Proc. 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2017)

Two-Stream RNN/CNN for Action Recognition in 3D Videos

Rui Zhao, Haider Ali, Patrick van der Smagt

***Abstract*— The recognition of actions from video sequences has many applications in health monitoring, assisted living, surveillance, and smart homes. Despite advances in sensing, in particular related to 3D video, the methodologies to process the data are still subject to research. We demonstrate superior results by a system which combines recurrent neural networks with convolutional neural networks in a voting approach. The gated-recurrent-unit-based neural networks are particularly well-suited to distinguish actions based on long-term information from optical tracking data; the 3D-CNNs focus more on detailed, recent information from video data. The resulting features are merged in an SVM which then classifies the movement. In this architecture, our method improves recognition rates of state-of-the-art methods by 14% on standard data sets.**

I. Introduction

Recognition of human activity in 3D videos has received increasing attention since 2010 [1]–[7]. Compared to 2D videos, 3D videos provide more spatial information and could be more informative. Action recognition with 3D videos is applied in different fields, such as health monitoring for patients, assisted living for disabled people, and robot perception and cognition.

Following this line of research, this paper proposes and applies novel deep-learning methods on what is currently the largest 3D action recognition dataset. Our results are compared with existing best approaches and are shown to be superior. Our proposed deep-learning methods consist mainly of three parts: a novel skeleton-based recurrent neural network structure, using a 3D-convolutional [8] neural network for RGB videos, and sketching a new two-stream fusion method to combine RNN and CNN. All methods are evaluated on the NTU RGB+D Dataset [2]. The dataset was published in 2016 and contains more than 56k action samples in four different modalities: RGB videos, depth map sequences, 3D skeletal data, and infrared videos. The dataset consists of 60 different action classes including daily, health-related, and mutual actions. In this paper, we use both the 3D skeletal data and RGB videos.

Traditional studies on 3D action recognition use different kinds of methods [1], [9]–[52] to compute handcrafted features, while deep-learning approaches [2]–[7], [53], [54] are end-to-end trainable and can be applied directly on raw data. Focussing on the latter, for skeleton-based activity analysis, [2]–[5] used different kinds of recurrent neural networks to acquire state-of-the-art performances on various of 3D action datasets. Du et al. [3] propose an hierarchical RNN, which is fed with manually divided five groups of the human skeleton, such as two hands, two legs, and one torso. Inspired from this, Shahroudy et al. [2] present a novel long short-term memory (LSTM) [55] cell, called part-aware LSTM, which is also fed with separated five parts of skeleton. Evolved from these two ideas, Zhu et al. [5] provide a novel deep RNN structure, which can automatically learn the co-occurrence, similar to grouping data into five human body parts, from skeleton data. Most recently, Liu et al. [4] propose a skeleton tree traversal algorithm and a new gating mechanism to improve robustness against noise and occlusion.

However, our proposed RNN structure makes a different contribution. Our method is inspired by recent normalization technologies [56] and a novel recurrent neuron mechanism [57]. With these advanced deep learning technologies embedded into our RNN structure, it can be trained with 13 times fewer iterations and for each iteration consumes 20% less computational time, compared to a normal RNN model with LSTM cells. Our contribution focuses more on making the network much easier to train, less inclined to overfitting, and deep enough to represent the data. More importantly, our proposed RNN structure outperforms all other skeleton-based methods on the largest 3D action recognition dataset.

To process RGB videos, our method is inspired by different kinds of convolutional neural networks [8], [58]–[60]. We use a 3D-CNN [8] model on the RGB videos of the NTU RGB+D dataset. We compare the results with proposed RNN models, and fuse their output.

To combine the RNN and CNN models, we propose two fusion structures: decision and feature fusion. The first is very simple to use, whereas the second provides a better performance. For decision fusion, we illustrate a voting method based on the confidence of the classifiers. For feature fusion, we propose a novel two-stream RNN/CNN structure, shown in Fig. 4, which combines temporal and spatial features from the RNN and CNN models and boost the performance by a significant margin. Our two-stream RNN/CNN structure outperforms the current state-of-the-art method [4] more than 14% on both cross subject and cross view settings.

To summarize, our contributions in this paper are:

- A novel RNN structure is proposed, which converges 13 times faster during training and costs 20% less computational power at each forward pass, compared to a normal LSTM;
- Two fusion methods are proposed to combine the proposed RNN structure and a 3D-CNN structure [8]. The decision fusion is easier to use, while the feature fusion has superior performance.

RZ is with Siemens AG & Ludwig Maximilian University of Munich ruizhao@siemens.com. HA is with the Robotics and Mechatronics Center (RMC), German Aerospace Center (DLR) & Johns Hopkins University Haider.Ali@dlr.de. PvdS is with Data:Lab, Volkswagen Group. Work done when RZ at DLR-RMC & Technische Universität München.

II. Methodology

In this section, we first introduce the concept of recurrent neural networks and batch normalization, and then describe the proposed RNN structure: a deep bidirectional gated recurrent neural network with batch normalization and dropout. Afterwards, the applied 3D-CNN model and two-stream RNN/CNN fusion architectures, i.e., decision fusion and feature fusion, are introduced.

A. Recurrent Neural Network

1) Vanilla Recurrent Neural Network: Recurrent neural networks can handle sequence information with varied lengths of time steps. This transforms the input $\mathbf{X}$ to a internal hidden state $\mathbf{h}_t$ at each time step. The network passes the state $\mathbf{h}_t$ along with the next input $\mathbf{x}_{t+1}$ to the neuron, time step after time step. The neuron learns when to remember and forget information with nonlinear activation functions:

$$\mathbf{h}_t = \sigma\left(\mathbf{W}\begin{pmatrix}\mathbf{x}_t\\ \mathbf{h}_{t-1}\end{pmatrix}\right) \tag{1}$$

$$\mathbf{y}_t = \sigma\left(\mathbf{V}\mathbf{h}_t\right) \tag{2}$$

where $t \in \{1, \ldots, T\}$ represents time steps, and σ represents a nonlinear activation function such as a standard logistic sigmoid function $\mathrm{sigm}(x) = 1/(1+\mathrm{e}^{-x})$ or a hyperbolic tangent function $\tanh(x)$.

