# Multi-person video understanding with deep neural networks

by

**Srikanth Muralidharan**

M.Sc. (Computing Science), Simon Fraser University, Canada, 2016
B.Tech (Electrical Engineering), Indian Institute of Technology, 2014

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

in the
School of Computing Science
Faculty of Applied Sciences


**SIMON FRASER UNIVERSITY**
**Summer 2020**

# Declaration of Committee

| | |
|---|---|
| **Name:** | **Srikanth Muralidharan** |
| **Degree:** | **Doctor of Philosophy** |
| **Thesis title:** | **Multi-person video understanding with deep neural networks** |
| **Committee:** | **Chair:** Yagiz Aksoy<br>Assistant Professor, Computing Science |
| | **Greg Mori**<br>Supervisor<br>Professor, Computing Science |
| | **Anoop Sarkar**<br>Committee Member<br>Professor, Computing Science |
| | **Ghassan Hamarneh**<br>Examiner<br>Professor, Computing Science |
| | **Jim Little**<br>External Examiner<br>Professor, Computer Science<br>The University of British Columbia |

# Abstract

In this thesis, we present new methods to address multi-person scene understanding. Specifically, we focus on a multi-person task known as group activity recognition. We analyze multi-person scene understanding from the perspective of group activity recognition. We identify key challenges in group activity recognition, and present deep neural networks based approaches to handle these challenges. We show that our proposed approaches achieve competitive performance for group activity recognition. We also study one of the key components of group activity recognition in more detail, that is the problem of sequence modeling, where we apply new sequence modeling methods to the task of dense video captioning. In the end, we also investigate how to compress these large deep neural networks for efficient recognition on specialized domain tasks.

**Keywords:** Group activity recognition, Video Understanding, Network Compression, Dense Video Captioning

# Table of Contents

# List of Figures

# Chapter 1

# Introduction

Multi person video understanding is a computer vision task that has applications in sports analytics, robotics, surveillance, information retrieval, to name a few. Group activity understanding is one of the multi-person video understanding problems in computer vision. Group activity understanding deals with problems including temporal modeling of individuals/group activity, and capturing structured interactions between individual action/pose dynamics with group activity [30, 57, 79, 81]. These problems are studied in other computer vision areas and have their own set of applications. In this thesis, we first focus on the problem of group activity recognition, and introduce new techniques to perform group activity recognition. Second, we focus more on the problem of temporal modeling in videos, an important component of group activity recognition, where we delve into other video understanding applications, namely the problem of dense video captioning. Lastly, we introduce a principled method known as “fine pruning" for pruning large neural networks to deal with specialized domain recognition tasks.

*Group activity recognition* [30, 57] is a multi-person scene understanding task where the goal is to capture relevant multi-person information for the right classification of “group activity". A multi-person scene in the context of group activity recognition task consists of numerous individuals, each performing an “action", interaction between multiple individuals, and a scene level description known as “group activity" which depends on the context of the scene. Therefore, group activity recognition models mainly involve temporal understanding individual actions and the nature of the interaction between these individual actions, which influences overall group activity. Note the term "action" to refer to individual actions as opposed to "group activity", which is a scene level description involving a context-dependent aggregation of multiple individual actions. The main set of challenges in group activity involves inter-class similarities between individual actions, frequent occlusion of individuals in the scene, and intra-class variability in group activities.

In this thesis, we aim to address these important issues by using representations involving deep neural networks. First, we propose a novel hierarchical deep temporal model to capture temporal dynamics for learning group activity representations. [57]. Second, we propose a new deep structured model for capturing interaction between individual label representations [30] and group activity representation. While our hierarchical deep temporal model [57] focuses on building mod-

els to capture temporal dynamics of the individual actions and group activity, the deep structured model captures interactions between individual action and group activity representation through our message passing neural network [30].

*Temporal understanding* of long video sequence is an integral part of multiple computer vision problems including group activity recognition, dense video captioning and action recognition, to name a few. In the next part of this thesis, we focus on the task of dense video captioning to study temporal modeling in videos. The main goal in dense video captioning is to detect events happening in a long, multi-event video and describe each event with a natural language caption. The input consists of a multi-event video, where the occurring events are often correlated to each other. Models built for dense video captioning aim to capture these correlations. Traditional methods for dense video captioning use recurrent neural network representation to encode videos and to caption the events occurring in the video. However, they are limited in their ability to model long term dependencies and capturing long term correlation. Therefore, it is necessary to build models that can robustly capture these long term dependencies.

*Network compression* is a useful tool in efficient deep learning, which helps compress a large deep neural network for efficiently performing transfer learning tasks. The network is usually pretrained on a large scale image classification task. For transfer learning, this pretrained network is used for the new task by replacing classification component with a desired new set of layers and performing a “finetuning" operation. However, this neural network might contain many redundant parameters, considering the problem of specialized domain recognition. Therefore it is possible to achieve robust performance on this task with a model that has a much smaller set of parameters compared to the original model. Network pruning techniques are one such category of network compression methods that aim to reduce the number of model parameters. In the previous network pruning methods, pruning is performed layer-wise, independent of each layer, involving multiple pruning hyperparameters that control the trade-off between pruning ratio and accuracy [45]. There are multiple potential issues associated with traditional pruning methods. First, the pruning parameters are set manually and remain fixed during training. Second, pruning is independent of fine-tuning operation. Third, pruning is done independently for convolutional layers and fully connected layers. We propose a novel principled method known as fine pruning to address these issues. We conduct experiments on two specialized domain image classification datasets [21, 173] to demonstrate the effectiveness of fine pruning.

In the subsequent sections, we elaborate on each of the core contributions of the thesis and summarize these contributions at the end of this chapter.

## 1.1 Hierarchical deep temporal model for group activity recognition

We propose a new deep hierarchical model and a new dataset for group activity recognition. The proposed model consists of a two-level hierarchical model that focuses on modelling individual person dynamics at the bottom level of the hierarchy and the top level of the hierarchy. Each hierarchy

level uses a long short term memory (LSTM) network [53] to capture their respective dynamics. In addition, person-level representation finetunes an Imagenet [28] pretrained convolutional neural network for group activity recognition.

Our second contribution is a new volleyball dataset for group activity recognition. While the previous datasets for group activity recognition mainly focus on surveillance settings, our new dataset consists of volleyball sports videos, which adds diversity to group activity recognition benchmarks. This dataset brings a new relationship between group activity and actions of individual persons, different from surveillance videos, thereby diversifying the task in multiple aspects. The final dataset consists of 4830 video clips with frames of size 1280 by 720. We use this new dataset as well as the benchmark of collective activity recognition dataset for our experiments to show the effectiveness of our new model.

## 1.2 Deep structured model for group activity recognition

We propose a new deep structured model consisting of one or more “message passing" blocks that capture relevant dependencies of group activity, which ultimately refines the group activity label prediction. In this part, we propose a neural graphical model for group activity recognition. Group activity depends on the properties of individual motion: individual actions and pose. The goal of our neural graphical model is to capture these dependencies to refine group activity prediction. Our model consists of message passing blocks, which contain a neural factor layer that generates refined individual/group activity label predictions. Thus, these message passing blocks refine the label predictions of our network. We use supervision in the form of individual action and pose labels, in addition to group activity (also referred to as “scene" label) labels. The training process consists of finetuning the deep convolutional neural network and training the message passing blocks jointly to perform the final refined group activity prediction. We show that our novel deep structured network achieves competitive performance for group activity recognition on two surveillance datasets [18, 78].

## 1.3 Fine Pruning for Network compression

We propose a principled method known as “fine pruning" that performs fine-tuning and pruning of a deep neural network with large number of parameters. Our method is iterative, with each iteration consists of fine tuning, selecting optimal pruning hyperparameters, and pruning the network with these pruning hyperparameter settings. Our task’s main objective is to optimally prune the deep neural network while maintaining the main performance measure of the task (classification accuracy on the validation set in our case) intact. In each iteration, we have a hyperparameter candidate pool, which picks the candidate that achieves the optimal value of our objective function. Computing the best hyperparameter candidate in a brute force fashion is computationally expensive overall candidates in the pool. Therefore, we conduct hyperparameter search efficiently using Bayesian

optimization [137], although we note here that our method is flexible to use any hyperparameter search method. Thus, we feed this optimal hyperparameter to the pruning algorithm, which prunes the network at the last step of the current iteration.

## 1.4 Memory-augmented recurrent neural networks for dense video captioning

Dense video captioning involves detecting and captioning all the events happening in a multi-event video. We improve over previous recurrent representations for dense video captioning by augmenting them with external memory to capture long-term dependencies. We use a Bidirectional long short term memory (Bi-LSTM) network, one for video encoding, and one for captioning each event. To demonstrate the effectiveness of our memory-augmented representation, we focus solely on the captioning task, where we assume groundtruth event segments are already provided.

We train this model end-to-end, which gives us a video encoder representation sensitive to all the events happening within the video. We show that our model using an external memory for video encoding and captioning obtains competitive performance over baseline methods that do not have at least one external memory augmentation. We use two benchmarks in our experiments: ActivityNet captions dataset and YouCook II dataset.

## 1.5 Summary of contributions of the thesis

In summary, the core contributions of this thesis are:

- **Hierarchical deep temporal model** for group activity recognition: We propose a novel hierarchical deep temporal model for group activity recognition. We also introduce a new dataset for group activity recognition known as the volleyball dataset. This work was published in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) 2016 [57], and our volleyball dataset has been used by recent work in their research [131, 90, 110, 9, 166, 56].

- **Deep Structured Models** for group activity recognition: We propose a new deep structured model for group activity recognition. This approach uses a novel neural message passing neural network that mimics message passing operation and refines the label predictions. This work was published in British Machine Vision Conference (BMVC) 2015 [30].

- **FinePruning:** We introduce a principled method for network pruning that jointly performs finetuning and pruning operations by adaptively setting pruning hyperparameters in each iteration. This work was published in British Machine Vision Conference (BMVC) 2017 [150].

- **Memory-augmented recurrent neural network** for dense video captioning: We propose a new approach to dense video captioning that uses a memory-augmented multi-event video encoder and a memory-augmented caption decoder that generates a caption corresponding to each event in the video.

# Chapter 2

# Related Work

In this chapter, we review seminal work that is related to this thesis. We begin with a seminal work review in group activity recognition, a multi-object scene understanding problem. Group activity recognition is a task related to other computer vision tasks, including video understanding for modeling individual actions, and structured prediction models capturing different dependencies between group activity and individual actions. Therefore, research areas related to group activity recognition include action recognition, sequence understanding, event recognition, object detection/tracking, graphical models, and video understanding. Applications related to group activity recognition include surveillance (such as monitoring crowded locations like vehicular traffic, public transportation terminals, places of attractions), and sports analytics (especially suitable for team sports like volleyball, basketball). We review methods and problems related to group activity recognition.

Then we turn our attention to the problem of dense video captioning. Dense video captioning deals with the problem of describing multiple events that occur in a multi-event video. It is related to other tasks in computer vision, including sequence modeling applications in videos similar to group activity recognition, as well as multimodal computer vision tasks such as image captioning, dense captioning, and video captioning. We present a review of methods and problems related to dense video captioning.

The final part of our research deals with compression of deep neural network models for specialized domain recognition tasks. Network pruning, which we perform in our case, is one of the many techniques used for network compression. Other network compression methods, including weight quantization, knowledge distillation are also discussed later in this chapter. In the end, we present a review of methods related to network pruning and network compression.

## 2.1 Review of Group Activity Recognition

### 2.1.1 Methods for Group Activity Recognition

Intille et al. [58] deal with the task of multi-person action recognition using a structured model that captures individual actions, interactions between individuals, and integrates multiple feature sources for multi-person action recognition. The input to the system is a video, with a set of in-

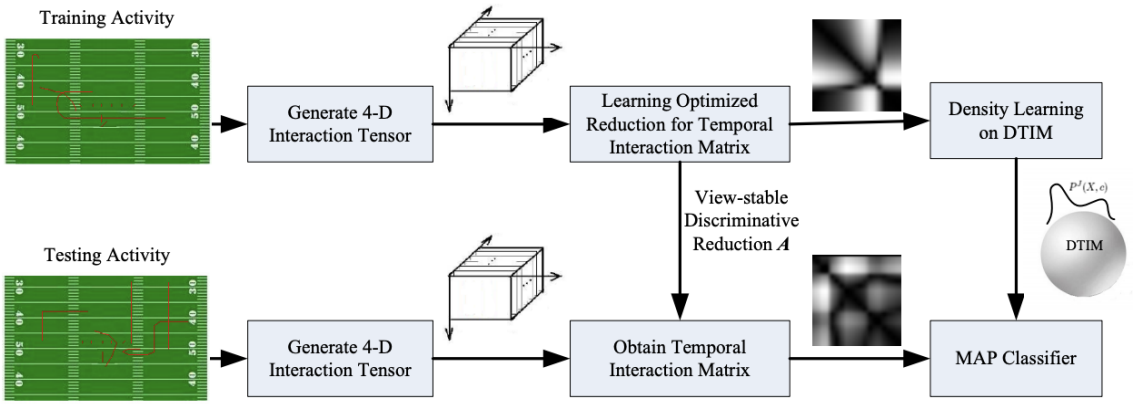



Figure 2.1: Illustration of train/test group activity recognition pipeline proposed in Li et al. [88] (Fig.obtained from [88])

dividual trajectories in the video. These trajectories are input to a visual network that performs individual action recognition. Next, a set of temporal analysis functions compute temporal relations between the individual actions. A multiagent belief network integrates information from the output of temporal analysis functions to predict multi-person action recognition.

Gong et al. [40] build a model based on dynamic probabilistic models to capture interactions among multiple individuals for group activity recognition. Towards this end, they propose a dynamically multi-linked hidden Markov model (DML-HMM). The proposed model overcomes the limitations of previous work in that this model is capable of dynamically capturing topology of the underlying spatio-temporal interactions. The proposed model establishes links to states from previous timesteps by factorization of state transition matrices, picking the most essential states. Experiments on an outdoor surveillance dataset using both clean and noisy sequences demonstrate that the proposed DML-HMM model outperforms previous state-of-the-art models for group activity recognition.

Li et al. [88] analyze data-driven discriminative regions in a temporal interaction manifold for group activity recognition. The pipeline of the proposed approach is shown in Fig. 2.1. Given individual trajectories and roles, the work attempts to model group activity using temporal interaction matrix, which to used to construct a manifold on which all the group activities could be projected, known as discriminative temporal interaction manifold (DTIM). A probabilistic generative distribution is constructed for temporal information matrix on this manifold in order to capture distribution of group activity classes. Group activity classes are then associated with a likelihood density, and further, a maximum aposteriori classifier is trained using this likelihood density for group activity recognition. Experiments are conducted on a sports dataset and it is shown that the proposed probabilistic modeling DTIM outperforms other baseline classifiers.

Ryoo et al. [124] introduce a stochastic model involving a hierarchical representation to perform group activity recognition and localize individuals performing the group activity. The proposed system models group activity as a formal representation capable of incorporating stochastic structures

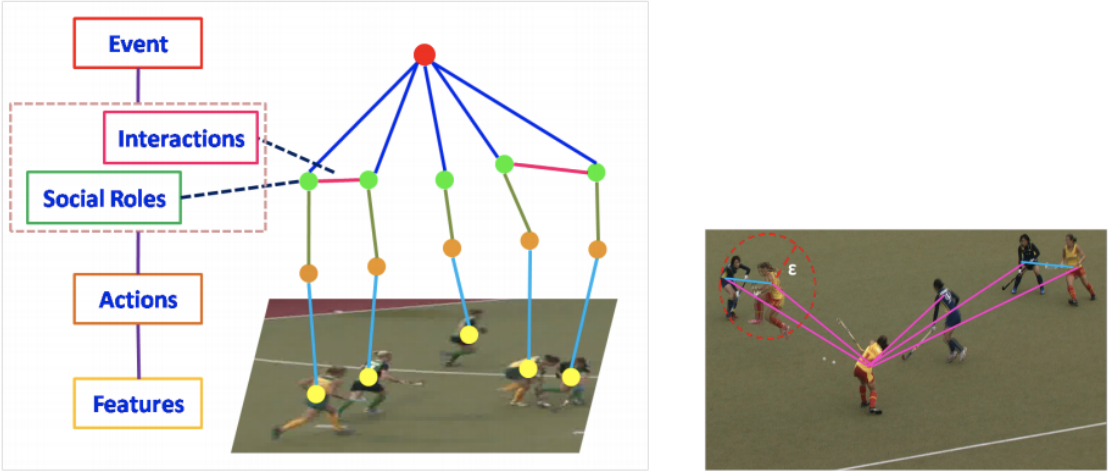



Figure 2.2: Overview of hierarchical model for group activity recognition, with colour coding of various model components (Fig.obtained from [79])

present in group activities annotated manually by a human expert. Further, a hierarchical model is used to localize a subset of individuals among the entire scene performing group activity, using stochastic sampling to filter candidate individuals who are not part of group activity. The hierarchical model performs Bayesian inferencing to compute group activity. Experiments conducted on the newly constructed dataset show that the proposed stochastic model outperforms a previous deterministic method used for group activity recognition.

Choi et al. [19] deal with group activity recognition by building contextual feature representation for individuals by considering neighbourhood into account. The neighbourhood context is used to construct feature representation for each person, for classifying each person into one of group activity classes. Neighborhood context is captured using a spatio-temporal representation is constructed using a random forest based approach, which partitions the resultant spatio-temporal space and constructs the feature representation by choosing the most discriminating volume. Finally, a Markov random field is used to address the noise in the neighbourhood based individual feature representation occurring due to errors in human pose or trajectory estimation. Experiments show that the proposed system obtains state-of-the-art performance while also outperforming simpler models that do not include at least one of the system's components.

Lan et al. [81] propose a discriminative structured model for group activity recognition. The proposed model explores contextual features for group activity recognition such as interaction among individual actions, interaction between group activity and individual actions. These interactions are captured in multiple ways. First, the interactions automatically inferred using a structured model in latent variable framework. Second, spatio-temporal "action-context" descriptor that encodes individual features interactions in space-time proximity are extracted and utilized by the model. Learning the structured model and performing inference of the graph structure and the labels are both

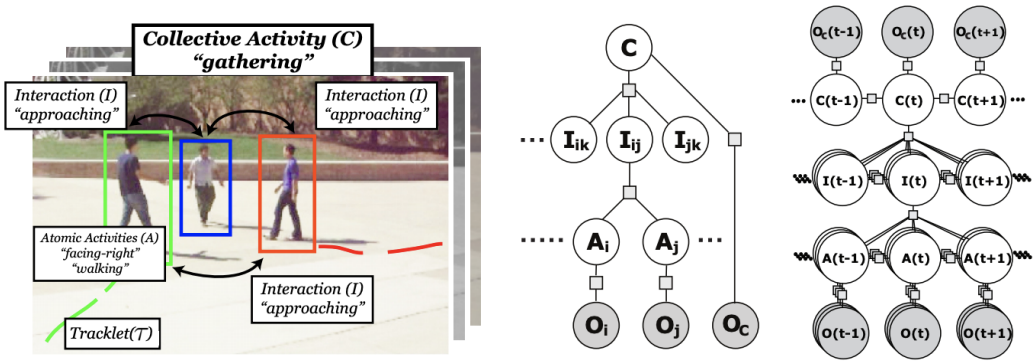



Figure 2.3: Overview of unified tracking and group activity recognition framework proposed in Choi et al. (Fig.obtained from [17])

performed using an approach closely related to Latent SVM [182] formulation. Experiments on multiple datasets demonstrate that the model that combines these two modes of capturing interactions outperforms previous methods for group activity recognition and simpler baselines that do not incorporate both these interaction features.

Lan et al. [79] introduce a hierarchical model for group activity recognition. The proposed model, illustrated in Fig. 2.2 performs human action understanding at the lower level of the hierarchy, and capture interactions between individuals in the scene at the higher level of the hierarchy, eventually to perform group activity recognition. They capture inter person interactions as "social roles". The proposed structured model is learned using Structured SVM [62]. Experiments are conducted on the newly constructed hockey dataset to show the effectiveness of the proposed model, and the ablation studies indicate the importance of the hierarchical model in achieving the best recognition performance. Choi et al. [17] propose a unified framework for tracking people and determining group activity in a multi-person scene. The proposed approach, illustrated in 2.3 consists of a tracking component and a hierarchical structure to capture a set of actions in the scene. First, a hierarchical structure is used for modelling individual actions, interactions between individual actions, and group activity. Second, tracks of individual persons in the video are computed using multi-target tracking, influenced by interactions captured in the hierarchical model, formulated as a min-cost network problem. The model learning and the inference during test time is performed jointly. Experiments on multiple datasets show that the unified framework achieves competitive performance with respect to the state-of-the-art for group activity classification and multi-target tracking.

Lan et al. [78] identify action primitives in a weakly supervised learning framework, which are shown to be effective for group activity recognition. These primitives are learned using a discriminative max-margin clustering method. The method first identifies primitives within a given video and then proceeds to use these intra-video clusters to build action primitives that are visually consistent across all the training data. The first step, the intra-video clustering, is used to identify individuals from adjacent frames. The second step, the inter-video clustering, is used to identify in-

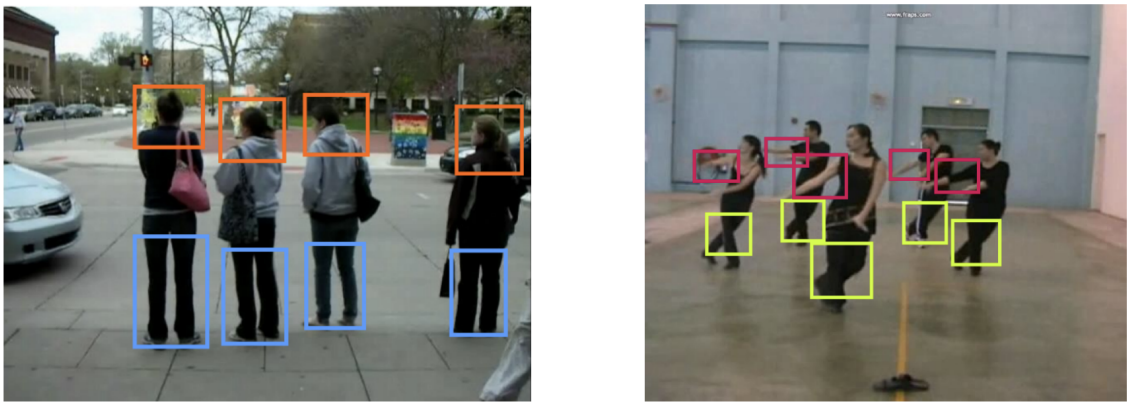
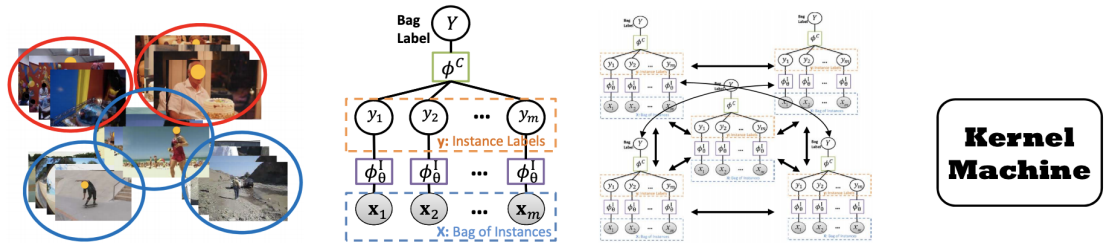


Figure 2.4: Multiple,semi-local candidate activity constituents being identified for group activity recognition in Antic et al. [6](Fig.obtained from [6])

dividual actions that are consistent across videos to obtain action primitives. Finally, group activity representation is learnt using a latent SVM framework, where the representation models interactions between action primitives found in a given video and group activity, as well as interactions between group activity and individual actions. In addition to group activity recognition, action localization model is also built on top of action primitives. Experiments on both these tasks show that action primitives are effective for all the considered tasks, and offers explainable solution to group activity.

Antic et al. [6] characterize group activity as being dependent on spatio-temporal dynamics and interactions between multiple local regions. To construct this dependency the model, illustrated in Fig. 2.4 first clusters local regions of the video and trains a classifier that captures these constituent local regions that could be meaningful and related to the overall group activity. The constituent classifier learning is framed as a multiple instance learning, with the example consisting of multiple noisy local regions from multiple examples which were clustered based on functional and visual similarity. These classifiers aid in filtering the relevant local regions. The constituent classifiers are learned in a multiple instance learning framework jointly with the group activity classifier in a max-margin framework. Experiments indicate that the proposed system outperforms the ablation models and achieves state-of-the-art performance.

Tran et al. [149] characterize the group activity recognition task as having to identify the dominant sub-group among multiple candidate groups of people in the scene, and learn to perform group activity recognition by learning from spatio-temporal feature maps of this dominant group. They identify this group using two novel “social signaling cues", which describes the degree of interaction between people in a group. These include social distance cue and visual focus of attention cue. Interactions between individuals are represented through a graph where interactions correspond to edges and graph clustering is performed to identify each group in the scene. A novel local group activity feature descriptor is also proposed to describe dominant group. Experiments on multiple datasets show that the proposed approach outperforms baseline methods and achieves performance comparable to previous state-of-the-art group activity recognition methods.




(a) Preparing Training Data as Positive and Negative Bags of Instances

(b) Learning the Cardinality Model

(c) Computing Cardinality Kernels for all Bag Pairs

(d) Using a Kernel Method to Train a Classifier

Figure 2.5: Overview of cardinality kernel for visual recognition tasks (Fig.obtained from [47])

Amer et al. [4] deal with the task of group activity recognition using a new graphical model that attempts to localize long-term collective activity that could occur in a video scene by capturing spatio-temporal dependencies in a hierarchical fashion. The model aims to identify the individual action relevant to the activity in the lower level of hierarchy, and model long term temporal dependencies in the higher level of hierarchy. For the hierarchical model, the work modifies previous work on hierarchical conditional random field to have only vertical connections to capture long term dependencies, and the inference is carried out using linear program to compute latent variables at each level of the hierarchy one at a time. Experiments show that the proposed modification and the resultant structure of the graphical model used in this work outperforms previous hierarchical conditional random field models, and achieves state-of-the-art classification performance.

Hajimirsadeghi et al. [47] propose a new framework to capture relations between a recognition task and cardinalities of the constituent units under consideration. The proposed method, illustrated in Fig. 2.5 involves learning cardinality relations in a multiple instance learning setting, and the model parameters are learned using hidden conditional random field. They adopt a kernelized framework to learn the instance cardinality - bag label relations. They conduct experiments in multiple applications, including group activity recognition and action recognition to show that the proposed cardinality kernel achieves comparable performance to the corresponding state-of-the-art models in the considered tasks.

Deng et al. [29] performs group activity recognition by learning to infer relations between individual actions and group activity for refining individual action as well as group activity predictions. The proposed approach consists on a recurrent neural network that performs message passing between different individual/group activities for refining their values and uses gating mechanism to learn the structure of the scene, which is comprised of individual actions and group activities. Experiments on multiple datasets show that the proposed approach achieves competitive performance for group activity recognition.

Shu et al. [131] deal with group activity recognition involving hierarchical temporal models. They address brittleness in predictions made by cascade LSTMs by casting prediction problem as energy minimization with confident predictions as opposed to softmax based predictions. Their base model consists of a hierarchical model that captures individual actions, interaction between

individuals in the lower hierarchy, eventually predicting group activity at the upper level of hierarchy. However, for prediction of individual actions and group activity, they use an additional energy minimization layer on top of softmax operation. The energy minimization layer is also regularized by a prediction confidence term to improve the reliability and numerical stability of energy layers' predictions. Experiments conducted on two datasets shows the effectiveness of the whole model over baselines and previous hierarchical methods that perform softmax based predictions.

Li et al. [90] propose semantics based group activity recognition. The proposed method consists of a "captioning" network that learns to predicts attributes of a scene, i.e. individual action labels as a caption, and then uses this sentence to learn group activity representation. The input to the caption network is an aggregated visual feature representation from a two streamed network. The caption word matrix is then used as input to another convolutional neural network with an additional LSTM at the top that is learnt to predict the right group activity label. Experiments conducted on multiple datasets show that the proposed approach achieves competitive group activity recognition performance.

Bagautdinov et al. [9] propose a method that jointly detects individuals and performs group activity as well as individual action classification using a fully convolutional network based representation and a matching recurrent neural network. The proposed approach consists of two fully convolutional network blocks, one which is directly used for activity classification, while the other block is used for dense person detection using multi scale features. The detection component further uses a Markov random field to refine the predictions made by FCN based detections. It is also used by the matching recurrent neural network to predict individual and group activities. The detection and classification tasks are trained jointly. Experiments on multiple datasets show that the proposed approach achieves state-of-the-art group activity recognition as well as multi-person detection.

Qi et al. [110] introduce a semantic model for group activity recognition that learns contextual spatio-temporal features for capturing interactions between individuals eventually useful for group activity recognition. The proposed approach constructs spatio-temporal graph representation of scene and uses a semantic RNN to learn interactions between different individuals. The learning of contextual features in the semantic graph representation using semantic RNN occurs through message passing and factor sharing to infer the spatio-temporal graph structure. Finally, a novel spatio-temporal attention mechanism is employed to identify key actors and keyframe information, which is eventually helpful for performing group activity recognition. Experiments on multiple datasets demonstrate the effectiveness of the complete model in achieving competitive performance for group activity recognition.

Ibrahim et al. [56] build a hierarchical relational network for multi-person scene understanding tasks. The proposed relational network consists of relational layers that encodes inter-person relations in the scene. Each person in the scene is encoded by a relational layer using a shared relational unit representation. Multiple relational layers are then stacked to build a hierarchical relational representation for the whole scene. The network is operated under supervised setting for group activity recognition and under unsupervised setting where person level and scene level feature


Figure 2.6: Sample examples from the collective activity dataset (Image obtained from dataset homepage)

representation are constructed using a hierarchical relational model for scene/person retrieval tasks. Experiments conducted on these tasks demonstrate that hierarchical relational network is able to learn a discriminative feature representation for group activity recognition as well as scene/action retrieval tasks.

