Neurocomputing 330 (2019) 449–464

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Localizing salient body motion in multi-person scenes using convolutional neural networks

Florian Letsch[a,b,1,*], Doreen Jirak[a], Stefan Wermter[a]

[a] Department of Computer Science, Knowledge Technology, University of Hamburg, Vogt-Koelln-Strasse 30, Hamburg D-22527, Germany
[b] Twenty Billion Neurons GmbH, Berlin, Germany

ARTICLE INFO

Article history:
Received 5 January 2018
Revised 12 November 2018
Accepted 15 November 2018
Available online 17 November 2018

Communicated by Prof. Zidong Wang

Keywords:
Convolutional neural networks
Gestures
Computer vision
Detection
Localization
Saliency

ABSTRACT

With modern computer vision techniques being successfully developed for a variety of tasks, extracting meaningful knowledge from complex scenes with multiple people still poses problems. Consequently, experiments with application-specific motion, such as gesture recognition scenarios, are often constrained to single person scenes in the literature. Therefore, in this paper we address the challenging task of detecting salient body motion in scenes with more than one person. We propose a neural architecture that only reacts to a specific kind of motion in the scene: A limited set of body gestures. The model is trained end-to-end, thereby avoiding hand-crafted features and the strong reliance on pre-processing as it is prevalent in similar studies. The presented model implements a saliency mechanism that reacts to body motion cues which have not been included in previous computational saliency systems. Our architecture consists of a 3D Convolutional Neural Network that receives a frame sequence as its input and localizes active gesture movement. To train our network with a large data variety, we introduce an approach to combine Kinect recordings of one person into artificial scenes with multiple people, yielding a large diversity of scene configurations in our dataset. We performed experiments using these sequences and show that the proposed model is able to localize the salient body motion of our gesture set. We found that 3D convolutions and a baseline model with 2D convolutions perform surprisingly similar on our task. Our experiments revealed the influence of gesture characteristics on how well they can be learned by our model. Given a distinct gesture set and computational restrictions, we conclude that using 2D convolutions might often perform equally well.

1. Introduction

1.1. Motivation

As computer vision techniques become continually better at detection and classification tasks, scenes containing visual clutter and multiple people still pose difficult challenges. Occlusions, varying object shapes and distracting motion cues make it hard to extract meaningful high-level information. The full detail of the low-level signal is irrelevant for most applications and the important information is often found in the salient parts of an image or a video. In order to use this challenging visual data, robust computer vision techniques are needed that are capable of solving these problems. One of the neural architectures that can reliably extract features from such complex and noisy visual data are Convolutional Neural Networks (CNNs) [28]. Their capabilities to generalize from noisy and complex visual data make them particularly well suited for real-world vision applications.

In our study, we analyzed the potential of using CNNs to localize salient body movement in scenes with multiple people because such scenes are typical for vision applications in real environments. We investigated a scenario that extended a typical one-person lab experiment to contain several people. One of those people performed a dynamic body gesture while the others were passive observers and only performed subtle movements. The task of the CNN was to detect and localize the person performing a gesture from an image sequence and ignore non-gesture movements.

To achieve this, we propose a 3D CNN architecture that takes a sequence of depth frames from an RGB-D device as its input. It was trained end-to-end to produce a 2D output with bright pixels representing the location of the active gesture performer. We used 3D convolution kernels that allowed to input a fixed-size sequence

* Corresponding author at: Department of Computer Science, Knowledge Technology, University of Hamburg, Vogt-Koelln-Strasse 30, Hamburg D-22527, Germany.
E-mail address: florian.letsch@20bn.com (F. Letsch).
[1] This research was done prior to joining Twenty Billion Neurons GmbH.

https://doi.org/10.1016/j.neucom.2018.11.048

to the network. As a baseline performance indicator, we also designed a 2D CNN that only received a still frame as its input.

1.2. Related work and overview of our objectives

Our experimental setup comprises several subtasks like the identification of persons in the scene and the corresponding tracking, the differentiation between the predominant, active and negligible persons in the environment and, finally, extracting the silhouettes of people performing meaningful body gestures. All of these extracted tasks have been addressed in the computer vision community, which is why we would like to separate related work in the specific areas highlighting both the advantages but also downsides of standard and recent approaches, culminating in a motivation of our proposed approach.

1.2.1. People detection and localization

The detection and correct localization of people in visual data has traditionally been approached using features such as Haar wavelet templates [38], Histograms of Oriented Gradient (HOG) [8] and Histograms of Flow (HOF) in combination with HOG [57] on usual camera images. The release of affordable devices like the Kinect delivering depth images provided a simplified approach to segmentation tasks and influenced the creation of datasets in recent years [7,61]. Consequently, standard features have been extended to the depth channel as well like the Histograms of Oriented Depth (HOD) [48]. Specifically for robots, using color and depth channels (RGB-D) for moving robots were shown to add substantially to successful tracking when numerous persons are present [34].

A particular challenge in person detection and localization is posed by unconstrained environments (i.e. when no depth images are necessarily available) and considering multiple people or even crowded scenes as in pedestrian streets or on a university campus. A recent study [31] on the issues involved in the task, e.g. occlusions, proposed a trajectory model based on a multi-label conditional random field (CRF), a probabilistic graph model assuming nodes belonging to either a specific class of superpixel or background, connected by a set of edges defined by a certain neighbourhood. The strength of the proposed model is that the integration of a varying number of persons entering in the scene is allowed, regularized only by preferring lower numbers of (possible) people to circumvent the number of misclassifications. However, the construction of bounding boxes in combination with a nontrivial, energy-based optimization procedure on the graph model makes this approach less flexible for a bottom-up approach extending people movements to be classified into e.g. actions.

With the availability of both data and computing power, CNNs [28] gained increasing popularity in the vision research community. A lot of studies employing CNNs were also applied for pedestrian detection and used popular datasets [8,10] containing images from traffic scenes with cars and pedestrians and showed competitive performance compared to conventional feature-engineering approaches [45]. However, the research presented so far focused on a robust detection and a meaningful segmentation challenged by an unconstrained environment, thus did not take into account specific body actions or salient movements. In the following, we will outline approaches addressing the challenge of detecting and classifying particular movements among multiple persons in a meaningful way.

1.2.2. Motion of multiple people

The differentiation between random movements and meaningful gestures or actions is already challenging when scenarios are designed for only one subject. Typical processing steps before arriving at descriptive features include skin color modeling [22] or motion analysis [58]; for an overview see [32,41]. The issues of learning color or motion models increase for multiple person setups including performance variability, which explains the sparsity of gesture studies involving multiple people. Widely used datasets (see [43] for an overview) contain scenes with only one person and the resulting studies implicitly assume that the input to their system only includes a single person. A direction towards multiple person action recognition has been made with the release of the *Pose Track* dataset [17] comprising sequences of team sport activities.

As pointed out in the previous section, deep learning approaches have successively complemented or even substituted traditional feature-engineering approaches in the recent years. They showed comparable to superior performance in many benchmark learning tasks, especially for computer vision related topics, e.g. the *Chalearn* benchmark [11] for co-speech gestures [36] or the *KTH* dataset [44] for actions [1,20,60]. In addition, studies investigating the potential of a multi-channel CNN employing both 2D and 3D kernel [2] demonstrated good performance on a common dataset with high subject variances in gesture performance [53] combined with an analysis on a newly recorded dataset from a humanoid robot. Interestingly, the evaluation revealed that the performance was not much affected by using either the 2D or 3D kernel. In a further analysis on that dataset, it was even shown that less complex learning employing sparse autoencoders achieved similar performance [21], thus it can be concluded that the strength of the CNNs is rather based on using multiple input channels for boosting the feature representations.

A research area where the constraint on a single subject dramatically limits the number of sensible applications is Human-Robot Interaction (HRI). For a robot acting in everyday scenarios, e.g. in an office [33] or a museum [5,50], it is important to select one person as the interaction partner. In related studies, high-level knowledge about people in the scene was computed, e.g. face position and size, distance to the robot and the time elapsed since someone had last spoken. Based on this knowledge, the decision on the active person was then performed using some heuristic, e.g. a social distance score [51]. These studies evaluated their systems based on qualitative observations and user feedback but did not provide a quantified performance measure. Also, while taking cues such as audio volume into account, none of these proposed systems took into consideration if a person was actively trying to communicate with the robot. As these approaches do not work on the low-level signal data directly, they compute high-level knowledge about face locations or similar concepts that might not be relevant at all if an application only focuses on one person in the scene. In contrast, our approach focused on localizing the person who was actively addressing the sensor. Also, our evaluation used the low-level signal directly and we report the system performance using a quantitative metric.

Another important factor we suggested already is the role of attention, also called saliency. The latter term refers to the physiological response of the human eye when a notable stimulus hits the retina, for instance, a light stimulus or when an object is moving across our field of view. This way, we become aware of a particular *event* in our world and do not steadily parse our environment. That such a mechanism is beneficial for multi-person actions was pointed out by Ramanathan et al. [40] ruling out ineffective information by introducing time-varying attention weights learnt by recurrent neural networks (RNN). For an evaluation of their proposed architecture for detection and event classification, the authors [40] presented sequences of a basketball game, where naturally the roles of individual players change over time. In line with the argumentation that saliency characteristics in a scene can

effectively be used for multi-person settings, we proceed summarizing studies on this mechanism in the next section, and explain our access to this topic in the context of our implementation.

1.2.3. Saliency

In research studies on saliency, we identified related approaches that determine the relevant parts of the visual data directly on the low-level signal. Saliency is a broad term that describes distinct elements in some data prominently differing from the rest of the data. Many saliency studies look at human eye tracking data to compare the results of their computational systems, for an overview see the MIT saliency benchmark by Bylinskii et al. [6]. Traditional saliency and visual attention models used features known from neurobiological research in the actual human visual system [18,25,54]. According to [6], the most accurate computational saliency models at the moment are, again, Deep Learning architectures such as by Kruthiventi et al. [27] who use CNNs. These approaches attempt to reproduce human gaze movement and evaluate their results for the benchmark on static images. However, the world we experience is not static, and motion has in fact been shown to prominently influence and attract our attention [55]. Some studies even suggest that specific human motions like body gestures direct our attention [37]. However, [15] dispute this fact and attribute the observation of [37] to their specific experimental setup. In a more general sense, [15] conclude that the way humans attend to gestures is heavily influenced by their social context.

As saliency is also used as a broader term to describe distinct parts of visual data, some studies have looked at specific scene setups where a certain kind of movement is considered salient. Riche et al. [42] computed optical flow features from RGB-D data and identified image regions moving differently from other motion in the scene. These were then interpreted as salient motion. Their experiments included a scenario with multiple people moving in a scene, with one person moving into a different direction than the others. However, their system did not distinguish between different kinds of motion. Also, their approach relied on optical flow features and needed preprocessing for perspective corrections and a manual fusing of several saliency maps.

In [39], only the salient movement of one hand of a person was used for the task of gesture recognition. Other motion information was discarded with the motivation that the dominant hand will attract the attention of the observer. While this approach could potentially be extended to multiple people, their approach made use of the Kinect skeleton data and the extracted hand position. This limits the number of people who can be present in the scene and will also fail if a person is occluded in a way that no skeleton data is available. For our implementation, we envisaged a scenario where particular attention is given to the subject performing a gesture while other, low-level or random movements are considered obsolete and filtered out, i.e. the saliency cue is a specific gesture. In contrast to [42], our system was trained end-to-end and did not require any processing steps. Also, different from the experimental settings proposed in [39] we used the low-level signal depth data directly and designed the setup from the start to work with multiple people in the scene.

1.2.4. CNNs for localization tasks on sequential input data

Convolutional Neural Networks (CNNs) [28] are a specialized kind of feedforward neural network, specifically designed for visual processing tasks. A typical CNN is often designed to take a single image as its input (one grayscale channel or multiple color channels), which is fed into a cascading system of filters convolved with the image (often called receptive fields in analogy to the human retina) and a pooling mechanism downsampling the image over the different layers. A final hidden layer in the network produces a vector of class scores as its output. The output layer then has one output neuron for each class of the dataset.

