Structural Recurrent Neural Network (SRNN) for Group Activity Analysis

Sovan Biswas
University of Bonn
sbiswas@uni-bonn.de

Juergen Gall
University of Bonn
gall@iai.uni-bonn.de

Abstract

A group of persons can be analyzed at various semantic levels such as individual actions, their interactions, and the activity of the entire group. In this paper, we propose a structural recurrent neural network (SRNN) that uses a series of interconnected RNNs to jointly capture the actions of individuals, their interactions, as well as the group activity. While previous structural recurrent neural networks assumed that the number of nodes and edges is constant, we use a grid pooling layer to address the fact that the number of individuals in a group can vary. We evaluate two variants of the structural recurrent neural network on the Volleyball Dataset.

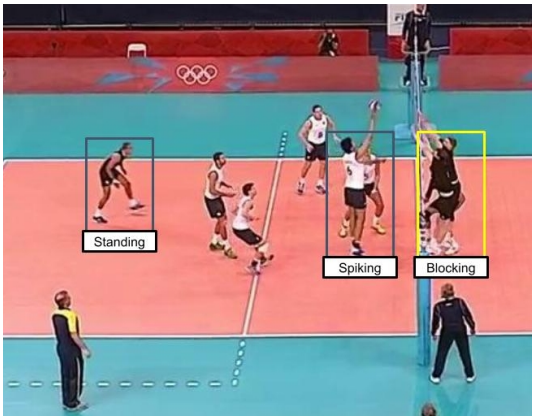

Figure 1. A frame labeled as group activity "Left Spike" and bounding boxes around each team player are annotated in the dataset with individual actions [12].

1. Introduction

Activity analysis has been of great interest in computer vision since decades. In recent years, deep learning approaches such as [4, 15, 24, 26, 16] have been proposed to recognize activities in videos. Most of these approaches, however, focus on single person activity analysis and estimate only one activity per video clip. Similar to other recent works [21, 8, 1, 23], the goal of this paper is to understand and analyze the actions of individuals and their interactions, and subsequently use them for predicting the group activity.

Recent deep learning approaches for group activity analysis such as [11, 23] use a multi-level hierarchy of recurrent neural networks (RNNs) for group activity recognition. In these approaches, the lower level RNNs focus on understanding and modeling the actions of individuals and the higher level RNNs in the architecture model the group activity. These approaches are trained using a two-step process. The first step focuses on improving the recognition of the actions of each individual independently and the subsequent step focuses on recognizing the group activity given the recognized actions of the individuals.

Apart from the hindrance of two-step training, these methods lack the capability of capturing interactions between individual persons when present within a group. For example, in a volleyball game as shown in Figure 1, a person from the team on the left-hand side of the volleyball court performs the individual action "spiking" whereas the player from the opponent team performs a "blocking" action. Looking only at the individuals makes it difficult to distinguish between the individual actions, but there is a strong correlation between the two activities. Similarly, when walking in a crowd, people move and walk in various directions just to avoid colliding with each other. In general, the action of an individual in a group is influenced by the actions of the other individuals in the group. This phenomenon not only provides context that helps to recognize the individual actions but also provides a key information about the group level actions such as in the case of volleyball as shown in Figure 1. Thus, there is an imperative need to analyze interactions between individuals and to capture the influence over time when analyzing a group of humans.

The main focus of the paper is to harness such interactions within a group to improve the recognition of the group activity as well as the individual actions. To this end, we build on the recently proposed structural recurrent neural network (SRNN) [14] which has the unique capability of capturing interactions as contextual information using an interconnected set of RNNs. While in [14], the number of nodes and edges and therefore the number of RNNs is con-

stant, we extend the approach to handle a varying number of nodes and edges as it is required for analyzing group activities.

The rest of the paper is structured as follows. We start with a brief discussion of the related work in Section 2, followed by a short introduction of SRNNs in Section 3. In Section 4, we introduce two variants of an SRNN. We then evaluate the two variants in Section 5 and conclude with a summary in Section 6.