Multiple layers of RNN can be stacked to increase the complexity:

$$\mathbf{h}_t^{(l)} = \sigma\left(\mathbf{W}^{(l)}\begin{pmatrix}\mathbf{h}_t^{(l-1)}\\ \mathbf{h}_{t-1}^{(l)}\end{pmatrix}\right) \tag{3}$$

$$\mathbf{h}_t^{(0)} := \mathbf{x}_t \tag{4}$$

$$\mathbf{y}_t = \sigma\left(\mathbf{V}\mathbf{h}_t^{(L)}\right) \tag{5}$$

where $l \in \{1, ..., L\}$ denotes the layer number.

In practice, a vanilla RNN does not remember information over a longer time; a problem which is related to the vanishing gradient problem.

2) Long Short-Term Memory: This problem can be solved by LSTM [55] which stores information in gated cells at the neurons. This allows errors to be backpropagated through hundreds or thousands of time steps:

$$\begin{pmatrix}\mathbf{i}\\ \mathbf{f}\\ \mathbf{o}\\ \hat{\mathbf{c}}_t\end{pmatrix} = \begin{pmatrix}\mathrm{sigm}\\ \mathrm{sigm}\\ \mathrm{sigm}\\ \tanh\end{pmatrix}\mathbf{W}\begin{pmatrix}\mathbf{x}_t\\ \mathbf{h}_{t-1}\end{pmatrix} \tag{6}$$

$$\mathbf{c}_t = \mathbf{f}\odot\mathbf{c}_{t-1} + \mathbf{i}\odot\hat{\mathbf{c}}_t \tag{7}$$

$$\mathbf{h}_t = \mathbf{o}\odot\tanh(\mathbf{c}_t) \tag{8}$$

where $\mathbf{i}, \mathbf{f}$ and $\mathbf{o}$ denote input gate, forget gate, and output gate, respectively. $\hat{\mathbf{c}}_t$ represents new candidate values, which could be added to the cell state $\mathbf{c}_t$. We use $\odot$ for element-wise multiplication.

3) Gated Recurrent Unit: An improvement to LSTM called gated recurrent unit (GRU) was proposed in [57]. GRU has a simpler structure and can be computed faster. The three gates from LSTM are combined into two gates, respectively updating gate $\mathbf{z}$ and resetting gate $\mathbf{r}$ in GRU. GRU also combines cell state $\mathbf{c}_t$ and hidden state $\mathbf{h}_t$ into one state $\mathbf{h}_t$. The mathematical description is as follows:

$$\begin{pmatrix}\mathbf{z}\\ \mathbf{r}\end{pmatrix} = \begin{pmatrix}\mathrm{sigm}\\ \mathrm{sigm}\end{pmatrix}\left(\mathbf{W}\begin{pmatrix}\mathbf{x}_t\\ \mathbf{h}_{t-1}\end{pmatrix}\right) \tag{9}$$

$$\hat{\mathbf{h}_t} = \tanh\left(\mathbf{W}\begin{pmatrix}\mathbf{x}_t\\ r\odot\mathbf{h}_{t-1}\end{pmatrix}\right) \tag{10}$$

$$\mathbf{h}_t = \mathbf{z}\odot\mathbf{h}_{t-1} + (1-\mathbf{z})\odot\hat{\mathbf{h}}_t \tag{11}$$

where $\hat{\mathbf{h}}_t$ denotes new candidate state values.

4) Bidirectional Recurrent Neural Network: A bidirectional RNN [61] performs a forward pass and a backward pass, which runs input data from $t = T$ to $t = 1$ and from $t = 1$ to $t = T$, respectively.

For classification, the output of an RNN $\mathbf{y}_t$ can be passed to a fully-connected layer with softmax activation functions; this allows us to interpret the output as a probability.

5) Batch Normalization: To train a deep neural network, the *internal covariate shift* [56] slows down the training process. The *internal covariate shift* is the distribution of each layer's input changes during training, because the parameters in the previous layer are changing. To reduce the *internal covariate shift*, we could whiten the layer activations, but this takes too much computation power. Batch normalization, a part of the neural network structure, approximates this process by standardizing the activations $\mathbf{x}$ using a statistical estimate of the mean $\widehat{\mathrm{E}}(\mathbf{x})$ and standard deviation $\widehat{\mathrm{Var}}(\mathbf{x})$ for each training mini-batch. It can be shown that

$$\mathrm{BN}(\mathbf{x};\gamma,\beta) = \gamma\frac{\mathbf{x}-\widehat{\mathrm{E}}(\mathbf{x})}{\sqrt{\widehat{\mathrm{Var}}(\mathbf{x})+\epsilon}} + \beta \tag{12}$$

where $\gamma \in \mathbb{R}^d$ and $\beta \in \mathbb{R}^d$ are scale and shift parameters for the activation $\mathbf{x} \in \mathbb{R}^{n\times d}$. With these, identity transformation for each activation could be presented. $\epsilon \in \mathbb{R}$ is a constant added as a regularization parameter for numerical stability. The division in Eq. (12) is performed element-wise. γ and β are learned during training and fixed during inference.

