Deep Models for Multi-Person Activity Understanding

by

Mostafa S. Ibrahim

M.Sc., Cairo University, 2012

Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor in Philosophy in the School of Computing Science Faculty of Applied Sciences

© Mostafa S. Ibrahim 2018
SIMON FRASER UNIVERSITY
Fall 2018

Copyright in this work rests with the author. Please ensure that any reproduction or re-use is done in accordance with the relevant national copyright legislation.
### Approval

<table>
<thead>
<tr>
<th>Name</th>
<th>Mostafa S. Ibrahim</th>
</tr>
</thead>
<tbody>
<tr>
<td>Degree</td>
<td>Doctor in Philosophy (Computing Science)</td>
</tr>
<tr>
<td>Title</td>
<td>Deep Models for Multi-Person Activity Understanding</td>
</tr>
</tbody>
</table>
| Examining Committee: | Chair: Mark Drew  
|                  | Professor                  |
|                  | Greg Mori                  
|                  | Senior Supervisor           
|                  | Professor                   |
|                  | Oliver Schulte             
|                  | Supervisor                  
|                  | Professor                   |
|                  | Ping Tan                   
|                  | Internal Examiner           
|                  | Professor                   
|                  | School of Computing Science |
|                  | Karteek Alahari            
|                  | External Examiner           
|                  | Researcher                  
|                  | Inria Research Centre       |

| Date Defended    | November 20th, 2018        |
Abstract

Multi-person activity recognition is an important and challenging problem for the computer vision community with several applications such as visual surveillance and video summarization. For a long time, shallow architectures (e.g., SVM) were used with manually extracted features to answer the intended queries, but with unsatisfactory performance due to limitedness of feature engineering which may drop significant explanatory factors of data. An alternative is to automatically learn features at multiple levels of abstraction from raw visual data through Deep Convolutional Neural Networks (DCNN). In this thesis we make three contributions toward human activity understanding based on DCNN. 1) We propose hierarchical deep temporal models that automatically learn feature representation for individual person actions as well as the whole group activity while capturing temporal dynamics that exist at both levels. 2) We investigate approaches for action localization, a critical sub-problem in the multi-person activity recognition problem. 3) A graph-based network module for relational reasoning is introduced to capture hierarchical relationships among people in a video scene. Overall, the proposed models recognize the collective activity of individuals and their complex interactions by modeling different types of cues in a deep hierarchical temporal manner.

Keywords: Multi-person Activity Recognition; Hierarchical Models; LSTM; Sports Analysis; Deep Learning; Relational Layers
Dedication

To my amazing mother, Amal Aboelfadl,
    for her love, prayers and extreme efforts to educate me.

To my wonderful wife, Asmaa Morsy,
    for her continuous support to complete this research.

To my lovely child, Belal Ibrahim,
    for bringing happiness and joy to my life.

To my dearest friend, Ahmed Ibrahim,
    for keeping in touch all the time and encouragement.

To all my relatives, colleges, friends, students and trainees
    who wished me all the best and have confident in me.
Acknowledgements

Firstly, I would like to express my sincere gratitude to my advisor Prof. Greg Mori for his invaluable support and guidance throughout my studies, and especially for his confidence in me. Greg taught me how to identify an appropriate research problem, approach it from different angles and push the boundary of the proposed novelties while keeping the big picture of my line of research. Right now, I see myself as an independent researcher who is ready for a challenging research career. Greg, thank you for such a memorable experience. I cannot imagine a better adviser both professionally and personally.

I am also grateful to my committee members - Prof. Oliver Schulte and Prof. Ping Tan for their valuable comments during our discussions. The first time to learn about Deep Learning was in one of Oliver’s courses and my project in this course was the seed for my first published paper. In fact, the problem we tackled in this project is as same as my thesis dissertation. Oliver, thanks for pushing us to identify a good research project. It is also my honor to have Dr. Karteek Alahari as my external examiner. I am grateful to Mark Drew for serving as chair of my committee.

To my colleagues in the Vision and Media Lab, thank you for all the fruitful discussions and generous help. I’d like to give special thanks to Arash Vahdat, Srikant Muralidharan, Jiawei He, Mehran Khodabandeh, Megha Nawhal, Akash Abdu Jyothi, Nazanin Mehrasa, Mengyao Zhai and Zhiwei Deng. I was so lucky to work one more time under the supervision of Arash Vahdat during my internship at D-Wave Systems. I learned a lot from your research ideas and personal behavior. Arash, thanks for all the encouragement and support.

Last but not least, I would like to thank my mother, Amal Aboelfadl, for her love, prayers and extreme efforts to educate me. Words can not express how grateful I am to my lovely wife, Asmaa Morsy, for her continuous support to complete this research. Special thanks to my child, Belal, for bringing happiness and joy to my life. Thanks to my dearest friend, Ahmed Ibrahim, for keeping in touch all the time and encouragement.
Table of Contents

Approval ........................................................... ii
Abstract ........................................................... iii
Dedication .......................................................... iv
Acknowledgements ................................................ v
Table of Contents ................................................ vi
List of Tables ....................................................... viii
List of Figures ...................................................... x

1 Introduction ....................................................... 1
   1.1 Hierarchical Deep Temporal Models ......................... 3
   1.2 Tubelets for Action Localization ............................. 3
   1.3 Hierarchical Relational Networks ........................... 5
   1.4 Summary of Contributions .................................. 5

2 Previous Work .................................................... 7
   2.1 Applications ................................................ 8
   2.2 Related Problems .......................................... 8
   2.3 Handcrafted Era ........................................... 10
   2.4 Deep Learning Approaches ................................ 22
   2.5 Datasets ..................................................... 23

3 Hierarchical Deep Temporal Models for Group Activity Recognition ........................................... 28
   3.1 Overview .................................................... 28
   3.2 Related Work ............................................... 31
   3.3 Proposed Approach ......................................... 36
      3.3.1 Temporal Model of Individual Action .................. 36
      3.3.2 Hierarchical Model for Group Activity Recognition ... 38
      3.3.3 Handling sub-groups .................................. 39
List of Tables

Table 3.1 Comparison of our method with baseline methods on the Collective Activity Dataset. .......................................................... 43
Table 3.2 Comparison of our method with previously published works on the Collective Activity Dataset. ......................................................... 44
Table 3.3 Statistics of the group activity labels in the Volleyball Dataset. . . . . . . 45
Table 3.4 Statistics of the action labels in the Volleyball Dataset. ....................... 46
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. ............................... 46
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. ............... 47
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. ................................................................. 47
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 ................................................................. 48
Table 3.9 Comparison of the team activity recognition of our model against improved dense trajectory approach approach. Approaches with * are recent publications. ................................................................. 48
Table 4.1 Ablation study of our proposed framework on J-HMDB21 (split 1). Remaining classes in the next table. ....................................................... 62
Table 4.2 Remaining classes for the ablation study of our proposed framework on J-HMDB21 (split 1). ................................................................. 62
Table 4.3 Comparison to state-of-the-art methods on three datasets. Methods with * are considered as comparable basic models with the same VGG network setup. The results on the J-HMDB21 data set are averages over all three splits. The experiments on the UCF-101 data set are performed without temporal localization. Only [101] reports mAP both with and without temporal localization, therefore we only compare with [101] (without temporal localization) on this dataset. ........................................ 65
Table 4.4  Classification accuracy on J-HMDB21 dataset (averaged over all three splits). ................................................................. 65

Table 5.1  Volleyball Dataset: Left table is for versions of our model using single frame (last row shows state-of-the-art using a single frame). Right table is for 10-timesteps input clips performance of our best models versus state-of-the-art. .......................................................... 80

Table 5.2  Scene retrieval compared to baselines. ........................................ 83

Table 5.3  Scene retrieval compared to model variants. .............................. 83

Table 5.4  Scene Retrieval using Denoising Autoencoder (-D) with 50% possible drop for people in test data for models and baselines. Our model is robust; the No Autoencoder model performance drops significantly. 83

Table 5.5  Person Retrieval on Volleyball Dataset: Hit@K results of our method and baselines. Last column is mean average precision of query results. Our model outperforms the normal autoencoder model, and is competitive with a 32x larger sparse representation. ........................................ 84
List of Figures

Figure 2.1 From [53]. "A temporal structure description for the s51 play example: 4 agents (e.g., obj1) with their goals (e.g., goal obj1-act1) and the temporal constraints to be verified (e.g., before goal obj1-act1 goal obj1-act2)." 12

Figure 2.2 From [110]. The behavior agent Baysian netowrk. BEH (behaviour), DYN(dynamics), TRAJ(trajector), ACC(Accerlation), CURV(curvature). 13

Figure 2.3 From [58]. Attribute grammar for casing vehicles in a parking lot. 14

Figure 2.4 From [12]. "Truth values are assigned to event occurrences, which have the form E@I, for event symbol E and time interval I = [a, b], where a and b are positive integers such that a < b. Asserting that E@I is true means that an instance of E occurred precisely over interval I. In the figure, the noisy detections are combined with the PEL knowledge base (KB)." 16

Figure 2.5 From [15]. "The social interaction ontology in a nursing home." 17

Figure 2.6 From [18]. "We seek to recognize collective activities such as queuing (left picture) or talking (right picture). In isolation, the highlighted individuals have very similar appearance, and thus it is not possible to identify whether they are talking (red) or standing in a queue (blue). However, by considering the spatio-temporal distribution of others (i.e., the crowded context), it becomes easier to recognize that the two individuals are performing different activities and to identify which activities are being performed." 18

Figure 2.7 From [17]. "Spatio-Temporal Local Descriptor. (a) Space around anchor person (blue) is divided into multiple bins. The pose of the anchor person (blue arrow) locks the orientation of the descriptor which induces the location of the reference bin 1. (b) Example of STL descriptor - the descriptor is a histogram capturing people and pose distribution in space and time around the anchor person. (c) Classification of STL descriptor is achieved by decomposing the histogram in different levels along the temporal axis." 19
Figure 2.8 From [76]. 'Illustration of construction of our action context descriptor. (a) Spatio-temporal context region around focal person, as indicated by the green cylinder. In this example, we regard the fallen person as focal person, and the people are standing and walking as context. (b) Spatio-temporal context region around focal person is divided in space and time. The blue region represents the location of the focal person, while the pink regions represent locations of the nearby people. The first 3-bin histogram captures the action of the focal person, which we call the action descriptor. The latter three 3-bin histograms are the context descriptor, and capture the behavior of other people nearby. (c) The action context descriptor is formed by concatenating the action descriptor and the context descriptor.'

Figure 2.9 From [108]. 'When people interact in an event, they assume event-specific social roles. Social roles act as identities for the individuals and can help us describe the event in terms of these roles. Role recognition is fundamental in understanding a human event.'

Figure 2.10 From [25]. '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.'

Figure 2.11 Activities from VIRAT dataset.

Figure 2.12 Interactions from UT-Interaction dataset.

Figure 2.13 Collective activity dataset [17]. Four images for scene categories: crossing, queuing, waiting and walking.

Figure 2.14 Basketball dataset [106]: 'The figure is from their deep learning model over the dataset. The position of the ball in each frame is shown in yellow.'

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.

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.

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

Figure 4.1 Action localization with our Tube Proposal Network produces generic class-independent tubelets. The tubelets are classified with a Temporal Understanding Network that can perform detailed spatio-temporal analysis.

Figure 4.2 The proposed framework consists of two components: a tubelet proposal network (TPN) and a temporal understanding network (TUN). A TUN based on fused LSTMs is used to classify the generic class-independent tubelet proposals generated by the TPN.

Figure 4.3 Examples of hard cases for class-dependent action proposals. The first image action label is walking, while the second one is running. The action region in the third image has significant motion blur. The three frames come from the UCF-Sports dataset.
Figure 4.4 Examples from the J-HMDB dataset. Each video is represented by four frames. The red bounding boxes are the detected regions, and the blue ones are the ground-truth regions. The first three rows show successful cases (action classification is correct and averaged localization overlap is larger than 0.5). The bottom left one is a failed case where the averaged overlap ratio is smaller than 0.5. The bottom right one is a failed case in which the ground-truth label is Clap, but is classified as Sit.

Figure 4.5 Example action tubes from the UCF-Sports dataset. Each video is represented by four frames. The red bounding boxes are the detected regions, and the blue ones are the ground-truth regions. As can be seen, our generated tubes are very accurate. Also, our framework can handle multiple-person scenarios (upper left video), fast moving actions like Swing, and actions with large intra-class shape variation like Lifting. Left bottom shows a failure case. The ground-truth of this video only focused on the middle person with label Kicking. Our algorithm generates two high-scoring tubes for this video, one overlaps with the ground-truth region, another is the left person with label Walk, which we think is also reasonable.

Figure 4.6 Example action tubes from the UCF-101 dataset. Each video is represented by four frames. The red bounding boxes are the detected regions, and the blue ones are the ground-truth regions. The first two rows show successful cases, while the last row shows failure cases. Our framework generates accurate action tubes for complicated actions. However, the bottom-left case shows that our framework tends to generate only one action tube if multiple action instances are close to each other. The bottom-right and top-left cases show performance in complicated scenes such as basketball games (top-left correct, bottom-right inaccurate localization).
A single relational layer. The layer can process an arbitrary sized set of people from a scene, and produces new representations for these people that capture their relationships. The input to the layer is a set of $K$ people and a graph $G^\ell$ encoding their relations. In the relational layer, a shared neural network ($F^\ell$) maps each pair of person representations to a new representation that also encodes relationships between them. These are aggregated over all edges emanating from a person node via summation. This process results in a new, relational representation for each of the $K$ people. By stacking multiple relational layers, each with its own relationship graph $G^\ell$, we can encode hierarchical relationships for each person and learn a scene representation suitable for group activity recognition or retrieval.

Relational unit for processing one person inside a relational layer. The feature representation for a person (red) is combined with each of its neighbours'. Resultant vectors are summed to create a new feature representation for the person (dark red).

Our relational network for group activity recognition for a single video frame. Given $K$ people and their initial feature vectors, these vectors are fed to 3 stacked relational layers (of output sizes per person: 512, 256, 128). Each relational layer is associated with a graph $G^\ell$ (disjoint cliques in this example: layer 1 has 4 cliques, each of size 3; layer 3 is a complete graph). The shared MLP $F^\ell$ of each layer computes the representation of 2 neighbouring players. Pooling of the output $K$ feature vectors is used for group activity classification.

Our relational autoencoder model. The relationship graph for this volleyball scene is 2 disjoint cliques, one for each team and fixed for all layers. $K$ input person feature vectors, each of length 4096, are fed to a 4-layer relational autoencoder (sizes 256-128-256-4096) to learn a compact representation of size 128 per person.

Visualizations of scene retrieval using our relational autoencoder. Each 2 rows are a query: Query image first (blue box), followed by the closest 5 retrievals. Green Framed boxes are correct matches. The last query is for *Right team winpoint event*, and its results are 3 consecutive *Right team winpoint events* followed by 2 *Left team winpoint events*.
Chapter 1

Introduction

Multi-person activity recognition problem, where an activity is performed by a group of people in a video sequence, finds a lot of applications in the context of video surveillance, sports analytics, video search and retrieval. Classifying the group activity can be addressed by examining individual person actions and their relations. By "action" we mean a simple motion pattern by a single person (e.g., a walking person) while "activity" means a sequence of actions performed by several individuals evolving under spatial and temporal constraints (e.g., blocking activity by three volleyball players). It is a challenging problem due to factors such as variability within action classes, similarity between different action classes, and complex interactions between people, to name a few.

One of the key success factors of machine learning algorithms is data representation that captures important aspects or concepts relative to a specific problem. Based on that, several questions may come to mind when tackling the multi-person activity recognition problem such as the following: How could we build an appropriate feature representation of person actions in a video sequence? Or infer representations of his relationships with the surrounding people? Or in a more complex setup, how could we represent a whole dynamic scene of people and identify their collective activity? Given the spatial and temporal nature of this problem and the need to translate low-level pixels information to high-level semantic information about a dynamic video scene, it is a very challenging task.

For a long while, researchers used to build hand-crafted representation for the visual objects in an image and build models based on them. By "hand-crafted representation" we mean features designed by humans based on heuristics (e.g., using SIFT features [85] for image representation). These manual features are typically limited as significant explanatory factors of data might be dropped. Following are some examples for models used with this feature representation style. Several types of graphical models [53, 110, 39] were used to handle multi-persons interactions in a probabilistic manner. Domain knowledge may help in handling problems arising from uncertainty in low-level components or from complexity of the task itself. Based on that, several paths were tried such as grammar-based methods [58, 103], logical rules [87, 137, 12] or ontology-based approaches [15, 44]. These knowledge-
based approaches typically involved manual efforts (e.g., need for experts) that eventually limits models capabilities. Context-based approaches [17, 108, 73] help in resolving scene confusions using context cues (a jumping person in volleyball might perform a fake jump or a spike of the ball, but actions of surrounding players resolve this confusion).

Recently, representation learning using Deep Convolutional Neural Networks has shown impressive performance on a variety of computer vision tasks including image classification [70, 127] and action recognition [128, 154]. By stacking multiple network layers for features extraction and transformation, DCNN could learn a hierarchy of representations that correspond to different levels of abstraction. In this thesis, the proposed models are based on DCNN. Following is a summary of our three contributions followed by sections for their details.

- **Hierarchical Deep Temporal Models:** In the first part of this thesis we use DCNN to build automatically feature representation of both individual person and whole scene while capturing temporal dynamics that exist at both levels. For temporal consistency purpose, Recurrent Neural Network (RNN) based models are used for handling variable length space-time inputs. Low-quality and small-scale datasets such as the Collective Activity Dataset [17] make training modern deep learning models difficult. To alleviate this shortcoming we build a new Volleyball Dataset of bigger size and higher quality than available ones.

- **Tubelets for Action Localization:** In the second part of this thesis we investigate the problem of action localization. The task of action localization is to localize the spatio-temporal extent (e.g., tublet) of a human action in video sequences. This is a core part of understanding human activity in general, since for many applications we would like to know when and where each individual human action takes place. We propose a framework that generates multiple generic class-independent tubelet proposals in each video (through a tube proposal network) then classifies them (through temporal understanding network).

- **Hierarchical Relational Network:** In a multi-person activity scene, incorporating contextual information regarding the actions of other people can provide useful cues for recognizing the action of an individual. One way to represent contextual information is through graphs where nodes are persons in the scene and edges encode their relationships (e.g., two persons are connected if their Euclidean distance is less than a threshold). To make use of such contextual input, we propose a Hierarchical Relational Network that computes relational representations of people, given graph structures describing potential interactions. Each relational layer is fed individual person representations and a potential relationship graph. Relational representations of each person are created based on their connections in this particular graph and the whole scene is represented by pooling the final individual representations.
1.1 Hierarchical Deep Temporal Models

The first component of this thesis examines the multi-person activity recognition problem. This work [50], and its extension [51], can be summarized in three-folds:

- Major advances in machine learning field don’t result only from advances in learning algorithms but also from availability of datasets. The previous standard dataset for multi-person activity recognition was the Collective Activity Dataset [17]. However, its small scale (2547 videos clips) and low resolution (720 by 480) make training modern deep learning models difficult. We contribute a new Volleyball Dataset that consists of 4830 video clips (1200 by 720), and is becoming a standard dataset (79 citations).