2. Related Work

One of the earlier approaches for analyzing the activities of a group or crowd was proposed by [6]. They introduce crowd context in recognizing the activity being performed by each individual in the group. Traditionally graphical models with key contextual features [5, 7] have been deployed rigorously towards group analysis. However these models with handcrafted features [22] were outperformed by newer deep neural network architectures such as [8, 1, 23]. For group activity recognition, most of these deep neural networks are inspired by [11] which uses a multi-level cascade of recurrent neural networks for group activity recognition. In this approach, humans are detected and tracked to form multi-person tracklets. These tracklets along with their deep visual features are fed to the lower level RNNs. The focus of these lower level RNNs is to understand and model the actions of the individual persons. The higher level RNNs in the architecture instead focus on understanding the group activity. The individual actions and group activity predictions are done using softmax in a feed-forward way. However, each method tackles a very different problem in the same framework. Shu *et al*. [23] use a similar hierarchical architecture but the approach differs from previous work by proposing an energy based approach that works significantly better if the amount of data is small. Furthermore, this approach also explores human interaction, but holistically by convolutional features extracted from both humans. [3] propose a joint approach for detecting humans and predicting their actions.

Spatio-temporal graphs have been used in computer vision for various applications such as predicting human movements [13] or learning human activities and object affordances [18]. The spatio-temporal graphs represent in these works spatio-temporal relations between joints or joints and objects in a video. The methods [19, 2] use handcrafted features along with graphical models, conditional random field or random forest for the aforementioned applications. Recently, neural networks have been deployed to solve spatio-temporal graph problems. For example, [9] uses deep networks followed by inference using a probabilistic graphical model to recognize the actions in a group. Our approach builds on the work [14] where a set of coupled RNNs are used to represent spatio-temporal graphs.

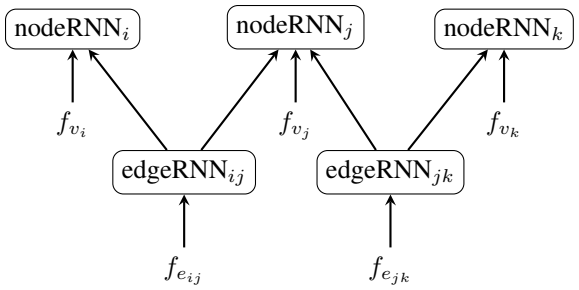

Figure 2. Feedforward network of a Structural RNN (SRNN) when trained with respect to node labels.

3. Structural RNN

Recurrent neural networks are very effective in modeling temporal sequences. In case of a single person, features f^t are extracted at each frame and used as input for an RNN to predict the action classes y^t over time. However when in a group, a person performs an action based on its interaction with other persons and the group objective. So, a single recurrent neural network is incapable of capturing the interactions and group dynamics, thus reducing its effectiveness. For solving similar problems, Jain *et al*. [14] proposed an interconnected set of recurrent neural networks that not only captures the individual behavior over time but also integrates the interactions between the individuals through edges.

An example of an SRNN is illustrated in Figure 2. It consists of three nodes v_i, v_j, and v_k and the goal is to predict for each node the class labels over time, which are denoted by $y^t_{v_i}$, $y^t_{v_j}$, and $y^t_{v_k}$, respectively. Each node is modeled by an RNN, termed nodeRNN. It takes as input some features $f^t_{v_i}$, which are extracted for a node v_i, but also the output of RNNs that model the interactions with other nodes. The second type of RNN is termed edgeRNN. The edgeRNNs take as input some features $f^t_{e_{ij}}$ based on the spatio-temporal relation between two nodes v_i and v_j and predicts a latent representation $h^t_{e_{ij}}$, which is forwarded to the corresponding nodeRNNs. The advantage of such an SRNN is that the nodeRNNs and the edgeRNNs can be trained jointly such that the prediction of the node labels depends not only on the features that are extracted for each node but also on the interactions between the nodes.

4. Group Activity Analysis

In this section, we will first briefly introduce the problem in Section 4.1. This is followed by introducing two different variants of an SRNN for group activity recognition in Section 4.2.