Wu et al. [166] model relations between individuals using a relation graph for group activity recognition. The proposed approach learns the graph representation using graph convolution network. Further, it is also shown that sparsification of graph representation through the proposed spatio-temporal sparsification mechanisms yields effective learning of group activity representation. The proposed approach learns graph representation to generate a set of person-level relational features to predict group activities. The proposed approach is also shown to generate graphs that could be interpreted to show inferred relations between individuals and group activity. Experiments on two datasets show that the model achieves state-of-the-art group activity recognition performance.

### 2.1.2 Datasets

**1. Collective Activity dataset:** Choi et al. [18] introduce a new dataset known as Collective Activity dataset, consisting of pedestrian videos where people perform group activities such as queuing, walking, to name a few. Further, they propose an extension of Kalman filtering to track the detected people and a local spatio-temporal descriptor for performing collective activity recognition. Their method outperforms previous state-of-the-art techniques for collective activity recognition. Collective activity dataset consists of 44 surveillance scene videos, 8 individual pose labels, 5 individual action labels (illustrative examples shown in Fig. 2.7), and five group activity labels (also known as

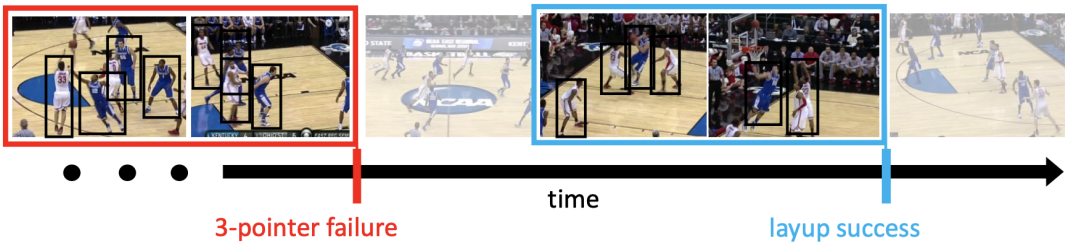



Figure 2.7: Example dense annotation in the NCAA Basketball dataset, highlighting timestamp and event label annotations (Image obtained from Ramanathan et al. [111])

scene labels). The scene labels are the same for crossing, waiting, queuing, walking, and talking. A scene is assigned a group activity label based on the majority of what people are doing.

**2. UCLA Courtyard dataset:** Amer et al. [3] introduce UCLA Courtyard dataset, which consists of 106-minute multi-person, multi-scale videos with multiple scales of annotated semantic information. The dataset consists of 6 group activities, 10 primitive actions, and 17 object annotations. They propose a new model that models individual actions, group activities, and relevant objects in a hierarchical structure using an AND-OR graph. Inference is conducted using an efficient exploration-exploitation strategy in an iterative fashion. Experiments are conducted on multiple tasks, including group activity recognition and multi-scale semantic activity detection on the newly constructed dataset. These experiments show that the proposed inference strategy achieves competitive performance with decreased time complexity.

**3. Nursing Home dataset:** The Nursing home dataset [195] consists of indoor surveillance videos obtained from long-term care facilities. It consists of 123 short clips, with six action labels. In this work, the main task objective is to detect the “fall" action, and therefore dealt with a binary classification problem.

**4. NCAA Basketball dataset:** Ramanathan et al. [111] identify key actors using attention maps in a multi-person video for event recognition task. The proposed approach tracks people and uses recurrent neural network representation to encode their feature representations. An attention map then computes a weighted sum over all these feature representations, which is utilized by another recurrent neural network for performing multi-person event recognition.

Experiments are performed on the newly constructed dataset consisting of basketball videos with event annotations. The dataset consists of 14548 24-frame clips from 257 basketball games and eleven unique labels. Furthermore, they also annotated 9000 frames from the training videos with bounding box annotations of individual locations for training an object detector. Finally, for evaluation purposes, they annotated the ball’s position, consisting of 850 test video clips.

Experimental results show that the proposed approach outperforms the previous state-of-the-art multi-person event recognition methods and baselines. Besides, it was also shown that the proposed attention mechanism is able to localize key actors of the scene effectively.

### 2.1.3 Related research Work

Group activity recognition methods operate on multi-person videos and build models by capturing individual actions and interactions between them. These sub-problems are related to other research areas including action recognition and multi-object tracking. Finally, we also discuss other multi-person recognition tasks and techniques related to group activity recognition.

#### Action Recognition

Action recognition is a research area closely related to group activities, where learning individual action representation, also known as “atomic" action forms part of group activity recognition models [57, 79, 30, 81]. Action recognition research concerns with video scenes with a single person performing some action. The main goal is to classify a person’s actions into one of the predefined set of categories. Several action recognition datasets exist, which focus on different scenarios. Action recognition datasets consist of different types of videos, such as egocentric videos, indoor videos, sport videos to name a few. The scale of action recognition datasets gradually progressed in scale from KTH dataset [125] one of the small scale action recognition datasets has few hundred curated videos with a few actions, to recent datasets containing YouTube videos which are millions in numbers [67] with hundreds of action categories.

Earlier methods for action recognition involve building different kinds of handcrafted feature representations for action recognition [125, 82, 36]. Recent techniques, inspired by the success of deep learning techniques in other computer vision and machine learning tasks, utilize deep neural networks for action recognition [67, 135, 32, 147]. Deep neural networks for action recognition employ different networks involving architectures such as 2d/3d convolutions, recurrent neural networks, and optical flow-based models. Inspired by these methods, we build our deep models for group activity recognition. We refer to multiple surveys for a more detailed review of seminal work in action recognition [108, 1, 51].

#### Object tracking

Group activity recognition task consists of multi-person videos, and group activity recognition methods often detect and track individuals. The most relevant research problem relevant to this framework is multi-target tracking. Multi-target tracking involves computing a set of “tracks", each corresponding to an object in the video. Each track is characterized by a set of bounding boxes from each frame corresponding to the location of the object under consideration, which has a center, width and height. Tracking has applications in video surveillance, robotics, sports analytics, to name a few, and is used in multiple computer vision problems, including action recognition, visual navigation, pedestrian analysis, to name a few. Challenges to multi-target tracking include occlusions, presence of indistinguishable objects, object clutter.

Here, we reviewed group activity recognition methods that deal with people trajectories, also known as “tracks", including work that make use of trajectories for group activity recognition in sev-

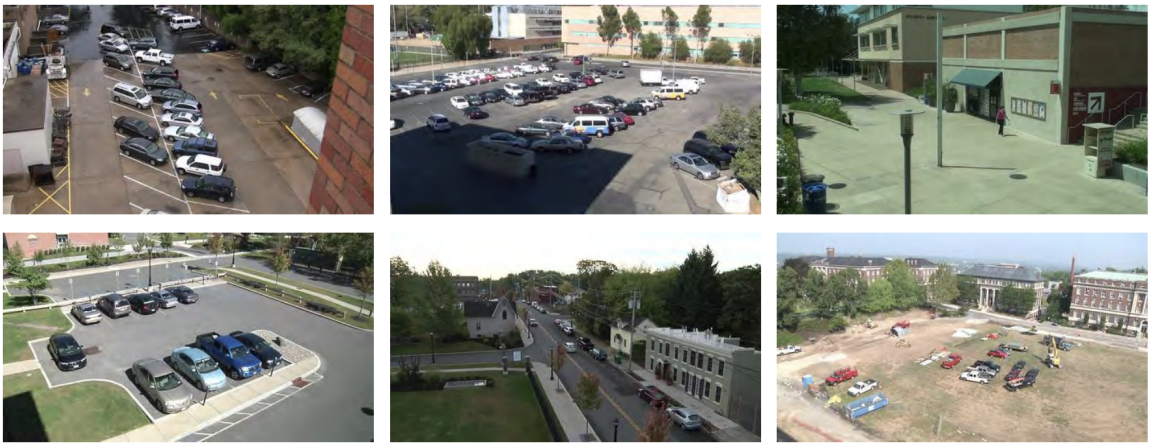


Figure 2.8: Sample scenes from the VIRAT dataset (Image obtained from Oh et al. [103])

eral different ways. Some methods approach group activity recognition by using the given trajectories for group activity recognition [58, 88, 142], while other methods extract different neighborhood representations for group activity recognition [18, 19, 101] and other methods that jointly learns individual action/group activity representation alongside multi-target tracking, thereby aiming to improve multi-target tracking using the former as contextual information [123, 17].

### Other Related Work

Morariu et al. [97] address the problem of **multi-agent event recognition** in videos. The input to the model is a set of person trajectories, rules, and other meta information relevant to the input scene, which is necessary to generate observations and the event interval outputs. The proposed approach first processes the input to generate hypothesized events and perform event reasoning using interval logic representation. Second, Markov logic networks generate probabilistic event predictions from these noisy first-order logic event hypothesis by constructing a factor graph representation of the logic. Experiments show that the proposed probabilistic MLN framework refines the hypothesis predictions made using interval logic representation. **Event recognition** task has application in surveillance videos, where the task focuses on recognition of events, including individual-object interactions such as vehicle opening, for instance, individual pair interactions and individual-centric events, such as an individual exiting a room [103, 123]. Oh et al. [103] introduce a new dataset known as “VIRAT dataset" for event recognition in surveillance videos. The dataset consists of aerial outdoor video scenes and provides the annotations of localized events and tracks of moving objects in the scene. Thus, the dataset acts as a benchmark for event recognition in continuous videos and multi-object tracking. The dataset contains 23 types of events and 17 scenes, with 10-2500 samples per event category.

Shu et al. [132] address several tasks pertaining to the **multi-person aerial videos**: multi-person event recognition, grouping people, and recognizing social roles among people in these groups. The work proposes a spatio-temporal AND-OR graph-based framework that addresses all these

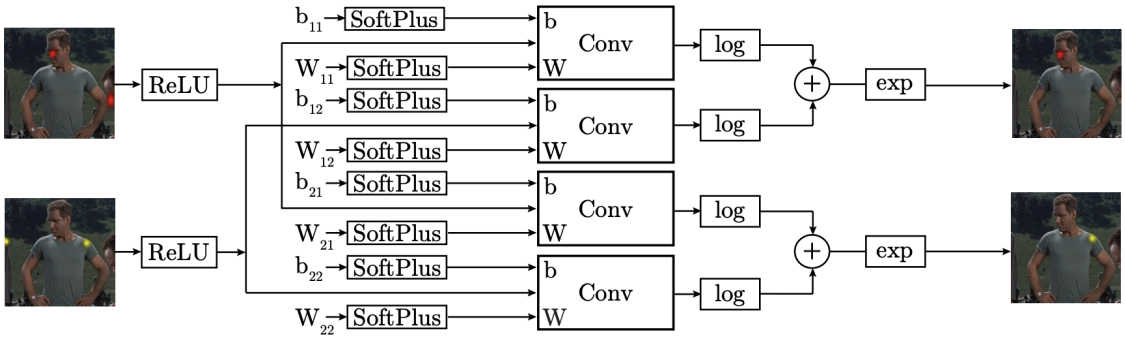



Figure 2.9: Message passing network for human pose estimation (Fig.obtained from [146])

problems jointly. This graph representation encodes relations between human roles, objects as latent spatio-temporal templates, and uses dynamic programming to perform all the tasks under consideration efficiently. Experiments on their newly constructed dataset show that their holistic, joint formulation outperforms ablation models that deal with each task separately.

Lucey et al. [94] use role-based individual representation in a multi-person sports video to understand **team activities**. The approach first builds a novel role representation for individuals and then employs this representation to discover team dynamics. The proposed approach constructs role-based adversarial representation, using expert role annotations/individual trajectories involving a bilinear spatio-temporal basis that encodes trajectories and second, using an automatic approach involving Hungarian algorithm. Finally, it is shown that this role-based representation could denoise player detections, which aids in team understanding applications. Experiments are performed on hockey videos to demonstrate the effectiveness of this framework in team activity understanding.

Direkoglu et al. [31] propose a new method for **team activity recognition**. Given a sequence of players' locations on a playground, the task is to classify the sequence into one of the "team activities". The proposed approach utilizes a novel motion information image sequence for team activity classification by first constructing a sequence of Poisson distribution images, which encodes the aggregate position of the entire team. The motion information image is obtained by computing frame difference and optical flow map on Poisson distribution image. Using this sequence, the spatio-temporal motion descriptors are computed, and a linear classifier is trained for team activity recognition. Experiments on multiple tasks, including team activity classification show that the proposed method outperforms the previous state-of-the-art team activity classification model while keeping the classifier efficiency intact at test time.

Tompson et al. [146] introduce a novel hybrid architecture for performing **human pose estimation** using a part detector built using multi-resolution deep convolutional neural network, illustrated in Fig. 2.9. Second, a spatial model is used to refine the part detector's output, which is inspired by Markov random fields and is constructed using a convolutional neural network that mimics relevant operations like sum-product belief propagation. This spatial model refines the first stage detector's pose prediction by enforcing global pose consistency and inter-joint connection patterns.

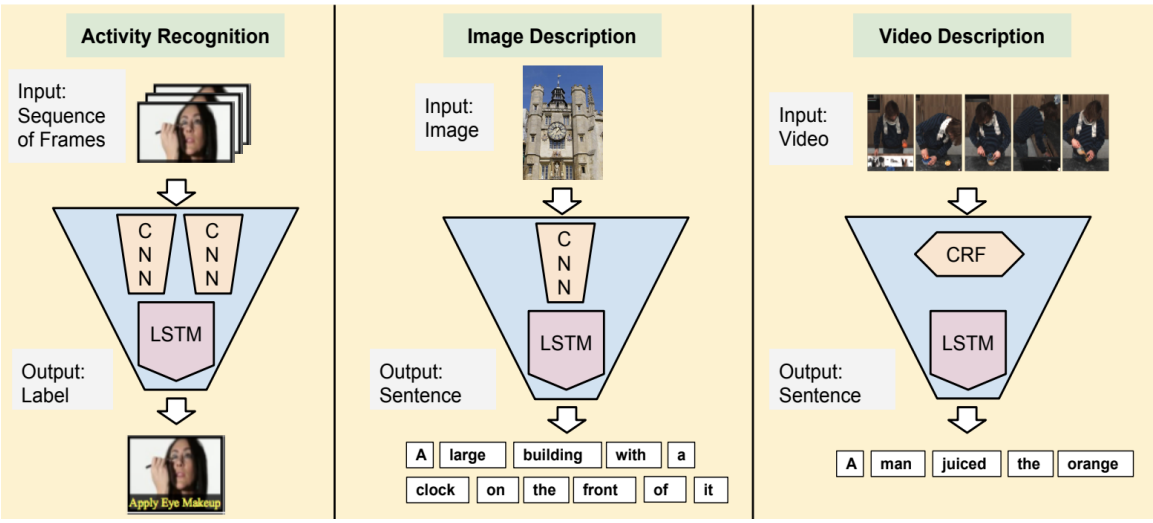



Figure 2.10: Overview of LRCN architecture applied to multiple visual recognition tasks (Fig.obtained from [32])

Both models are trained in a unified manner. The proposed framework outperforms the previous state-of-the-art methods on two different pose estimation datasets.

Donahue et al. [32] propose a new recurrent convolutional architecture that can deal with problems where both visual input and sequences are involved. The proposed approach, illustrated in Fig. 2.10 handles various of these tasks by a novel network architecture that comprises a convolutional neural network to deal with visual input LSTM model to deal with sequences. Models are designed using this framework for multiple sequence understanding based visual tasks including action recognition, image captioning and video description. Experiments conducted using the new architectures for each of these tasks show that the proposed models achieve competitive performance in all these tasks, demonstrating the promising applications of using recurrent convolutional recurrent architecture for computer vision tasks.

Ramanathan et al. [112] perform weakly supervised **social role discovery** in multi-person videos. They propose a model that consists of a conditional random field (CRF) that captures these interactions. The model infers the role labels using variational inference to set the model weights. The proposed model consists of components capturing human interaction features in space-time and proxemics. To test the model, they annotate a subset of Trecvid-med11 dataset, where each annotated example contains groundtruth information about social roles of each individual in the video. These annotations are one to one mapped to role clusters that the model predicts in a fashion that the mapping maximizes the model's performance. Experiments and ablation studies indicate that the whole CRF model with all the interaction features incorporated achieves better performance than the rest. Alahi et al. [2] introduce a new model called "Social LSTM", which they use to perform **trajectory prediction** of all the people jointly in a crowded scene. The proposed approach, illustrated in Fig. 2.11 augments LSTMs to capture dependencies between multiple sequences (trajectories in this case) using a "social pooling" that lets the model share information between spatially proximal

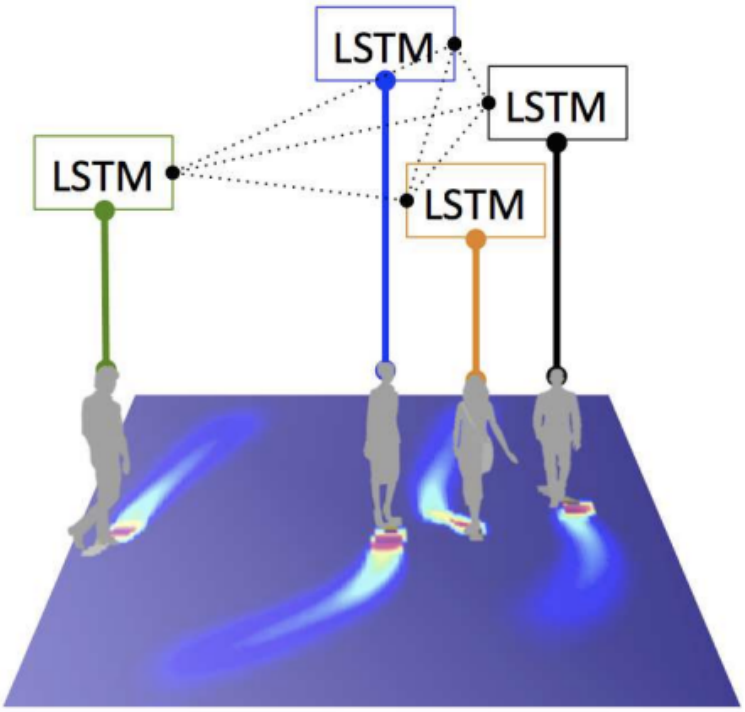



Figure 2.11: High level illustration of Social LSTM framework (Fig.obtained from [2])

sequences automatically. Their approach outperforms the previous pedestrian trajectory prediction models, which use handcrafted feature representations for trajectory prediction.

## 2.2 Review of Dense Video Captioning

### 2.2.1 Methods for Dense Video Captioning

Dense video captioning methods focus on the task of captioning each event that happens in a multi-event video. These methods first perform event detection, followed by a captioning component that describes each of the detection events in sentences. Here we review methods for dense video captioning.

Das et al. [26] propose a hybrid dense video captioning system that combines low-level multimodal topic model translation that learns keyword representations and a high-level graph based language learning which also involves relating them to the detected visual concepts “stitched" over time. Finally, a semantic verification system picks the most promising descriptions by comparing predictions made by these bottom-up and top-down predictions. Experiments on multiple datasets show the proposed approach’s effectiveness in generating semantically meaningful captions and competitive performance on caption metrics.

Rohrbach et al. [119] explore the task of multi-sentence video captioning, where they deal with generating captions at multiple levels of detail. The proposed approach generates a multi-sentence, multi-granular caption for a video by treating it as a novel concept based video segmentation that

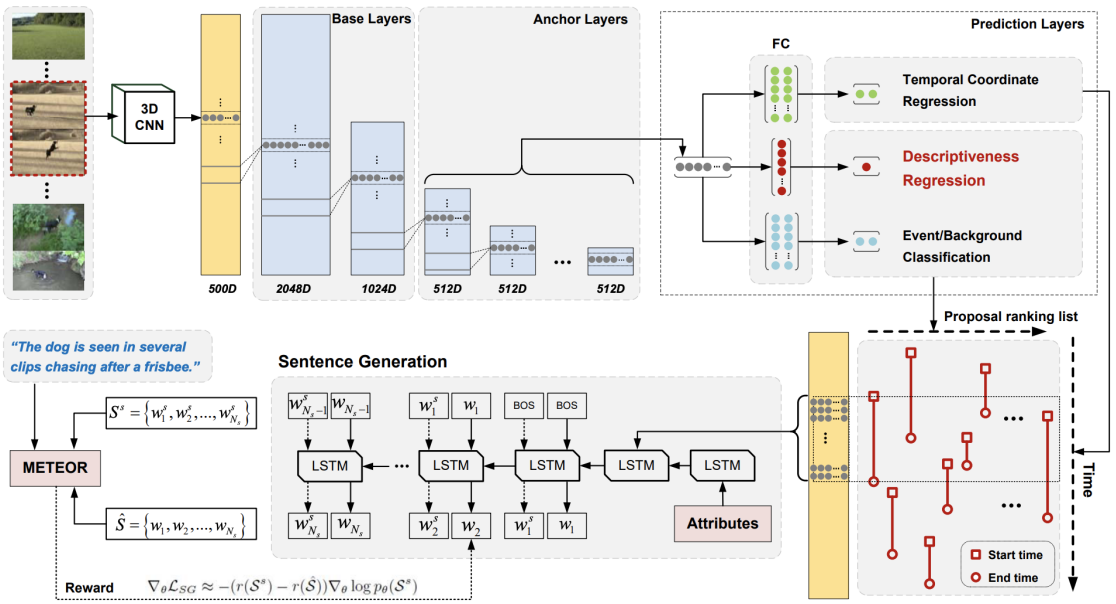



Figure 2.12: Overview of joint event detection and dense video captioning proposed in Li et al. [91] (Fig.obtained from [91])

automatically figures out granularity of description and generates a semantically coherent description. The approach first constructs an intermediate probabilistic semantic representation of the input video and then generates caption by using this representation. Finally, caption generation is enhanced by making use of task-specific semantic feature representation. Experiments conducted on visual recognition and multi sentence generation on TACoS dataset tasks demonstrate the effectiveness of the overall approach.

Li et al. [91] propose a new dense captioning method that jointly learns to detect events and caption them. The proposed framework, shown in 2.12 consists of a novel event detection system that leverages a captioning system to measure descriptiveness of a candidate proposal and a captioning system that uses reinforcement learning to describe events. The captioning system consists of an attribute prediction component used to predict attributes for an event proposal, which is used to generate caption. The captioning system is learned using reinforcement learning, where the captioning metric is used as the reward function, and the training objective is to maximize the expected reward. Experiments show that the overall proposed system achieves state-of-the-art event detection and dense video captioning results.

Xu et al. [168] propose an end-to-end dense video captioning system, involving a segment proposal network and a hierarchical captioning network. The segment proposal network receives C3D [147] map of video as input, and consists of 3D convolutional layers that learns to predict bounding box offsets at a set of anchors. The hierarchical captioning module is a two level hierarchical model that has a “controller LSTM" that utilizes language context from neighborhood events in addition to the visual context in order to refine the caption prediction and a “captioner LSTM" at

the lower level of hierarchy which captions each candidate proposal utilizing controller LSTM features at each timestep. Experiments conducted on multiple datasets demonstrate the effectiveness of the end-to-end contextual model over its baseline methods and previous state-of-the-art dense captioning methods.

Wang et al. [161] propose a novel dense video captioning framework, with contributions in the video proposal generation as well as the captioning framework. Their system consists of a novel Bidirectional Single Stream Temporal Action Proposal Network to generate proposals that consider context from the future into account. For captioning, they introduce a novel framework in the aspect of visual features they use for captioning. First, they use video encoder features at the boundary of a detected event as a context feature for captioning and is used in addition to the attended detection proposal features. Second, they fuse this context vector with the attended proposal feature using a gating mechanism. Experiments show that the proposed system achieves competitive performance compared to previous methods and better performance than baseline methods.

Zhang et al. [185] adopt a hierarchical cross-modal embedding based framework for multiple vision and language tasks, including dense video captioning. The key contribution of this work is to include a reconstruction objective at each level of hierarchy and to each modality to previous hierarchical cross modal learning methods for coherent embedding of each input component at multiple levels of hierarchy. Further, cross-modal alignments are also established at each level of hierarchy instead of previous work that focuses on global cross-modal alignments. Results on multiple sequential tasks, including video/paragraph retrieval, action recognition and dense video captioning, demonstrate the usefulness of adding the proposed unsupervised reconstruction loss in achieving competitive performance in these tasks.

Zhou et al. [192] propose a transformer-based end-to-end model for dense video captioning. The proposed approach utilizes transformers to encode video for performing event detection and captioning each of these events. These transformers consist of stacked blocks of self-attention layers and feedforward layers, similar to previous transformer-based models [153]. The method also proposes a differentiable masking network that enables them to train the entire pipeline, including the event detection component and the end of the captioning component. The proposed approach achieves competitive performance for event detection and dense video captioning sub-tasks on multiple dense video captioning datasets.

Mun et al. [98] consider and introduce a model that considers temporal dependencies between events. The model consists of an event sequence generation network that generates an ordered event sequence by selecting a subset of candidate event proposals using a pointer network architecture [157]. This enables the model to encode temporal dependencies between events in the event sequence. Then, a sequential captioning network captions each event in this sequence using reinforcement learning that comprises of multi level (individual event and episode level) rewards, and the training involves maximizing a weighted combination of these two rewards, thus encouraging coherence between caption generation for each individual event. Experiments show that the model achieves state-of-the-art captioning performance.

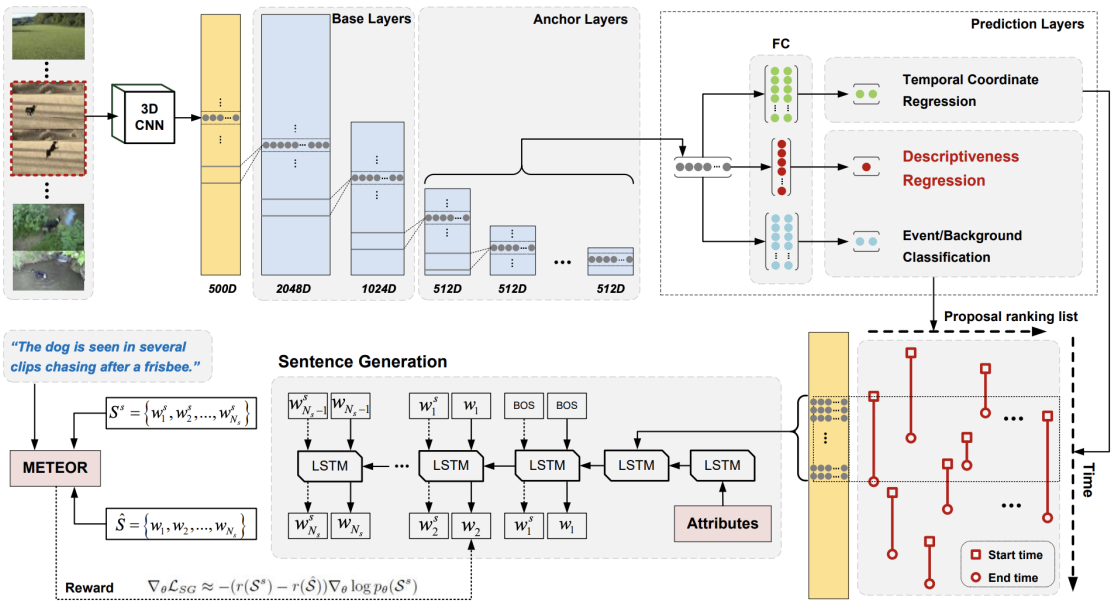



Figure 2.13: A sample dense caption from ActivityNet caption dataset [73] (Fig.obtained from [73])

### 2.2.2 Datasets for Dense Video Captioning

Regneri et al. [116] introduce "TACoS dataset", a dense video captioning benchmark in the cooking domain consisting of multiple, multi-resolution captions for a given video. It consists of 127 videos with 20 different textual descriptions, leading to 2540 annotations, with each annotation containing 5-15 sentences, with each sentence aligned to a video timestamp. On average, each description covers 2.7 low-level activities, with difference in granularity. This dataset's principal task is to ground action verbs and phrases, where many existing semantic similarity models were evaluated and compared to show that prediction models that use visual and textual information obtain better performance than baseline methods that rely on unimodal data.

Krishna et al. [73] introduce dense video captioning and build a new benchmark known as ActivityNet captions dataset for dense video captioning. The proposed approach consists of an online proposal module and a captioning module and utilizes contextual information to perform captioning in the form of neighbouring events. The new dataset they construct consists of 20K videos, which amount to 849 hours of video with 100,000 total event descriptions. On average, each video contains 3.65 temporally localized sentences, and each caption description has an average sentence length of 13.48 words. A sample from ActivityNet dataset is shown in Fig. 2.13. Multiple tasks are performed on this dataset, including dense captioning, event localization, and video/paragraph retrieval. They show that the proposed approach online and utilizes contextual information outperforms baseline methods that exclude context.

Zhou et al. [191] propose a new framework based on one-dimensional convolutions for event detection in a multi-event video. The proposed approach handles event detection as a temporal

segmentation task, consisting of context aware video encoder, temporal convolution based segment proposals, and finally a sequence prediction to yield final set of proposals using a recurrent neural network. The work also introduces a new dataset for event detection and dense video captioning, known as Youcook2 dataset. This dataset contains 2000 videos from 89 recipes with a total length of 176 hours. Each video contains 3âĂŞ16 event segments, also known as procedure segments, temporally localized and captioned manually. They conduct experiments on this dataset to show the proposed approach's effectiveness over baseline methods for event detection.

### 2.2.3 Other Related Work

**Image Captioning** task deals with describing images using natural language sentence(s). Describing images using a natural language sentence is known as Image captioning, whereas describing images using multiple sentences, corresponding to several image regions is known as **dense captioning**. Earlier approaches to Image Captioning include template-based methods [35, 89, 76], and retrieval based methods [141, 104, 41, 54]. Template based methods complete predefined caption templates using the information obtained from the images, such as a set of objects present in the image, inferring attributes of these objects and finding relationships between objects. Retrieval based methods caption an image by finding similar images from the training set and describe the image using captions of these similar images. More recently, deep learning based captioning models overcome the restrictive nature of these earlier methods and are shown to generate custom caption for each image. Deep learning methods for image captioning learn to generate captions using visual information [70, 66, 15, 117]. Supervised methods for image captioning use the image-caption pair provided in the training data to learn image captioning models. Multiple supervised deep learning approaches have been introduced for performing image captioning [158, 59, 95, 34, 148, 170, 61, 167, 145, 14, 65, 178, 181, 177, 5], including encoder decoder models, models utilizing attention, compositional models, to name a few.