The interesting element in CNNs and key to their success [26,28] is that the employed filters are not specified in advance as, for instance, in standard convolution masks for edge extraction in an image [19]. Instead, they are learned based on the input data and the hierarchical computation across the network realizes the extraction of characteristic, composite and invariant image features.

Based on their growing success in image processing, CNNs have also been applied to localization tasks on images by using a sliding window approach [45] or region proposals [12,13]. An alternative approach is the semantic segmentation of an image so that each pixel is assigned an object category [29]. The CNN then produces a 2D output, which is the same output format we chose for our architecture.

While originally introduced for single image processing, CNNs have also been applied to tasks with video sequences as the input data, for example video classification [23,52], action recognition [1] and gesture recognition [3]. One approach to use CNNs on videos is to transform a frame sequence into a single frame that somehow contains the motion information, e.g. differential motion images [3]. The CNN architecture can also be set up in a way to take a sequence of multiple frames as its input, without the need for such a preprocessing step. As shown in [20], 2D convolution kernels can be extended to an additional dimension. These 3D convolution kernels perform a cubic convolution and can, therefore, extend into the temporal domain when using a sequence of frames as the input. The filter kernels then learn spatio-temporal features which have improved the performance of classification tasks on videos [20,23]. In our proposed architecture, we also made use of 3D convolutions so that we could directly input a sequence of frames.

1.3. Objectives

The objective of the present study originates from the motivation to develop a system applicable in domestic or health care domains for sensible interaction scenarios, where naturally multiple persons occur in the scene. Our goal is to design a scenario, where an active person can be localized and distinguished from other people from a data stream. *Active* is defined as a person performing a gesture, the salient cue, which inevitably needs to be correctly identified as a meaningful movement. As we described in the previous sections, CNNs have been shown to robustly process images and videos while overcoming image preprocessing techniques commonly used in computer vision. In addition, we pointed out prominent and recent datasets on some of the subtasks but none of them satisfied our needs as their application goal differed each. Therefore, we introduce a novel dataset defining useful dynamic command gestures convenient in HRI scenarios. To capture the dynamics, we employ a 3D CNN to localize a gesture movement. We interpret our proposed system as a saliency mechanism on image sequences that reacts to body motion cues which have not been included in previous computational saliency systems yet. We designed our network architecture in a way that it outputs a 2D representation of the scene so that it can easily be interpreted by a human observer and be used for future applications. Finally, we aim at analyzing different network architectures for our scenario and, therefore, compare our favored 3D CNN with a baseline 2D CNN and the popular VGG-16 network.

2. Methods

As described in the previous section, no dataset for localizing a gesturing person within a multi-person scenario exists. Therefore,

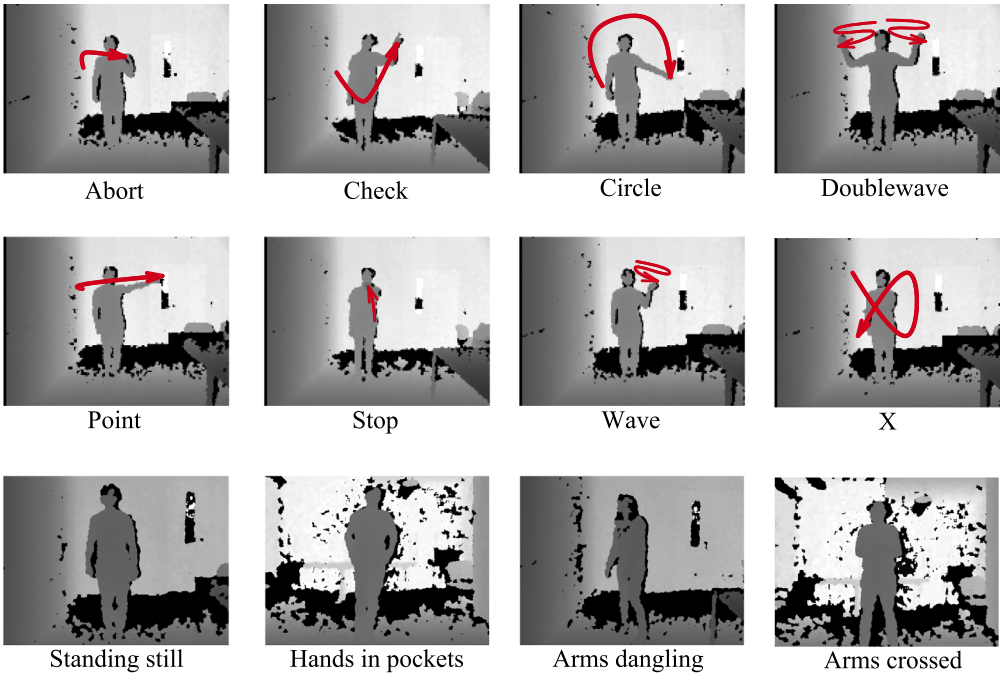

Fig. 1. The gesture set we developed in our study. We recorded 8 different command-like gestures characterized by specific arm movement. Every gesture is shown with a line indicating the hand movement. The last row shows four examples of *Passive* recordings, cropped for better visualization. Passive recordings showed a variety of orientations towards the sensors and different arm positions.

our experimental design comprises some constraints on the number of subjects and gesture types to decrease confound variables, which allowed us to reasonably evaluate our proposed neural architecture with an integrated saliency mechanism. At the same time, we implemented the data collection introducing a data augmentation procedure that offers the opportunity for scene adaptation regarding the number of samples in general, but also the number of persons and positions as well as an extension on the gesture vocabulary. In this section, we will describe how we collected such a dataset. We will then continue to present our proposed 3D CNN architecture that performed the detection and localization task on our data.

2.1. Dataset

To evaluate a scenario with salient movement and multiple people, we designed an experiment with multiple people in a scene and one of them performing a dynamic body gesture. We defined the latter to be the *active* person, positioned at random locations in the room. As CNNs typically require large and well-prepared datasets to achieve good results [26], we decided to record clips with one person and combine these recordings systematically into multi-person sequences to be used for training. We recorded RGB-D data with a Microsoft Kinect and used the depth channel to create artificial multi-person scenes. The advantage of using an RGB-D sensor is the availability of the depth signal. Larger values in the signal mean a larger distance to the sensor. This allowed us to conveniently combine multiple recordings so that the result looked like an actual recording from a Kinect sensor. In contrast to an RGB signal, where slicing and stitching rectangles produce visible and unnatural edges, the available depth signal allowed to mask the body shape of the actor and only use the signal values inside that mask (Fig. 3).

During data collection, we recorded active gesture executions and passive body positions. In these recordings, a single person was placed in front of the Kinect sensor in the center of a large room, so as to have a recording available where the person could easily be cropped without any furniture causing problems. Additionally, we recorded a number of empty rooms that were used as background frames for the generated sequences.

The gesture set the participants performed were eight command-like body gestures. All of these contained distinct arm movement (see Fig. 1). The participants also performed passive body positions, such as standing in different angles to the sensor while having the arms hanging down, having the arms crossed or slightly moving the legs, but not moving the arms distinctively (see last row of Fig. 1).

While every gesture execution of the same person was different (intra-subject variability), the difference was a lot larger between multiple people (inter-subject variability). In order to have a larger variety in the recorded gesture executions, 6 people were recorded, each performing the gestures listed in Fig. 1. One person was female, five were male, and the age ranged from 25 to 34 years. Each gesture was performed multiple times by each subject. In total, all participants combined performed 568 instances of gestures and we had 43,034 frames available for the dataset generation. These frames were labeled with bounding boxes of the full body and bounding boxes of the head (Fig. 2) using the Vatic video annotation tool by Vondrick et al. [56].

With the recorded and labeled data, it was then possible to generate sequences according to any desired scene description. Along with these sequences (our data samples) we also generated a teacher frame (our data labels). This 2D frame represented the locations of people in the scene and showed the active person as colored in white. How we generated these frames is described further below. The underlying strategy to generate a sequence was isolating a person's silhouette from the depth channel of a recording and placing it at a random position on top of a background recording, see Fig. 3. We performed this random placement for multiple people according to a number of scene setups. We generated more scenes of multiple people than empty or single-person

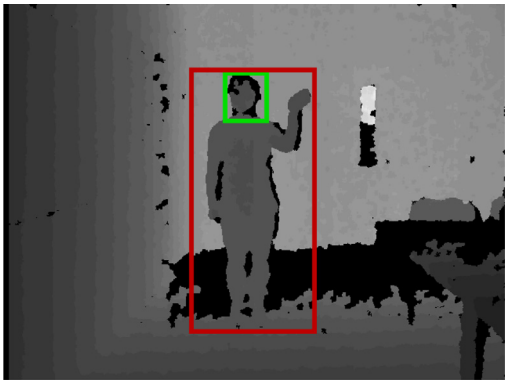

Fig. 2. All recorded frames were annotated with bounding boxes for the body (large red square) and for the head (small green square). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

because our focus was on scenes where both active and passive people were present.

1. An empty room without people: 10% of all sequences
2. A single person performing a gesture: 20% of all sequences
3. A single person performing no gesture: 20% of all sequences
4. Two people, one of them is active: 30% of all sequences
5. Three people, one of them is active: 20% of all sequences

Sequence length. We generated every sample of the dataset as a sequence of 15 consecutive frames. With a frame rate of 30 fps, this length corresponds to half a second from the recorded gesture performances. 15 frames was a length that would result in sequences containing some arm movement, no matter from which gesture recording the frames were taken. Our recorded gestures had a large variety in length (see Table 1), but the task of our network was not to classify a gesture. It therefore did not need to see a full gesture execution in the dataset samples. We concluded that showing some arm movement in every sequence would be enough to determine the salient movement in the scene.

Training, validation and test set. We combined the recorded gestures in a way that would allow a clean separation in three subsets for training, validation and testing. The network should not have the possibility to memorize a specific data sample from the original recordings to increase its performance on the generated sequences. To achieve this clean separation, the dataset generation was designed to produce 5 different small subsets (or folds). Each fold used an entirely different collection of recorded frames so that there was no overlap across these 5 folds. The network could then be trained using three folds as the training set, and one fold each for validation and testing. Each fold was set to contain at least 512 sequences in total (512 being a power of 2 makes for good handling of batch sizes during training). Our script always created sequences in groups of 10, so instead of 512 sequences, each fold then contained 520 sequences. With all 5 folds combined, the total dataset contained 2600 sequences (of 15 frames each), which we considered a reasonable size while keeping computation time manageable. Both the sequences and the teacher frames were saved as PNG files in a resolution of 320 pixels by 240 pixels.