- Instead of limited hand-crafted representations, we use DCNN to build automatically feature representations of both individual person and the whole scene. Specifically, we build a model on top of features extracted from AlexNet [70] for boxes corresponding to people actions in a video sequence.

- We use Long Short-Term Memory (LSTM) [47] to capture the temporal dynamics existing at the person-level as well as the scene representations.

Formally, we present a deep architecture model for the multi-person activity recognition problem that consists of 2 stages. In the first stage, person actions are modeled using a CNN network extended with a temporal layer (LSTM). Once this stage is trained, we can express any detected person using spatio-temporal features. In the second stage, the representations of all players are combined (using a pooling strategy such as max-pool) and feed to a second temporal network. This second network captures the whole scene activity based on the actions of all players. The model is very generic and can handle a wide variety of multi-person activity domains, such as sports and surveillance domains. Also, we extend the model [51] to consider sub-grouping for people in the scene. For example, in a volleyball game, one can combine the team spatio-temporal features first and then concatenate teams representations. Such extensions help resolving confusions between teams in sports domains.

1.2 Tubelets for Action Localization

In the second part of the thesis we work on the action localization problem: given a trimmed video, recognize which specific actions are present and localize their spatio-temporal extent (typically represented as a sequence of bounding boxes in 2D frames, a.k.a tubelet, tube). The multi-person activity recognition problem is all about identifying the tubelets in a video sequence, classifying their actions and determining the whole scene activity. It is clear that spatio-temporal action localization problem is a sub-task in the multi-person activity understanding and finding better tubelets directly assists our thesis purpose.
One straightforward approach for identifying tublets, as done in our first proposed model, is to detect people in the first frame then use a visual-based tracker to track the remaining of the tublet for some $K$ steps. However, the tracker is run independently per person and no considerations for tublets intersections where tublets may get switched (e.g., the first part of tublet 1 is linked with the end part of tublet 2). To alleviate such shortcomings, we proposed a novel framework [45] that generates multiple generic class-independent tubelet proposals in each video then classifies them. Our approach consists of a tube proposal network (TPN) and a temporal understanding network (TUN). The TPN generates a set of video-level generic class-independent tube proposals and the TUN utilizes these class-independent video-level tube proposals to perform more sophisticated temporal understanding in each video. The output of the entire framework is a set of labeled action tubelets for each video. The proposed framework is described in detail as follows.

**Tube Proposal Network**

To generate accurate video-level tube proposals, state-of-the-art approaches typically require accurate 2D spatial box proposal. By "proposal boxes" we mean identifying set of bounding boxes that includes the objects in the scene. They are typically ranked based on score (estimate how likely that box encapsulates an object) and followed by algorithms that consider the boxes only rather than the whole search space of boxes in an image. We utilize the faster R-CNN [111] framework as the box proposal generator. However, we treat the detections in each frame as generic box proposals, not class-dependent ones (detections are considered as action foreground). Note that although the class-specific confidence scores are not used to generate generic tubelet proposals, they are used later in the TUN network. At time-step $t$, proposals in frame $t$ are linked using greedy algorithm to the tubelets generated from previous time-steps based on a linking schema. We greedily choose the top-$k$ tubelet proposals at each time-step. With the greedy algorithm, at each time-step, only the box proposals from the current frame and the previously computed tubelet proposals are needed.

**Temporal Understanding Network**

The video-level generic tubelet proposals are fed to the temporal understanding network (TUN) to perform more sophisticated temporal understanding. LSTM is performed on each of the generic tubelets to classify the generic tubelet proposals. We noticed that LSTM improves the classification performance on some action classes, and decreases the performance on others. Therefore, we fuse the confidence score from the TPN for this tube with the confidence score from LSTM for the same tube. The confidence score for a tube is calculated by averaging the confidence score from all frames in the video.
1.3 Hierarchical Relational Networks

Modeling structured relationships between people in a scene can provide useful cues for recognizing the action of an individual. For example, a jumping person in volleyball might perform a fake jump or a spike of the ball, but actions of surrounding players may resolve this confusion. Graphs can be used to represent these relationships: nodes are persons in a scene and edges encode their relationships (e.g., two persons are connected if their Euclidean distance is less than a threshold). We proposed a Hierarchical Relational Network [49] that computes relational representations of people, given graph structures describing potential interactions. Each relational layer is fed individual person representations and a potential relationship graph. Relational representations of each person are created based on their connections in this particular graph and the whole scene is generated by pooling the final individual representations.

Our relational network for multi-person activity recognition processes one video frame at a time. An input video frame has $K$ initial person feature vectors (where $K$ is the number of detected people in the scene and each person is represented with some DCNN features) associated with multiple potential relationship graphs. A single relational layer is fed with both $K$ feature representations and a relationship graph, and maps them to $K$ new relational representations.

The building block for our model is a relational unit that processes an individual person in the scene. Each person’s feature vector is mapped to a new representation by aggregating information from each neighbouring person in the relationship graph. The relational unit has a shared MLP (one per layer) with input that accepts two persons representations and learns an edge representation for their relationship. By aggregating all the edges representations of a specific person we obtain a new relational representation for him.

By stacking multiple layers, each with its own graph and relational unit parameters, we learn hierarchical relationship representations for the people. Pooling of the final person representations is used to construct the scene representation.

1.4 Summary of Contributions

We can summarize the contribution of the thesis as:

- **Hierarchical Deep Temporal Models:** We build deep temporal feature representation of person-level as well as the whole scene-level. We also introduce a Volleyball Dataset of bigger size and higher quality than the previous benchmarks. This work was published [50] as "Mostafa S. Ibrahim, Srikanth Muralidharan, Zhiwei Deng, Arash Vahdat, and Greg Mori. A hierarchical deep temporal model for group activity recognition. In CVPR, 2016."
• **Tubelets for Action Localization:** We propose a framework for the action localization problem: multiple generic class-independent tubelet proposals are generated in each video (through a *tube proposal network*) then classifies them (through *temporal understanding network*). This work was published [45] as "Jiawei He, Zhiwei Deng, Mostafa S. Ibrahim, and Greg Mori. Generic tubelet proposals for action localization. In WACV, 2018."

• **Hierarchical Relational Network:** We propose a Hierarchical Relational Network that computes relational representations of people, given graph structures describing potential interactions. Relational representations of each person are created based on their connections in this particular graph and the whole scene is represented by pooling the final individual representations. This work was published [49] as "Mostafa S. Ibrahim and Greg Mori. Hierarchical relational networks for group activity recognition and retrieval. In ECCV, 2018."
Chapter 2

Previous Work

Event recognition and detection in videos play a critical role in several computer vision systems such as visual surveillance, video summarization, and videos indexing. Such systems can be applied in several domains such as airport terminals, parking lots, traffic scenes, and banks. The complexity of a video scene may vary from a single person (or generally an object) doing a simple action up to a group of persons doing some collective activity. The community focused for a while on developing models for the single person case. With more understanding for the action level and the availability of complex datasets, researchers have been working on the multi-person scenarios as a more realistic case (e.g., recognizing a bank attack by robbers). Given the spatial and temporal nature of the problem and the need to translate low-level pixels information to high-level semantic information about the scene, this problem is very challenging.

There are several keywords related to this task such as gesture, action, behavior, activity, event, and agent. Unfortunately, the definitions vary from one research work to another. To avoid confusions, we will focus on a subset of these keywords and use the following definitions. Similar to Turaga et al. [138], action means a simple motion pattern by a single person (such as a walking person) while activity means a sequence of actions performed by several individuals evolving under spatial and temporal constraints (such as a group of players in a volleyball team playing their next attack). Notice that the temporal duration of any activity typically is larger than a single action. Also, for an activity, the same person may do several actions (e.g., a volleyball player jumps first, then spikes the ball). It is easy to find cases where such definitions have shortcomings (e.g., in music domains). However, such descriptions may fit with a wide variety of domains and datasets. We will use the terms group activity, multi-person activity and multi-agent activity to mean the same activity definition as defined.

This chapter will organize and summarize the previous research efforts about the group activity recognition problem using a single camera. However, the wider scope in literature may consider other cases such as vehicle-persons interactions [58], two persons interactions [98], multi-camera setups or sensor-based systems (e.g., using such as wearable cameras [80],
laser rangefinders [8] or GPS [100]). In the next sections, we will highlight the applications and related problems of our problem of interest. Then we will discuss and classify the literature research work for tackling the group activity recognition problem.

2.1 Applications

There are several applications where human activity recognition plays a critical role. In this section, we will refer to some of these applications.

**Sports Analysis:** Sports are as old as humans. We love to play and watch them. Moreover, we always have the interest to analyze and listen to analysis about different sports. That is why it is popular to find former athletes analyzing the new sports games, and people are giving them much interest. From the computer automation perspective, we may also have a stake to understand what happens in a match recorded or streamed as a frame sequence. Consider a volleyball game. Two teams, each of six players, are playing and interacting together and trying to win. The extreme amount of interactions and the non-linear tracklets of each player makes the problem very interesting. Meanwhile, we may wish to index the actions of players and activities of the whole teams in a database for search and indexing purposes.

**Security and Surveillance:** Think about all kind of institutions that have an interest in monitoring the safety of their buildings. For example, a bank wants to detect any suspicious behavior from people accessing the bank. Another example is in prisons where officers want to catch any violence and fightings between prisoners. Generally speaking, there is a broad type of applications that fit under this umbrella. Hence, security agencies try to build computer vision systems that detect, track and interpret human activities.

**Video search and Retrieval:** With the continuous growth of video materials on the internet, especially uploading services such as Youtube, we need to summarize and index the content of these videos for search and retrieval purposes. The wide variety of applications where human activity is the primary concern has several moments of interest for the gesture, action, event and activity elements. Indexing such information can be of great benefit for many parties. For example, one may want to get all winning moments for a particular team in the volleyball games of last Olympic.

2.2 Related Problems

Group activity recognition is typically viewed as a hierarchical and structured problem where we have several people doing specific actions evolving spatially and temporally, and their overall behavior is our interest. Due to that, several sub-problems are considered as part of these challenging problems. For example, we may detect the people in the scene and classify them. We also need to track these people through a temporal window. In this
section, we will highlight some research work in these related sub-problems.

**Action Recognition:** An extensive amount of research effort has been devoted to action recognition. See recent surveys [1] [21] for a summary of methods based on hand-crafted features. Following the impressive performance of CNNs (Convolutional Neural Network) on object recognition, deep learning approaches have been applied to the action recognition problem. Karpathy et al. [60] use frame stacks as input to the network to learn a deep temporal representation. However, it is a computationally expensive approach due to the learning of thousands of 3D convolutional filters. Although different temporal fusion strategies are evaluated, the performance is much lower than hand-crafted representations. Simonyan and Zisserman [128] train two separate ConvNets (one for an appearance feature using RGB images and another for motion features using optical flow images), and fuse their results to achieve performance competitive with the Improved Dense Trajectory approach [146]. Later work builds over this two-stream structure to boost the performance by utilizing motion cues. Zha et al. [154] aggregate CNN features from VGGnet [127] with dense trajectory features [146] using simple pooling strategies to produce spatio-temporal features for video classification. In their work, in-depth exploration of the right choices for the classification pipeline (such as pooling, feature normalization, CNN layers selection, classifiers) led to significant performance gain compared to other contemporary approaches.

**Object Detection:** The object detection task is to generate a set of bounding boxes, one around each object instance in an image. Hand-crafted features such as SIFT or HOG were heavily utilized for decades until the era of deep learning. The R-CNN approach [36] was the first to deploy CNNs for object detection successfully. It extracts a set of region proposals from an image using selective search [140], which is then fed through a fine-tuned CNN network separately to classify the object inside the proposed region. Although the approach achieved impressive performance, it is slow during both training and testing. Two critical improvements followed that approach: Fast R-CNN [37], where the features of the region proposals are computed efficiently using a shared network per image; and Faster R-CNN [111], where a Region Proposal Network (RPN) is introduced to generate the candidate regions efficiently. Importantly, the RPN and the R-CNN are jointly trainable.

**Object Tracking** The object tracking problem is to identify the tracklet (trajectory, the sequence of the bounding box of the person in the temporal domain) of the people. Notice that, we don’t care about the object label (contrary to action localization) and typically appearance is used for the tracking. Nillius et al. [91] link player trajectories to maintain identities via reasoning in a Bayesian network formulation. Morariu et al. [88] track players, infer part locations, and reason about temporal structure in 1-on-1 basketball games. In Soomro et al. [130], 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. [10] 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.

**Action Localization:** The action localization task is to both localize the action precisely in each frame, and classify the action correctly. Seminal work includes Ke et al. [61], who proposed a template-based method to build models for human action localization in crowded areas. These hand-labeled templates are matched based on shape and motion features against over-segmented spatio-temporal clips. Shechtman and Irani [122] proposed a space-time correlation method for actions in video segments with an action template based on enforced consistency constraints on the local intensity patterns of spatio-temporal tubes. A more promising direction than hand-crafted representations is recent attempts to learn deep networks for the action localization task. Typically, 2-D action regions are detected in each frame and linked to generate 3-D action volumes. One of the principal directions in this vein is Gkioxari and Malik’s work [38] that creates action tubes based on appearance and motion cues from two-stream CNN networks. In their approach, candidate regions are produced from each frame (using selective search [140]) and only the regions with enough motion saliency are kept. These regions are classified and scored using SVM and linked in time to build the actual action tubes. Several approaches [148] [116] [101] follow a similar direction with some changes. Weinzaepfel et al. [148] use a linking approach based on tracking, and the region candidates are generated using EdgeBoxes [159]. Saha et al. [116] come up with a better heuristic to fuse the two stream detections in each frame and also introduces a class-dependent hyperparameter to smooth path labeling. Peng and Schmid [101] further expands the two-stream structure into four streams by dividing the proposal region in each frame into an upper and lower region.

### 2.3 Handcrafted Era

For a long while, researchers used to build hand-crafted representation for the visual objects in an image. Recently with the appearance of deep learning, in many problems, it is the machine learning models responsibility to learn this representation by itself. In this section, we will introduce several types of such hand-crafted approaches.
Graphical Models

Graphical models combine probability theory and graph theory to express the conditional dependence structure between random variables in a graph format. A Bayesian network (BN), one of the graphical models, encodes the dependencies in directed graph format based on conditional probability densities (CPD). A Dynamic belief network (DBN) is a generalization that incorporates and considers the temporal dimension of the data, hence suitable to video sequence problems. The advantages of these probabilistic models include handling the uncertainty of the scene and allowing incomplete data. On the other side, fixing the network topology or finding proper priors are critical obstacles to the performance. We will introduce some of the important work that utilizes such techniques.

One of the remarkable work is Intille et al. [53] where large belief networks represent multi-agent interactions to recognize manually defined plays in American football. Each football play has a pre-defined temporal structure description representing the agents (individuals) and their goals (e.g., catch a pass) to be detected, such as in Fig 2.1 for the s511 play. Each goal is represented using a single belief network (manually specified in a tree-like shape). For agents, visual networks are applied where visual evidence is observed locally in space and time, and then incorporated into other visual goal networks. For each goal of an agent in a given frame, the likelihood curve over time is computed from its visual goal network. These likelihood values are integrated into a multi-agent belief network (generated automatically from the temporal structures) to recognize the final multi-agent action at a specific moment. Although the temporal modeling and constraining were only for around and before activities, the performance over the football games is good.

Although Intille et al. [53] made good utilization of the domain knowledge of American football, such pre-defined structure may not represent all the different temporal constraints in a game. Besides, typically such pre-defined style approaches are limited and can’t be transferred to other games that are less systematic or even to the general problem of group activity recognition.

Remagnino et al. [110] propose two-stage Bayesian network for modeling multi-agent actions in pedestrian and vehicles surveillance domain. In stage 1, a behavior agent Bayesian network represents the evolution of an agent (person or car) and is used to describe its actions. In Fig 2.2, the used fixed topology is presented. The topology is three levels with the lowest level being information such as object speed for a parent node representing the object dynamics. The top level represents the overall object behavior. In stage 2, the Bayesian network represents the situation agents. In this layer, every time two behavior agents come close to each other (using Euclidean distance measure) a situation agents BN is created. Such networks are used to compute final scene activities. Although the model is elegant, its components force limitations. For example, the fixed topology in stage one may not be able to represent complex scenarios in surveillance domains and more general
domains. Also in the second stage, having a BN representing only two interacting objects won’t fit with complex situations. For example, this method may fail in a multi-person activity where multiple people are interacting in a sophisticated manner. Later, in the deep learning era, Ibrahim et al. [50] [51] will utilize the two-stage model in a more stronger framework to resolve such problems.

Gong et al. [39] propose an extension of Hidden Markov Model (HMM) named the Dynamically Multi-linked Hidden Markov Model (A Dynamic Bayesian Network where variables are related to each other over adjacent time-steps) for the group activity recognition problem, especially for noisy outdoor scenes. In contrast to fully connected HMM approaches where a complete network structure is connecting every state at time t with all previous states, the state transition matrices where factorized to reduce the need for all such links. Such reduction in DML-HMM reduces the unnecessary parameters and builds a better state network. Having a smaller network also allows for faster computations when it comes to a reasoning about group activities with many objects evolving temporally. Once the parameters of the network are learned (e.g., through expectation minimization), we can extract an activity graph from the network’s transition matrices. Such graphs inform us of the temporal order of the different activity stages. Experimental results with DML-HMM shows superior performance compare to other HMM extensions in aircraft cargo activities.
However, with a small dataset generated from a fixed camera for only four different activities, the model’s actual performance for real life scenarios is not evaluated (e.g., efficiency and effectiveness in multi-agent scenarios where a large feature space is available).

Some weakly related problems to the multi-person activity recognition problems were addressed using graphical approaches. For example, the two-person interaction problem and human-object interactions problem. Park et al. [98] propose a three-level hierarchical Bayesian network for the recognition of two-person interactions such as pointing, punching, and hugging. In their model, the first level focused on simultaneously tracked body parts pose estimation, the second level on the overall body pose estimation and finally the third level to model the temporal evolution of the body (using a dynamic Bayesian network). A manual modeling for the collective experience of people interaction with the body poses is used to do the final recognition.

**Syntactic Approaches**

One exciting way in representing activities and their primitive actions is grammar based approaches through their production rules. A production rule sentence specifies a symbol substitution rule that can be applied recursively to generate new sentences. The activity recognition to grammar mapping can be thought as: the whole grammar rules represents the activity model, sentence represents activity and sentence word represents activity primitives. Brand et al. [11] propose one of the earliest work for utilizing context-free grammar (CFG) for visual activity modeling. In their work, they used simple grammar rules to recognize hand manipulations in sequences of discrete behaviors. There are extensions for such simple grammar approaches such as stochastic context-free grammar (SCFG) and attribute grammar.