Encoder decoder models consist of two main components known as encoder and decoder [158, 59, 95]. The encoder component processes images to build visual features, and the decoder component uses these visual features from the encoder to "decode" caption for the input image. Attention based models [170, 61, 167, 145, 14] generate caption by learning to recursively "attend" to multiple regions in the image, where the attention operation could be interpreted as the visual information extracted by the model to generate a particular word in the output word sequence. Compositional models [34, 148] consist of multiple modules gather different types of information about the image (e.g. objects, interactions, attributes) and generate natural language descriptions using this extracted information, additionally also consisting of modules that compose the linguistic output. Other image captioning methods include learning from auxiliary data [177, 5], image captioning using semantic understanding of images [65, 178, 181], unsupervised image captioning methods [130, 24], and reinforcement learning based methods for image captioning [117, 118, 186].

**Video Captioning** task deals with describing videos using natural language captions. Video captioning methods use a template based approach or a sequence learning approach. Template

based methods [44, 120, 72, 171] make use of fixed templates and parse the input video to complete the templates using visual properties such as objects, attributes, interactions and relationships. Sequence learning approaches [155, 129, 176, 169, 106, 38, 107, 174, 162] generate a distinct caption for each video using deep neural networks. Video captioning methods use different deep neural network models such as encoder-decoder models, attention based models or use auxiliary data for video captioning [154], similar to image captioning.

**Dense Image Captioning**, or dense captioning, generates multiple captions for an image instead of image captioning, which generates a single caption for an image. The dense captioning task involves discovering salient image regions and describing each identified salient region using a caption. Johnson et al. [64] propose a dense captioning problem and formulate a method that efficiently localizes a set of salient image regions and describe each region using an LSTM. Yang et al. [172] utilize image region-caption description correspondences and neighbourhood context for jointly predicting dense captions with localization of salient regions. Yin et al. [180] perform dense captioning by extracting contextual information and grounding attributes in captions through multi-scale message propagation.

## 2.3 Review of Network pruning

Specialized domain recognition tasks deal with small scale datasets with a fraction of label categories compared to object recognition problems involving hundreds or thousands of label categories. Therefore, to achieve high performance on specialized domain recognition tasks, we explore network pruning to efficiently perform recognition tasks with fewer parameters. Network pruning is one technique among many techniques for network compression. In this section, we review different methods to perform pruning [85, 49].

Other standard techniques to perform network compression include reducing the precision of weights from single-precision to bit-level precision, such as weight quantization [48] / weight binarization [23, 114], knowledge distillation [52, 121], structured projection of fully connected layers [16, 96, 175] to name a few. We review these methods in order to study network compression in detail.

Network pruning is one of several approaches for performing Network Compression. Lecun et al. [85] introduce a new technique called optimal brain damage to prune a deep neural network. This work aims to identify parameters that have the least impact on the objective function. For assessing the importance of parameters, it uses the second derivative of the objective function with respect to the parameters to compute their “saliencies", and finds the set of parameters that perturb the objective function by the least amount. Iteratively, they compute the second derivatives efficiently similar to backpropagation and delete the parameters with the lowest value of saliency. Experiments on handwritten digits dataset show that their computing method and using their saliency values to prune weights is more robust to approach, which uses magnitude-based weight pruning.

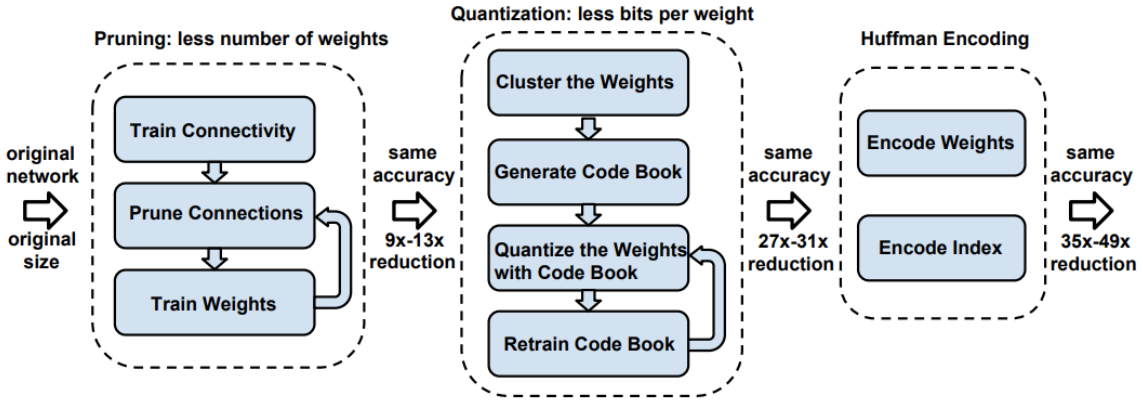



Figure 2.14: Overview of three stage compression pipeline proposed in Han et al. [48] (Fig.obtained from [48])

Hassibi et al. [49] approach the problem of network compression by using second-order derivatives to find weights that would least affect the network performance. Unlike previous methods, this work does not assume any form for the second-order derivative matrix. Once a weight is removed, this method also re-computes other weights' strength without a need for back-propagation. The method also describes a simplified method to calculate the inverse Hessian matrix by using its recursive relation to the network's structure. Experiments were conducted on multiple datasets and outperformed previous network pruning methods considerably, thereby showing the effectiveness of their method in pruning the right set of weights against other methods. Han et al. [48] perform network compression in order to enable them to run object recognition models on a smaller hardware platform. In order to do so, they propose a three stage pipeline, illustrated in Fig. 2.14, which performs redundant connection pruning, weight quantization to reduce the storage requirements and Huffman coding to make use of non-uniform distribution in this codebook to encode weights based on their frequency. Experiments show that each step in this pipeline brings down the number of unique parameters, aiding in memory storage requirements of the network while maintaining the classification accuracy intact with each pipeline stage's addition.

Srinivas et al. [140] approach network pruning in a data-free manner, where they remove neurons one at a time in a systematic fashion. They come up with heuristics to perform network pruning by measuring similarity between two neurons associated with the weight values of their connections, achieved through the computation of a "saliency" matrix and removing one of the two similar neurons. Before computing saliency matrix, they perform weight normalization, which enables them to scale all the weights to a range, which helps to remove more weights. Experiments on multiple datasets and networks suggest that the proposed approach is tolerant to aggressive pruning than previous network pruning methods.

Li et al. [86] prune filters in convolutional layers that have limited impact on accuracy. The method proposes a structured, one-shot approach to pruning weights across multiple layers, which

is efficient, especially with networks with many convolutional layers. For pruning, filters that have the smallest expected impact on activations are chosen, and network sensitivity to pruning is measured. Multiple strategies are explored for retraining pruned networks(to improve accuracy), for simultaneous pruning of multiple layers together. Experiments on multiple datasets with different networks show that the approach is effective in being able to reduce the inference costs using simple filter pruning techniques significantly.

Guo et al. [45] propose dynamic network surgery, a new method for network compression by performing on-the-fly connection pruning. The proposed method performs pruning in such a way that it is possible for the network to re-establish a previously pruned connection using splicing operation if it is deemed as necessary later. This framework lets the network prune more efficiently and increase the pruning ratio without a sharper drop in the classification accuracy. The pruning/splicing operations are performed by measuring every parameter's importance and is determined by the magnitude of their values. The pruning operation is performed independently for each layer in the network. Experiments show that the proposed method outperforms the previous state-of-the-art pruning method on multiple datasets and different networks.

Network compression has had several approaches, all of which aim to reduce the spatial size of a deep neural network. One such strategy is Network quantization [23]. In Courbariaux et al., a deep neural network consisting of binary weights is used for object recognition task, which replaces space-consuming multiply-accumulate operators in conventional deep neural networks with binary multipliers. The proposed method introduces a binarization scheme in order to achieve this objective. In this scheme, first, a network layer is binarized scholastically using a sigmoid function. Second, binarization is applied during forward and backward operations, while weight precision is maintained during parameter updates. Using this scheme, the resulting deep neural network achieves comparable recognition performance to the uncompressed neural network with a smaller network size.

Lebedev et al. [84] perform network compression by utilizing sparsity constraints. They propose a structured convolutional kernel tensor pruning strategy called group-wise pruning, which groups the tensor's entries. Multiple algorithms to perform group-sparse convolutions are explored, which differ in how they handle sparsification and finetuning. They also make use of group-sparsity regularization [184] to learn the grouping patterns automatically, which aids them in speeding up generalized convolutional operations. The proposed approach is tested on multiple datasets with standard convolutional networks, and it is shown that the proposed method outperforms previous sparsification strategies involving tensor decomposition.

Knowledge distillation [52] is another technique to perform Network Compression. In this work, Hinton et al. attempt to perform various recognition tasks efficiently, including a speech recognition task which originally uses a cumbersome model, by replacing this system with a simpler, single model while not compromising on performance. To train the smaller model, the proposed knowledge distillation method constructs an objective function that aims to predict the right groundtruth. It also has an additional term that encourages the smaller model to predict the same

class probability distribution as the cumbersome model. Experiments conducted on speech recognition and MNIST digits recognition task show that the distilled smaller model achieves competitive performance compared to cumbersome, ensemble models.

Cheng et al. [16] approach network compression by dealing with redundancies in fully connected layers. It aims to reduce the number of parameters in fully connected layers by enforcing a particular form of structure on them known as “circulant" structure, which also lets them perform speed up in computation through Fast Fourier transform. They also show that this structure manages to preserve global information capturing intact, and the classification performance is not compromised due to the structuring which led to decrease in the number of independent parameters. Experiments show that the proposed structuring reduces both the space and time complexity of the resultant model on multiple object recognition datasets and neural network architectures.

# Chapter 3

# Hierarchical Deep Temporal Models for Group Activity Recognition

## 3.1 Abstract

In this chapter, we present an approach for classifying the activity performed by a group of people in a video sequence. The problem of group activity recognition is addressed by examining individual actions and their relations. Temporal dynamics exist both at the level of individual actions and group activity. We build a deep model to capture these dynamics based on LSTM (long short-term memory) models. To model both person-level and group-level dynamics, we present a 2-stage deep temporal model for the group activity recognition problem. In our approach, we use two LSTM models, to represent action dynamics of the individuals in a video and to aggregate person-level information for group activity recognition, respectively. We collected a new dataset consisting of volleyball videos labelled with individual and group activities to evaluate our method. Experimental results on this new Volleyball Dataset and the standard benchmark Collective Activity Dataset demonstrate the efficacy of the proposed models.

## 3.2 Introduction

We could describe the action that is happening in Figure 3.1 in numerous levels of abstraction. For instance, we could describe the scene in terms of what each individual is doing. This task of person-level action recognition is an essential component of visual understanding. At another level of detail, we could instead ask what the overarching group activity that is depicted is. For example, this frame could be labelled as the “right team setting." In this chapter, we focus on this higher-level group activity task, devising methods for classifying a video according to the group activity.

Human activity recognition is a challenging computer vision problem and has received attention from the research community. It is a challenging problem due to factors such as the variability within action classes, background clutter, and similarity between different action classes, to name a few. Group activity recognition finds numerous applications in the context of video surveillance, sports analytics, video search and retrieval. A particular challenge of group activity recognition is

that the inference of the label for a scene can be quite sensitive to context. For example, in the volleyball scene shown in Fig. 3.1, the group activity hinges on the action of one key individual who is performing the “setting" action – though other people in the scene certainly provide helpful information to resolve the ambiguity. In contrast, for group activity categories such as “talking" or “queuing" (e.g. Fig. 3.4), the group activity label depends on the actions of many inter-related people in a scene. As such, successful models likely require the ability to aggregate information across the many people present in a scene and make distinctions utilizing all of this information.

Spatio-temporal relations among the people in the scene have been at the crux of several approaches in the past that dealt with group activity recognition. The literature shows that spatio-temporal appearance/motion properties of an individual and their relations can discern which group activity is present. A volume of research has explored models for this type of reasoning [18, 79, 112, 4]. These approaches utilize underlying person-level action recognition based on hand-crafted feature representations, including the histogram of gradients (HOG) or motion boundary histograms (MBH), both in a dense and sparse fashion [159], [125]. However, since they rely on shallow hand-crafted feature representation, they are limited by their representational abilities to model a complex learning task. Similarly, the higher-level group activity recognition models utilize probabilistic or discriminative models built from relatively limited components.

On the other hand, deep representations have overcome this limitation and yielded state-of-the-art results in several computer vision benchmarks [135], [67], [74]. One direct approach to group activity recognition with a deep model would be to treat an image as a holistic input. One could train a model to classify this image according to the group activity taking place. However, multiple uninteresting regions (e.g. volleyball courts) in the frame will be considered in the classification model. One way to resolve this is to learn attention models [170, 128] to highlight specific key regions in the scene. Another direction might be to detect the people in the scene and learn a model that focuses on these detections rather than the whole region of the scene (explicitly attend to specific regions). We make use of the latter direction.

The inter-class distinctions in group activity recognition arise from the variations in spatio-temporal relations between people, beyond just global appearance. Utilizing a deep model to learn invariance to translation, to model relations between individuals, presents a significant challenge to the learning algorithm. Similar challenges exist in the object recognition literature, and research often focuses on designing pooling operators for deep networks (e.g. [144]) that enable the network to learn effective classifiers.

Group activity recognition presents a similar challenge – it is necessary to model appropriate networks that allow the learning algorithm to focus on differentiating higher-level classes of activities. A simple solution to come up with such a representation is to have a layered approach in which each layer focuses on a subset of the image, and a given layer collects the information learnt from its previous layer to learn the higher-level information. Hence, we develop a novel hierarchical deep temporal model. Our model consists of one dedicated layer, which reasons about individual peo-

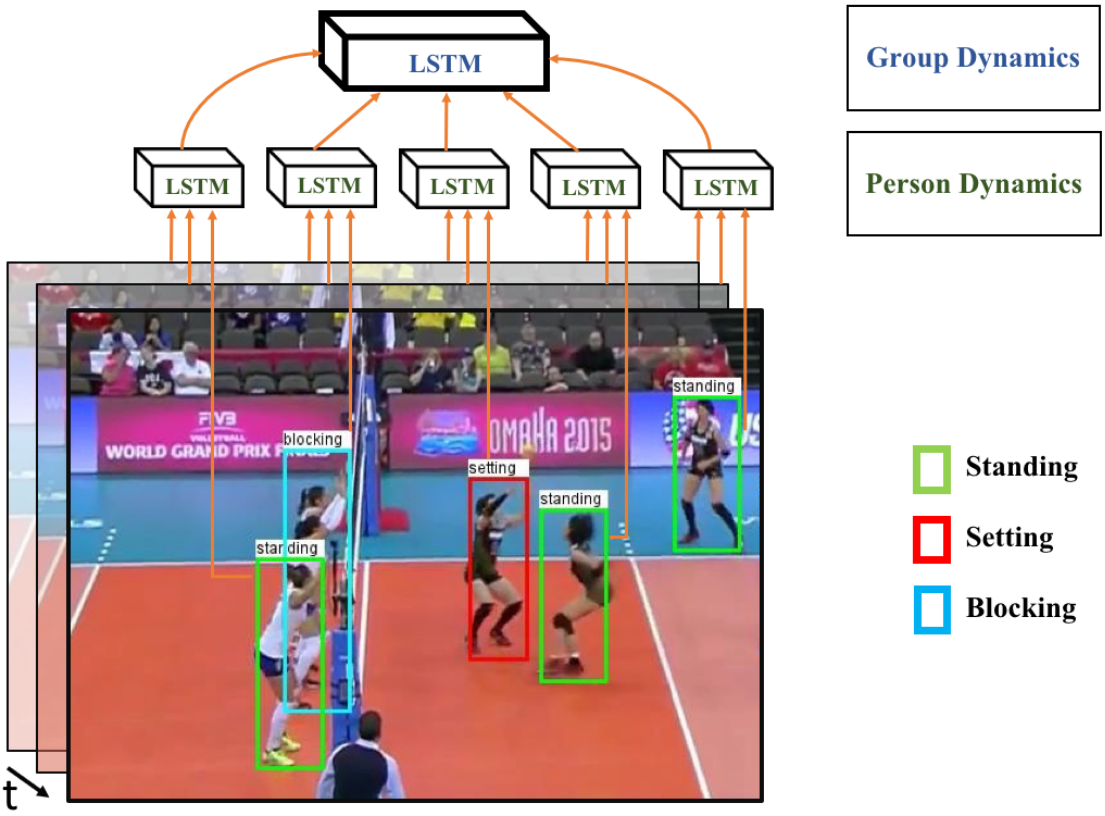



Figure 3.1: Group activity recognition via a hierarchical model. Each person in a scene is modeled using a temporal model that captures his/her dynamics. These models are integrated into a higher-level model that captures scene-level group activity.

ple and a second higher-level layer that collects the information from the previous layer and learns discriminative frame level information for group activity recognition.

Our method starts with a set of detected and tracked people. Given a set of detected and tracked people, we use deep temporal networks (LSTMs) to analyze each individual. These person-level LSTMs are aggregated over the people in a scene into a higher level deep temporal model. This allows the deep model to learn the relations between the people (and their appearances) that contribute to recognizing a particular group activity. Through this chapter, we show that we can use LSTMs as a plausible deep learning alternative to the graphical models previously used for this task.

The contribution of this chapter is our novel deep architecture that models group activities in a principled structured temporal framework. Our 2-stage approach models individual actions in its first stage, and then combines person-level information to represent group activities. The model's temporal representation is based on the long short-term memory (LSTM): recurrent neural networks such as these have recently demonstrated successful results in sequential tasks such as image captioning [32] and speech recognition [42]. Through the model structure, we aim at constructing a

representation that leverages the discriminative information in the hierarchical structure between individual actions and group activities.

We show that our algorithm works in two scenarios. First, we demonstrate performance on the Collective Activity Dataset [18], a surveillance-type video dataset. We also propose a new Volleyball Dataset with individual bounding box location annotations, the annotated individual action and group activity labels. Experimentally, the model is effective in recognizing the overall team activity based on recognizing and integrating player actions.

This chapter builds upon a previous version of this work [57]. Here, we present a modified model for alternative pooling structures, an enlarged Volleyball Dataset, and additional empirical evaluations and analyses.

This chapter is organized as follows. In Section 3.3, we provide a brief overview of the literature related to activity recognition. In Section 3.4, we elaborate details of the proposed group activity recognition model. In Section 3.5, we tabulate the performance of the approach, and end in Section 3.6 with concluding remarks.

## 3.3 Related Work

Human action recognition is an active area of research, with many existing algorithms. Surveys by Weinland et al. [165] and Poppe [108] explore the vast literature in activity recognition. Here, we will focus on the group activity recognition problem and recent related advances in deep learning.

### 3.3.1 Group Activity Recognition

Group activity recognition has attracted a large body of work recently. Most previous work has used hand-crafted features fed to structured models representing information between individuals in space-time domains. For example, Choi et al. [18] use hand-crafted spatio-temporal feature representations of relative human actions. Lan et al. [81] proposed an adaptive latent structure learning that represents hierarchical relationships ranging from lower person-level information to higher group-level interactions.

Lan et al. [81] and Ramanathan et al. [112] explore the idea of social roles, where the expected behaviour of an individual person in the context of group, is modelled in a fully supervised and weakly supervised setting respectively. Lan et al. [81] map the features defined on individuals to group activity by constructing a hierarchical model consisting of individual action, role-based unary components, pairwise roles, and scene level group activities. The interactions and unary roles/activities are represented using an undirected graphical model. The parameters of this model are learnt using a structured SVM formulation in a max margin framework, and operates under completely supervised settings.

Ramanathan et al. [112] define a CRF-based social role model under a weakly supervised setting. To learn model parameters, a joint variational inference procedure is adapted. HOG3D [71], spatio-temporal features [159], object interaction feature [87], and social role features [193] are used

as unary component representations. A subsequent layer consisting of pairwise spatio-temporal interaction features is used to refine the noisy unary component features. Finally, variational inference is used to learn the unknown role labels and model parameters.

Choi and Savarese [17] unified tracking multiple people, recognizing individual actions, interactions and collective activities in a joint framework. The model is based on the premise that correlation exists between individual actions, their neighbourhood. Following this intuition, they develop a hierarchical structure of activity types that maps the individual activity to overall group activity. In this process, they simultaneously track atomic activities, interactions and overall group activities. The parameters of this model (and the inference) are learnt by combining belief propagation with the branch and bound algorithm.

Chang et al. [13] employ a probabilistic grouping strategy to perform high-level recognition tasks happening in the scene. Specifically, group structure is determined by soft grouping structures to facilitate the representation of dynamics present in the scene. Secondly, they also use probabilistic motion analysis to extract interesting spatio-temporal patterns for scenario recognition. Vascon et al. [152] detect conversational groups in crowded scenes of people. The approach uses pairwise affinities between people based on pose and a game-theoretic clustering procedure.

Choi et al. [19] use a random forest structure to sample discriminative spatio-temporal regions from the video. They input this representation to a 3D Markov random field to localize collective activities in a scene. Shu et al. [132] detect group activities from aerial video using an AND-OR graph formalism. Lillo et al. [92] proposed a hierarchical model of multiple levels: pose level, action level and activity level. These methods use shallow hand-crafted features, and typically adopt a linear model that suffers from representational limitations.

### 3.3.2 Sport Video Analysis

Computer vision-based analysis of sports video is a burgeoning area for research, with many recent papers and workshops focused on this topic. Work on sports video analysis has spanned a range of topics from individual player detection, tracking, and action recognition, to player-player interactions, to team-level activity classifications. Much work spans many of these taxonomy elements, including the seminal work of Intille and Bobick [58], who examined stochastic representations of American football plays.

**Player tracking**: Nillius et al. [102] link player trajectories to maintain identities via reasoning in a Bayesian network formulation. Morariu et al. [97] track players, infer part locations, and reason about temporal structure in 1-on-1 basketball games. In Soomro et al. [138], a graph-based optimization technique is applied to address the task of tracking in broadcast soccer videos where a disjoint temporal sequence of soccer videos is present. They first extract panoramic view video clips, and subsequently detect and track multiple players by a two step bipartite matching algorithm. Bo et al. [11] introduced a novel approach to scale and rotation invariant tracking of human body parts. They use a dynamic programming-based approach that optimizes the assembly of body part

region proposals, given spatio-temporal constraints under a loopy body part graph construction, to enable scale and rotation invariance.

**Actions and player roles:** Turchini et al. [151] perform activity recognition by first obtaining dense trajectories [159], clustering them, and finally employ a cluster set kernel to learn the action representations. Kwak et al. [77] optimized based on a rule-based depiction of interactions between people.

Wei et al. [164] compute a role ordered feature representation to predict the ball owner at each time instance in a given video. They start from the annotated positions of each player, permute them and obtain the feature representation ordered by relative position (called as role) with respect to other players.

**Team activities:** Siddiquie et al. [133] proposed sparse multiple kernel learning to select features incorporated in a spatio-temporal pyramid. In Bialkowski et al. [10], two detection-based representations that are based on team occupancy map and team centroid map respectively, are shown to effectively detect team activities in field hockey videos. First, players are detected in each of the eight camera views that are used, and then team level aggregations are computed after classifying each player into one of the two teams. Finally, using these aggregated representations, team activity labels are computed.

Atmosukarto et al. [8] define an automated approach for recognizing offensive team formation in American football. First, the frame pertaining to the offensive team formation is first identified, line of scrimmage is obtained, and eventually the team formation label is obtained by learning a SVM classifier on top of the offensive team side's features inferred using the line of scrimmage. Direkoglu and O'Connor [31] solved a Poisson equation to generate a holistic player location representation. Swears et al. [143] used the Granger Causality statistic to automatically constrain the temporal links of a Dynamic Bayesian Network (DBN) for handball videos.

In Gade et al. [37], player occupancy heat maps are employed to handle sport type classification. People are first detected, and the occupancy maps are obtained by summing their locations over time. Finally, a sport type classifier is trained on top of Fisher vector representations of the heat maps to infer the sports type happening in a test scene.

### 3.3.3 Deep Learning

Deep Convolutional Neural Networks (CNNs) have shown impressive performance by unifying feature and classifier learning, enabled by large labelled training datasets. They have been applied successfully to multiple computer vision tasks, including image classification [74, 134] and action recognition [135, 67]. More flexible recurrent neural network (RNN)-based models are used for handling variable-length space-time inputs. Specifically, LSTM [53] models are popular among RNN models due to the tractable learning framework that they offer when it comes to deep representations. These LSTM models have been applied to a variety of tasks [32, 42, 100, 156].

For instance, in Donahue et al. [32], the so-called Long term Recurrent Convolutional network, formed by stacking an LSTM on top of pre-trained CNNs, is proposed for handling sequential tasks

such as activity recognition, image description, and video description. In this chapter, they show that it is possible to jointly train LSTMs along with convolutional networks and achieve comparable results to the state-of- the-art for time-varying tasks. For example, in video captioning, they first construct a semantic representation of the video using maximum a posteriori estimation of a conditional random field to caption a natural sentence using LSTMs. Our model stacks LSTMs in a related manner. However, in our hierarchical model, multiple LSTMs from the first layer feed their input to the 2nd layer in a per-player approach where data come from individual image regions.

In Karpathy et al. [67], structured objectives are used to align CNNs over image regions and bi-directional RNNs over sentences. A deep multi-modal RNN architecture is used for generating image descriptions using the deduced alignments. In the first stage, words and image regions are embedded onto an alignment space. Image regions are represented by RCNN embeddings, and words are represented using bi-directional recurrent neural network [126] embeddings. In the second stage, using the image regions and textual snippets, or full image and sentence descriptions, a generative model based on an RNN outputs a probability map of the next word.

Alahi et al. [2] present a separate LSTM network per pedestrian to predict his trajectory in crowded scenes. The proposed social pooling layer connects and shares information between these LSTMs. Relative to our model, we use a shared LSTM network per person in the scene with a pooling layer that fuses the different people's representations to predict the whole scene activity. Du et al. [33] also used multiple LSTMs to model action recognition from human skeletons.

In this chapter, we build a hierarchical structured model that incorporates a deep LSTM framework to recognize individual actions and group activities. Previous work in the area of deep structured learning includes Tompson et al. [146] for pose estimation, and Zheng et al. [188] and Schwing et al. [127] for semantic image segmentation.

In Deng et al. [30] a similar framework is used for group activity recognition, where a neural network-based hierarchical graphical model refines person action labels and learns to predict the group activity simultaneously. While these methods use neural network-based graphical representations, in our current approach, we leverage LSTM-based temporal modelling to learn discriminative information from time-varying sports activity data. In [179], a new dataset is introduced containing dense, multiple labels per frame for underlying action, and a novel Multi-LSTM is used to model the temporal relations between labels present in the dataset. Ramanathan et al. [111] develop LSTM-based methods for analyzing sports videos, using an attention mechanism to determine who is the principal actor in a scene. In a sense, this work is complementary to our pooling-based models that perform aggregations of all people involved in a group activity.

### 3.3.4 Datasets

Popular datasets for activity recognition include the Sports-1M dataset [67], UCF 101 database [139], and the HMDB movie database [75]. These datasets were part of a shift in focus toward unconstrained Internet videos as a domain for action recognition research. These datasets are challenging because they contain substantial intra-class variation and clutter both in extraneous background

objects and varying temporal duration of the action of interest. However, these datasets focus on individual human actions, in contrast to the group activities we consider in this chapter.

Scenes involving multiple, potentially interacting people present significant challenges. In the context of surveillance video, the TRECVid Surveillance Event Detection [105], UT-Interaction [123], VIRAT [103], and UCLA Courtyard datasets [3] are examples of challenging tasks including individual and pairwise interactions.

Datasets for analyzing group activities include the Collective Activity Dataset [18]. This dataset consists of real-world pedestrian sequences where the task is to find the high-level group activity. The S-HOCK dataset [22] focuses on crowds of spectators and contains more than 100 million annotations ranging from person body poses to actions to social relations among spectators. In this chapter, we experiment with the Collective Activity Dataset, and also introduce a new dataset for group activity recognition in sports footage which contains annotations of the player pose, location, and group activities[1].

## 3.4 Proposed Approach

Our goal in this chapter is to recognize activities performed by a group of people in a video sequence. The input to our method is a set of tracks of the people in a scene. The group of people in the scene could range from players in a sports video to pedestrians in a surveillance video. In this chapter, we consider three cues that aids what a group of people is doing:

- **Person-level actions** collectively define a group activity. Person action recognition is a first step toward recognizing group activities.
- **Temporal dynamics of a person's action** is higher-order information that can serve as a strong signal for group activity. Knowing how each person's action is changing over time is used to infer the group activity.
- **Temporal evolution of group activity** represents how a group's activity is changing over time. For example, in a volleyball game, a team may move from defence phase to pass and then attack.

Many classic approaches to the group activity recognition problem model these elements as a structured prediction based on hand crafted features [159, 125, 81, 79, 112]. Inspired by the success of deep learning-based solutions, in this chapter, a novel hierarchical deep temporal model is proposed that is capable of learning low-level image features, person-level actions, their temporal relations, and temporal group dynamics in a unified end-to-end framework.

Given the sequential nature of group activity analysis, our proposed model is based on a Recurrent Neural Network (RNN) architecture. RNNs consist of non-linear units with internal states

[1]The dataset is available for download: `https://github.com/mostafa-saad/deep-activity-rec`.