Ground plane removal. When isolating the silhouette of a person in the depth channel of the Kinect data, we found it beneficial to remove the ground plane before slicing the person out of its surrounding (Fig. 4). This produced cleaner silhouettes, as there were fewer pixels of the floor left that would otherwise create a false silhouette, in particular around the feet of a person. In our recordings, the rough camera position was known since the source recordings were created under controlled conditions. We

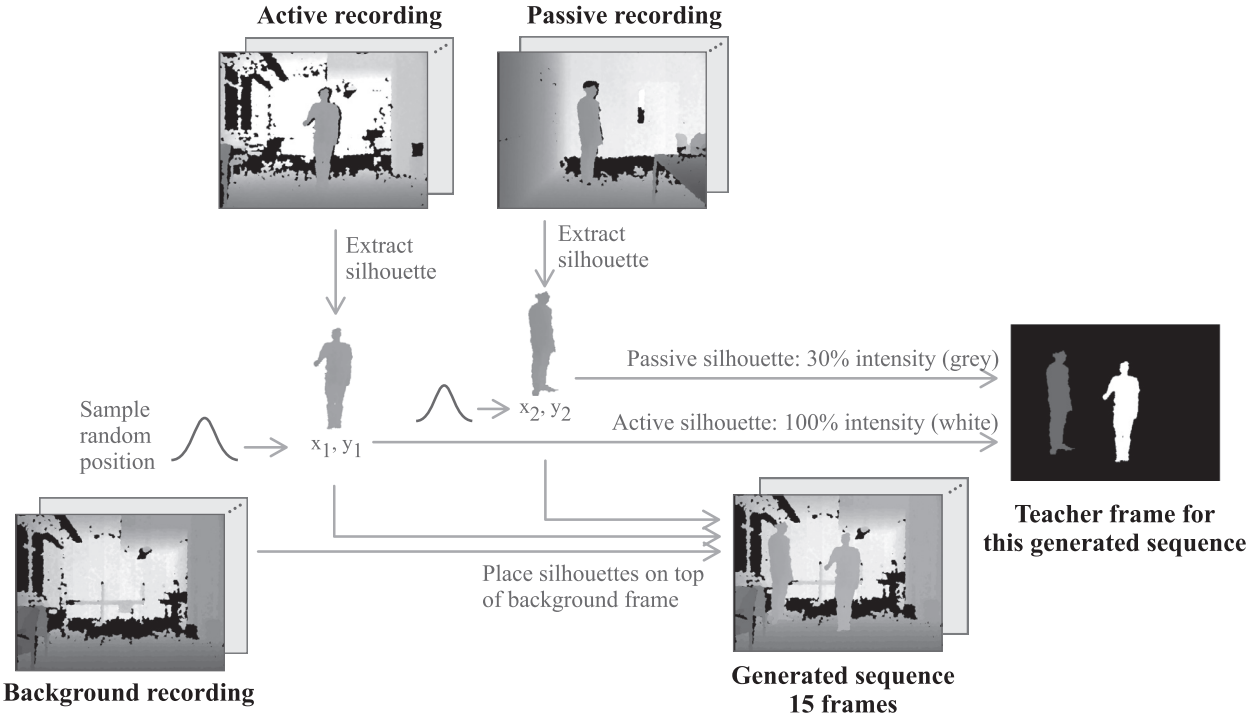

Fig. 3. We generated samples for our dataset by combining the depth frames from several recordings into sequences containing multiple people. This figure shows the procedure to generate a sequence containing 2 people, one of them performing a gesture. For each frame in the sequence, the silhouettes of each person are extracted (according to bounding box annotations) and placed on top of a background frame. Additionally, we generated a teacher frame for each sequence which represents the locations of people in the scene and marked the active person using a white silhouette.

Table 1
Overview of recorded gestures. Each instance refers to a person performing a gesture from the start frame to the end frame. Different gestures had varying average execution times. Short gestures were performed more often by the subjects, which is why there are different instance counts for different gestures. Having multiple people perform these gestures caused deviations in the average lengths. Passive recordings were significantly longer because in contrast to gesture recordings they were not split into multiple shorter instances.

Gesture	Instances	Average number of frames	Total number of frames
Abort	62	52.3 ± 7.8	3243
Check	66	55.2 ± 14.8	3646
Doublewave	66	73.9 ± 10.3	4876
Circle	72	62.1 ± 15.3	4473
Point	110	55.7 ± 8.0	6125
Stop	53	53.9 ± 7.1	2858
Wave	62	61.7 ± 14.5	3824
X	63	66.9 ± 8.2	4212
Passive	14	698.4 ± 260.3	9777
Σ	568		43,034

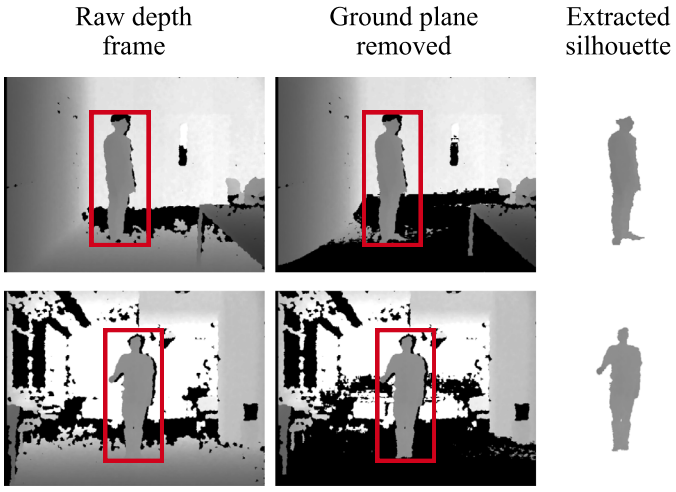

Fig. 4. Person extraction from a single frame. The red rectangle shows the bounding box that we annotated for each frame (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

could therefore assume a straight ground floor and a straight orientation of the camera which made the ground floor appear as a linear gradient in the depth signal. To remove the ground plane, we assumed all pixels in a close range to this gradient to belong to the floor and masked out the matching values. This simple step significantly improved the silhouette extraction of our approach, both when stitching together multiple recordings and also when generating the teacher frame.

Data augmentation. The sequence generation itself could already be viewed as a data augmentation method, as we could produce any number of dataset samples from the limited number of recordings. Randomly placing people in the scene and combining active and passive recordings produced a large variety of generated sequences. Additionally, we changed the brightness of each person before placing them inside the scene. This change in brightness simulated a different position – closer or further away from the sensor and was sampled from a Gaussian distribution. During training of the networks, we used additional spatial augmentation techniques, as we will describe in Section 3.2.

Teacher frame generation. To use the dataset for supervised learning, we assigned a label to every generated sequence. This label was a frame we called the teacher frame. It showed the locations of all people in the scene and encoded who the active person is. Each person was marked using their silhouette. Every silhouette was colored in either white (active person) or a darker tone of gray (passive person), similar to how [4] encoded neutral and positive emotions in their work (in a ratio of 1/3 to 2/3 in their experiments). As the intention of generating this frame was the training of the network, it was called the teacher frame (see Figs. 5 and 6). As every sequence consisted of 15 frames with movement in them, we picked only one of these to extract the silhouettes: the last one. The details inside each silhouette were ignored. Instead, the complete body shape was assigned a solid color. Every pixel of the frame was supposed to encode the pixel's content: is this an active person, a passive person or a background pixel. The brightness of a silhouette could have two different values, depending on the person performing an active gesture or being passive in the scene. An active person was assigned the full brightness (gray value 255), and a passive person 30% of the full brightness ($0.3 \cdot 255 \approx 76$). Background pixels were generated to be black.

2.2. Network architectures

We will now describe the design of the two network architectures used in our study. After looking at the required input and output format, we present the two CNN architectures, one that uses 3D convolutions and a standard 2D CNN that was used for performance comparison during evaluation.

Network input. Our main architecture with 3D convolutions received sequences of a fixed length of 15 frames as its input, with each input frame being 160 pixels wide and 120 pixels high. To match this input size from the original 320 pixels by 240 pixels resolution, we performed down-scaling and cropping, as we will describe in Section 3.2. Our comparison architecture with 2D convolutions used the same frame format but only received a single frame as its input.

Network output. Both architectures produced a two-dimensional frame as their output, with low values (dark) representing background and high values (bright) representing the people in the scene. The units of the output layer produced the single pixels of this output frame. Choosing a size for the output layer meant a trade-off between visual descriptive power of the output and the computational cost of training the network. We decided on an output dimension of 80 pixels in width and 60 pixels in height. This is exactly half the width and half the height of the input and a size where a human observer can still clearly identify the single people in the scene. When receiving an output from the network, each pixel value was a float value, mostly in the range of -1 to $+1$,

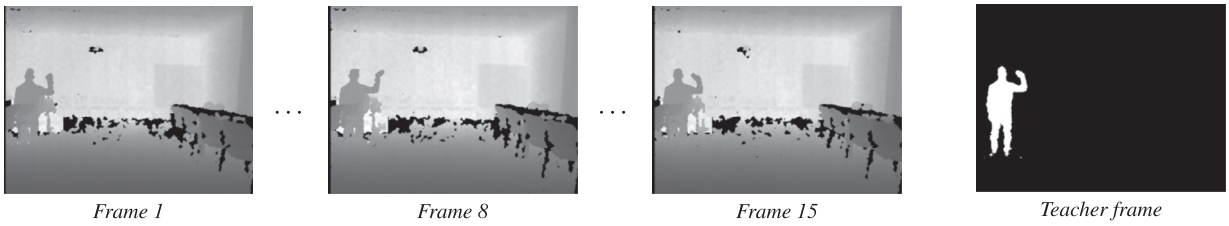

Fig. 5. A generated sequence with a single actor performing the *Wave* gesture. Even though the person was recorded in the center of a room, the generated sequences contain a large variety of generated positions. Here, the person is placed on the left side of the frames.

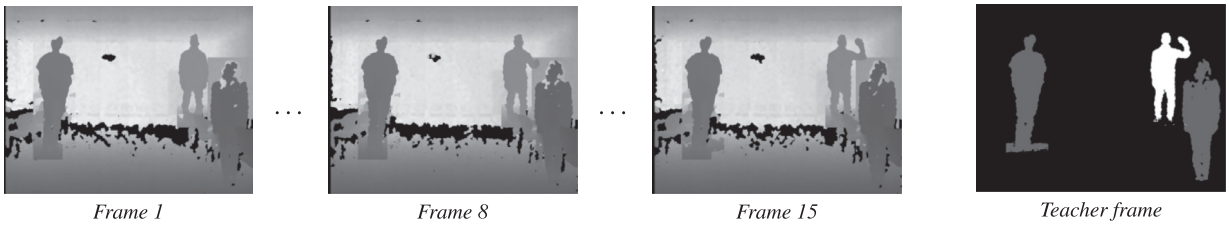

Fig. 6. A generated sequence with an actor performing the *Wave* gesture and two passive people in the scene.

but also having a few activations outside this range. To store these images for visualization, the pixel values were shifted and rescaled linearly to fit in the 8-bit range from 0 to 255, with values outside the range being mapped to 0 or 255.

Regression task. To perform the task of producing a 2D output frame from the extracted features, we used a Multilayer Perceptron (MLP) with one hidden layer. Its output layer needed one unit per pixel, which resulted in $60 \cdot 80 = 4800$ units in total.

Training. The regression task of the network was trained by the minimization of a Mean-Squared-Error loss function. Weights were updated using a gradient descent strategy with the Adam Optimizer algorithm [24]. To reduce overfitting, we used Dropout regularization [49] after the hidden layer of the MLP.

3D-Net architecture. Our main network had three convolutional layers with 3D convolutions. Every convolution was followed by a ReLU activation [35] and a max-pooling layer. The activation maps from 3D convolutions also had an additional dimension, so that the max-pooling operation also was a 3D max-pooling operation. The feature extraction block was followed by an MLP to perform the regression task. It had one hidden layer and one output layer. We refer to the complete architecture as "3D-Net".

2D-Net architecture. As a performance comparison, we chose a CNN architecture that did not take the temporal information into account, but only worked on a single frame input. From the general architecture, it was the same as the previously described 3D-Net, with the main difference that it used 2D convolutions instead of 3D convolutions. The input to the first layer could therefore not be a sequence of frames. Instead, only the last frame of every sequence was considered as the input to the network. We refer to this architecture as "2D-Net".

Implementation and libraries. We collected the Kinect recordings using OpenNI 1.5.x.[2] These were converted to regular videos using the open source tool oni2avi.[3] We annotated these videos using the browser interface provided by Vatic [56]. Sequences and teacher frames were generated with a custom Python script using multiple open source libraries: Open CV[4], Numpy[5] and SciPy.[6] We implemented our architectures in Python using TensorFlow and Keras and trained them on Nvidia GPUs GTX 1060 and GTX 1080Ti. Code of our architecture implementations and training experiments is available online[7], as is our dataset.[8]

3. Experiments

To evaluate how well the proposed architectures were able to learn the desired saliency mechanism, we conducted a number of experiments. We trained both architectures with the generated multi-person dataset and evaluated on the test samples from that dataset. To compare the results in a quantitative manner, we used a metric to describe how well the gesture movement was detected and localized. In this section, we describe the parameters of the two networks, the training procedure and the motion detection metric used for evaluation.