4.1. Problem Formulation

Our objective is to predict jointly the group activity label y_g^t of a group as well as the action label $y_{v_i}^t$ for each individual v_i of the group over time. We assume that the bounding box for each person has been already extracted and we compute features for each individual person $f_{v_i}^t$ and for each edge $f_{e_{ij}}^t$ between two individuals. The features are described in Section 5.2 and we denote the set of all node and edge features for a frame by F^t. In order to learn the parameters θ of the models which are described in Section 4.2, we minimize the loss

$$\arg\min_{\theta} \Big[L\left(\psi_g\left(F^t;\theta\right), y_g^t\right) + \frac{1}{n}\sum_{i=1}^{n} L\left(\psi_v\left(F^t;\theta\right), y_{v_i}^t\right)\Big], \quad (1)$$

where L denotes the cross-entropy loss, $\psi_g(.)$ the prediction function of the model for the group activity, and $\psi_v(.)$ the prediction function for the actions of the individuals.

4.2. SRNN for Group Activity Analysis

The proposed group activity recognition approach is formulated as a two-level hierarchy of recurrent neural networks similar to [11, 23]. The lower level predicts individual actions followed by the higher level recurrent network that estimates the group activity. In Sections 4.2.1 and 4.2.2, we discuss two SRNN variants that jointly estimate the group activity and the individual actions by modeling the interactions between individuals. The two SRNNs are shown in Figures 3 and 4.

4.2.1 SRNN-MaxNode

As shown in Figure 1, when a person on the left hand side jumps to perform the action "spiking", the opponents jump to block the spike, resulting in the action "blocking" across the volleyball net. This is an example where persons in a group perform contextual actions. Structural RNNs are an efficient approach to model such relations.

As discussed in Section 3, SRNNs are a hierarchy of RNNs consisting of edgeRNNs and nodeRNNs. The first variant that we propose for group activity analysis is shown in Figure 3. The lowest level consists of edgeRNNs that model interactions between two individuals based on their relative position, which is encoded by the feature vector $f_{e_{ij}}^t$. The output of the edgeRNNs is feedforwarded to the nodeRNNs. The number of individuals, however, varies in a group and each individual might have a different number of neighbors and in very dense crowds the number of neighbors could be very high. The approach proposed in [14] cannot handle such cases since it concatenates the features, assuming that the number of edges and nodes is constant. To address this problem, we propose a grid pooling layer that combines for a node v_i the output from all edgeRNNs e_{ij} based on the position of the neighboring persons v_j in a prescribed grid. The prescribed grid regions are arranged as shown in Figure 5. If several neighbors are in the same grid cell, we sum the output of the edgeRNNs instead of averaging them. This means that the values are usually larger when more persons are in a cell. The grid pooling provides for each node v_i features for 8 cells that are then concatenated with additional CNN features $f_{v_i}^t$, which are extracted from the frame t for each person. The output of the nodeRNNs is used in two ways. First, the action class $y_{v_i}^t$ of the person v_i is predicted using an additional softmax layer. Second, the group activity is estimated similar to [11] by max-pooling the outputs of the nodeRNNs of a group at a time instant t, which is then used as input for an RNN that predicts the group activity y_g^t over time.

In summary, the SRNN-MaxNode model is defined as follows:

$$\begin{aligned} h_{e_{ij}}^t &= \text{RNN}_e(h_{e_{ij}}^{t-1}, f_{e_{ij}}^t) \\ h_{C_i}^t &= \sum_{j \in S_{C_i}} h_{e_{ij}}^t \\ h_{e_i}^t &= [h_{L^i}^t \dots h_{Q_4^i}^t] \\ h_{v_i}^t &= \text{RNN}_v(h_{v_i}^{t-1}, h_{e_i}^t, f_{v_i}^t). \end{aligned} \quad (2)$$

While $h_{e_{ij}}^t$ denotes the output from the edgeRNN for the nodes v_i and v_j at frame t, S_{C_i} denotes the set of neighboring nodes of v_i in the cell $C_i \in \{L^i, R^i, A^i, B^i, Q_1^i, Q_2^i, Q_3^i, Q_4^i\}$, which are the grid regions for v_i as shown in Figure 5. The accumulated values for each cell $h_{C_i}^t$ are then concatenated to the vector $h_{e_i}^t$ and the output of the nodeRNN is denoted by $h_{v_i}^t$.

