the code can spread out [2] . Two lengths of binary code (12 and 16) are tested for evaluation. And the 16–bit feature distribution using TSNE (Maaten and Hinton, 2008) is visualized in Fig. 5.2 (a) and (b), and the binary code histogram comparison in Fig. 5.3. In Fig. 5.2, compared with (b), (a) is more uniformly distributed in the feature space. The quantitative results ("acc" in the Table is the classification accuracy.) are summarized in Table 5.2. As Lin *et al.* (2015) jointly solve binary code generation and classification, both retrieval performance (mAP) and classification performance (Acc) are reported in this experiment. It is shown that the proposed method can significantly enhance the binary space utility while keeping the performance almost intact.

[2]Higher binary code space utility can enhance the hashing speed and save the storage.

The same network structure as a high-cited method DCH (Lin *et al.*, 2015)is conducted for this experiment. DPP and WGAN loss is applied on the second last fully connected layer (the dimension of this layer corresponds to the length of digits in the hashing code).

The parameters in the hashing related experiments are used as following: variance for Gaussian Kernel $\sigma = 2$, the coefficients for the loss term of DPP is $\lambda_1 = 10^2$ and for the loss term of discriminator and generator in WGAN is 10 and 1 respectively. The batch size is set to 500 and the learning rate is initialized to 0.01 with a changing rate of 0.1 at every 150 epoch. The total number of epoch is set to 350 and the Adam optimizer is adopted to update the proposed model.

Degradation on CIFAR-10 image hashing For the performance degradation with DPP on hashing task, The Fig. 5.3 can be taken as an example to explain. In the figure, the histograms up and down correspond to DPP+WGAN and Lin *et al.* (2015), respectively. It is shown that original DCH features concentrate on several digits (generally 10 digits corresponding to 10 classes), while DPP features diffuse to almost the whole discrete space. In this sense, if one retrieves the k-th closest hashing code, DCH can find the hashing code with a small searching radius. However, one has to greatly enlarge the search radius for k-th closest code in DPP feature space since the distribution is much more even. In this sense, DPP will inevitably causes degradation since large searching radius will more likely to reach a code in other class. Therefore, it is naturally to think "utility vs mAP" is an intrinsic conflict and needs to reach a trade-off.

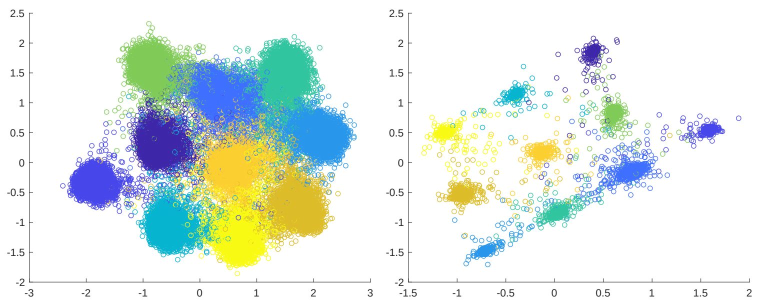
Table 5.2: Image Hashing on CIFAR–10.

	mAP-k (%)			Acc
k	50	100	all	
12–bit				
DCH	82.9	83.9	85.9	83.5
DPP	81.7	81.9	81.7	89.9
16–bit				
DCH	84.9	85.4	86.7	92.0
DPP	83.9	83.7	82.9	91.5

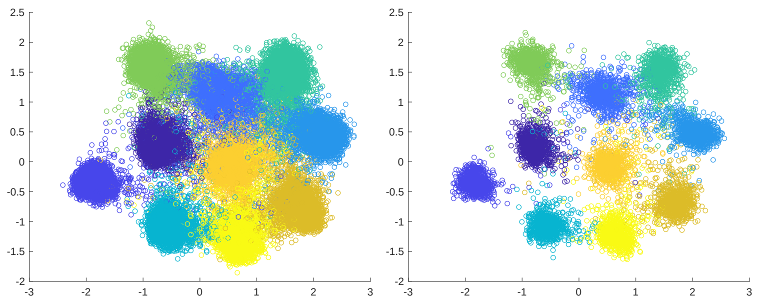
CIFAR–100 metric learning All the convolutional layers in VGG-19 (Simonyan and Zisserman, 2014b) are employed as the base and discard its final fully connected layers. Thus the output size of this base VGG-19 network is $1 \times 1 \times 512$. Then 3 fully connected layers with ReLU activation are concatenated on each after that with dimensions 512, 100 and 20, respectively. Contrastive loss is applied on the 20-dimensional space. The DPP and WGAN loss, together with contrastive loss, is applied on the final fully connected layer (20-dimension). The network is trained from scratch without any pre-training.

The parameter setting of CIFAR–100 metric learning is as follows: $\alpha = 1$, $\lambda_1 = 10^3$, $\lambda_2 = 10^3$, margin $\mu = 0.8$, variance for Gaussian kernel $\sigma = 0.2$ and $\Delta = 10^{-6}$. The rest of the settings are the same as those of MNIST test.

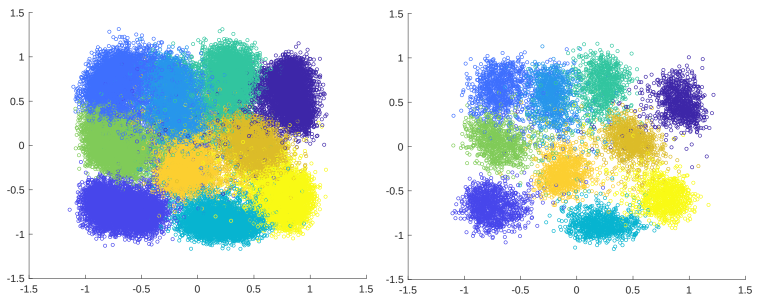
For image retrieval task, the top-k mean average precision (abbreviated as mAP-k) is adopted to evaluate the performance. The top-k average precision (abbreviated as Precision-k) is also presented, which is calculated as:



(a)



(b)



(c)

Figure 5.1: Feature Distribution of Mnist Dataset under Different Settings.

$$\text{Precision}(b_j)@K = \frac{\sum_{i=1}^{K} \mathbb{I}(b_j) P(b_j)@i}{\sum_{i=1}^{K} \mathbb{I}(b_j)} \tag{5.14}$$

where b is the corresponding class and $\mathbb{I}$ is the indicator function:

$$\mathbb{I}(b) = \begin{cases} 1 & \text{if b is a true positive} \\ 0 & \text{if b is a false positive} \end{cases} \tag{5.15}$$

Thus mAP-k is the reweighted version of Precision-k:

$$\text{mAP}(b_j) = \frac{\sum_{j=1}^{N} \text{Precision}(b_j)@K}{N} \tag{5.16}$$

Aside from mAP, the top-k average precision (Precision-k) and the Wasserstein distance to the pre-defined distribution (Gap to $\mathcal{P}$) are reported as well. The performance on coarse (20 classes) and fine (100 classes) levels can be found in Table 5.3. In either setting, it is demonstrated that DPP+WGAN significantly outperform the baseline. Thus it can be inferred that the DPP term can serve as a regularization not only for the feature itself but also for the smoothness of the mapping. Since the DPP term avoids the features from concentrating too much, the learned mapping should also be from a smoother function family.