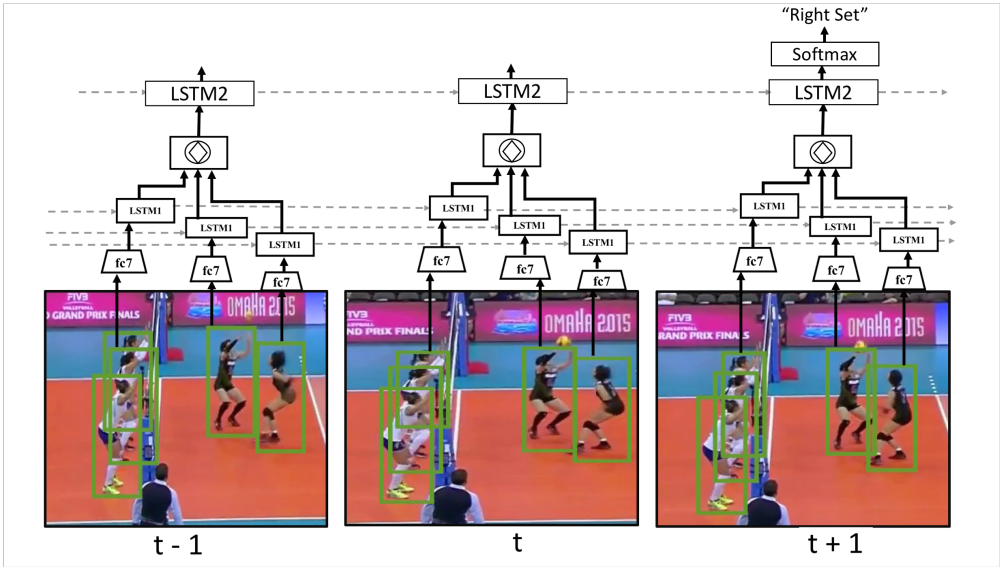



Figure 3.2: Our two-stage model for a volleyball match. Given tracklets of K players, we feed each tracklet to a CNN, followed by a person LSTM layer to represent each player's action. We then pool temporal features over all people in the scene. The output of the pooling layer is fed to the second LSTM network to identify the whole team's activity.

that can learn dynamic temporal behaviour from a sequential input with arbitrary length. Therefore, they overcome the limitation of CNNs that expect constant length input. This capability makes them widely applicable to video analysis tasks such as activity recognition.

Our model is inspired by the success of hierarchical models for group activity recognition. Here, we aim to mimic a similar intuition using recurrent networks. We propose a deep model by stacking several layers of RNN-type structures to model a range of low-level to high-level dynamics defined on top of people and entire groups. Fig. 3.2 provides an overview of our model. We describe the use of these RNN structures for individual and group activity recognition next.

### 3.4.1 Temporal Model of Individual Action

Given the tracks of each person in a scene, we use long short-term memory (LSTM) models to represent the action of each individual person temporally. Such temporal information is complementary to spatial features and is critical for performance. LSTMs have been used successfully for many sequential problems in computer vision. Each LSTM unit consists of several cells with a memory that stores information for a short temporal interval. The memory content of an LSTM makes it suitable for modelling complex temporal relationships that may span a long time range.

The content of the memory cell is regulated by several gating units that control the flow of information in and out of the cells. The control they offer also helps avoid spurious gradient updates

that can typically happen in training RNNs when the length of a temporal input is large. This property enables us to stack many such layers to learn complex dynamics present in the input in different ranges.

We use a deep Convolutional Neural Network (CNN) to extract features from the bounding box around the person in each time step on an individual trajectory. The output of the CNN, represented by $x_t$, is a complex image-based feature describing the spatial region around a person. Assuming $x_t$ as the input of an LSTM cell at time $t$, the cell activition is formulated as :

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \tag{3.1}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \tag{3.2}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \tag{3.3}$$

$$g_t = \phi(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{3.4}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{3.5}$$

$$h_t = o_t \odot \phi(c_t) \tag{3.6}$$

Here, $\sigma$ stands for a sigmoid function, and $\phi$ stands for the tanh function. $x_t$ is the input, $h_t \in \mathbb{R}^N$ is the hidden state with N hidden units, $c_t \in \mathbb{R}^N$ is the memory cell, $i_t \in \mathbb{R}^N$, $f_t \in \mathbb{R}^N$, $o_t \in \mathbb{R}^N$, and, $g_t \in \mathbb{R}^N$ are input gate, forget gate, output gate, and input modulation gate at time $t$ respectively. $\odot$ represents element-wise multiplication.

When modelling individual actions, the hidden state $h_t$ is used to model a person's action at time $t$. Note that the cell output is evolving over time based on past memory content. Due to the deployment of gates on the information flow, the hidden state will be formed based on a short-range memory of the person's past behaviour. Therefore, we can pass the output of the LSTM cell at each time to a softmax classification layer[2] to predict individual person-level action for each track.

The LSTM layer on top of person trajectories (extracted as appearance image sequence of persons) forms the first stage of our hierarchical model. The person trajectories This stage is designed to model **person-level actions and their temporal evolution**. Our training proceeds in a stage-wise fashion, first training to predict person level actions, and then passing the hidden states of the LSTM layer to the second stage for group activity recognition, as discussed in the next section.

### 3.4.2 Hierarchical Model for Group Activity Recognition

At each time step, the memory content of the first LSTM layer contains discriminative information describing the subject's action and past changes in their action. If the memory content is gathered over all the individuals in the scene to describe the whole scene's group activity.

Further, we observe that the image-based features extracted from the spatial domain around a person carry a discriminative signal for the current activity. Therefore, a deep CNN model is used

[2]More precisely, a fully connected layer fed to softmax loss layer.

to extract complex features for each person and the temporal features captured by the first LSTM layer.

The concatenation of the CNN features and the LSTM layer represent temporal features for a person. Various pooling strategies are used to aggregate these features over all people in the scene at each time step. The output of the pooling layer forms our representation for the group activity. The second LSTM network, working on top of the temporal representation, is used to model the **temporal dynamics of group activity** directly. The LSTM layer of the second network is connected to a classification layer to recognize group activity classes in a video.

Mathematically, the pooling layer can be expressed as the following:

$$P_{tk} = x_{tk} \oplus h_{tk} \tag{3.7}$$

$$Z_t = P_{t1} \diamond P_{t2} \ldots \diamond P_{tk} \tag{3.8}$$

In this equation, $h_{tk}$ corresponds to the first stage LSTM output, and $x_{tk}$ corresponds to the AlexNet fc7 feature, both obtained for the k$^{\text{th}}$ person at time t. We concatenate these two features (represented by $\oplus$) to obtain the temporal feature representation $P_{tk}$ for k$^{\text{th}}$ person. We then construct the frame-level feature representation $Z_t$ at time t by applying a max-pooling operation (represented by $\diamond$) over the features of all the people. Finally, we feed the frame-level representation to our second LSTM stage that operates similar to the person level LSTMs that we described in the previous subsection and learn the group-level dynamics. $Z_t$, passed through a fully connected layer, is given to the second-stage LSTM layer's input. The hidden state of the LSTM layer represented by $h_t^{group}$ carries temporal information for the whole group dynamics. $h_t^{group}$ is fed to a softmax classification layer to predict group activities.

### 3.4.3 Handling sub-groups

In team sports, there might be several sub-groups of players with common responsibilities within a team. For example, the front players of a volleyball team are responsible for blocking the ball. Max-pooling all players' representation in one representation reduces the model capabilities (e.g. causes confusion between the left and right team activities). To consider that, we propose a modified model where we split the players to several sub-groups and recognize the team activity based on the concatenation of each subgroup's representation. In our experiments, we consider a set of different possible spatial sub-groupings of players (c.f. standard spatial pyramids [83]). Figure 3.3 illustrates this variant of the model, showing splitting into two team-based groups. Mathematically, the pooling

layer can be re-expressed as the following:

$$P_{tk} = x_{tk} \oplus h_{tk} \tag{3.9}$$

$$S_m = (m-1) * k/d + 1 \tag{3.10}$$

$$E_m = m * k/d \tag{3.11}$$

$$G_{tm} = P_{tSm} \diamond P_{t(Sm+1)} \ldots \diamond P_{tEm} \tag{3.12}$$

$$Z_t = G_{t1} \oplus G_{t2} \ldots \oplus G_{td} \tag{3.13}$$

where again, $t$ indexes time, $k$ indexes players, $h_{tk}$ corresponds to first stage LSTM output, $x_{tk}$ to fc7 features, and $P_{tk}$ is the spatio-temporal feature representation for the player. Assume that the $K$ players are ordered in a list (e.g. based on the top-left point of a bounding box), $d$ is the number of sub-groups and $m$ indexes the groups. $S_m$ and $E_m$ are the start and end positions of the m-th group players. $G_{tm}$ is the $m$-th group representation: a max-pooling on all group players' representation in this group. $Z_t$ is the frame-level feature representation constructed by the concatenation operator (represented by $\oplus$) of the $d$ sub-groups.

### 3.4.4 Implementation Details

We trained our model in two steps. In the first step, the person-level CNN and the first LSTM layer are trained in an end-to-end fashion using a set of training data consisting of person tracks annotated with action labels. This network is shared among all people in the scene. We implement our model using Caffe [60]. Similar to other approaches [32, 30, 156], we initialize our CNN model with the pre-trained AlexNet network, and we fine-tune the whole network for the first LSTM layer.

After training the first LSTM layer, we concatenate the fc7 layer of AlexNet and the LSTM layer for every person and pool over all people in a scene. The pooled features, which correspond to frame-level features, are fed to the second LSTM network.

For training all our models, we follow the same training protocol. We use a fixed learning rate of 0.00001, a momentum of 0.9, batch size 250 and a Tesla K40C GPU (12 GB RAM) for the computations. In testing, only one network is loaded and used by the N players. Time complexity is linear in the number of people N and the target number of time steps. For tracking subjects in a scene, we used the tracker by Danelljan et al. [25], implemented in the Dlib library [69]. The baseline models are structured and trained in a similar manner to our two-stage model.

## 3.5 Experiments

In this section, we evaluate our model by running ablation studies using several baselines and comparing them to previously published works on the Collective Activity Dataset [18]. First, we describe our baseline models for ablation studies. Then, we present our results on the Collective Activity Dataset, followed by experiments on the Volleyball Dataset.

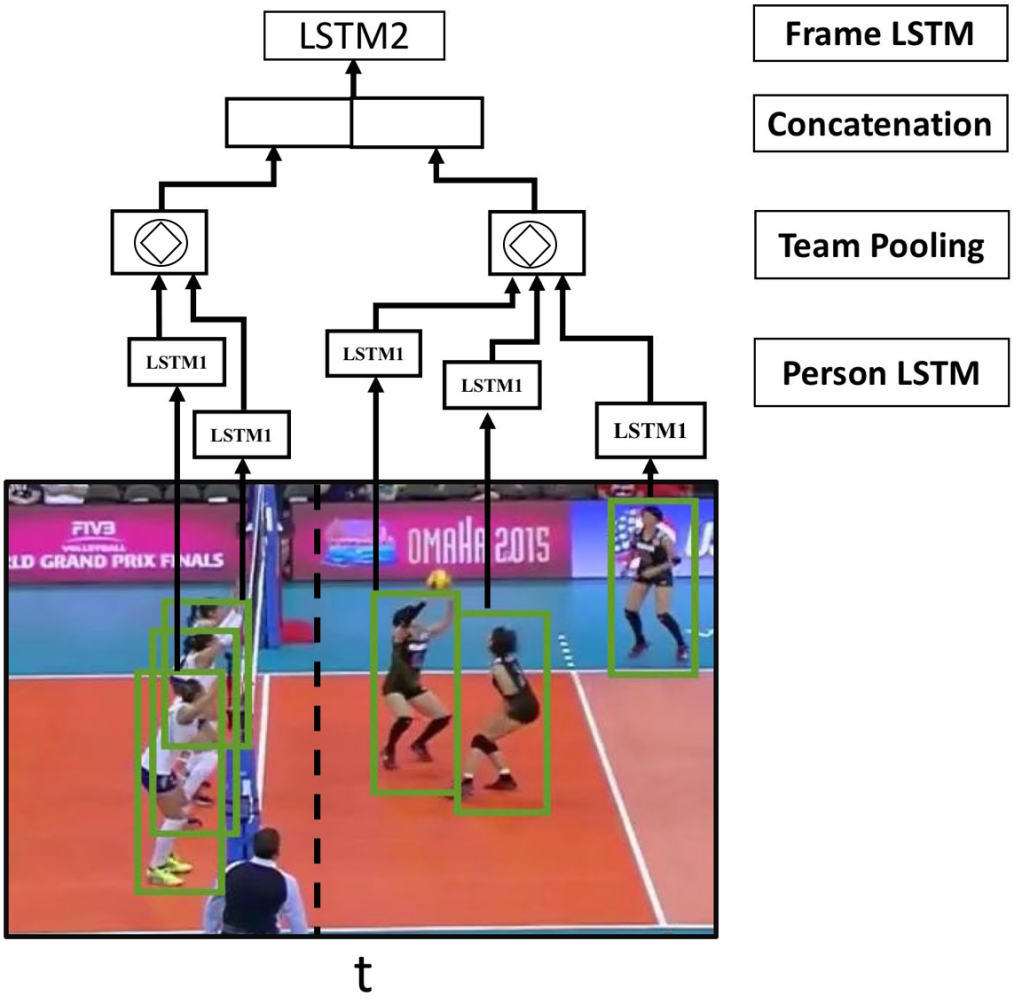



Figure 3.3: Illustration of 2-group pooling to capture spatial arrangements of players.

### 3.5.1 Baselines

The following baselines are considered in all our experiments in order to assess the contributions of components of our proposed model.

B1) **Image Classification:** This baseline is the basic AlexNet model fine-tuned for group activity recognition in a single frame.

B2) **Person Classification:** In this baseline, the AlexNet CNN model is deployed on each person, fc7 features are pooled over all people, and are input to a softmax classifier to recognize group activities in every frame.

B3) **Fine-tuned Person Classification:** This baseline is similar to the previous baseline with one distinction. The AlexNet model on each player is finetuned to recognize person-level actions. Then, fc7 features are pooled over all individuals and are input to a softmax classifier to recognize group activities at every frame. The rationale behind this baseline is to examine a scenario where person-level action annotations and group activity annotations are used in a deep learning model that does not model the temporal aspect of group activities. This baseline is very similar to our two-stage model but without the temporal modelling.

B4) **Temporal Model with Image Features:** This baseline is a temporal extension of the first baseline. It examines the idea of feeding image-level features directly to a LSTM model to recognize group activities. In this baseline, the AlexNet model is used on the whole image, and the resulting fc7 features are input to the LSTM model. This baseline is a reimplementation of Donahue et al. [32].

B5) **Temporal Model with Person Features:** This baseline is a temporal extension of the second baseline: fc7 features pooled over all people are fed to a LSTM model to recognize group activities.

B6) **Two-stage Model without LSTM 1:** This baseline is a variant of our model, omitting the person-level temporal model (LSTM 1). Instead, the person-level classification is done only with the finetuned person CNN.

B7) **Two-stage Model without LSTM 2:** This baseline is a variant of our model, omitting the group-level temporal model (LSTM 2). In other words, we do the final classification based on the outputs of the temporal models for individual action labels, but without an additional group-level LSTM.

B8) **Simple Temporal Model:** This baseline is based on a simple temporal model to evaluate the need for the LSTM as a non-trivial temporal model. AlexNet is changed to include an extra layer after fc7 of 256 nodes. The network then is fine tuned to represent person actions. Then, the 256 features are max-pooled over all people in a single frame. Finally, the features of $T$

frames are concatenated and fed to a softmax classifier to recognize group activities in this temporal clip.

### 3.5.2 Experiments on the Collective Activity Dataset

The Collective Activity Dataset [18] has been widely used for evaluating group activity recognition approaches in the computer vision literature [4, 30, 3]. This dataset consists of 44 videos, eight person-level pose labels (not used in this chapter), five person-level action labels, and five group-level activities. A scene is assigned a group activity label based on the majority of what people are doing. Data-split strategies are different between different papers (e.g. 1/4 data for testing in [4] vs 1/3 data in [30, 47]). We followed the division by [30, 47] and compare against the best performance among these. In this section, we present our results on this dataset.

**Model details:** In the Collective Activity Dataset, nine timesteps and 3000 hidden nodes are used for the first LSTM layer, and a softmax layer is used for the classification layer in this stage. The classification performance on the test set of this first, person action classification stage of our model is 65%. The second network consists of a 3000-node fully connected layer followed by a 9-timestep 500-node LSTM layer, which is input to a softmax layer trained to recognize group activity labels.

| Method | Accuracy |
|---|---|
| B1-Image Classification | 63.0 |
| B2-Person Classification | 61.8 |
| B3-Fine-tuned Person Classification | 66.3 |
| B4-Temporal Model with Image Features | 64.2 |
| B5-Temporal Model with Person Features | 64.0 |
| B6-Two-stage Model without LSTM 1 | 70.1 |
| B7-Two-stage Model without LSTM 2 | 76.8 |
| B8-Simple Temporal Model | 77.7 |
| **Two-stage Hierarchical Model** | **81.5** |

Table 3.1: Comparison of our method with baseline methods on the Collective Activity Dataset.

| Method | Accuracy |
|---|---|
| Contextual Model [81] | 79.1 |
| Deep Structured Model [30] | 80.6 |
| **Our Two-stage Hierarchical Model** | 81.5 |
| Cardinality kernel [47] | **83.4** |

Table 3.2: Comparison of our method with previously published works on the Collective Activity Dataset.

**Ablation studies:** In Table 3.1, the classification results of our overall proposed architecture is compared with the baselines. As shown in the table, our two-stage LSTM model significantly outperforms the baseline models. A comparison is made between temporal and frame-based counterparts, including B1 vs. B4, B2 vs. B5 and B3 vs. our two-stage model. We observe that adding temporal information using LSTMs improves the performance of these baselines.

**Comparison to other methods:** Table 3.2 compares our method with state-of-the-art methods for group activity recognition. Fig. 3.4 provides visualizations of example results. The performance of our two-stage model is comparable to the state-of-the-art methods. Note that only Deng et al. [30] is a previously published deep learning model. In contrast, the cardinality kernel approach [47] outperformed our model. It should be noted that this approach works on handcrafted features fed to a model highly optimized for a cardinality problem (i.e. counting the number of actions in the scene), which is exactly the way group activities are defined in this dataset.

#### Discussion

The confusion matrix obtained for the Collective Activity Dataset using our two-stage model is shown in Figure 3.7. We observe that the model performs almost perfectly for the talking and queuing classes, and gets confused between crossing, waiting, and walking. Such behaviour is perhaps due to a lack of consideration of spatial relations between people in the group, which is shown to boost previous group activity recognition methods: for instance, crossing involves the walking action, but is restricted in a path which people perform in an orderly fashion. Therefore, our model designed only to learn the dynamic properties of group activities gets confused with the walking action.

In summary, our two-stage model obtains better performance compared to the baselines. The temporal information improves performance. Further, finding and describing the elements of a video (i.e. persons) provides benefits over utilizing frame level features.

### 3.5.3 Experiments on the Volleyball Dataset

To evaluate the performance of our model for team activity recognition on sport footage, we collected a new dataset using publicly available YouTube volleyball videos. We annotated 4830 frames handpicked from 55 videos with nine player action labels and eight team activity labels. We used frames from 2/3$^{rd}$ of the videos for training, and the remaining 1/3$^{rd}$ for testing. The list of action and activity labels and related statistics are tabulated in Tables 3.3 and 3.4.

From the tables, we observe that the group activity labels are relatively more balanced compared to the player action labels. This observation follows from the fact that we often have people present in static actions like standing compared to dynamic actions (setting, spiking, to name a few). Therefore, our dataset presents a challenging team activity recognition task, where we have interesting

actions that can directly determine the group activity that rarely occurs in our dataset. The dataset is publicly available to facilitate future comparisons[3].

**Model details:** The model hyperparameters for the Volleyball Dataset include 5 timesteps and 3000 hidden nodes for the first LSTM layer. The classification performance of this first, person action classification stage of our model is 74.4%. The second network uses 10 timesteps and 2000 hidden nodes for the second LSTM layer.

We further experiment with a set of different player sub-grouping approaches for pooling. To find the sub-groups, we follow a simple strategy. First, we order players based on their top-left bounding box coordinates. To split players into two groups (e.g. left/right teams), we consider the first half of players as group one. Similarly, to split into four groups, we consider the first quarter of players as group one, the second quarter as group two, etc. If players cannot be partitioned evenly (for instance, there are missing players), the last sub-groups will have fewer players.

| Group Activity Class | # Instances |
|---|---|
| Right set | 644 |
| Right spike | 623 |
| Right pass | 801 |
| Right winpoint | 295 |
| Left winpoint | 367 |
| Left pass | 826 |
| Left spike | 642 |
| Left set | 633 |

Table 3.3: Statistics of the group activity labels in the Volleyball Dataset.

[3] `https://github.com/mostafa-saad/deep-activity-rec`

| Action Class | # Instances |
|---|---|
| Waiting | 3601 |
| Setting | 1332 |
| Digging | 2333 |
| Falling | 1241 |
| Spiking | 1216 |
| Blocking | 2458 |
| Jumping | 341 |
| Moving | 5121 |
| Standing | 38696 |

Table 3.4: Statistics of the action labels in the Volleyball Dataset.

**Ablation studies:** In Table 3.5, the classification performance of our main proposed model is compared against the baselines. Similar to the performance in the Collective Activity Dataset, our two-stage LSTM model outperforms the baseline models.

Moreover, explicitly modelling people is necessary for obtaining better performance in this dataset, since the background is rapidly changing due to a fast-moving camera, and therefore, it corrupts the foreground's temporal dynamics. This could be verified from the performance of our baseline model B4, which is a temporal model that does not consider people explicitly, showing inferior performance compared to the baseline B1, which is a non-temporal image classification style model. On the other hand, baseline model B5, which is a temporal model that explicitly considers people, performs comparably to the image classification baseline, despite the problems that arise due to tracking and motion artifacts.

| Method | Accuracy |
|---|---|
| B1-Image Classification | 66.7 |
| B2-Person Classification | 64.6 |
| B3-Fine-tuned Person Classification | 68.1 |
| B4-Temporal Model with Image Features | 63.1 |
| B5-Temporal Model with Person Features | 67.6 |
| B6-Two-stage Model without LSTM 1 | 74.7 |
| B7-Two-stage Model without LSTM 2 | 80.2 |
| B8-Simple Temporal Model | 78.1 |
| **Our Two-stage Hierarchical Model** | **81.9** |

Table 3.5: Comparison of the team activity recognition performance of baselines against our model evaluated on the Volleyball Dataset. Experiments are using 2 group styles with max pool strategy.

In both the datasets, an observation from the tables is that while both LSTMs contribute to overall classification performance, having the first layer LSTM (B7 baseline) is relatively more critical to the performance of the system, compared to the second layer LSTM (B6 baseline).

To further investigate players sub-grouping, in Table 3.6, we run experiments over 4 sub-groups: left-team-back players, left-team-front players, right-team-back players and front-team-bottom players. We also show max-pooling results versus the avg pooling operator.

The results demonstrate that brute force sub-grouping does not improve the performance of the system. It shows that extracting additional information by segregating players based on their position renders information from static/insignificant players results in more confusion, leading to degradation in performance. Therefore, from this experiment, it is evident that not all types of explicit spatio-temporal relation modelling lead to an improvement in performance. One might expect the average pooling to perform better than the max-pooling operator in the sense that average pooling is utilizing all players while max pooling is only picking the one with the highest values. One interpretation might be that a few players might be the key actors in a scene, and the max-pooling identifies them. Similar observations are contained in contemporary work by Ramanathan

et al. [111], which used average weighting (an attention-based model) for player representation to detect the key players in basketball games.

| Method | Accuracy |
|---|---|
| Our Model - 1 group - max pool | 70.3 |
| Our Model - 1 group - avg pool | 68.5 |
| Our Model - 2 groups - max pool | **81.9** |
| Our Model - 2 groups - avg pool | 80.7 |
| Our Model - 4 groups - max pool | 81.5 |
| Our Model - 4 groups - avg pool | 79.6 |

Table 3.6: Comparison of the team activity recognition of our model using 2 sub-groups vs. 4 sub-groups with both average and max pooling.

To evaluate the effect of the number of LSTM nodes of the model's two networks, we conducted a set of experiments outlined in Table 3.7. Similarly, we evaluate the effect of the number of timesteps of the model's two networks. The results are outlined in Table 3.8.

| Method | No. Person LSTM Nodes | No. Scene LSTM Nodes | Accuracy |
|---|---|---|---|
| Our Model | 1000 | 1000 | 79.4 |
| Our Model | 2000 | 1000 | 80.3 |
| Our Model | 3000 | 1000 | 81.2 |
| Our Model | 3000 | 2000 | **81.9** |
| Our Model | 3000 | 3000 | 81.2 |

Table 3.7: Comparison of the team activity recognition of our model using 2 groups style over different numbers of LSTM nodes in the second, group-level LSTM layer.

| Method | No. Person timesteps | No. Scene timesteps | Accuracy |
|---|---|---|---|
| Our Model | 5 | 10 | **81.9** |
| Our Model | 5 | 20 | 81.7 |
| Our Model | 10 | 10 | 81.7 |
| Our Model | 10 | 20 | 81.3 |

Table 3.8: Comparison of the team activity recognition of our model using 2 groups style over different number of timesteps in the model 2 networks

**Comparison to other methods:** In Table 3.9, we compare our model to the improved dense trajectory approach [160]. Dense trajectories are a hand-crafted approach that competes strongly versus deep learning features. In addition, we also created two variations of [160], where the considered trajectories are only the ones inside the players' bounding boxes, in other words, ignoring

background trajectories. The variations emulate our model with one group and two groups style. That is, the first variation represents the players from the whole team, while the second represents each team and then concatenates the two representations to get the whole scene representation.

| Method | Accuracy |
|---|---|
| Our Model using 2 groups style | **81.9** |
| IDTF [160] - all trajectories | 73.4 |
| IDTF [160] - 1 group-box trajectories | 71.7 |
| IDTF [160] - 2 groups-box trajectories | 78.7 |

Table 3.9: Comparison of the team activity recognition of our model against improved dense trajectory approach approach.

The traditional dense trajectories approach and its one group style show similar but relatively low performance, whereas the 2-group trajectories variation yield higher performance. Perhaps, this is due to the reduction of confusion between the left and the right team distinction. However, our model outperforms these dense trajectory-based baseline methods.

**Discussion**

Figure 3.8 shows the confusion matrix obtained for the Volleyball Dataset using our two-stage model by grouping all players (no sub-groups) in one representation using max-pooling operation, similar to [57]. From the confusion matrix, we observe that our model generates accurate high-level action labels. Nevertheless, our model has some confusion between *left winpoint* and *right winpoint* activities. Contrary to [57], the confusion between set and pass activities is resolved, probably due to using more data.

Figures 3.9 shows the confusion matrix obtained for the Volleyball Dataset using our two-stage model, but by sub-grouping left team and right team first. From the confusion matrix, we observe that our model generates more accurate high level action labels than using no groups. In addition, the confusion between *left winpoint* and *right winpoint* activities is reduced.

In Figure 3.10, we show the visualizations of our detected activities with different failure and success scenarios.

## 3.6 Conclusion

In this chapter, we presented a novel deep structured architecture to deal with the group activity recognition problem. Through a two-stage process, we learn a temporal representation of person-level actions and combine individual people's representation to recognize group activity. We created a new Volleyball Dataset to train and test our model. We also evaluated our model on the Collective Activity Dataset. Results show that our architecture can improve upon baseline methods lacking hierarchical consideration of individual and group activities using deep learning.

The model could be trained with much less training data without significant loss in performance. For example, if training annotations for the person level are not available, we can use a pre-trained network for representing people. However, we would expect the model's performance to degrade. Further, at testing time, one can learn an object detector from training data (e.g. Faster-RCNN) and use in testing. Overall, the model itself is flexible and can be adjusted to scenarios with varying data availability.


Figure 3.4: Visualizations of the generated scene labels from the Collective Activity Dataset using our model. Green denotes correct classifications, red denotes incorrect. The incorrect ones correspond to the confusion between different actions in ambiguous cases (h and j examples), or in the cases where there is an anomalous camera zoom.

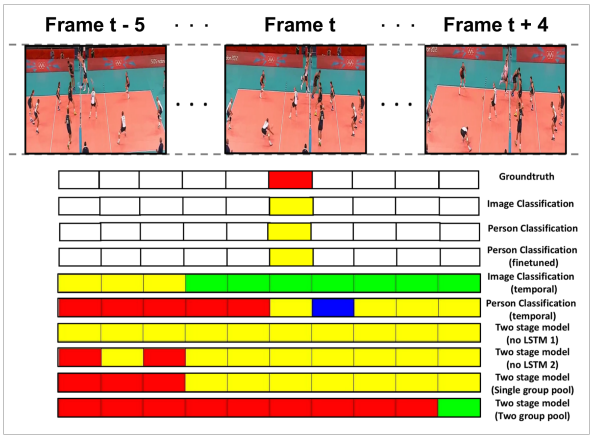



Figure 3.5: Visualization of the generated labels by different baselines/models for a sample video extracted from the Volleyball Dataset. In this figure, yellow, red, blue and green colors denote the right spike, left pass, left spike, and left set group activities respectively.

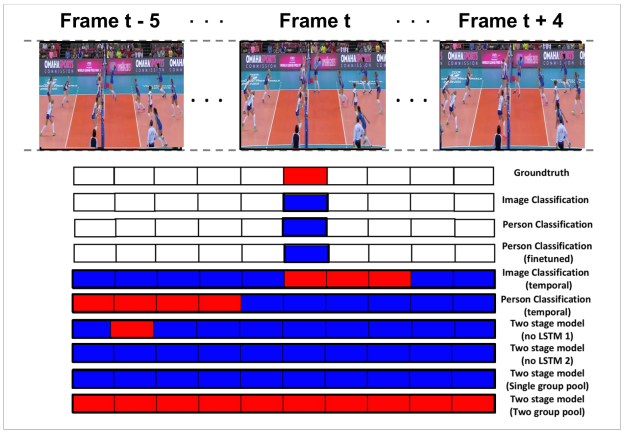



Figure 3.6: Visualization of the generated labels by different baselines/models for another sample video extracted from the Volleyball Dataset. In this figure, red, blue colors denote the right spike and right set group activities respectively.


Figure 3.7: Confusion matrix for the Collective Activity Dataset obtained using our two-stage model.