6) Proposed RNN Structure: For skeleton-based action recognition tasks, the data set consists of the 3D coordinates of a number of body joints. We feed this information, together with action labels, to an RNN. This RNN network has two bidirectional layers, each of which consists of 300 GRU cells. After the recurrent layers follows the batch normalization layer, which standardizes the activations from the RNN layer. Then the normalized activations flow to the next fully-connected layer with 600 rectified linear unit (ReLU) [62] activation functions. During training, in each iteration the network randomly drops out 25% of the neurons between the batch normalization layer and the next fully-connected layer to reduce overfitting. Lastly, a softmax layer

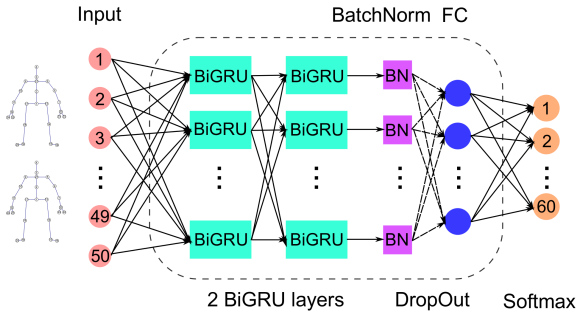

Fig. 1: Proposed RNN structure using two bidirectional gated recurrent layers with batch normalization, dropout, one hidden fully-connected layer, and one output softmax layer. For clarity, the temporal recurrent structure of GRU cells is not shown here.

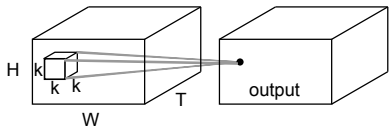

Fig. 2: Illustration of 3D convolution operation on a video volume resulting in another volume. H and W are height and width of a frame, T means the maximal time step of a video. k denotes the size of a kernel.

maps the compressed motion information (features) to 60 action classes. Fig. 1 shows the structure of this RNN network.

To highlight the improvements of this final proposed model, we compare our approach to simpler models. These models are a standard RNN; an LSTM-RNN; LSTM plus batch normalization ("LSTM-BN"), LSTM-BN with dropout ("LSTM-BN-DP"), GRU-BN-DP, and a bidirectional GRU-BN-DP which we call "BI-GRU-BN-DP". All these models have one recurrent layer. The next complexity is adding an extra layer of hidden units to the last model ("2 layer BI-GRU-BN-DP"). Finally, we add another fully-connected layer on top, before the softmax layer, and call this model "2 layer BI-GRU-BN-DP-H". Sec. III discusses the results of all models.

B. Convolutional Neural Network

To process RGB videos, we choose to use the 3D-CNN model from [8], as it shows promising performances on 2D video action recognition tasks. We believe that 3D convolution nets are more suitable for learning features from videos than 2D convolution nets.

2D convolution generates a series of 2D feature maps from images. Inspired by this, a 3D convolution processes frame clips, where the third dimension is time step, which results in a series of 3D feature volumes, as shown in Fig. 2. This compressed representation contains spatiotemporal information from the video clips. To learn a rich amount of features, multiple layers of convolution and max-pooling operations are stacked into one model.

To be specific, the 3D-CNN model [8], which we choose, has five convolutional groups, each group has one or two convolutional layers and one max-pooling layer, two fully-connected layers, and one softmax output layer. The details of this model are presented in Fig. 3.

Conv1	Conv2	Conv3	Conv4	Conv5	FC6	FC7	Output
64	128	2x256	2x512	2x512	4096	4096	60
Pool 1	Pool 2	Pool 3	Pool 4	Pool 5			

Fig. 3: 3D-CNN structure (C3D): 8 convolution, 5 max-pooling, 2 fully-connected layers, and 1 softmax output layer. All convolution kernels are of size $3 \times 3 \times 3$ with stride 1. Number of filters are shown in each box. All pooling kernels are $2 \times 2 \times 2$, except the first one, which is $1 \times 2 \times 2$. Each fully-connected layer has 4096 units.

We finetune this model with pretrained parameters [8] on Sports-1M Dataset, which has approximately one million YouTube videos. This reduces overfitting and demands less training time on the current dataset.

C. Two-stream RNN/CNN

As having the proposed RNN structure for the skeleton data and the 3D-CNN model for the RGB videos, we want to combine the strengths of RNN and CNN nets. To improve the performance, we propose two fusion models, decision fusion and feature fusion.

1) Decision Fusion: In the case of decision fusion, we use a simple but efficient voting method, inspired by majority voting. As a result of having only two classifiers, we cannot apply majority voting. Instead, the fusion method predicts based on voting confidence.

We first split the dataset into training, validation and testing. The same training set is used to train the RNN and CNN nets. The validation set is then used to find the best parameters, trust weights w_r and w_c for the voting method. We initialize the trust weights with equal values, which means $w_r = 1.00$ and $w_c = 1.00$ for both RNN and CNN classifiers. Afterwards, we compare the confidences, which are the highest probabilities of softmax output from both classifiers for each prediction. The more confident one wins:

$$y(x_i) = \begin{cases} y_r(x_i), & \text{if } w_r \times y_r(x_i) > w_c \times y_c(x_i) \\ y_c(x_i), & \text{otherwise} \end{cases} \quad (13)$$

where $y(x_i)$ is the fused prediction for sample x_i; y_r and y_c denote RNN and CNN prediction, respectively.

Based on this concept, we develop a way to fuse the predictions from RNN and CNN. We evaluate the performance with the validation dataset and search for the best trust weights for decision fusion. Having only two parameters w_r and w_c, only little tuning is needed.

2) Feature Fusion: Another way to combine these two neural networks is feature fusion.