3.1. Network configurations

Both architectures 2D-Net and 3D-Net were structurally very similar, with the main difference being the use of 2D and 3D convolutions. The actual hyperparameters of 3D-Net (Table 2) were the result of a hyperparameter optimization experiment. We iterated over different convolution filter numbers (4, 8, 16 for layers 1, 2 and 3 compared with 8, 16, 32 and 16, 32, 64) numbers of hidden units (384, 768 and 1536) and dropout rate during training (0.25 and 0.5). The best configuration for 3D-Net was determined by the best validation score achieved during training. We then chose the same configuration for 2D-Net, so that the only difference between these two architectures was the use of 2D convolution instead of 3D convolution.

[2] Github: OpenNI. https://github.com/OpenNI/OpenNI, 20th June, 2018.

[3] KirillLyko: oni2avi. Command-line converter from oni (OpenNI) to avi data format. https://github.com/KirillLykov/oni2avi, 20th June, 2018.

[4] Open CV computer vision library. http://opencv.org, 20th June, 2018.

[5] Scientific and numerical computation package for Python. http://www.numpy.org, 20th June, 2018.

[6] Scientific computing tools for Python. https://www.scipy.org, 20th June, 2018.

[7] Architectures and training code available at https://github.com/florianletsch/publication-salient-body-motion.

[8] Dataset available at https://www.inf.uni-hamburg.de/en/inst/ab/wtm/research/corpora.html.

Table 2
Hyperparameters of our two architectures. 3D-Net was the main architecture in our experiments and used 3D convolution kernels. 2D-Net only used 2D convolution kernels and was used as a performance comparison.

	3D-Net	2D-Net
Convolution kernel dimension	3D	2D
Conv Layer 1	16 filters @ $3 \times 3 \times 3$	16 filters @ 3×3
Conv Layer 2	32 filters @ $3 \times 3 \times 3$	32 filters @ 5×5
Conv Layer 3	64 filters @ $3 \times 3 \times 3$	64 filters @ 3×3
Max Pooling	$2 \times 2 \times 2$	2×2
Activation function after convolution	ReLU	ReLU
Hidden units	384	384
Input size	$120 \times 160 \times 15$	120×160
Output size	60×80	60×80

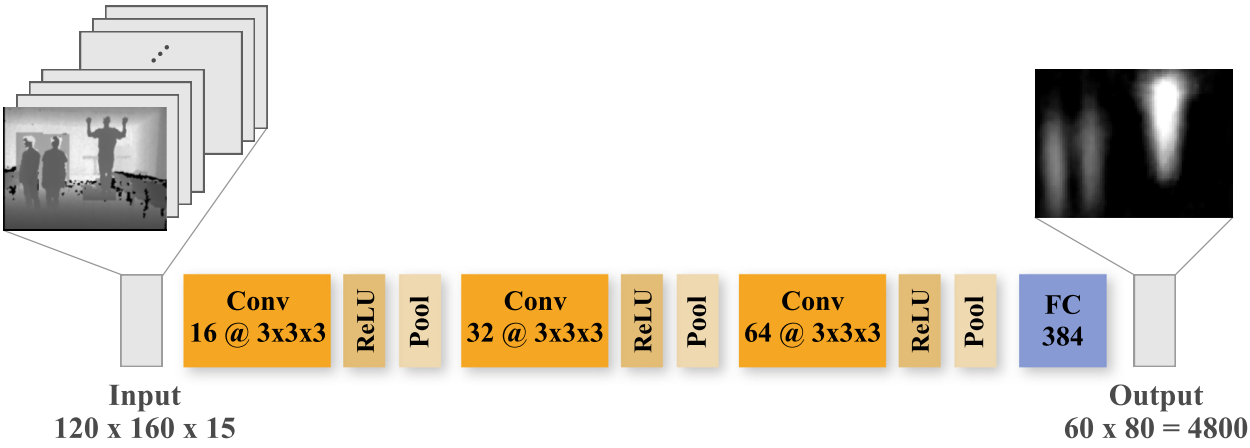

Fig. 7. The main 3D-Net architecture. It received 15 depth frames of 120×160 dimensions as its input. The architecture had three convolutional layers with 3D filter kernels, ReLU activations and max-pooling. The MLP after the feature extraction block has used one hidden layer and produced a 60×80 frame as the output.

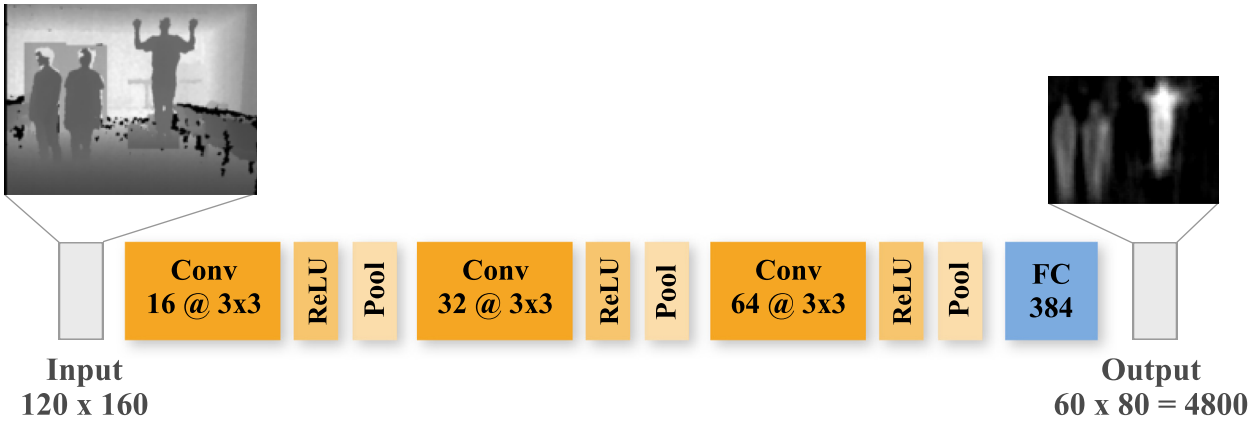

Fig. 8. Architecture used for performance comparison. The 2D-Net received a single 120×160 frame as its input. The architecture had three convolutional layers with 2D filter kernels, ReLU activations and max-pooling. Just like the 3D-Net architecture, the MLP following the feature extracting had one hidden layer and produced a 60×80 frame as the output.

3D-Net. Our main architecture (Fig. 7) had three convolutional layers (8, 16 and 32 filters of dimensions $3 \times 3 \times 3$) with ReLU activations, each followed by $2 \times 2 \times 2$ max-pooling. After the convolutional feature extraction part, we added an MLP with a hidden layer of 384 units, on which we applied dropout regularization during training.

2D-Net. Our baseline architecture (Fig. 8) had three convolutional layers (16, 32 and 64 filters of dimensions 3×3) with ReLU activations, a hidden layer with 384 units, used 2×2 max-pooling after each convolution layer and dropout regularization during training. The dropout layer was located after the hidden layer.

3.2. Preprocessing

To match the network input size of 160 pixels by 120 pixels, the dataset frames were resized from their original 320 pixels by 240 pixels resolution to be 192 pixels wide and 144 pixels high. From these downsized frames, random crops of 160 pixels by 120 pixels were taken during training and fed to the network. This effectively augmented the dataset with different versions of

the training samples. For validation and test time, instead of taking a random crop, we cropped the same dimensions from the center of each frame.

The configuration we used for scaling was the result of additional experiments, where we compared scaling by factors 0.9, 0.8, 0.7, 0.6, and 0.5. The best validation accuracy was achieved with a scaling factor of 0.6, which was why we scaled from 320 pixels by 240 pixels to 192 pixels by 144 pixels.

Before passing a frame to the networks, we normalized all depth values to the range of −1 to 1 to unify the input values and center them around zero.

3.3. Training

The weights of both networks were initialized using the Xavier initialization method [14]. Training was then performed by minimizing a Mean-Squared-Error (MSE) loss function. Weight updates were performed using the Adam Optimizer [24] with a learning rate of 0.001 and a learning rate decay of 0.9, with mini-batch training and a batch size of 4.

We trained the networks 3 times each to compute the average performance scores on the test set. 2D-Net and 3D-Net were both trained for 50 epochs, before calculating the final score of each model instance on the test set. During training, the dropout probability was set to 50%. For evaluation, we set the dropout probability to 0 to disable dropout.

3.4. Comparison with VGG16

As a comparison to 2D-Net and 3D-Net, we performed experiments using VGG16 features. VGG [46] is an architecture designed to perform classification or localization on RGB datasets like ImageNet [9]. The network expects an input of a minimum frame size (larger than 48 by 48 in the implementation which we used, which was provided by Keras[9]) and 3 input channels. To use this network on our single-channel data, we fed in three identical copies of the channel into a batch normalization layer [16]. We added this batch normalization layer, so that the network could potentially learn a mapping from the depth data of our dataset into a space more similar to RGB characteristics. The output of that layer was then fed to the VGG network. We removed the last layers of the network originally designed to perform classification and only used the VGG16 feature extractor. Instead, we added an MLP to the end of the network just like we did on 2D-Net and 3D-Net (384 hidden units). We performed experiments with this network initialized from scratch and experiments with the VGG16 features pre-trained on ImageNet. In the latter case, the feature extraction layers were frozen during training and only the initial batch normalization layer and the final MLP layers were trained.

3.5. Motion detection metric

To measure if a network correctly detected the active gesture performer, we defined a metric to compare a teacher frame from the dataset with the two-dimensional network prediction (Fig. 9).

The teacher frames in the dataset encoded the active performer as having a white silhouette. A correct network prediction should also have its brightest value inside that silhouette. The basic idea of the metric is to simply look at the location of the brightest pixel in the network output and compare if this location is a white pixel in the teacher frame (Fig. 9). However, due to re-sampling of the original teacher frame, it consisted of more than just black (background), white (foreground) and 0.3 brightness (passive performers). Therefore, we did not check for white explicitly, but instead

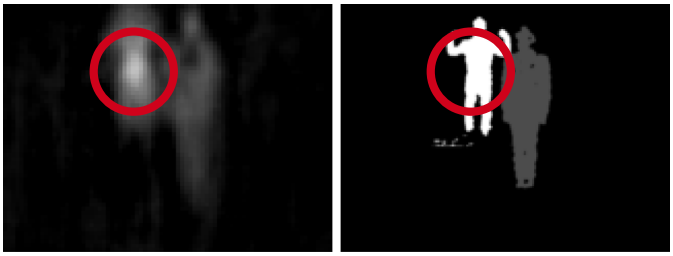

Fig. 9. The motion detection metric compared a network prediction (left) with the corresponding teacher frame (right). We looked for the brightest pixel in the prediction (circled in red) and checked if the teacher frame also had a bright foreground value at the same location. That was the case in this example and the metric returned True. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

compared the value with a threshold of 0.3. A value above that threshold was considered bright enough and the function returned True, otherwise it returned False. The threshold of 0.3 was chosen because it was the brightness for a passive person. Anything above that threshold was considered active.

Some sequences did not contain an active gesture performer and the teacher frame therefore did not have any non-black pixels. In this case, we checked if the network prediction included any brightness value above the threshold of 0.3. If that was the case, the network had incorrectly detected a gesture and the metric returned `False`. If all pixel values were below that threshold, the metric returned `True`.

For the comparison of two frames, the metric returned either `True` or `False` as a decision if the output was correct. This correctness metric was then used across the whole test set to calculate a motion detection score of the network.

$$A_{\text{motion}} = \frac{\text{Number of correct results}}{\text{Total number of samples}}$$

To show how well a network had learned the motion detection task, we determined a baseline score that would occur when performing random guesses. Guessing a pixel location in a teacher frame would lead to a positive result if it was inside of an actual bright silhouette. We computed the random guessing baseline for the motion performance metric to be 5%. This value is the ratio of white to black pixels over all teacher frames from the test set. Randomly setting a pixel to be white would therefore be correct with this probability.