**Stochastic context-free grammar (SCFG)** is the probabilistic extension for the CFG where every production rule is assigned a probability. One motivation behind this
direction is representing the low-level visual appearance of the activity using probabilistic approaches such as HMM and DPN where the errors in low-level processing stage (e.g., tracking and detection) can be considered. In Ivanov et al. work [54], a two-stage framework is proposed for activity recognition. In the first stage, low-level information is modeled using an HMM. In the second stage, Stochastic Context Free Grammer (SCFG) model is used to represent high-level semantics. In other words, the SCFG uses grammar production rules augmented with probabilities to compute the probability of the temporal actions of the objects trajectories which are provided from the HMM in the first level. In their model, longer range temporal constraints are handled by the SCFG which allow resolving uncertainty from the lower model.

To further support the CFG and SCFG approaches, attribute grammar can be used where additional attributes (e.g., object location) are associated with the production rules. Such attributes can be used to enforce semantic limitations on the interpretations of the grammar rules hence leading to a lower error rate and more consistent results. Joo et al. [58] use probabilistic attribute grammars for multi-agent activity recognition in surveillance domains. The primitive events (e.g., moveclose, moveaway of an agent) are associated with attributes (e.g., location, identity). Such attributes enforce semantic constraints on the activeness of the production rules. In Fig 2.3, attribute grammar rules for casing vehicles in a parking lot is associated with the grammar production rules.

SCFG and attribute grammars methods tend to be inefficient when comes to inference. As a result, typically, a sparse set of primitives actions is considered. To resolve this concern, Pirsiavash et al. [103] build a grammar model that captures the hierarchical temporal structure of the activities while still linear in time and constant in storage. In their work, they discover the sub-actions of an action using a latent SVM framework.

The evolution of CFG to SCFG and probabilistic attribute grammars resolved some of the challenges in activity recognition by better error considerations (e.g., considering
uncertainty) and semantic corrections. However, another important aspect of the evaluation is multi-person activity recognition where actions evolve temporally, and people interact together. In such models, the temporal constraints play a role in interpreting the scene. The nature of grammar based approach makes such task very challenging (e.g., manual modeling of rules). Results for learning such difficult rules from the training data seems not a promising direction so far [22] [92].

Logic-Based Approaches

What about utilizing domain knowledge? One of the logical directions is to formulate systems in terms of logical rules based on their domain knowledge. There are several sources of mistakes in recognition; however, semantics in a domain might help. Problems may come from the uncertainty in low-level components (such as trackers and detectors) or from the complexity of the task itself (multiple agents doing complex temporal activities under all natural conditions in computer vision (such as occlusions and camera view). One might express the domain knowledge as logical rules and model parts of the system through different levels (e.g., handle the uncertainty of low-level components or enforce semantics on high-level behavior). Let’s explore some approaches about logical-based approaches for knowledge domain representation.

Medioni et al. [87] describe a behavior analysis algorithm for a single agent activity. In their work, actions are categorized into several layers forming a hierarchical activity representation. Different scenarios (and their sub-scenarios) are manually determined and plugged into an automaton model (to handle the temporal evolution of the sub-scenarios) which is used to detect the final activities. Hongeng et al. [48] extend this approach to deal with multiple persons. The automaton model extension is to recognize human interactions (e.g., taking objects away) based on object trajectories.

Dinh et al. [137] model person-person and person-vehicle interactions using Markov Logic Networks (MLN) [112]. In MLN, the domain knowledge is expressed as first-order logic production rules. Each rule is assigned a weight indicating the belief in the accuracy of this rule. These rules are used by the Markov Logic Network to create the final network for probabilistic inference. In MLN, every predicate is assigned to all possible time intervals, however, this is computationally expensive. Morariu et al. [88] also use MLN to express temporal events using first-order logic rules through utilizing the domain knowledge. To avoid enumerating all possible time intervals, they use a bottom-up algorithm to discard the unlikely event hypotheses.

Close to MLN, Brendel et al. [12] propose a probabilistic event logic (PEL) approach for modeling the intervals of the events and imposing soft temporal constraints over them. In their work, the constraints are expressed based on weighted event-logic formulas, and the inference is implemented through an efficient stochastic local search algorithm that tries to avoid enumerating all the search space. In contrast to MLN, PEL utilizes a spanning-
Figure 2.4: From [12]. "Truth values are assigned to event occurrences, which have the form E@I, for event symbol E and time interval I = [a, b], where a and b are positive integers such that a < b. Asserting that E@I is true means that an instance of E occurred precisely over interval I. In the figure, the noisy detections are combined with the PEL knowledge base (KB)."

interval data structure to express intervals of a predicate ([s, e] in MLN to express 1 interval vs s=[s1, e1], e=[s2, e2] to express all intervals that start in the first range times all interval that ends in the second range). Fig 2.4 shows to the event logic representation for this approach.

Although logic-based methods are natural choices for domain knowledge modeling, they face several problems. For example, much effort is needed for a field to express its possible rules by domain experts. Also, one has to decide about where exactly to incorporate the domain knowledge (e.g., intermediate level vs high level or in all possible levels). On the other side, and more importantly, such methods are limited to activities with simple structural information. In other words, in some complex domains, such approaches are not fit at all due to failing to list logical rules to the people behavior.

**Ontology Approaches**

In addition to the logic-based methods in representing the domain knowledge, one other direction is through ontological modeling, representation, and reasoning. Through ontology modeling, we can express the domain entities with their (temporal) relationships and later plug-in the model in an inference technique to do reasoning about the situation inputs. We will highlight some approaches in this direction.

Chen et al. [15] propose an ontology-based methodology for understanding social interactions in nursing rooms (e.g., patient-patient, patient-nurse). To identify the core concepts in such ontology, 80 hours of video streams for nursing rooms are analyzed and knowledge is extracted and organized. Information on the individual level and people interactions were considered. In their work, they identified entities such as nursing home entity, person, facility, door, body, and face. For the inference, a dynamic Bayesian network corresponding
to the hierarchical structure of the concepts in the ontology is modeled to recognize the temporal evolving of the interaction between people in the nursing rooms. For the ontology structure, see Fig 2.5.

Hakeem et al. [44] build a system based on ontology modeling for meeting classification. Initially, an event ontology is built based on the movements of people hands. This ontology is extended to recognize people behaviors in the meetings (built based on the study of video streams of several meetings). The state machine is modeled (mapped from the event ontology) to detect different people events which are used by a rule-based system (mapped from the behavior ontologies) for recognizing overall people behaviors. Similarly, Georis et al. [35] model a system for semantic video interpretation of specific scenarios (e.g., bank attack) for bank monitoring.

As you may notice, several research directions are manually defining a specific ontology for a particular domain. To resolve this problem, [89] propose a video event ontology as a common knowledge base for the video surveillance concepts. Also, [33] propose VERL, the corresponding representation language for the video event ontology. Akdemir et al. [2] build over these generic representations while showing how to adapt them to a particular domain.
Figure 2.6: From [18]. "We seek to recognize collective activities such as queuing (left picture) or talking (right picture). In isolation, the highlighted individuals have very similar appearance, and thus it is not possible to identify whether they are talking (red) or standing in a queue (blue). However, by considering the spatio-temporal distribution of others (i.e., the crowded context), it becomes easier to recognize that the two individuals are performing different activities and to identify which activities are being performed."

in a systematic style, hence gaining more flexibility between the two directions (specific model versus general ontology).

As a knowledge-based approach, ontology modeling faces similar concerns such as Logic-based methods (e.g., critical need of domain experts to identify the events, concepts, temporal relationships, scenarios, etc.).

**Context-based Approaches**

Context plays a critical role in many computer vision problems such as image classification and object detection [97] [94]. Oliva et al. [94] refers to early studies that show the importance of context at multiple levels such as semantics (e.g., a table and chair are typically in the same image), and pose (e.g., chairs are oriented towards the tablet). In human-object interaction activities, [151] models mutual context of the object and human pose due to observing that both types serve as mutual context to each other and recognizing one facilitates the recognition of the other. In the group activity recognition problem, many works also utilize the context to build robust and discriminant classifiers [17] [18] [76]. To motivate the usage of context in this problem, think about two humans one queuing in video 1, and the another is talking in video 2. Notice how the visual appearance of both of cases is very close, and a classifier can’t differentiate between them. Only by considering the context, we may be able to resolve the conflict. Fig 2.6 highlights this concern. We will highlight several important approaches that utilized context to resolve these confusions [17] [18] [76].
Choi et al. [17] proposed the collective activity problem and collected a small dataset for it. In this problem, people are interacting together and doing the same activity. However, there might be few persons doing other activities. To resolve such confusions, the whole scene activity is defined as the major activity by the people. It is clear how the context plays a role in such a problem. In their work, they propose a spatio-temporal local (STL) descriptor based on histograms of people poses (facing left, right, forward, or backward) distributed around an anchor person. To create a temporal descriptor, the descriptors are combined over time and used as the feature vector for the scene classification stage. In this way, the descriptor holds the people spatial variations over time, hence captures some properties of the relative motions of the people. Such contextual performance helps final results to be robust. To get a more robust system, the movement information per individual is captured independently and used to assist the final performance. Fig 2.7 illustrates this descriptor.

Choi et al. [18] extend the STL descriptor [17] from a rigid decomposition of the feature space to an adaptive one. Their adaptive approach decomposes the space to variable spatial regions with variable temporal windows (versus fixed spatial and temporal ranges). To obtain such decomposition, random forests are used to compute the optimal number, shape, and size of the bins from the training data. Due to such adaptive approach, the model complexity and training time increases, but testing time doesn’t increase significantly. For
Figure 2.8: From [76]. 'Illustration of construction of our action context descriptor. (a) Spatio-temporal context region around focal person, as indicated by the green cylinder. In this example, we regard the fallen person as focal person, and the people are standing and walking as context. (b) Spatio-temporal context region around focal person is divided in space and time. The blue region represents the location of the focal person, while the pink regions represent locations of the nearby people. The first 3-bin histogram captures the action of the focal person, which we call the action descriptor. The latter three 3-bin histograms are the context descriptor, and capture the behavior of other people nearby. (c) The action context descriptor is formed by concatenating the action descriptor and the context descriptor.'

further output regularization, Markov Random Field (MRF) is applied over the random forest output.

Lan et al. [76] propose an action context descriptor centered around the person of interest aggregating person’s shape, motion and context. The context is mainly based on the people in the spatio-temporal volume around the target person. Instead of considering every person surrounding the focal person, they learn an adaptive graphic structure to decide whether two individuals should be regarded as interacting or not. In other words, instead of constructing a fully connected graph, they learn an adaptive graph structure by considering the structure as a latent variable. To construct an action descriptor for a person of interest, a multi-class SVM classifier is trained on the person features extracted from its region. Then a K-dimensional vector (k is the number of action classes) is constructed to represent the SVM scores for the k classes. Finally, to build the final action context descriptor for a person, the action descriptor for that person is concatenated with the action descriptors for the nearby considered persons (context descriptor). Fig 2.8 explains construction of the AC descriptor.

One of the exciting directions in the context-based approaches is the recognition of social roles [108] [32] [73] in the group activity recognition. In some types of group interactions (such as a birthday event), there are typically some social roles for the involved persons (such as the child having his birthday). The social role is a behavioral pattern assigned to a person from the social perspective and typically indicates the expectation from a person
while interacting with others. Identifying the social roles in such events may help to build robust models. Fig 2.9 illustrate a simple example for social roles. Lan et al. [73] model hockey games using a hierarchical approach of 3 levels: actions in lowest levels, social roles in intermediate level and finally the game event. They considered 11 actions (such as pass, receive, run, etc.), five social roles (such as attacker and defender) and 3 scene events (such as attack play). The social roles express the context information between people interactions (e.g., the first defender is typically close to the attacker). Different potentials are computed to capture the constraints in the different levels, and a max-margin framework is used for the training.

Close to social roles direction is the social interaction between people for event recognition [155] [16]. Zhang et al. [155] propose an approach for recognizing social patterns in prisons where interactions among people are the major concern (such as group dispersion, formation, and fighting). In their work, histogram features are extracted from each frame (based on probabilistic group analysis), then a frame is represented by the concatenating of its histogram to $T$ frames before and after the current frame. Using a bag-of-words style with an SVM classifier, they create a model to classify the videos. Choi et al. [16] modeled both people tracking and group activity recognition problems jointly where people social interactions is the key connection between the modeling of the two problems. In their work, a multi-layer framework is developed to represent a hierarchy of information (atomic actions, interactions and activities). Due to jointly handling the two problems, under several activity types considerations, a complex and slow inference mechanism (branch-and-bound, conditional random field, belief propagation, integer programming) is used. Similarly, Khamis et al. [63] [62] model jointly activity recognition object tracking and formulate the inference as an optimization problem (e.g., min-cost flow problem).

Most of the current state-of-the-art results incorporate context to resolve confusions. Incorporating context at several levels might be challenging and leads to slow approximate inference techniques [16] [73].
2.4 Deep Learning Approaches

Deep Convolutional Neural Networks (CNNs) have shown impressive performance by unifying feature and classifier learning, enabled by the availability of large labeled training datasets. Successes have been demonstrated on a variety of computer vision tasks including image classification [70] [127] and action recognition [128] [60]. More flexible recurrent neural network (RNN) based models are used for handling variable length space-time inputs. Specifically, LSTM [47] 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 [29] [40] [90]. In this section, we will highlight some of the deep learning approaches for the group activity recognition problem. Note that, these approaches were published in parallel to our efforts, not before it. More related work using deep learning is addressed in the next chapters.

Ibrahim et al. [50], our work, present a deep architecture model for the group activity recognition problem. Their model consists of 2 stages. In the first stage, the person actions are modeled using a CNN network extended with a temporal layer (long short-term memory (LSTM)). Once this stage is trained, we can express any detected person using spatio-temporal features. In the second stage, the representations of all players are combined (using a pooling strategy such as max-pool) and fed to a second temporal network. This second network captures the whole scene activity based on the actions of all the players. They applied their model to 2 datasets, collective activity dataset, and volleyball dataset. The model is very generic and can handle a wide variety of group activity domains. Later [51], they extend the model to consider sub-grouping for the people in the scene. For example, in a volleyball game, one can combine the team spatio-temporal features first and then concatenate teams’ representations. Such extensions should resolve confusions between teams in sports domains.

Close to our work [51], Ranamathan et al. [106] propose a similar 2-stage model, where the first stage builds representation for the players while the second stage classifies the whole scene based on fusing players information. Differences include the following. They try to attend for the key player in the frame to use it for the group activity label. That is, they train a model that can identify which player is the most informative at a given time \( t \). For that purpose, they collected a dataset for the basketball game where such requirement is available. In contrast to [51] that use max/average pooling strategies to fuse the players feature vectors, they use a weighting function based on attention models to recognize the key actor in the scene. One final difference between the two approaches is that this work uses weakly annotated dataset (specifically no labels per person). Technically speaking, this has no effect as both models build on deep learning representations such as classical CNN network (just [51] fine tune the network using players annotations to get better performance).
Bagautdinov et al. [7] propose a joint training to do action localization with the group activity recognition problem. Specifically, they propose a multi-person object detection approach that finds the persons in a frame, extracts their representation from the feature maps (using ROI layer) and links these detections based on Euclidean distance. After that, similar to [51], they feed the people representations in an RNN layer to get temporal representations for the people. Finally, they apply max-pooling over all persons to get the scene representation and apply softmax classification to compute the final prediction. In fact, the core of their work is about object detection. In comparison with [51], they use a multi-person object detector and link detections based on Euclidean distances, while [51] use a visual-based tracker based on the first detection. Still possible this change is behind the performance gap. In terms of the group activity recognition network, [51] use 2 stage processing networks (one for the person and other for the scene) while [7] use simplified network that learns person actions mainly. This change shouldn't affect performance, as [51] reported lower performance increase from their 2nd temporal network.

On other track, combining graphical models with deep CNN models is a possible direction. Deng et al. [25] build a 2-stage model that jointly learn the group activity, the person actions and the interactions between them. The first stage consists of 3 CNNs to learn the classification of the whole scene, person actions and persons poses. In the second stage, a graphical model approach is applied to refine the actions of the individuals and the overall scene activity. Specifically, the second stage is a neural network that mimics the message passing algorithm. Fig 2.10 highlights the big picture of this model. Deng et al. [24] propose another 2-stage model where the first stage is close to [25], and the second stage is graphical model to refine the scores from the first stage. This graphical model represents the scene with two different graph node types: a node for each person in the scene and another node for the scene itself. In their model, iteratively, the built inference machine reasons about people interactions and the main individuals involved in the group activity.

2.5 Datasets

Human activity recognition has a broad range of scenarios varying from videos with a single person up to many individuals evolving temporally in a complex way. The focus of this work is on the multi-persons setup. In the next section, we will highlight a summary over datasets with a focus on a single person, then in further sections, we will discuss the datasets that are more related to the group activity recognition setup.

Action Recognition Datasets

Action recognition problem mainly started with small scale datasets of focus over a single person. Recent datasets examples of such focus are KTH [118] and HMDB [71]. By the time, more sophisticated datasets were introduced such as TRECVID-MED [96]. Clips in these
datasets are typically trimmed (e.g., the event cover the whole clip frames). Other datasets (such as THUMOS [56] and ActivityNet [46]) provide untrimmed videos with temporal start/end annotations for temporal localization purposes. In the deep learning era, a vast scale dataset of 1 million sports video (Sports-1M) is introduced [60]. As mentioned, these datasets focus mainly on a single person, while our work is about the multi-person scenarios.

**Interaction Datasets**

Another related style of datasets is the interaction datasets, such as human-human interactions and human-object interactions. VIRAT dataset [93] is a large scale activity dataset for surveillance videos of realistic scenes involving persons, objects, and vehicles. Its videos vary in scales, resolutions, duration (2-15 minutes) and include 30 events. Some of its activities are such as person loading an object to a vehicle, person getting into a vehicle, person carrying an object and person walking. Fig. 2.11 highlights an example for some of its activities.

The UT-Interaction dataset [114] focuses on human-human interactions. It consists of 6 types of human activities: hand-shaking, hugging, kicking, pointing, punching, and pushing. The dataset consists of 20 video sequences (around 1 minute each), and each sequence contains one execution per activity. Ground truth of a sequence consists of the time interval and bounding boxes of people in the sequence. Fig. 2.12 shows the interactions.

Although these datasets are of good scale, their focus is on the interaction of very few objects.
Collective Activity Dataset

The Collective Activity Dataset [17] has been widely used for evaluating group activity recognition approaches in the computer vision literature [5, 25, 4]. This dataset consists of 44 videos, eight person-level pose labels (right, right-front, ..., right-back), five-person level action labels, and five group-level activities. The scene labels are the same for the people: crossing, waiting, queuing, walking, and talking. A scene is assigned a group activity label based on the majority of what people are doing. Notice that, only every 10th frame is annotated, not the whole video frames sequence. Hence, to build a temporal window for the people trajectories, we need to use a tracker.