The full architecture can thus be defined as:

$$\begin{aligned} h_{v_i}^t &= \text{SRNN}(f_{v_i}^t, f_{e_{ij}}^t) \\ y_{v_i}^t &= \phi_v(h_{v_i}^t, W_v) \\ h_p^t &= \max\left(h_{v_1}^t \dots h_{v_n}^t\right) \\ h_g^t &= \text{RNN}_g(h_g^{t-1}, h_p^t) \\ y_g^t &= \phi(h_g^t, W_g), \end{aligned} \quad (3)$$

where $h_{v_i}^t$ denotes the output after the SRNN (2) and W_v denotes the weights used in the softmax function $\phi_v(.)$ to predict the individual actions $y_{v_i}^t$. h_p^t denotes the max-pooled representation over the complete group and h_g^t denotes the output of the group RNN, which is then used by a softmax function $\phi(.)$ with weights W_g to predict the group activity y_g^t.

4.2.2 SRNN-MaxEdge

While the SRNN in Figure 3 uses the edgeRNNs to provide contextual information for the nodeRNNs, we also compare

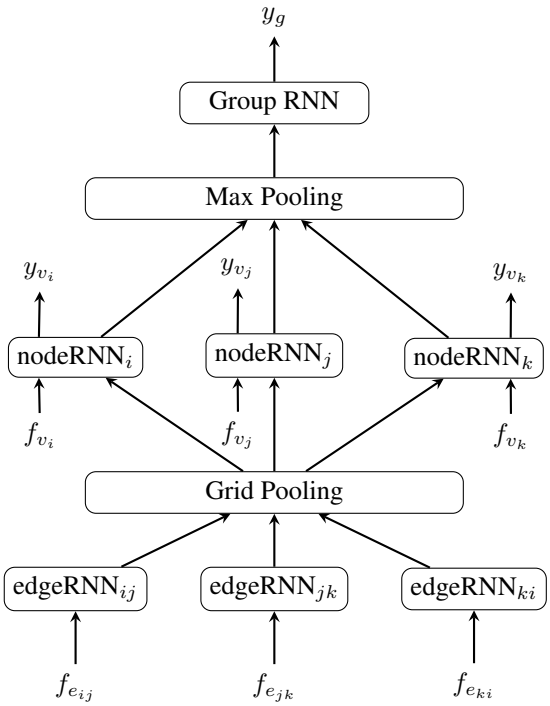

Figure 3. SRNN-MaxNode: Feedforward SRNN where max pooling is performed over the nodeRNNs. The nodeRNNs are enriched using the output of the edgeRNNs using a novel grid pooling approach.

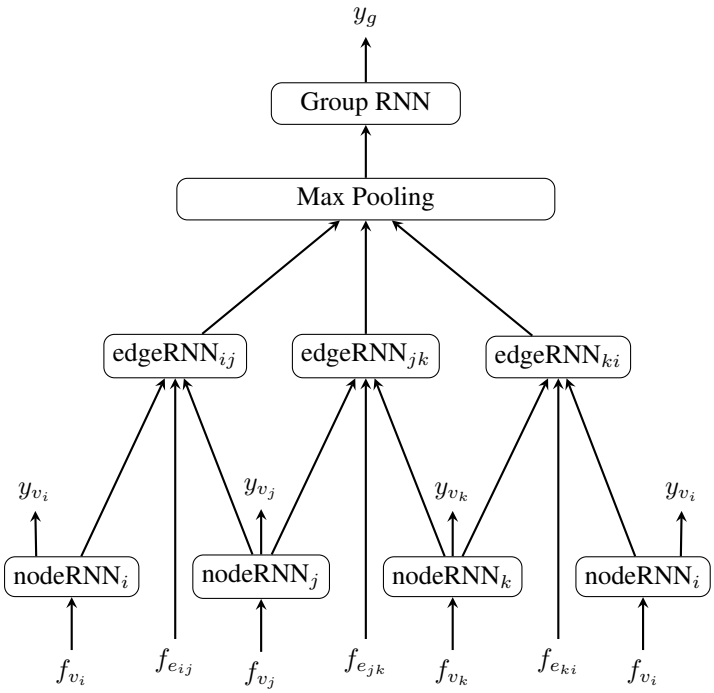

Figure 4. SRNN-MaxEdge: Feedforward SRNN where max pooling is performed over the edgeRNNs. The edgeRNNs are enriched using the output of the nodeRNNs.