Table 5.3: Metric Learning Performance on CIFAR–100 Dataset with Course (20) and Fine (100) Classes.

	mAP-k (%)			Precision-k (%)			Gap to $\mathcal{P}$
k	10	20	50	10	20	50	
On coarse (20) classes							
Baseline	6.98	6.74	6.82	9.44	9.36	9.35	–
DPP*	45.35	48.09	48.74	55.62	53.04	51.08	–
DPPW*	47.30	52.18	54.37	60.60	58.44	57.39	0.046
On fine (100) classes							
Baseline	17.98	18.24	18.06	23.21	23.47	22.76	–
DPP*	28.43	28.49	28.37	35.18	34.79	33.26	–
DPPW*	30.50	31.28	32.49	40.15	40.76	38.36	0.032

Batch size VS. performance The influences of batch size on the performance with DPP regularization is also studied in this dissertation. To this end, the performance on CIFAR-100 100-class retrieval with different batch sizes is reported. The results are shown in Table 5.4. Generally, with larger batch size, the algorithm can reach out better mAP. It is notified that the computational efficiency of DPP sub-gradients is high, which adds very slight overhead (even with 500 batch size) to each iteration of common back-propagation under contrastive loss, which can be neglected.

Table 5.4: Impact of Batch Size (B-size) in CIFAR-100 100 Classes.

	mAP-k(%)		
b-size	10	20	50
200	30.50	31.28	32.49
300	30.78	32.27	32.29
400	31.44	33.46	33.51
500	33.97	34.49	35.38

5.4.2 Local Descriptor Retrieval

This evaluation utilizes the UBC Phototour dataset (Brown and Lowe, 2007), which consists of three subsets (Liberty, Notre Dame, and Yosemite) with around 400k 64×64 local patches for each. This experiment follows the protocol in Mishchuk *et al.* (2017) to treat two subsets as the training set and the third one as the testing set. As each pair of matched image patches includes only two patches, there is no need to apply balanced sampling in this test. The DPP regularization term is simply added to the objective of state-of-the-art algorithm HardNet (Mishchuk *et al.*, 2017). The batch size is 512. The FPR (false positive rate) and FDR (false discovery rate) following Mishchuk *et al.* (2017); Han *et al.* (2015) is reported in this section. Results are summarized in Table 5.5. In the Table 5.5, Notre, Yose and Lib are short for “Notre Dame”, “Yosemite” and “Liberty”, respectively. Following HardNet

Mishchuk *et al.* (2017), the FPR at true positive rate is reported at 95%. The best results are in **bold**. Several baselines are selected for comparison (i.e. SIFT (Lowe, 1999), MatchNet (Han *et al.*, 2015), TFeat-M (Balntas *et al.*, 2016), L2Net (Tian *et al.*, 2017) and HardNet (Mishchuk *et al.*, 2017)). The latest version (termed as HardNet+) is also compared with the proposed method following the update of the (Mishchuk *et al.*, 2017). And the proposed method is conducted only under DPPSG* setting and name the proposed method HardDPP. It is demonstrated that with DPP regularization, the performance of HardNet can be further enhanced. Note that in HardNet there is no WGAN integrated as the mapped features lie in the surface of a hyper unit sphere. While the sampling strategy of HardNet emphasizes the embedding behavior near the margin, DPP regularization can further focus on global feature distribution.

Table 5.5: Performance of UBC Phototour Comparison.

Training	Notre + Yose		Lib + Yose		Lib + Notre		Mean	
Testing	Lib		Notre		Yose		FDR	FPR
SIFT	29.84		22.53		27.29			26.55
MatchNet	7.04	11.47	3.82	5.65	11.6	8.7	7.74	8.05
TFeatM	7.39	10.31	3.06	3.8	8.06	7.24	6.47	6.64
PCW	7.44	9.84	3.48	3.54	6.56	5.02		5.98
L2Net	3.64	5.29	1.15	1.62	4.43	3.3		3.24
HardNet	3.06	4.27	0.96	1.4	3.04	2.53	3.0	2.54
HardNet+	1.47	2.67	0.62	0.88	2.14	1.65		1.57
HardDPP	**1.21**	**2.17**	**0.58**	**0.70**	**1.79**	**1.32**	**1.31**	**1.17**

5.4.3 Overhead of Determinatal Point Process

Calculating SVD or matrix inversion on a large number of features can be time consuming. However, in the proposed setting, a common practice in deep learning – mini-batch – is employed to avoid such computation on a whole batch. It can be concluded that mini-batch strategy can limit the computational cost such that the extra overhead of DPP is only dependent on the batch size (thus other parts of the networks have no impact on this overhead). Therefore, although the complexity of the proposed method is $\mathcal{O}(n^3)$, n only corresponds to the batch size rather than whole sample number in the proposed setting, which is much more manageable in practice. The average overhead comparison on CIFAR-10 hashing task with varying batch sizes (100, 200, 250, 400 and 500) on a GTX 1080 GPU as in Table 5.6 (time in seconds) is reported as:

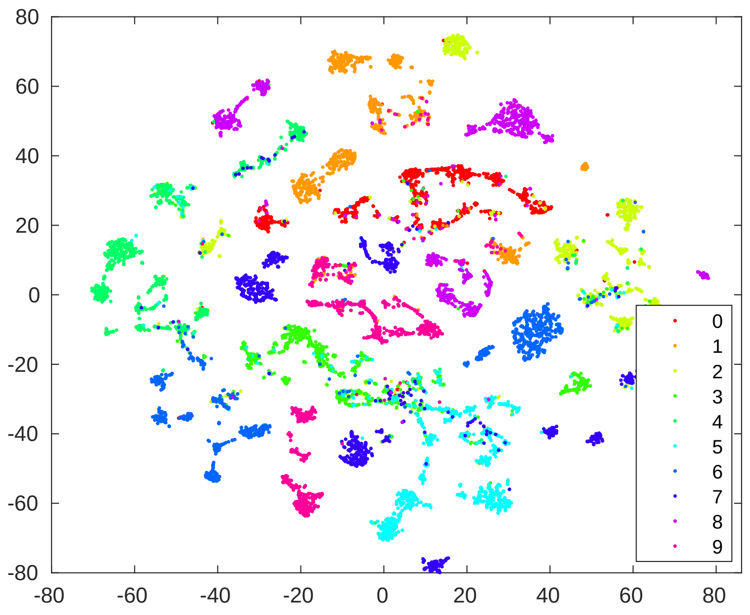
Table 5.6: Overhead of a Single Batch and a DPP Calculation on CIFAR-10 Hashing Task with Varying Batch Size. Time Is in Seconds.