We first train the RNN and CNN models on the training dataset. As in training, neural nets can learn discriminant information from raw data. Thus, we use the trained RNN model to extract temporal features from 3D skeleton data

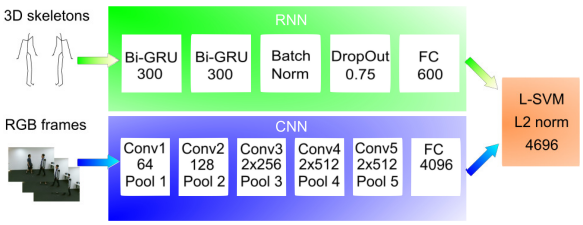

Fig. 4: Two-stream RNN/CNN structure: The RNN stream is fed with the 3D coordinates of two human skeletons as input, then followed by two bidirectional gated recurrent layer with 300 units in each direction. The output from recurrent layers is later batch-normalized. Dropout is only enabled during training with 75% keep probability. The RNN features come from the fully-connected layer. The CNN stream is fed with RGB clips (16 frames as a clip) and consists of five convolution groups and a fully-connected layer (fc-6). The CNN features are extracted from the fc-6 layer and later combined with RNN features, then L2-normalized, and fed to a linear SVM, which predicts the actions.

and use the trained CNN model to learn spatiotemporal features from RGB videos. Both features come from the first fully-connected layer in each model. The features are concatenated, L2 normalized, and eventually, fed to a linear SVM classifier.

The SVM parameter C is found using the validation dataset. Then the model is tested on the test set. The feature fusion structure for two streams of RNN and CNN features is presented in Fig. 4.

III. Experiments

The models introduced in the previous section are evaluated in the experiments. The dataset is first introduced in this section, then the setups and parameter settings for the experiments are illustrated. We compare the results of the proposed models with the current best methods. In the end, we analyze and discuss the problems related to deep learning methods for 3D action recognition.

A. NTU RGB+D Dataset [2]

The proposed approaches are evaluated on the NTU RGB+D dataset [2], which we know as the current largest publicly available 3D action recognition dataset. The dataset consists of more than 56k action videos and 4 million frames, which were collected by 3 Kinect V2 cameras from 40 distinct subjects, and divided into 60 different action classes including 40 daily (drinking, eating, reading, etc.), 9 health-related (sneezing, staggering, falling down, etc.), and 11 mutual (punching, kicking, hugging, etc.) actions. It has four major data modalities provided by the Kinect sensor: 3D coordinates of 25 joints for each person (skeleton), RGB frames, depth maps, and IR sequences. In this paper, we use the first two modalities, since they are the two most informative modalities.

The large intra-class and view point variations make this dataset challenging. However, the large amount of action samples makes it highly suitable for data-driven methods.

This dataset has two standard evaluation criteria [2]. The first one is a cross-subject test, in which half of the subjects are used for training and the other half are used for testing. The second one is a cross-view test, in which two viewpoints are used for training and one is excluded for evaluation.

B. Implementation details

In our experiments, the implementation consists of RNN, CNN, and Fusion. For all these models we use the same training, validation and testing splits. The validation set is composed of 10% of the subjects in the training set in [2]. The remaining subjects in the training set [2] make up the training set.

1) RNN Implementation: In the RNN experiments, we have two human skeletons as input, each skeleton has 25 3D coordinates. Since the longest time step is 300, we pad all the action sequences to a length of 300. The dimension of each action sample is 300 (time steps) $\times$ 150 (coordinates).

We use TensorFlow [63] with TFlearn [64] and run the experiments on either one NVIDIA GTX 1080 GPU or one NVIDIA GTX TITAN X GPU. We train the network using RMSprop [65] optimizer and set learning rate as 0.001, decay as 0.9, and momentum as 0. We train the network from scratch using mini-batches of 1000 sequences for one-layer models and use mini-batches of 650 sequences for two-layer models. For all RNN nets, we use 300 neurons for each single-directional layer, double the amount of neurons for bidirectional layers, and we use a 75% keep probability for dropout. For batch normalization, we initialize γ as 1.0, β as 0.0, and set ϵ as 1×10^{-5}. The estimated means and variances are fixed during inference.

As a comparison, the mentioned parameters are the same for all proposed RNN models, only the structure changes.

2) CNN Implementation: We use the 3D-CNN model [8] in Caffe [66] and train it on RGB frames from the NTU RGB+D dataset, with pretrained parameters [8] from the Sport1M dataset. From RGB videos, we extract the frames, crop and resize them from 1920×1080 pixels to 320×240 pixels [8]. Videos are split into non-overlapped 16-frame clips.

We refer to the input of CNN model as a size of $c \times t \times h \times w$, where c is the number of channels, t is the number of time steps, h and w are the height and width of the frame, respectively. The network takes video clips as input and predicts the 60 action labels which belong to the 60 different actions. It further resizes the input frames to 128×171 pixel resolution. The input dimensions are $3\times16\times128\times171$ pixel. During training we use jittering on the input clips by random cropping them into $3\times16\times112\times112$ pixel. We fine-tune the network with stochastic gradient descent optimizer using mini-batches of 44 clips, with initial learning rate of 0.0001. The learning rate is then reduced by half, when no training progress was observed [65]. The training stopped after around 20 epochs.

For video-based prediction, the model averages the predictions over all 16-frame clips split from the same video and provides the final prediction for the input video. A similar

idea is applied for extracting features from fc-6 layer, which averages the 4096-dimensional feature vectors over all clips in the same video, resulting in one 4096-dimensional vector for each video.

3) Fusion Implementation: We fuse the best RNN structure, 2 layer BI-GRU-BN-DP-H, with the 3D-CNN model, first using decision fusion, then using feature fusion.

For decision fusion, we first extract the softmax output, then search for the fusion parameters, trust weight w_r and w_c for RNN and CNN from the validation split. The parameters are $w_r = 1.00$ and $w_c = 2.88$ for the cross subject setup, and $w_r = 1.00$ and $w_c = 3.02$ for the cross view setup.