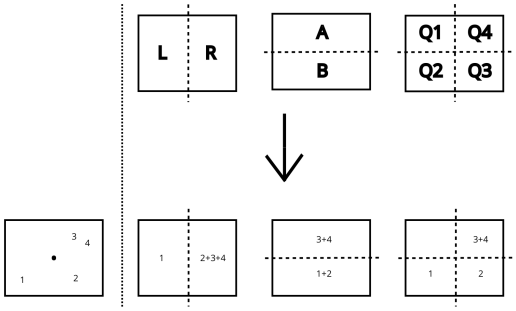

Figure 5. Grid pooling. Left: 1, 2, 3 and 4 denote four persons in the neighborhood of the person •. Right: We define three grid structures (top) and we sum the outputs of the edgeRNNs where the neighbors of • are in the same cell. We then concatenate the features of the eight cells. If a cell is empty, the feature vector is set to zero.

it to an SRNN where the nodeRNNs are at the lowest level of the hierarchy. In this case, the max pooling is not performed over the nodeRNNs but over the edgeRNNs and an additional grid pooling is not required since each edge consists of two nodes. We denote the second variant, which is shown in Figure 4, SRNN-MaxEdge.

For SRNN-MaxEdge, the lower level of the hierarchy consists of nodeRNNs that predict the individual actions based on the individual CNN features $f_{v_i}^t$. The output from the nodeRNNs is forwarded to the corresponding edgeRNNs, which also take the edge features $f_{e_{ij}}^t$ as input. The output of the edgeRNNs at a time instant t is max pooled and then used as input for an RNN that predicts the group activity y_g^t over time.

In summary, the SRNN-MaxEdge model is defined as follows:

$$\begin{aligned} h_{v_i}^t &= \text{RNN}_v(h_{v_i}^{t-1}, f_{v_i}^t) \\ y_{v_i}^t &= \phi_v(h_{v_i}^t, W_v) \\ h_{e_{ij}}^t &= \text{RNN}_e(h_{e_{ij}}^{t-1}, h_{v_i}^t, h_{v_j}^t, f_{e_{ij}}^t), \end{aligned} \tag{4}$$

where $h_{v_i}^t$ denotes the output of the nodeRNN for person v_i at a given time t and W_v are the weights used in the softmax function $\phi_v(.)$ to predict the individual action $y_{v_i}^t$. $h_{e_{ij}}^t$ denotes the output of the edgeRNNs.

The full architecture can thus be defined as:

$$\begin{aligned} h_{e_{ij}}^t &= \text{SRNN}(f_{v_i}^t, f_{e_{ij}}^t) \\ h_p^t &= \max(h_{e_{12}}^t \dots h_{e_{mn}}^t) \\ h_g^t &= \text{RNN}_g(h_g^{t-1}, h_p^t) \\ y_g^t &= \phi(h_g^t, W_g), \end{aligned} \tag{5}$$

where $h_{e_{ij}}^t$ denotes the output from the SRNN (4). While h_p^t denotes the max pooled representation over all edges in

the group, h_g^t denotes the output of the group RNN, which is then used by a softmax function $\phi(.)$ with weights W_g to predict the group activity y_g^t.

5. Experiments

5.1. Datasets

We evaluate our framework on the recently introduced volleyball dataset [12]. This dataset has 55 volleyball game video sequences with 4830 labeled frames, where each player is labeled and subsequently annotated with the bounding box. Each player performs one of the 9 individual actions resulting in one of the 8 group activity labels. Furthermore, the whole dataset is divided into non-overlapping sets of 24 sequences for training, 15 sequences for validation and the remaining sequences are used for testing. Similar to [11, 23], we have used both training and validation sequences for training. Since not all frames are annotated by bounding boxes, the Dlib tracker [17] is used to propagate the ground-truth bounding boxes to the unannotated frames.

5.2. Implementation Details

For the RNNs, we use standard LSTMs [10] and the implementation is done using the Tensorflow library. At the node level, each LSTM is connected with a deep convolutional network such as Alexnet [20] or VGG 16 [25] to compute the visual features f_{v_i} based on the annotated and tracked bounding boxes of the persons. Similar to [11, 23], we initialize the CNNs by a model that has been pre-trained on ImageNet. During training, we fine-tune only the last two fully connected layers of the CNNs.