batch size	100	200	250	400	500
overhead-all	0.175	0.263	0.308	0.411	0.493
overhead-DPP	0.011	0.029	0.033	0.042	0.056

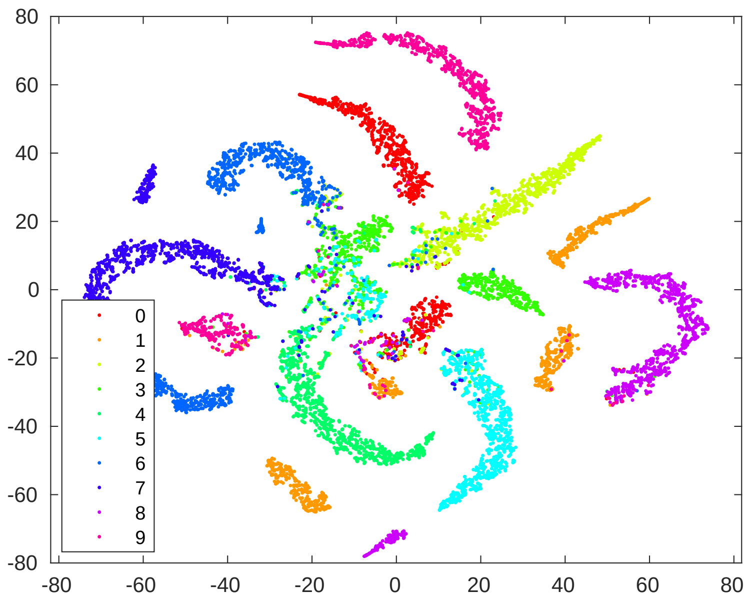
where "overhead-all" and "overhead-DPP" refer to the average time cost (s) for a single batch on all the computation and only DPP computation (both forward and backward), respectively. It can be concluded as that, compared to other computation, the extra overhead of DPP is small (even in a simple network as CIFAR-10 hashing). Besides, a batch size up to 500 is considered to be sufficient in most of the applications. In practice, any trick is not employed to reduce such overhead (since it is out of the main focus) but simply utilized standard functions provided by PyTorch.

5.5 Conclusion

In this chapter, the problem of learning diverse features via a determinantal point process under deep learning framework is investigated. To overcome the instability in computing the gradient which involves the matrix inverse, an efficient and reliable procedure called proper spectral sub-gradient generation is proposed. The generated proper sub-gradient can replace the true gradient and performs well in applications. And how to constrain the features into a bounded space is also considered in this chapter, since in such a way one can ensure the behavior of the network more predictable. To this end, the Wasserstein GAN is further incorporated into the developed framework. Together, DPP+WGAN showed significant performance on both some common criteria and feature space utility.



(a) DPP+WGAN



(b) Lin *et al.* (2015)

Figure 5.2: Visualization of 16–bit Hashing Code Results on CIFAR–10.

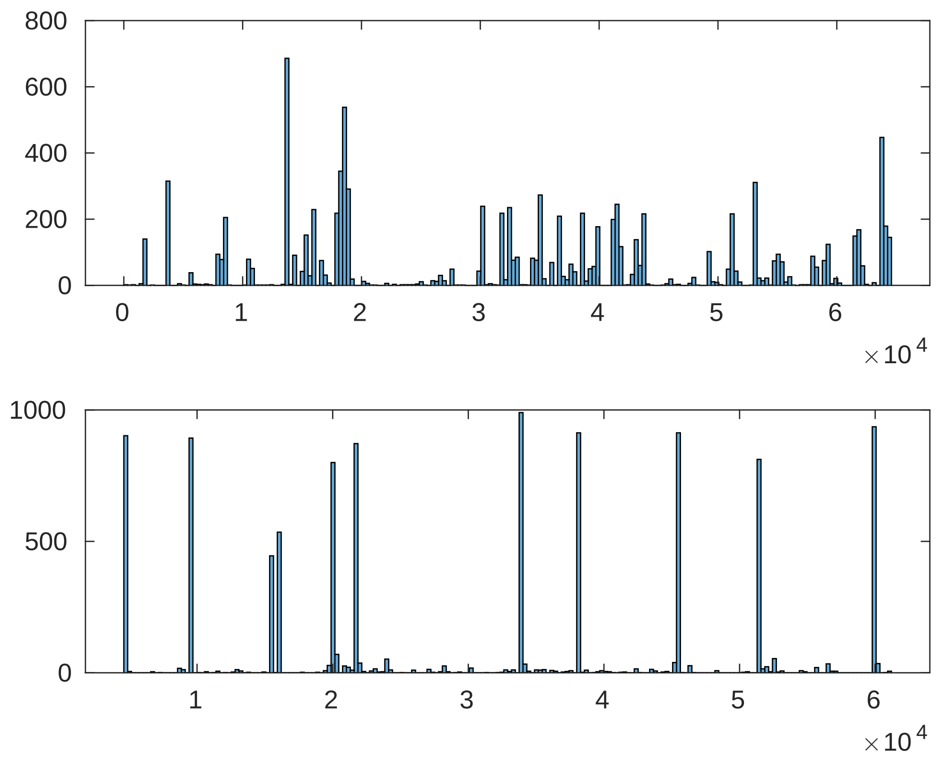


Figure 5.3: Binary Code Histogram.

Chapter 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

In this dissertation, three different kinds of deep neural network structures are proposed for extracting video representative features. The essentials of obtaining video representative features concentrate on acquiring temporal information along with appearance information. Even though the informative spatial features may be enough for easy action recognition for some simple video clips (i.e. some simple actions like "waving hands" with no scenes changing in KTH dataset Schuldt *et al.* (2004)), the incorporation of temporal information may boost the performance on several visual tasks. Two main methods of temporal information acquisition have been utilized in this dissertation. One is leveraging the Convolutional Neural Networks (CNNs) to obtain temporal information from the optical flow maps which are built by warping the motion vectors into a 2D matrix Brox *et al.* (2004). The CNNs treat the optical flow maps as images and obtain the temporal information by convolving computation. Another method is utilizing the Recurrent Neural Network (RNN) to encode the order information of spatial information. The RNN can obtain temporal information by learning the sequential order of the frame feature vectors.