Later, an extended version of this dataset is published by Choi et al. [16]. The extended CAD consists of 32 videos with eight poses annotated as the first dataset. However, action labels are different from scene labels, and there are extra interaction annotations added. Specifically: 3 individual actions (walking, standing still, and running), nine people interaction types (approaching, walking-in-opposite-direction, facing-each-other, standing-in-a-row, walking-side-by-side, walking-one-after-the-other, running-side-by-side, running-one-after-the-other, and no-interaction) and 6 activities (gathering, talking, dismissal, walking together, chasing and queuing).
Nursing Home Dataset

One of the interesting datasets is the Nursing Home Dataset [158]. It consists of 80 videos to detect falling persons in nursing rooms (e.g., dining rooms and corridors rooms). It has 6 actions (walking, standing, bending, squatting, sitting and falling). Short clips of 120 frames are sampled and assigned a binary label for evaluation purposes (e.g., falling vs non-falling). The dataset is used by some research work such as [24] [25].

SFU Volleyball Dataset

Sports area is another attractive area for the group activity recognition problem, where multiple players are moving in a complex and structured way to achieve some goals. This dataset is recently collected by Ibrahim et al. [50] for the volleyball game. It is relatively a large-scale one and contains two levels of annotations: 1) player action attached to his bounding box, 2) the overall scene activity of the whole players. Having annotations per a player is expensive annotating effort but allow creating strong models that understand each player independently. The dataset consists of 55 videos for volleyball games collected from Youtube with an overall of 4830 frames; each one is labeled for both the people and the scene. Specifically, there are nine individual actions (waiting, setting, digging, falling, spiking, blocking, jumping, moving, standing) and eight scene activities (right set, right spike, right pass, right winpoint, left winpoint, left pass, left spike, left set). In this dataset, the model should classify each testing frame. Notice that no people annotations are provided around the frame of interest, but one can track them similar to [50]. In the era of deep
learning, this one is a good start. However, the research community should target building larger scales datasets.

**NCAA Basketball Dataset**

Similar to the volleyball dataset, [106] collected a basketball dataset, but with 2 major differences. First, the dataset doesn’t have annotations for people actions (weakly annotated); hence the model has to find its way to compute scene labels without such annotation. Second, the dataset is much larger than the volleyball one (note that, annotating a scene is less time consuming than annotating both scene and actions). The dataset consist of 14548 short clips from 257 games, each clip has 24 frames and assigned one of 11 labels (3-point succ, 3-point fail, free-throw succ, free-throw fail, layup succ, layup fail, 2-point succ, 2-point fail, slam dunk succ, slam dunk fail and steal). For the purpose of learning a person detector, they also annotated 9000 frames from the training videos (e.g., one can use an object detector and train it using them). Finally, for evaluation purposes, they annotated the position of the ball on the frame where the shooter attempts a shot over (annotations from 850 test video clips). Fig 2.14 highlights this dataset.
Chapter 3

Hierarchical Deep Temporal Models for Group Activity Recognition

In this chapter we present our first approach for classifying the activity performed by a group of people in a video sequence. This problem of group activity recognition can be addressed by examining individual person actions and their relations. Temporal dynamics exist both at the level of individual person actions as well as at the level of group activity. Given a video sequence as input, methods can be developed to capture these dynamics at both person-level and group-level detail. We build a deep model to capture these dynamics based on LSTM (long short-term memory) models. In order 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, one LSTM model is designed to represent action dynamics of individual people in a video sequence and another LSTM model is designed to aggregate person-level information for group activity recognition. We collected a new dataset consisting of volleyball videos labeled with individual and group activities in order 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.

3.1 Overview

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 player is doing. This task of person-level action recognition is an important component of visual understanding. At another level of detail, we could instead ask what is the overarching group activity that is depicted. For example, this frame could be labeled as "right team setting." In this chapter, we focus on this higher-level group activity task, devising methods for classifying a video according to the activity that is being performed by the group as a whole.
Human activity recognition is a challenging computer vision problem and has received a lot of 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 a lot of applications in the context of video surveillance, sport analytics, video search and retrieval. A particular challenge of group activity recognition is the fact 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 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 [17, 73, 108, 5]. These approaches that utilize underlying person-level action recognition based on hand-crafted feature representations including histogram of gradients (HOG) or motion boundary histograms (MBH) both in a dense and sparse fashion [144], [118]. However, since they rely on shallow handcrafted feature representation, they are limited by their representational abilities to model a complex learning task. Similarly, the higher-level group activity recognition models utilize on 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 [128], [60], [70]. 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 [150, 121] 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 focus on the relations between people, presents a significant challenge to the learning algorithm. Similar challenges exist in the object recogni-
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.

In the recognition literature, and research often focuses on designing pooling operators for deep networks (e.g., [134]) that enable the network to learn effective classifiers.

Group activity recognition presents a similar challenge – appropriate networks need to be designed 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. This consists of one dedicated layer which reasons about individual people 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 temporal deep networks (LSTMs) to analyze each individual person. 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 work 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 the novel deep architecture that models group activities in a principled structured temporal framework. Our 2-stage approach models individual person activities 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 [29] and speech recognition [40]. Through the model structure, we aim at constructing a representation that leverages the discriminative information in the hierarchical structure between individual person actions and group activities.

We show that our algorithm works in two scenarios. First, we demonstrate performance on the Collective Activity Dataset [17], a surveillance-type video dataset. We also propose a new Volleyball Dataset that offers person detections, and both the person action label as well as the group activity label. 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 [50]. 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 3.2, we provide a brief overview of the literature related to activity recognition. In Section 3.3, we elaborate details of the proposed group activity recognition model. In Section 3.4, we tabulate the performance of the approach, and end in Section 3.5 with a conclusion of this work.

3.2 Related Work

Human activity recognition is an active area of research, with many existing algorithms. Surveys by Weinland et al. [21] and Poppe [104] explore the vast literature in activity recognition. Here, we will focus on the group activity recognition problem and recent related advances in deep learning.

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 that represent information between individuals in space and/or time domains. For example, Choi et al. [17] craft spatio-temporal feature representations of relative human actions. Lan et al. [76] proposed an adaptive latent structure learning that represents hierarchical relationships ranging from lower person-level information to higher group-level interactions.
Lan et al. [73] and Ramanathan et al. [108] explore the idea of social roles, the expected behaviour of an individual person in the context of group, in fully supervised and weakly supervised frameworks respectively. Lan et al. [73] 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. [108] define a CRF-based social role model under a weakly supervised setting. To learn model parameters and role labels, a joint variational inference procedure is adapted. HOG3D [68], spatio-temporal features [144], object interaction feature [81], and social role features [157] 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 [16] unified tracking multiple people, recognizing individual actions, interactions and collective activities in a joint framework. The model is based on the premise that strong correlation exists between an individual’s activity, and the activities of the other nearby people. Following this intuition, they come up with 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. [14] 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 a probabilistic motion analysis to extract interesting spatio-temporal patterns for scenario recognition. Vascon et al. [142] detect conversational groups in crowded scenes of people. The approach uses pairwise affinities between people based on pose and a game-theoretic clustering procedure.

In other work [18], a random forest structure is used to sample discriminative spatio-temporal regions from input video fed to 3D Markov random field to localize collective activities in a scene. Shu et al. [124] detect group activities from aerial video using an AND-OR graph formalism. Lillo et al. [84] proposed a hierarchical model of multiple levels: pose level, action level and activity level. The above-mentioned methods use shallow hand crafted features, and typically adopt a linear model that suffers from representational limitations.
Sport Video Analysis

Computer vision-based analysis of sports video is a burgeoning area for research. 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 [52], who examined stochastic representations of American football plays.

**Player tracking:** Nillius et al. [91] link player trajectories to maintain identities via reasoning in a Bayesian network formulation. Morariu et al. [88] track players, infer part locations, and reason about temporal structure in 1-on-1 basketball games. In Soomro et al. [130], 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. [10] 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. [139] perform activity recognition by first obtaining dense trajectories [144], clustering them, and finally employ a cluster set kernel to learn a action representations. Kwak et al. [72] optimize based on a rule-based depiction of interactions between people.

Wei et al. [147] 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. [126] proposed sparse multiple kernel learning to select features incorporated in a spatio-temporal pyramid. In Bialkowski et al. [9], 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. [6] 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 [27] solved a Poisson equation to
generate a holistic player location representation. Swears et al. [133] used the Granger Causality statistic to automatically constrain the temporal links of a Dynamic Bayesian Network (DBN) for handball videos.

In Gade et al. [34], 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.

**Deep Learning**

Deep Convolutional Neural Networks (CNNs) have shown impressive performance by unifying feature and classifier learning, enabled by the availability of large labeled training datasets. Successes have been demonstrated on a variety of computer vision tasks including image classification [70, 127] and action recognition [128, 60]. More flexible recurrent neural network (RNN) based models are used for handling variable length space-time inputs. Specifically, LSTM [47] 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 [29, 40, 90, 143].

For instance, in Donahue et al. [29], 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 work, they showed 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. This is then used to construct 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. [59], 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 [119] 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 is constructed, that outputs a probability map of the next word.

Alahi et al. [3] present a separate LSTM network per pedestrian to predict his trajectory in crowded scenes. A social pooling layer is introduced to connect and share 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. [30] also used multiple LSTMs to model action recognition from human skeletons. Based on human physical structure, they divide it to five parts and assign a separate LSTM to each part.

In this work, we aim at building 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. [135] for pose estimation, and Zheng et al. [156] and Schwing et al. [120] for semantic image segmentation.

In Deng et al. [25] 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 [152], a new dataset is introduced that contains 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. [106] 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 represent aggregations of all people involved in a group activity.

Datasets

Popular datasets for activity recognition include the Sports-1M dataset [59], UCF 101 database [131], and the HMDB movie database [71]. 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 terms of extraneous background objects and varying temporal duration of the action of interest. However, these datasets tend to focus on individual human actions, as opposed to the group activities we consider in our work.

Scenes involving multiple, potentially interacting people present significant challenges. In the context of surveillance video, the TRECVis Surveillance Event Detection [95], UT-Interaction [113], VIRAT [93], and UCLA Courtyard datasets [4] are examples of challenging tasks including individual and pairwise interactions.

Datasets for analyzing group activities include the Collective Activity Dataset [17]. This dataset consists of real-world pedestrian sequences where the task is to find the high level group activity. The S-HOCK dataset [19] 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 sport footage which is annotated with player pose, location, and group activities.
3.3 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 tracklets 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 can aid in determining 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 can be used to infer the group’s 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 have modeled these elements in a form of structured prediction based on hand crafted features [144, 118, 76, 73, 108]. Inspired by the success of deep learning based solutions, in this chapter, a novel hierarchical deep learning based model is proposed that is potentially 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 that can learn dynamic temporal behavior from a sequential input with arbitrary length. Therefore, they overcome the limitation of CNNs that expect constant length input. This makes them widely applicable to video analysis tasks such as activity recognition.

Our model is inspired by the success of hierarchical models. 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.

3.3.1 Temporal Model of Individual Action

Given tracklets of each person in a scene, we use long short-term memory (LSTM) models to represent temporally the action of each individual person. Such temporal information is
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.

complementary to spatial features and is critical for performance. LSTMs, originally proposed by Hochreiter and Schmidhuber [47], have been used successfully for many sequential problems in computer vision. Each LSTM unit consists of several cells with memory that stores information for a short temporal interval. The memory content of a LSTM makes it suitable for modeling 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 in avoiding 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 a large number of such layers in order 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 a person trajectory. The output of the CNN, represented by $x_t$, can be considered as 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
activation can be formulated as:

\[ i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \] (3.1)
\[ f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \] (3.2)
\[ o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \] (3.3)
\[ g_t = \phi(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \] (3.4)
\[ c_t = f_t \odot c_{t-1} + i_t \odot g_t \] (3.5)
\[ h_t = o_t \odot \phi(c_t) \] (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 modeling individual actions, the hidden state \( h_t \) could be used to model the action a person is performing at time \( t \). Note that the cell output is evolving over time based on the 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 simply pass the output of the LSTM cell at each time to a softmax classification layer\(^1\) to predict individual person-level action for each tracklet.

The LSTM layer on top of person trajectories forms the first stage of our hierarchical model. 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 pasing the hidden states of the LSTM layer to the second stage for group activity recognition, as discussed in the next section.

### 3.3.2 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 as well as past changes in his action. If the memory content is collected over all people in the scene, it can be used to describe the group activity in the whole scene.

Moreover, it can also be observed that direct 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 to extract complex features for each person in addition to the temporal features captured by the first LSTM layer.

At this moment, the concatenation of the CNN features and the LSTM layer represent temporal features for a person. Various pooling strategies can be used to aggregate these

\(^1\)More precisely, a fully connected layer fed to softmax loss layer.
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 directly model the **temporal dynamics of group activity**. The LSTM layer of the second network is directly connected to a classification layer in order to detect group activity classes in a video sequence.

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

\[
P_{tk} = x_{tk} \oplus h_{tk}
\]

\[
Z_t = P_{t1} \circ P_{t2} \ldots \circ P_{tk}
\]

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 \( k^{th} \) person at time \( t \). We concatenate these two features (represented by \( \oplus \)) to obtain the temporal feature representation \( P_{tk} \) for \( k^{th} \) person. We then construct the frame level feature representation \( Z_t \) at time \( t \) by applying a max pooling operation (represented by \( \circ \)) 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 input of the second-stage LSTM layer. The hidden state of the LSTM layer represented by \( h^{\text{group}}_t \) carries temporal information for the whole group dynamics. \( h^{\text{group}}_t \) is fed to a softmax classification layer to predict group activities.

### 3.3.3 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 confusions between left team 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 sub-group’s representation. In our experiments we consider a set of different possible spatial sub-groupings of players (c.f. standard spatial pyramids [78]). 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:

\[ P_{tk} = x_{tk} \oplus h_{tk} \quad (3.9) \]

\[ S_m = (m - 1) * k/d + 1 \quad (3.10) \]

\[ E_m = m * k/d \quad (3.11) \]

\[ G_{tm} = P_{tSm} \odot P_{t(Sm+1)} \ldots \odot P_{tEm} \quad (3.12) \]

\[ Z_t = G_{t1} \oplus G_{t2} \ldots \oplus G_{td} \quad (3.13) \]

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 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 the frame level feature representation constructed by the concatenation operator (represented by \( \odot \)) of the \( d \) sub-groups.

3.3.4 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 tracklets annotated with action labels. This network is shared among all people in the scene. We implement our model using Caffe [55]. Similar to other approaches [29, 25, 143], 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. [20], implemented in the Dlib library [65]. The baseline models are structured and trained in a similar manner as our two-stage model.

3.4 Experiments

In this section, we evaluate our model by running ablation studies using several baselines and comparing to previously published works on the Collective Activity Dataset [17]. First,
Figure 3.3: Illustration of 2-group pooling to capture spatial arrangements of players.
we describe our baseline models for the ablation studies. Then, we present our results on
the Collective Activity Dataset followed by experiments on the Volleyball Dataset.

3.4.1 Baselines

The following baselines are considered in all our experiments in order to assess the contri-
butions 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 fed to a softmax classifier to
recognize group activities in each single frame.

B3) **Fine-tuned Person Classification:**

This baseline is similar to the previous baseline with one distinction. The AlexNet
model on each player is fine-tuned to recognize person-level actions. Then, fc7 features
are pooled over all people, and are fed to a softmax classifier to recognize group
activities in each single frame. The rationale behind this baseline is to examine a
scenario where person-level action annotations as well as group activity annotations
are used in a deep learning model that does not model the temporal aspect of group
activities. This is very similar to our two-stage model without the temporal modeling.

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
deployed on the whole image and resulting fc7 features are fed to a LSTM model.
This baseline can be considered as a reimplementation of Donahue et al. [29].

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, omit-
ting the person-level temporal model (LSTM 1). Instead, the person-level classification
is done only with the fine-tuned 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 person action
labels, but without an additional group-level LSTM.
B8) **Simple Temporal Model:** To evaluate the need for the LSTM as a non-trivial temporal model, this baseline is based on a simple 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.

### 3.4.2 Experiments on the Collective Activity Dataset

The Collective Activity Dataset [17] has been widely used for evaluating group activity recognition approaches in the computer vision literature [5, 25, 4]. This dataset consists of 44 videos, eight person-level pose labels (not used in our work), 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 research works (e.g., 1/4 data for testing in [5] vs 1/3 data in [25, 43]). We followed the division by [25, 43] 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, 9 timesteps and 3000 hidden nodes are used for the first LSTM layer and a softmax layer is deployed 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 passed to a softmax layer trained to recognize group activity labels.

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>B1-Image Classification</td>
<td>63.0</td>
</tr>
<tr>
<td>B2-Person Classification</td>
<td>61.8</td>
</tr>
<tr>
<td>B3-Fine-tuned Person Classification</td>
<td>66.3</td>
</tr>
<tr>
<td>B4-Temporal Model with Image Features</td>
<td>64.2</td>
</tr>
<tr>
<td>B5-Temporal Model with Person Features</td>
<td>64.0</td>
</tr>
<tr>
<td>B6-Two-stage Model without LSTM 1</td>
<td>70.1</td>
</tr>
<tr>
<td>B7-Two-stage Model without LSTM 2</td>
<td>76.8</td>
</tr>
<tr>
<td>B8-Simple Temporal Model</td>
<td>77.7</td>
</tr>
<tr>
<td><strong>Two-stage Hierarchical Model</strong></td>
<td><strong>81.5</strong></td>
</tr>
</tbody>
</table>

Table 3.1: Comparison of our method with baseline methods 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 can be made between
<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contextual Model [76]</td>
<td>79.1</td>
</tr>
<tr>
<td>Deep Structured Model [25]</td>
<td>80.6</td>
</tr>
<tr>
<td><strong>Our Two-stage Hierarchical Model</strong></td>
<td>81.5</td>
</tr>
<tr>
<td>Cardinality kernel [43]</td>
<td>83.4</td>
</tr>
<tr>
<td>Shu et al. [123]</td>
<td><strong>87.2</strong></td>
</tr>
</tbody>
</table>

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

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. [25] is a previously published deep learning model. In contrast, the cardinality kernel approach [43] outperformed our model. It should be noted that this approach works on hand crafted 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 the performance of previous group activity recognition methods: e.g., crossing involves the walking action, but is confined in a path which people perform in orderly fashion. Therefore, our model that is designed only to learn the dynamic properties of group activities often gets confused with the walking action.

It is clear that our two-stage model has improved performance with compared to 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.

**3.4.3 Experiments on the Volleyball Dataset**

In order 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 that were 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 follows from the fact that we often have people present in static actions like standing compared to dynamic actions (setting, spiking, etc.). Therefore, our dataset presents a challenging team activity recognition task, where we have interesting actions that can directly determine the group activity occur rarely in our dataset. The dataset has been made publicly available to facilitate future comparisons.

**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 point (x-axis first). To split players to two groups (e.g., left/right teams), we consider the first half of players as group one. Similarly, to split to four groups, we consider the first quarter of players as group one, second quarter as group two and so on. If players cannot be divided evenly (missing players), the last sub-groups will have fewer players.

