



Evaluating fusion of RGB-D and inertial sensors for multimodal human action recognition

Javed Imran¹ · Balasubramanian Raman¹

Received: 22 November 2018 / Accepted: 3 February 2019 / Published online: 12 February 2019
© Springer-Verlag GmbH Germany, part of Springer Nature 2019

Abstract

Fusion of multiple modalities from different sensors is an important area of research for multimodal human action recognition. In this paper, we conduct an in-depth study to investigate the effect of different parameters like input preprocessing, data augmentation, network architectures and model fusion so as to come up with a practical guideline for multimodal action recognition using deep learning paradigm. First, for RGB videos, we propose a novel image-based descriptor called stacked dense flow difference image (SDFDI), capable of capturing the spatio-temporal information present in a video sequence. A variety of deep 2D convolutional neural networks (CNN) are then trained to compare our SDFDI against state-of-the-art image-based representations. Second, for skeleton stream, we propose data augmentation technique based on 3D transformations so as to facilitate training a deep neural network on small datasets. We also propose a bidirectional gated recurrent unit (BiGRU) based recurrent neural network (RNN) to model skeleton data. Third, for inertial sensor data, we propose data augmentation based on jittering with white Gaussian noise along with deep a 1D-CNN network for action classification. The outputs of all these three heterogeneous networks (1D-CNN, 2D-CNN and BiGRU) are combined by a variety of model fusion approach based on score and feature fusion. Finally, in order to illustrate the efficacy of the proposed framework, we test our model on a publicly available UTD-MHAD dataset, and achieved an overall accuracy of 97.91%, which is about 4% higher than using each modality individually. We hope that the discussions and conclusions from this work will provide a deeper insight to the researchers in the related fields, and provide avenues for further studies for different multi-sensor based fusion architectures.

Keywords Human action recognition · Deep learning · Convolutional neural network · Recurrent neural network · Multimodal fusion

1 Introduction

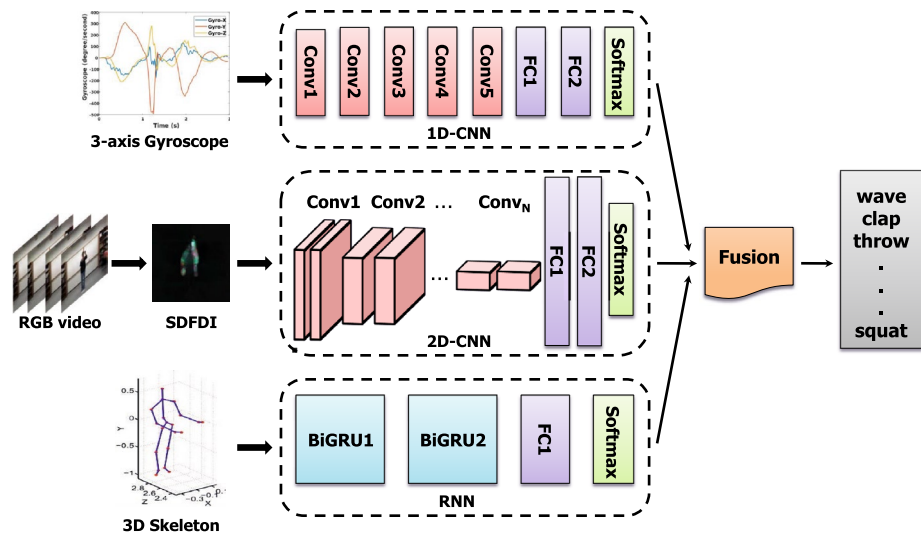
Human action recognition is a hot topic in the area of computer vision research as it has wide variety of application in surveillance, video indexing and retrieval, human–computer interactions, health monitoring, intelligent systems, and similar other domains (Chikhaoui et al. 2017; Yan et al. 2018a, b). There have been several attempts in the past to recognize actions using vision based RGB and Depth cameras (Donahue et al. 2015; Satyamurthi et al. 2018; Shahroudy

et al. 2016; Simonyan and Zisserman 2014; Wang and Wang 2017; Wang et al. 2016a, b; Zhang et al. 2017), as well as using wearable inertial sensors like accelerometer (Altun and Barshan 2010; Ermes et al. 2008; Lefebvre et al. 2013; Li et al. 2018a, 2009; Roy et al. 2016; Sarcevic et al. 2017). Though most of these approaches work quite well, yet they have their own limitations. For instance, RGB and depth cameras can effectively capture the high-dimensional visual and depth information of the scene. But they suffer from various drawbacks like occlusion (both self-occlusion and occlusion from other objects) as well as high dimensional feature space, thus, limiting their performance for real-time action recognition. At the same time, the actor needs to be present in the field of view (FOV) of the camera, so as to record and recognize different activities correctly. Wearable devices, on the other hand, are both view and occlusion invariant, but can only capture low-dimensional sensor

✉ Javed Imran
javed.dcs2016@iitr.ac.in
Balasubramanian Raman
balarfma@iitr.ac.in

¹ Department of Computer Science and Engineering, Indian Institute of Technology Roorkee, Roorkee, India

Fig. 1 Our proposed three-stream architecture consisting of RGB, inertial and skeleton streams. The input to inertial stream is 3-axis gyroscope signal. For RGB stream, each video clip is first converted into SDFDI, and then given as input. For skeleton stream, 3D joint coordinates acts as the input feature vector. Finally, the outputs of all the streams are combined by late fusion to predict the final class label



data from accelerometer, gyroscope, etc. Moreover, very few attempts (Chen et al. 2016; Delachaux et al. 2013; Gasparini et al. 2016; Liu et al. 2014) have been made in the past for fusion of RGB, depth and inertial sensor data for human activity recognition.

During last several years, deep neural networks have shown extraordinary performance in a number of computer vision tasks because of their ability to learn hierarchal features implicitly. CNNs have been successfully applied to image classification (He et al. 2016; Krizhevsky et al. 2012), image segmentation (Bi et al. 2018; Li et al. 2018b; Long et al. 2015), object detection (Girshick et al. 2014), object recognition (Zhou et al. 2018), facial expression recognition (Gogić et al. 2018; Yu et al. 2017) and video classification (Simonyan and Zisserman 2014) tasks, while RNNs are more suited to modeling sequential data like image captioning (Jiang et al. 2018; Vinyals et al. 2015; Xu et al. 2015), language translation (Sutskever et al. 2014), video analysis (Alahi et al. 2016; Deng et al. 2016; Donahue et al. 2015) and 3D action recognition (Ma et al. 2018; Shahroudy et al. 2016; Wang and Wang 2017; Zhang et al. 2017). Besides this, two-stream and multi-stream deep networks (Feichtenhofer et al. 2016; Li et al. 2016; Simonyan and Zisserman 2014; Wang et al. 2016a, b; Zhao et al. 2017) have significantly outperformed the handcrafted based methods (Wang and Schmid 2013; Wang et al. 2011) in the field of video classification task. Therefore, this paper also proposes a three-stream architecture which intends to extract complementary features from RGB, skeleton and inertial streams for multimodal action recognition. To the best of our knowledge, this is the first attempt for fusion of RGB, skeleton and inertial data using deep neural networks. Our proposed architecture is shown in Fig. 1. It consists of three different types of neural networks: 1D-CNN for inertial sensor gyroscope data, 2D-CNN for RGB videos, and a two-layer

BiGRU for skeleton stream. In order to come up with the best architecture, we investigate the performance of various CNN and RNN architectures on validation set by incorporating suitable data augmentation techniques to prevent overfitting as much as possible. Finally, the output of all the three streams is combined by late fusion using either softmax score fusion or feature fusion.

To summarize, we made the following five contributions:

- A novel three-stream architecture is proposed for multimodal action recognition.
- A novel feature descriptor called Stacked dense flow difference image (SDFDI) is proposed for RGB videos.
- A systematic approach is presented to find the best neural network corresponding to each type of input modality.
- An in-depth analysis is performed to compare the performance of different fusion schemes based on softmax score and feature fusion. A statistical feature fusion technique based on canonical correlation analysis (CCA) is also proposed.
- We achieve state-of-the-art results on publicly available UTD-MHAD dataset.

In Sect. 2, we present the related work in the area of action recognition using RGB, skeleton and inertial sensor data. Section 3 presents the basic overview of CNN, RNN and Multi-stream fusion. Section 4 describes the details of proposed approach. The experimental results and discussions are described in Sect. 5. Finally, Sect. 6 concludes this paper.

2 Related work

In this section, we review the related work in the area of action recognition using RGB, skeleton and inertial data.

2.1 Action recognition using RGB data alone

RGB videos are the most common source of input for human action recognition due to a wide variety of available video datasets like UCF101 (Soomro et al. 2012) and ActivityNet (Caba Heilbron et al. 2015). Several handcrafted features (Sargano et al. 2017) have been proposed for video classification task. Among them, improved dense trajectories proposed by Wang and Schmid (2013) gives the best result. However, after the breakthrough performance of deep neural networks for image classification (Krizhevsky et al. 2012), CNNs have been successfully applied for video classification as well. Karpathy et al. (2014) proposed a one-stream network for action recognition in videos. Consecutive frames are given as input to the network, and then features are combined at different levels using late, early and slow fusion. However, this approach lacks learning the motion features, which are absolutely necessary for modeling any action recognition task. Simonyan and Zisserman (2014) explicitly model the motion information by adding a temporal-stream based on stacked optical flow vectors, along with spatial-stream based on single frame. The output of both the streams is then combined by late fusion. Donahue et al. (2015) added a long short-term memory (LSTM) unit on top of a CNN network, and proposed an end-to-end network for video classification. Tran et al. (2015) proposed a C3D network to simultaneously learn spatial and temporal features from raw RGB frames by extending 2D convolutional and pooling operation in temporal domain. However, the amount of temporal information learnt by C3D network was limited to 16 frames only. Feichtenhofer et al. (2016) improved the two-stream architecture by proposing fusion at multiple levels. Wang et al. (2016a) proposed a temporal segment network (TSN) along with some good practices like pre-training, use of batch normalization and better data augmentation techniques. To learn long-term temporal information, Varol et al. (2018) proposed a network with long-term temporal convolutions. They experimented with different temporal resolutions ranging from $t = 16$ –100, and found that network accuracy increases with increasing value of t .

One common element in most of the aforementioned work is the use of stacked dense optical flow to learn short-term motion information. Our proposed RGB stream differs from these methods by stacking difference of optical flows, which enables learning global motion information for the entire video sequence.