The proposed methods in this dissertation demonstrate the capability and superiority of deep learning based structures in acquiring spatio-temporal information by conducting several different visual tasks. The Zero-Shot Learning (ZSL) problem and the semantic video retrieval, which are presented in **Chapter 2**, shows the proposed two-stream based deep neural networks have the capability of elaborating the semantic embedding space with the extracted spatio-temporal features to form a representative feature which contains high-level human semantic meaning. However, the two-stream based deep models are weak at dealing with very long videos and acquiring sub-actions information. The experiments of simultaneous event recognition and localization task mentioned in **Chapter 3** demonstrate the two-stream based ConvRNN structure can generate heatmaps for categorizing events and locating the corresponding time period by a weakly-supervised way. This proposed two-stream based ConvRNN method is better at acquiring sub-actions information compared with the proposed methods in **Chapter 2**. But the gradient vanishing/explosion problem still heavily impacts the RNN related methods. Therefore, a newly RNN based structure called RhyRNN is proposed in **Chapter 4** which will greatly mitigate the gradient vanishing/explosion problem. Moreover, by implementing a diversity-driven sequential pooling layer and a bilinear reweighting layer with the RhyRNN, the proposed method is capable of handling very long video clips (i.e. number of sequences is larger than 3000). The experiments in **Chapter 4** demonstrates that the proposed method can extract effective representative features from a very long and complex video with only event-level label information. The Multi-Label classification experiment in **Chapter 4** also shows the capability of the proposed method in obtaining structural sub-action information even without any sub-action level label knowledge. Even though all the proposed methods in **Chapter 2, 3, 4** can complete the classification/recognition or localization tasks very well, some visual tasks, like semantic retrieval, learning hashing code, or graph knowledge mapping, require a diverse feature space. Therefore, a method, which incorporates the Determinantal Point Process (DPP) with the deep learning methods, is proposed to learn a diverse feature space. A newly designed strategy of gradient computation for the determinant matrix makes the incorporation of the Determinantal Point Process with deep learning more convenient and efficient. Moreover, the experiments in **Chapter 5** demonstrate the efficiency and capability of obtaining a more diverse feature space for better performance in visual tasks like hashing and semantic retrieval.

In summary, all the proposed methods in this dissertation present the core idea of obtaining video representative features by conducting deep neural networks to extract temporal information and collaborating the corresponding spatial information. The newly formed video representative features are shown to be useful and effective in several visual tasks by all the experiments mentioned in this dissertation.

6.2 Future Work

Since the temporal information is more and more important in the understanding of the recent video data, a better way of extracting temporal information is the future direction of the related research. Similarly, the technique in Natural Language Processing (NLP) requires the capability of obtaining the order information of texts or languages. Therefore, it is natural to link the technique of NLP with video processing. Recently, the BERT Devlin *et al.* (2018) and Multi-Head Self Attention Vaswani *et al.* (2017) becomes more and more popular in the NLP field since it depends on CNN-based attention mechanism so that the gradient vanishing problem is no longer considerable. Therefore, applying the BERT or Multi-Head Self Attention on extracting the order/sequential information is a good way to overcome the challenges in obtaining video representative features.

Considering the challenging problems about the visual tasks in very long and complex video data raised in **Chapter 5**, another possible method is to acquire the latent information (i.e. the sub-actions) should be a new future direction. The conventional topic model is a good way to extract latent information for documents. It is able to summarize the documents with topic distributions. Therefore, it is easy to think about utilizing the topic model to extract the topic distribution which can be viewed as the sub-action distribution for each video. However, the difficulty in the topic model is also obvious. The fixed number of topics is not flexible compared with current methods. Therefore, the way to alleviate the impact of the fixed number of topics is an essential part of developing a topic model based video processing method.

Because of the structure of the video sequential data, it is natural to think that the video can be summarized with several thumbnail frames or the most informative frames. According to the current research of the video summarization algorithm, the thumbnail frames or the most informative frames make the most contribution to the final event or activity recognition. However, sometimes the selected thumbnail frames are hard to capture the potential sequential information for a better understanding of the complex events. Therefore, the graph-based method should be a good way to enhance the potential sequential information or the correlation of the selected frames. The Graph Convolutional Network (GCN) can be applied to the thumbnail frames if a good adjacent matrix could be obtained or defined before the thumbnail frames selection. And as has been demonstrated in many visual papers, the GCN is good at learning a better knowledge map for the semantic embedding of images which can be extended to the video processing area. Thus obtaining better representative features for understanding video contents.

REFERENCES

Affandi, R. H., E. Fox, R. Adams and B. Taskar, “Learning the parameters of determinantal point process kernels”, in “ICML”, (2014).

Affandi, R. H., E. Fox and B. Taskar, “Approximate inference in continuous determinantal processes”, in “NIPS”, (2013).

Antol, S., A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. Lawrence Zitnick and D. Parikh, “Vqa: Visual question answering”, in “ICCV”, (2015).

Arjovsky, M., S. Chintala and L. Bottou, “Wasserstein gan”, arXiv preprint arXiv:1701.07875 (2017).

Arjovsky, M., A. Shah and Y. Bengio, “Unitary evolution recurrent neural networks”, in “ICML”, (2016).

Baccouche, M., F. Mamalet, C. Wolf, C. Garcia and A. Baskurt, “Action classification in soccer videos with long short-term memory recurrent neural networks”, in “International Conference on Artificial Neural Networks”, pp. 154–159 (Springer, 2010).

Badrinarayanan, V., A. Kendall and R. Cipolla, “Segnet: A deep convolutional encoder-decoder architecture for image segmentation”, PAMI (2017).

Balntas, V., E. Riba, D. Ponsa and K. Mikolajczyk, “Learning local feature descriptors with triplets and shallow convolutional neural networks.”, in “BMVC”, (2016).

Bardenet, R. and M. Titsias, “Inference for determinantal point processes without spectral knowledge”, in “NIPS”, (2015).

Bilen, H., B. Fernando, E. Gavves and A. Vedaldi, “Action recognition with dynamic image networks”, PAMI **40**, 12, 2799–2813 (2017).

Bilen, H., B. Fernando, E. Gavves, A. Vedaldi and S. Gould, “Dynamic image networks for action recognition”, in “CVPR”, (2016).