<table>
<thead>
<tr>
<th>Group Activity Class</th>
<th># Instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Right set</td>
<td>644</td>
</tr>
<tr>
<td>Right spike</td>
<td>623</td>
</tr>
<tr>
<td>Right pass</td>
<td>801</td>
</tr>
<tr>
<td>Right winpoint</td>
<td>295</td>
</tr>
<tr>
<td>Left winpoint</td>
<td>367</td>
</tr>
<tr>
<td>Left pass</td>
<td>826</td>
</tr>
<tr>
<td>Left spike</td>
<td>642</td>
</tr>
<tr>
<td>Left set</td>
<td>633</td>
</tr>
</tbody>
</table>

Table 3.3: Statistics of the group activity 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 modeling 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 temporal dynamics of the foreground. 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, in spite of the problems that arise due to tracking and motion artifacts.

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>B1-Image Classification</td>
<td>66.7</td>
</tr>
<tr>
<td>B2-Person Classification</td>
<td>64.6</td>
</tr>
<tr>
<td>B3-Fine-tuned Person Classification</td>
<td>68.1</td>
</tr>
<tr>
<td>B4-Temporal Model with Image Features</td>
<td>63.1</td>
</tr>
<tr>
<td>B5-Temporal Model with Person Features</td>
<td>67.6</td>
</tr>
<tr>
<td>B6-Two-stage Model without LSTM 1</td>
<td>74.7</td>
</tr>
<tr>
<td>B7-Two-stage Model without LSTM 2</td>
<td>80.2</td>
</tr>
<tr>
<td>B8-Simple Temporal Model</td>
<td>78.1</td>
</tr>
<tr>
<td><strong>Our Two-stage Hierarchical Model</strong></td>
<td><strong>81.9</strong></td>
</tr>
</tbody>
</table>

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 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.

It seems from the results that more brute force sub-grouping doesn’t improve the performance of the system for this dataset. It shows that extracting additional information by
segregating players on basis of their position renders information from static/insignificant players results in more confusion, and perhaps leading to degradation in performance. Therefore, from this experiment, it is evident all types of explicit spatio-temporal relation modelling does not 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 highest signals per a bin. One interpretation for that might be 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. [106], which used average weighting (an attention based model) for player representation to detect the key players in basketball games.

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Model - 1 group - max pool</td>
<td>70.3</td>
</tr>
<tr>
<td>Our Model - 1 group - avg pool</td>
<td>68.5</td>
</tr>
<tr>
<td>Our Model - 2 groups - max pool</td>
<td>81.9</td>
</tr>
<tr>
<td>Our Model - 2 groups - avg pool</td>
<td>80.7</td>
</tr>
<tr>
<td>Our Model - 4 groups - max pool</td>
<td>81.5</td>
</tr>
<tr>
<td>Our Model - 4 groups - avg pool</td>
<td>79.6</td>
</tr>
</tbody>
</table>

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 number of LSTM nodes of the model’s two network, we conducted set of experiments outlined in Table 3.7. Similarly, we evaluate the effect of the number of timesteps of the model’s two network, we conducted set of experiments outlined in Table 3.8.

<table>
<thead>
<tr>
<th>Method</th>
<th>No. Person LSTM Nodes</th>
<th>No. Scene LSTM Nodes</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Model</td>
<td>1000</td>
<td>1000</td>
<td>79.4</td>
</tr>
<tr>
<td>Our Model</td>
<td>2000</td>
<td>1000</td>
<td>80.3</td>
</tr>
<tr>
<td>Our Model</td>
<td>3000</td>
<td>1000</td>
<td>81.2</td>
</tr>
<tr>
<td>Our Model</td>
<td>3000</td>
<td>2000</td>
<td>81.9</td>
</tr>
<tr>
<td>Our Model</td>
<td>3000</td>
<td>3000</td>
<td>81.2</td>
</tr>
</tbody>
</table>

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.

**Comparison to other methods:** In Table 3.9, we compare our model to the improved dense trajectory approach [146]. Dense trajectories is a hand-crafted approach that competes strongly versus deep learning features. In addition, we also created two variations of [146], where the considered trajectories are only the ones inside the players’ bounding boxes, in
<table>
<thead>
<tr>
<th>Method</th>
<th>No. Person timesteps</th>
<th>No. Scene timesteps</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Model</td>
<td>5</td>
<td>10</td>
<td>81.9</td>
</tr>
<tr>
<td>Our Model</td>
<td>5</td>
<td>20</td>
<td>81.7</td>
</tr>
<tr>
<td>Our Model</td>
<td>10</td>
<td>10</td>
<td>81.7</td>
</tr>
<tr>
<td>Our Model</td>
<td>10</td>
<td>20</td>
<td>81.3</td>
</tr>
</tbody>
</table>

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

other words, ignoring background trajectories. The variations emulate our model with one group and 2 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.

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Model using 2 groups style</td>
<td>81.9</td>
</tr>
<tr>
<td>IDTF [146] - all trajectories</td>
<td>73.4</td>
</tr>
<tr>
<td>IDTF [146] - 1 group-box trajectories</td>
<td>71.7</td>
</tr>
<tr>
<td>IDTF [146] - 2 groups-box trajectories</td>
<td>78.7</td>
</tr>
<tr>
<td>Shu et al. [123]*</td>
<td>83.3</td>
</tr>
<tr>
<td>Bagautdinov et al. [7]*</td>
<td><strong>90.6</strong></td>
</tr>
</tbody>
</table>

Table 3.9: Comparison of the team activity recognition of our model against improved dense trajectory approach approach. Approaches with * are recent publications.

The traditional dense trajectories approach and its one group style show close performance, but the 2-groups trajectories variation yields higher performance. Probably, this is due the reduction of confusions between left and right teams’ activities. However, our model outperforms these strong 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 [50]. 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. On the contrary to [50], 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.

3.5 Summary

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 the representation of individual people to recognize the group activity. We created a new Volleyball Dataset to train and test our model, and 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 can work with much less training data. For example, if training annotations for the person level are not available, we can just use a pre-trained network for representing people. However, we would expect the model’s performance to degrade. Also, 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

Generic Tubelet Proposals for Action Localization

In this chapter we investigate the problem of action localization. The task of action localization is to localize the spatio-temporal extent (a.k.a tublet) of a human action in video sequences. This is a core part of understanding human activity in general, since for many applications we would like to know when and where each individual human action takes place. We propose the Tube Proposal Network (TPN), which can generate generic, class-independent, video-level tubelet proposals in videos. The generated tubelet proposals can be utilized in various video analysis tasks, including recognizing and localizing actions in videos. In particular, we integrate these generic tubelet proposals into a unified temporal deep network for action classification. Compared with other methods, our generic tubelet proposal method is accurate, general, and is fully differentiable under a smoothL1 loss function. We demonstrate the performance of our algorithm on the standard UCF-Sports, J-HMDB21, and UCF-101 datasets. Our class-independent TPN outperforms other tubelet generation methods, and our unified temporal deep network achieves state-of-the-art localization results on all three datasets.

4.1 Overview

Action understanding in videos has many applications such as sports analysis, video surveillance, and content-based retrieval. Detailed understanding requires localizing and classifying human actions in videos, answering the questions "what is the action in the video?" and also "where exactly is the action in the video?"

Breakthroughs in image understanding have been achieved using the Convolutional Neural Network (CNN) structure. The efficacy and efficiency of such structures in the action recognition area lag behind. Strong results have been achieved for video-level analysis by adapting CNN architectures to classification tasks (e.g., [29, 60]). However, in action-related videos actions typically occur only within a limited spatial extent inside each frame. CNN
Figure 4.1: Action localization with our Tube Proposal Network produces generic class-independent tubelets. The tubelets are classified with a Temporal Understanding Network that can perform detailed spatio-temporal analysis.

Frameworks that represent entire video frames lack focus on relevant features, are susceptible to background clutter and camera motion, and cannot perform action localization.

Finding the action region in videos can aid in developing more accurate models. Recent advances in object detection using region proposal networks (e.g., [111]) have triggered a flurry of research in action localization. A major challenge is addressing the large search space of video content in terms of analyzing potential regions as action proposals. Most recent work on spatial-temporal localization shares a similar framework [38, 101, 116]. These approaches first detect specific actions at a frame level by utilizing 2-D detection networks (R-CNN [36] or faster R-CNN [111]). The class-specific detections in each frame are then linked or tracked, resulting in class-specific action tubes. To summarize, these methods tend to treat a video as a set of independent images. Therefore, action localization is performed on each image separately, losing much potential to exploit temporal relationships in each video.

Figure 4.2: The proposed framework consists of two components: a tubelet proposal network (TPN) and a temporal understanding network (TUN). A TUN based on fused LSTMs is used to classify the generic class-independent tubelet proposals generated by the TPN.

Based on the discussion above, we propose a novel framework that can utilize both spatial and temporal information to spatially localize and classify actions in a video sequence. The framework can automatically generate multiple generic class-independent tubelet proposals in each video. This approach with generic tubelets alleviates important shortcomings with previous methods. Notably, the challenging classification step can be conducted using...
all available temporal information in a tubelet. This obviates the need to make difficult classification decisions at a frame level, allows correction for camera motion, and discards background clutter. We delay the classification till a final step where both per-frame classification scores and temporal scores are both available. The temporal information is crucial due to the fact that actions are defined over time, not on each single frame. Further, a single set of tubelets is generated in one pass, usable for all further analysis. Finally, the proposal method is generic and can be easily integrated into other spatio-temporal deep networks.

We term our framework as the combination of a Tube Proposal Network (TPN) and a Temporal Understanding Network (TUN). The TPN generates multiple video-level, generic, class-independent tubelet proposals. A tubelet proposal is a spatio-temporal region proposal, analogous to the generic region proposals used in object detection. These generic tubelet proposals are then fed into the TUN, which can incorporate temporal information for action analysis. A variety of temporal models such as temporal pooling [60], LSTM [47], or C3D [136] could be easily used in the TUN. In this work, a temporal model based on fused LSTMs is utilized as the TUN to classify the generated generic tubelet proposals.

In summary, the main contributions of this chapter are: (1) a novel tubelet proposal network generating generic video-level tubelet proposals to enable person-centric action recognition; (2) a general TPN / TUN structure that can be easily instantiated using a variety of video understanding network architectures; (3) development of a specific TUN that can exploit spatio-temporal features for classification in each tubelet; and (4) state-of-the-art performance on the three standard datasets for spatial action localization.

4.2 Related Work

Inspired by the promising performance of CNNs in image classification and object detection, deep learning architectures have been increasingly utilized in video analysis. We start this section by reviewing relevant work in general action recognition and object detection, before turning our attention to action localization.

Action Recognition: Extensive research effort has been devoted to action recognition. Recent surveys [1, 21] summarize methods based on hand-crafted features.

Following the impressive performance of CNNs in object recognition, deep learning approaches were applied to action recognition. Karpathy et al. [60] use frame stacks as input to a network to learn a deep temporal representation. Simonyan and Zisserman [128] train two separate ConvNets (one for an appearance feature using RGB images and another for motion features using optical flow images), and fuse their results to achieve performance competitive with the Improved Dense Trajectory approach [146]. Later work builds on this two-stream structure to boost accuracy by utilizing motion cues. Zha et al. [154] aggregate CNN features from VGGnet [127] with dense trajectory features [146] using simple pooling strategies to produce spatio-temporal features for video classification. An in-depth
exploration of the right choices for the classification pipeline (pooling, feature normalization, layers, classifiers) led to significant performance gain compared to contemporary approaches.

**Object Detection:** The object detection task is typically posed as generating a bounding box around each object instance in an image. Hand-crafted features such as SIFT or HOG were utilized in the pre-AlexNet era. The R-CNN approach [36] was the first to successfully deploy CNNs for object detection. It extracts a set of region proposals from an image using selective search [140], which are then fed through a fine-tuned CNN network separately to classify the object inside the proposed region. Although the approach achieved impressive performance, it is slow during both training and testing. Two critical improvements followed that approach: Fast R-CNN [111], where the features of the region proposals are computed efficiently using a shared network per image; and Faster R-CNN [111], where a region proposal network (RPN) is introduced to efficiently generate the candidate regions.

**Action Localization:** The action localization task is similar to object detection – the goal is to spatially/temporally localize a recognized action within a video, often using a per-frame bounding box representation.

Seminal work includes Ke et al. [61], who proposed a template-based method to build models for human action localization in crowded areas. These hand-labeled templates are matched based on shape and motion features against over-segmented spatio-temporal clips. Shechtman and Irani [122] proposed a space-time correlation method for actions in video segments with an action template based on enforced consistency constraints on the local intensity patterns of spatio-temporal tubes. Lan et al. [74] used latent SVM learning to jointly detect and recognize actions in videos based on a figure-centric visual word representation. Van Gemert et al. [141] utilize dense trajectory features for region proposal and classification.

A promising direction is recent attempts to learn deep networks for action localization. Typically, 2-D action regions are detected in each frame and linked to generate 3-D action volumes. One of the leading directions in this vein is Gkioxari and Malik’s work [38] that generates action tubes based on appearance and motion cues from two-stream CNN networks. In their approach, candidate regions are generated from each frame (using selective search [140]) and only the regions with enough motion saliency are kept. These regions are classified and scored using an SVM and linked in time to build the final action tubes.

Several approaches [148, 116, 101] present interesting improvements on this direction. weinzaepfel15ActionLocalization et al. [148] uses a linking approach based on tracking, with region candidates generated using EdgeBoxes [159]. Saha et al. [116] proposes a better method to fuse the two stream detections in each frame and smooth path labeling. Peng and Schmid [101] further expands the two-stream structure into four streams by dividing the proposal region in each frame into upper and lower regions. These two methods improve performance significantly compared with [38], by updating the CNN from Alexnet (used in [38]) to VGGnet, and replacing selective search with the region proposal network for
more accurate 2-D region proposal generation. Further improvements are possible by class-specific approaches, e.g., [116] obtains a 6% increase in performance via a class-dependent hyper-parameter.

Although the recent state-of-the-art methods [38, 116, 101] are very effective, they all treat each frame independently in many classification stages, reducing the temporal information present in a video sequence. Different from these methods, we first generate a set of generic class-independent tubelet proposals for each video, which are then classified using a temporal model. Our model thereby can build detailed person-centric models for exploiting both spatial and temporal information for localizing and classifying actions.

4.3 Proposed Approach

A key challenge in action recognition is that individual frames are ambiguous – given a single image of a person in an upright pose, this could correspond to a person walking, standing, kicking a ball, or myriad other actions that include a fleeting moment of similarly neutral posture. Furthermore, human action involves a variety of different poses. Accurately detecting people contorted into varied poses is a challenge, and utilizing all available "human action-like" data to build a detector seems favourable to building action-specific human detectors.

For these reasons, we advocate the building of generic tubelets for action localization. Generic tubelets can leverage all available human action training data for their detection. Person-centric tubelets can be constructed, then classified into action categories in their entirety. In order to accurately recognize actions in video, it makes sense to delay making hard classification decisions until all information present in video sequence has been observed.

We operationalize these ideas via our proposed approach, consisting of a tube proposal network (TPN) and a temporal understanding network (TUN), as illustrated in Fig. 4.2. The TPN generates a set of video-level generic tubelet proposals and the TUN utilizes these tubelet proposals to perform more sophisticated temporal understanding in each video. The output of the entire framework is a set of labeled action tubes for each video. The proposed framework is described in detail below.

4.3.1 Tube Proposal Network

The goal of the TPN is to generate spatio-temporal region proposals (tubelet proposals) where it is likely that an action occurred. Importantly, at this stage of analysis we are not concerned with the action category, just whether any action is taking place at this location.

A tubelet proposal is constructed by linking frame-level spatial proposals in time. Frame-level proposals are represented as bounding boxes, ideally covering the action region. The obvious advantage of this method is that the tubelet proposal moves with the person in time, focusing on the action region specifically. This mechanism enables person-centric region-of-
interest (RoI) feature extraction, which we show leads to improved classification accuracy. Furthermore, since the proposals we link are generic, the localization accuracy can be improved (see Sec. 4.3.1). As a result, our framework improves both localization and classification accuracy. The TPN consists of two parts: 2-D generic region proposal generation, and 3-D generic tubelet proposal generation.

2-D Region Proposal Generation

To generate accurate video-level tubelet proposals, we first acquire a set of 2-D action region proposals in each frame, which breaks down the task to frame-level action detection. A typical object detection framework in images consists of two parts: region proposal generation / sliding window, and a network to classify and regress the proposals into final bounding boxes with labels. The regressor in recent object detection works is class-dependent: each region proposal will be regressed into a set of bounding boxes, based on classification. Therefore, it is crucial to classify the region correctly. However, this is not practical in action analysis tasks since an action often occurs over a sequence of frames. When examining a single frame, it can be difficult to discern the action category, as per the examples in Fig. 4.3. As a result, a class-specific regressor will face challenges in creating accurate spatial proposals that are consistent across time and action category.

However, a generic, class-agnostic proposal mechanism can alleviate these shortcomings. As long as a bounding box appears to contain human action, it should be included as a potential region for some action to have occurred. The actual decision on which action it is can be deferred to a future analysis step.

As an example, consider the second image in Fig. 4.3, which could reasonably be proposed as either a walking, standing, or running region. Based on this intuition, we perform a generic class-agnostic region proposal and regression, by treating the regressed bounding box with the highest action score for each proposal as our generic region proposal. The aforementioned discussion shows that treating the regressed bounding boxes as generic region proposals is essential and effective.

For implementation, we utilize the faster R-CNN [111] framework as the 2-D region proposal generator. Note that although the class-specific action scores are not used to generate tubelet proposals, they are saved to help with classification, as described in Sec. 4.3.2.
Table 4.1: Ablation study of our proposed framework on J-HMDB21 (split 1). Remaining classes in the next table.

<table>
<thead>
<tr>
<th>video-AP</th>
<th>brush</th>
<th>hair</th>
<th>catch</th>
<th>clap</th>
<th>climb stairs</th>
<th>golf</th>
<th>jump</th>
<th>kick ball</th>
<th>pick</th>
<th>pour</th>
<th>pullup</th>
<th>push</th>
<th>run</th>
</tr>
</thead>
<tbody>
<tr>
<td>RGB w/o</td>
<td>79.95</td>
<td>30.6</td>
<td>65.79</td>
<td>70.51</td>
<td>92.78</td>
<td>8.15</td>
<td>38.03</td>
<td>75.02</td>
<td>95.72</td>
<td>100</td>
<td>87.81</td>
<td>35.72</td>
<td></td>
</tr>
<tr>
<td>Flow w/o</td>
<td>87.89</td>
<td>47.2</td>
<td>86.08</td>
<td>61.06</td>
<td>100</td>
<td>59.22</td>
<td>27.57</td>
<td>75.35</td>
<td>88.29</td>
<td>98.13</td>
<td>82.14</td>
<td>45.52</td>
<td></td>
</tr>
<tr>
<td>Fuse w/o</td>
<td>92.83</td>
<td>50.75</td>
<td>88.86</td>
<td>71.83</td>
<td>100</td>
<td>41.79</td>
<td>40.25</td>
<td>81.41</td>
<td>93.94</td>
<td>99.63</td>
<td>90.85</td>
<td>55.25</td>
<td></td>
</tr>
<tr>
<td>RGB with TUN</td>
<td>89.1</td>
<td>24.48</td>
<td>86.84</td>
<td>70.37</td>
<td>94.1</td>
<td>5.6</td>
<td>95.45</td>
<td>93.56</td>
<td>100</td>
<td>100</td>
<td>95.33</td>
<td>74.1</td>
<td></td>
</tr>
<tr>
<td>Flow with TUN</td>
<td>99.36</td>
<td>47.1</td>
<td>89.87</td>
<td>58.62</td>
<td>100</td>
<td>59.32</td>
<td>25.08</td>
<td>100</td>
<td>93.16</td>
<td>99.63</td>
<td>98.81</td>
<td>40.35</td>
<td></td>
</tr>
<tr>
<td>Fuse with TUN</td>
<td>100</td>
<td>46.09</td>
<td>97.09</td>
<td>69.93</td>
<td>100</td>
<td>29.33</td>
<td>91.42</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>78.36</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.2: Remaining classes for the ablation study of our proposed framework on J-HMDB21 (split 1).