2.2 Action recognition using skeleton data alone

With the release of depth sensors like Kinect, along with deep neural networks like CNN and RNN, a large boost in recognition accuracy is achieved in action recognition task. Skeleton stream can be assumed as temporal data, so

most of the previous attempts have used RNN or its variants like LSTM for recognizing actions using skeleton data. Chikhaoui et al. (2017) extracted joint-based and body-based features from skeleton data, and then combine them using ensemble learning based rotation forests. Du et al. (2015) divided the human skeleton into five different parts according to the human physical structure, and separately fed them into five LSTMs. The representations from each subnet are then hierarchically fused so as to obtain the higher level representation. Zhu et al. (2016) proposed an LSTM network which can learn co-occurrence features of skeleton joints directly. They also incorporated dropout regularization scheme within the LSTM unit so as to facilitate the training of deep model. Shahroudy et al. (2016) proposed a part-aware LSTM (P-LSTM) where the local dynamics of five body parts (torso, two hands, and two legs) are independently modeled. The global information of the entire body structure is obtained by concatenating these five different memory cells. Zhang et al. (2017) extracted various geometrical features from skeleton data to train a 3-layer LSTM network. The feature based on distance between joints and selected lines resulted in sparse distribution of weights learned by the first layer of LSTM, and gives the best recognition accuracy. Liu et al. (2016) proposed a spatio-temporal LSTM (ST-LSTM) to model the skeleton stream based on previous frames as well as neighboring joints. They also added trust-gate mechanism within the LSTM cell so as to handle the noisy 3D coordinates without affecting the long-term joint dependencies. Wang and Wang (2017) proposed a two-stream RNN network to model the spatial and temporal dynamics of the skeleton data. They also proposed data augmentation techniques based on 3D transformation of skeletons so as to avoid overfitting.

Different from above mentioned approaches, we use Gated Recurrent Unit (GRU) for implementing our skeleton stream. GRU has fewer number of parameters than LSTM (as discussed in Sect. 3.2), thus overfits less. Moreover, we apply GRU in both forward and backward directions to process the input skeletons. This bi-directional GRU approach performs better than traditional one-directional LSTM models.

2.3 Action recognition using inertial data alone

Inertial sensors like data like accelerometers, gyroscopes and magnetometers have been actively used in the past in areas like action classification (Altun and Barshan 2010; Ordóñez and Roggen 2016), gesture recognition (Lefebvre et al. 2013; Li et al. 2018a), fall detection (Li et al. 2009), sports activities (Ermes et al. 2008) and aggressive behavior recognition (Chikhaoui et al. 2017, 2018). Ermes et al. (2008) proposed a decision tree and artificial neural network based hybrid classifier for recognizing daily and sports activities. They

recorded data both in supervised and unsupervised mode, and found that higher recognition rate is achieved when both supervised and unsupervised data is used for activity recognition. Altun and Barshan (2010) recorded different daily and sports activities using eight body worn inertial sensors. A comparative study is then performed by using different classifiers like Bayesian decision, k-nearest neighbor, SVM, etc. Li et al. (2009) presented a fall detection system based on static postures and dynamic transition between these postures. Chikhaoui et al. (2018) proposed an aggressive behavior recognition system using accelerometer data. The first extracted several statistical features like mean, variance, entropy, etc., and then feature reduction is performed using non-negative matrix factorization method. Rotation random forest based ensemble method is finally used for classification. Sarcevic et al. (2017) build a wireless prototype system to monitor human movement using different inertial sensors. An extensive study is conducted on multiple datasets so as to explore the affect of different feature extraction and processing techniques on the recognition accuracy. Ordóñez and Roggen (2016) proposed a deep Convolutional LSTM network for wearable sensor based activity recognition. They also demonstrated how fusion of more than one sensor can increasing recognition accuracy. In the field of gesture recognition, Lefebvre et al. (2013) proposed a Bidirectional LSTM for 3D gesture recognition using six dimensional values from accelerometer and gyroscope data. Similar to this, Li et al. (2018a) proposed a fisher discriminant based Bidirectional LSTM (F-BiLSTM) and Bidirectional GRU (F-BiGRU) for gesture recognition using mobile devices. They augmented traditional softmax loss function with fisher criterion, and achieved better recognition accuracy than plain BiLSTM or BiGRU.

Although, as discussed above, RNN and its variants like LSTM and GRU have given encouraging results for inertial sensor based activity recognition, yet we have used a 1D-CNN network in our inertial stream. The reason for doing this is to increase diversity of deep models used in our three-stream architecture.

2.4 Action recognition using RGB, skeleton and inertial data

In order to develop a more robust activity recognition system, fusion of RGB, depth and inertial sensor is also reported in the literature. Delachaux et al. (2013) proposed an indoor activity recognition strategy by combining skeleton and accelerometer data using one-vs-all neural networks. Liu et al. (2014) proposed a hand gesture recognition system using accelerometer, gyroscope and skeleton joint coordinates as feature vectors, followed by HMM based probabilistic classification. Gasparrini et al. (2016) proposed a fall detection system by combining variation in skeleton joint

position and acceleration magnitude. Chen et al. (2016) extracted features from depth, skeleton and inertial sensor data, and then fed it to collaborative representation classifier. Using decision level fusion, an accuracy of 93.7% is obtained on a dataset with ten action classes. Zhang et al. (2018) used 3D-CNN to extract spatio-temporal features from RGB and depth data for hand gesture recognition. The features obtained are then fused together followed by classification using SVM. Zhao et al. (2017) designed a two-stream RNN/CNN architecture based on Bidirectional GRU and 3D-CNN (Tran et al. 2015). Using NTU-RGBD dataset (Shahroudy et al. 2016), RNN stream is used to model skeleton data, while 3D-CNN stream is finetuned using RGB modality.

However, all the aforementioned approaches combine either RGB with skeleton data or inertial with skeleton data for multi-modal action recognition. In Sect. 4, we present a three-stream deep neural network for combining RGB, skeleton and inertial data into a true multi-modal multi-stream fusion framework.

3 Overview of CNN, RNN and multi-stream fusion

To make this paper self-contained, in this section we briefly review the convolutional neural network (CNN) and recurrent neural network (RNN), based on which our three-stream architecture is built. We also discuss various fusion strategies to combine the outputs of multi-stream network.

3.1 Convolutional neural network

CNN is a feedforward artificial neural network with multiple hidden layers such as convolutional, pooling, normalization and fully connected layers. A convolutional layer consists of a set of learnable filters, capable of learning a particular type of feature in the given input. Pooling layer is used to reduce the spatial size of the representation, thus reducing the workload of successive layers. For normalization, Batch Normalization (BN) (Ioffe and Szegedy 2015) is the most commonly used layer, as it helps to achieve faster convergence by reducing internal covariate shift. Finally, one or more fully connected layers are added at the end of a CNN to learn high-level reasoning from the features extracted by the previous layers.

3.1.1 2D-CNN

Table 1 shows the details of state-of-the-art 2D-CNNs along with their top-1 accuracy on ImageNet dataset (Deng et al. 2009). InceptionV3 (Szegedy et al. 2016), proposed by Google, is the first 2D-CNN with more than 100 layers. Such

Table 1 Comparison of recognition accuracy of different state-of-the-art 2D-CNNs on ImageNet dataset (Deng et al. 2009)

Model	Size (MB)	Accuracy (%)	Parameters (M)	Depth
MobileNet (Howard et al. 2017)	16	70.4	4.2	88
ResNet50 (He et al. 2016)	99	74.9	25.6	168
InceptionV3 (Szegedy et al. 2016)	92	77.9	23.8	159
Xception (Chollet 2017)	88	79.0	22.9	126
InceptionResNetV2 (Szegedy et al. 2017)	215	80.3	55.8	572

is deep architecture is accomplished by using the concept of *Inception* module (Fig. 2a), which is based on 1×1 , 3×3 and 5×5 convolutions. Such large variety of convolutions facilitates learning *global* as well as *local* information throughout an input image. Further, a large reduction in the number of parameters is achieved by using *averaging pooling* instead of traditional fully connected layers. Apart from this, He et al. (2016) from Microsoft Research, proposed ResNet50 architecture based on *residual learning*. As shown in Fig. 2b, a residual connection is used between the input \mathbf{x} and the output $\mathcal{F}(\mathbf{x})$, as it easier to optimize this residual mapping $\mathcal{F}(\mathbf{x}) + \mathbf{x}$ than the original unreferenced mapping. This is because a residual connection helps to propagate gradients both in forward and backward direction, making the *vanishing gradient* problem less severe. In terms of performance, InceptionResNetV2 (Szegedy et al. 2017) has the highest accuracy of 80.3% on ImageNet dataset, as it incorporates the advantages of both Inception and ResNet architecture. But the downside is the large number of parameters, making it unsuitable for memory constraint environments.

Recently, Howard et al. (2017) proposed a 2D-CNN called MobileNet, specifically designed for mobile vision applications. The architecture of MobileNet is based on *depthwise separable convolutions* (Chollet 2017), in which the traditional 3×3 convolution operation is split into a 3×3 depthwise convolution and a 1×1 pointwise convolution. Such a formulation leads a significant reduction in the number of parameters, while decreasing the network accuracy only marginally.

3.1.2 1D-CNN

A 1D-CNN is a CNN in which all the operations are performed in one-dimension. For instance, a 1D convolution operation can be assumed as the dot product between weight vector $\mathbf{w} \in \mathbb{R}^m$ and input vector $\mathbf{x} \in \mathbb{R}^s$, where m is the size of the filter and s denotes the dimension of the input sequence.

$$o_i = f\left(\sum_i w_i \cdot x_i + b\right) \tag{1}$$

Here, b is the bias and o_i is the output of convolution operation, which is followed by a non-linear activation function f (such as ReLU) and pooling operation (like max pooling). Furthermore, a batch normalization layer can also be added

after every pooling operation in order to achieve faster convergence. However, since there are no pre-trained 1D-CNNs available, so in our proposed approach (Sect. 4), a novel 1D-CNN with five convolutional blocks is presented for inertial sensor stream.

3.2 Recurrent neural network

3.2.1 Vanilla recurrent neural network

Recurrent neural network (RNN) is a class of artificial neural network which can map an input sequence X to another

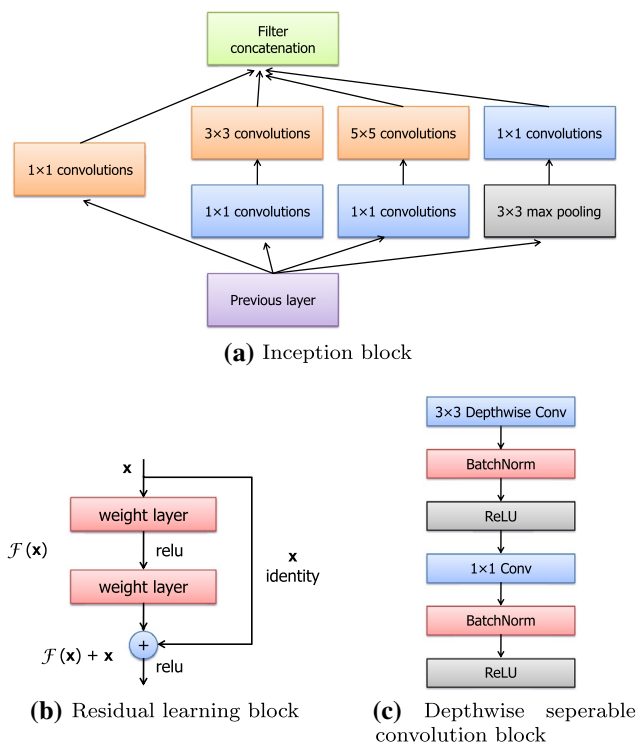


Fig. 2 Basic building blocks of various state-of-the-art 2D-CNNs. **a** InceptionV3 is based on 1×1 , 3×3 and 5×5 convolutions. Such large variety of convolutions facilitates learning global as well as local information. **b** Resnet50 employs a residual connection between the input \mathbf{x} and the output $\mathcal{F}(\mathbf{x})$. Such a configuration helps the gradient to propagate to deeper layers. **c** MobileNet architecture is based on *depthwise separable convolutions*, in which the traditional 3×3 convolution operation is split into a 3×3 depthwise convolution and a 1×1 pointwise convolution

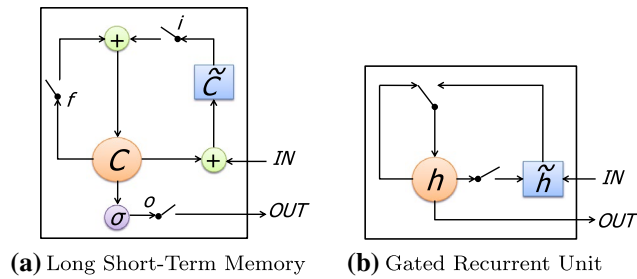


Fig. 3 Left: An LSTM block with input (i), output (o), and forget (f) gates. Right: A GRU block with update (z) and reset (r) gates

output sequence Y . RNNs implements a form of memory which makes them suitable to process sequences of arbitrary length. The current hidden state h_t and output y_t of an RNN cell is generated as

$$h_t = F(W_h h_{t-1} + U_h x_t + b_h), \quad (2)$$

$$y_t = F(W_y h_t + b_y), \quad (3)$$

where x_t is the current input, h_{t-1} is the previous hidden state, and W , U and b are parameter matrices and biases. F is the activation function (like ReLU or sigmoid) to provide non-linearity to the system.