Brown, M. and D. G. Lowe, “Automatic panoramic image stitching using invariant features”, IJCV **74**, 1, 59–73 (2007).

Brox, T., A. Bruhn, N. Papenberg and J. Weickert, “High accuracy optical flow estimation based on a theory for warping”, in “European Conference on Computer Vision (ECCV)”, vol. 3024 of *Lecture Notes in Computer Science*, pp. 25–36 (Springer, 2004), URL `http://lmb.informatik.uni-freiburg.de//Publications/2004/Bro04a`.

Campos, V., B. Jou, X. Giró-i Nieto, J. Torres and S.-F. Chang, “Skip rnn: Learning to skip state updates in recurrent neural networks”, arXiv preprint arXiv:1708.06834 (2017).

Carreira, J. and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset", in "proceedings of the IEEE Conference on Computer Vision and Pattern Recognition", pp. 6299–6308 (2017).

Carreira-Perpinán, M. A. and R. Raziperchikolaei, "An ensemble diversity approach to supervised binary hashing", in "NIPS", (2016).

Chang, X., Y.-L. Yu, Y. Yang and E. P. Xing, "Semantic pooling for complex event analysis in untrimmed videos", PAMI **39**, 8, 1617–1632 (2017).

Chao, W.-L., B. Gong, K. Grauman and F. Sha, "Large-margin determinantal point processes.", in "UAI", (2015).

Chao, Y.-W., S. Vijayanarasimhan, B. Seybold, D. A. Ross, J. Deng and R. Sukthankar, "Rethinking the faster r-cnn architecture for temporal action localization", in "CVPR", (2018).

Chatfield, K., K. Simonyan, A. Vedaldi and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets", CoRR **abs/1405.3531**, URL `http://arxiv.org/abs/1405.3531` (2014).

Chen, Y., H. Zhang, Y. Tong and M. Lu, "Diversity regularized latent semantic match for hashing", Neurocomputing **230**, 77–87 (2017).

Cheng, H.-T., M. Griss, P. Davis, J. Li and D. You, "Towards zero-shot learning for human activity recognition using semantic attribute sequence model", in "Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing", UbiComp '13, pp. 355–358 (ACM, New York, NY, USA, 2013), URL `http://doi.acm.org/10.1145/2493432.2493511`.

Chéron, G., I. Laptev and C. Schmid, "P-CNN: Pose-based CNN Features for Action Recognition", in "ICCV 2015 - IEEE International Conference on Computer Vision", (Santiago, Chile, 2015), URL `https://hal.inria.fr/hal-01187690`.

Cho, K., B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation", arXiv preprint arXiv:1406.1078 (2014).

Chollet, F., "Xception: Deep learning with depthwise separable convolutions", in "CVPR", (2017).

Chung, J., C. Gulcehre, K. Cho and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling", arXiv preprint arXiv:1412.3555 (2014).

Deng, J., W. Dong, R. Socher, L.-J. Li, K. Li and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database", in "CVPR09", (2009).

Devlin, J., M.-W. Chang, K. Lee and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding", arXiv preprint arXiv:1810.04805 (2018).

Dinu, G. and M. Baroni, “Improving zero-shot learning by mitigating the hubness problem”, CoRR **abs/1412.6568**, URL `http://arxiv.org/abs/1412.6568` (2014).

Donahue, J., L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko and T. Darrell, “Long-term recurrent convolutional networks for visual recognition and description”, in “CVPR”, (2015).

Du, Y., W. Wang and L. Wang, “Hierarchical recurrent neural network for skeleton based action recognition”, in “CVPR”, (2015).

Duan, L., D. Xu, I. W.-H. Tsang and J. Luo, “Visual event recognition in videos by learning from web data”, PAMI **34**, 9, 1667–1680 (2012).

Elfeki, M., C. Couprie, M. Riviere and M. Elhoseiny, “Gdpp: Learning diverse generations using determinantal point process”, in “ICML”, (2019).

Fayyaz, M., M. H. Saffar, M. Sabokrou, M. Fathy, R. Klette and F. Huang, “Stfcn: Spatio-temporal fcn for semantic video segmentation”, arXiv preprint arXiv:1608.05971 (2016).

Feichtenhofer, C., H. Fan, J. Malik and K. He, “Slowfast networks for video recognition”, in “ICCV”, (2019).

Feichtenhofer, C., A. Pinz and A. Zisserman, “Convolutional two-stream network fusion for video action recognition”, in “CVPR”, (2016).

Fernando, B., E. Gavves, J. Oramas, A. Ghodrati and T. Tuytelaars, “Rank pooling for action recognition”, PAMI **39**, 4, 773–787 (2016).

Fernando, B., E. Gavves, J. M. Oramas, A. Ghodrati and T. Tuytelaars, “Modeling video evolution for action recognition”, in “CVPR”, pp. 5378–5387 (2015).

Fernando, B., C. Tan and H. Bilen, “Weakly supervised gaussian networks for action detection”, in “WACV”, (2020).

Friedman, J., T. Hastie and R. Tibshirani, *The elements of statistical learning*, vol. 1 (Springer series in statistics New York, 2001).

Frome, A., G. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato and T. Mikolov, “Devise: A deep visual-semantic embedding model”, in “Advances In Neural Information Processing Systems, NIPS”, (2013).

Gan, C., M. Lin, Y. Yang, Y. Zhuang and A. G.Hauptmann, “Exploring semantic inter-class relationships (sir) for zero-shot action recognition”, (2015).

Gartrell, M., U. Paquet and N. Koenigstein, “Low-rank factorization of determinantal point processes”, in “AAAI”, (2017).

Gers, F. A., J. Schmidhuber and F. Cummins, “Learning to forget: Continual prediction with lstm”, (1999).

Gillenwater, J., A. Kulesza and B. Taskar, "Near-optimal map inference for determinantal point processes", in "NIPS", (2012).

Gillenwater, J. A., A. Kulesza, E. Fox and B. Taskar, "Expectation-maximization for learning determinantal point processes", in "NIPS", (2014).

Girdhar, R., J. Carreira, C. Doersch and A. Zisserman, "Video action transformer network", in "Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition", pp. 244–253 (2019).

Girdhar, R. and D. Ramanan, "Attentional pooling for action recognition", in "NIPS", (2017).

Gitte, M., H. Bawaskar, S. Sethi and A. Shinde, "Content based video retrieval system", Int J Res Eng Technol **3**, 6 (2014).

Gkioxari, G. and J. Malik, "Finding action tubes", in "Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition", pp. 759–768 (2015).

Gong, B., W.-L. Chao, K. Grauman and F. Sha, "Diverse sequential subset selection for supervised video summarization", in "NIPS", (2014).