<table>
<thead>
<tr>
<th>video-AP</th>
<th>run</th>
<th>shoot ball</th>
<th>shoot bow</th>
<th>shoot gun</th>
<th>sit</th>
<th>stand</th>
<th>swing baseball</th>
<th>throw</th>
<th>walk</th>
<th>wave</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>RGB w/o</td>
<td>35.72</td>
<td>34.16</td>
<td>100</td>
<td>72.67</td>
<td>24.72</td>
<td>27.95</td>
<td>68.42</td>
<td>22.5</td>
<td>50.86</td>
<td>69.8</td>
<td>59.58</td>
</tr>
<tr>
<td>Flow w/o</td>
<td>45.52</td>
<td>39.67</td>
<td>96.24</td>
<td>64.33</td>
<td>86.61</td>
<td>87.99</td>
<td>94.17</td>
<td>28.8</td>
<td>77.45</td>
<td>18.43</td>
<td>69.15</td>
</tr>
<tr>
<td>Fuse w/o</td>
<td>55.25</td>
<td>46.21</td>
<td>100</td>
<td>71.35</td>
<td>83.7</td>
<td>94.77</td>
<td>31.81</td>
<td>77.64</td>
<td>48.99</td>
<td>73.15</td>
<td></td>
</tr>
<tr>
<td>RGB with TUN</td>
<td>74.1</td>
<td>29.65</td>
<td>100</td>
<td>71.13</td>
<td>28.57</td>
<td>28.5</td>
<td>79.15</td>
<td>22.43</td>
<td>39.07</td>
<td>63.64</td>
<td>66.15</td>
</tr>
<tr>
<td>Flow with TUN</td>
<td>40.35</td>
<td>46.26</td>
<td>94.62</td>
<td>68.17</td>
<td>97.22</td>
<td>88.27</td>
<td>92.39</td>
<td>28.69</td>
<td>85.16</td>
<td>11.94</td>
<td>72.57</td>
</tr>
<tr>
<td>Fuse with TUN</td>
<td>78.36</td>
<td>40.72</td>
<td>100</td>
<td>77.1</td>
<td>54.49</td>
<td>75.91</td>
<td>93.79</td>
<td>29.35</td>
<td>89.27</td>
<td>45.88</td>
<td>77.08</td>
</tr>
</tbody>
</table>

Video-level Generic Tubelet Proposal Generation

Given the 2-D region proposals, the natural next step would be to link them into tubelets. We build our tubelets in an incremental fashion by linking existing tubelets to 2-D region proposals in the subsequent frame. Since there are multiple region proposals in each frame, it is important to identify which region proposals should be linked given a tubelet proposal from previous time steps. The goal is to ensure that a tubelet does not shift between different people over time.

In this work, we utilize a simple yet effective greedy linking algorithm to generate tubelet proposals. Firstly, the objectness score $o_i^t$ of a 2-D region proposal $i$ in frame $t$ is defined as $o_i^t = \max_{c \in C}(s_i^t(c))$, where $s_i^t(c)$ denotes the class-dependent action score of region $i$ in frame $t$, for class $c \in C$, the set of action classes in the data set. With this per-region objectness score, we describe the linking algorithm as follows. Consider two consecutive frames at time $t-1$ and $t$, and assume $R_i^t$ is spatial location of the $i^{th}$ region proposal in frame $t$. The linking score $\ell(R_i^t, R_j^{t-1})$ between two regions $R_i^t$ and $R_j^{t-1}$ is defined to be

$$\ell(R_i^t, R_j^{t-1}) = o_i^t + o_j^{t-1} + \bigcap_{c \in C}(R_i^t, R_j^{t-1})$$ (4.1)

where $\bigcap$ denotes the intersection over union of the two regions. At time-step $t$, for each tubelet proposal, linking scores are computed between the region in that tubelet at frame $t-1$ and all possible proposals in the frame $t$. The proposal with largest $\ell(\cdot, \cdot)$ is linked to the tubelet. For efficiency, we greedily prune to the top-$K$ ($K = 10$ in experiments) tubelet proposals at each time-step $t$, starting from time-step 2, based on the tubelet score $\tau_i^t$ for
each tubelet $m$:

$$\tau_t^m = \sum_{k=2}^t \ell(R_k^{i(m,k)}, R_{k-1}^{i(m,k-1)})$$  \hfill (4.2)

where $i(m, k)$ indexes the regions $R_k^{i(m,k)}$ for tubelet $m$ at each time step $k$. A final video-level non-maximum suppression is run to produce a set of class-independent tubelet proposals for the video. Algorithm 1 summarizes this process.

**Input:** Proposals $R_i^t$ in each frame $t$

for $t = 1 \ldots T$ do

if $t == 1$ then

for each $R_i^t$ do

| init new tubelet from $R_i^t$;

if $t > 1$ then

for each tubelet $m$ in frame $t - 1$ do

| Link $i$ with best $R_j^t$ by Eq. 4.1;
| keep top-K tubelets as scored by Eq. 4.2;

video-level NMS;

**Output:** tubelet proposals

**Algorithm 1:** Tubelet Generation Algorithm

### 4.3.2 Temporal Understanding Network

Videos contain rich information in both spatial and temporal dimensions. Simple temporal approaches, such as averaging frame-level spatial features, are not sufficient to represent temporal information. Therefore, we input the video-level tubelet proposals to a temporal understanding network (TUN) to perform more sophisticated temporal analysis. Various temporal network structures can be easily utilized here, e.g., temporal pooling, LSTM, C3D, etc. In this work, we build an LSTM-based structure to classify the generic tubelets.

In order to utilize all available information to facilitate action classification, we fuse the frame-level action score $s_i^{(m,t)}(c)$ from the TPN with the one $\hat{s}_i^{(m,t)}(c)$ from the LSTM for the same tube $m$ to obtain video-level information. Therefore, the class-specific action score $S_{\text{tube}}^m(c)$ for a tubelet $m$ is calculated by averaging the scores of TPN and LSTM from all frames in the same tubelet:

$$S_{\text{tube}}^m(c) = \frac{\lambda_1}{T} \sum_{t=1}^T s_i^{(m,t)}(c) + \frac{1 - \lambda_1}{T} \sum_{t=1}^T \hat{s}_i^{(m,t)}(c)$$ \hfill (4.3)

The parameter $\lambda_1$ is set to 2/3 in this work, empirically.

Since the LSTM and the TPN share the convolution layers and fully connected layers (as shown in Fig. 4.2), the region-specific feature, namely, the fc7 feature vectors of selected region $R_i^{(m,t)}$ of tubelet $m$ in TPN is given into the LSTM as input. Region-specific feature extraction is proved to be effective for classification in Sec. 4.4.2.
4.3.3 Two-stream Network Structure

We further present a generalization of our approach to a two-stream structure that utilizes both RGB frames and optical flow input. Optical flow often proves to be effective in action classification since it uses motion information, complementary to the appearance cues in the RGB stream.

The main advantage for optical flow is in more accurate action classification\(^1\). Hence, we develop a method for matching optical flow-stream regions with their counterparts obtained from the RGB stream. The final fused score for a region will be the combination of both the RGB stream score and the optical flow stream score. In this work, the optical flow images are calculated following the method in [13]. There are two fusions in the network: the first one is the fusion of the 2-D region proposals in the TPN, and the second one is the fusion of the two LSTM outputs for the TUN.

**TPN Fusion:** Although both streams share the same RPN (shown in Fig. 4.2), the regressor and classifier of each stream is learned independently. Therefore, it is important to decide which flow-stream region proposal \(\bar{R}_j^{(i,t)}\) should be corresponding to the rgb-stream proposal \(R^i_t\) in frame \(t\). We calculate the IoU of each flow-stream proposal \(\bar{R}_k^i\) (\(k \in \{1, \ldots, K\}\), the set of proposals in flow stream) with a given rgb-stream proposal \(R^i_t\) in the same frame. The correspondence is established based on Eq. 4.4:

\[
j(i, t) = \arg\max_{k \in \{1, \ldots, K\}} \bigcap (R^i_t, \bar{R}_k^i)
\]  

(4.4)

Now, each rgb-stream proposal has a corresponding flow-stream proposal. The class-specific action scores are updated by averaging over the rgb-stream proposal \(i\) action score \(s^i_t(c)\) with the corresponding flow-stream proposal action score \(\bar{s}^{j(i,t)}_t(c)\) in frame \(t\):

\[
s^i_t(c) = \lambda_2 \times s^i_t(c) + (1 - \lambda_2) \times \bar{s}^{j(i,t)}_t(c)
\]

(4.5)

Since flow stream is more effective in classification, following [29], \(\lambda_2\) is set to 1/3. Once a tubelet proposal is generated in the rgb stream, a corresponding flow-stream tubelet proposal is also generated based on the per-proposal correspondence in Eq. 4.4.

**TUN Fusion:** A LSTM-based TUN is performed on each stream separately, and the class-specific action scores (output of LSTM) are averaged over two streams using Eq. 4.5. In conclusion, the final class-specific action score of a tube contains four components: TPN and TUN action scores in both the rgb and optical flow streams. A thorough ablation study is given in Sec. 4.4 examining these design choices.

\(^1\)Empirically, we noticed that a RPN trained on flow images is not effective. The simple reason is that optical flow only focuses on moving regions, and people performing actions are occasionally stationary. Therefore, in our work the RPN is always based on the RGB stream, and used in both the RGB and flow streams.
Table 4.3: Comparison to state-of-the-art methods on three datasets. Methods with * are considered as comparable basic models with the same VGG network setup. The results on the J-HMDB21 data set are averages over all three splits. The experiments on the UCF-101 data set are performed without temporal localization. Only [101] reports mAP both with and without temporal localization, therefore we only compare with [101] (without temporal localization) on this dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>UCF-Sports</th>
<th>J-HMDB21</th>
<th>UCF-101</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action Tubes [38]</td>
<td>-</td>
<td>75.8</td>
<td>-</td>
</tr>
<tr>
<td>STMH [148]</td>
<td>-</td>
<td>90.5</td>
<td>-</td>
</tr>
<tr>
<td>STAT [116]*</td>
<td>-</td>
<td>-</td>
<td>72.65</td>
</tr>
<tr>
<td>ST [129]*</td>
<td>-</td>
<td>-</td>
<td>69.36</td>
</tr>
<tr>
<td>TS R-CNN[101]*</td>
<td>94.8</td>
<td>94.8</td>
<td>-</td>
</tr>
<tr>
<td>MR-TS</td>
<td>94.8</td>
<td>94.7</td>
<td>-</td>
</tr>
<tr>
<td>CNN[101]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Our TPN*</td>
<td>95.8</td>
<td>95.5</td>
<td>74.68</td>
</tr>
<tr>
<td>Our TPN with LSTM</td>
<td><strong>96.0</strong></td>
<td><strong>95.7</strong></td>
<td><strong>79.84</strong></td>
</tr>
</tbody>
</table>

Table 4.4: Classification accuracy on J-HMDB21 dataset (averaged over all three splits).

<table>
<thead>
<tr>
<th>Method</th>
<th>145</th>
<th>148</th>
<th>38</th>
<th>116</th>
<th>101</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy(%)</td>
<td>56.6</td>
<td>61</td>
<td>62.5</td>
<td>70.0</td>
<td>71.08</td>
<td><strong>72.21</strong></td>
</tr>
</tbody>
</table>

4.3.4 Training

In our experiments, the TPN and TUN are trained independently in a two-stage fashion, which introduces practical benefits in terms of reduced memory consumption, the ability to use (memory intensive) state-of-the-art network architectures, and initializing from pre-trained models.

The implementation uses the Caffe toolbox. The TPN is based on a VGG-16 pre-trained on ImageNet. Note that the memory used to train the TPN alone is almost 11 GB (a 12 GB Titan X GPU is used in experiments). We fine-tune the TPN on each dataset respectively. The learning rate is set to $10^{-3}$ and decreases by 0.1 every 10k iterations. Training is stopped after 70k iterations. Softmax loss for classification and SmoothL1 loss for regression are used. RoI-fc7 features from VGG are used as the input of the LSTM, which has 3000 hidden nodes. The LSTM learning rate is set to $10^{-4}$, decreased by 0.1 per 10k iterations. Momentum and weight decay are set to 0.9 and $5e^{-4}$. Softmax loss is used in training the LSTM for classifying tubelets.

Based on this training setting, the whole framework is fully differentiable under the per-frame SmoothL1 loss for regression. Therefore, the TPN and TUN could be potentially
jointly optimized, trained in an end-to-end fashion. However, as mentioned above, memory
considerations preclude doing so at this time.

### 4.4 Experiments

We evaluate our approach on three standard datasets: UCF Sports, J-HMDB21, and UCF-101. On these data sets, we compare against other methods and show substantial improvement over state-of-the-art approaches.

**Datasets.** The UCF-Sports dataset consists of 150 videos with 10 different actions. We use the same evaluation protocol as [74]. The J-HMDB21 dataset contains about 900 videos of 21 different actions. The human figure in each frame has body joint annotations, which are used to generate the bounding box for each person. A subset of 24 classes out of 101 (‘split 1’) comes with spatio-temporal localisation annotation from [129]. We use these for measuring action localization, following the same protocol as [129, 116, 101].

**Evaluation Metric.** We use video mean average precision (mAP) as the main evaluation metric. Similar to object detection, an action tube is considered as a true positive if 1) the IoU of the action tube and the ground-truth tube is above a threshold $\delta$, and 2) the action in the tube is correctly classified. This is the same evaluation metric as in previous action localization works such as [38, 116, 101, 129].

#### 4.4.1 Ablation Study

We conducted an ablation study to assess the utility of components of our approach. We compare the following variants on the first split of the J-HMDB21 dataset. Results are shown in Table 4.1.

**RGB-stream model without TUN.** This simple baseline averages the class-specific action scores from the fast R-CNN.

**RGB-stream model with TUN.** Generic tubelet proposals from the TPN are fed to an LSTM to improve the classification in each video.

**Flow-stream model without TUN.** This model is similar to the RGB-stream model without TUN, with the input being the optical flow images.

**Flow-stream model with TUN.** This model is similar to the RGB-stream model with TUN, with the input being the optical flow images.

**Two-stream fusion model without TUN.** This model fuses the 2-D region proposals to generate more accurate generic class-independent tubelet proposals. The fused score is used to classify these tubes.

**Two-stream fusion model with TUN.** This is our full proposed model.

It is clear from the ablation study that each part of our model improves the tubelet generation performance. Without using a TUN, the fusion model alone outperforms other comparable methods. The three methods that are considered as comparable share the same VGG16.
network structure, and all use faster R-CNN to generate 2-D region proposals/detections. In Table 4.1, the three comparable methods are marked with *. Furthermore, adding LSTM as an example of the TUN component increases the mAP, which shows that the generated generic tubelet proposals preserve the information for the action to be correctly classified.

4.4.2 Comparison with State-of-the-art Methods

We compared with other methods on standard action localization datasets: UCF-101, J-HMDB21, and UCF-Sports.

Comparison on the J-HMDB21 Dataset

The comparison with other methods on the J-HMDB21 dataset is presented in Table 4.3. The mAP is averaged over all three splits of the J-HMDB-21 dataset. Our TPN model without TUN already consistently outperforms other comparable methods, regardless of the overlap threshold. This comparison indicates that our proposed TPN can generate more accurate tubelet proposals than other methods. Furthermore, utilizing a LSTM in the TUN, we achieve state-of-the-art results consistently over all thresholds. Some example action tubes are shown in Fig. 4.4. Note that we only visualize the action tubes which have action score higher than 0.2 for visual clarity.

In the MR-TS R-CNN model [101], each of the two streams in the framework is expanded into a multi-region structure, therefore the entire network needs to be retrained. However, in our case, since our TPN and LSTM are combined in a plug-in style, the model trained for TPN can still be used, and the features are shared between the two parts to further reduce the computation.
Figure 4.5: Example action tubes from the UCF-Sports dataset. Each video is represented by four frames. The red bounding boxes are the detected regions, and the blue ones are the ground-truth regions. As can be seen, our generated tubes are very accurate. Also, our framework can handle multiple-person scenarios (upper left video), fast moving actions like Swing, and actions with large intra-class shape variation like Lifting. Left bottom shows a failure case. The ground-truth of this video only focused on the middle person with label Kicking. Our algorithm generates two high-scoring tubes for this video, one overlaps with the ground-truth region, another is the left person with label Walk, which we think is also reasonable.

We also report action classification results in Table 4.4. We pick the action class with the highest action score among all the tubelets for each video as the video-level classification result. Our proposed model achieves outperforms other tube-based methods on classification as well.

Comparison on the UCF-Sports Dataset

The localization results on the UCF-Sports dataset are reported in Table 4.3. Similar to the J-HMDB21 dataset, we consistently outperform other methods under different thresholds. Some example action tubes are shown in Fig. 4.5. Note that again we only visualize the action tubes which have action score higher than 0.2.

Comparison on the UCF-101 Dataset

The UCF-101 dataset is an untrimmed dataset. The action only occurs in some frames in each video. Therefore, it is also important to perform temporal localization. However, recent works [116, 101] either use simple heuristics or a sliding window approach to facilitate temporal localization. A full treatment of temporal localization would be an interesting extension to our work. To illustrate the potential of our method, we perform action localization over the entire untrimmed video. In this way, the precision and recall of the action tubes will be penalized because there are no ground-truth bounding boxes in some frames. Only [101] reports the mAP on UCF-101 without temporal localization, therefore, we only compare to [101] on this dataset. With temporal localization, the mAP of our proposed method could be further improved. Some example (> 0.2 score) action tubes are shown in Fig. 4.6.
Figure 4.6: Example action tubes from the UCF-101 dataset. Each video is represented by four frames. The red bounding boxes are the detected regions, and the blue ones are the ground-truth regions. The first two rows show successful cases, while the last row shows failure cases. Our framework generates accurate action tubes for complicated actions. However, the bottom-left case shows that our framework tends to generate only one action tube if multiple action instances are close to each other. The bottom-right and top-left cases show performance in complicated scenes such as basketball games (top-left correct, bottom-right inaccurate localization).