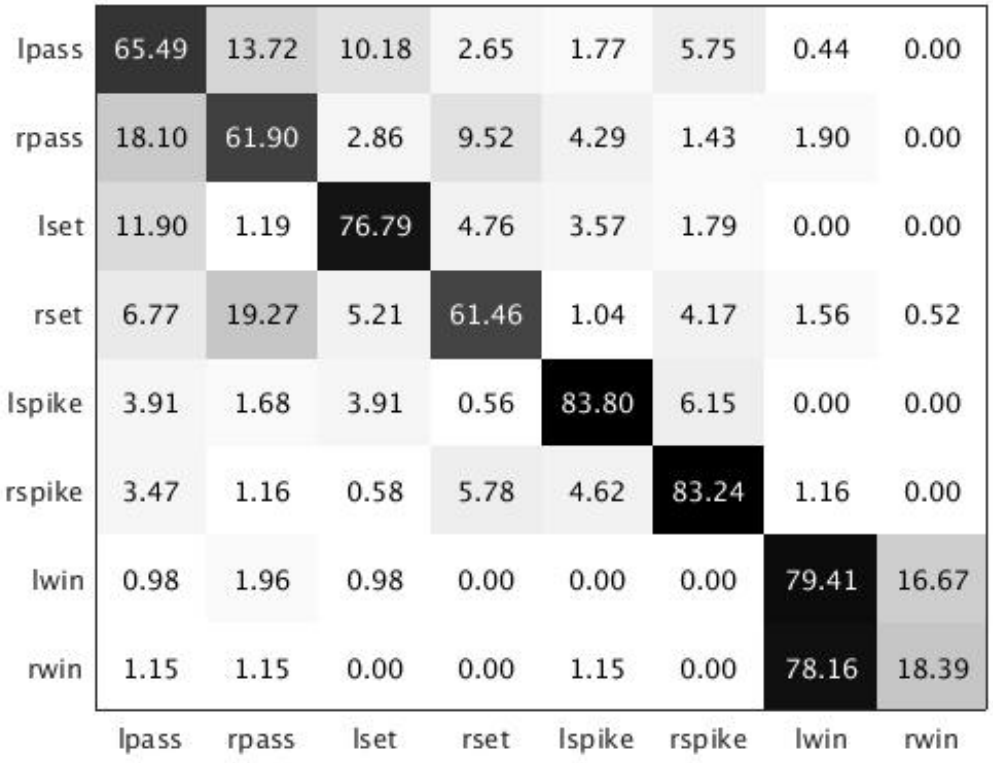



Figure 3.8: Confusion matrix for the Volleyball Dataset obtained using our two-stage hierarchical model, using 1 group style for all players.


Figure 3.9: Confusion matrix for the Volleyball Dataset obtained using our two-stage hierarchical model, using 2 groups style.


Figure 3.10: Visualizations of the generated scene labels from the Volleyball Dataset using our model. Green denotes correct classifications, red denotes incorrect. The incorrect ones correspond to the confusion between different actions in ambiguous cases (h and j examples), or in the left and right distinction (i example).

# Chapter 4

# Deep Structured Models For Group Activity Recognition

## 4.1 Abstract

This chapter presents a deep neural-network-based hierarchical graphical model for individual and group activity recognition in surveillance scenes. Deep networks are used to recognize the actions of individual people in a scene. Next, a neural-network-based hierarchical graphical model refines the predicted labels for each class by considering dependencies between the classes. This refinement step mimics a message-passing step similar to inference in a probabilistic graphical model. We show that this approach can be effective in group activity recognition, with the deep graphical model improving recognition rates over baseline methods.

## 4.2 Introduction

Event understanding in videos is a key element of computer vision systems in the context of visual surveillance, human-computer interaction, sports interpretation, and video search and retrieval. Therefore events, activities, and interactions must be represented in such a way that retains all of the important visual information in a compact and rich structure. Accurate detection and recognition of atomic actions of each individual person in a video is the primary component of such a system, and also the most important, as it affects the performance of the whole system significantly. Although there are many methods to determine human actions in uncontrolled environments, this task remains a challenging computer vision problem, and robust solutions would open up many useful applications. The standard and yet state-of-the-art pipeline for activity recognition and interaction description consists of extracting hand-crafted local feature descriptors either densely or at a sparse set of interest points (e.g., HOG, MBH, ...) in the context of a Bag of Words model [160]. These are then used as the input either to a discriminative or a generative model. In recent years, it has been shown that deep learning techniques can achieve state-of-the-art results for a variety of computer vision tasks including action recognition [135, 67].

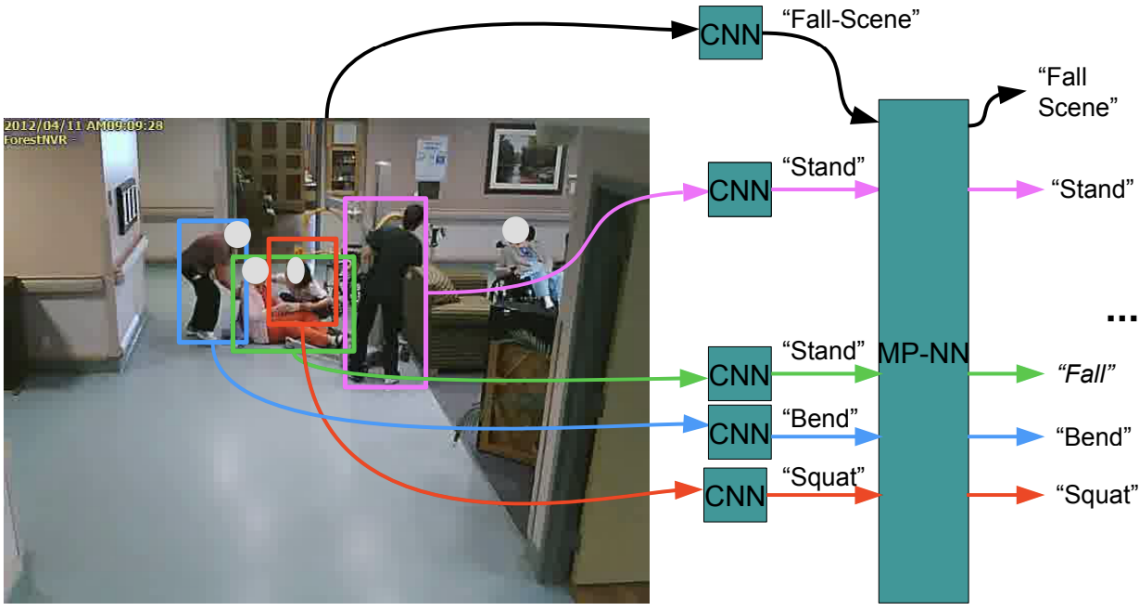



Figure 4.1: Recognizing individual and group activities in a deep network. Individual action labels are predicted via CNNs. Next, these are refined through a message passing neural network which considers the dependencies between the predicted labels.

On the other hand, understanding complex visual events in a scene requires exploitation of richer information rather than individual atomic activities, such as recognizing local pairwise and global relationships in a social context and interaction between individuals and/or objects [12, 79, 112, 124, 194]. This complex scene description remains an open and challenging task. It shares all of the difficulties of action recognition, interaction modeling, and social event description. Formulating this problem within the probabilistic graphical models framework provides a natural and powerful means to incorporate the hierarchical structure of group activities and interactions [80, 79]. Given that deep neural networks can achieve very competitive results on the single person activity recognition tasks, they can produce better results when they are combined with other methods, e.g. graphical models, to capture the dependencies between the variables of interest [146]. Following a similar idea of incorporating spatial dependency between variables into the deep neural network in a joint-training process presented [146], here we focus on learning interactions and group activities in a surveillance scene by employing a graphical model in a deep neural network paradigm.

In this chapter, our primary goal is to address the problem of *group activity understanding* and *scene classification* in complex surveillance videos using a deep learning framework. More specifically, we focus on learning individual activities and describing the scene simultaneously while considering the pair-wise interactions between individuals and their global relationship in the scene. This is achieved by combining a Convolutional Network (ConvNet) with a probabilistic graphical model as additional layers in a deep neural network architecture into a unified learning framework. The probabilistic graphical models can be seen as a refining process for predicting class labels

by considering dependencies between individual actions, body poses, and group activities. The probabilistic graphical model is modelled by a multi-step message passing neural network, and the predicted label refinement is carried out through belief propagation layers in the neural network. Figure 4.1 depicts an overview of our approach for label refinement. Experimental results show the effectiveness of our algorithm in both activity recognition and scene classification.

## 4.3 Related Work

The analysis of human activities is an active area of research. Decades of research on this topic have produced a diverse set of approaches and a rich collection of activity recognition algorithms. Readers can refer to recent surveys such as Poppe et al. [108] and Weinland et al. [165] for a review. Many approaches concentrate on an activity performed by a single person, including state of the art deep learning approaches [67, 135].

In the context of scene classification and group activity understanding, many approaches use a hierarchical representation of activities and interactions for collective activity recognition [79]. They have been focused on capturing spatio-temporal relationships between visual cues either by imposing a richer feature descriptor which accounts for context [18, 149] or a context-aware inference mechanism [4, 17]. Hierarchical graphical models [4, 79, 81, 124], AND-OR graphs [3, 46], and dynamic Bayesian networks [194] are among the representative approaches for group activity recognition.

In traditional approaches, local-hand crafted features/descriptors have been employed to recognize atomic activities. Recently, it has been shown that the use of deep neural networks can by itself outperform other algorithms for atomic activity recognition. However, no prior art in the CNN-based video description used activities and scene information jointly in a unified graphical representation for scene classification. Therefore, the main objective of this research is to develop a system for activity recognition and scene classification which simultaneously uses the action and scene labels in a neural network-based graphical model to refine the predicted labels via a multiple-step message passing.

More closely related to our approach is work combining graphical models with convolutional neural networks [27, 146]. In Tompson et al. [146], a one-step message passing is implemented as a convolution operation in order to incorporate spatial relationship between local detection responses for human body pose estimation. In another study, Deng et al. [27] propose an interesting solution to improve label prediction in large scale classification by considering relations between the predicted class labels. They employ a probabilistic graphical model with hard constraints on the labels on top of a neural network in a joint training process. In essence, our proposed algorithm follows a similar idea of considering dependencies between predicted labels for the actions, group activities, and the scene label to solve the group activity recognition problem. Here we focus on incorporating those dependencies by implementing the label refinement process via an inter-activity neural network, as

shown in Figure 4.2. The network learns the message passing procedure and performs inference and learning in a unified framework using back-propagation.

## 4.4 Model

Considering the architecture of our proposed structured label refinement algorithm for group activity understanding (see Figure 4.2), the key part of the algorithm is a multi-step message passing neural network. In this section, we describe how to combine neural networks and graphical models by mimicking a message passing algorithm and how to carry out the training procedure.

### 4.4.1 Graphical Models in a Neural Network

Graphical models provide a natural way to hierarchically model group activities and capture the semantic dependencies between group and individual activities [80]. A graphical model defines a joint distribution over states of a set of nodes. For instance, one can use a factor graph, in which each $\phi_i$ corresponds to a factor over a set of related variable nodes $x_i$ and $y_i$, and models interactions between these nodes in a log-linear fashion:

$$P(X,Y) \propto \prod_i \phi_i(x_i, y_i) \propto exp(\sum_k w_k f_k(x,y)) \tag{4.1}$$

where $X$ are the inputs and $Y$ the predicted labels, with weighted ($w_k$) feature functions $f_k$.

When performing inference in a graphical model, belief propagation is often adopted as a way to infer states or probabilities of variables. In the belief propagation algorithm, each step of message passing first collects relevant information from connected nodes to a factor node, which represents the joint distribution (dependencies) over states, then passes these messages to variable nodes by marginalizing over states of irrelevant variables.

Following this idea, we mimic the message passing process by representing each combination of states as a neuron in a neural network, denoted as a “factor neuron". While normal message passing calculates dependencies rigidly, a factor neuron can be used to learn and predict dependencies between states and pass messages to variable nodes. In the setting of neural networks, this dependency representation becomes more flexible and can adopt varied types of neurons (linear, ReLU, Sigmoid, to name a few). Moreover, by integrating graphical models into a neural network, the formulation of a graphical model allows for parameter sharing in the neural network, which not only reduces the number of free parameters to learn but also accounts for semantic similarities between factor neurons. Fig. 4.3 shows the parameter sharing scheme for different factor neurons.

### 4.4.2 Message Passing CNN Architecture for Group Activity

Representing group activities and individual activities as a hierarchical graphical model is shown to be successful [3, 17, 80]. We adopt a similar structured model that considers group activity, individual activity, and group-individual interactions together. We introduce a new message passing

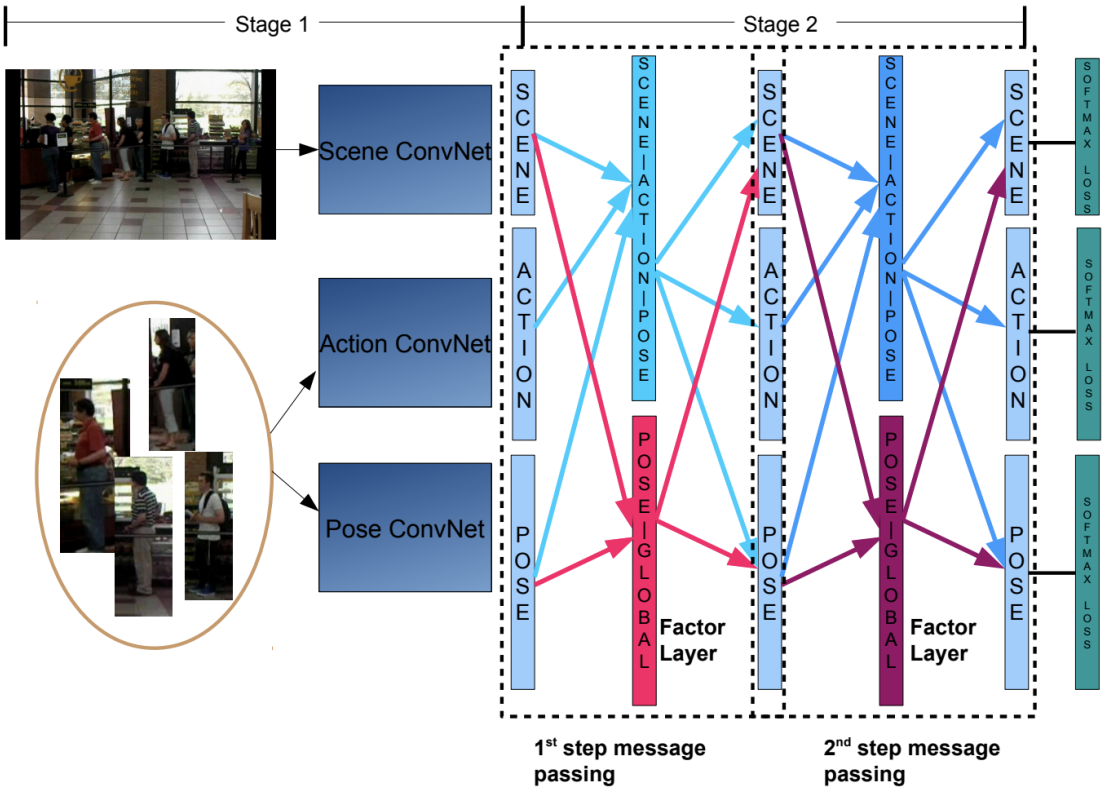



Figure 4.2: A schematic overview of our message passing CNN framework. Given an image frame and the detected bounding boxes around each person, our model predicts scores for individual actions and the group activities. The predicted labels are refined by applying a belief propagation-like neural network. This network considers the dependencies between individual actions and body poses, and the group activity. The model learns the message passing parameters and performs inference and learning in unified framework using backpropagation.

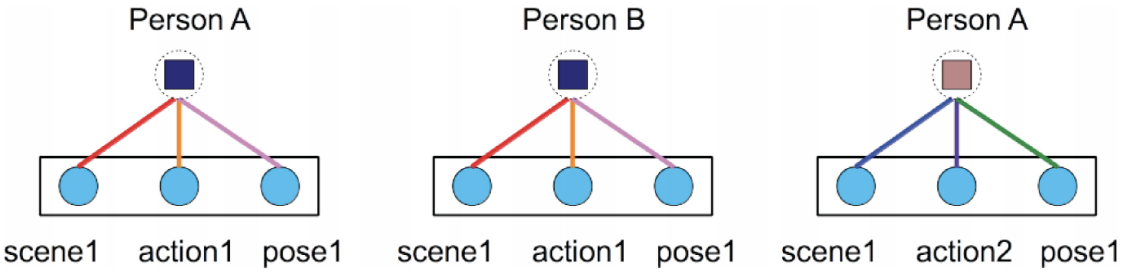



Figure 4.3: Weight sharing scheme in neural network. We use a sparsely connected layer to represent message passing between variable nodes and factor nodes. Each factor node only connects to its relevant nodes. And factor nodes of same type share a template of parameters. For example, factor node 1 and 2 gathers information from a scene's scene1, a person's action1 and pose1, and share one template of parameters. And factor node 3 adopts another set of weights.

Convolutional Neural Network framework as shown in Fig. 4.2. Our model has two main stages: (1) fine-tuned Convolutional Neural Networks that produce scene scores for a frame, and action and pose scores for each person in that frame; (2) Message Passing Neural Network phase capturing dependencies.

Given an image $I$ and a set of person detections $I_1, I_2, ..., I_M$, the first stage of our model outputs raw scores of scene, action and poses for image $I$ and all detections $I_i$ in the image using fine-tuned CNNs. After a softmax normalization for each scene and person, these raw scores are taken as input of the graphical model part in the second stage. In the graphical model, outputs from CNNs correspond to unary potentials. Denote the scenelevel, and per-person action and pose-level unary potentials for frame $I$ as $s^{(0)}(I), a^0(I_m), r^{(0)}(I_m)$ respectively. The superscript $(0)$ is the index of message passing steps. We use $G$ to denote all group activity labels, $H$ to represent all the action labels and $Z$ to denote all the pose labels. Then the group activity in one scene can be represented as $g_I$, $\{h_{I_1}, h_{I_2}, ..., h_{I_M}\}$,$\{z_{I_1}, z_{I_2}, ..., z_{I_M}\}$ where $g_I \in G$ is the group activity label for image $I$, $h_{I_i}$ and $z_{I_i}$ are action labels and pose labels for a person $I_m$.

Note that for training, the scene, action, and pose CNN models in stage 1 are fine-tuned from an AlexNet architecture pretrained using ImageNet data. The architecture is similar to that proposed by [74] for object classification with some minor differences such as pooling is done before normalization. The network consists of five convolutional layers followed by two fully connected layers, and a softmax layer that outputs individual class scores. We use the softmax loss, stochastic gradient descent and dropout regularization to train these three ConvNets.

In the second stage, we use the method mentioned in Sec. 4.4.1 to mimic message passing in a hierarchical graphical model for group activity in a scene. This stage can contain several steps of message passing. In each step, there are two types of passes: from outputs of step $k-1$ to factor layer and from factor layer to $k$ step outputs. In the $k^{th}$ message passing step, the first pass computes dependencies between states. . The inputs to the $k^{th}$ step message passing are $\{s_1^{(k-1)}(I), ....s_{|G|}^{(k-1)}(I), a_1^{(k-1)}(I_1), ....a_{|H|}^{(k-1)}(I_M), r_1^{(k-1)}(I_1), ....r_{|Z|}^{(k-1)}(I_M)\}$, where $s_g^{(k-1)}(I)$ is the scene score of image $I$ for label $g$, $a_h^{(k-1)}(I_m)$ is the action score of person $I_m$ for label $h$ and $r_z^{(k-1)}(I_m)$ is the pose score of person $I_m$ for label $z$. In the factor layer, the action, pose and scene interaction are calculated as:

$$\phi_j(s_g^{(k-1)}(I), a_h^{(k-1)}(I_m), r_z^{(k-1)}(I_m)) = \alpha_{g,h,z}[s_g^{(k-1)}(I), a_h^{(k-1)}(I_m), r_z^{(k-1)}(I_m)]^T \tag{4.2}$$

where $\alpha_{g,h,z}$ is a 3-d parameter template for combination of scene $g$, action $h$ and pose $z$. Similarly, pose interactions for all people in the scene are calculated as:

$$\psi_j(s_g^{(k-1)}(I), \mathbf{r}) = \beta_{tg}[s_g^{(k-1)}(I), \mathbf{r}]^T \tag{4.3}$$

where $\mathbf{r}$ is all output nodes for all people, $t$ is the factor neuron index for scene $g$. $T$ latent factor neurons are used for a scene $g$. Note that parameters $\alpha$ and $\beta$ are shared within factors that have the

same semantic meaning. For the output of $k^{th}$ step message passing, the score for the scene label to be $g$ can be defined as:

$$s_g^{(k)}(I) = s_g^{(k-1)}(I) + \sum_{j \in \epsilon_1^s} w_{ij}\phi_j(s_g^{(k-1)}(I), \mathbf{a}, \mathbf{r}; \alpha) + \sum_{j \in \epsilon_2^s} w_{ij}\psi_j(s_g^{(k-1)}(I), \mathbf{r}; \beta) \tag{4.4}$$

where $\epsilon_1^s$ and $\epsilon_2^s$ are the set of factor nodes that connected with scene $g$ in first factor component(scene-action-pose factor) and second factor component (pose-global factor) respectively. Similarly, we also define action and pose scores after the $k^{th}$ message passing step as:

$$a_h^{(k)}(I_m) = a_h^{(k-1)}(I_m) + \sum_{j \in \epsilon_1^a} w_{ij}\phi_j(a_h^{(k-1)}(I_m), \mathbf{s}, \mathbf{r}; \alpha) \tag{4.5}$$

$$r_z^{(k)}(I_m) = r_z^{(k-1)}(I_m) + \sum_{j \in \epsilon_1^r} w_{ij}\phi_j(r_z^{(k-1)}(I_m), \mathbf{a}, \mathbf{s}; \alpha) + \sum_{j \in \epsilon_2^r} w_{ij}\psi_j(r_z^{(k-1)}(I_m), \mathbf{r}; \beta) \tag{4.6}$$

Note that $\epsilon = \{\epsilon_1^s, \epsilon_2^s, \epsilon_1^a, \epsilon_1^r, \epsilon_2^r\}$ are connection configurations in the pass from factor neurons to output neurons. These connections are simply the reverse of the configurations in the first pass, from input to factors. The model parameters $\{W, \alpha, \beta\}$ are weights on the edges of the neural network. Parameter $W$ represents the concatenation of weights connected from factor layers to output layer (second pass), while $\alpha, \beta$ represent weights from the input layer of the $k^{th}$ message passing to factor layers (first pass).

**Components in Factor Layers**

Now we explain in detail the different components in our model.

**Unary component**: In our message passing model, the unary component corresponds to group activity scores for an image $I$, action and pose scores for each person $I_m$ in frame $I$, represented as $s_g^{(k-1)}(I)$, $a_h^{(k-1)}(I_m)$ and $r_z^{(k-1)}(I_m)$ respectively. These scores are acquired from the previous step of message passing and are directly added to the next message passing step's output.

**Group activity-action-pose factor layer** $\phi$: A group's activity is strongly correlated to the participating individuals' actions. This component for the model is used to measure the compatibility between individuals and groups. An individual's activity can be described by both pose and action, and we use this ternary scene-pose-action factor layer to capture dependencies between a person's fine-grained action (e.g. talking facing front-left) and the scene label for a group of people. Note that in this factor layer we used the weight sharing scheme mentioned in Sec. 4.4.1 to mimic the belief propagation.

**Poses-all factor layer** $\psi$: Pose information is very important in understanding a group activity. For example, when all people are looking in the same direction, there is a high probability that it's a *queueing* scene. This component captures this global pose information for a scene. Instead of naively enumerating all combinations of poses for all individuals, we exploit the sparsity of

beneficial and frequent patterns, and use $T$ factor nodes for one scene label. In our experiments, we set $T$ to be 10.

### 4.4.3 Multi Step Message Passing CNN Training

The steps of message passing depend on the structure of the graphical model. In general, graphical models with loops or a large number of levels will lead to more steps belief propagation for sharing local information globally. In our model, we adopt two message passing steps, as shown in Fig. 4.2.

**Multi-loss training**: Since our model's goal is to recognize group activities through global features and individual actions in that group, we adopt an alternative strategy for training the model. For the $k^{th}$ message passing step, we first remove the loss layers for actions and poses to learn parameters for group activity classification alone. In this phase, there is no back-propagation on action and pose classification. Since group activity heavily depends on an individual's activity, we then fix the softmax loss layer for scene classification and learn the model for actions and poses. The trained model is used for the next message passing step. Note that in each message passing step, we exploit the benefit of the neural network structure and jointly trained the whole network.

**Learning semantic features for group activity**: Traditional convolutional neural networks mainly focus on learning features for basic classification or localization tasks. However, in our proposed message passing CNN deep model, we not only learn features but also learn high-level semantic features for better representing group activities and interactions within the group. We explore different layers' features for this deep model, and results show that these semantic features can be used for better scene understanding and classification.

**Implementation details**: Firstly, in practice, it is not guaranteed that every frame has the same number of detections. However, the structure of neural network should be fixed. To solve this problem, denoting $M_{max}$ as the maximum number of people contained in one frame, we do dummy-image padding when the number of people is less than $M_{max}$. Then we filter out these dummy data by de-activating neurons connected with them in related layers. Secondly, After the first message passing step, instead of directly feeding the raw scores into the next message passing step, we first normalize the pose and action scores for each person and scene scores for one frame by a softmax layer, converting to probabilities similar to belief propagation.

## 4.5 Experiments

Our models are implemented using the Caffe library [60] by defining two types of sparsely connected and weight shared inner product layers. One is from variable nodes to factor nodes, another is the reverse direction. We used TanH neurons as the non-linearity of these two layers. To examine the performance of our model, we test our model for scene classification on two datasets: (1) Collective Activity [18], (2) a nursing home dataset consisting of surveillance videos collected from a nursing home.

We trained an RBF kernel SVM on features extracted from the graphical model layer after each step of message passing model. These SVMs are used to predict scene labels for each frame, the standard task in these datasets. We compare our model which performs two steps of message passing refinement to baseline which does a single step message passing. We note here that the performance after two steps saturated, perhaps because the interactions in our tasks can be sufficiently captured in two steps of refinement.

### 4.5.1 Collective Activity Dataset

The Collective Activity Dataset contains 44 video clips acquired using low resolution handheld cameras. Every person is assigned one of the following five action labels: crossing, waiting, queuing, walking and talking and one of the eight pose labels: right, front-right, front, front-left, left, back-left, back, back-right. Each frame is assigned one of the following five activities: crossing, waiting, queueing, walking, and talking. The activity category is attained by taking the majority of actions happening in one frame while ignoring the poses. We adopt the standard training test split used in [80].

In the Collective Activity dataset experiment, we further concatenate the global features for a scene with AC descriptors by HOG features [80]. We averaged AC descriptors features for all people and use this feature to serve as additional global information, as this feature does not truly participate in the message passing process. This additional global information assists in classification with the limited amount of training data available for this dataset[1].

We summarize the comparisons of activity classification accuracies of different methods in Table 4.1. The current best result using spatial information in graphical model is 79.1%, from Lan et al. [80], which adopted a latent max-margin method to learn graphical model with optimized structure. Our classification accuracies (the best is 80.6%) are competitive compared with the state-of-the-art methods. However, the benefits of the message passing are clear. Through each step of the message passing, the factor layer effectively captured dependencies between different variables and passing messages using factor neurons results in a gain in classification accuracy. Some visualization results are shown in Fig 4.4.

| | 1 Step MP | 2 Step MP |
|---|---|---|
| Pure DL | 73.6% | 78.4% |
| SVM+DL Feature | 75.1% | 80.6% |

| | |
|---|---|
| Latent Constituent [6] | 75.1% |
| Contextual model [80] | 79.1% |
| **Our Best Result** | **80.6%** |

Table 4.1: Scene classification accuracy on the Collective Activity Dataset.

[1]Scene classification accuracy solely using AlexNet is 48%.

### 4.5.2 Nursing Home Dataset

This dataset consists of 80 videos obtained in a nursing home, which has different kinds of rooms such as dining rooms, corridors, to name a few. The 80 surveillance videos are recorded at 640 by 480 pixels at 24 frames per second, and contain diverse actions and frequent cluttered scenes. This dataset contains typical actions include walking, standing, sitting, bending, squatting, and falling. For this dataset, the goal is to detect falling people. Therefore, we assign each frame one of two activity categories: fall and non-fall. A frame is assigned “fall" if any person falls and “non-fall" otherwise. Note that many frames are challenging, and others may occlude the falling person in the scene. We adopted a standard 2/3 and 1/3 training test split. In order to remove redundancy, we sampled 1 out of every 10 frames for training and evaluation. Since this dataset has a large intra-class diversity within actions, we used the action primitive-based detectors proposed in [78] for more robust detection results.

Note that since this dataset has no pose attribute, we used one scene-action factor layer to perform the two-step message passing. For the SVM classifier, only deep learning features are used. We summarize the comparisons of activity classification accuracies of different methods in Table 4.2.

| Ground Truth | Pure DL | SVM+DL Fea. |
|---|---|---|
| 1 Step MP | 82.5% | 82.3% |
| 2 Steps MP | 84.1% | 84.7% |

| Detection | Pure DL | SVM+DL Fea. |
|---|---|---|
| 1 Step MP | 74.4% | 76.5% |
| 2 Steps MP | 75.6% | 77.3% |

Table 4.2: Scene classification accuracy on the nursing home dataset.

The scene classification accuracy on the Nursing Home dataset by using a baseline AlexNet model is 69%. The results on scene classification for each step also show gains. Note that in this dataset, accuracy on the second message passing gains an increase of around 1.5% for both pure deep learning or SVM prediction. We believe that this is due to the fact that the dataset only contains two scene labels, fall or non-fall, so scene variables are not as informative as scenes in the Collective Activity Dataset.

## 4.6 Conclusion

We presented a deep learning model for group activity recognition, which jointly captures the group activity, the individual actions, and the interactions between them. We propose a way to combine graphical models with a deep network by mimicking the message passing process to perform inference. We successfully applied this model to real scene surveillance videos and showed its effectiveness in recognizing group activities.