4. Results

We first evaluated the overall performance of all network architectures described in the previous section using the motion metric A_{motion} on the test set. Apart from the baseline performance of randomly guessing the *active* persons' location, the VGG network achieved the lowest performance with an accuracy score of only 0.30, followed by the VGG pretrained network with 0.54, as demonstrated in Fig. 10. In contrast, both the 2D and the 3D CNNs showed superior performance with surprisingly similar scores of 0.78 and 0.81, the latter having a slightly higher standard deviation.

The results reflect our experimental observations. The VGG network did not converge fully on our dataset. This network was specifically designed for RGB data, and appears oversized for the simple characteristics of depth data and the comparably small size of our dataset when considering that VGG was originally trained on ImageNet. Even though we used dropout regularization and spatial data augmentation, this network overfit heavily on the training data. When using the pretrained VGG16 features,

[9] Keras Deep Learning Library, https://keras.io/.

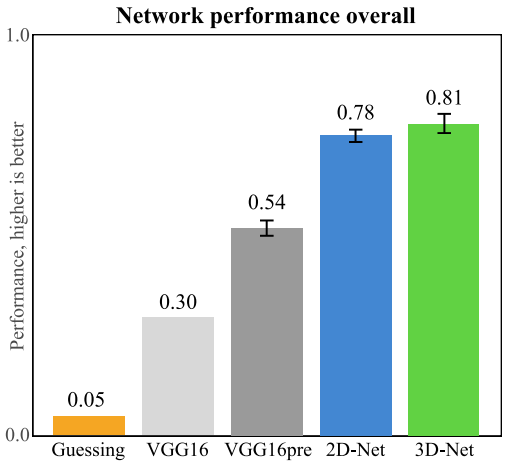

Fig. 10. Performance scores of the evaluated networks. The motion metric describes how well the networks were able to correctly detect and localize the *active* person performing a gesture.

the results were better which may show an advantage of transfer learning. However, transferring RGB data to our depth data has only limited success. Still, the dataset characteristics were better learned by our two custom CNN architectures 2D-Net and 3D-Net, which were trained from scratch, and both achieved significantly higher scores than VGG.

Based on these findings, we concluded to investigate further the potential of 2D and 3D CNNs for our defined gesture vocabulary. To compare the two networks' performance scores in a more fine-grained manner, we split the test set into multiple subsets, each containing only sequences of the same active gesture. This produced 8 subsets for each gesture plus two subsets of sequences with empty scenes and scenes with only passive people without any gesture performance.

Fig. 11 depicts the performance scores of the 2D and 3D CNN architecture on the different subsets. Notably, for most of the gesture types the scores are greater than 0.9 for both architectures, showing again similar results between the 2D and the 3D kernel. In case of empty scenes, both networks achieved perfect recognition with a score of 1.0. Our results also highlight two outliers of remarkably lower scores: the *Stop* gesture and samples with only passive people in the scene. When analyzing this, we realized that teacher frames of the *Stop* gesture had sometimes been generated incorrectly, so that some frames did not contain the human silhouette, but only a tiny silhouette of the hand (Fig. 12). This was caused by the hand being a lot closer to the sensor than in other gestures. When the teacher frames were generated, the silhouette was then generated incorrectly and ended up both in the training and test data. The metric compared these ill-generated frames with the network prediction and therefore produced low-performance scores for these sequences. When investigating why scenes of passive people showed very low performance, we noticed both networks often assign intensities to the passive person that were just slightly higher than the 0.3 threshold the metric was checking for. Even though in these cases, the silhouette was still a lot darker than the full 1.0 intensity, the metric punished these results as being wrong (Fig. 13). For both these outliers, the 3D-Net showed higher performance than the 2D-Net. In the passive scenarios, it also showed a much lower standard deviation. This shows how the 3D-Net was systematically better in distinguishing between an actual active person and a person who just appears more active in relation to other people in a frame.

These fine-grained experiments revealed that the two outliers explain why the overall motion detection performance shown in Fig. 10 was around 0.80 for both networks, even though all other gestures showed scores in the range from 0.90 to 1.00.

Regarding the different gesture types evaluated on the 2D-Net, our results revealed better performance for gestures with characteristics that were visible from still frames. This included the gestures *Abort*, *Check* and *Circle*, all containing distinct body postures with the arms raised upwards, as shown in Fig. 14.

In contrast, the 3D-Net showed higher performance for gestures with more varied arm positions, for example the *X* gesture which contained the drawing of a symbol in the air with a sequence of several strokes. This motion includes lower arm positions (Fig. 15) that might appear like a passive position from a single-frame perspective. As the 3D-Net received the full sequence as its input and its features are computed across an image stack, this may explain why the 3D-Net was better at detecting the active motion for this gesture.

In our experiments we also noticed that some teacher frames of the *Doublewave* gesture contained artifacts around peoples' silhouettes, demonstrated in Fig. 16. Even though these samples provided a wrong guide to the network during training, this appeared to not

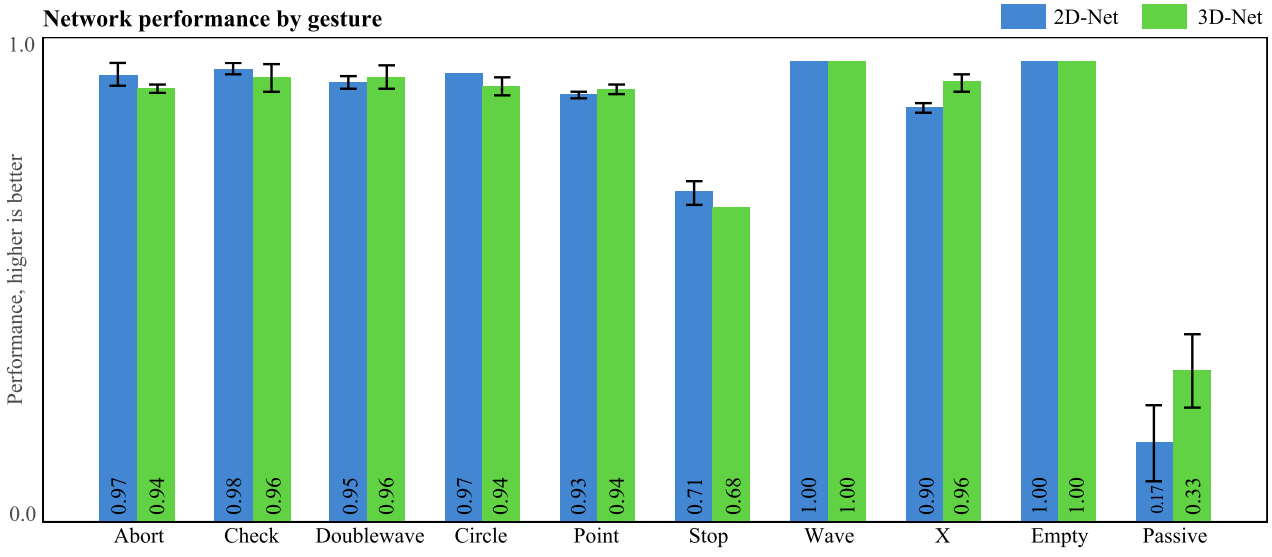

Fig. 11. Motion detection performance for both networks, evaluated on subsets of the test set, split by gesture.

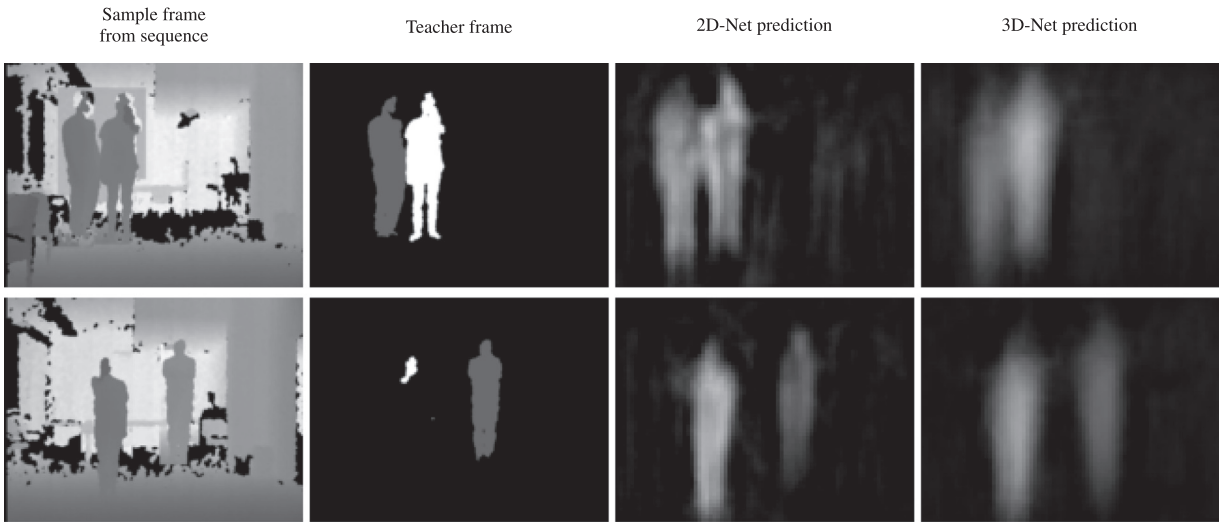

Fig. 12. Two still frames from a sequence containing the *Stop* gesture. Some teacher frames are correctly generated (top), but others do not contain the correct silhouette (bottom).

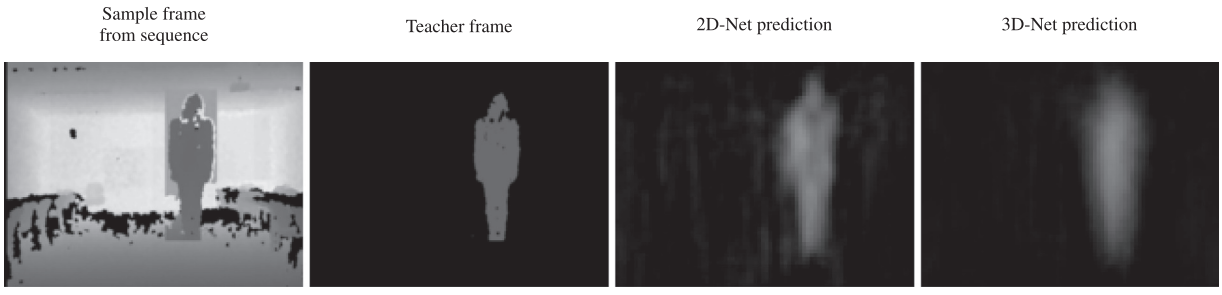

Fig. 13. A sequence containing a passive person. Both networks tended to assign an activity level to the passive person in the scene, even when they were not performing an active gesture. 3D-Net produced darker predictions, which is closer to the value of a passive silhouette, but still shows an intensity level slightly above 0.3 which the metric considered as being wrong.