Han, X., T. Leung, Y. Jia, R. Sukthankar and A. C. Berg, "Matchnet: Unifying feature and metric learning for patch-based matching", in "CVPR", (2015).

Hara, K., H. Kataoka and Y. Satoh, "Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?", in "CVPR", (2018).

He, K., X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition", in "CVPR", (2016).

Hennig, P. and R. Garnett, "Exact sampling from determinantal point processes", arXiv preprint arXiv:1609.06840 (2016).

Hochreiter, S. and J. Schmidhuber, "Long short-term memory", Neural computation **9**, 8, 1735–1780 (1997).

Hu, S.-H., Y. Li and B. Li, "Video2vec: Learning semantic spatial-temporal embeddings for video representation", in "ICPR", (2016).

Hussein, N., E. Gavves and A. W. Smeulders, "Timeception for complex action recognition", in "CVPR", (2019a).

Hussein, N., E. Gavves and A. W. Smeulders, "Videograph: Recognizing minutes-long human activities in videos", in "ICCVW", (2019b).

Ioffe, S. and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift", arXiv preprint arXiv:1502.03167 (2015).

Ji, S., W. Xu, M. Yang and K. Yu, "3d convolutional neural networks for human action recognition", PAMI **35**, 1, 221–231 (2013).

Jiang, Y.-G., S. Bhattacharya, S.-F. Chang and M. Shah, "High-level event recognition in unconstrained videos", International journal of multimedia information retrieval **2**, 2, 73–101 (2013).

Kang, B., "Fast determinantal point process sampling with application to clustering", in "NIPS", (2013).

Kanuparthi, B., D. Arpit, G. Kerg, N. R. Ke, I. Mitliagkas and Y. Bengio, "h-detach: Modifying the LSTM gradient towards better optimization", in "ICLR", (2019).

Karpathy, A., G. Toderici, S. Shetty, T. Leung, R. Sukthankar and L. Fei-Fei, "Large-scale video classification with convolutional neural networks", in "CVPR", (2014).

Kay, W., J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, A. Natsev, M. Suleyman and A. Zisserman, "The kinetics human action video dataset", ArXiv **abs/1705.06950** (2017).

Ke, Y., R. Sukthankar and M. Hebert, "Efficient visual event detection using volumetric features", in "ICCV", vol. 1, pp. 166–173 (2005).

Kim, J.-H., J. Jun and B.-T. Zhang, "Bilinear attention networks", in "NIPS", (2018).

Kläser, A., M. Marszałek and C. Schmid, "A spatio-temporal descriptor based on 3d-gradients", in "British Machine Vision Conference", pp. 995–1004 (2008), URL `http://lear.inrialpes.fr/pubs/2008/KMS08`.

Krizhevsky, A., I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", in "NIPS", (2012).

Kuehne, H., A. Arslan and T. Serre, "The language of actions: Recovering the syntax and semantics of goal-directed human activities", in "CVPR", (2014).

Kuehne, H., J. Gall and T. Serre, "An end-to-end generative framework for video segmentation and recognition", in "WACV", (2016).

Kuehne, H., H. Jhuang, E. Garrote, T. Poggio and T. Serre, "HMDB: a large video database for human motion recognition", in "ICCV", (2011).

Kulesza, A. and B. Taskar, "Structured determinantal point processes", in "NIPS", (2010).

Kulesza, A. and B. Taskar, "k-dpps: Fixed-size determinantal point processes", in "ICML", (2011a).

Kulesza, A. and B. Taskar, "Learning determinantal point processes", in "UAI", (2011b).

Kulesza, A., B. Taskar *et al.*, "Determinantal point processes for machine learning", Foundations and Trends® in Machine Learning **5**, 2–3, 123–286 (2012).

Lai, K.-T., F. X. Yu, M.-S. Chen and S.-F. Chang, "Video event detection by inferring temporal instance labels", in "CVPR", pp. 2243–2250 (2014).

Lampert, C., H. Nickisch and S. Harmeling, “Attribute-based classification for zero-shot visual object categorization”, Pattern Analysis and Machine Intelligence, IEEE Transactions on **36**, 3, 453–465 (2014).

Lan, T., Y. Zhu, A. Roshan Zamir and S. Savarese, “Action recognition by hierarchical mid-level action elements”, in “ICCV”, (2015).

Laptev and Lindeberg, “Space-time interest points”, in “Proceedings Ninth IEEE International Conference on Computer Vision”, pp. 432–439 vol.1 (2003).

Laptev, I., “On space-time interest points”, IJCV **64**, 2-3, 107–123 (2005).

Laptev, I., M. Marszalek, C. Schmid and B. Rozenfeld, “Learning realistic human actions from movies”, in “2008 IEEE Conference on Computer Vision and Pattern Recognition”, pp. 1–8 (IEEE, 2008).

Lee, Y. J., J. Ghosh and K. Grauman, “Discovering important people and objects for egocentric video summarization”, in “CVPR”, (2012).

Levitin, E. and B. Polyak, *Constrained Minimization Methods* (USSR Computational Mathematics and Mathematical Physics, 1966).

Li, H., L. Liu, F. Sun, Y. Bao and C. Liu, “Multi-level feature representations for video semantic concept detection”, Neurocomputing **172**, 64–70 (2016).

Li, S., W. Li, C. Cook, C. Zhu and Y. Gao, “Independently recurrent neural network (indrnn): Building a longer and deeper rnn”, in “CVPR”, (2018).

Lin, K., H.-F. Yang, J.-H. Hsiao and C.-S. Chen, “Deep learning of binary hash codes for fast image retrieval”, in “CVPRW”, (2015).

Liu, J., Z. Wu and F. Li, “Ranking video segments with lstm and determinantal point processes”, in “ICIP”, (2017).

Long, J., E. Shelhamer and T. Darrell, “Fully convolutional networks for semantic segmentation”, in “CVPR”, pp. 3431–3440 (2015).

Lowe, D. G., “Object recognition from local scale-invariant features”, in “ICCV”, (1999).

Maaten, L. v. d. and G. Hinton, “Visualizing data using t-sne”, JMLR **9**, Nov, 2579–2605 (2008).

Mariet, Z. and S. Sra, “Fixed-point algorithms for learning determinantal point processes”, in “ICML”, (2015).

Mikolov, T., K. Chen, G. Corrado and J. Dean, “Efficient estimation of word representations in vector space”, CoRR **abs/1301.3781**, URL `http://arxiv.org/abs/1301.3781` (2013a).