The edge features $f_{e_{ij}}^t$ model spatio-temporal relations between the bounding boxes of two persons. We take the center of each bound box and compute the difference vector (dx, dy). We then compute the basic distance values ($|dx|$, $|dy|$, $|dx + dy|$, $\sqrt{(dx)^2 + (dy)^2}$) and add the direction of the translational vector ($arctan(dy, dx)$, $arctan2(dy, dx)$). To further enhance these simple 6 interaction features, we also compute the difference of the 6 features between two consecutive time frames, which results in 6 additional features. We finally compute the 12 features not only for frame t, but also for the neighboring frames $t - 1$ and $t + 1$ to capture some short temporal information. All features are concatenated to obtain a 36 dimensional feature vector.

The training is performed in two stages. In the first stage, the nodeRNNs are trained independently using individual actions and the cross-entropy as loss function. For this stage, we have used a batch size of 36 for our experiments. In the next step, we train the whole architecture by minimizing the loss (1). We use Adam as optimizer with a learning rate of 0.00001. During training, the parameters of the nodeRNNs and the last two layers of the CNN are updated as well. The nodeRNNs have 3000 hidden units and the group RNN has 2000 hidden units. The number of nodes for the edgeRNNs differs between the SRNN-MaxNode and the SRNN-MaxEdge due to differences of the input features. While the edgeRNNs in the SRNN-MaxNode take as input the low dimensional features $f_{e_{ij}}^t$, the edgeRNNs in the SRNN-MaxEdge use the additional output of two nodeRNNs as input. For the edgeRNNs in the SRNN-MaxNode, we use therefore only 30 hidden units whereas 1000 hidden units are used for the edgeRNNs in the SRNN-MaxEdge. Since the two variants differ in memory consumption and the GPU memory is limited, we use a batch size of 30 for SRNN-MaxNode and a batch size of 16 is used for SRNN-MaxEdge.

In accordance with other approaches [11, 23, 3], we have also performed experiments where we divide the individuals in two groups. For this, we use the same approach as in Ibrahim *et al*. [12].

5.3. Experimental Evaluation

5.3.1 Variation of Hierarchical LSTM

As the proposed approaches are inspired by Hierarchical LSTMs [12], we have first performed a few baseline experiments to evaluate versions of the Hierarchical LSTM [12]:

- 2-layer LSTMs (V1): This is similar to [12] with a two-step training for person level RNNs followed by training of the group level RNN given the pre-trained person RNNs. Unlike [11], the group level takes as input only the output of person RNNs and does not use any additional CNN features.
- 2-layer LSTMs (V2): Reimplementation of [12].
- 2-layer LSTMs (V3): This is similar to V1 but person RNNs and group RNN are jointly trained using the loss (1). As described in Section 5.2, the last two layers of the Alexnet are fine-tuned.

The accuracy of recognizing the group activity as well as the actions of the individuals is reported in Table 1. The results show that training the person RNNs and the group RNN jointly (V3) using the loss (1) improves the accuracy of the group activity also for the Hierarchical LSTM [12], but it slightly decreases the individual action recognition results. Dividing the individuals into two groups as in [12] improves the accuracy by a large margin due to the volleyball scenario where two teams play against each other.

5.3.2 Comparison to state-of-the-art

Table 2 compares the proposed SRNN approaches with the Hierarchical LSTM V3 that is trained with the same loss function and uses the same Alexnet CNN. While SRNN-MaxNode outperforms the Hierarchical LSTM both for

Method	Group Activity Accuracy	Individual Action Recognition Accuracy
Hierarchical LSTM [12] (1 group)	70.3%	-
Hierarchical LSTM V1 (1 group)	68.37%	**76.32%**
Hierarchical LSTM V2 (1 group)	73.89%	**76.32%**
Hierarchical LSTM V3 (1 group)	**74.01%**	75.96%
Hierarchical LSTM [12] (2 groups)	81.9%	-
Hierarchical LSTM V1 (2 groups)	78.37%	**76.32%**
Hierarchical LSTM V2 (2 groups)	81.33%	**76.32%**
Hierarchical LSTM V3 (2 groups)	**83.12%**	75.96%

Table 1. Comparison of various variations of the Hierarchical LSTM [12] using Alexnet features.