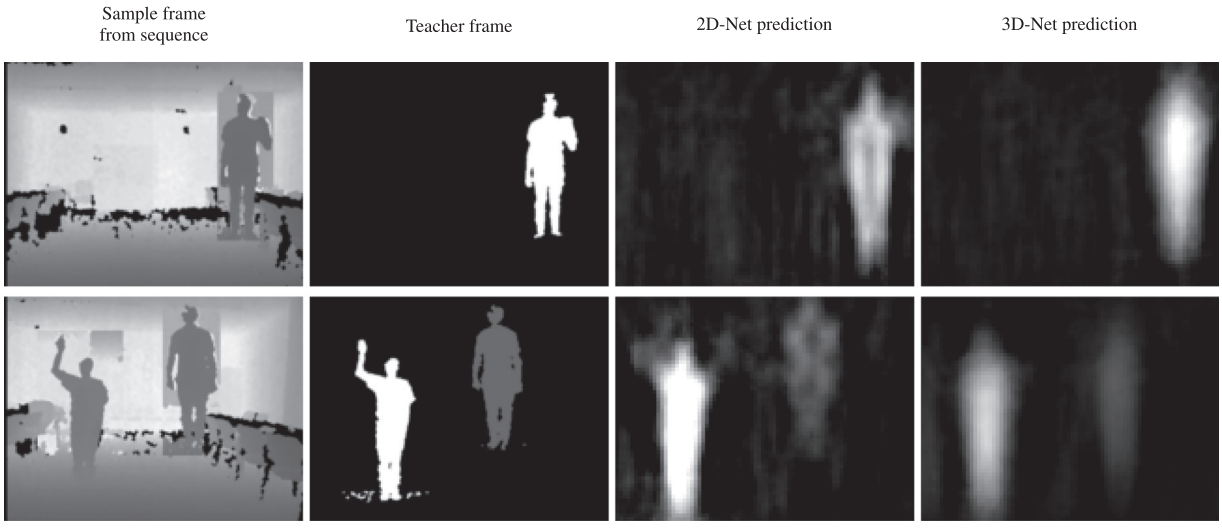

Fig. 14. Some gestures are easily recognized from a still frame: Examples show *Abort* (top) and *Circle* (bottom).

have influenced the learning in a negative way. However, the motion detection metric did not account for these cases and would produce low scores when comparing a network prediction with a teacher frame. Still, both networks showed high-performance scores of 0.95 and 0.96, so we conclude that the artifact appearances did not crucially influence the network training, but demonstrate the tolerance of CNNs towards noise in the training data.

4.1. Performance by scene configuration

After having looked at the performance depending on the gesture in the scene, we also analyzed how a different number of people in the scene affected the networks' performance scores. Just as before, we split the test set in multiple subsets: one subset with empty scenes and three subsets with one, two and three people

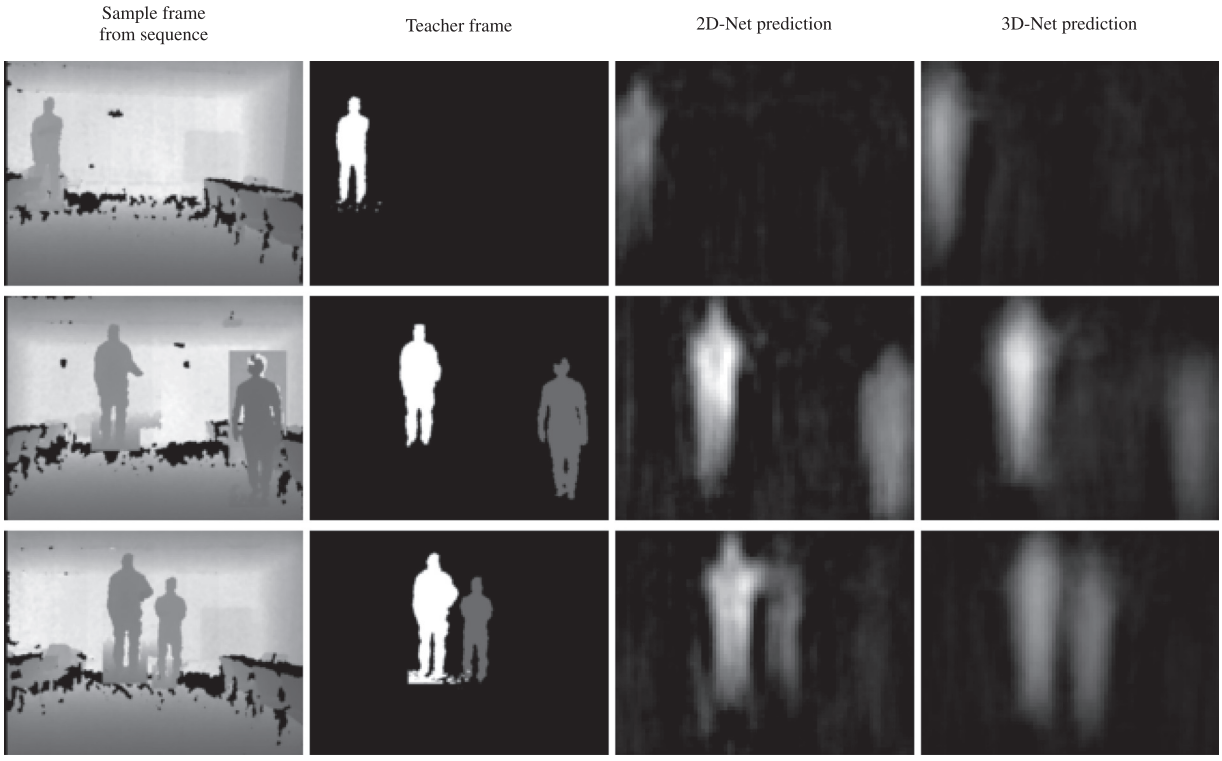

Fig. 15. Three gestures containing frames with lower arm positions: *X* (top), *Circle* (middle) and *Check* (bottom).

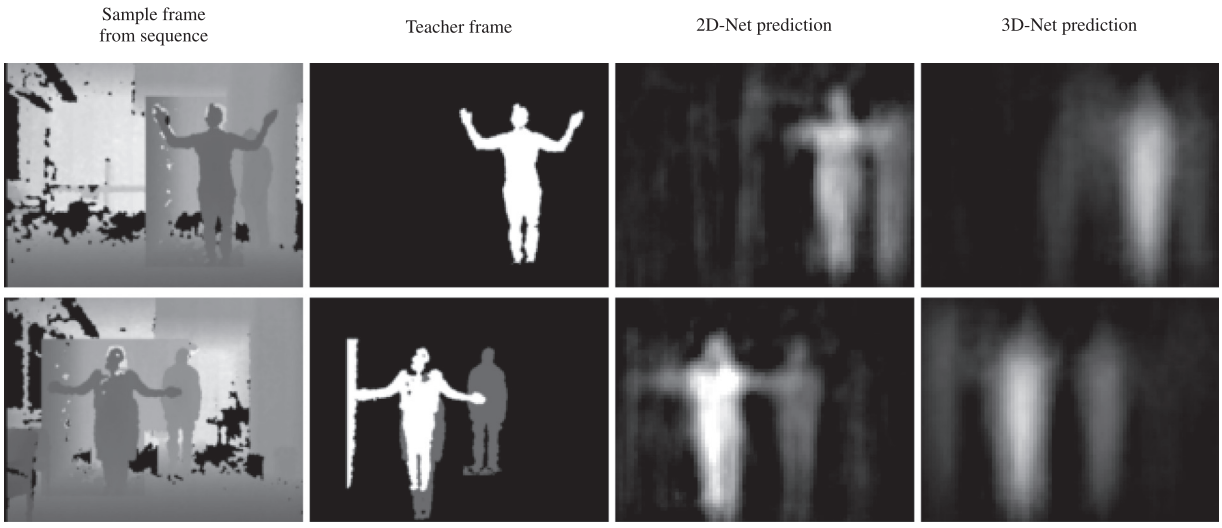

Fig. 16. Teacher frames of the *Doublewave* gesture are often generated correctly (top), but sometimes contain incorrect artifacts around the silhouette (bottom). The generalization capabilities of CNNs and their reliability towards noise prevented the networks from producing these artifacts.

in the scene. We then evaluated the motion performance metric on these subsets and compared the performance scores of the two networks (Fig. 17).

Again, Fig. 17 shows very similar results for the two architectures. Neither of the networks incorrectly detected an active gesture performer in empty scenes, which resulted in a perfect 1.00 performance for this subset. For the subsets of two and three people, the two networks performed almost equally well, in particular when taking into account the standard deviation.

Both networks performed worst for samples with one person in the scene. In half of these scenes, the person was a passive person (according to the distribution of scene configurations, see Section 2.1). As the earlier observations showed, both networks sometimes assigned intensity values to passive people which were just slightly higher than the threshold to consider a result as wrong. The fact that motion detection performance was worse for scenes with one person was caused by that data distribution and the hard thresholding of the motion metric. This caused a characteristic of the motion metric, which appears to represent well if the network has correctly localized the right subject in a multi-person scene. It does not fully reflect that an intensity value that is only slightly too high might still be acceptable in single-person scenes.

Still, it appears that both networks learned rather to decide who among multiple people is more active, instead of learning if a specific person is currently active (Fig. 18). The focus of this

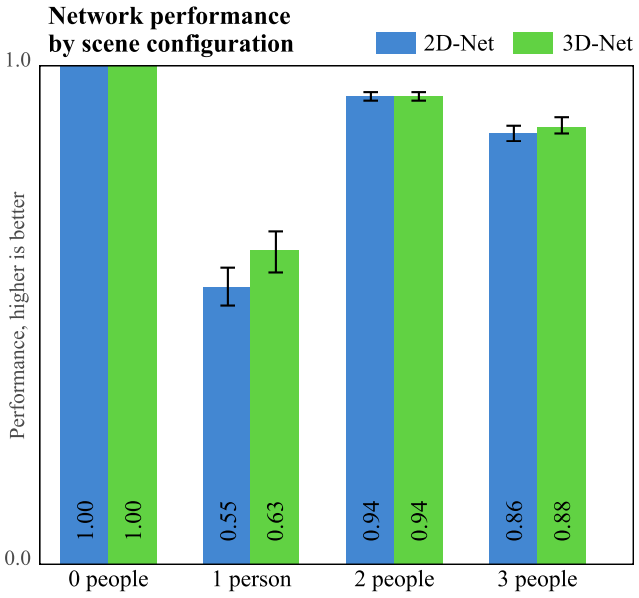

Fig. 17. Network performance for both networks, evaluated on subsets of the test set split by number of people in the scene.

study were scenes with multiple people, so it was interesting to see that both architectures performed their task better when presented with such scenes. Another observation we made explained why the 2D-Net showed comparable performance scores for the motion detection metric: many input frames with multiple people show clearly who of the actors is active and who is not (Fig. 19). In scenes with just one person, telling if the person is active depends on the temporal behavior. This is reflected by the higher score of 0.63 that 3D-Net achieved on the scenes with just 1 person, compared to 0.55 for the 2D-Net.

5. Discussion

Low-level signal data containing multiple moving people poses difficult challenges for computer vision applications. To extract meaningful information about the salient components in visual data, robust computer vision techniques are needed. In our study, we addressed the challenging task of detecting salient body movement in environments with multiple people. We designed a scenario with one person performing a dynamic body gesture and multiple passive people standing around the person. Our proposed network architecture is a 3D CNN that receives a sequence of these multi-person scenes as its input and detects and localizes the active gesture movement. The network was designed to produce a 2D output frame that showed the location of the active person with bright values in the region of salient movement. In our experiments, we compared the performance of the 3D CNN with a simpler 2D CNN that only receives a single frame as its input. Both of these architectures were trained and evaluated using a dataset we generated from multiple Kinect recordings.

Dataset. The potential of using a feature-learning approach such as ours is to encode a task simply by collecting a dataset and having the network learn all necessary features. The challenge of these data-driven approaches is that high-quality and typically very large datasets are needed so that the network can properly generalize for new and unseen data.

Artificial dataset generation has been used for different kinds of data, for example for text recognition [59], pedestrian detection [30] and action recognition [47]. As seen in the studies of [30,59], carefully synthesized datasets can be used to train classifiers that work on real data. Our approach of combining multiple Kinect recordings into scenes with multiple people resulted in generated frames that looked like an actual Kinect recording. This allowed us to create a large number of multi-person scenes according to our scene descriptions and tune parameters to experiment with different numbers of people and their positions in a frame. However, we also realized that it is essential to put effort into the generation of these frames in order to not include systematic artifacts or errors. Strategies such as removing the ground plane before isolating a person from a Kinect led to the removal of artifacts from the teacher frames and, therefore, from the output of the trained network. We also experienced some incorrectly generated teacher frames. However, as these did not introduce a systematic error in the data, our model was still able to generalize and did not produce these artifacts.