However, vanilla RNN suffers from vanishing gradient problem, making them unsuitable for storing long range of information. To overcome this limitation, two variations of RNN, namely LSTM (Hochreiter and Schmidhuber 1997) and GRU (Cho et al. 2014) are proposed in the literature.

3.2.2 Long short-term memory

As shown in Fig. 3a, an LSTM unit contains following components:

- memory cell (c_t): stores a value/state for either long or short period of time,
- input gate (i_t): controls the flow of new information into the cell,
- output gate (o_t): controls the flow of information out of the cell,
- forget gate (f_t): determines when to forget the contents corresponding to the internal state of the cell.

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f), \quad (4)$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i), \quad (5)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o), \quad (6)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma(W_c x_t + U_c h_{t-1} + b_c), \quad (7)$$

$$h_t = o_t \circ \sigma(c_t). \quad (8)$$

3.2.3 Gated recurrent unit

GRU, which is a simplified version of LSTM, performs almost at the same level (or even better), especially for small datasets. As shown in Fig. 3b, it is obtained by combining three gates of LSTM into two gates: update gate (z_t) and reset gate (r_t). Mathematically, the forward pass of GRU can be written as

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z), \quad (9)$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r), \quad (10)$$

$$h_t = (1 - z_t) \circ h_{t-1} + z_t \circ \sigma(W_h x_t + U_h (r_t \circ h_{t-1}) + b_h). \quad (11)$$

3.3 Multi-stream fusion

Fusion of multi-stream networks is a crucial step in building any multi-modal system. The main idea is to exploit the complementary information extracted from different modalities, and combine that in such a manner so as to improve the overall recognition performance. The most common technique of achieving this is to apply late fusion using either Score fusion or Feature fusion.

3.3.1 Score fusion

Softmax score fusion is the most commonly used technique for combining the results of multi-stream networks (Khaire et al. 2018; Simonyan and Zisserman 2014; Wang et al. 2016b). Kitter et al. (1998) proposed a theoretical framework for combining the scores obtained from different classifiers using methods like sum rule, product rule and max rule. Let the posterior probability generated by softmax layer of i -th stream for a given input \mathbf{x}_i belonging to class ω_j be $P(\omega_j | \mathbf{x}_i)$. Let $c \in \{1, 2, \dots, m\}$ be the class to which the input \mathbf{x} is finally assigned. Then, the different rules of score fusion can be defined as follows:

- *Sum rule:*

$$c = \operatorname{argmax}_j \sum_{i=1}^n P(\omega_j | \mathbf{x}_i). \quad (12)$$

- *Product rule:*

$$c = \operatorname{argmax}_j \prod_{i=1}^n P(\omega_j | \mathbf{x}_i). \quad (13)$$

- *Max rule:*

$$c = \operatorname{argmax}_j \max_i P(\omega_j | \mathbf{x}_i), \quad (14)$$

where ω_j denotes the j -th class. The main advantage of score fusion based rules is their fast and easy calculation, without requiring training an external classifier.



Fig. 4 Samples of RGB videos. The top two rows shows *swipe left* and *baseball swing* actions from UTD-MHAD dataset (Chen et al. 2015), while bottom two rows shows *cutting in kitchen* and *clean and jerk* actions from UCF101 dataset (Soomro et al. 2012). It is

evident that both *cutting in kitchen* and *clean and jerk* can be easily distinguished using spatial information like *knife* and *dumbbells*, respectively. However, no such discriminative spatial cue is present for actions present in top two rows

3.3.2 Feature fusion

Another alternative approach to combine outputs of multi-stream network is to extract features from fully-connected layers, combine them, and feed them to any classical machine learning classifier like linear support vector machine (LSVM) or kernel extreme learning machine (KELM). Let $f_i \in \mathbb{R}^d$ be the d -dimensional feature vector extracted from the last fully connected layer of the i -th stream for an input Z . Then all these feature vectors can be combined together either by stacking them horizontally as $F = [f_1, f_2, \dots, f_n]$ or by taking their average $F = \frac{1}{n} \sum_{i=1}^n f_i$, where n is the number of streams. Furthermore, to scale all the dimensions of resulting feature vector F within a common range, $L2$ -normalization is also commonly performed before training a classifier. The fusion results obtained using this approach generally performs better than score fusion.

4 Methodology

As shown in Fig. 1, our proposed approach can be divided into four main components: (1) RGB stream using 2D-CNN, (2) Inertial stream using 1D-CNN, (3) Skeleton stream using RNN, and (4) Three-stream network fusion.

4.1 RGB stream using 2D-CNN

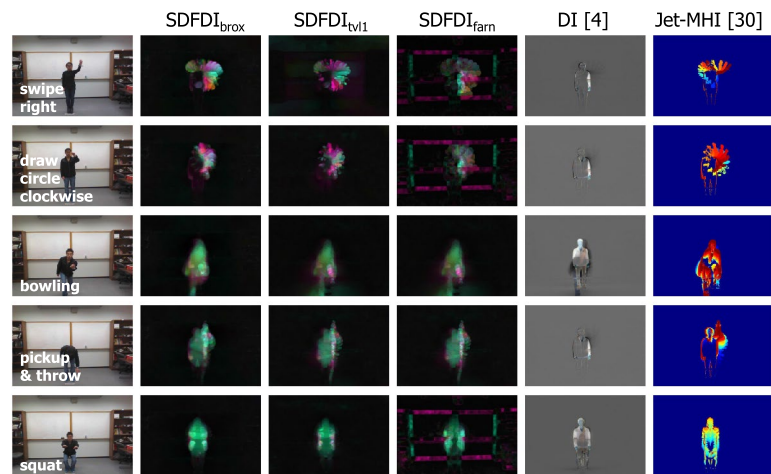
Our proposed RGB stream is composed of three stages. First, conversion of input video into a single image-based representation called SDFDI. Second, data augmentation to generate more training samples. And third, fine-tuning a 2D-CNN using generated SDFDIs.

4.1.1 RGB video to image-based representation

The input to RGB stream is traditional 3-channel RGB videos captured using Kinect camera. Contrary to the video samples present in large video action recognition datasets like UCF101 (Soomro et al. 2012) or ActivityNet (Caba Heilbron et al. 2015), the input videos of our RGB stream differs in two manners. First, our video clips are of very short duration (typically in the range of 2–8 s). Second, these clips do not contain any spatial information (Simonyan and Zisserman 2014) which could be an important source of discriminative cue for activity classification. For instance, as shown in Fig. 4, certain activities like *cutting in kitchen* and *clean and jerk* can be easily recognized from the presence of *knife* and *dumbbells*, respectively, in a video frame.

Based on the above two observations, we argue that RGB stream should be built using *global temporal* information only. In deep learning literature (Feichtenhofer et al. 2016; Simonyan and Zisserman 2014; Wang et al. 2016a), temporal-stream based on stacked dense optical flow fields has given state-of-the-art performance for video action recognition. To be more specific, if u_i^x and u_i^y represents the horizontal and vertical components of the optical flow field calculated using a pair of two consecutive frames i and $i + 1$, then the motion across a sequence of L consecutive frames can be modeled by stacking the flow field u_i^{xy} to form a total of $2L$ input channels. But such a formulation has two main drawbacks. First, the amount of temporal information captured is limited by the number of frames L to form stacked dense optical flow. For instance, a small value of $L=10$ [used in (Feichtenhofer et al. 2016; Simonyan and Zisserman 2014)] cannot model actions with long temporal extents (Wang et al. 2016a). Second, all the standard 2D-CNNs use 3-channel RGB input image. So fine-tuning a pretrained 2D-CNN require changing the dimension of input layer from $w \times h \times 3$ to $w \times h \times 2L$, where w and h are height and width of input video sample.

Fig. 5 Comparison of proposed SDFDI against state-of-the-art video to image based representations



To overcome the aforementioned drawbacks, we propose a novel RGB descriptor called stacked dense flow difference image (SDFDI), capable of representing global temporal information using differences of adjacent dense optical flow fields. Formally, if \mathbf{d}_i represents the optical flow image constructed by stacking horizontal component u_i^x as first channel, vertical component u_i^y as second channel and magnitude $\sqrt{(u_i^x)^2 + (u_i^y)^2}$ as the third channel, then SDFDI is calculated as

$$SDFDI = \sum_{i=2}^{n-1} i \times |\mathbf{d}_i - \mathbf{d}_{i-1}|. \quad (15)$$

The detailed formulation of constructing an SDFDI is given in Algorithm 1. The horizontal and vertical optical flow components computed in Step 1 can be obtained using any one of the commonly used methods like Brox (Brox et al. 2004), TV-L1 (Zach et al. 2007) or Farneback (Farneback 2003) algorithm. The resulting SDFDI (see Fig. 5) is a

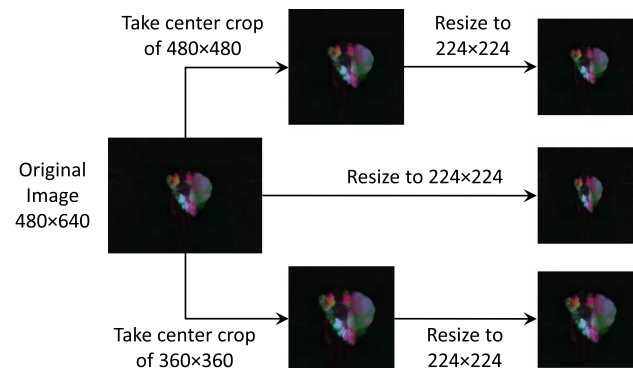


Fig. 6 Illustration of data augmentation for RGB stream. From each SDFDI of size 480×640 , three SDFDI of size 224×224 , thereby increasing the training set size by 3 times

3-channel image, and can be directly used to fine-tune any pre-trained 2D-CNN.