For feature fusion, we extract the RNN features (600 dimensions) from the fully-connected layer, and extract CNN features (4096 dimensions) from the fc-6 layer [8]. We then concatenate them into one feature array (4,696 dimensions) and apply L2 normalization. In the end, we have normalized RNN/CNN features from training, validation, and testing splits. We use training and validation splits to find the optimal value of C for linear SVM [67] model. For both cross-subject and cross-view setups, we find that $C = 8.0$ gives the best validation accuracy.

Among all the models in this paper, feature fusion model shows the best testing results. We refer to this model as a two-stream RNN/CNN structure as shown in Fig. 4.

C. Experimental Results and Analysis

The evaluation results are shown in Tab. I. The first 16 rows are skeleton-based methods. The 3D-CNN model (17th row) uses RGB videos as input. The decision fusion (18th row) and feature fusion (19th row) models use the best RNN structure, which is the 2 Layer BI-GRU-BN-DP-H (16th row), and the 3D-CNN (17th row) model.

Tab. I shows that our RNN structure, the 1 Layer LSTM-BN, already outperforms the baseline method part-aware LSTM reported in [2] because batch normalization improves the LSTM model. Adding a dropout procedure reduces overfitting and further improves the results (rows 11, 12). From rows 12 and 13 we can see that the performances of LSTM and GRU cells are similar [68]. GRU is better in the cross-subject test and LSTM is better in the cross-view test. On the other hand, GRU is faster than LSTM both in computational speed and converge rate. As presented in Fig. 5 left, for 1k training steps, the same model performs 5.42% more accurately and takes 20% less computational time when using GRU cells than when using LSTM cells.

The addition of the extra fully-connected layer brings another significant improvement. This increases the complexity of the neural network, which helps the model capture more inherent features from the 3D skeleton data [69]. The recurrent layers before the fully-connected layer can be seen as a temporal feature extractor, which compact input information (dimension 300×150) into 600 dimensions. The latter part of the RNN structure can be considered as a classifier learning to map these 600-dimensional features to 60 different action categories. Altogether, our novel RNN model, 2 Layer BI-GRU-BN-DP-H, outperforms all the other skeleton-based models including ST-LSTM (Tree traversal) + Trust Gate [4].

Nr.	Method	cross subject	cross view
01	Skeleton Quads [2], [9]	38.62%	41.36%
02	Lie Group [2], [10]	50.08%	52.76%
03	FTP Dynamic Skeletons [2], [11]	60.23%	65.22%
04	HBRNN-L [2], [3]	59.07%	63.97%
05	Deep RNN [2]	56.29%	64.09%
06	Deep LSTM [2]	60.69%	67.29%
07	Part-aware LSTM [2]	62.93%	70.27%
08	ST-LSTM (Tree) + Trust Gate [4]	**69.2%**	**77.7%**
09	1 Layer RNN	18.74%	20.27%
10	1 Layer LSTM	60.99%	64.68%
11	1 Layer LSTM-BN	64.07%	71.86%
12	1 Layer LSTM-BN-DP	64.69%	73.48%
13	1 Layer GRU-BN-DP	65.21%	70.36%
14	1 Layer BI-GRU-BN-DP	64.78%	73.12%
15	2 Layer BI-GRU-BN-DP	66.21%	72.46%
16	2 Layer BI-GRU-BN-DP-H	**70.70%**	**80.23%**
17	3D-CNN [8]	**79.75%**	**83.95%**
18	Decision Fusion	82.05%	86.68%
19	Feature Fusion	**83.74%**	**93.65%**

TABLE I: Comparison of testing accuracies on the NTU RGB+D dataset. The first 8 rows are the results reported from other papers. Rows 9 to 19 are results of the methods evaluated in this paper. The first 16 rows are skeleton-based methods. Our 2 Layer BI-GRU-BN-DP-H model outperforms other methods on both evaluation protocols. In addition, the feature fusion model further boosts accuracy by more than 13% on both settings.

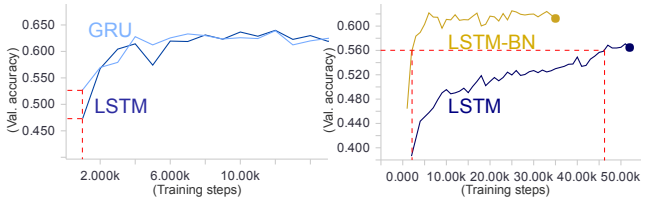

Fig. 5: The left figure shows: For 1k training steps, the model with GRU cells takes 20 m 31 s, reaches 52.75% validation accuracy, while the same model with LSTM cells takes 24 m 27 s and only has 47.33% validation accuracy. The right figure shows: For validation accuracy 56%, LSTM-BN takes 2 k training steps, while LSTM needs 46 k training steps. Batch normalization makes the model converge 13 times faster.

Then, we use the RGB video data to train the 3D-CNN model. We use the voting method based on confidence to fuse the 2 Layer BI-GRU-BN-DP-H and 3D-CNN model. In the next step, we utilize a linear SVM [8] to fuse the fc-6 features from the CNN and the fc features from the RNN. This further improves results by over 13% in comparison to our best RNN, and by more than 14% compared to literature models [2], [4]. This boosting is due to the features from RNN and CNN model being highly complementary. The RNN model uses 50 3D coordinates for two human bodies over 300 time steps, and learns to find the long-term motion pattern. Whereas the CNN model has 2D RGB frames, which additionally has spatiotemporal information about objects, such as cups, pens, and books. However, the CNN model can