Method	Group Activity Accuracy	Individual Action Recognition Accuracy
Hierarchical LSTM [12] (1 group)	70.3%	-
Hierarchical LSTM V3 (1 group)	74.01%	75.96%
CERN [23] (1 group)	73.5%	69%
SRNN-MaxNode (1 group)	**74.39%**	**76.65%**
SRNN-MaxEdge (1 group)	68.39%	76.03%
Hierarchical LSTM[12] (2 groups)	81.9%	-
Hierarchical LSTM V3 (2 groups)	83.12%	75.96%
CERN [23] (2 groups)	83.3%	69%
SRNN-MaxNode (2 groups)	**83.47%**	**76.65%**
SRNN-MaxEdge (2 groups)	79.86%	76.03%
Social Scene [3] (2 groups)	**89.9%**	**82.4%**

Table 2. Comparison to the state-of-the-art.

group activity recognition as well as the recognition of the individual actions, SRNN-MaxEdge achieves a lower group activity accuracy than SRNN-MaxNode and Hierarchical LSTM. It shows that the max pooling over the nodeRNNs is better than pooling over the edgeRNNs on this dataset since the max pooling over the nodeRNNs forwards the features of the most important individual to the group RNN. This works for group activities as shown in Figure 1 very well since the group activity can be well inferred from the "spiking" person. Our proposed approach SRNN-MaxNode also outperforms the approach CERN [23], which also uses Alexnet features. However, the recent approach [3], which builds on the Inception-V3 CNN [27], achieves the highest accuracy on this dataset.

5.3.3 Impact of CNN architecture

We have also analyzed the impact of the CNN architecture and compare the used Alexnet CNN with the larger VGG 16 network. The results are reported in Table 3. The accuracy of the VGG 16 network decreases the accuracy of the Hierarchical LSTM as well as the proposed SRNN-MaxNode. For SRNN-MaxEdge the accuracy remains nearly the same. The decrease in accuracy might be due to overfitting, but it needs further investigation to analyze the impact of the used CNN model in more detail.

5.3.4 Impact of deep edge features

For the model SRNN-MaxNode, we use a low dimensional feature vector $f_{e_{ij}}^t$ that encodes simple spatio-temporal relations between two bounding boxes. We also investigated if the accuracy can be improved when the edgeRNNs not only take $f_{e_{ij}}^t$ as input feature but also $f_{v_i}^t$ and $f_{v_j}^t$. Since this increases the dimensionality of the input feature from 36 to 8228 (4096+4096+36), we also increase the number of hidden units of the EdgeRNNs from 30 to 1000 to address the higher dimensionality. As shown in Table 4, adding $f_{v_i}^t$ and $f_{v_j}^t$ does not improve the accuracy. This is expected since the features $f_{v_i}^t$ are already added to the nodeRNNs and adding them twice does not provide additional information for the model.

6. Conclusions

In this work, we have proposed two variants of structural recurrent neural networks (SRNN) to recognize the actions of individuals as well as the activity of the entire group jointly. The advantage of the SRNN approach is that it explicitly models relations between individuals and all RNNs can be trained together using a single loss function. We

Feature	Method	Group Activity Accuracy	Individual Action Recognition Accuracy
Alexnet	H. LSTM V3 - (1 group)	74.01%	75.96%
	SRNN-MaxNode - (1 group)	**74.39%**	**76.65%**
	SRNN-MaxEdge - (1 group)	68.39%	76.03%
VGG 16	H. LSTM V3 - (1 group)	70.34%	75.30%
	SRNN-MaxNode - (1 group)	71.20%	74.85%
	SRNN-MaxEdge - (1 group)	68.29%	75.96%
Alexnet	H. LSTM V3 - (2 groups)	83.12%	75.96%
	SRNN-MaxNode - (2 groups)	**83.47%**	**76.65%**
	SRNN-MaxEdge - (2 groups)	79.86%	76.03%
VGG 16	H. LSTM V3 - (2 groups)	81.34%	75.30%
	SRNN-MaxNode - (2 groups)	82.86%	74.85%
	SRNN-MaxEdge - (2 groups)	79.92%	75.96%

Table 3. Comparison of the proposed SRNN approaches with Hierarchical LSTM V3 using Alexnet or VGG 16 as CNN.