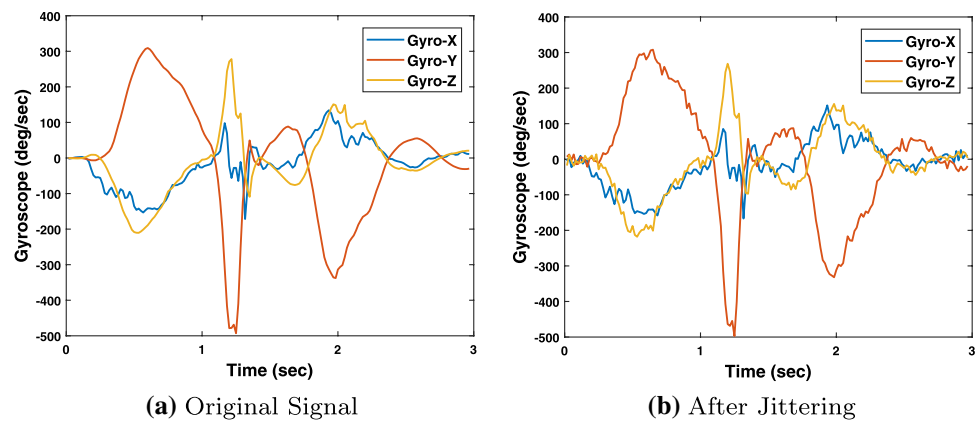
Algorithm 1 Generation of Stacked Dense Flow Difference Image (SDFDI)

Input: A video sample V with n frames: f_1, f_2, \dots, f_n

Output: SDFDI of input video sample

- 1: for each pair of consecutive frames f_i and f_{i+1} , extract horizontal flow component u_i^x and vertical flow component u_i^y
 - 2: Compute $mag = \sqrt{(u_1^x)^2 + (u_1^y)^2}$
 - 3: Compute first *optical flow image* \mathbf{d}_1 by stacking u_1^x as first channel, u_1^y as second channel, and mag as third channel.
 - 4: Initialize $SDFDI = 0$
 - 5: **for** $i = 2$ to $n - 1$ **do**
 - 6: Compute $mag = \sqrt{(u_i^x)^2 + (u_i^y)^2}$
 - 7: Compute next *optical flow image* \mathbf{d}_2 by stacking u_i^x as first channel, u_i^y as second channel, and mag as third channel.
 - 8: $SDFDI = SDFDI + i * |\mathbf{d}_2 - \mathbf{d}_1|$
 - 9: $\mathbf{d}_1 = \mathbf{d}_2$
 - 10: **end for**
 - 11: **return** $SDFDI$
-

Fig. 7 Left: original gyroscope signal. Right: signal obtained after jittering with Gaussian noise



4.1.2 Data augmentation

Data augmentation is a necessary step when training a deep neural network. For our RGB stream, each of the generated SDFDI is augmented based on the traditional data augmentation techniques proposed for image and video classification (Krizhevsky et al. 2012; Wang et al. 2016a). Figure 6 shows the three types of augmentation. In each case, the final size is kept as 224×224 so as to match the input size of standard pre-trained 2D-CNNs. It is also worth to mention here that in all the three types of data augmentation, no horizontal flipping is done. This is important as there are certain pair of actions (like *draw circle clockwise* and *draw circle anti-clockwise*) which are mirror image of each other. Flipping images horizontally would lead to confusion in recognizing such pair of actions.

4.1.3 Fine-tuning a 2D-CNN

The final step is to use any pre-trained 2D-CNN and fine-tune it using the generated SDFDI. Fine-tuning prevents overfitting and also reduces the training time considerably. In our proposed RGB stream, we evaluated all the state-of-the-art pretrained CNNs listed in Table 1 so as to come up with the best network configuration. We add two fully connected (FC) layers and a Softmax layer at the end of global average pooling (GAP) layer of each model. For direct classification using CNN, posterior probabilities generated by softmax layer are used; for feature extraction, output of the last fully connected layer is extracted.

4.2 Inertial stream using 1D-CNN

The inertial stream can consist of accelerometer, gyroscope, magnetometer or similar other sensor data captured using wearable devices like smartphones, smartbands, etc. Our inertial stream is comprised of two steps. First, proposing an appropriate data augmentation scheme, and second, proposing a 1D-CNN for network training.

4.2.1 Data augmentation

In our proposed approach, we have utilized only 3-axis gyroscope data as input to the inertial stream, where each axis can be assumed as a one-dimensional signal. Analogous to *color jittering* (Krizhevsky et al. 2012) data augmentation technique for image dataset (where an image can be assumed as a two-dimensional signal), we propose a *signal jittering* based data augmentation for inertial data. Specifically, as given in Algorithm 2, each axis of gyroscope data (denoted by G) is jittered with white Gaussian noise based on an input jitter *factor* and the calculated *smallestDifference* equal to the minimum difference between adjacent values for that corresponding axis (Chambers et al. 1983). Figure 7 illustrates the result obtained after signal jittering. It can be observed that although slight perturbations are added to the original signal, yet the resulting signal still belongs to the original class. In our proposed inertial stream, we repeat this process three times so as to triple the size of our training dataset.

Algorithm 2 Data augmentation for Inertial stream

Input: 3-axis gyroscope data G with m timesteps ($G \in \mathbb{R}^{m \times 3}$), jitter factor ($factor$)

Output: A vector $G' \in \mathbb{R}^{m \times 3}$ obtained after jittering G with white Gaussian noise.

```

1: for  $i = 1$  to 3 do
2:    $data = G(1 : m, i)$  // read the  $i$ th axis of  $G$  for all timesteps
3:    $dataUnique = unique(sort(data))$  // sort  $data$  and find unique values
4:    $dataDifferences = diff(dataUnique)$  // calculate differences between adjacent elements of  $dataUnique$ 
5:    $smallestDifference = min(dataDifferences)$  // find the smallest difference
6:    $scaleFactor = 0.2 * factor * smallestDifference$ 
7:    $G'(1 : m, i) = data + scaleFactor * randn(m)$ 
8: end for
9: return  $G'$ 

```

4.2.2 1D-CNN network

Since sensor output can be assumed to be a time-series data, so the most obvious choice to model it is to use an RNN or its variant like LSTM. However, in our implementation, a 1D-CNN is employed to model inertial sensor data. The main reason for doing this is to increase the diversity of deep networks used in our proposed three-stream architecture, thus leading to more complementary feature extraction from different modalities.

The 1D-CNN network used in our proposed approach is shown in Fig. 8. An extensive set of experiments are conducted in Sect. 5 to determine the number of convolutional blocks in the proposed 1D-CNN network. The final configuration consists of five convolutional blocks (each made up of a convolutional, batch normalization and ReLU layer) followed by two fully connected layers. The number of filters in each convolutional layer is increased as the power of 2, starting from $2^5 = 32$ filters in first layer, and so on. The kernel size is kept as 3 and stride as 2. Max pooling is done with pool size as 2 and stride as none. A softmax layer, which is basically a combination of a fully connected layer (with neurons equal to number of classes C) and softmax

activation function is placed at the end to obtain the posterior probability P for the c -th class as:

$$P(y = c | \mathbf{x}) = \frac{e^{\mathbf{x}^T \mathbf{w}_c}}{\sum_{i=1}^C e^{\mathbf{x}^T \mathbf{w}_i}}. \quad (16)$$

4.3 Skeleton stream using RNN

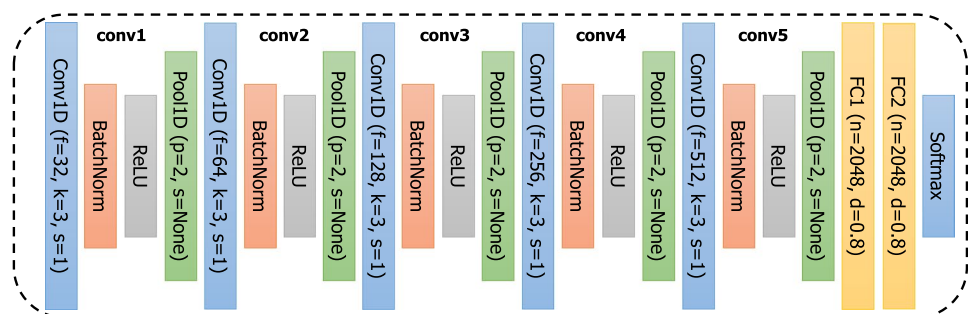
Similar to inertial stream, our proposed skeleton stream is composed of two stages. First, data augmentation using 3D transformations of skeleton joints. Second, proposing an RNN based Bi-directional gated recurrent unit (BiGRU) network for action classification.

4.3.1 Data augmentation

For skeleton stream, 3D rotation transformation is used for data augmentation. Each 3D joint is rotated around x and y -axis by using Euler's rotation theorem

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}, \quad (17)$$

Fig. 8 Proposed 1D-CNN network for inertial stream. There are five convolutional blocks with the following notations: f = no_of_filters, k = kernel_size, s = stride, p = pool_size, n = no_of_neurons, d = dropout_ratio



$$R_y(\gamma) = \begin{bmatrix} \cos\gamma & 0 & \sin\gamma \\ 0 & 1 & 0 \\ -\sin\gamma & 0 & \cos\gamma \end{bmatrix}. \tag{18}$$

By combining (17) and (18) using matrix multiplication as $R = R_x(\theta)R_y(\gamma)$, a general rotation matrix can be obtained to rotate a skeleton sequence around x and y -axis simultaneously. Such a transformation provides view-invariance and leads to higher performance in cross-view experimental settings.

4.3.2 RNN network

Our proposed skeleton stream is based on BiGRU, in which an input sequence is processed both in forward and backward direction. While keeping the number of neurons fixed to 512, we experimented with one, two and three layers network, and found that two layer BiGRU gives the best validation accuracy (Sect. 5). For classification, one fully connected layer is added at the top, followed by a softmax layer.

4.4 Three-stream network fusion

The final step of our proposed method is to combine the complementary information from all the three streams using an efficient fusion framework. We propose two modes of late fusion: (1) Score fusion, and (2) Feature fusion. In score fusion, the softmax scores are combined using either sum, product or max rule as discussed in Sect. 3.3. For feature fusion, 2048-dimensional feature vectors are extracted from the last fully connected layers of all three streams, and then can be combined by concatenating horizontally or by taking their average. However, these methods of feature aggregation do not take into account the statistical correlations between pairwise features from different modalities. The averaging method may reduce the effect of one good feature due to the addition of another one, while concatenating method increases redundancy as well as the dimension of resulting feature vector; ultimately slowing down the training process.