In summary, the experiments show that our generic tubelet proposal paradigm consistently outperforms the class-specific action tube generation paradigm. Compared with previous methods [101, 116], our framework does not include any class-dependent hyper parameters, and is compact and easy to generalize.

4.5 Summary

In this chapter, a novel generic class-independent Tube Proposal Network is proposed. This network can generate a set of generic class-independent tubelet proposals to localize the action regions in the video sequence precisely. These generic tubelet proposals can be fed into various temporal understanding networks, to perform more sophisticated temporal understanding. The proposed framework is utilized in action localization tasks, which requires the system to both localize the action region precisely, and classify the action class correctly. Our method consistently outperforms other methods across all three datasets. In our experiments, we observe that a limitation lies in handling the cases where multiple action instances are close to each other. Also, how to utilize the TPN as a backbone structure to facilitate temporal localization in untrimmed videos is an interesting direction. Further, these generic tubelets could be used in more complicated tasks such as interpreting scenes with multiple, interacting humans. These directions will be addressed in future work.
Chapter 5

Hierarchical Relational Networks for Group Activity Recognition and Retrieval

Modeling structured relationships between people in a scene is an important step toward visual understanding. In this chapter we present a Hierarchical Relational Network that computes relational representations of people, given graph structures describing potential interactions. Each relational layer is fed individual person representations and a potential relationship graph. Relational representations of each person are created based on their connections in this particular graph. We demonstrate the efficacy of this model by applying it in both supervised and unsupervised learning paradigms. First, given a video sequence of people doing a collective activity, the relational scene representation is utilized for multi-person activity recognition. Second, we propose a Relational Autoencoder model for unsupervised learning of features for action and scene retrieval. Finally, a Denoising Autoencoder variant is presented to infer missing people in the scene from their context. Empirical results demonstrate that this approach learns relational feature representations that can effectively discriminate person and group activity classes.

5.1 Overview

Human activity recognition is a challenging computer vision problem and has received a lot of attention from the research community. Challenges include factors such as the variability within action classes, background clutter, and similarity between different action classes. Group activity recognition arises in the context of multi-person scenes, including in video surveillance, sports analytics, and video search and retrieval. A particular challenge of group activity recognition is the fact that inferring labels for a scene requires contextual reasoning about the people in the scene and their relations. In this chapter we develop a novel deep network layer for learning representations for capturing these relations.
Figure 5.1: A single relational layer. The layer can process an arbitrary sized set of people from a scene, and produces new representations for these people that capture their relationships. The input to the layer is a set of \( K \) people and a graph \( G^\ell \) encoding their relations. In the relational layer, a shared neural network \( (F^\ell) \) maps each pair of person representations to a new representation that also encodes relationships between them. These are aggregated over all edges emanating from a person node via summation. This process results in a new, relational representation for each of the \( K \) people. By stacking multiple relational layers, each with its own relationship graph \( G^\ell \), we can encode hierarchical relationships for each person and learn a scene representation suitable for group activity recognition or retrieval.

Fig. 5.1 provides a schematic of our relational layer and Fig. 5.2 highlights the processing of a single person inside the layer. Initially, each person in a scene can be represented by a feature, \( e^g \), derived from a standard CNN. We amalgamate these individual representations via stacking multiple relational layers – deep network layers that combine information from a set of (neighbouring) person representations. These layers are utilized in a hierarchy, refining representations for each individual person based on successive integration of information from other people present in the scene.

Recent deep learning approaches [50, 106, 123] for group activity recognition use a 2-stage processing pipeline where first each person is represented using a large feature vector (e.g., fc7 features). Then, the person representations are pooled together to construct the final features for the scene. The typical scene pooling is max / average / attentional pooling over people, which reduces dimensionality, but loses information. First, all spatial and relational information is dropped. Second, features about individual people, which actually define actions, are lost. Finally, although such a scene representation is optimized for group activity recognition, it cannot be used for analysis tasks based on individual actions.

Our models utilize a similar 2-stage processing framework, but work on solving these drawbacks in an efficient and effective manner. Given initial feature representations for each person and a relationship graph, we present a relational layer that jointly computes
Figure 5.2: Relational unit for processing one person inside a relational layer. The feature representation for a person (red) is combined with each of its neighbours’. Resultant vectors are summed to create a new feature representation for the person (dark red).

...a compact representation for each person that encodes inter-person relations. By stacking multiple relational layers, this hierarchical relational network learns a compact relational representation per person.

Our contributions can be summarized as follows:

- A relational layer that jointly infers relational representations for each person based on a relationship graph. The layer can operate on a variable sized set of people in a scene. Given features for $K$ people, the layer maps the given $K$ feature vectors to $K$ new ones, capturing relations and preserving correspondence between each feature vector and each person.

- A relational scene representation. By stacking multiple relational layers, each with its own relationship graph, we build a scene representation encoding hierarchical relationship representations. This representation is suitable for scenes of multiple related objects, such as in multi-person activity recognition.

- A novel autoencoder architecture that stacks multiple relational layers to jointly encode/decode each person’s features based on relationship graphs. In unsupervised domains where no action labels are available, such representations can be used for scene retrieval based on nearest neighbour matching. A denoising autoencoder variant is also presented that infers missing people.

- Demonstrating the utility of these modules for (supervised) group activity recognition and (unsupervised) action/scene retrieval.
5.2 Related Work

We develop methods for multi-person activity recognition and retrieval by learning relational features. Below, we review related work in these areas.

**Multi-person activity recognition:** Recent deep learning approaches to multi-person activity recognition include Ibrahim et al. [50], which presents a 2-stage deep model. Person actions are modeled using a long short-term memory (LSTM) temporal layer. Scene dynamics are captured by adding a max-pooling layer which is fed to a higher-level LSTM. Ramanathan et al. [106] formulate an attention model to highlight key players in a scene, resulting in a weighted summation of person feature representations. Bagautdinov et al. [7] propose a joint model of action localization and group activity recognition. A multi-person object detection approach finds people and extracts their feature representations, which are linked based on Euclidean distance and fed to temporal recurrent network. Shu et al. [123] extend this pipeline with an energy layer and confidence measure to consider reliability and numerical stability of the inference. Our work follows these 2-stage processing pipelines, but introduces a new relational layer that can learn compact relational representations for each person.

**Imageretrieval:** Content-based retrieval for structured scenes is an active research area [115, 102, 132, 69]. Siddiquie et al. [125] extract multi-attributes and their correlations from a text query. Lan et al. [77] introduce queries that specify the objects that should be present in the scene, and their spatial relations (e.g., “car on the road”). Kim et al. [64] retrieve video clips that emphasize the progression of the text query. Johnson et al. [57] consider scene graph queries (objects and relationships). Xu et al. [149] generate scene graphs via a message passing neural network. In the realm of multi-person activity recognition, hard-coded representations of spatial relations have been developed previously [18, 75]. We show how our relational layers can be used in structured scene image retrieval, by matching frames of similar visual structure of people and their actions.

**Relational networks:** Recent work with deep networks includes capturing object relationships through aggregating with every-pair-relation models. Santoro et al. [117] introduce a relational network module that infers relationships between image objects. A multi-layer perceptron (MLP) learns the relationship of two objects, the scene is represented as summation of all object pairs. In a similar manner, Guttenberg et al. [42] use an MLP to learn a permutation-equivariant representation of a group of objects based on the relationship of every pair of objects. Inspired by these simple relation networks, we introduce our hierarchical relational network to build a compact relational scene representation, while preserving the correspondence between the feature representation and each person.

**Graph Neural Networks:** Our work can be viewed as a variant of a Graph Neural Network (GNN) [31, 86, 82, 23, 67]. GNNs generalize Convolutional Neural Networks to learn a function of features on a graph (where graph nodes are associated with initial feature
representation), and builds a node-level output (the updated nodes representations based on the graph structure). Qi et al. [105] propose a GNN model for semantic segmentation on 3D point clouds. Each node in their graph represents a set of 3D points and its initial feature representation is from a CNN applied on 2D images. The graph nodes are connected based on nearest neighbor search on the point cloud. The representation of a node is updated based on messages from its neighbors and RNN per node is used to capture the temporal dynamics of running this model for $T$ timesteps. In our work, each node represents a person and the graph structure is based on Euclidian distances between people 2D positions. CNN extracts feature representation from a person’s bounding box for the node’s initial representation. We use a shared MLP that learns edges representation, and a person’s feature representation is the result of aggregating edges representation from each neighboring person in the graph. Ying et al. [153] introduce a hierarchical graph representation by learning to cluster nodes at each layer. In our work, we manually build hierarchical graphs in terms of cliques for Volleyball games to administrate the importance of hierarchical graph representations. Liao et al. [83] tackles the scalability issues of GNNs by introducing a model that partitions the graph to subgraphs and the model alternates between locally propagating messages within each sub-graph and globally propagating messages between the sub-graphs.

5.3 Proposed Approach

This chapter introduces a Hierarchical Relational Network that builds a compact relational representation per person. Recent approaches [50, 106, 42] represent people in a scene then directly (max/average) pool all the representations into a single scene representation. This final representation has some drawbacks such as dropping relationships between people and destroying the individual person features. We tackle these challenges through a relational layer that jointly creates $K$ person representations for the $K$ people in a scene. By stacking multiple relational layers, we compactly encode hierarchical relationship representations. In the next subsections, we elaborate on the details of the Relational Network, then show its applications in supervised classification and unsupervised retrieval settings.

5.3.1 Hierarchical Relational Network

Our relational network for multi-person activity recognition processes one video frame at a time. An input video frame has $K$ initial person feature vectors (where $K$ is the number of detected people in the scene and each person is represented with some DCNN features) associated with multiple potential relationship graphs (e.g., based on spatial Euclidean distance thresholds). A single relational layer is fed with both $K$ feature vectors and a relationship graph, and maps them to $K$ new relational representations.

The building block for our model is a relational unit that processes an individual person in the scene. Each person’s feature vector is mapped to a new representation by aggregating
Given $K$ people and their initial feature vectors, these vectors are fed to 3 stacked relational layers (of output sizes per person: 512, 256, 128). Each relational layer is associated with a graph $G^\ell$ (disjoint cliques in this example: layer 1 has 4 cliques, each of size 3; layer 3 is a complete graph). The shared MLP $F^\ell$ of each layer computes the representation of 2 neighbouring players. Pooling of the output $K$ feature vectors is used for group activity classification.

Information from each neighbouring person in the relationship graph. This is accomplished via a network that processes the person combined with each neighbour, followed by aggregation. This relational unit is depicted Fig. 5.2.

Within one relational layer, every person in the scene is processed using this unit. This results in new feature representations for each person in the scene, capturing their individual features as well as those from his/her neighbours.

By stacking multiple layers, each with its own graph and relational unit parameters, we learn hierarchical relationship representations for the people. Pooling of the final person representations is used to construct the scene representation. An overview of our relational network for multi-person activity recognition in a single frame is shown in Fig. 5.3.

Formally, given a video frame, the $i^{th}$ person representation $P^\ell_i$ in the $\ell^{th}$ relational layer is computed as follows:

$$P^0_i = CNN(I_i)$$

$$P^\ell_i = \sum_{j \in \mathcal{E}^\ell_i} F^\ell(P_{i}^{\ell-1} \oplus P_j^{\ell-1}; \theta^\ell)$$

where $P^0_i$ is the initial $i^{th}$ person representation derived from a CNN on cropped image $I_i$, $\mathcal{E}^\ell_i$ is the set of relationship edges from the $i^{th}$ person in the graph $G^\ell$ used for the $\ell^{th}$ layer,
and \( \oplus \) is the concatenation operator. \( P^\ell_i \in \mathbb{R}^{N_\ell} \) where \( N_\ell \) is the output size per-person for the \( \ell \)th layer.

The function \( F^\ell \) is a shared MLP for the \( \ell \)th network layer with parameters \( \theta^\ell \) (end-to-end differentiable model). The MLP has input size \( 2N_{\ell-1} \) and output size \( N_\ell \). Given two concatenated vectors, \( F^\ell \) maps them to a new vector capturing the given pair’s content and relationship.

The relational layer feeds each edge in \( G^\ell \) through its own shared MLP to compute the \( K \) new representations. Equation 5.2 computes a relationship representation between the \( i \)th person and his/her neighbours. This network structure and the use of layer-wise shared parameters results in relationship representations per layer – treating each pair of people within one network layer equivalently. This results in efficient parameter reuse while letting the representation be driven by the graph structure at each layer. Importantly, this representation can also be used with any number of people \( K \), including situations where \( K \) can vary per time step due to occlusions or false positive detections.

By stacking multiple compressive relational layers, each with its own graph, we can construct reduced dimension person features from one layer to another until a desired compact relational representation has been formed. The final scene representation \( S \) is the pooling of person representations from the last relational layer output and defined as:

\[
S = P^L_1 \oplus P^L_2 \oplus \ldots \oplus P^L_k
\]

where \( P^L_i \) is the \( i \)th person output representation of last relational layer \( L \) and \( \oplus \) is a pooling operator (such as vector concatenation or element-wise max pooling).

### 5.3.2 Supervised Learning: Group Activities

The activity of a group of people is a function of the persons’ actions. We can utilize our model to represent each scene and learn its parameters in a supervised fashion. We utilize an Imagenet pre-trained VGG network [127] to represent each single person bounding box. The whole network is fine-tuned using action-labeled bounding boxes. Once trained, each person bounding box can be represented with the last layer in VGG19 (4096-d fc7 features).

Given the bounding boxes of the people in the scene in a video sequence, we recognize the overall multi-person activity. Each bounding box at the \( t \)th frame is modeled and represented with an initial feature vector as explained above and fed to the relational network. The relational layer jointly maps the representations to ones that encode the relationship representation of a person based on connections to other people. To capture the temporal dynamics of the video scene, the output of the final relational layer is pooled to the \( t \)th scene representation \( S_t \) and fed to an LSTM layer with a softmax output for group activity classification. Fig. 5.3 illustrates this model for a single frame.
5.3.3 Unsupervised Learning: Action Retrieval

Detailed annotation of individual person bounding boxes in video is a time-consuming process [41]. As an alternative, one could utilize unsupervised autoencoder mechanisms to learn feature representations for people in scenes. These representations could potentially be general-purpose: allowing comparison of person features based on relations and context for single-person action retrieval, and retrieval of scenes of similarly structured sets of actions.

Recent efforts in object recognition [99, 28] and temporal sequence learning [107, 79] aimed to learn effective feature representations in unsupervised encoding frameworks. In a similar vein, we propose unsupervised autoencoders that learn relational representations for all people in the scene.

Our relational layer is well-suited to this task since it: 1) encodes person relationships, 2) preserves action features for individual people, and 3) has compact size, efficient for retrieval. In other words, our scene representation is both efficient (compact size) and effective (relationship-based). Further, the model has the same parameter count as a simple autoencoder of a single person, as each layer has a shared network.

For the encoder, given \( K \) feature vectors for the people in the scene, we stack multiple relational layers of decreasing size that encode features to a final compact representation. The decoder is the inverse of these layers. That is, we again stack multiple relational layers of increasing size that decode a compressed feature vector to its original CNN representation. Each relational layer jointly maps a person representation from a given input size to a required output size considering graph connections. An Euclidean loss is computed between the initial \( K \) feature vectors and the corresponding decoded ones. An overview of the autoencoder model is shown in Fig. 5.4.

The reconstruction loss \( \mathcal{L} \) of the input scene and its reconstructed one is given by:

\[
\mathcal{L}(S_{\text{cnn}}, S'_{\text{cnn}}) = \sum_{i=1}^{K} \|P_{0i} - P_{Li}\|^2
\]

where \( P_{0i} \) and \( P_{Li} \) are similar to Eq. 5.2 (but for a singel frame), \( S_{\text{cnn}} \) is the concatenation of the \( K \) initial feature vectors \( P_{0i} \), and \( S'_{\text{cnn}} \) is the reconstructed output of our network extracted from the last layer \( L \). This novel autoencoder preserves features for individual people, so can be used for both scene and action retrieval.

**Denoising Relational Autoencoder:** What if some persons are missing in the scene (e.g., due to person detector failures, fast camera movement, or low image quality)? Denoising the input \( K \) feature vectors by dropping the whole vector for some of the persons allows our relational autoencoder to construct person representations from incomplete scenes. That is, our model infers the missing people from their context. To implement this, the input layer is followed by a dropout layer that drops a complete vector (not just subset of features) with probability \( P \) [109].
Retrieval: Given a single frame of $K$ people, suppose we wish to search a video database for a matching frame with similar action structure. Note, the purpose is not retrieving a scene with the same overall activity, but a similar structured scene of actions. The pooled representation style, such as in [50], fits with group activity classification, but not with scene retrieval based on the matching of the actual actions due to losing person features for sake of a global scene representation. On the contrary, our representation for the scene preserves the individual person actions explicitly in a compact sized feature.

For the retrieval mechanism, we use a simple K-Nearest-Neighbour technique with a brute-force algorithm for comparison. To avoid comparison with each possible permutation, people are ordered based on the top corner ($x$, $y$) of a person’s bounding box (on $x$ first, and on $y$ if tied). Euclidean distance is used to compare feature vectors.

5.4 Experiments

To demonstrate the power of our relational network, we evaluate it for two tasks: group activity recognition and action scene retrieval. The results are evaluated on the recent Volleyball Dataset [50]. The dataset consists of 4830 short clips gathered from 55 volleyball games, with 3493 training clips and 1337 for testing. Each clip is classified to one of 8 scene activity labels. Only the middle frame of each clip is fully annotated with the players’ bounding boxes and their action labels (out of 9 actions). Clips of 10 frames (centered
around the annotated middle frame) are used for the activity recognition task and the middle frame is used for the action scene retrieval task.

Our relational layer accepts free-form graph relationships. For volleyball, one suitable style is graphs of disjoint cliques based on person spatial locations. For example, in volleyball games there might be 3 potential graphs: I) All players are in 1 clique (1C), represents all pairwise relationships; II) each team can be a clique (2C); III) each team can be composed of 2 cliques, a total of 4 cliques (4C). We base our experiments on these clique-based groupings.

For the final scene pooling, instead of just max-pooling all persons, we use a slight variant [51] that reduces confusions between actions of the two team. Specifically, we max-pool each team individually, then concatenate the two representations. This is the default pooling strategy unless otherwise mentioned. In addition, due to the final person features’ compact size, we could also do all-persons concatenation pooling. The concatenation pooling is neither effective nor efficient in other recent approaches [50] [123] due to the large dimensionality of the final person representation.

5.4.1 Group Activity Recognition

We refer to our activity recognition model as RCRG: Relational Compact Representation for Group activity recognition. RCRG is a 2-stage processing model and its input is clips of 10 timesteps, centered around the middle annotated frame. In the first stage, we fine-tune an ImageNet-pretrained VGG19 network using the annotated person bounding boxes (not a temporal model). This trained network is then used to represent each person bounding box using the penultimate network layer (fc7, 4096-d features). The person action recognition accuracy from the VGG19 model is 81%. In the second stage, K person representations are fed to our hierarchical relational network (associated with a relationship graph per layer) as in Fig. 5.3.