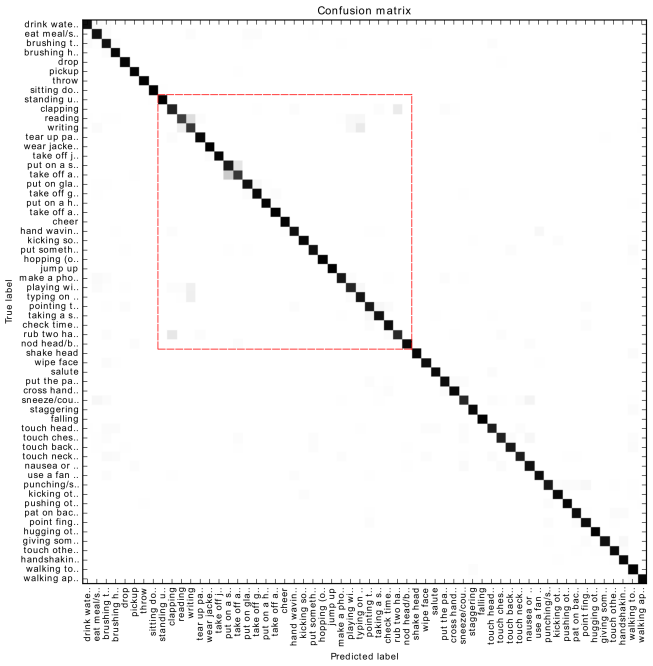

Fig. 6: The confusion matrix of the results from the feature fusion method using the cross-view test. The rectangle area is shown in more detail in Fig. 7.

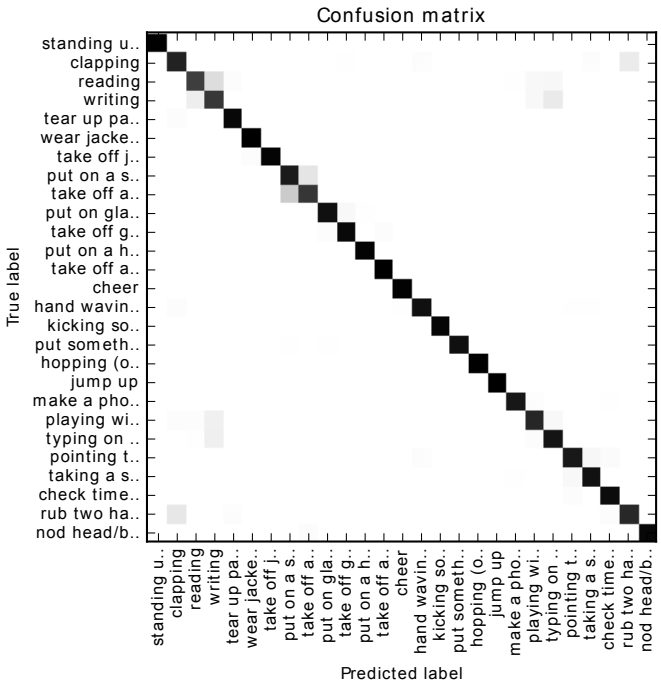

Fig. 7: A part of the confusion matrix shown in Fig. 6. As the figure shows, action pairs such as reading vs. writing, putting on a shoe vs. taking off a shoe, and rubbing two hands vs. clapping, are relatively confusing to the two-stream RNN/CNN structure.

only memorize information for 16 time steps long—longer memorization is prohibited by GPU memory limitations. These facts make the features from RNN and CNN model highly complementary, as the testing results show in row 16, 17, and 19 in Tab. I.

D. Discussion

To better analyze and improve the performance of the model, we take a closer look at actions that are highly confusing to the two-stream RNN/CNN structure. As presented in Fig. 6 and 7, such action pairs include reading vs. writing, putting on a shoe vs. taking off a shoe, and rubbing two hands vs. clapping. These actions are shown in a video at https://www.youtube.com/watch?v=G0PXKCEgIoA. Fig. 8 shows some classified action samples.

There could be several reasons for this observation. First, these actions are sometimes inherent confusing. Secondly, there are flaws in the data. Kinect depth information, from which the NTU skeleton data is created, is quite noisy [70], [71]. Correspondingly, the 3D skeleton data used in our RNNs are also quite noisy [4]. RGB videos data are more accurate and stable, but single frames carry no 3D information. Thirdly, the 3D-CNN model [8] is trained with small video clips, which are 16 time steps long. The CNN model is adapted to find only short-term temporal features in these clips. As GPU memory and computing power increase, the model could also be adapted to find long-term temporal features in each whole video. Lastly, although, the RNN model can memorize the whole action sequences and give final predictions, it has no information about the appearances and movements of surrounding objects, which could be discriminative information for the classification task.

IV. Conclusion and Future Work

In this paper, we propose a novel RNN structure for 3D skeletons that achieves state-of-the-art performance on the largest 3D action recognition dataset. The proposed RNN model can also be trained 13 times faster and saves 20% computational power on each training step. Additionally, the RGB videos from the same dataset are used to finetune a 3D-CNN model. In the end, an efficient fusion structure, two-stream RNN/CNN, is introduced to fuse the capabilities of both RNN and CNN models. The results of this method are 13% higher than using the proposed RNN alone, and 14% higher than the best published result in the literature. In the future, we want to consider using the other sensor modalities such as depth maps and IR sequences and see what is the best architecture to fuse all these modalities.

References

[1] W. Li, Z. Zhang, and Z. Liu, "Action recognition based on a bag of 3D points," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops*. IEEE, 2010, pp. 9–14.

[2] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang, "NTU RGB+D: A large scale dataset for 3D human activity analysis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[3] Y. Du, W. Wang, and L. Wang, "Hierarchical recurrent neural network for skeleton based action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1110–1118.

Fig. 8: Some correctly classified action samples (first four frames with green predictions) and some mis-classified action samples (last two frames with red predictions). These samples are randomly picked from the feature fusion model cross view testing results.