To overcome these limitations, we propose to fuse features extracted from different streams using canonical correlation analysis (CCA) (Sun et al. 2005), so that the resulting feature vector has maximum pair-wise correlation with each other. Let $X \in \mathbb{R}^{p \times n}$ and $Y \in \mathbb{R}^{q \times n}$ are two matrices containing n feature vectors extracted from two different modalities. The covariance matrix of $\begin{pmatrix} X \\ Y \end{pmatrix}$ can be written as:

$$S = \begin{pmatrix} Var(X) & Cov(X, Y) \\ Cov(Y, X) & Var(Y) \end{pmatrix} = \begin{pmatrix} S_{xx} & S_{xy} \\ S_{yx} & S_{yy} \end{pmatrix}, \tag{19}$$

where $S_{xx} \in \mathbb{R}^{p \times p}$ and $S_{yy} \in \mathbb{R}^{q \times q}$ represents the within-set covariance matrices of X and Y , and $S_{xy} \in \mathbb{R}^{p \times q}$ denotes between-set covariance matrix ($S_{yx} = S_{xy}^T$). However, it is possible that these two sets of feature vectors may not follow consistent pattern (Haghighat et al. 2016), leading to difficulty in finding a relationship between them directly through S . CCA aims to find the linear combinations, $X = W_x^T X$ and $Y = W_y^T Y$, that maximize the pair-wise correlations across the two feature sets:

$$Corr(X, Y) = \frac{Cov(X, Y)}{Var(X).Var(Y)}, \tag{20}$$

where $Cov(X, Y) = W_x^T S_{xy} W_y$, $Var(X) = W_x^T S_{xx} W_x$ and $Var(Y) = W_y^T S_{yy} W_y$. After evaluating W_x and W_y , the fused feature vector Z is obtained as:

$$Z = W_x^T X + W_y^T Y = \begin{pmatrix} W_x \\ W_y \end{pmatrix}^T \begin{pmatrix} X \\ Y \end{pmatrix}. \tag{21}$$

Finally, Z is $L2$ normalized, and fed to LSVM or KELM for classification.

5 Experiments

In this section, we evaluate our proposed framework on UTD-MHAD dataset (Chen et al. 2015). We first evaluate the different parameters and network configurations for RGB, inertial and skeleton stream using cross-validation. Then based on the best validation parameters, we conduct experiments on test set and fuse the results using the proposed approach (Sect. 4.4). Finally, the results obtained are compared with the current best methods.

5.1 UTD-MHAD dataset

UTD-MHAD (Chen et al. 2015) is a multimodal human action dataset¹. To the best of our knowledge, this is the only publicly available dataset that contains RGB, depth, skeleton and inertial sensor modalities. Eight subjects (4 males and 4 females) are used to collect 27 different actions with at most 4 repetitions. RGB data contains videos with a resolution of 480 × 640; skeleton data consists of 3D coordinates of 20 joint points; inertial data consists of 3-axis acceleration and 3-axis gyroscope signals. For evaluation (Chen et al. 2015), we used subjects 1, 3, 5, 7 for training and subjects 2, 4, 6, 8 for testing.

¹ This dataset can be downloaded from <http://www.utdallas.edu/~cxc123730/UTD-MHAD.html>.

Table 2 Empirical study of performance of different pre-trained CNN models on our proposed RGB stream

Input representation	Accuracy (%)				
	InceptionV3	Xception	Resnet50	Inception-ResNetV2	MobileNet
DI (Bilen et al. 2016)	62.55	67.67	67.67	69.30	69.53
Jet-MHI (Imran and Kumar 2016)	72.55	73.72	70.69	75.34	73.02
SDFDI _{far}	74.76	74.76	71.49	73.83	75.71
SDFDI _{lv1}	79.53	81.06	83.12	82.79	82.09
SDFDI _{brox}	79.76	81.16	83.25	83.02	83.48

Bold represents the highest result

Among different input representations, SDFDI_{brox} using MobileNet model performs best

5.2 Implementation details

5.2.1 Experimental environment

We conduct our experiments on a PC with Intel Core i5 CPU @ 2.8 GHz \times 4, 16 GB RAM and NVIDIA Geforce GTX 1060 GPU with 6GB VRAM. For data preprocessing we use Matlab R2017b, while Keras library (Chollet 2015) is used for CNN and RNN implementation.

5.2.2 Data augmentation

For all the three input modalities (i.e., RGB, gyroscope and skeleton data), data augmentation is performed in such a manner so as to increase the training set size by three times. In case of RGB data, each training video is first converted into its corresponding SDFDI. To extract dense optical flow during SDFDI generation, OpenCV GPU implementation of Brox, TV-L1 and Farneback algorithms [provided by Wang (2017)] is used. Then as discussed in Sect. 4.1.2, data augmentation is done by generating three 224×224 SDFDIs from the original 480×640 SDFDI. For inertial stream, each gyroscope data is jittered three times with white Gaussian noise (*jitter factor* = 500) using Algorithm 2. For skeleton stream, each skeleton joints is rotated three times in the range $-5^\circ:5^\circ$. Since the original UTD-MHAD dataset contains 431 training samples, so after augmentation, the training set size becomes $431 \times 3 = 1293$ samples. For test set, we did not perform any augmentation, and its size remains as 430.

5.2.3 Network training

For RGB stream, all the 2D-CNN models (listed in Table 1) are trained using Adam optimizer with initial learning rate set to 0.0001. For MobileNet, batch size is kept as 32, while for the rest of the models, a batch size of 16 is used. Regularization is achieved by dropping 80% of the neurons between

FC1 and FC2 layers, and similarly between FC2 and softmax layer. All the networks are optimized within 50 epochs.

For inertial stream, 3-axis gyroscope data is used as input. The input length (L) and the number of layers in 1D-CNN network is fixed by cross-validation. The number of neurons in both the fully connected layers is fixed as 2048. Dropout layer is also added between the fully connected and softmax layer with 80% keep probability. Stochastic Gradient Descent (SGD) with batch size as 16, initial learning rate as 0.0002, momentum as 0.9 and weight decay as 0.0 is used to optimize the network. Training is stopped after 200 epochs.

For skeleton stream, the input frame length (T) and the number of BiGRU layers is fixed by cross-validation. On the top, there is a fully connected layer with 2048 neurons, followed by a softmax layer. To prevent complex co-adaptations, 80% of the neurons are randomly dropped from FC layer during training. The entire RNN stream is trained using Backpropagation Through Time (BPTT) and RMSprop optimizer with learning rate as 0.001, weight decay as 0.0 and batch size as 16. The network is converged within 50 epochs.

5.3 Experimental results

5.3.1 Experiments on RGB data

After encoding each RGB video into an SDFDI, we conduct an exhaustive set of experiments with five different types of pre-trained CNNs: MobileNet (Howard et al. 2017), ResNet50 (He et al. 2016), InceptionV3 (Szegedy et al. 2016), Xception (Chollet 2017), InceptionResNetV2 (Szegedy et al. 2017). Furthermore, we also compare the performance of our proposed SDFDI against other state-of-the-art image-based representations like Dynamic Image (DI) (Bilen et al. 2016) and Jet-colored Motion History Image (Jet-MHI) (Imran and Kumar 2016). Table 2 shows the results obtained for the RGB stream. It is evident that proposed SDFDI clearly outperforms DI and Jet-MHI. Among the three type of dense optical flow algorithm used to generate SDFDI, Brox algorithm performs best due to it

Fig. 9 Confusion matrix for RGB stream. Average accuracy obtained is 83.48%

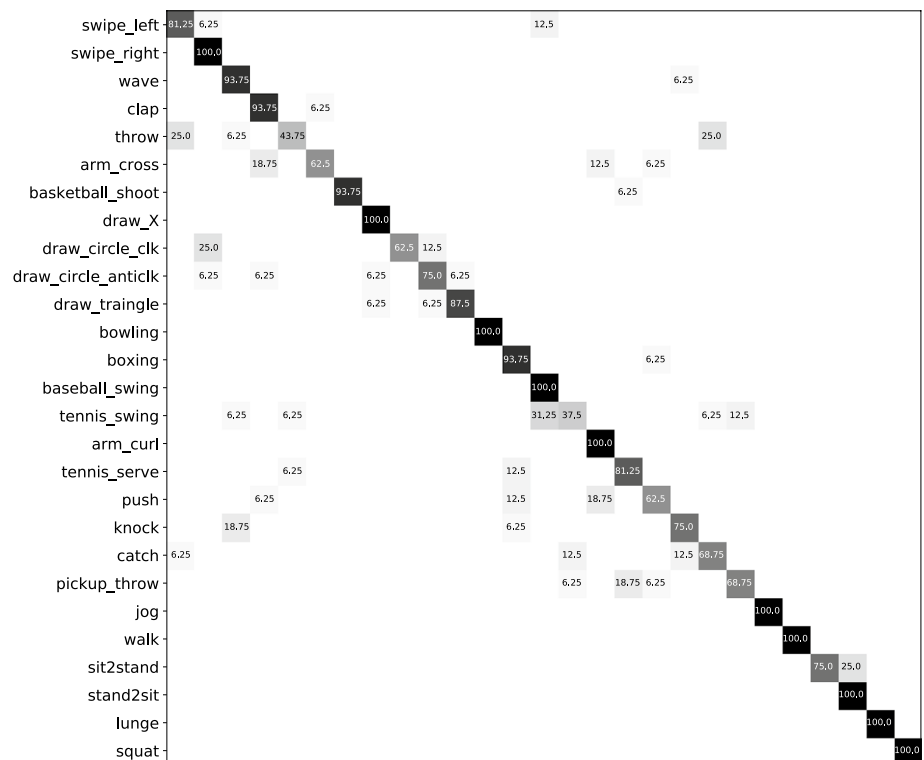


Table 3 Empirical study of different 1D-CNN configurations using fourfold cross-validation

Length (L)	Accuracy (%)	
	C32-64-128-256	C32-64-128-256-512
107	83.76	83.99
180	76.80	80.05
250	75.42	78.42
326	74.48	77.04

Bold represents the highest result

L denotes the accelerometer signal length. C32-64-128-256 denotes four convolutional layers with 32, 64, 128, 256 filters, and similarly C32-64-128-256-512 denotes five convolutional layers with 32, 64, 128, 256, 512 filters

capability to capture motion more accurately (Varol et al. 2018). Finally, on comparing different CNN models, it can be observed that MobileNet generally overfits less due to its fewer number of parameters. So in our final three-stream architecture, we use SDFDI_{brox} with MobileNet as our RGB stream. The corresponding confusion matrix is also shown in Fig. 9. Out of 27 actions, our proposed RGB stream achieves 100% accuracy on 10 actions. The least accuracy is obtained on *tennis swing* and *throw* actions (37.5% and 43.75%, respectively). This is because both of these actions involve strong movement in depth (or z-axis) direction, which is not possible to learn from 2D RGB videos.

Table 4 Test accuracy obtained on inertial stream with and without data augmentation

Inertial stream	Accuracy (%)
w/o Augmentation	83.02
w/Augmentation	86.51

Bold represents the highest result

5.3.2 Experiments on inertial data

The length (L) of 3-axis gyroscope signal present in UTD-MHAD dataset varies from 107 to 326. Since a 1D-CNN network takes fixed length input for batch training, so we performed fourfold cross validation using our training set with four subjects: 1, 3, 5, 7 to find the optimal L . In other words, we select one subject for validation, and remaining three subjects for training and repeat experiments four times. In each fold, L is varied as {107, 180, 250, 326} by random sampling and zero-padding. Further, we also investigated two different network configurations: (1) 1D-CNN with four convolutional blocks (C32-64-128-256), and (2) 1D-CNN with five convolutional blocks (C32-64-128-256-512).