Edge feature	Group Activity Accuracy	Individual Action Recognition Accuracy
$f_{e_{ij}}^t$ (1 group)	74.39%	**76.65%**
$(f_{e_{ij}}^t, f_{v_i}^t, f_{v_j}^t)$ (1 group)	**74.48%**	75.89%
$f_{e_{ij}}^t$ (2 groups)	**83.47%**	**76.65%**
$(f_{e_{ij}}^t, f_{v_i}^t, f_{v_j}^t)$ (2 groups)	83.27%	75.89%

Table 4. Comparison of edge features using SRNN-MaxNode.

evaluated the models on the Volleyball Dataset and showed that the SRNN model outperforms hierarchical LSTMs.

Acknowledgment

The work has been financially supported by the ERC Starting Grant ARCA (677650).

References

[1] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, F. Li, and S. Savarese. Social LSTM: Human Trajectory Prediction in Crowded Spaces. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 961–971, 2016.

[2] M. R. Amer, P. Lei, and S. Todorovic. Hirf: Hierarchical Random Field for Collective Activity Recognition in videos. In *European Conference on Computer Vision*, pages 572–585, 2014.

[3] T. M. Bagautdinov, A. Alahi, F. Fleuret, P. Fua, and S. Savarese. Social Scene Understanding: End-to-End Multi-person Action Localization and Collective Activity Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3425–3434, 2017.

[4] J. Carreira and A. Zisserman. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4724–4733, 2017.

[5] W. Choi and S. Savarese. Understanding Collective Activities of people from videos. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(6):1242–1257, 2014.

[6] W. Choi, K. Shahid, and S. Savarese. What are they doing?: Collective activity classification using spatio-temporal relationship among people. In *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1282–1289. IEEE, 2009.

[7] W. Choi, K. Shahid, and S. Savarese. Learning context for collective activity recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3273–3280, 2011.

[8] Z. Deng, A. Vahdat, H. Hu, and G. Mori. Structure Inference Machines: Recurrent Neural Networks for Analyzing Relations in Group Activity Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4772–4781, 2016.

[9] Z. Deng, M. Zhai, L. Chen, Y. Liu, S. Muralidharan, M. J. Roshtkhari, and G. Mori. Deep Structured Models for Group Activity Recognition. In *Proceedings of the British Machine Vision Conference*, pages 179.1–179.12, 2015.

[10] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997.

[11] M. S. Ibrahim, S. Muralidharan, Z. Deng, A. Vahdat, and G. Mori. A Hierarchical Deep Temporal Model for Group Activity Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[12] M. S. Ibrahim, S. Muralidharan, Z. Deng, A. Vahdat, and G. Mori. Hierarchical Deep Temporal Models for Group Activity Recognition. *arXiv preprint arXiv:1607.02643*, 2016.

[13] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human3.6M: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(7):1325–1339, 2014.

[14] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena. Structural-RNN: Deep Learning on Spatio-Temporal Graphs. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5308–5317, 2016.

[15] S. Ji, W. Xu, M. Yang, and K. Yu. 3D Convolutional Neural Networks for Human Action Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(1):221–231, 2013.

[16] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and F. Li. Large-Scale Video Classification with Convolutional Neural Networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[17] D. E. King. Dlib-ml: A machine Learning Toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

[18] H. S. Koppula, R. Gupta, and A. Saxena. Learning human activities and object affordances from RGB-D videos. *I. J. Robotics Res.*, 32(8):951–970, 2013.

[19] H. S. Koppula and A. Saxena. Anticipating Human Activities using Object Affordances for Reactive Robotic Response. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(1):14–29, 2016.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, pages 1106–1114, 2012.

[21] V. Ramanathan, J. Huang, S. Abu-El-Haija, A. N. Gorban, K. Murphy, and L. Fei-Fei. Detecting Events and Key Actors in Multi-person Videos. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3043–3053, 2016.

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

[23] T. Shu, S. Todorovic, and S. Zhu. CERN: Confidence-Energy Recurrent Network for Group Activity Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4255–4263, 2017.

[24] K. Simonyan and A. Zisserman. Two-Stream Convolutional Networks for Action Recognition in Videos. In *Advances in Neural Information Processing Systems*, pages 568–576, 2014.

[25] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[26] B. Singh, T. K. Marks, M. J. Jones, O. Tuzel, and M. Shao. A Multi-stream Bi-directional Recurrent Neural Network for fine-grained action detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1961–1970, 2016.

[27] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the Inception Architecture for Computer Vision. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.