Figure 4.4: Results visualization for our model. Green tags are ground truth, yellow tags are predicted labels. From left to right is without message passing, first step message passing and second step message passing

# Chapter 5

# Fine-Pruning: Joint Fine-Tuning and Compression of a Convolutional Network with Bayesian Optimization

## 5.1 Abstract

When approaching a novel visual recognition problem in a specialized image domain, a common strategy is to start with a pre-trained deep neural network and fine-tune it to the specialized domain. If the target domain covers a smaller visual space than the source domain used for pre-training (e.g. ImageNet), the fine-tuned network is likely to be over-parameterized. However, applying network pruning as a post-processing step to reduce the memory requirements has drawbacks: fine-tuning and pruning are performed independently; pruning parameters are set once and cannot adapt over time; and the highly parameterized nature of state-of-the-art pruning methods make it prohibitive to manually search the pruning parameter space for deep networks, leading to coarse approximations. We propose a principled method for jointly fine-tuning and compressing a pre-trained convolutional network that overcomes these limitations. Experiments on two specialized image domains (remote sensing images and describable textures) demonstrate the validity of the proposed approach.

## 5.2 Introduction

Convolutional neural networks (CNNs) have been widely adopted for visual analysis tasks such as image classification [74, 134], object detection [93, 115], action recognition [135, 147], place recognition [7, 189], 3D shape classification [63, 109], image colorization [187], and camera pose estimation [68]. CNNs learn rich image and video representations that have been shown to generalize well across vision tasks.

When faced with a recognition task in a novel domain or application, a common strategy is to start with a CNN pre-trained on a large dataset, such as ImageNet [122], and fine-tune the network to the new task (Fig. 5.1a). Fine-tuning involves adapting the structure of the existing network to enable the new task, while re-using the pre-trained weights for the unmodified layers of the network.

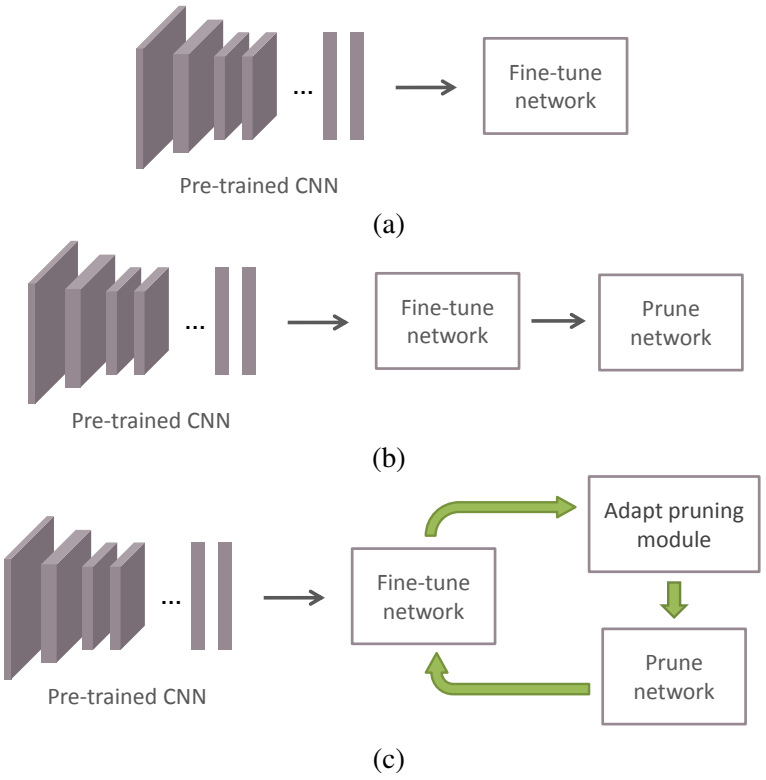



Figure 5.1: Consider the task of training a deep convolutional neural network on a specialized image domain (e.g. remote sensing images). (a) The conventional approach starts with a network pre-trained on a large, generic dataset (e.g. ImageNet) and fine-tunes it to the specialized domain. (b) Since the specialized domain spans a narrower visual space, the fine-tuned network is likely to be over-parameterized and can be compressed. A natural way to achieve this is to perform network pruning after fine-tuning, however this approach has limitations (see discussion in Section 6.2). (c) Fine-pruning addresses these limitations by jointly fine-tuning and compressing the pre-trained network in an iterative process. Each iteration consists of network fine-tuning, pruning module adaptation, and network pruning.

For example, a common and simple form of fine-tuning involves replacing the final fully-connected layer of the pre-trained CNN, which has an output dimensionality-based on the pre-training dataset (e.g. 1000 dimensions for ImageNet), with a new fully-connected layer with a dimensionality that matches the target dataset.

Fine-tuning allows powerful learned representations to be transferred to novel domains. Typically, we fine-tune complex network architectures that have been pre-trained on large databases containing millions of images. For example, we may fine-tune AlexNet [74] pre-trained on ImageNet's 1.2 million images (61 million parameters). In this way, we adapt these complex architectures to smaller and more specialized domains, such as remote sensing images. However, the specialized domain may not span the full space of natural images on which the original network was pre-trained. This suggests that the network architecture may be over-parameterized, and therefore inefficient in terms of memory and power consumption, with respect to the more constrained

novel domain, in which a much more lightweight network would suffice for good performance. In applications with tight constraints on memory and power, such as mobile devices or robots, a more lightweight network with comparable classification accuracy may be valuable.

Given a fine-tuned network, a straightforward way to obtain a more lightweight network is to perform network pruning [45, 48, 140] (Fig. 5.1b). However, this strategy has drawbacks: (1) the fine-tuning and pruning operations are performed independently; (2) the pruning parameters are set once and cannot adapt after training has started; and (3) since state-of-the-art pruning methods are highly parameterized, manually searching for good pruning hyperparameters is often prohibitive for deep networks, leading to coarse pruning strategies (e.g. pruning convolutional and fully connected layers separately [45]).

We propose a novel process called *fine-pruning* (Fig. 5.1c) that addresses these limitations:

1. Fine-pruning obtains a lightweight network specialized to a target domain by jointly fine-tuning and compressing the pre-trained network. The compatibility between the target domain and the pre-training domain is not normally known in advance (e.g. how similar are remote sensing images to ImageNet?), making it difficult to determine a priori how effective knowledge transfer will be, how aggressively compression can be applied, and where compression efforts should be focused. The knowledge transfer and network compression processes are linked and inform each other in fine-pruning.

2. Fine-pruning applies a principled adaptive network pruning strategy guided by Bayesian optimization, which automatically adapts the layer-wise pruning parameters over time as the network changes. For example, the Bayesian optimization controller might learn and execute a gradual pruning strategy in which network pruning is performed conservatively and fine-tuning restores the original accuracy in each iteration; or the controller might learn to prune aggressively at the outset and reduce the compression in later iterations (e.g. by splicing connections [45]) to recover accuracy.

3. Bayesian optimization enables efficient exploration of the pruning hyperparameter space, allowing all layers in the network to be considered together when making pruning decisions.

## 5.3 Related Work

**Network pruning.** Network pruning refers to the process of reducing the number of weights (connections) in a pre-trained neural network. The motivation behind this process is to make neural networks more compact and energy efficient for operation on resource constrained devices such as mobile phones. Network pruning can also improve network generalization by reducing overfitting. The earliest methods [49, 85] prune weights based on the second-order derivatives of the network loss. Data-free parameter pruning [140] provides a data-independent method for discovering and removing entire neurons from the network. Deep compression [48] integrates the complementary techniques of weight pruning, scalar quantization to encode the remaining weights with

fewer bits, and Huffman coding. Dynamic network surgergy [45] iteratively prunes and splices network weights. The novel splicing operation allows previously pruned weights to be reintroduced. Weights are pruned or spliced based on thresholding their absolute value. All weights, including pruned ones, are updated during backpropagation.

**Other network compression strategies.** Network pruning is one way to approach neural network compression. Other effective strategies include weight binarization [23, 114], architectural improvements [55], weight quantization [48], sparsity constraints [84, 190], guided knowledge distillation [52, 121], and replacement of fully connected layers with structured projections [16, 96, 175]. Many of these network compression methods can train compact neural networks from scratch, or compress pre-trained networks for testing in the same domain. However, since they assume particular types of weights, mimic networks trained in the same domain, or modify the network structure, most of these methods are not easily extended to the task of fine-tuning a pre-trained network to a specialized domain.

In this chapter, we consider joint fine-tuning and network pruning in the context of transferring the knowledge of a pre-trained network to a smaller and more specialized visual recognition task. Previous approaches for compressing pre-trained neural networks aim to produce a compact network that performs as well as the original network *on the dataset on which the network was originally trained*. In contrast, our focus is on the fine-tuning or *transfer learning* problem of producing a compact network for a small, specialized target dataset, given a network pre-trained on a large, generic dataset such as ImageNet. Our approach does not require the source dataset (e.g. ImageNet) on which the original network was trained.

## 5.4 Method

Each fine-pruning iteration comprises three steps: fine-tuning, adaptation of the pruning module, and network pruning (Fig. 5.1c). Fine-pruning can accommodate any parameterized network pruning module. In our experiments, we use the state-of-the-art dynamic network surgery method [45] for the network pruning module, but fine-pruning does not assume a particular pruning method. Pruning module adaptation is guided by a Bayesian optimization [39, 136] controller, which enables an efficient search of the joint pruning parameter space, learning from the outcomes of previous exploration. This controller allows the pruning behaviour to change over time as connections are removed or formed.

Bayesian optimization is a general framework for solving global minimization problems involving blackbox objective functions:

$$\min_{\mathbf{x}} \ \ell(\mathbf{x})\,, \tag{5.1}$$

where $\ell$ is a blackbox objective function that is typically expensive to evaluate, non-convex, may not be expressed in closed form, and may not be easily differentiable [163]. Eq. 5.1 is minimized by constructing a probabilistic model for $\ell$ to determine the most promising candidate $\mathbf{x}^*$ to evaluate

**Algorithm 1** Fine-Pruning

**Require:** Pre-trained convolutional network, importance weight $\lambda$

Fine-tune network {▷ Fig. 5.1a}
**repeat**
  **repeat** {▷ Bayesian optimization controller}
    Select next candidate parameters to evaluate as $\mathbf{x}^* = \arg\max_{\hat{\mathbf{x}}} \text{EI}(\hat{\mathbf{x}})$
    Evaluate $\ell(\mathbf{x}^*)$
    Update Gaussian process model using $(\mathbf{x}^*, \ell(\mathbf{x}^*))$
  **until** converged or maximum iterations of Bayesian optimization reached
  Prune network using best $\mathbf{x}^*$ found
  Fine-tune network
**until** converged or maximum iterations of fine-pruning reached

next. Each iteration of Bayesian optimization involves selecting the most promising candidate $\mathbf{x}^*$, evaluating $\ell(\mathbf{x}^*)$, and using the data pair $(\mathbf{x}^*, \ell(\mathbf{x}^*))$ to update the probabilistic model for $\ell$.

In our case, $\mathbf{x}$ is a set of pruning parameters. For example, if the network pruning module is deep compression [48], $\mathbf{x}$ consists of the magnitude thresholds used to remove weights; if the network pruning module is dynamic network surgery [45], $\mathbf{x}$ consists of magnitude thresholds as well as cooling function hyperparameters that control how often the pruning mask is updated. We define $\ell$ by

$$\ell(\mathbf{x}) = \varepsilon(\mathbf{x}) - \lambda \cdot s(\mathbf{x}) \,, \tag{5.2}$$

where $\varepsilon(\mathbf{x})$ is the top-1 error on the held-out validation set obtained by pruning the network according to the parameters $\mathbf{x}$ and then fine-tuning; $s(\mathbf{x})$ is the sparsity (proportion of pruned connections) of the pruned network obtained using the parameters $\mathbf{x}$; and $\lambda$ is an importance weight that balances accuracy and sparsity, which is set by held-out validation (we set $\lambda$ to maximize the achieved compression rate while maintaining the held-out validation error within a tolerance percentage, e.g. 2%).

We model the objective function as a Gaussian process [113]. A Gaussian process is an uncountable set of random variables, any finite subset of which is jointly Gaussian. Let $\ell \sim \mathcal{GP}(\mu(\cdot), k(\cdot,\cdot))$, where $\mu(\cdot)$ is a mean function and $k(\cdot,\cdot)$ is a covariance kernel such that

$$\begin{aligned} \mu(\mathbf{x}) &= \mathbb{E}\left[\ell(\mathbf{x})\right] \,, \\ k(\mathbf{x}, \mathbf{x}') &= \mathbb{E}\left[(\ell(\mathbf{x}) - \mu(\mathbf{x}))(\ell(\mathbf{x}') - \mu(\mathbf{x}'))\right] \,. \end{aligned} \tag{5.3}$$

Given inputs $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ and function evaluations $\ell(\mathbf{X}) = \{\ell(\mathbf{x}_1), \ell(\mathbf{x}_2), ..., \ell(\mathbf{x}_n)\}$, the posterior belief of $\ell$ at a novel candidate $\hat{\mathbf{x}}$ can be computed in closed form. In particular,

$$\tilde{\ell}(\hat{\mathbf{x}}) \sim \mathcal{N}\left(\tilde{\mu}_\ell(\hat{\mathbf{x}}), \tilde{\Sigma}^2_\ell(\hat{\mathbf{x}})\right) \,, \tag{5.4}$$

where

$$\tilde{\mu}_\ell(\hat{\mathbf{x}}) = \mu(\hat{\mathbf{x}}) + k(\hat{\mathbf{x}}, \mathbf{X})k(\mathbf{X}, \mathbf{X})^{-1}(\ell(\mathbf{X}) - \mu(\mathbf{X}))\,, \tag{5.5}$$
$$\tilde{\Sigma}^2_\ell(\hat{\mathbf{x}}) = k(\hat{\mathbf{x}}, \hat{\mathbf{x}}) - k(\hat{\mathbf{x}}, \mathbf{X})k(\mathbf{X}, \mathbf{X})^{-1}k(\mathbf{X}, \hat{\mathbf{x}})\,.$$

The implication of the closed form solution is that, given a collection of parameters and the objective function evaluated at those parameters, we can efficiently predict the posterior at unevaluated parameters.

To select the most promising candidate to evaluate next, we use the expected improvement criterion. Let $\mathbf{x}^+$ denote the best candidate evaluated so far. The expected improvement of a candidate $\hat{\mathbf{x}}$ is defined as

$$\mathrm{EI}(\hat{\mathbf{x}}) = \mathbb{E}\left[\max\left\{0, \ell(\mathbf{x}^+) - \tilde{\ell}(\hat{\mathbf{x}})\right\}\right], \tag{5.6}$$

For a Gaussian process, the expected improvement of a candidate can also be efficiently computed in closed form. Specifically,

$$\mathrm{EI}(\hat{\mathbf{x}}) = \tilde{\Sigma}_\ell(\hat{\mathbf{x}})(Z\Phi(Z) + \phi(Z))\,, \tag{5.7}$$
$$Z = \frac{\tilde{\mu}_\ell(\hat{\mathbf{x}}) - \ell(\mathbf{x}^+)}{\tilde{\Sigma}_\ell(\hat{\mathbf{x}})}\,,$$

where $\Phi$ is the standard normal cumulative distribution function and $\phi$ is the standard normal probability density function. For a more detailed discussion on Gaussian processes and Bayesian optimization, we refer the interested reader to [39], [113], and [136]. We use the publicly available code of [39] and [113] in our implementation.

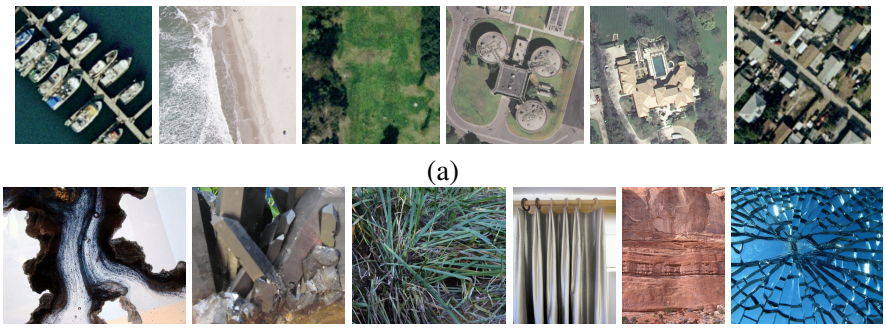
The complete fine-pruning process is summarized in Algorithm 1.

## 5.5 Experiments

**Datasets.** We performed experiments on two specialized image domains:

- Remote sensing images: The UCMerced Land Use Dataset [173] is composed of public domain aerial orthoimagery from the United States Geological Survey. The dataset covers 21 land-use classes, such as agricultural, dense residential, golf course, and harbor. Each land-use class is represented by 100 images. We randomly split the images into 50% for training, 25% for held-out validation, and 25% for testing.

- Describable textures: The Describable Textures Dataset [21] was introduced as part of a study in estimating human-describable texture attributes from images, which can then be used to improve tasks such as material recognition and description. The dataset consists of 5,640 images covering 47 human-describable texture attributes, such as blotchy, cracked,



(a)

(b)

Figure 5.2: Sample images from the two specialized domain datasets used in our experiments: (a) Remote sensing images from the UCMerced Land Use Dataset [173]; (b) Texture images from the Describable Textures Dataset [21].

crystalline, fibrous, and pleated. We use the ten provided training, held-out validation, and testing splits.

Fig. 5.2 shows examples of images from the two datasets.

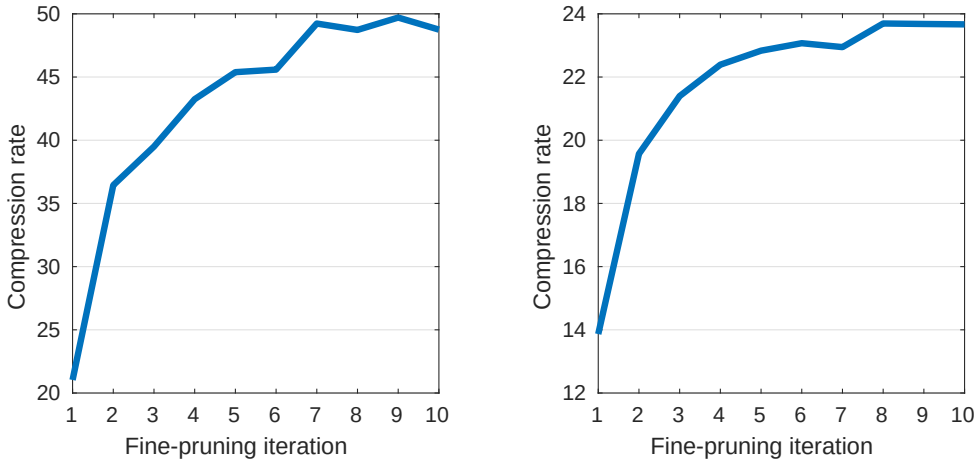
**Baselines.** We compare fine-pruning with a fine-tuning only baseline (Fig. 5.1a) as well as independent fine-tuning followed by pruning (Fig. 5.1b), which for brevity we will refer to as the independent baseline. All experiments start from an ImageNet-pretrained AlexNet [74]. For a controlled comparison, we run the same state-of-the-art pruning method, dynamic network surgery [45], in both the independent baseline and fine-pruning. In the original dynamic network surgery paper [45], the authors prune convolutional and fully connected layers separately due to the prohibitive complexity of manually searching for layer-wise pruning parameters. To more fairly illustrate the benefit of fine-pruning, we set layer-wise pruning parameters for dynamic network surgery in the independent baseline using Bayesian optimization as well.

**Implementation details.** We set all parameters by held-out validation on the two datasets. The importance weight $\lambda$ is set to 1 on both datasets. We warm-start both the independent baseline and fine-pruning with identical parameters obtained by random search. Fine-pruning is run to convergence or to a maximum of 10 iterations. In each fine-pruning iteration, Bayesian optimization considers up to 50 candidates and network fine-tuning is performed with a fixed learning rate of 0.001 (the same learning policy used to obtain the initial fine-tuned network) to 10 epochs.

**Results.** Table 5.1 summarizes our experimental comparison of fine-tuning, independent fine-tuning and pruning, and fine-pruning, on the UCMerced Land Use and Describable Textures datasets. On UCMerced Land Use, the independent baseline produces sparse networks with 1.78 million parameters on average over ten runs, representing a reduction in the number of weights by 31.9-fold, while maintaining the test accuracy within 1% of the dense fine-tuned network. Fine-pruning achieves further improvements in memory efficiency, producing sparse networks of 1.17 million parameters on average, or a 48.8-fold reduction in the number of weights, while maintaining the test accuracy within 1% of the dense fine-tuned network. On Describable Textures, we average

| | Accuracy (Val.) | Accuracy (Test) | Parameters | Compression Rate |
|---|---|---|---|---|
| UCMerced Land Use Dataset [173] | | | | |
| Fine-tuning only (Fig. 5.1a) | 94.7% | 94.3% | 57.0 M | – |
| Independent fine-tuning and pruning (Fig. 5.1b) | 92.7±0.7% | 93.8±0.7% | 1.78±0.41 M | 31.9 × |
| Fine-pruning (Fig. 5.1c) | 92.5±0.9% | 94.1±0.6% | 1.17±0.39 M | **48.8** × |
| Describable Textures Dataset [21] | | | | |
| Fine-tuning only (Fig. 5.1a) | 53.5±0.8% | 53.7±0.9% | 57.1 M | – |
| Independent fine-tuning and pruning (Fig. 5.1b) | 52.8±1.2% | 53.4±1.5% | 3.62±0.54 M | 15.8 × |
| Fine-pruning (Fig. 5.1c) | 53.0±0.9% | 52.8±0.8% | 2.41±0.68 M | **23.7** × |

Table 5.1: Experimental results on two specialized image domains: remote sensing images and describable textures. All experiments start with ImageNet-pretrained AlexNet [74] and use the state-of-the-art dynamic network surgery method [45] for network pruning. For a fair comparison, the pruning parameters in the independent fine-tuning and pruning baseline are also tuned by Bayesian optimization. We average results over ten runs on the remote sensing dataset and the ten provided splits on the describable textures dataset.




(a) (b)

Figure 5.3: Compression as a function of fine-pruning iteration. On both the (a) UCMerced Land Use Dataset and (b) Describable Textures Dataset, the pruning module adaptation, guided by Bayesian optimization, learns a policy of starting with a strong initial prune and tapering off in later iterations.

the results over the ten provided splits. Similar improvements are obtained on this harder dataset. The independent baseline reduces the number of weights by 15.8-fold while maintaining the test accuracy within 1% of the dense fine-tuned network. Fine-pruning lifts the compression rate to 23.7-fold while maintaining the test accuracy within 1% of the dense fine-tuned network.

Fig. 5.3 shows how the compression rate varies with the fine-pruning iteration. We observe that, on both datasets, the pruning module adaptation learns to start with a strong initial prune and then gradually increase pruning aggressiveness in later iterations until the network converges. This behaviour can also be observed by examining the pruning parameters $\mathbf{x}^*$ selected by Bayesian optimization.

Table 5.2 illustrates the average number of weights layer by layer after fine-pruning for both datasets. We observe that the original fine-tuned networks in both cases are highly over-parameterized, and a significant reduction in memory can be obtained by fine-pruning. A large proportion of the original network parameters reside in the fully connected layers fc6 and fc7. Provided that the underlying network pruning module allows for pruning parameters to be set on an individual layer basis, our Bayesian optimization controller automatically learns to prioritize the compression of these layers because they have the largest influence on $s(\mathbf{x})$ in the objective function (Eq. 5.2).

## 5.6 Conclusion

In this chapter, we have presented a joint process for network fine-tuning and compression that produces a memory-efficient network tailored to a specialized image domain. Our process is guided by a Bayesian optimization controller that allows pruning parameters to adapt over time to the charac-

| | Parameters: Before | Parameters: After | Percentage Pruned |
|---|---|---|---|
| UCMerced Land Use Dataset [173] | | | |
| conv1 | 35 K | 26 K | 26.1% |
| conv2 | 307 K | 92 K | 70.2% |
| conv3 | 885 K | 261 K | 70.5% |
| conv4 | 664 K | 218 K | 67.2% |
| conv5 | 443 K | 181 K | 59.1% |
| fc6 | 37.8 M | 313 K | 99.2% |
| fc7 | 16.8 M | 60 K | 99.6% |
| fc8 | 86 K | 17 K | 80.3% |
| total | 57.0 M | 1.17 M | 98.0% |
| Describable Textures Dataset [21] | | | |
| conv1 | 35 K | 32 K | 8.3% |
| conv2 | 307 K | 245 K | 20.4% |
| conv3 | 885 K | 343 K | 61.3% |
| conv4 | 664 K | 442 K | 33.4% |
| conv5 | 443 K | 216 K | 51.2% |
| fc6 | 37.8 M | 401 K | 98.9% |
| fc7 | 16.8 M | 661 K | 96.1% |
| fc8 | 193 K | 72 K | 62.4% |
| total | 57.1 M | 2.41 M | 95.8% |

Table 5.2: Layer-wise compression results. Our Bayesian optimization controller automatically learns to prioritize the compression of the fc6 and fc7 layers, which have the most parameters.

teristics of the changing network. Fine-pruning is general and can accommodate any parameterized network pruning algorithm.

# Chapter 6

# Memory-Augmented Recurrent Neural Networks for Dense Video Captioning

## 6.1 Abstract

Dense video captioning is a challenging computer vision task that involves effectively understanding long video sequences. In this work, we address this problem by augmenting recurrent neural network architectures with external memory. We propose a dense captioning model that incorporates external memory augmentation both to encode video and densely caption it. We demonstrate that recurrent video encoder and dense captioner networks augmented with external memory can be used to effectively encode frames based on the content of the entire video, as well as for generating dense captions better than recurrent networks without external memory augmentation. We conduct experiments on the ActivityNet Captions and YouCook II datasets to demonstrate the potential of external memory augmentation.

## 6.2 Introduction

Describing the content of a video in natural language is a fundamental artificial intelligence problem with many applications, such as video search, video summarization, and acessibility for the visually impaired. In the task of dense video captioning, we are given video input that consists of multiple events, often chronologically related to each other, and the goal is to detect these events and describe each of them using a natural language sentence.

Numerous methods involving recurrent neural networks (RNNs) have been proposed to address this task [161, 91, 168]. RNNs can be used either for video encoding or captioning the events in these videos, or for both of these purposes. While RNNs are shown to be effective at sequence understanding, understanding long sequences is still a difficult problem.

We present a novel architecture for dense video captioning based on memory-augmented neural networks. A large and sparsely written external memory can offer a potential benefit to recurrent nets in understanding long sequences [43, 20, 99]. Dense video captioning involves two main problems: dense event detection and dense captioning. In this chapter, we focus on dense captioning.

memory-augmented recurrent neural networks have ideal properties for understanding long videos and densely captioning them. In particular, they enable the storage and access of memory cells that can capture the content of events as they evolve over varying timescales. Furthermore, cells in the external memory are written sparsely, which lets them store data reliably for long timescales, unlike the neurons in RNN models where whole neurons are updated at every iteration. This property provides memory-augmented networks a mechanism to store information about the long sequence of events which would enable a contextual understanding.

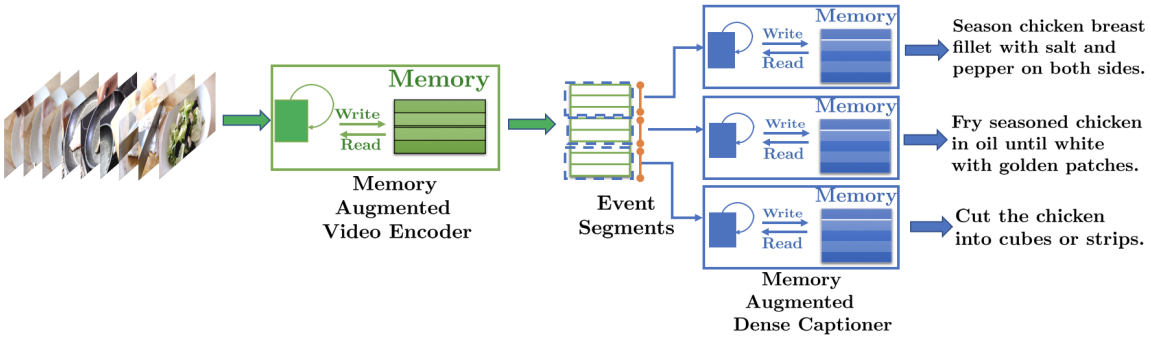



Figure 6.1: An overview of our memory-augmented recurrent neural network based model for dense video captioning. We use a memory-augmented recurrent representation to encode the whole video and also to caption each event segment.

As shown in Fig. 6.1, we use memory-augmented recurrent neural networks for our model components. First, a memory-augmented video encoder is used to produce a feature representation of event segments in a holistic manner based on context from other events occurring in the video. Second, each of the event segments is captioned coherently by a memory-augmented network using these holistically learnt event segment representations. We demonstrate the effectiveness of this over baseline methods that do not have memory augmentation.

## 6.3 Related Work

**Dense Video Captioning**. Krishna et al. [73] proposed the ActivityNet Captions dataset that aims to benchmark event detection algorithms which can also provide a natural language description of these detected events. Zhou et al. [191] introduce a new procedural video dataset called YouCookII which contains YouTube cooking videos annotated densely with event segments and a natural language description of each of these events. Wang et al. [91] use a bi-directional recurrent representation to encode video frames for event detection and also to extract event context vectors for dense captioning. Xu et al. [168] learn to perform joint detection of events and describe them using 3D convolutional representation to detect events and a hierarchical LSTM representation to densely caption the video about these events. Zhou et al. [192] address dense captioning by using transformer based end-to-end event segment detection and captioning model with multiple layers of multi-headed self attention. Li et al. [91] propose to jointly learn to detect and caption the events by

using “descriptiveness" regression to refine the segment boundaries and caption using an attribute augmented captioning architecture. Wang et al. [161] propose an adaptive bidirectional context fusion based on a gating mechanism for both the event proposal and event caption generation. Zhang et al. [185] utilize cross-modal hierarchical sequential embedding that learns multi-granular correspondances between image/video and text for performing different tasks including dense video captioning. Unlike many of the previous approaches for this task that rely on recurrent neural network representations to function both as a memory bank and video-caption representation learners, in our model, we provide dedicated stable external memories both for our video generator and captioner.