Table 3 shows the results of fourfold cross validation on inertial stream, from which following two conclusions can be drawn:

1. Zero-padding the signal length has negative effect on the accuracy of both the network configurations.

Fig. 10 Confusion matrix for inertial stream. Average accuracy obtained is 86.51%

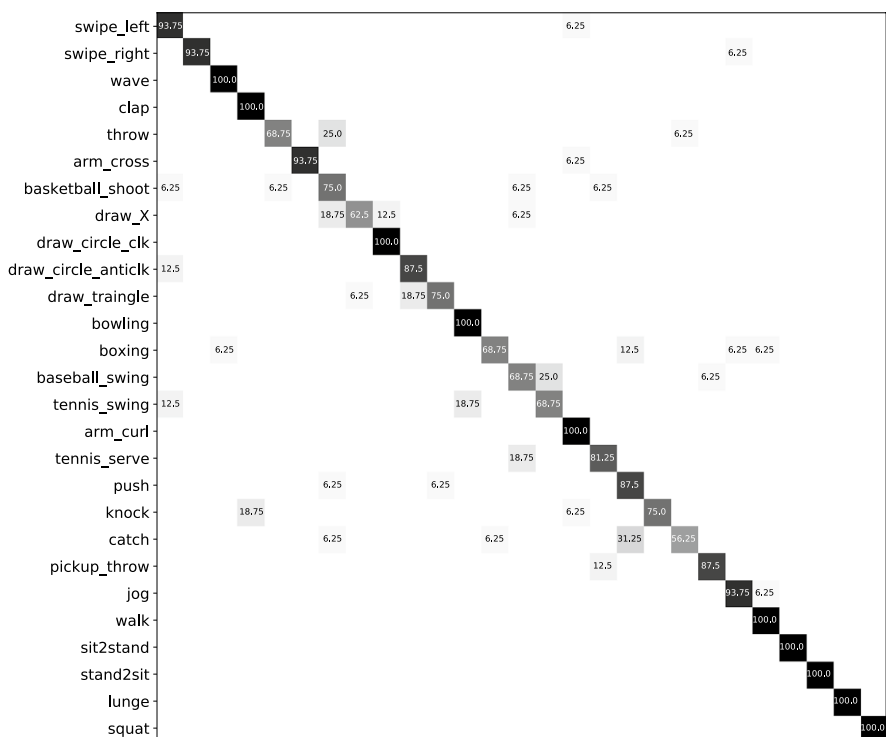


Table 5 Empirical study of different RNN configurations using fourfold cross-validation

#Frames (T)	All joints accuracy (%)			Imp joints accuracy (%)		
	R1-512	R2-512	R3-512	R1-512	R2-512	R3-512
41	90.95	90.04	91.18	88.63	91.88	92.34
70	87.94	87.25	87.71	90.03	89.10	86.79
100	90.03	89.56	90.27	92.33	90.26	92.12
125	90.50	91.65	89.79	91.19	92.35	87.24

Bold represents the highest result

T denotes the frame length. R1-512 denotes one RNN layer with 512 neurons, R2-512 denotes two RNN layers each with 512 neurons, and R3-512 denotes three RNN layers each with 512 neurons

2. 1D-CNN with five convolutional blocks performs better than 1D-CNN with four convolutional blocks.

Based on the above observations, we select five layer network configuration (C32-64-128-256-512) for conducting our experiments. Taking $L = 107$, the size of input feature vector is $3 \text{ (axis)} \times 107 \text{ (signal length)} = 321$. The results obtained are presented in Table 4. Using proposed data augmentation technique, we achieve 86.51% accuracy, which is about 3% higher than without augmentation. The corresponding confusion matrix is also shown in Fig. 10. Out of 27 actions, our proposed inertial stream achieves 100% accuracy on 10 actions. The least accuracy is obtained on *catch* (56.25%) and *draw X* (62.50%) actions, due to their similar 3D rotational movement as *punch* and *draw circle clockwise* actions, respectively.

Table 6 Test accuracy obtained on skeleton stream with and without data augmentation

Skeletonstream	Accuracy (%) using BiLSTM	Accuracy (%) using proposed BiGRU
w/o Augmentation	83.16	89.76
w/Augmentation	86.65	93.48

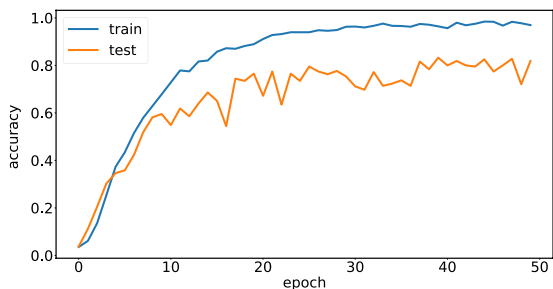
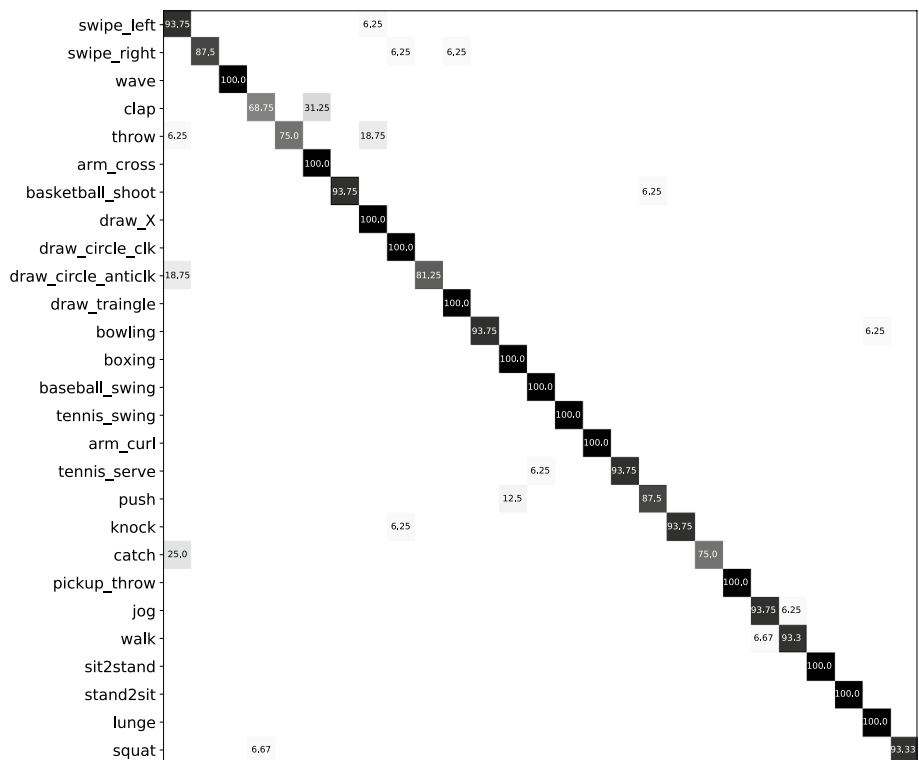
Bold represents the highest result

Based on fourfold cross-validation, frame length T is kept as 125, while an RNN with two layer 512 BiGRU (R2-512) units is used to train the network

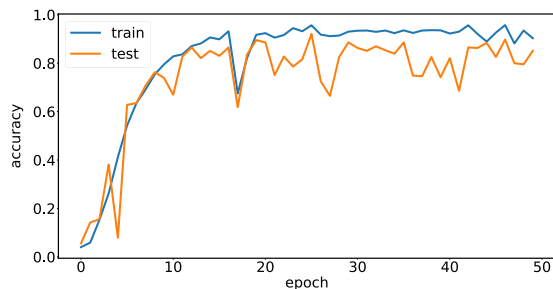
5.3.3 Experiments on skeleton data

The skeleton data consists of 3D coordinates of 20 joints. To find out whether all the joints are important for action classification, we conduct two sets of experiments: (1)

Fig. 11 Confusion matrix for skeleton stream. Average accuracy obtained is 93.48%



(a) Skeleton stream training using BiLSTM



(b) Skeleton stream training using BiGRU

Fig. 12 Performance comparison between BiLSTM (left) and BiGRU (right). After 50 epochs, our proposed BiGRU achieves about 93.48% accuracy, while BiLSTM manages only 86.65% accuracy

with all the 20 joints, and (2) by keeping only 9 most important joints: head, left elbow, left hand, right elbow, right hand, left knee, left foot, right knee and right foot. Further, the minimum and maximum number of skeleton frames (T) in UTD-MHAD dataset varies from 41 to 125. Since RNN takes fixed length input, so similar to inertial stream, we perform 4-fold cross validation and vary $T \in \{41, 70, 100, 125\}$ by random sampling and zero padding. Finally, to come up with best network configuration, we experimented with one, two and three layer BiGRU (denoted as R1-512, R2-512 and R3-512, respectively),

while keeping the number of neurons fixed to 512 in each layer.

Table 5 shows the results of fourfold cross validation on skeleton stream, from which following three conclusions can be drawn:

1. In most cases, zero padding increases the accuracy slightly, with highest accuracy achieved with $T = 125$ frames.
2. Two layer RNN (R2-512) generally performs better than one and three layers RNN.

Table 7 Comparison of two types of late fusion on UTD-MHAD dataset

Row#	Stream	Accuracy (%)
1	RGB	83.48
2	Inertial	86.51
3	Skeleton	93.48
4	RGB + inertial	92.32
5	Inertial + skeleton	96.04
6	RGB + skeleton	96.27
7	Max rule	96.51
8	Product rule	96.97
9	Sum rule	97.20
10	Feature avg with linear SVM	97.44
11	Feature avg with KELM	97.44
12	CCA with linear SVM	97.20
13	CCA with KELM	97.91

Bold represents the highest result

Rows 7, 8 and 9 correspond to Score fusion using max, product and sum rule. Rows 10 and 11 correspond to simple feature averaging followed by classification using Linear SVM and KELM. Rows 12 and 13 shows the result of proposed CCA based fusion followed by classification using Linear SVM and KELM

- Input feature vector based on important joints performs better than all joints. This is due to the fact that out of the total 20 joints, some are redundant, or even noisy, which degrades classification accuracy. Furthermore, retaining only important joints reduces input feature length, resulting in less overfitting.

Based on the above three observations, we select $T = 125$ and number of joints = 9, which yields an input feature vector of length = 3 (axis) \times 9 (joints) \times 125 (frame

length) = 3375. Two layer RNN with 512 neurons is used to conduct experiments on test set, and the results obtained are shown in Table 6. With BiGRU, data augmentation using 3D rotation achieves 93.48% accuracy, which is about 4% higher than without augmentation. The corresponding confusion matrix is shown in Fig. 11. Out of 27 actions, our proposed skeleton stream achieves 100% accuracy on 13 actions. The least accuracy of 75% is obtained for *throw* and *catch* actions, due to their confusion with *draw X* and *swipe left* actions, respectively.