Results. Based on our dataset, we evaluated two CNN architectures using either a 2D or 3D kernel, the former originally intended to serve as a baseline network for performance comparison that would allow us to analyze the distinguishing properties of CNNs

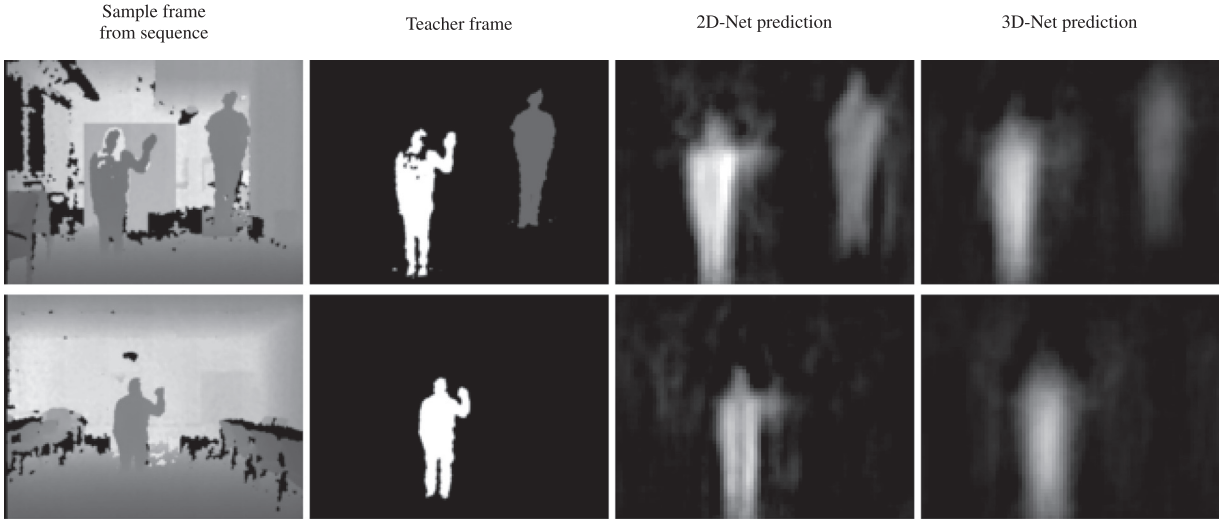

Fig. 18. Wave gesture performed in a scene with two people (top row) and one persone (bottom row). Both networks produced a brighter silhouette for the active performer when multiple people are in the scene.

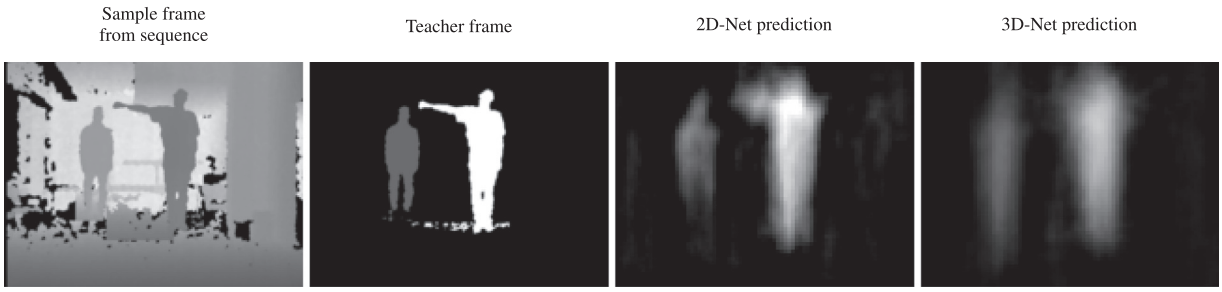

Fig. 19. Scene with two people. The still frame shows a person performing the Point gesture and one passive person. From the still frame alone, it is clear that the person on the right is the active person because of the distinct arm position.

equipped with different kernels. In addition, we employed also the popular VGG network including a pretrained version, as this network demonstrated superior performance in object detection tasks.

Both CNN networks successfully learned the desired saliency task from the dataset. They were able to detect and localize active gestures in the scene with accuracy scores which were significantly higher than the random guessing baseline and the results using the VGG16 features. As the overall performance of the two CNN models was remarkably similar, with the 2D CNN performing surprisingly well, we further focused on the analysis of the two CNNs to reveal possible differences in the specific gesture classification task.

In general, our observations of the similar performance of the two architectures align well with other studies of comparable architectures. In a video classification task, [23] found that the "variation among different CNN architectures turns out to be surprisingly insignificant", referring to 2D and 3D CNN architectures with similar designs such as ours. When [52] investigated similar architectures, they found varying the temporal kernel depth to improve the results of their 3D CNN, while still staying in the same range of results compared to a 2D CNN network. We conclude that when deploying the network on actual hardware with computational constraints, using 2D kernels exclusively appears to yield a compact feature representation while performing well on our gesture set. Faster training times and a smaller memory footprint make this simple architecture promising for real-world applications.

However, our results on the gesture performances also showed that the two networks differ in their ability to learn specific gesture characteristics. In detail, the 2D-Net was well able to detect gestures with very distinct arm poses like *Abort* and *Point* but performed worse detecting gestures with more motion like *Check, X* and *Circle*. Here, the strength of the 3D-Net became apparent, as the cubic kernel allowed learning the important consecutive nature of the gesture execution. Although the deviation in the individual performances appear small for now, our findings show a direct implication on the choice of gesture types for dataset recordings and the decision on a network architecture for detection and classification when up-scaling our experiments for more gestures. An arbitrary selection of gestures may negatively impact the network performance, for instance, the performance for the 3D-Net may drop further when including more pose-like gestures having a specific *peak* as is evident for pointing gestures. Vice versa, a 2D-Net may insufficiently learn dynamic gestures, where the inherent temporal structure of the gesture *stroke* is crucial for the correct classification. Therefore, a careful selection of gesture types is demanded when either setting up a similar scenario or when considering the extension of our dataset.

Architecture. Our proposed CNN architecture allowed supervised end-to-end learning. During training, we presented the network with image sequences and a teacher frame. The dataset defined the task to be learned and the network then learned the important features for this task. This removed the feature-engineering effort and allowed us to focus on collecting a useful dataset to describe the task to the network. This shift of engineering effort is characteristic of the computer vision research of recent years [26] and has also been seen in the area of Pedestrian Detection [45]. Our approach makes use of the same feature learning techniques but additionally includes the notion of salient movement and an active person in the scene, which pedestrian detection does not focus on.

Pre-processing steps such as perspective corrections of saliency map fusion such as in [42] are not required in our approach. In their work, these steps assured their system worked for different distances of the person to the sensor. This was not needed in our case, as our dataset contained people at different distances due to our data augmentation techniques. The CNN is able to generalize from this data and learns these invariances without manual preprocessing. Our effort was therefore moved to the creation of an appropriate dataset.

Our proposed system implements a saliency mechanism that reacts to specific body motion in the scene, i.e. gestures in our scenario. While most popular computational saliency systems [6] work on static images, our focus was on salient body motion. Motion has in fact been included in other saliency systems [55]. Also, [42] have looked at salient motion in groups of people. In contrast to these approaches, our system uses a very specific group of body movements (i.e. gestures) as the main saliency influence. Other body motion in our experiments was ignored, for example slow movement of the passive people in the scene or leg movement. This allows using our approach for applications where other irrelevant motion needs to be ignored.

Our architecture directly worked on the low-level signal data and was trained with noisy frames from the Microsoft Kinect. It does therefore not rely on the computation of some high-level knowledge like face locations [5,51] or a full skeleton model [39]. These might fail easily in challenging visual environments with occlusions, multiple people and changing lighting conditions. Some of our test samples contained occlusions caused by furniture or people, and the network showed that it was still able to localize the person performing a gesture.

6. Conclusions and future research

The present study integrates important, yet challenging topics from computer vision for the domain of human-robot interaction. We diverged from the typical one-person lab environment setup represented in the literature and introduced a novel multi-person dataset comprising salient body gestures, designed and implemented as to provide easy extension and adaptation offered to the community. Based on our dataset, we proposed a 3D CNN architecture supplied with a saliency mechanism that reacts to specific body motion cues which have not been included in previous, similar systems and demonstrated that it could reliably detect and

localize a person performing salient motion in these scenes. Our comparison on different network architectures revealed a consistently good performance of a 2D CNN, which is in line with previous critical evaluations of different kernels in a CNN. In fact, our results have an important implication of possible applications in real-world scenarios using mobile systems including robots, which are known to have limited computational capacities. Using a more lightweight architecture while maintaining good performance is definitely preferred in those situations. Finally, our analysis on learning diverse gesture types puts emphasis on a careful selection of network architectures to capture both important spatial as well as temporal characteristics and may guide other researchers in that field.

With the proposed mechanism for salient motion localization in place, we see three directions for future work: first, to investigate further improvement of the existing architectures, second, to look deeper into the specific problem context of feature learning on gestures and third, to demonstrate the application of the developed mechanism to allow gesture recognition in scenes with multiple people.

To further improve the performance of our proposed architecture, we want to investigate alternative architecture decisions. Our convolution kernels and pooling parameters had the same extent through all three dimensions. However, the spatio-temporal feature learning could change significantly when choosing separate settings for the spatial and temporal domains. As the temporal dimension of our input sequences (15 frames) was a lot smaller than both of the spatial dimensions (120 and 160), reflecting this in the convolution and pooling operations could be promising, as suggested by Tran et al. [52]. We also expect the addition of more modalities, most notably RGB frames, to increase the performance of the system. Additionally, investigating a hybrid network architecture that uses both 2D and 3D convolutions could potentially combine the strengths of the 2D-Net and 3D-Net as identified in our results. Also, network models learning significant feature representations in an unsupervised way like autoencoders or generative adversarial networks (GAN) have evolved in related vision domains and thus may also complement CNN architectures to capture the different aspects of the input data. At last, to extend the task of the models from pure localization to a more accurate reconstruction of the silhouettes, alternatives to the MLP could potentially provide a better pixel-accurate network output, for example by using de-conv operations which have shown good results on semantic segmentation tasks.

As we have learned from our results, gesture characteristics influence the learning and the performance of the trained model. Understanding which kinds of body movement can be learned by which kind of architecture is crucial to the choice of a network architecture given a specific gesture set. Hence, we see questions arising of how well architectures can be scaled up to larger gesture sets. As our architecture only ever saw 15 frames, we would expect it to learn a general notion of typical arm movement when trained with a sufficient variety of gestures. We would expect such a model to also locate gestures with similar movements, even though it has never seen the specific gesture before. Future research taking into consideration the varying nature of gestures and their impact on performance will offer improved applications for novel interaction systems.

To demonstrate the application potential of our proposed architectures, we would like to include them in a gesture recognition system. The main motivation of our study was to localize salient motion in order to use this information in an application, such as a recognition task. Transferring the saliency mechanism we developed into a real online application context with multiple people is an exciting direction for future work and our study provides a crucial step towards that goal.

Acknowledgments

The authors gratefully acknowledge partial support from the German Research Foundation DFG under project CML (TRR 169), and the European Union under projects SECURE (No. 642667), and SOCRATES (No. 721619). Florian Letsch would like to thank Twenty Billion Neurons GmbH for supporting him working on this manuscript.