Mikolov, T., I. Sutskever, K. Chen, G. S. Corrado and J. Dean, "Distributed representations of words and phrases and their compositionality", in "Advances in Neural Information Processing Systems 26", edited by C. Burges, L. Bottou, M. Welling, Z. Ghahramani and K. Weinberger, pp. 3111–3119 (Curran Associates, Inc., 2013b).

Mishchuk, A., D. Mishkin, F. Radenovic and J. Matas, "Working hard to know your neighbor's margins: Local descriptor learning loss", in "NIPS", (2017).

Ng, J. Y., M. J. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga and G. Toderici, "Beyond short snippets: Deep networks for video classification", CoRR **abs/1503.08909**, URL `http://arxiv.org/abs/1503.08909` (2015).

Nguyen, P., T. Liu, G. Prasad and B. Han, "Weakly supervised action localization by sparse temporal pooling network", in "CVPR", (2018).

Norouzi, M., T. Mikolov, S. Bengio, Y. Singer, J. Shlens, A. Frome, G. Corrado and J. Dean, "Zero-shot learning by convex combination of semantic embeddings", CoRR **abs/1312.5650**, URL `http://arxiv.org/abs/1312.5650` (2013).

Oh, S., A. Hoogs, A. Perera, N. Cuntoor, C. Chen, J. T. Lee, S. Mukherjee, J. K. Aggarwal, H. Lee, L. Davis, E. Swears, X. Wang, Q. Ji, K. Reddy, M. Shah, C. Vondrick, H. Pirsiavash, D. Ramanan, J. Yuen, A. Torralba, B. Song, A. Fong, A. Roy-Chowdhury and M. Desai, "A large-scale benchmark dataset for event recognition in surveillance video", in "CVPR", (2011).

Oliva, A. and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope", IJCV **42**, 3, 145–175 (2001).

Paszke, A., S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga and A. Lerer, "Automatic differentiation in pytorch", (2017).

Piergiovanni, A. and M. S. Ryoo, "Learning latent super-events to detect multiple activities in videos", in "CVPR", (2018).

Rashid, M., H. Kjellström and Y. J. Lee, "Action graphs: Weakly-supervised action localization with graph convolution networks", arXiv preprint arXiv:2002.01449 (2020).

Schuldt, C., I. Laptev and B. Caputo, "Recognizing human actions: A local svm approach", in "ICPR", (2004).

Sharghi, A., A. Borji, C. Li, T. Yang and B. Gong, "Improving sequential determinantal point processes for supervised video summarization", in "ECCV", (2018).

Sharma, S., R. Kiros and R. Salakhutdinov, "Action recognition using visual attention", in "NIPS Time Series Workshop", (2015).

Shou, Z., D. Wang and S.-F. Chang, "Temporal action localization in untrimmed videos via multi-stage cnns", in "CVPR", (2016).

Siam, M., S. Valipour, M. Jagersand, N. Ray and S. Yogamani, “Convolutional gated recurrent networks for video semantic segmentation in automated driving”, in “Intelligent Transportation Systems (ITSC), International Conference on”, pp. 1–7 (2017).

Sigurdsson, G. A., S. Divvala, A. Farhadi and A. Gupta, “Asynchronous temporal fields for action recognition”, in “CVPR”, (2017).

Sigurdsson, G. A., G. Varol, X. Wang, A. Farhadi, I. Laptev and A. Gupta, “Hollywood in homes: Crowdsourcing data collection for activity understanding”, in “ECCV”, (2016).

Simonyan, K. and A. Zisserman, “Two-stream convolutional networks for action recognition in videos”, in “NIPS”, (2014a).

Simonyan, K. and A. Zisserman, “Very deep convolutional networks for large-scale image recognition”, arXiv preprint arXiv:1409.1556 (2014b).

Snoek, C. G. M. and M. Worring, “Concept-based video retrieval”, Found. Trends Inf. Retr. **2**, 4, 215–322, URL `http://dx.doi.org/10.1561/1500000014` (2009).

Soomro, K., A. R. Zamir and M. Shah, “UCF101: A dataset of 101 human actions classes from videos in the wild”, CoRR **abs/1212.0402**, URL `http://arxiv.org/abs/1212.0402` (2012).

Srivastava, N., E. Mansimov and R. Salakhutdinov, “Unsupervised learning of video representations using lstms”, CoRR **abs/1502.04681**, URL `http://arxiv.org/abs/1502.04681` (2015).

Tang, K., L. Fei-Fei and D. Koller, “Learning latent temporal structure for complex event detection”, in “CVPR”, pp. 1250–1257 (2012).

Tapaswi, M., Y. Zhu, R. Stiefelhagen, A. Torralba, R. Urtasun and S. Fidler, “Movieqa: Understanding stories in movies through question-answering”, in “CVPR”, (2016).

Tian, Y., B. Fan and F. Wu, “L2-net: Deep learning of discriminative patch descriptor in euclidean space”, in “CVPR”, (2017).

Tran, D., L. Bourdev, R. Fergus, L. Torresani and M. Paluri, “Learning spatiotemporal features with 3d convolutional networks”, in “ICCV”, (2015).

Tran, D., H. Wang, L. Torresani, J. Ray, Y. LeCun and M. Paluri, “A closer look at spatiotemporal convolutions for action recognition”, in “CVPR”, (2018).

Tran, D., J. Yuan and D. Forsyth, “Video event detection: From subvolume localization to spatiotemporal path search”, PAMI **36**, 2, 404–416 (2014).

Tran, S. D. and L. S. Davis, “Event modeling and recognition using markov logic networks”, in “ECCV”, (2008).

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser and I. Polosukhin, "Attention is all you need", in "NIPS", (2017).

Veeriah, V., N. Zhuang and G.-J. Qi, "Differential recurrent neural networks for action recognition", in "ICCV", (2015).

Veltkamp, R., H. Burkhardt and H.-P. Kriegel, *State-of-the-art in content-based image and video retrieval*, vol. 22 (Springer Science & Business Media, 2013).

Venkatesan, R. and B. Li, "Diving deeper into mentee networks", arXiv preprint arXiv:1604.08220 (2016).

Venugopalan, S., M. Rohrbach, J. Donahue, R. Mooney, T. Darrell and K. Saenko, "Sequence to sequence-video to text", in "Proceedings of the IEEE International Conference on Computer Vision", pp. 4534–4542 (2015).

Venugopalan, S., H. Xu, J. Donahue, M. Rohrbach, R. Mooney and K. Saenko, "Translating videos to natural language using deep recurrent neural networks", arXiv preprint arXiv:1412.4729 (2014).