**Image and Video Captioning** is a computer vision task that has an extensive body of literature behind it. Some of the earlier examples of image and video captioning methods include Donahue et al. [32] that uses an encoder decoder technique to caption images and videos. You et al. [181] invoke semantic attribute attention maps on both the input and output caption representations to learn a captioning model. Karpathy et al. [65] propose to describe images by inferring a latent image to caption alignment by performing multimodal embedding and using a structured objective. Yu et al. [183] learn a hierarchical representation for paragraph captioning of video by incorporating temporal and spatial attention mechanisms.

**Memory models** are frequently used to learn sequence representations for performing various tasks like question answering in text [43], movie question answering [99] and image captioning [20]. Graves et al. [43] introduce a novel external memory-augmented recurrent neural network to perform multiple tasks that include question answering in synthetic dataset settings, and three graph processing based tasks. Na et al. [99] use a write CNN to encode multimodal data content and a read CNN to read this content as well as the question to learn the answer representation. Park et al. [20] learn a personalized image captioning representation by involving a memory which is used as a storage bank to capture contextual data representations that are pertinent to hashtag prediction and post generation which are primary tasks with in this chapter. Wang et al. [162] propose a multimodal memory for video captioning. Differing from these methods, we focus on the task of dense video captioning which involves generating a caption for each of the events in videos.

## 6.4 Method

The proposed model consists of a recurrent external memory aided video encoder and caption generator. We first provide a brief description of external memory-augmented neural networks and follow it up with a description of our dense captioning model aided by this network.

### 6.4.1 Preliminaries

Inspired by Graves et al. [43], our memory encoder consists of an external memory that enhances the storage capacity of recurrent neural networks and a memory controller that accesses this mem-

ory to store and retrieve history. We provide an overview of each of these components and their functionalities.

### Memory Controller

The memory controller consists of a recurrent neural network that uses the external memory to store information. Iteratively, it reads and writes to the external memory, and in this process, it encodes the temporal dynamics of the input. The controller encoded input representation comprises of the controller recurrent neural network output, and in addition, a set of “read vectors" being read from the external memory. Mathematically, the controller operation can be described as:

$$o_t = C(f_t; M^r_{t-1}); M^r_t \tag{6.1}$$

Here, $C$ denotes a controller recurrent network, $f_t$, $M^r_t$ are the feature input and external memory read vectors at time $t$. “;" is the concatenation operator. The read vectors are obtained from the controller's read to the external memory. The controller performs iterative read/write operations on the external memory, described later in this section. The controller network emits this output $o_t$. The final encoded representation is obtained by computing a residual connection over $o_t$ at time $t$.

$$\tilde{o}_t = o_t + g(f_t) \tag{6.2}$$

Here, function $g$ maps input $f_t$ to the output space. $\tilde{o}_t$ is the final output of the network.

### External Memory

Our external memory design and operations are similar to the method described in Graves et al. [43]. The external memory consists of a block of memory cells used by the controller to store memory. At each timestep, the controller reads a set of data from the memory and writes another set of data to the memory. The memory read/write locations are chosen by a probabilistic addressing mechanism, including content based addressing. In addition to the content based addressing, additional addressing components specific to read/write operations are also utilized to serve customized functionalities accordingly. First we describe the memory interface functions that the controller uses to interact with the external memory. Then, we describe memory read/write operations.

**Memory Interface functions:** Before delving into the details of read write operation, we first list the set of functions that controller generates to interact with memory. These functions are obtained by computing a non-linear transformation on the current memory output. We assume that there are N memory cells of size M and $R$ read heads. We have a single write head, following Graves et al. [43].

Here, each of the key vectors are of size $M$, so are the erase and the write vectors. Each of the strengths, allocation gate, write gate, free gates are scalars. The read mode is a 3 dimensional vector.

| Operation | Function | Notation |
|---|---|---|
| Read | Key | $\{\mathrm{k}_t^{r,i}\}_{i=1}^{R}$ |
| | Strength | $\{\tilde{s}_t^{r,i}\}_{i=1}^{R}$ |
| | Mode | $\{\pi_t^i\}_{i=1}^{R}$ |
| Write | Key | $\mathrm{k}_t^w$ |
| | Strength | $\tilde{s}_t^w$ |
| | Free gate | $\{{\delta_t}^i\}_{i=1}^{R}$ |
| | Erase vector | $\mathrm{e}_t$ |
| | Write vector | $\alpha_t$ |
| | Allocation gate | $\mathrm{g}_t^a$ |
| | Write gate | $\mathrm{g}_t^w$ |

**Content based addressing:** Both the read and the write operations involve content based addressing mechanism to find the most appropriate memory location. At each timestep $t$, content based addressing chooses the most appropriate location to perform the operation by computing a probability map over locations. It uses a key vector $k$ and computes cosine similarity between this key vector and content at memory locations:

$$c_t = softmax(cos(M_{t-1}, k_t)\tilde{s}_t) \tag{6.3}$$

Here, $c_t$ denotes content weight for memory locations in $M_t$, and $\tilde{s_t}$ denotes "strength" value constrained to a range between $[1, \infty)$, all at time $t$. At each timestep, content based addressing uses a read and write key/strength pair to perform read/write operation. We now describe the read/write operation.

**Write Operation:** The write operation involves computing the content (known as "write vector") and the location to write to the memory (known as "write weightings"), determined by a set of differentiable components. Intuitively, the write location is determined by parameters that choose between re-writing a location that has been written by **content based addressing** (Eqn. 6.3) and writing to a new location by a technique called **dynamic memory addressing**. In addition to this component, the write operation also enables an operation of preventing any write operation using a learnable "write" gate and to trigger a memory reset using an "erase" vector. We now describe the mathematical formulation of the write process, involving content based memory addressing and dynamic memory allocation.

- **Dynamic Memory Allocation:** Dynamic memory allocation component is a differentiable process that determines a time varying set of new memory locations to write. iteratively, the new locations to be picked for writing is determined by an "allocation weighting". It involves computing "usages" of memory locations and assigning high allocation weights to the most free locations. Mathematically usage of locations is represented by a usage vector $u_t$ and free locations are mathematically represented by free list $\eta$. The usage vector is also influenced by free gates (a set of interface functions) $\delta$ which determine whether recently read locations

can be freed, in form of a retention vector $\gamma$. We now discuss mathematical formulation of these components. We first build towards computing the usage vector, and then define the allocation weightings.

First, the retention vector is computed using free gates as:

$$\gamma_t = \prod_{i=1}^{R} (1 - \delta_t^i w_{t-1}^{r,i}) \tag{6.4}$$

Note that free gate is a scalar, changes with time.Typically, a location $i$'s content is retained if their retention vector value is close to 1 ($\gamma[i] \rightarrow 1$), and as seen from Equation 6.4, its value is controlled by the free gates $\delta$. Conversely, a location can be freed by the free gates by reseting its retention value to 0. Using this retention vector, $\gamma$, usage vector at time t can be computed as:

$$u_t = (u_{t-1} + w_{t-1}^w - u_{t-1} \circ w_{t-1}^w) \circ \gamma_t \tag{6.5}$$

Here, $\circ$ stands for elementwise multiplication. From Equation 6.5, it is seen that the usage value of a location increases with a write from previous timestep. Also, it can be seen that the usage of a location is reset to 0 if its retention value is 0. Finally, by sorting the usage vector, free list $\eta$ is obtained and allocation weighting $a_t$ is iteratively computed for each location in the increasing order of its usage using the free list as:

$$a_t[\eta[j]] = (1 - u_t[\eta_t[j]]) \prod_{i=1}^{j-1} u_t[\eta_t[i]] \tag{6.6}$$

- **Content based addressing:** Content based addressing, introduced earlier, also influences the write operation:

$$c_t^w = softmax(cos(M_{t-1}, k_t^w)\tilde{s}_t^w) \tag{6.7}$$

Here, $c_t^w$ stands for content weightings, $k_t^w$ stands for write key, and $\tilde{s}_t^w$ stands for write strength, $cos$ denotes cosine similarity. Intuitively, this component picks the write location with a content that is most similar to the write key.

- **Final Memory Write:** Write weightings are computed as a weighted combination of allocation weighting and content weightings (LHS terms in Equations 6.6 and 6.7 respectively). Mathematically, write weightings at time $t$ are computed as:

$$w_t^w = g_t^w[g_t^a a_t + (1 - g_t^a)c_t^w] \tag{6.8}$$

Here, $g_t^w$ and $g_t^a$ are write gate and allocation gate respectively, generated by the controller. If the write gate is zero, write operation is disabled. Similarly, the allocation gate decides relative weights of content based addressing and allocating a new location to write. Using the

write weightings, memory $M$ at time $t$ is updated as:

$$M_t = M_{t-1} \circ (J_{M,N} - w_t^w e_t^T) + w_t^w \alpha_t^T \tag{6.9}$$

where $J$ is a matrix of ones, $e$ stands for the erase vector, $\alpha$ stands for the write vector, $T$ denotes vector transpose operation and $\circ$ stands for elementwise multiplication.

**Read Operation:** Similar to the write operation, the read operation involves computing the location to read, known as "read weightings". A read operation is determined by multiple factors. Following Equation 6.3, the content based addressing weight component corresponding to read weights is computed for memory locations. In addition to the content based addressing component, a read operation also consists of a component that tracks the temporal order in which the contents are recorded in the memory. To do so, first, a **precedence vector** is computed which measures frequency of write operations at a given location. Second, using this precedence vector, a temporal **linkage matrix** is computed, which is used in encoding the write location order (known as temporal links) in the form of **forward** and **backward** link weights. Final read weightings are computed as a weighted combination of these three components. Next, we describe the mathematical formulation of the read operation.

- **Content based addressing:** Similar to write operation, content based addressing in read operation involves a read key, and a read strengths to compute read location weightings corresponding to this component:

$$\begin{aligned} c_t^{r,i} &= softmax(cos(M_{t-1}, k_t^{r,i})\tilde{s}_t^{r,i}) \\ i &= 1, 2, ...R \end{aligned} \tag{6.10}$$

  Here, we have distinct read keys $k_t^{r,i}$ corresponding to each of the $R$ read heads indexed with $i$, so are read strengths $\tilde{s}_t^{r,i}$. Intuitively, this component picks the read location with a content that is most similar to the read key.

- **Temporal Linkage:** This component involves iteratively updating a temporal linkage matrix that intuitively captures the order in which memory locations are written. Subsequently, the link matrix contributes to read weights, two components that captures order with one in the forward direction and the other in the backward direction. Computing linkage matrix involves computing "precedence weight" which quantifies write frequency of memory locations:

$$p_t = (1 - \sum_{i=1}^{N} w_t^w[i])p_{t-1} + w_t^w \tag{6.11}$$

  Here $p_t$ stands for precedence weight at time $t$, $w_t^w$ corresponds to write weightings defined in Equation 6.8, $N$ is the number of memory cells. Next, it is then used to compute the link matrix.

Entries of the link matrix are iteratively updated so as to encode the most recent link between a given pair of locations. Mathematically, the link matrix which is a square matrix of size $N$ at time $t$ is obtained as:

$$L_t[i,j] = (1 - w_t^w[i] - w_t^w[j])L_{t-1}[i,j] + w_t^w[i]p_{t-1}[j] \tag{6.12}$$

Note from this equation that link weights $L_t[i,j]$ are updated whenever a write occurs at either of those locations. Using link matrix, forward and backward link weights are computed as:

$$f_t^i = L_t^T w_{t-1}^{r,i} \tag{6.13}$$

$$b_t^i = L_t^T w_{t-1}^{r,i}, \tag{6.14}$$

$$i = 1, 2, ....R$$

Here, we see that both forward link weights $f_t^i$ and backward link weights $b_t^i$ are indexed, one for each of the $R$ read heads. Both these weights are computed iteratively.

- **Read Vectors:** Finally, read weightings are computed as a weighted combination of $c_t^{r,i}$ (obtained from Equation 6.10), $f_t^i$ (obtained from Equation 6.13), and $b_t^i$ (obtained from Equation 6.14):

$$w_t^{r,i} = \pi_1^i c_t^{r,i} + \pi_2^i f_t^i + \pi_3^i b_t^i, \tag{6.15}$$

$$i = 1, 2, ....R$$

Using these read weightings $w_t^{r,i}$, one for each of the $R$ read heads, the final read vectors corresponding to each read head at time $t$ is computed as:

$$m_t^{r,i} = M_t^T w_t^{r,i}, \tag{6.16}$$

$$i = 1, 2, ...R$$

Here, the memory vector $m_t^{r,i}$, is computed as a weighted combination of memory locations, where the read weighting corresponding to the read head is used as the weight, and $T$ denotes transpose operation. Using these operations, we obtain $M_t^r$ as the concatenation of $R$ read vectors as:

$$M_t^r = [m_t^{r,1}; m_t^{r,2}..m_t^{r,i}...m_t^{r,R}] \tag{6.17}$$

$$i = 1, 2, ...R$$

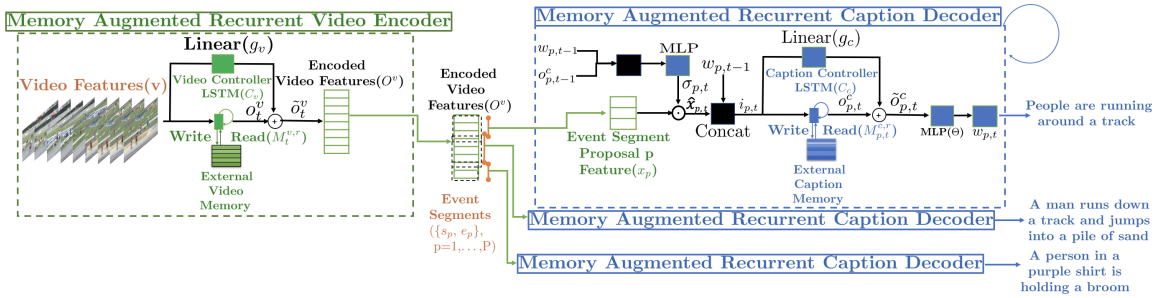



Figure 6.2: Illustration of our dense captioning model with references to variables in our equations. Encoder components are shown in green and decoder components are shown in blue.

### 6.4.2 Memory-Augmented Recurrent Video Encoder

In this problem, we are given an input video that contains multiple segments of interest that need to be captioned, for which we employ a memory-augmented representation to encode the video in a holistic manner and to caption all the events that occur in the video. A detailed illustration of our model is shown in Fig. 6.2. We now describe each of these components.

We use a memory-augmented recurrent neural network to encode the multi-event video in a holistic manner. The recurrent neural network in our video controller, $C_v$ of our video encoder is a single layer bidirectional LSTM (Bi-LSTM). Each frame in the input $v_t$ is encoded using this representation as:

$$o_t^v = (C_v(v_t; M_{t-1}^{r,v}); M_t^{r,v}) \tag{6.18}$$

where $M_t^{r,v}$ are a set of read vectors from the timestep $t$ (obtained by the computations mentioned in the section 6.4.1), and $o_t^v$ is the output of video controller network. We further compute a transformation over residual connection [50] on controller to get the final encoded video representation:

$$\tilde{o}_t^v = o_t^v + g_v(v_t) \tag{6.19}$$

Here, $g_v$ is a function that maps video input to the video encoder network's output space. We refer to the entire encoded video representation as $O^v$, which has features corresponding to $T$ video frames stacked together:

$$O^v = (\tilde{o}_1^v, \tilde{o}_2^v, \ldots, \tilde{o}_T^v) \tag{6.20}$$

### 6.4.3 Memory-Augmented Recurrent Dense Caption Generator

We use the encoded video representation $O^v$ to caption the events that occur in the video. The input to this module is the encoded video representation $O^v$ and a set of $P$ event segments. An event segment $p$ is defined as a tuple $(s_p, e_p)$ of its start and end locations. Using these inputs, we first extract event segment features as:

$$x_p = (O_{s_p}^v, O_{s_p+1}^v, \ldots, O_{e_p}^v) \tag{6.21}$$

We caption these segments $x_p$ using our dense captioning model.

We enable the event captioner to focus on different parts of the event segment appropriately while generating the caption word-by-word, using temporal attention. At each word generation step, event segment features are temporally attended to compute the video feature input to the caption decoder at time $t$, denoted by $\hat{x}_{p,t}$:

$$\hat{x}_{p,t} = \sigma_{p,t} \cdot x_p \tag{6.22}$$

Here, $\sigma_{p,t}$ is the temporal attention weight vector for the $p^{th}$ event segment at time $t$, and "$\cdot$" denotes inner product. We use this attended event segment representation to the caption network, which we use to compute the attention weights $\sigma_{p,t}$ described later.

We use a representation consisting of a memory-augmented recurrent neural network for the caption controller network. It consists of a single layer Bi-LSTM, which is used to decode the caption word by word. We use this network in decoding the next word of the caption as follows:

$$i_{p,t} = [y_{p,t-1}; \hat{x}_{p,t}] \tag{6.23}$$

$$o^c_{p,t} = [(C_c(i_{p,t}; M^{r,c}_{p,t-1})); M^{r,c}_{p,t}] \tag{6.24}$$

$$\tilde{o}^c_{p,t} = \theta(o^c_{p,t} + g_c(i_{p,t})) \tag{6.25}$$

$$y_{p,t} = softmax(\tilde{o}^c_{p,t}) \tag{6.26}$$

In the above equations, $p$ and $t$ correspond to event segment and time indices respectively, $i_{p,t}$ is the input to the caption controller network. $C_c$ is the Bi-LSTM neural network part of the caption controller, and $M^{r,c}_{p,t}$ are a set of read vectors at time $t$ (obtained by the computations mentioned in the section 6.4.1). The function $g_c$ maps input $i_{p,t}$ to the caption controller network's output space and is used to compute the output word representation. Finally, $theta$ is a non-linear transformation that projects caption controller output to the output space, denoted by $\tilde{o}^c_{p,t}$. We use this output to compute the probability distribution over the vocabulary for the next word $y_{p,t}$ by a softmax operation. Note from Equation 6.25 that similar to the video encoder, the captioning network too computes a residual connection across the caption controller. Using caption controller network outputs from the last timestep, the video attention weight $\sigma_{p,t}$ used in Equation 6.22 is recursively computed using the controller hidden state as:

$$\sigma_{p,t} = softmax(\phi([o^c_{p,t-1}; y_{p,t-1}])) \tag{6.27}$$

Here, $o^c_{p,t}$ is the caption controller network output defined in Equation 6.24, and $y_{p,t-1}$ is the previous caption word distribution generated using Equation 6.26. $\phi$ is a linear transformation map that is used to compute the final video attention weight $\sigma_{p,t}$.

| Method | BLEU 1 | BLEU 2 | BLEU 3 | BLEU 4 | METEOR |
|---|---|---|---|---|---|
| LSTM-YT [156] | 18.40 | 8.76 | 3.99 | 1.53 | 8.66 |
| HRNN [183] | 18.41 | 8.80 | 4.08 | 1.59 | 8.81 |
| Krishna et al. [73] | 18.13 | 8.43 | 4.09 | 1.60 | 8.88 |
| Li et al. [91] | 19.57 | 9.90 | 4.55 | 1.62 | 10.33 |
| Zhang et al. [185] | 19.8 | 9.4 | 4.3 | 2.1 | 9.2 |
| LSTM-vid/LSTM-cap | 18.91 | 7.75 | 3.09 | 1.55 | 8.95 |
| Ours-Mem-Vid/LSTM-cap | **21.75** | **10.06** | 4.30 | 1.92 | 9.76 |
| Ours-Mem-Vid/Mem-cap | 21.67 | 9.87 | 4.15 | 1.90 | 9.84 |
| Zhou et al. [192][1] | - | - | **5.80** | **2.77** | **11.2** |

Table 6.1: Experimental results on ActivityNet Captions dataset for all our methods using ground truth event segments (Numbers for the first four rows obtained from Li et al. [91]).

The output of dense caption generator is then a set of captions corresponding to each event segment:

$$((y_1^1, y_2^1, \ldots, y_{L_c^1}^1), (y_1^2, y_2^2, \ldots, y_{L_c^2}^2)), \ldots,$$
$$(y_1^p, y_2^p, \ldots, y_{L_c^p}^p), \ldots, (y_1^P, y_2^P, \ldots, y_{L_c^P}^P))$$

Here, index $p$ corresponds to event segment index, and $L_c^1, L_c^2, \ldots, L_c^P$ represent the length of $P$ captions, corresponding to each event segment respectively. In summary, we propose a novel dense video captioning model consisting of a memory-augmented video encoder and memory-augmented dense caption generator. We generate captions, one corresponding to each event in the video, independently of each other, by performing temporal attention over encoded event segment features.

## 6.5 Training

To learn our model, we are provided with a training set with ground truth event segments and a caption associated with each event segment. Mathematically, each instance in the training set is represented by:

$$t_n = \{(s^i, e^i, g^i), i = 1, 2, \ldots, P_n\} \tag{6.28}$$

Here, a training datapoint $t_n$, indexed by $n$, has an annotated set of $P_n$ event segments, and each event segment annotation consists of start time $s^i$, end time $e^i$ and a caption $g^i$.

We train the model end-to-end, and use cross entropy loss over words across all the $P_n$ captions as our loss function for this training example:

$$Loss = \sum_{i=1}^{P_n} \sum_{t=1}^{L_c^i} CE(y_t^i, g_t^i) \tag{6.29}$$

[1] Uses different features

Here, $y_t^i$ denotes the network predicted word distribution at time $t$, $g_t^i$ denotes the one hot representation of the ground truth word of $i^{th}$ caption at time $t$, and $L_c^i$ denotes the length of the $i^{th}$ caption. For a video, we compute this loss as a summation over all captions. We perform teacher forcing over the entire duration of training, where we input the ground truth caption word at each time $t$ instead of the word predicted by the model. During test time, we input the previously predicted word instead of the ground truth.

## 6.6 Experiments

### 6.6.1 Datasets

We conduct experiments on two datasets: the YouCookII [191] and the ActivityNet Captions [73] datasets. The YouCook II dataset has 2000 videos, with 1333 videos for training and 457 videos for validation. The videos in this dataset have an average event count of 7.70. The ActivityNet Captions dataset has 10k training videos and 4917 validation videos. The videos have an average event count of 3.65 in this dataset. In both the datasets, the length of the event caption does not have correlation to the event duration. We use ResNet-34 features provided in case of YouCookII dataset, and C3D features in case of ActivityNet Captions dataset. We perform experiments using the validation set as test data.

### 6.6.2 Model Settings

We use two external memory-augmented recurrent neural networks, one for video encoding, one for captioning, with the same parameter dimensions for both the networks. We use an external memory of 5 memory cells with size 1024. We have 4 read heads that result in 4 read vectors at each timestep, and 1 write head. For the controller, we use a Bi-LSTM for the video, and an LSTM cell for the caption controller, each with size 1024. We vary the learning rate between 0.1 and 0.01 with step size of 2 upon attaining training error plateau. We train our system for a fixed training time of 50 epochs. We report maximum BLEU and METEOR scores that we obtain for each of our baseline methods and the model.

We report results obtained using the ground truth event segments. We restrict ourselves to ground truth event segments when comparing with previous methods as our models focus on the dense captioning task and do not perform end-to-end training with an event detector as in some previous work [161, 192].

### 6.6.3 Baselines

**LSTM Captioner/video encoder:** We use Bi-LSTM for the video encoder and LSTM for the captioner, both without the external memory. We refer to this baseline as LSTM-vid/LSTM-cap in the results section. This is the baseline method for our model variants.

**LSTM Captioner, memory-augmented LSTM video encoder:** We use Bi-LSTM with the exter-

nal memory for the video encoder and LSTM with no external memory for the captioner. We refer to this model variant as Mem-Vid/LSTM-cap.

**Memory-Augmented Captioner, Memory-Augmented LSTM video encoder:** We use Bi-LSTM with the external memory for both video encoder and captioner. We refer to this model variant as Mem-Vid/Mem-cap.

### 6.6.4 Results

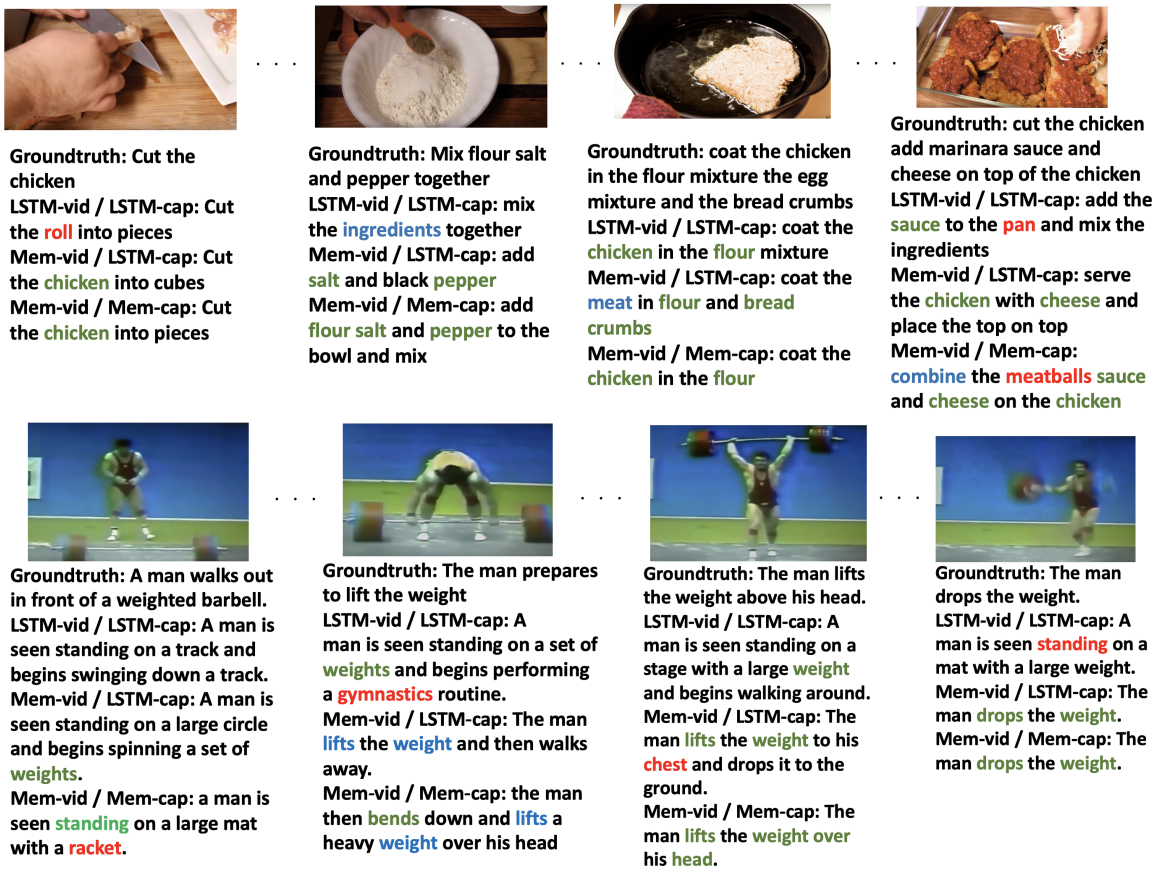



Figure 6.3: Sample qualitative results comparing ground truth captions with baseline and model variants. Note that the full model is able to generate the relevant content, shown in green (for exact attributes) and blue (for related attributes), while making fewer mistakes, shown in red.

| Method | BLEU 4 | METEOR |
|---|---|---|
| LSTM-vid/LSTM-cap | 0.93 | 9.15 |
| Ours-Mem-Vid/LSTM-cap | 1.49 | 9.74 |
| Ours-Mem-Vid/Mem-cap | **1.64** | 10.08 |
| Zhou et al. [192][1] | 1.42 | **11.2** |

Table 6.2: Experimental results on YouCookII dataset obtained using ground truth event segments.

Tab. 6.2 lists the performance of several methods including our model on the YouCook II dataset using ground truth events. We obtain state-of-the-art performance on the BLEU 4 metric and also achieve better performance compared to the baseline methods. Tab. 6.1 compares our method with previous dense video captioning methods applied to this problem and other dense video captioning/video captioning methods. We show that our model achieves competitive performance compared to these previous methods.

Relative to previous methods, our model achieves much more competitive performance in the case of the YouCookII dataset. The plausible reason for this observation lies in the nature of these two datsets. While YouCookII has events that are sequentially highly correlated events, ActivityNet videos have more independent events. The former scenario is more favourable to our model, as it aims to capture these long term correlations. Nevertheless, our models achieve competitive performance on both the datasets.

Fig. 6.3 shows qualitative results and compares the results of different baselines. It can be seen that the memory-augmented models refer to the relevant attributes of the event more often than the LSTM only baseline. This shows that memory augmentation improves the performance of recurrent models for dense captioning, and therefore could be extended to previous dense captioning methods involving recurrent neural network representations [91, 161].

## 6.7 Conclusion

In this chapter, we proposed a new model for dense video captioning involving an external memory-augmented video encoder and an external memory-augmented dense captioner. We showed that our model considerably improves the performance of recurrent neural network based dense captioning method, and is competitive with respect to previous state-of-the-art dense captioning methods on two datasets.

# Chapter 7

# Conclusion and Future Work

In this dissertation, we presented different methods and applications related to multi-person video scene understanding. We presented novel approaches to a specific multi-person scene understanding problem known as group activity recognition. We made critical contributions to group activity recognition. We introduced new deep learning-based solutions to model hierarchical temporal dynamics in group activity and a novel neural message-passing model that captures interactions between multiple atomic actions and group activity. We showed that our hierarchical deep temporal model [57] effectively captures the temporal dynamics of the individual actions and group activity, and our deep structured model effectively captures interactions between individual action and group activity representations [30]. We also presented finepruning, a principled method for performing network pruning jointly alongside finetuning for specialized domain tasks. Finally, we studied the problem of sequence understanding in videos in more detail, an essential component for group activity recognition, for which we proposed a new deep learning-based solution to dense video captioning.