Baselines:

We perform ablation studies with the following non-temporal (single frame) variants of our model to help us understand the performance of the model. The default pooling strategy is max-pooling unless -conc postfix is used to indicate concatenation pooling.

B1) B1-NoRelations: In the first stage, the ImageNet-pretrained VGG19 network is fine-tuned and a person is represented with fc7, 4096-d features. In the second stage, each person is connected to a shared dense layer of 128 features, then the person representations (each of length 128 features) are pooled, then fed to a softmax layer for group activity classification. This variant compresses person representations and represents the scene without inferring relationship representations.
Table 5.1: Volleyball Dataset: Left table is for versions of our model using single frame (last row shows state-of-the-art using a single frame). Right table is for 10-timesteps input clips performance of our best models versus state-of-the-art.

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>B1-NoRelations</td>
<td>85.1</td>
</tr>
<tr>
<td>RCRG-1R-1C</td>
<td>86.5</td>
</tr>
<tr>
<td>RCRG-1R-1C-!tuned</td>
<td>75.4</td>
</tr>
<tr>
<td>RCRG-2R-11C</td>
<td>86.1</td>
</tr>
<tr>
<td>RCRG-2R-21C</td>
<td>87.2</td>
</tr>
<tr>
<td>RCRG-3R-421C</td>
<td>86.4</td>
</tr>
<tr>
<td>RCRG-2R-11C-conc</td>
<td>88.3</td>
</tr>
<tr>
<td>RCRG-2R-21C-conc</td>
<td>86.7</td>
</tr>
<tr>
<td>RCRG-3R-421C-conc</td>
<td>87.3</td>
</tr>
<tr>
<td>Bagautdinov et al. [7]-single</td>
<td>83.8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bagautdinov et al. [7]</td>
<td>90.6</td>
</tr>
<tr>
<td>RCRG-2R-11C-conc</td>
<td>89.5</td>
</tr>
<tr>
<td>RCRG-2R-21C</td>
<td>89.4</td>
</tr>
<tr>
<td>Shu et al. [123]</td>
<td>83.3</td>
</tr>
<tr>
<td>Ibrahim et al. [51]</td>
<td>81.9</td>
</tr>
</tbody>
</table>

B2) **RCRG-1R-1C**: Same as previous variant, but the shared dense layer is replaced with a single relational layer (1R), all people in 1 clique (1C), i.e. all-pairs relationships. The layer maps each person from input size 4096 to 128 features jointly considering the given relationships.

B3) **RCRG-1R-1C-!tuned**: Same as previous variant, but ImageNet-pretrained VGG19 without fine-tuning.

B4) **RCRG-2R-11C**: Close to the RCRG-1R-1C variant, but uses 2 relational layers (2R) of sizes 256 and 128. The graphs of these 2 layers are 1 clique (11C) of all people. This variant and the next ones explore stacking layers with different graph structures.

B5) **RCRG-2R-21C**: Same as the previous model, but the first layer has 2 cliques, one per team. The second layer is all-pairs relations (1C). RCRG-2R-21C-conc replaces the max pool strategy with concatenation pooling.

B6) **RCRG-3R-421C**: Close to the previous model, but 3 relational layers (of sizes 512, 256 and 128) with clique sizes of the layers set to (4, 2, 1). The first layer has 4 cliques, with each team divided into 2 cliques. This model is in Fig. 5.3.

**Implementation Details:**

We utilize the available dataset annotations for implementation. We follow Ibrahim et al. [50] to compute 10-frame tracklets of each person across the video sequence [20].

For training all the models and baselines, the same training protocols are followed using a Tesla K40C GPU (12 GB RAM) and Lasagne Framework [26]. Stochastic gradient descent
is used train the model for 200 epochs and initial learning rate $10^{-4}$ with ADAM [66] optimizer, with fixed hyper-parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$. We fine-tune the whole pre-trained VGG19 network [127] using batch-size 64 (small due to memory limits). For the relational model, a batch size of 250 is used. The input layer in our relational model is followed by a 50% dropout layer. Two-layer MLP networks are used of sizes $N_{\ell}$. The first layer uses a linear activation function ($f(x) = x$) and the second uses ReLU non-linearities. Note, the models are end-to-end differentiable, but due to memory limits we implement it in a 2-stage style, similar to recent approaches.

In testing, only one shared person network is loaded and used by the K players to extract their features. The time complexity of a relational layer depends on the summation of the nodes degrees in the layer’s graph. In other words, for each directed edge, the MLP of a layer is evaluated.

To determine graph cliques, we follow a simple approach [51]. People are ordered based on the upper left corner ($x, y$) of their bounding box (on $x$ first, and on $y$ if tied). Cliques are generated by sweeping this ordered list. For example, to divide 12 people to 4 cliques of equal size, each 3 consecutive people are grouped as a clique. More sophisticated grouping (e.g., color/motion clustering) or gating functions [24] would be potential extensions.

**Results:**

Tables 5.1 compare the classification performance of our compact representation with the baselines and the state-of-the-art approaches.

**Discussion:**

Our non-temporal models’ performance is superior to state-of-the-art corresponding models and outperform compact baselines. Note even without temporal information this model is superior to 2 recent temporal models (In right Table 5.1). It seems from the results that stacking 2 layers is enough in this domain: in a volleyball scene inter-person relationships are strong. Max-pooling is effective at a scene level. Likely, this is due to the domain; a few players are the key actors, and max-pooling can keep the right features.

### 5.4.2 Experiments for Action and Scene Retrieval

We evaluate our retrieval model trained using unsupervised learning, termed RAER (Relational AutoEncoder for Retrieval). Our main model is shown in Fig. 5.4. It consists of 4 relational layers (256-128-256-4096 sizes) and it assumes the graph is 2 cliques (one per team) in all layers. We denote this structure by RAER-4L-2222C. This means, each team is compressed jointly, but all people per layer use the same shared relational MLP. Once the network is trained, each person is represented with 128 features from the compressed layer and used for scene and person retrieval.
Performance Measure: We consider two volleyball dataset frames as a correct match if the IoU (intersection over union) of the distributions of actions of the two frames is $\geq 0.5$). For example, if the person actions of frame 1 are 7 people standing and 5 moving, and frame 2 are 4 standing, 6 moving, and 2 jumping then $\text{IoU} = \frac{4 + 5 + 0}{7 + 6 + 2} = 0.6$, hence a match.

Baselines:

We compare with the following single-frame baseline models. One naive way to implement such a retrieval system is to learn a person action autoencoder, with its input and output a single person feature vector. Then concatenating the persons in the scene can be used for scene match. However, such direct reduction ignores all relationships in the scene ending with a weak scene representation. Another possibility is a direct concatenation of original persons feature vectors (e.g., 4096). Such a large scene representation may work in some domains, however, this large scene dimensionality is problematic.

B1) B1-Compact128: Autoencoder with input/output of a single person feature vector of length 4096 from the fc7 layer of a pre-trained VGG19 network. The 4096-d vector is fed to network layers of sizes 256, 128, 256, 4096. The middle layer (128 features) is used as a compressed representation of the person. This network is structured similar to our model and of same compact person size (128 features) for fair comparison.

B2) B2-VGG19: No autoencoder. Each single person is represented directly with a feature vector of length 4096 from the fc7 layer of a pretrained VGG19 network. Note that this baseline uses a much larger dimensionality (4096 vs. 128 features per person) and is especially problematic for representing scenes of many people.

Implementation Details:

The same settings are used as Sec. 5.4.1 except the following. We trained these models without person action labels for 150 epochs and initial learning rate $10^{-4}$. The MLP in the last relational layer ends with sigmoid non-linearities instead of ReLU. For person modeling, the ImageNet-pretrained VGG19 network is used as-is, without fine-tuning. The same setup is used for the Denoising Autoencoder, but with initial learning rate $10^{-3}$.

Results:

In this section we list our results for the retrieval tasks. We present the scene retrieval results, followed by single person retrieval. Then we discuss the performance of the models.

Table 5.2 compares the scene retrieval performance of our relational autoencoder with the baselines. We compute the Hit@K measure for $K \in \{1, 2, \ldots, 5\}$. Specifically, given a query frame, the frame is encoded using the autoencoder model and the closest $K$ matches in the database are retrieved. Recall, two frames are a match if the IoU of their actions

82
Table 5.2: Scene retrieval compared to baselines.

<table>
<thead>
<tr>
<th>Method</th>
<th>Hit@1</th>
<th>Hit@2</th>
<th>Hit@3</th>
<th>Hit@4</th>
<th>Hit@5</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>B1-Compact128</td>
<td>49.4</td>
<td>68.7</td>
<td>80.4</td>
<td>87.7</td>
<td>91.4</td>
<td>35.4</td>
</tr>
<tr>
<td>B2-VGG19</td>
<td>55.0</td>
<td>73.9</td>
<td>82.7</td>
<td>87.5</td>
<td>91.5</td>
<td>36.4</td>
</tr>
<tr>
<td>RAER-4L-2222C</td>
<td>57.4</td>
<td>76.7</td>
<td>85.3</td>
<td>90.4</td>
<td>93.3</td>
<td>36.8</td>
</tr>
</tbody>
</table>

Table 5.3: Scene retrieval compared to model variants.

<table>
<thead>
<tr>
<th>Method</th>
<th>Hit@1</th>
<th>Hit@2</th>
<th>Hit@3</th>
<th>Hit@4</th>
<th>Hit@5</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAER-2L-11C</td>
<td>56.8</td>
<td>74.9</td>
<td>84.5</td>
<td>89.8</td>
<td>92.6</td>
<td>36.8</td>
</tr>
<tr>
<td>RAER-2L-22C</td>
<td>56.9</td>
<td>75.6</td>
<td>84.9</td>
<td>90.0</td>
<td>93.3</td>
<td>36.7</td>
</tr>
<tr>
<td>RAER-4L-4224C</td>
<td>55.8</td>
<td>76.1</td>
<td>84.0</td>
<td>88.9</td>
<td>92.7</td>
<td>36.6</td>
</tr>
<tr>
<td>RAER-4L-2222C</td>
<td>57.4</td>
<td>76.7</td>
<td>85.3</td>
<td>90.4</td>
<td>93.3</td>
<td>36.8</td>
</tr>
</tbody>
</table>

≥ threshold (0.5). Mean average precision is also reported: mean of the average precision values for each image query where Euclidean distance is used as the confidence indicator. The training and testing sets are the ground truth annotated scenes in the Volleyball Dataset. Results indicate how this novel architecture is capable of capturing the context and encoding it within each person. Surprisingly, our model even beats the uncompressed VGG19, though VGG should be much stronger due to its size and sparsity.

In Table 5.3, we explore variants of our scene retrieval model. Specifically, we try 2 models with only 2 relational layers (128, 4096): One of these models uses 1 clique in all layers (RAER-2L-11C, all pair relationships) and the second uses 2 cliques (RAER-2L-22C, all pairs within a team). The complex version (RAER-4L-4224C) is 2 layers as our main model, but layer cliques are (4, 2, 2, 4). This means the decoder has to learn how to decode such hierarchical information.

In Table 5.4, we show the results for the Denoising Autoencoder when a person might be missing with probability 0.5 in the test data.

Table 5.4: Scene Retrieval using Denoising Autoencoder (-D) with 50% possible drop for people in test data for models and baselines. Our model is robust; the No Autoencoder model performance drops significantly.

<table>
<thead>
<tr>
<th>Method</th>
<th>Hit@1</th>
<th>Hit@2</th>
<th>Hit@3</th>
<th>Hit@4</th>
<th>Hit@5</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>B1-Compact128-D</td>
<td>38.1</td>
<td>58.8</td>
<td>70.5</td>
<td>78.2</td>
<td>84.7</td>
<td>34.6</td>
</tr>
<tr>
<td>B2-VGG19-D</td>
<td>34.0</td>
<td>51.1</td>
<td>62.2</td>
<td>70.0</td>
<td>76.0</td>
<td>34.9</td>
</tr>
<tr>
<td>RAER-4L-2222C-D</td>
<td>43.0</td>
<td>65.0</td>
<td>78.7</td>
<td>85.8</td>
<td>90.7</td>
<td>35.2</td>
</tr>
</tbody>
</table>

Table 5.5 compares the person retrieval performance of using the same relational autoencoder model with the baselines. The training and testing sets are the ground truth.
Table 5.5: Person Retrieval on Volleyball Dataset: Hit@K results of our method and baselines. Last column is mean average precision of query results. Our model outperforms the normal autoencoder model, and is competitive with a 32x larger sparse representation.

<table>
<thead>
<tr>
<th>Method</th>
<th>Hit@1</th>
<th>Hit@2</th>
<th>Hit@3</th>
<th>Hit@4</th>
<th>Hit@5</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>B1-Compact128-P</td>
<td>37.7</td>
<td>54.7</td>
<td>64.6</td>
<td>71.7</td>
<td>76.4</td>
<td>22.8</td>
</tr>
<tr>
<td>B2-VGG19-P</td>
<td>47.3</td>
<td>63.2</td>
<td>72.1</td>
<td>77.4</td>
<td>81.2</td>
<td>25.4</td>
</tr>
<tr>
<td>RAER-2L-11C-P</td>
<td>45.5</td>
<td>62.2</td>
<td>70.9</td>
<td>76.1</td>
<td>80.1</td>
<td>25.8</td>
</tr>
<tr>
<td>RAER-4L-2222C-P</td>
<td>42.6</td>
<td>58.3</td>
<td>68.3</td>
<td>73.7</td>
<td>77.8</td>
<td>25.2</td>
</tr>
</tbody>
</table>

bounding boxes of annotated actions in the Volleyball Dataset. Note that the Volleyball dataset consists of 9 action labels, with standing class representing ≈70% of the action labels, so a retrieval system that keeps retrieving standing samples will score high results. To avoid that, the standing class is removed from both the training and test sets in the person retrieval task. After training the model, we extract the compressed person representations for each person action and build a retrieval model for them. Results indicate that our compact person representation works well and beats the alternative compression baseline.

**Discussion:** The high Hit@K results indicate that the autoencoder approach works well for this task. From the scene and action retrieval results, we notice that our relational autoencoder outperforms the normal autoencoder model of the same structure and compression size due to encoding/decoding of person relationships. Of particular note, the autoencoder outperforms high-dimensional VGG features for scene retrieval. We hypothesize that this is due to the ability of the relational layers to capture contextual information among people in the scene. Fig 5.5 visualizes scene retrieval results.

5.5 Summary

We proposed a hierarchical relational network for learning feature representations. The network can be used in both supervised and unsupervised learning paradigms. We utilized this network for group activity recognition, based on the final compact scene layer. We also showed how the relational layer can be the main building block in novel autoencoder models that jointly encode/decode each person’s feature representation using a shared memory. Results in both tasks demonstrate the effectiveness of the relational network. The relationship graph associated with each layer allows explicit relationship consideration that can be applied to other visual understanding tasks.
Figure 5.5: Visualizations of scene retrieval using our relational autoencoder. Each 2 rows are a query: Query image first (blue box), followed by the closest 5 retrievals. Green Framed boxes are correct matches. The last query is for Right team winpoint event, and its results are 3 consecutive Right team winpoint events followed by 2 Left team winpoint events.
Chapter 6

Conclusion and Future Work

The primary focus of this dissertation is on representing and modeling the multi-person activity recognition problem. The material presented in this manuscript suggests that deep learning representation for individual people-level as well as the whole scene-level is effective and RNNs can capture the temporal dynamics existing at both levels. In addition, the thesis administrates the importance of considering relationships among people to gain more cues in resolving confusions regarding people actions. The main questions that this dissertation raises are: How could we build an appropriate feature representation of person actions in a video sequence? Or infer representations of his relationships with the surrounding people? Or in a more complex setup, how could we represent a whole dynamic scene of people and identify their collective activity?

In Chapter 3, we presented a novel deep structured architecture to deal with the multi-person activity recognition problem. Through a two-stage process, we learn a temporal representation of person-level actions and combine the representation of individual people to recognize the group activity. LSTMs were used to capture the temporal dynamics that exist at the level of individual person actions as well as at the level of group activity. Specifically, one LSTM model is designed to represent action dynamics of individual people in a video sequence and another LSTM model is designed to aggregate person-level information for group activity recognition. AlexNet was used for visual features extraction from people bounding boxes and an appearance-based tracker was used to track the people bounding boxes for $T$ timesteps.

There are two limitations in the previous model that are addressed in Chapter 4 and Chapter 5. First, an appearance-based tracker is run independently per person and no considerations for tublets intersections where tublets may get switched (e.g., the first part of tublet 1 is linked with the end part of tublet 2). In Chapter 4 we present an approach for finding tublets in an efficient and effective way to alleviate such shortcomings. Second, the proposed models don’t make use of the relationships between people, although information regarding the actions of other people can provide useful cues for recognizing the action of an
individual. A relational network module is presented in Chapter 5 that utilizes the provided relationships to build relational representations of people.

To find better tublets than the previous model, in Chapter 4 we propose a framework that generates multiple generic class-independent tubelet proposals in each video then classifies them. First, through our Tube Proposal Network, we use Faster-Rcnn to find generic class-agnostic region proposal that probably contain human actions. Given the bounding boxes proposals of the current frame, we use a greedy algorithm to link these proposals to the current set of tublets computed up to the last frame. The next stage is the Temporal Understanding Network where the tublets are fed to an LSTM-based network to classify them. We use weighted average of actions score from the TPN and the TUN to compute the final tubelet actions scores.

To address the context issue of the model presented in Chapter 3, we explicitly represent the relationships between people as a graph in Chapter 5. Our Hierarchical Relational Network computes relational representations of people, given graph structures describing potential interactions. Each relational layer is fed individual person representations and a potential relationship graph. Relational representations of each person are created based on their connections in this particular graph and the whole scene is generated by pooling the final individual representations. Beyond relationship representation in the multi-person activity recognition problem, our relational network is a generic relational module that can process any $K$ objects with a given potential graph for their relationships. For example, we may have representation for $K$ documents that have a relationship graph computed using some documents similarity algorithm.

Though the models presented in Chapter 4 and Chapter 5 work on limitations of our first model, they have their own limitations. For example, our presented action tubes algorithm is designed mainly for trimmed videos while real video scenes need untrimmed videos. A temporal localization approach is needed to identify where a tubelet start and end before our approach is applied for such trimmed sequence of frames. An important drawback in our relational module is the need for an explicit graph that encodes the potential relationships between people in the scene. In the future, we may try to jointly learn a graph representation of people in the scene while computing relational representations of them (e.g., using gating functions [24]).

Beyond identifying a person primitive action or the whole scene activity, in the future we may target complex event retrieval problems where the query is a sequence of actions with spatial and temporal constraints. For example, given a video clip for a volleyball game, answer the following text query: Find a set action followed by a move action of the same player. After that, a player from the same team is doing a jump action close to another player doing a spike action. The previous query involves 3 persons and both spatial and temporal constraints. Having a system that retrieves such complex queries may assist us finding useful information in multi-person setups.
Bibliography