[4] J. Liu, A. Shahroudy, D. Xu, and G. Wang, "Spatio-temporal LSTM with trust gates for 3d human action recognition," *arXiv preprint arXiv:1607.07043*, 2016.

[5] W. Zhu, C. Lan, J. Xing, W. Zeng, Y. Li, L. Shen, and X. Xie, "Co-occurrence feature learning for skeleton based action recognition using regularized deep lstm networks," *arXiv preprint arXiv:1603.07772*, 2016.

[6] K. Wang, X. Wang, L. Lin, M. Wang, and W. Zuo, "3d human activity recognition with reconfigurable convolutional neural networks," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 97–106.

[7] P. Wang, W. Li, Z. Gao, J. Zhang, C. Tang, and P. O. Ogunbona, "Action recognition from depth maps using deep convolutional neural networks," 2015.

[8] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2015, pp. 4489–4497.

[9] G. Evangelidis, G. Singh, and R. Horaud, "Skeletal quads: Human action recognition using joint quadruples," in *International Conference on Pattern Recognition*, 2014, pp. 4513–4518.

[10] R. Vemulapalli, F. Arrate, and R. Chellappa, "Human action recognition by representing 3D skeletons as points in a lie group," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 588–595.

[11] J.-F. Hu, W.-S. Zheng, J. Lai, and J. Zhang, "Jointly learning heterogeneous features for RGB-D activity recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 5344–5352.

[12] H. Rahmani, A. Mahmood, D. Huynh, and A. Mian, "Histogram of oriented principal components for cross-view action recognition," 2016.

[13] S. Gaglio, G. L. Re, and M. Morana, "Human activity recognition process using 3-d posture data," *IEEE Transactions on Human-Machine Systems*, vol. 45, no. 5, pp. 586–597, 2015.

[14] C. Chen, K. Liu, and N. Kehtarnavaz, "Real-time human action recognition based on depth motion maps," *Journal of real-time image processing*, pp. 1–9, 2013.

[15] B. Ni, G. Wang, and P. Moulin, "Rgbd-hudaact: A color-depth video database for human daily activity recognition," in *Consumer Depth Cameras for Computer Vision*. Springer, 2013, pp. 193–208.

[16] J. Sung, C. Ponce, B. Selman, and A. Saxena, "Unstructured human activity detection from rgbd images," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 842–849.

[17] J. Wang, Z. Liu, Y. Wu, and J. Yuan, "Mining actionlet ensemble for action recognition with depth cameras," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 1290–1297.

[18] L. Xia, C.-C. Chen, and J. Aggarwal, "View invariant human action recognition using histograms of 3d joints," in *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, 2012, pp. 20–27.

[19] V. Bloom, V. Argyriou, and D. Makris, "Dynamic feature selection for online action recognition," in *International Workshop on Human Behavior Understanding*. Springer, 2013, pp. 64–76.

[20] Y.-C. Lin, M.-C. Hu, W.-H. Cheng, Y.-H. Hsieh, and H.-M. Chen, "Human action recognition and retrieval using sole depth information," in *Proceedings of the 20th ACM international conference on Multimedia*. ACM, 2012, pp. 1053–1056.

[21] C. Zhang, Y. Tian, and E. Capezuti, "Privacy preserving automatic fall detection for elderly using rgbd cameras," in *International Conference on Computers for Handicapped Persons*. Springer, 2012, pp. 625–633.

[22] O. Oreifej and Z. Liu, "Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 716–723.

[23] H. S. Koppula, R. Gupta, and A. Saxena, "Learning human activities and object affordances from rgb-d videos," *The International Journal of Robotics Research*, vol. 32, no. 8, pp. 951–970, 2013.

[24] F. Negin, F. Özdemir, C. B. Akgül, K. A. Yüksel, and A. Erçil, "A decision forest based feature selection framework for action recognition from rgb-depth cameras," in *International Conference Image Analysis and Recognition*. Springer, 2013, pp. 648–657.

[25] P. Wei, N. Zheng, Y. Zhao, and S.-C. Zhu, "Concurrent action detection with structural prediction," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 3136–3143.

[26] M. Munaro, S. Michieletto, and E. Menegatti, "An evaluation of 3d motion flow and 3d pose estimation for human action recognition," in *RSS Workshops: RGB-D: Advanced Reasoning with Depth Cameras*, 2013.

[27] C. Ellis, S. Z. Masood, M. F. Tappen, J. J. Laviola Jr, and R. Sukthankar, "Exploring the trade-off between accuracy and observational latency in action recognition," *International Journal of Computer Vision*, vol. 101, no. 3, pp. 420–436, 2013.

[28] A. Mansur, Y. Makihara, and Y. Yagi, "Inverse dynamics for action recognition," *IEEE transactions on cybernetics*, vol. 43, no. 4, pp. 1226–1236, 2013.

[29] Z. Yang, L. Zicheng, and C. Hong, "Rgb-depth feature for 3d human activity recognition," *China Communications*, vol. 10, no. 7, pp. 93–103, 2013.

[30] V. Carletti, P. Foggia, G. Percannella, A. Saggese, and M. Vento, "Recognition of human actions from rgb-d videos using a reject option," in *International Conference on Image Analysis and Processing*. Springer, 2013, pp. 436–445.

[31] D. Kastaniotis, I. Theodorakopoulos, G. Economou, and S. Fotopoulos, "Gait-based gender recognition using pose information for real time applications," in *Digital Signal Processing (DSP), 2013 18th International Conference on*. IEEE, 2013, pp. 1–6.

[32] A.-A. Liu, W.-Z. Nie, Y.-T. Su, L. Ma, T. Hao, and Z.-X. Yang, "Coupled hidden conditional random fields for rgb-d human action recognition," *Signal Processing*, vol. 112, pp. 74–82, 2015.