Wang, H. and C. Schmid, "Action recognition with improved trajectories", in "IEEE International Conference on Computer Vision", (Sydney, Australia, 2013), URL `http://hal.inria.fr/hal-00873267`.

Wang, L., Y. Qiao and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors", in "CVPR", (2015).

Wang, L., Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang and L. Van Gool, "Temporal segment networks: Towards good practices for deep action recognition", in "ECCV", (2016a).

Wang, X., R. Girshick, A. Gupta and K. He, "Non-local neural networks", in "Proceedings of the IEEE conference on computer vision and pattern recognition", pp. 7794–7803 (2018).

Wang, X. and Q. Ji, "Hierarchical context modeling for video event recognition", PAMI **39**, 9, 1770–1782 (2017).

Wang, Y., J. Song, L. Wang, L. Van Gool and O. Hilliges, "Two-stream sr-cnns for action recognition in videos.", in "BMVC", (2016b).

Wang, Y., S. Wang, J. Tang, N. O'Hare, Y. Chang and B. Li, "Hierarchical attention network for action recognition in videos", arXiv preprint arXiv:1607.06416 (2016c).

Wang, Z. and B. Li, "Human activity encoding and recognition using low-level visual features.", in "Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)", pp. 1876–1883 (2009).

Wu, C.-Y., C. Feichtenhofer, H. Fan, K. He, P. Krahenbuhl and R. Girshick, "Long-term feature banks for detailed video understanding", in "CVPR", (2019).

Wu, Y., S. Zhang, Y. Zhang, Y. Bengio and R. R. Salakhutdinov, “On multiplicative integration with recurrent neural networks”, in “NIPS”, (2016).

Xie, P., R. Salakhutdinov, L. Mou and E. P. Xing, “Deep determinantal point process for large-scale multi-label classification”, in “ICCV”, (2017).

Xie, S., C. Sun, J. Huang, Z. Tu and K. Murphy, “Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification”, in “ECCV”, (2018).

Xingjian, S., Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong and W.-c. Woo, “Convolutional lstm network: A machine learning approach for precipitation nowcasting”, in “NIPS”, pp. 802–810 (2015).

Xu, S., Y. Cheng, K. Gu, Y. Yang, S. Chang and P. Zhou, “Jointly attentive spatial-temporal pooling networks for video-based person re-identification”, in “ICCV”, (2017).

Xu, X., T. Hospedales and S. Gong, “Semantic embedding space for zero-shot action recognition”, in “Image Processing (ICIP), 2015 IEEE International Conference on”, pp. 63–67 (2015a).

Xu, Y., C. Zhang, Z. Cheng, J. Xie, Y. Niu, S. Pu and F. Wu, “Segregated temporal assembly recurrent networks for weakly supervised multiple action detection”, in “AAAI”, (2019).

Xu, Z., Y. Yang and A. G. Hauptmann, “A discriminative cnn video representation for event detection”, in “CVPR”, (2015b).

Yeung, S., O. Russakovsky, N. Jin, M. Andriluka, G. Mori and L. Fei-Fei, “Every moment counts: Dense detailed labeling of actions in complex videos”, IJCV **126**, 2-4, 375–389 (2018).

Yeung, S., O. Russakovsky, G. Mori and L. Fei-Fei, “End-to-end learning of action detection from frame glimpses in videos”, in “CVPR”, (2016).

Zadeh, S. A., M. Ghadiri, V. S. Mirrokni and M. Zadimoghaddam, “Scalable feature selection via distributed diversity maximization.”, in “AAAI”, (2017).

Zhang, K., W.-L. Chao, F. Sha and K. Grauman, “Video summarization with long short-term memory”, in “ECCV”, (2016).

Zhang, X., X. Y. Felix, S. Kumar and S.-F. Chang, “Learning spread-out local feature descriptors.”, in “ICCV”, (2017).

Zhou, B., A. Andonian, A. Oliva and A. Torralba, “Temporal relational reasoning in videos”, in “ECCV”, (2018).

APPENDIX A

RELATED PUBLICATION

- **Yikang Li**, S. Hu, B. Li, ”Recognizing Unseen Action in a Domain-Adaptive Embedding Space”, IEEE International Conference on Image Processing (ICIP) 2016

- **Yikang Li**, S. Hu, B. Li, ”Video2Vec: Learning Semantic Spatio-Temporal Embeddings for Video Representation”, IEEE International Conference on Pattern Recognition (ICPR) 2016

- **Yikang Li***, T. Yu*, B. Li, ”Simultaneous Event Localization and Recognition for Surveillance Video”, IEEE International Conference on Advanced Video and Signal-based Surveillance (AVSS) 2018 Oral.

- **Yikang Li***, T. Yu*, B. Li, ”RhyRNN: Rhythmic RNN for Recognizing Events in Long and Complex Videos”, European Conference on Computer Vision 2020.

- T. Yu, **Yikang Li**, B. Li, ”Learning Diverse Features via Determinantal Point Process”, IEEE International Conference on Learning Representations 2020.

This dissertation is based on the papers listed above. Furthermore, in the processing of pursuing my PhD degree, I also completed other manuscripts or papers, which were related to the general effort of my research but were not included in the dissertation. Those papers will be listed as following:

- **Yikang Li**, T. Yu, B. Li, ”Recognizing Video Events with Varying Rhythm”, arXiv:2001.05060.

- Y. Zha, **Yikang Li**, T. Yu, S. Kambhampati, B. Li, ”Plan-Recognition-Driven Attention Modeling for Visual Recognition”, Workshop on Plan, Activity, and Intent Recognition at AAAI 2019.

- Y. Zha, **Yikang Li**, S. Gopalakrishnan, B. Li, ”Recognizing Plans by Learning Embeddings from Observed Action Distributions”, International Conference on Autonomous Agents and Multiagent Systems (AAMAS) 2018

- **Yikang Li***, P.L.K. Ding*, B. Li, ”Mean Local Group Average Precision (mL-GAP): A New Performance Metric for Hashing-based Retrieval”, arXiv preprint arXiv:1811.09763

- **Yikang Li***, P.L.K. Ding*, B. Li, ”Training Neural Networks by Using Power Linear Units (PoLUs)”, arXiv preprint arXiv:1802.00212

- P.S. Chandakkar, **Yikang Li**, P.L.K. Ding, B. Li, ”Strategies for Re-Training a Pruned Neural Network in an Edge Computing Paradigm”, IEEE International Conference on Edge Computing (EDGE) 2017

- Z. Tu, J. Cao, **Yikang Li**, B. Li, ”MSR-CNN: Applying Motion Salient Region Based Descriptors for Action Recognition”, IEEE International Conference on Pattern Recognition (ICPR) 2016