Table 6 also shows the performance comparison between BiLSTM and BiGRU. Our proposed skeleton stream using BiGRU gives about 7% higher result than BiLSTM when the network is trained for 50 epochs (see Fig. 12). This is due to the fact that GRU have lesser number of parameters than LSTM (as discussed in Sect. 3.2), thus leading to lesser overfitting. That is why we have only considered Bidirectional GRU as the RNN model for sensor stream.

5.3.4 Fusion of three streams

Based on the results obtained in Tables 2, 4 and 6, we select MobileNet as 2D-CNN, C32-64-128-256-512 as 1D-CNN and R2-512 as RNN for fusion of RGB, inertial and skeleton streams, respectively. We evaluate different late fusion techniques based on Score and Feature fusion, and the results obtained are presented in Table 7. For Score fusion, softmax scores from all the streams are combined by max rule (row 7), product rule (row 8) and sum rule (row 9). For Feature fusion, 2048-dimensional feature vector is extracted from last fully connected layer of each stream. These three feature vectors are then combined by either performing simple averaging (row 10 and 11) or by using proposed CCA

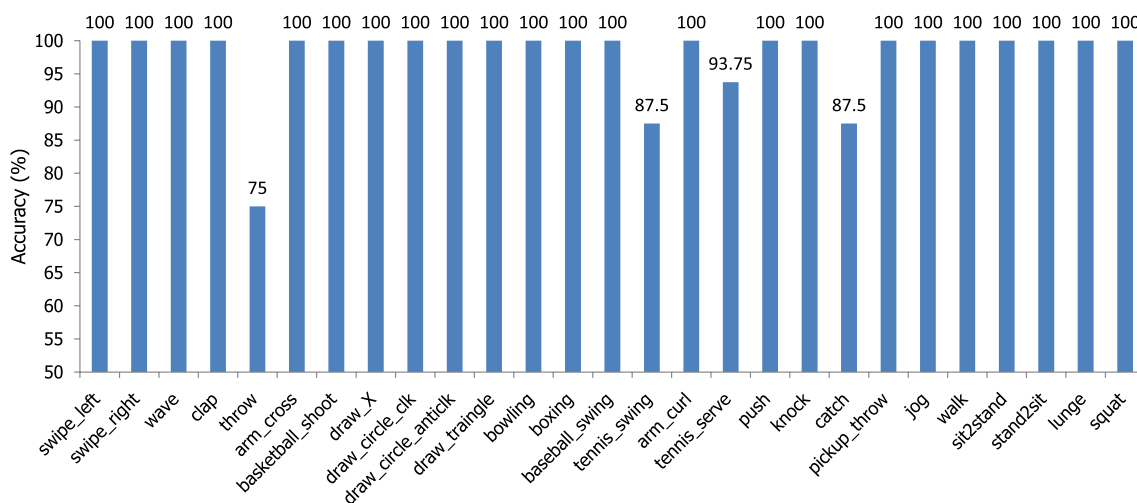
**Fig. 13** Best class-wise accuracy obtained on UTD-MHAD using CCA with KELM (row 13 of Table 7)

Table 8 Comparison of accuracies of the proposed approach with the previous methods on UTD-MHAD dataset

Method	Modality	Accuracy (%)
Chen et al. (2015)	Depth + inertial	79.10
Hussein et al. (2013)	Skeleton	85.58
Wang et al. (2016c)	Skeleton	85.81
Hou et al. (2018)	Skeleton	86.97
Wang et al. (2017)	Depth + skeleton	89.04
Imran and Kumar (2016)	RGB + depth	91.20
El Madany et al. (2016)	Depth + Inertial + skeleton	93.26
Khaire et al. (2018)	RGB + depth + skeleton	95.11
Proposed method	RGB	83.48
	Inertial	86.51
	Skeleton	93.48
	RGB + inertial + skeleton	97.91

Bold represents the highest result

Table 9 Memory requirement of the proposed three-stream architecture

Model	Params (M)	Space (MB)
1D-CNN	5.9	46.7
2D-CNN	9.6	115.2
RNN	8.6	68.3
Total	24.1	230.2

Table 10 Average computational speed of different blocks of our proposed three-stream architecture

Row#	Block	Speed (ms)
1	RGB stream (generation of SDFDI + inference time of 2D-CNN)	4624.06
2	Inertial stream (inference time of 1D-CNN)	0.08
3	Skeleton stream (inference time of RNN)	2.22
4	3-Stream fusion using CCA	2.99
5	Classification using KELM	0.05
6	Total	4629.40

based fusion (row 12 and 13). For classification, we fixed regularization coefficient as $C = 100$ for both LSVM and KELM. It is clear that Feature fusion performs better than Score fusion. The best accuracy (97.91%) is obtained when features are combined with CCA, followed by classification using KELM. We can see that CCA with Linear SVM (row 12) performs lower than Feature averaging with Linear SVM (row 10). This is due to the fact that Linear SVM works well with high dimensional features, while CCA preprocess data with principal component analysis (PCA) to reduce feature

dimension before performing feature fusion. Because of this, row 12 result is lower than row 10 result.

Figure 13 shows the class-wise accuracy obtained using the fusion result obtained in row 13 of Table 7. Out of 27 actions, we achieve 100% accuracy on 23 classes. The least accuracy is obtained on *throw* action because of its similar spatial and temporal variation with many other actions like *catch*, *draw X*, *swipe left*, etc.

5.3.5 Comparison with the state-of-the-art

Table 8 shows the comparison of our three-stream method against other recent results on UTD-MHAD dataset. It is clear that just by using only skeleton data, our proposed RNN stream outperforms most of the previous methods. By combining RGB + inertial + skeleton stream, we achieve 97.91% accuracy, which is about 4% higher than the individual streams. This proves that features learned by our three-stream architecture are highly complementary to each other.

5.3.6 Computational complexity

Table 9 shows the memory requirement of our proposed three-stream architecture. The total number of trainable parameters are 24.1 M, while the space requirement is just 230.2 MB. This shows that our proposed framework can not only be deployable on desktops, but also on mobile devices as well. Further, if we analyze the computational speed given in Table 10, the total inference time of one input sample is 4629.40 ms \approx 4.7 s. The majority of the time is spent in RGB stream, as it involves extraction of computationally expensive dense optical flow for SDFDI generation. However, it can be reduced by replacing dense optical flow with other less accurate technique like Enhanced Motion Vectors (Zhang et al. 2016). Apart from this, our inertial and skeleton stream takes only 0.08 and 2.22 ms, respectively, and thus can be easily implemented in real-time applications.

6 Conclusion

In this paper, we propose a three-stream architecture for fusion of RGB, inertial and skeleton data. We conduct an exhaustive set of experiments, and the conclusions derived can be summarized as follows:

- Proposed SDFDI for RGB video-to-image based representation outperforms current state-of-the-art techniques like DI and Jet-MHI.
- For RGB stream, MobileNet should be employed as the 2D-CNN model because of its less chance of overfitting as well as better inference time.

- To convert all the input sequences to same length for batch processing, random sampling works well for inertial data, while zero-padding should be used for skeleton data.
- Signal jittering using white Gaussian noise proves an effective way for inertial stream data augmentation.
- For skeleton stream, not all body joints are useful for recognition. Important joints, when selected, reduce input feature dimension, as well as give better classification accuracy.
- Incorporating different variety of deep neural networks (like 1D-CNN, 2D-CNN and BiGRU) help to extract more diverse and complementary features.
- Instead of simply averaging or concatenating the feature vectors extracted from different streams, statistical methods like CCA should be employed for maximum pairwise correlation feature fusion.

In future, this work can be extended to other modalities like depth and infrared videos, as well as other application domains like gesture recognition and video surveillance.

Funding No funding source available

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

- Alahi A, Goel K, Ramanathan V, Robicquet A, Fei-Fei L, Savarese S (2016) Social lstm: human trajectory prediction in crowded spaces. In: IEEE conference on computer vision and pattern recognition, pp 961–971
- Altun K, Barshan B (2010) Human activity recognition using inertial/magnetic sensor units. In: Springer international workshop on human behavior understanding, pp 38–51
- Bi L, Feng D, Kim J (2018) Dual-path adversarial learning for fully convolutional network (FCN)-based medical image segmentation. *Vis Comput* 34:1–10
- Bilen H, Fernando B, Gavves E, Vedaldi A, Gould S (2016) Dynamic image networks for action recognition. In: IEEE conference on computer vision and pattern recognition, pp 3034–3042
- Brox T, Bruhn A, Papenberger N, Weickert J (2004) High accuracy optical flow estimation based on a theory for warping. In: Springer European conference on computer vision, pp 25–36
- Caba Heilbron F, Escorcia V, Ghanem B, Carlos Nibbles J (2015) Activitynet: a large-scale video benchmark for human activity understanding. In: IEEE conference on computer vision and pattern recognition, pp 961–970
- Chambers J, Cleveland W, Tukey P, Kleiner B (1983) Graphical methods for data analysis. Wadsworth statistics/probability series
- Chen C, Jafari R, Kehtarnavaz N (2015) Utd-mhad: a multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor. In: IEEE international conference on image processing, pp 168–172
- Chen C, Jafari R, Kehtarnavaz N (2016) Fusion of depth, skeleton, and inertial data for human action recognition. In: IEEE international conference on acoustics, speech and signal processing, pp 2712–2716
- Chikhaoui B, Ye B, Mihailidis A (2017) Feature-level combination of skeleton joints and body parts for accurate aggressive and agitated behavior recognition. *J Ambient Intell Hum Comput* 8(6):957–976
- Chikhaoui B, Ye B, Mihailidis A (2018) Aggressive and agitated behavior recognition from accelerometer data using non-negative matrix factorization. *J Ambient Intell Hum Comput* 9(5):1375–1389
- Cho K, Van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y (2014) Learning phrase representations using rnn encoder-decoder for statistical machine translation. In: Conference on empirical methods in natural language processing, pp 1724–1734
- Chollet F (2015) Keras (online). <https://github.com/keras-team/keras>. Accessed 10 Oct 2018
- Chollet F (2017) Xception: deep learning with depthwise separable convolutions. In: 2017 IEEE conference on computer vision and pattern recognition, pp 1800–1807
- Delachaux B, Rebetez J, Perez-Urbe A, Mejia HFS (2013) Indoor activity recognition by combining one-vs.-all neural network classifiers exploiting wearable and depth sensors. In: Springer international work-conference on artificial neural networks, pp 216–223
- Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L (2009) Imagenet: a large-scale hierarchical image database. In: IEEE conference on computer vision and pattern recognition, pp 248–255
- Deng Z, Vahdat A, Hu H, Mori G (2016) Structure inference machines: recurrent neural networks for analyzing relations in group activity recognition. In: IEEE conference on computer vision and pattern recognition, pp 4772–4781
- Donahue J, Anne Hendricks L, Guadarrama S, Rohrbach M, Venugopalan S, Saenko K, Darrell T (2015) Long-term recurrent convolutional networks for visual recognition and description. In: IEEE conference on computer vision and pattern recognition, pp 2625–2634
- Du Y, Wang W, Wang L (2015) Hierarchical recurrent neural network for skeleton based action recognition. In: IEEE conference on computer vision and pattern recognition, pp 1110–1118
- El Madany NED, He Y, Guan L (2016) Human action recognition via multiview discriminative analysis of canonical correlations. In: IEEE international conference on image processing, pp 4170–4174
- Ermes M, Pärkkä J, Mäntyjärvi J, Korhonen I (2008) Detection of daily activities and sports with wearable sensors in controlled and uncontrolled conditions. *IEEE Trans Inf Technol Biomed* 12(1):20–26
- Farneback G (2003) Two-frame motion estimation based on polynomial expansion. In: Springer scandinavian conference on image analysis, pp 363–370
- Feichtenhofer C, Pinz A, Zisserman (2016) Convolutional two-stream network fusion for video action recognition. In: IEEE conference on computer vision and pattern recognition, pp 1933–1941
- Gasparrini S, Cippitelli E, Gambi E, Spinsante S, Wählslén J, Orhan I, Lindh T (2016) Proposal and experimental evaluation of fall detection solution based on wearable and depth data fusion. In: ICT innovations 2015, Springer, pp 99–108
- Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: IEEE conference on computer vision and pattern recognition, pp 580–587
- Gogić I, Manhart M, Pandžić IS, Ahlberg J (2018) Fast facial expression recognition using local binary features and shallow