References

[1] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, A. Baskurt, Sequential deep learning for human action recognition, in: Proceedings of the International Workshop on Human Behavior Understanding, Springer, 2011, pp. 29–39.
[2] P. Barros, S. Magg, C. Weber, S. Wermter, A multichannel convolutional neural network for hand posture recognition, in: Proceedings of the Artificial Neural Networks and Machine Learning - ICANN 24th International Conference on Artificial Neural Networks, Hamburg, Germany, 2014, pp. 403–410.
[3] P. Barros, G.I. Parisi, D. Jirak, S. Wermter, Real-time gesture recognition using a humanoid robot with a deep neural architecture, in: Proceedings of the 14th IEEE-RAS International Conference on Humanoid Robots (Humanoids), IEEE, 2014, pp. 646–651.
[4] P. Barros, G.I. Parisi, C. Weber, S. Wermter, Emotion-modulated attention improves expression recognition: a deep learning model, Neurocomputing 253 (2017) 104–114. Learning Multimodal Data https://doi.org/10.1016/j.neucom.2017.01.096.
[5] M. Bennewitz, F. Faber, D. Joho, M. Schreiber, S. Behnke, Towards a humanoid museum guide robot that interacts with multiple persons, in: Proceedings of the 5th IEEE-RAS International Conference on Humanoid Robots, IEEE, 2005, pp. 418–423.
[6] Z. Bylinskii, T. Judd, A. Borji, L. Itti, F. Durand, A. Oliva, A. Torralba, MIT Saliency Benchmark, 2015, (http://saliency.mit.edu).
[7] L. Chen, H. Wei, J. Ferryman, A survey of human motion analysis using depth imagery, Pattern Recogn. Lett. 34 (15) (2013) 1995–2006. Smart Approaches for Human Action Recognition doi: 10.1016/j.patrec.2013.02.006.
[8] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 1, 2005, pp. 886–893, doi:10.1109/CVPR.2005.177.
[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: a large-scale hierarchical image database, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,(CVPR), Ieee, 2009, pp. 248–255.
[10] P. Dollár, C. Wojek, B. Schiele, P. Perona, Pedestrian detection: a benchmark, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009, pp. 304–311.
[11] S. Escalera, X. Baró, J. Gonzàlez, M.A. Bautista, M. Madadi, M. Reyes, V. Ponce-López, H.J. Escalante, J. Shotton, I. Guyon, Chalearn looking at people challenge 2014: dataset and results, in: L. Agapito, M.M. Bronstein, C. Rother (Eds.), Proceedings of the Computer Vision - ECCV 2014 Workshops, Springer International Publishing, Cham, 2015, pp. 459–473.
[12] R. Girshick, Fast R-CNN, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1440–1448.
[13] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 580–587.
[14] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proceedings of the Artificial Intelligence and Statistics, 9, 2010, pp. 249–256.
[15] M. Gullberg, K. Holmqvist, What speakers do and what addressees look at: visual attention to gestures in human interaction live and on video, Pragmat. Cognit. 14 (1) (2006) 53–82.
[16] S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift, CoRR abs/1502.03167 (2015).
[17] U. Iqbal, A. Milan, J. Gall, Posetrack: joint multi-person pose estimation and tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
[18] L. Itti, C. Koch, A saliency-based search mechanism for overt and covert shifts of visual attention, Vision Res. 40 (10) (2000) 1489–1506.
[19] B. Jähne, H. Scharr, S. Körkel, B. Jähne, H. Haußecker, Geißler, in: Principles of Filter Design, 2, Academic Press, 1999, pp. 125–151.
[20] S. Ji, W. Xu, M. Yang, K. Yu, 3D convolutional neural networks for human action recognition, in: Proceedings of the International Conference on Machine Learning, 2010, pp. 495–502.
[21] D. Jirak, S. Wermter, Sparse autoencoders for posture recognition, in: Proceedings of the International Joint Conference on Neural Networks (IJCNN 2018), 2018.
[22] P. Kakumanu, S. Makrogiannis, N. Bourbakis, A survey of skin-color modeling and detection methods, Pattern Recogn. 40 (3) (2007) 1106–1122.
[23] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, F. Li, Large-scale video classification with convolutional neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2014, pp. 1725–1732, doi:10.1109/CVPR.2014.223.
[24] D. Kingma, J. Ba, Adam: A Method for Stochastic Optimization, Proceedings of the 3rd International Conference on Learning Representations, 2015.

[25] C. Koch, S. Ullman, Shifts in selective visual attention: towards the underlying neural circuitry, Human Neurobiol. 4 (4) (1985) 219–227.
[26] A. Krizhevsky, I. Sutskever, G. Hinton, classification with deep convolutional neural networks, in: Proceedings of the Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.
[27] S.S. Kruthiventi, K. Ayush, R.V. Babu, Deepfix: a fully convolutional neural network for predicting human eye fixations, CoRR abs/1510.02927 (2015).
[28] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
[29] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.
[30] J. Marin, D. Vázquez, D. Gerónimo, A.M. López, Learning appearance in virtual scenarios for pedestrian detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2010, pp. 137–144.
[31] A. Milan, L. Leal-Taix, K. Schindler, I. Reid, Joint tracking and segmentation of multiple targets, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 5397–5406, doi:10.1109/CVPR.2015.7299178.
[32] S. Mitra, T. Acharya, Gesture recognition: a survey, IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. 37 (3) (2007) 311–324.
[33] T. Miyashita, M. Shiomi, H. Ishiguro, Multisensor-based human tracking behaviors with Markov chain monte carlo methods, in: Proceedings of the 4th International Conference on IEEE/RAS, 2, IEEE, 2004, pp. 794–810.
[34] M. Munaro, E. Menegatti, Fast Rgb-d people tracking for service robots, Auton. Robots 37 (3) (2014) 227–242, doi:10.1007/s10514-014-9385-0.
[35] V. Nair, G.E. Hinton, Rectified linear units improve restricted Boltzmann machines, in: Proceedings of the 27th International Conference on Machine Learning (ICML-10), 2010, pp. 807–814.
[36] N. Neverova, C. Wolf, G.W. Taylor, F. Nebout, Multi-scale deep learning for gesture detection and localization, in: L. Agapito, M.M. Bronstein, C. Rother (Eds.), Proceedings of the Computer Vision - ECCV 2014 Workshops, Springer International Publishing, Cham, 2015, pp. 474–490.
[37] S. Nobe, S. Hayamizu, O. Hasegawa, H. Takahashi, Hand gestures of an anthropomorphic agent: listeners' eye fixation and comprehension, Cogn. Stud. 7 (1) (2000) 86–92.
[38] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, T. Poggio, Pedestrian detection using wavelet templates, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE, 1997, pp. 193–199.
[39] G.I. Parisi, D. Jirak, S. Wermter, HandSOM - neural clustering of hand motion for gesture recognition in real time, in: Proceedings of the 23rd IEEE International Symposium on Robot and Human Interactive Communication, IEEE, 2014, pp. 981–986.
[40] V. Ramanathan, J. Huang, S. Abu-El-Haija, A.N. Gorban, K. Murphy, L. Fei-Fei, Detecting events and key actors in multi-person videos., CoRR abs/1511.02917 (2015).
[41] S.S. Rautaray, A. Agrawal, Vision based hand gesture recognition for human computer interaction: a survey, Artif. Intell. Rev. 43 (1) (2015) 1–54.
[42] N. Riche, M. Mancas, B. Gosselin, T. Dutoit, 3D saliency for abnormal motion selection: the role of the depth map, in: Proceedings of the International Conference on Computer Vision Systems, Springer, 2011, pp. 143–152.
[43] S. Ruffieux, D. Lalanne, E. Mugellini, O.A. Khaled, A survey of datasets for human gesture recognition, in: Proceedings of the International Conference on Human-Computer Interaction, Springer, 2014, pp. 337–348.
[44] C. Schuldt, I. Laptev, B. Caputo, Recognizing human actions: a local SVM approach, in: Proceedings of the 17th International Conference on Pattern Recognition, ICPR, 3, IEEE, 2004, pp. 32–36.
[45] P. Sermanet, K. Kavukcuoglu, S. Chintala, Y. LeCun, Pedestrian detection with unsupervised multi-stage feature learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 3626–3633.
[46] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, CoRR abs/1409.1556 (2014).
[47] C.R. de Souza, A. Gaidon, Y. Cabon, A.M.L. Peña, Procedural generation of videos to train deep action recognition networks, CoRR abs/1612.00881 (2016).
[48] L. Spinello, K.O. Arras, People detection in RGB-D data, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2011, pp. 3838–3843.
[49] N. Srivastava, G.E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.
[50] T. Tasaki, K. Komatani, T. Ogata, H.G. Okuno, Spatially mapping of friendliness for human-robot interaction, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2005, pp. 1277–1282.
[51] T. Tasaki, S. Matsumoto, H. Ohba, M. Toda, K. Komatani, T. Ogata, H.G. Okuno, Dynamic communication of humanoid robot with multiple people based on interaction distance, in: Proceedings of the IEEE International Workshop on Robot and Human Interactive Communication (ROMAN), IEEE, 2004, pp. 71–76.
[52] D. Tran, L. Bourdev, R. Fergus, L. Torresani, M. Paluri, Learning spatiotemporal features with 3D convolutional networks, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015.
[53] J. Triesch, C. von der Malsburg, Classification of hand postures against complex backgrounds using elastic graph matching, Image Vis. Comput. 20 (13) (2002) 937–943, doi:10.1016/S0262-8856(02)00100-2.
[54] J.K. Tsotsos, Analyzing vision at the complexity level, Behav. Brain Sci. 13 (03) (1990) 423–445.
[55] J.K. Tsotsos, Y. Liu, J.C. Martinez-Trujillo, M. Pomplun, E. Simine, K. Zhou, Attending to visual motion, Comput. Vis. Image Understand. 100 (1) (2005) 3–40.
[56] C. Vondrick, D. Patterson, D. Ramanan, Efficiently scaling up crowdsourced video annotation, Int. J. Comput. Vis. (2012) 1–21. 10.1007/s11263-012-0564-1
[57] S. Walk, N. Majer, K. Schindler, B. Schiele, New features and insights for pedestrian detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010.
[58] H. Wang, A. Kläser, C. Schmid, C.-L. Liu, Dense trajectories and motion boundary descriptors for action recognition, Int. J. Comput. Vis. 103 (1) (2013) 60–79.
[59] T. Wang, D.J. Wu, A. Coates, A.Y. Ng, End-to-end text recognition with convolutional neural networks, in: Proceedings of the 21st International Conference on Pattern Recognition (ICPR),, IEEE, 2012, pp. 3304–3308.
[60] B. Xiao, H. Wu, Y. Wei, Simple baselines for human pose estimation and tracking, in: V. Ferrari, M. Hebert, C. Sminchisescu, Y. Weiss (Eds.), Proceedings of the Computer Vision – ECCV, Springer International Publishing, Cham, 2018, pp. 472–487.
[61] J. Zhang, W. Li, P.O. Ogunbona, P. Wang, C. Tang, Rgb-d-based action recognition datasets: a survey, Pattern Recogn. 60 (2016) 86–105, doi:10.1016/j.patcog.2016.05.019.

Florian Letsch received his Bachelor's degree and Master's degree both in Computer Science from the University of Hamburg, Germany. His main research interests include deep learning for novel human-robot interaction applications and deep learning on video data. He is employed as a deep learning engineer at Twenty Billion Neurons GmbH in Berlin where he works on video understanding of human actions.

Doreen Jirak received her Diploma degree and the Doctorate both in Computer Science from the University of Hamburg. Her research interests include neural network analysis, neural vision and sensorimotor transformations, and gesture recognition in human-robot interaction. She is currently employed as a postdoctoral researcher and teaching associate in the Knowledge Technology group at the University of Hamburg.

Stefan Wermter received the Diplom from the University of Dortmund, the M.Sc. from the University of Massachusetts, and the Doctorate and Habilitation from the University of Hamburg, all in Computer Science. He has been a visiting research scientist at the International Computer Science Institute in Berkeley before leading the Chair in Intelligent Systems at the University of Sunderland, UK. Currently Stefan Wermter is Full Professor in Computer Science at the University of Hamburg and Director of the Knowledge Technology institute. His main research interests are in the fields of neural networks, hybrid systems, cognitive neuroscience, bio-inspired computing, cognitive robotics and natural language processing. In 2014 he was general chair for the International Conference on Artificial Neural Networks (ICANN). He is also associate editor of the journals Transactions on Neural Networks and Learning Systems, Connection Science, and the International Journal for Hybrid Intelligent Systems and he is on the editorial board of the journals Cognitive Systems Research and Journal of Computational Intelligence.