[33] D. Huang, S. Yao, Y. Wang, and F. De La Torre, "Sequential max-margin event detectors," in *European conference on computer vision*. Springer, 2014, pp. 410–424.

[34] I. Lillo, A. Soto, and J. Carlos Niebles, "Discriminative hierarchical modeling of spatio-temporally composable human activities," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 812–819.

[35] G. Yu, Z. Liu, and J. Yuan, "Discriminative orderlet mining for real-time recognition of human-object interaction," in *Asian Conference on Computer Vision*. Springer, 2014, pp. 50–65.

[36] C. Wu, J. Zhang, S. Savarese, and A. Saxena, "Watch-n-patch: Unsupervised understanding of actions and relations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4362–4370.

[37] J.-F. Hu, W.-S. Zheng, J. Lai, S. Gong, and T. Xiang, "Exemplar-based recognition of human–object interactions," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 4, pp. 647–660, 2016.

[38] C. Chen, R. Jafari, and N. Kehtarnavaz, "Utd-mhad: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor," in *Image Processing (ICIP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 168–172.

[39] Z. Cheng, L. Qin, Y. Ye, Q. Huang, and Q. Tian, "Human daily action analysis with multi-view and color-depth data," in *European Conference on Computer Vision*. Springer, 2012, pp. 52–61.

[40] Z. Zhang, W. Liu, V. Metsis, and V. Athitsos, "A viewpoint-independent statistical method for fall detection," in *Pattern Recognition (ICPR), 2012 21st International Conference on*. IEEE, 2012, pp. 3626–3630.

[41] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy, "Berkeley mhad: A comprehensive multimodal human action database," in *Applications of Computer Vision (WACV), 2013 IEEE Workshop on*. IEEE, 2013, pp. 53–60.

[42] S. M. Amiri, M. T. Pourazad, P. Nasiopoulos, and V. C. Leung, "Non-intrusive human activity monitoring in a smart home environment," in *e-Health Networking, Applications & Services (Healthcom), 2013 IEEE 15th International Conference on*. IEEE, 2013, pp. 606–610.

[43] P. Wei, Y. Zhao, N. Zheng, and S.-C. Zhu, "Modeling 4d human-object interactions for event and object recognition," in *2013 IEEE International Conference on Computer Vision*. IEEE, 2013, pp. 3272–3279.

[44] J. Wang, X. Nie, Y. Xia, Y. Wu, and S.-C. Zhu, "Cross-view action modeling, learning and recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2649–2656.

[45] H. Rahmani, A. Mahmood, D. Q. Huynh, and A. Mian, "Hopc: Histogram of oriented principal components of 3d pointclouds for action recognition," in *European Conference on Computer Vision*. Springer, 2014, pp. 742–757.

[46] A.-A. Liu, Y.-T. Su, P.-P. Jia, Z. Gao, T. Hao, and Z.-X. Yang, "Multipe/single-view human action recognition via part-induced multitask structural learning," *IEEE transactions on cybernetics*, vol. 45, no. 6, pp. 1194–1208, 2015.

[47] Y. Song, J. Tang, F. Liu, and S. Yan, "Body surface context: A new robust feature for action recognition from depth videos," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 6, pp. 952–964, 2014.

[48] K. Yun, J. Honorio, D. Chattopadhyay, T. L. Berg, and D. Samaras, "Two-person interaction detection using body-pose features and multiple instance learning," in *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, 2012, pp. 28–35.

[49] T. Hu, X. Zhu, W. Guo, and K. Su, "Efficient interaction recognition through positive action representation," *Mathematical Problems in Engineering*, vol. 2013, 2013.

[50] C. Wolf, E. Lombardi, J. Mille, O. Celiktutan, M. Jiu, E. Dogan, G. Eren, M. Baccouche, E. Dellandréa, C.-E. Bichot *et al.*, "Evaluation of video activity localizations integrating quality and quantity measurements," *Computer Vision and Image Understanding*, vol. 127, pp. 14–30, 2014.

[51] V. Bloom, V. Argyriou, and D. Makris, "G3di: A gaming interaction dataset with a real time detection and evaluation framework," in *Workshop at the European Conference on Computer Vision*. Springer, 2014, pp. 698–712.

[52] C. Van Gemeren, R. T. Tan, R. Poppe, and R. C. Veltkamp, "Dyadic interaction detection from pose and flow," in *International Workshop on Human Behavior Understanding*. Springer, 2014, pp. 101–115.

[53] L. Lin, K. Wang, W. Zuo, M. Wang, J. Luo, and L. Zhang, "A deep structured model with radius–margin bound for 3d human activity recognition," *International Journal of Computer Vision*, pp. 1–18, 2015.

[54] P. Wang, W. Li, Z. Gao, C. Tang, J. Zhang, and P. Ogunbona, "Convnets-based action recognition from depth maps through virtual cameras and pseudocoloring," in *Proceedings of the 23rd ACM international conference on Multimedia*. ACM, 2015, pp. 1119–1122.

[55] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

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

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

[58] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2625–2634.

[59] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in Neural Information Processing Systems*, 2014, pp. 568–576.

[60] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 221–231, 2013.

[61] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[62] V. Nair and 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.

[63] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[64] A. Damien *et al.*, "Tflearn," https://github.com/tflearn/tflearn, 2016.

[65] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural Networks for Machine Learning*, vol. 4, no. 2, 2012.

[66] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 675–678.

[67] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

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

[69] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.

[70] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore, "Real-time human pose recognition in parts from single depth images," *Communications of the ACM*, vol. 56, no. 1, pp. 116–124, 2013.

[71] T. Mallick, P. P. Das, and A. K. Majumdar, "Characterizations of noise in kinect depth images: a review," *IEEE Sensors Journal*, vol. 14, no. 6, pp. 1731–1740, 2014.