- neural networks. *Vis Comput.* <https://doi.org/10.1007/s00371-018-1585-8>
- Haghighat M, Abdel-Mottaleb M, Alhalabi W (2016) Discriminant correlation analysis: real-time feature level fusion for multimodal biometric recognition. *IEEE Trans Inf Forensics Secur* 11(9):1984–1996
- He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: *IEEE conference on computer vision and pattern recognition*, pp 770–778
- Hochreiter S, Schmidhuber J (1997) Long short-term memory. *Neural Comput* 9(8):1735–1780
- Hou Y, Li Z, Wang P, Li W (2018) Skeleton optical spectra-based action recognition using convolutional neural networks. *IEEE Trans Circuits Syst Video Technol* 28(3):807–811
- Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H (2017) Mobilenets: efficient convolutional neural networks for mobile vision applications. [arXiv:1704.04861](https://arxiv.org/abs/1704.04861)
- Hussein ME, Toriki M, Gawayyed MA, El-Saban M (2013) Human action recognition using a temporal hierarchy of covariance descriptors on 3d joint locations. *Int Jt Conf Artif Intell* 13:2466–2472
- Imran J, Kumar P (2016) Human action recognition using rgb-d sensor and deep convolutional neural networks. In: *IEEE international conference on advances in computing, communications and informatics*, pp 144–148
- Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. In: *International conference on machine learning*, pp 448–456
- Jiang T, Zhang Z, Yang Y (2018) Modeling coverage with semantic embedding for image caption generation. *Vis Comput.* <https://doi.org/10.1007/s00371-018-1565-z>
- Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R, Fei-Fei L (2014) Large-scale video classification with convolutional neural networks. In: *IEEE conference on computer vision and pattern recognition*, pp 1725–1732
- Khaire P, Kumar P, Imran J (2018) Combining CNN streams of RGB-D and skeletal data for human activity recognition. *Pattern Recognit Lett* 107–116
- Kittler J, Hatef M, Duin RP, Matas J (1998) On combining classifiers. *IEEE Trans Pattern Anal Mach Intell* 20(3):226–239
- Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: *Neural information processing systems*, pp 1097–1105
- Lefebvre G, Berlemont S, Mamalet F, Garcia C (2013) BLSTM-RNN based 3d gesture classification. In: *Springer international conference on artificial neural networks*, pp 381–388
- Li Q, Stankovic JA, Hanson MA, Barth AT, Lach J, Zhou G (2009) Accurate, fast fall detection using gyroscopes and accelerometer-derived posture information. In: *IEEE sixth international workshop on wearable and implantable body sensor networks*, pp 138–143
- Li Q, Qiu Z, Yao T, Mei T, Rui Y, Luo J (2016) Action recognition by learning deep multi-granular spatio-temporal video representation. In: *ACM international conference on multimedia retrieval*, pp 159–166
- Li C, Xie C, Zhang B, Chen C, Han J (2018a) Deep fisher discriminant learning for mobile hand gesture recognition. *Pattern Recognit* 77:276–276
- Li X, Huang H, Zhao H, Wang Y, Hu M (2018b) Learning a convolutional neural network for propagation-based stereo image segmentation. *Vis Comput.* <https://doi.org/10.1007/s00371-018-1582-y>
- Liu J, Shahroudy A, Xu D, Wang G (2016) Spatio-temporal lstm with trust gates for 3d human action recognition. In: *Springer European conference on computer vision*, pp 816–833
- Liu K, Chen C, Jafari R, Kehtarnavaz N (2014) Fusion of inertial and depth sensor data for robust hand gesture recognition. *IEEE Sens J* 14(6):1898–1903
- Long J, Shelhamer E, Darrell T (2015) Fully convolutional networks for semantic segmentation. In: *IEEE conference on computer vision and pattern recognition*, pp 3431–3440
- Ma C, Wang A, Chen G, Xu C (2018) Hand joints-based gesture recognition for noisy dataset using nested interval unscented kalman filter with LSTM network. *Vis Comput* 34(6–8):1053–1063
- Ordóñez FJ, Roggen D (2016) Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors* 16(1):115
- Roy N, Misra A, Cook D (2016) Ambient and smartphone sensor assisted adl recognition in multi-inhabitant smart environments. *J Ambient Intell Humaniz Comput* 7(1):1–19
- Sarcevic P, Kincses Z, Pletl S (2017) Online human movement classification using wrist-worn wireless sensors. *J Ambient Intell Humaniz Comput* 10:1–18
- Sargano AB, Angelov P, Habib Z (2017) A comprehensive review on handcrafted and learning-based action representation approaches for human activity recognition. *Appl Sci* 7(1):110
- Satyamurthi S, Tian J, Chua MCH (2018) Action recognition using multi-directional projected depth motion maps. *J Ambient Intell Humaniz Comput.* <https://doi.org/10.1007/s12652-018-1136-1>
- Shahroudy A, Liu J, Ng TT, Wang G (2016) NTU RGB+ D: a large scale dataset for 3d human activity analysis. In: *IEEE conference on computer vision and pattern recognition*, pp 1010–1019
- Simonyan K, Zisserman A (2014) Two-stream convolutional networks for action recognition in videos. In: *Advances in neural information processing systems*, pp 568–576
- Soomro K, Zamir AR, Shah M (2012) Ucf101: a dataset of 101 human actions classes from videos in the wild. [arXiv:1212.0402](https://arxiv.org/abs/1212.0402)
- Sun QS, Zeng SG, Liu Y, Heng PA, Xia DS (2005) A new method of feature fusion and its application in image recognition. *Pattern Recognit* 38(12):2437–2448
- Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks. In: *Advances in neural information processing systems*, pp 3104–3112
- Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z (2016) Rethinking the inception architecture for computer vision. In: *IEEE conference on computer vision and pattern recognition*, pp 2818–2826
- Szegedy C, Ioffe S, Vanhoucke V, Alemi AA (2017) Inception-v4, inception-resnet and the impact of residual connections on learning. *AAAI Conf Artif Intell* 4:12
- Tran D, Bourdev L, Fergus R, Torresani L, Paluri M (2015) Learning spatiotemporal features with 3d convolutional networks. In: *IEEE international conference on computer vision*, pp 4489–4497
- Varol G, Laptev I, Schmid C (2018) Long-term temporal convolutions for action recognition. *IEEE Trans Pattern Anal Mach Intell* 40(6):1510–1517
- Vinyals O, Toshev A, Bengio S, Erhan D (2015) Show and tell: a neural image caption generator. In: *IEEE conference on computer vision and pattern recognition*, pp 3156–3164
- Wang H, Schmid C (2013) Action recognition with improved trajectories. In: *IEEE international conference on computer vision*, pp 3551–3558
- Wang H, Wang L (2017) Modeling temporal dynamics and spatial configurations of actions using two-stream recurrent neural networks. In: *IEEE conference on computer vision and pattern recognition*
- Wang H, Kläser A, Schmid C, Liu CL (2011) Action recognition by dense trajectories. In: *IEEE conference on computer vision and pattern recognition*, pp 3169–3176
- Wang L (2017) OpenCV implementation of different optical flow algorithms (online). https://github.com/wanglimin/dense_flow. Accessed 10 Oct 2018

- Wang L, Xiong Y, Wang Z, Qiao Y, Lin D, Tang X, Van Gool L (2016a) Temporal segment networks: towards good practices for deep action recognition. In: Springer European conference on computer vision, pp 20–36
- Wang P, Li W, Gao Z, Zhang J, Tang C, Ogunbona PO (2016b) Action recognition from depth maps using deep convolutional neural networks. *IEEE Trans Hum Mach Syst* 46(4):498–509
- Wang P, Li Z, Hou Y, Li W (2016c) Action recognition based on joint trajectory maps using convolutional neural networks. In: ACM multimedia conference, pp 102–106
- Wang P, Wang S, Gao Z, Hou Y, Li W (2017) Structured images for RGB-D action recognition. In: IEEE international conference on computer vision, pp 1005–1014
- Xu K, Ba J, Kiros R, Cho K, Courville A, Salakhudinov R, Zemel R, Bengio Y (2015) Show, attend and tell: neural image caption generation with visual attention. In: International conference on machine learning, pp 2048–2057
- Yan C, Li L, Zhang C, Liu B, Zhang Y, Zhang Y, Dai Q (2018a) A fast UYGHUR text detector for complex background images. *IEEE Trans Multimed*
- Yan C, Xie H, Chen J, Zhang Y, Dai Q (2018b) Cross-modality bridging and knowledge transferring for image understanding. *IEEE Trans Multimed*
- Yu Z, Liu Q, Liu G (2017) Deeper cascaded peak-piloted network for weak expression recognition. *Vis Comput* 34:1–9
- Zach C, Pock T, Bischof H (2007) A duality based approach for real-time TV-L1 optical flow. In: Springer joint pattern recognition symposium, pp 214–223
- Zhang B, Wang L, Wang Z, Qiao Y, Wang H (2016) Real-time action recognition with enhanced motion vector CNNs. In: IEEE conference on computer vision and pattern recognition, pp 2718–2726
- Zhang S, Liu X, Xiao J (2017) On geometric features for skeleton-based action recognition using multilayer LSTM networks. In: IEEE winter conference on applications of computer vision, pp 148–157
- Zhang Z, Tian Z, Zhou M (2018) Handsense: smart multimodal hand gesture recognition based on deep neural networks. *J Ambient Intell Humaniz Comput*. <https://doi.org/10.1007/s12652-018-0989-7>
- Zhao R, Ali H, van der Smagt P (2017) Two-stream RNN/CNN for action recognition in 3d videos. In: IEEE international conference on intelligent robots and systems, pp 4260–4267
- Zhou F, Hu Y, Shen X (2018) Msanet: multimodal self-augmentation and adversarial network for RGB-D object recognition. *Vis Comput*. <https://doi.org/10.1007/s00371-018-1559-x>
- Zhu W, Lan C, Xing J, Zeng W, Li Y, Shen L, Xie X (2016) Co-occurrence feature learning for skeleton based action recognition using regularized deep LSTM networks. *AAAI Conf Artif Intell* 2:8

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.