In Chapter 3, we introduced a novel hierarchical deep temporal model for group activity recognition. Our method maps multi-person video clips to group activities. The hierarchical model involved two levels of hierarchies. Each hierarchy was trained separately in a two-stage framework, with the lower level in the hierarchy dedicated to individual action representation and the higher level to modelling group activities. The lower level hierarchy is trained to predict the right individual action category from short video clips. The higher-level hierarchy used an aggregated feature representation of all the individuals in the scene and was trained to predict the group activity label. We conducted experiments on multiple datasets and showed that our model obtained competitive performance and outperforms its baselines.

In Chapter 4 we presented a new deep structured model for group activity recognition. We presented a novel deep message passing neural network that performs refinement of predictions of all the labels present in the scene, such as group activity label and individual pose/action labels. Our model consisted of multiple blocks of neural network layers that mimic message passing through a simple forward computation to update label predictions. These layers are comprised of unary terms corresponding to each label prediction and pairwise terms that capture multiple interactions.

The first pairwise component captured interactions between individual pose information and group activity, whereas the second pairwise component captured interaction among all individual pose labels. We conducted experiments on multiple datasets and showed that our model obtained competitive recognition performance, and stacking our neural message passing layers further improves the performance.

In Chapter 5, we focused on efficient recognition in specialized domain tasks like group activity recognition. We presented a new principled method for pruning large neural networks for specialized domain tasks. This method, known as “finepruning", is an iterative method in which the network was jointly finetuned and pruned. Further, unlike previous pruning methods that preset pruning hyperparameters during the whole pruning, we adaptively set the pruning hyperparameters during each iteration by adding an “adapt pruning" module. In each iteration, we finetuned the network, efficiently searched for pruning hyperparameters using Bayesian optimization as our adapt pruning module, and pruned the network using these hyperparameter values. We showed that our method achieves the state-of-the-art pruning ratios on two different specialized domain datasets.

Finally, in Chapter 6, we turned our attention to the problem of long sequence understanding, which forms one of the essential components in group activity recognition. Specifically, we focused on the problem of dense video captioning, which involves captioning each event in a multi-event video. We presented a novel approach to dense video captioning. We proposed a memory-augmented recurrent neural network for dense video captioning, with a memory-augmented encoder that encodes video frames and a memory-augmented caption decoder that decodes caption word by word for every event. Using our method, we encoded the whole multi-event video using a memory-augmented encoder LSTM. Then, for each event annotation provided in the ground-truth, we used the provided ground-truth event segments to obtain event-level encoded features and decoded caption word by word by adaptively attending over different parts of the event segment for generating each word. We showed that our proposed method outperforms baseline methods with no memory augmentation on two different dense video captioning datasets.

Our models for group activity recognition have inspired many recent methods for group activity recognition. They sought to build better models for capturing interactions and relations between individuals, building better spatio-temporal feature representations, capturing important relevant relations, leveraging advances in other related fields such as human pose estimation for group activities for group activity recognition. In future work, we could expect further advances in this direction, where more methods would focus on improving the interaction modelling and feature representational learning.

Second, group activity recognition datasets which consist of trimmed video clips could be extended to untrimmed videos, which would contain multiple group activities and models built for recognizing in such videos shall detect these group activities in the video. Further, group activity task could be extended to include a hierarchy of group activities for recognizing a hierarchical set of group activities. Furthermore, current group activity models could be extended to predict natural language dense caption descriptions. Finally, network pruning methods shall be extended to work

with videos where models could skip processing a subset of frames in order to reduce computational costs for an efficient group activity recognition in videos.

# Bibliography


[1] Jake K Aggarwal and Lu Xia. Human activity recognition from 3d data: A review. *Pattern Recognition Letters*, 2014.

[2] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2016.

[3] Mohamed R. Amer, Dan Xie, Mingtian Zhao, Sinisa Todorovic, and Song-Chun Zhu. Cost-sensitive top-down / bottom-up inference for multiscale activity recognition. In *Proceedings of the European Conference on Computer Vision*, 2012.

[4] Mohamed Rabie Amer, Peng Lei, and Sinisa Todorovic. Hirf: Hierarchical random field for collective activity recognition in videos. In *Proceedings of the European Conference on Computer Vision*, 2014.

[5] Lisa Anne Hendricks, Subhashini Venugopalan, Marcus Rohrbach, Raymond Mooney, Kate Saenko, and Trevor Darrell. Deep compositional captioning: Describing novel object categories without paired training data. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2016.

[6] Borislav Antic and Björn Ommer. Learning latent constituents for recognition of group activities in video. In *Proceedings of the European Conference on Computer Vision*, 2014.

[7] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2016.

[8] Indriyati Atmosukarto, Bernard Ghanem, Shaunak Ahuja, Karthik Muthuswamy, and Narendra Ahuja. Automatic recognition of offensive team formation in american football plays. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2013.

[9] Timur Bagautdinov, Alexandre Alahi, François Fleuret, Pascal Fua, and Silvio Savarese. Social scene understanding: End-to-end multi-person action localization and collective activity recognition. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2017.

[10] Alina Bialkowski, Patrick Lucey, Peter Carr, Simon Denman, Iain Matthews, and Sridha Sridharan. Recognising team activities from noisy data. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2013.

[11] Yihang Bo and Hao Jiang. Scale and rotation invariant approach to tracking human body part regions in videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2013.

[12] William Brendel and Sinisa Todorovic. Learning spatiotemporal graphs of human activities. In *Proceedings of the IEEE International Conference on Computer Vision*, 2011.

[13] Ming-Ching Chang, Nils Krahnstoever, and Weina Ge. Probabilistic group-level motion analysis and scenario recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, 2011.

[14] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2017.

[15] Xinlei Chen and C Lawrence Zitnick. Learning a recurrent visual representation for image caption generation. *arXiv preprint arXiv:1411.5654*, 2014.

[16] Yu Cheng, Felix X Yu, Rogerio S Feris, Sanjiv Kumar, Alok Choudhary, and Shi-Fu Chang. An exploration of parameter redundancy in deep networks with circulant projections. In *Proceedings of the IEEE International Conference on Computer Vision*, 2015.

[17] Wongun Choi and Silvio Savarese. A unified framework for multi-target tracking and collective activity recognition. In *Proceedings of the European Conference on Computer Vision*, 2012.

[18] Wongun Choi, Khuram Shahid, and Silvio Savarese. What are they doing?: Collective activity classification using spatio-temporal relationship among people. In *IEEE International Conference on Computer Vision (ICCV) Workshops*, 2009.

[19] Wongun Choi, Khuram Shahid, and Silvio Savarese. Learning context for collective activity recognition. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2011.

[20] Cesc Chunseong Park, Byeongchang Kim, and Gunhee Kim. Attend to you: Personalized image captioning with context sequence memory networks. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2017.

[21] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[22] Davide Conigliaro, Paolo Rota, Francesco Setti, Chiara Bassetti, Nicola Conci, Nicu Sebe, and Marco Cristani. The s-hock dataset: Analyzing crowds at the stadium. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2015.

[23] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. In *Advances in Neural Information Processing Systems*, 2015.

[24] Bo Dai, Sanja Fidler, Raquel Urtasun, and Dahua Lin. Towards diverse and natural image descriptions via a conditional gan. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[25] Martin Danelljan, Gustav HÃd'ger, Fahad Shahbaz Khan, and Michael Felsberg. Accurate scale estimation for robust visual tracking. In *British Machine Vision Conference (BMVC)*, 2014.

[26] Pradipto Das, Chenliang Xu, Richard F Doell, and Jason J Corso. A thousand frames in just a few words: Lingual description of videos through latent topics and sparse object stitching. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2013.

[27] Jia Deng, Nan Ding, Yangqing Jia, Andrea Frome, Kevin Murphy, Samy Bengio, Yuan Li, Hartmut Neven, and Hartwig Adam. Large-scale object classification using label relation graphs. In *Proceedings of the European Conference on Computer Vision*, 2014.

[28] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer vision and Pattern recognition*, 2009.

[29] Zhiwei Deng, Arash Vahdat, Hexiang Hu, and Greg Mori. Structure inference machines: Recurrent neural networks for analyzing relations in group activity recognition. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2016.

[30] Zhiwei Deng, Mengyao Zhai, Lei Chen, Yuhao Liu, Srikanth Muralidharan, Mehrsan Javan Roshtkhari, and Greg Mori. Deep structured models for group activity recognition. In *British Machine Vision Conference*, 2015.

[31] Cem DirekoÇğlu and Noel E OâĂŹConnor. Team activity recognition in sports. In *Proceedings of the European Conference on Computer Vision*, 2012.

[32] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2015.

[33] Yong Du, Wei Wang, and Liang Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2015.

[34] Hao Fang, Saurabh Gupta, Forrest Iandola, Rupesh K Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C Platt, et al. From captions to visual concepts and back. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2015.

[35] Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. Every picture tells a story: Generating sentences from images. In *Proceedings of the European Conference on Computer Vision*, 2010.

[36] Alireza Fathi and Greg Mori. Action recognition by learning mid-level motion features. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[37] Rikke Gade and Thomas B. Moeslund. Sports type classification using signature heatmaps. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition (CVPR) Workshops*, 2013.

[38] Lianli Gao, Zhao Guo, Hanwang Zhang, Xing Xu, and Heng Tao Shen. Video captioning with attention-based lstm and semantic consistency. *IEEE Transactions on Multimedia*, 2017.

[39] J. R. Gardner, M. J. Kusner, Z. Xu, K. Q. Weinberger, and J. P. Cunningham. Bayesian optimization with inequality constraints. In *International Conference on Machine Learning*, 2014.

[40] Shaogang Gong and Tao Xiang. Recognition of group activities using dynamic probabilistic networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 2003.

[41] Yunchao Gong, Liwei Wang, Micah Hodosh, Julia Hockenmaier, and Svetlana Lazebnik. Improving image-sentence embeddings using large weakly annotated photo collections. In *Proceedings of the European Conference on Computer Vision*, 2014.

[42] Alex Graves and Navdeep Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *International Conference on Machine Learning*, 2014.

[43] Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 2016.

[44] Sergio Guadarrama, Niveda Krishnamoorthy, Girish Malkarnenkar, Subhashini Venugopalan, Raymond Mooney, Trevor Darrell, and Kate Saenko. Youtube2text: Recognizing and describing arbitrary activities using semantic hierarchies and zero-shot recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, 2013.

[45] Yiwen Guo, Anbang Yao, and Yurong Chen. Dynamic network surgery for efficient dnns. In *Advances in Neural Information Processing Systems*, 2016.

[46] Abhinav Gupta, Praveen Srinivasan, Jianbo Shi, and Larry S Davis. Understanding videos, constructing plots learning a visually grounded storyline model from annotated videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[47] Hossein Hajimirsadeghi, Wang Yan, Arash Vahdat, and Greg Mori. Visual recognition by counting instances: A multi-instance cardinality potential kernel. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015.

[48] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[49] Babak Hassibi and David G Stork. Second order derivatives for network pruning: Optimal brain surgeon. In *Advances in Neural Information Processing Systems*, 1993.

[50] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2016.

[51] Samitha Herath, Mehrtash Harandi, and Fatih Porikli. Going deeper into action recognition: A survey. *Image and vision computing*, 60, 2017.

[52] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015.

[53] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 1997.

[54] Micah Hodosh, Peter Young, and Julia Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, 2013.

[55] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. arXiv:1602.07360, 2016.

[56] M. Ibrahim and G. Mori. Hierarchical relational networks for group activity recognition and retrieval. In *Proceedings of the European Conference on Computer Vision*, 2018.

[57] Mostafa S. Ibrahim, Srikanth Muralidharan, Zhiwei Deng, Arash Vahdat, and Greg Mori. A hierarchical deep temporal model for group activity recognition. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2016.

[58] Stephen S Intille and Aaron F Bobick. Recognizing planned, multiperson action. *Computer Vision and Image Understanding*, 2001.

[59] Xu Jia, Efstratios Gavves, Basura Fernando, and Tinne Tuytelaars. Guiding the long-short term memory model for image caption generation. In *Proceedings of the IEEE International Conference on Computer Vision*, 2015.

[60] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the ACM International Conference on Multimedia*, 2014.

[61] Junqi Jin, Kun Fu, Runpeng Cui, Fei Sha, and Changshui Zhang. Aligning where to see and what to tell: image caption with region-based attention and scene factorization. *arXiv preprint arXiv:1506.06272*, 2015.

[62] Thorsten Joachims. Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006.

[63] E. Johns, S. Leutenegger, and A. J. Davison. Pairwise decomposition of image sequences for active multi-view recognition. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2016.

[64] Justin Johnson, Andrej Karpathy, and Li Fei-Fei. Densecap: Fully convolutional localization networks for dense captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[65] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[66] Andrej Karpathy, Armand Joulin, and Li F Fei-Fei. Deep fragment embeddings for bidirectional image sentence mapping. In *Advances in Neural Information Processing Systems*, 2014.

[67] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2014.

[68] A. Kendall, M. Grimes, and R. Cipolla. PoseNet: A convolutional network for real-time 6-DOF camera relocalization. In *IEEE International Conference on Computer Vision*, 2015.

[69] Davis E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 2009.

[70] Ryan Kiros, Ruslan Salakhutdinov, and Rich Zemel. Multimodal neural language models. In *International conference on machine learning*, 2014.

[71] Alexander Klaser, Marcin Marszałek, and Cordelia Schmid. A spatio-temporal descriptor based on 3d-gradients. In *British Machine Vision Conference*, 2008.

[72] Atsuhiro Kojima, Takeshi Tamura, and Kunio Fukunaga. Natural language description of human activities from video images based on concept hierarchy of actions. *International Journal of Computer Vision*, 2002.

[73] Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. Dense-captioning events in videos. In *Proceedings of the IEEE Conference on Computer Vision*, 2017.

[74] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012.

[75] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, 2011.

[76] Girish Kulkarni, Visruth Premraj, Vicente Ordonez, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C Berg, and Tamara L Berg. Babytalk: Understanding and generating simple image descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013.

[77] Suha Kwak, Bohyung Han, and Joon Hee Han. Multi-agent event detection: Localization and role assignment. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2013.

[78] Tian Lan, Lei Chen, Zhiwei Deng, Guang-Tong Zhou, and Greg Mori. Learning action primitives for multi-level video event understanding. In *Proceedings of the European Conference on Computer Vision*, 2014.

[79] Tian Lan, Leonid Sigal, and Greg Mori. Social roles in hierarchical models for human activity recognition. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2012.

[80] Tian Lan, Yang Wang, Weilong Yang, and Greg Mori. Beyond actions: Discriminative models for contextual group activities. In *Advances in Neural Information Processing Systems*, 2010.

[81] Tian Lan, Yang Wang, Weilong Yang, Stephen N Robinovitch, and Greg Mori. Discriminative latent models for recognizing contextual group activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011.

[82] Ivan Laptev. *Local spatio-temporal image features for motion interpretation*. PhD thesis, Numerisk analys och datalogi, 2004.

[83] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2006.

[84] Vadim Lebedev and Victor Lempitsky. Fast convnets using group-wise brain damage. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[85] Yann LeCun, John S Denker, and Sara A Solla. Optimal brain damage. In *Advances in Neural Information Processing Systems*, 1990.

[86] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*, 2016.

[87] Li-Jia Li, Hao Su, Li Fei-Fei, and Eric P Xing. Object bank: A high-level image representation for scene classification & semantic feature sparsification. In *Neural Information Processing Systems*, 2010.

[88] Ruonan Li, Rama Chellappa, and Shaohua Kevin Zhou. Learning multi-modal densities on discriminative temporal interaction manifold for group activity recognition. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2009.

[89] Siming Li, Girish Kulkarni, Tamara L Berg, Alexander C Berg, and Yejin Choi. Composing simple image descriptions using web-scale n-grams. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, 2011.

[90] Xin Li and Mooi Choo Chuah. Sbgar: Semantics based group activity recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[91] Yehao Li, Ting Yao, Yingwei Pan, Hongyang Chao, and Tao Mei. Jointly localizing and describing events for dense video captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[92] Ivan Lillo, Alvaro Soto, and Juan Carlos Niebles. Discriminative hierarchical modeling of spatio-temporally composable human activities. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2014.

[93] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multibox detector. In *Proceedings of the European Conference on Computer Vision*, 2016.

[94] Patrick Lucey, Alina Bialkowski, Peter Carr, Stuart Morgan, Iain Matthews, and Yaser Sheikh. Representing and discovering adversarial team behaviors using player roles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[95] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2016.

[96] Marcin Moczulski, Misha Denil, Jeremy Appleyard, and Nando de Freitas. Acdc: A structured efficient linear layer. *arXiv preprint arXiv:1511.05946*, 2015.

[97] Vlad I Morariu and Larry S Davis. Multi-agent event recognition in structured scenarios. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2011.

[98] Jonghwan Mun, Linjie Yang, Zhou Ren, Ning Xu, and Bohyung Han. Streamlined dense video captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[99] Seil Na, Sangho Lee, Jisung Kim, and Gunhee Kim. A read-write memory network for movie story understanding. In *Proceedings of the International Conference on Computer Vision*, 2017.

[100] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2015.

[101] Bingbing Ni, Shuicheng Yan, and Ashraf Kassim. Recognizing human group activities with localized causalities. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[102] Peter Nillius, Josephine Sullivan, and Stefan Carlsson. Multi-target tracking-linking identities using bayesian network inference. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2006.

[103] Sangmin Oh, Anthony Hoogs, Amitha Perera, Naresh Cuntoor, Chia-Chih Chen, Jong Taek Lee, Saurajit Mukherjee, JK Aggarwal, Hyungtae Lee, Larry Davis, et al. A large-scale benchmark dataset for event recognition in surveillance video. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2011.

[104] Vicente Ordonez, Girish Kulkarni, and Tamara L Berg. Im2text: Describing images using 1 million captioned photographs. In *Advances in Neural Information Processing Systems*, 2011.

[105] Paul Over, George Awad, Martial Michel, Jonathan Fiscus, Wessel Kraaij, Alan F. Smeaton, Georges Queenot, and Roeland Ordelman. Trecvid 2015 – an overview of the goals, tasks, data, evaluation mechanisms and metrics. In *Proceedings of TRECVID 2015*, 2015.

[106] Yingwei Pan, Tao Mei, Ting Yao, Houqiang Li, and Yong Rui. Jointly modeling embedding and translation to bridge video and language. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2016.

[107] Yingwei Pan, Ting Yao, Houqiang Li, and Tao Mei. Video captioning with transferred semantic attributes. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2017.

[108] Ronald Poppe. A survey on vision-based human action recognition. *Image and vision computing*, 2010.

[109] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas. Volumetric and multi-view CNNs for object classification on 3D data. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2016.

[110] Mengshi Qi, Jie Qin, Annan Li, Yunhong Wang, Jiebo Luo, and Luc Van Gool. stagnet: An attentive semantic rnn for group activity recognition. In *Proceedings of the European Conference on Computer Vision*, 2018.

[111] Vignesh Ramanathan, Jonathan Huang, Sami Abu-El-Haija, Alexander Gorban, Kevin Murphy, and Li Fei-Fei. Detecting events and key actors in multi-person videos. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2016.

[112] Vignesh Ramanathan, Bangpeng Yao, and Li Fei-Fei. Social role discovery in human events. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2013.

[113] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[114] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *Proceedings of the European Conference on Computer Vision*, 2016.

[115] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2016.

[116] Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. Grounding action descriptions in videos. *Transactions of the Association for Computational Linguistics*, 2013.

[117] Zhou Ren, Xiaoyu Wang, Ning Zhang, Xutao Lv, and Li-Jia Li. Deep reinforcement learning-based image captioning with embedding reward. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2017.

[118] Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[119] Anna Rohrbach, Marcus Rohrbach, Wei Qiu, Annemarie Friedrich, Manfred Pinkal, and Bernt Schiele. Coherent multi-sentence video description with variable level of detail. In *German Conference on Pattern Recognition*, 2014.

[120] Marcus Rohrbach, Wei Qiu, Ivan Titov, Stefan Thater, Manfred Pinkal, and Bernt Schiele. Translating video content to natural language descriptions. In *Proceedings of the IEEE International Conference on Computer Vision*, 2013.

[121] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

[122] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. arXiv:1409.0575, 2014.

[123] M. S. Ryoo and J. K. Aggarwal. Spatio-temporal relationship match: Video structure comparison for recognition of complex human activities. In *Proceedings of the IEEE International Conference on Computer Vision*, 2009.

[124] MS Ryoo and JK Aggarwal. Stochastic representation and recognition of high-level group activities. *International Journal of Computer Vision*, 2011.

[125] Christian Schuldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: a local svm approach. In *Proceedings of the International Conference on Pattern Recognition*, 2004.

[126] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transcations on Signal Processing*, 1997.

[127] Alexander G Schwing and Raquel Urtasun. Fully connected deep structured networks. *arXiv preprint arXiv:1503.02351*, 2015.

[128] Shikhar Sharma, Ryan Kiros, and Ruslan Salakhutdinov. Action recognition using visual attention. In *Neural Information Processing Systems (NIPS)*, 2015.

[129] Rakshith Shetty and Jorma Laaksonen. Video captioning with recurrent networks based on frame-and video-level features and visual content classification. *arXiv preprint arXiv:1512.02949*, 2015.

[130] Rakshith Shetty, Marcus Rohrbach, Lisa Anne Hendricks, Mario Fritz, and Bernt Schiele. Speaking the same language: Matching machine to human captions by adversarial training. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[131] Tianmin Shu, Sinisa Todorovic, and Song-Chun Zhu. Cern: confidence-energy recurrent network for group activity recognition. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2017.

[132] Tianmin Shu, Dan Xie, Brandon Rothrock, Sinisa Todorovic, and Song Chun Zhu. Joint inference of groups, events and human roles in aerial videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[133] Behjat Siddiquie, Yaser Yacoob, and Larry Davis. Recognizing plays in american football videos. Technical report, Technical report, University of Maryland, 2009.

[134] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[135] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems*, 2014.

[136] J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, 2012.

[137] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, 2012.

[138] Khurram Soomro, Salman Khokhar, and Mubarak Shah. Tracking when the camera looks away. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, 2015.

[139] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[140] Suraj Srinivas and R. Venkatesh Babu. Data-free parameter pruning for deep neural networks. In *British Machine Vision Conference*, 2015.

[141] Chen Sun, Chuang Gan, and Ram Nevatia. Automatic concept discovery from parallel text and visual corpora. In *Proceedings of the IEEE International Conference on Computer Vision*, 2015.

[142] Eran Swears and Anthony Hoogs. Learning and recognizing complex multi-agent activities with applications to american football plays. In *2012 IEEE Workshop on the Applications of Computer Vision (WACV)*, 2012.

[143] Eran Swears, Anthony Hoogs, Qiang Ji, and Kim Boyer. Complex activity recognition using granger constrained dbn (gcdbn) in sports and surveillance video. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2014.

[144] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2015.

[145] Hamed R Tavakoli, Rakshith Shetty, Ali Borji, and Jorma Laaksonen. Paying attention to descriptions generated by image captioning models. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[146] Jonathan J Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in Neural Information Processing Systems*, 2014.

[147] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 2015.

[148] Kenneth Tran, Xiaodong He, Lei Zhang, Jian Sun, Cornelia Carapcea, Chris Thrasher, Chris Buehler, and Chris Sienkiewicz. Rich image captioning in the wild. In *IEEE Conference on Computer vision and Pattern Recognition workshops*, 2016.

[149] Khai N Tran, Apurva Gala, Ioannis A Kakadiaris, and Shishir K Shah. Activity analysis in crowded environments using social cues for group discovery and human interaction modeling. *Pattern Recognition Letters*, 2014.

[150] Frederick Tung, Srikanth Muralidharan, and Greg Mori. Fine-pruning: Joint fine-tuning and compression of a convolutional network with bayesian optimization. In *British Machine Vision Conference*, 2017.

[151] Francesco Turchini, Lorenzo Seidenari, and Alberto Del Bimbo. Understanding sport activities from correspondences of clustered trajectories. In *IEEE International Conference on Computer Vision (ICCV) Workshops*, 2015.

[152] S. Vascon, E. Zemene, M. Cristani, H. Hung, M. Pelillo, and V. Murino. A game-theoretic probabilistic approach for detecting conversational groups. In *Asian Conference on Computer Vision*, 2014.

[153] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

[154] Subhashini Venugopalan, Lisa Anne Hendricks, Raymond Mooney, and Kate Saenko. Improving lstm-based video description with linguistic knowledge mined from text. *arXiv preprint arXiv:1604.01729*, 2016.

[155] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence-video to text. In *Proceedings of the IEEE International Conference on Computer Vision*, 2015.

[156] Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, and Kate Saenko. Translating videos to natural language using deep recurrent neural networks. In *North American Chapter of the Association for Computational Linguistics*, 2015.

[157] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems*, 2015.

[158] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2015.

[159] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Action recognition by dense trajectories. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2011.

[160] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Proceedings of the IEEE International Conference on Computer Vision*, 2013.

[161] Jingwen Wang, Wenhao Jiang, Lin Ma, Wei Liu, and Yong Xu. Bidirectional attentive fusion with context gating for dense video captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[162] Junbo Wang, Wei Wang, Yan Huang, Liang Wang, and Tieniu Tan. M3: multimodal memory modelling for video captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[163] Z. Wang, M. Zoghi, F. Hutter, D. Matheson, and N. de Freitas. Bayesian optimization in high dimensions via random embeddings. In *International Joint Conference on Artificial Intelligence*, 2013.

[164] Xinyu Wei, Long Sha, Patrick Lucey, Peter Carr, Sridha Sridharan, and Iain Matthews. Predicting ball ownership in basketball from a monocular view using only player trajectories. In *IEEE International Conference on Computer Vision (ICCV) Workshops*, 2015.

[165] Daniel Weinland, Remi Ronfard, and Edmond Boyer. A survey of vision-based methods for action representation, segmentation and recognition. *Computer vision and image understanding*, 2011.

[166] Jianchao Wu, Limin Wang, Li Wang, Jie Guo, and Gangshan Wu. Learning actor relation graphs for group activity recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[167] ZYYYY Wu and RSWW Cohen. Encode, review, and decode: Reviewer module for caption generation. *arXiv preprint arXiv:1605.07912*, 2016.

[168] Huijuan Xu, Boyang Li, Vasili Ramanishka, Leonid Sigal, and Kate Saenko. Joint event detection and description in continuous video streams. *arXiv preprint arXiv:1802.10250*, 2018.

[169] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2016.

[170] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, 2015.

[171] Ran Xu, Caiming Xiong, Wei Chen, and Jason J Corso. Jointly modeling deep video and compositional text to bridge vision and language in a unified framework. In *AAAI Conference on Artificial Intelligence*, 2015.

[172] Linjie Yang, Kevin Tang, Jianchao Yang, and Li-Jia Li. Dense captioning with joint inference and visual context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[173] Y. Yang and S. Newsam. Bag-of-visual-words and spatial extensions for land-use classification. In *ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 2010.

[174] Yang Yang, Jie Zhou, Jiangbo Ai, Yi Bin, Alan Hanjalic, Heng Tao Shen, and Yanli Ji. Video captioning by adversarial lstm. *IEEE Transactions on Image Processing*, 2018.

[175] Zichao Yang, Marcin Moczulski, Misha Denil, Nando de Freitas, Alex Smola, Le Song, and Ziyu Wang. Deep fried convnets. In *Proceedings of the IEEE International Conference on Computer Vision*, 2015.

[176] Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, and Aaron Courville. Describing videos by exploiting temporal structure. In *Proceedings of the IEEE International Conference on Computer Vision*, 2015.

[177] Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. Incorporating copying mechanism in image captioning for learning novel objects. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2017.

[178] Ting Yao, Yingwei Pan, Yehao Li, Zhaofan Qiu, and Tao Mei. Boosting image captioning with attributes. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[179] Serena Yeung, Olga Russakovsky, Ning Jin, Mykhaylo Andriluka, Greg Mori, and Li Fei-Fei. Every moment counts: Dense detailed labeling of actions in complex videos. *arXiv preprint arXiv:1507.05738*, 2015.

[180] Guojun Yin, Lu Sheng, Bin Liu, Nenghai Yu, Xiaogang Wang, and Jing Shao. Context and attribute grounded dense captioning. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*.

[181] Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. Image captioning with semantic attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[182] Chun-Nam John Yu and Thorsten Joachims. Learning structural svms with latent variables. In *Proceedings of the 26th annual international conference on machine learning*, 2009.

[183] Haonan Yu, Jiang Wang, Zhiheng Huang, Yi Yang, and Wei Xu. Video paragraph captioning using hierarchical recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[184] Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2006.

[185] Bowen Zhang, Hexiang Hu, and Fei Sha. Cross-modal and hierarchical modeling of video and text. In *Proceedings of the European Conference on Computer Vision*, 2018.

[186] Li Zhang, Flood Sung, Feng Liu, Tao Xiang, Shaogang Gong, Yongxin Yang, and Timothy M Hospedales. Actor-critic sequence training for image captioning. *arXiv preprint arXiv:1706.09601*, 2017.

[187] R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. In *European Conference on Computer Vision*, 2016.

[188] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip H. S. Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 2015.

[189] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *Advances in Neural Information Processing Systems*, 2014.

[190] H. Zhou, J. M. Alvarez, and F. Porikli. Less is more: towards compact CNNs. In *Proceedings of the European Conference on Computer Vision*, 2016.

[191] Luowei Zhou, Chenliang Xu, and Jason J Corso. Towards automatic learning of procedures from web instructional videos. *arXiv preprint arXiv: 1703.09788*, 2017.

[192] Luowei Zhou, Yingbo Zhou, Jason J Corso, Richard Socher, and Caiming Xiong. End-to-end dense video captioning with masked transformer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[193] Xiangxin Zhu and Deva Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, 2012.

[194] Yingying Zhu, Nandita M Nayak, and Amit K Roy-Chowdhury. Context-aware modeling and recognition of activities in video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[195] Yuke Zhu, Tian Lan, Yijian Yang, Steve Robinovitch, and Greg Mori. Latent spatio-temporal models for action localization and recognition in nursing home surveillance video. In *MVA*, 2013.