Available online at www.sciencedirect.com

ScienceDirect

Procedia Manufacturing 00 (2019) 000–000

Procedia MANUFACTURING

www.elsevier.com/locate/procedia

48th SME North American Manufacturing Research Conference, NAMRC 48, Ohio, USA

Transferable Two-stream Convolutional Neural Network for Human Action Recognition

Qianqian Xiong[a], Jianjing Zhang[a], Peng Wang[b], Dongdong Liu[a,c], Robert X. Gao[a] *

[a] *Department of Mechanical and Aerospace Engineering, Case Western Reserve University, Cleveland, OH 44106, USA*
[b] *Department of Electrical and Computer Engineering and Department of Mechanical Engineering, University of Kentucky, Lexington, KY 40506, USA*
[c] *School of Mechanical Electronic and Control Engineering, Beijing Jiaotong University, Beijing 10004, China*

* Corresponding author. Tel.: +1-216-368-6045. *E-mail address:* Robert.Gao@case.edu

Abstract

Human-Robot Collaboration (HRC), which envisions a workspace in which human and robot can dynamically collaborate, has been identified as a key element in smart manufacturing. Human action recognition plays a key role in the realization of HRC as it helps identify current human action and provides the basis for future action prediction and robot planning. Despite recent development of Deep Learning (DL) that has demonstrated great potential in advancing human action recognition, one of the key issues remains as how to effectively leverage the temporal information of human motion to improve the performance of action recognition. Furthermore, large volume of training data is often difficult to obtain due to manufacturing constraints, which poses challenge for the optimization of DL models. This paper presents an integrated method based on optical flow and convolutional neural network (CNN)-based transfer learning to tackle these two issues. First, optical flow images, which encode the temporal information of human motion, are extracted and serve as the input to a two-stream CNN structure for simultaneous parsing of spatial-temporal information of human motion. Then, transfer learning is investigated to transfer the feature extraction capability of a pretrained CNN to manufacturing scenarios. Evaluation using engine block assembly confirmed the effectiveness of the developed method.

Peer review under the responsibility of the scientific committee of NAMRI/SME

Keywords: Human-robot Collaboration; Transfer learning; Temporal Information

1. Introduction

Traditionally, robots in manufacturing have only been programmed to do static, repeated tasks. Robots and human workers are strictly separated due to safety concerns. As the manufacturing industry is transforming itself into Industry 4.0, robots are increasingly required to achieve higher levels of cooperation and communication with humans in order to meet the emerging demand of flexibility, efficiency and safety in smart manufacturing [1].

Recently, human-robot collaboration (HRC) has emerged as a key component for flexible and intelligent manufacturing. Instead of strict separation between human and robot, HRC allows them to work as a team to collaboratively finish the same tasks in the same workspace. During this process, robots can assist human workers to reduce the workload and improve efficiency as well as reliability in manufacturing, while human workers can help robots to complete higher-level, nondeterministic tasks.

An HRC system consists of four basic elements: perception, recognition, prediction and action, as shown in Fig. 1. First, sensors monitor the manufacturing workspace and supply data (e.g. video footage) for the analysis of human actions. Then, human actions are recognized from the perceived data. Next, by analyzing the temporal information from the on-going action sequence, future human action is predicted. Finally, based on the anticipated future action, robot is able to assist human worker in a pro-active manner. As the first step after obtaining the sensing data, human action recognition plays a crucial role in HRC as it provides the basis for the subsequent action prediction and robot planning, and it has been an active research field in the scientific community. The main focus of the paper is to recognize human actions in manufacturing setting from video footage, which has been the predominant sensing modality in research on HRC.

Peer review under the responsibility of the scientific committee of NAMRI/SME

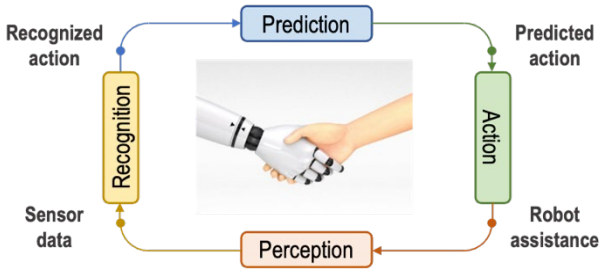


Fig. 1. Flowchart of HRC

Human action recognition traditionally involves two steps: feature extraction and action classification. For feature extraction, one of the most widely used methods is Scale Invariant Feature Transform (SIFT) [2]. In SIFT, a series of local feature vectors, which are invariant to transformation, are generated from the image to characterize human action. Similar methods, such as Speeded Up Robust Features (SURF) and Oriented FAST and rotated BRIEF (ORB) were also widely investigated [3, 4]. The other common approach is to utilize shape and contour of different human poses, by matching the captured video frames with a pre-constructed model. For example, Belongie *et al*. introduced a shape context descriptor in [5], and it is able to detect similar shapes in images. Skeleton model provides another method for human action characterization in which the information of human pose is reduced to the position and orientation of key body joints [6]. For action classification, Hidden Markov Model (HMM) has been widely investigated as a classifier which takes into account the uncertainty and variation embedded in human action. In [7], HMM has been used to analyze the 3D depth information and the developed model is able to characterize both human motion and human object interactions. Support Vector Machine (SVM) is another commonly used technique for classification tasks, which finds a hyperplane that provides the largest margin of separation for action-related image features [8, 9].

More recently, to overcome the limitation of the manual feature extraction step that is subjective by nature, Deep Learning (DL) has emerged as a new paradigm that is capable of automatically learning features from data in a supervised manner. Many real-time action recognition tasks have been adapted into DL framework and convolutional neural networks (CNN) has been the foremost choice of DL architecture due to its image analysis capability [10, 11]. Chaudhary *et al*. [12] integrated CNN and the Weber Motion History Images (WMHI) to realize human action recognition. Ijjina *et al*. [13] applied CNN to automatically learn the discriminative features from RGB-D videos and confirmed the robustness of the method. To realize online action recognition and take into account the non-stationary environment, Ullah *et al*. [14] used an optimized deep autoencoder (DAE) to extract information w.r.t. the temporal environment changes.

Despite the progress that has been made in human action recognition, some limitations remain. First, the conventional, single-stream CNN structure, which only receives one type of input, cannot simultaneously parse both spatial-temporal information of human action. Although this problem can be alleviated by using frame stacks as network input, the results were shown to be inferior than those methods using hand-crafted features. The second limitation is that the DL-based method requires large amount of training data for network weights optimization. However, training data is often difficult to obtain in manufacturing environment, due to the constraints such as continuous production scheduling and often, only a small amount of data can be accessed.

To tackle these limitations, a transferable two-stream CNN architecture (spatial and temporal) is proposed in this paper. First, optical flow is investigated to extract the temporal information directly from videos and a two-stream CNN structure is designed to parse both spatial and temporal information for action recognition. Then, transfer learning is investigated to pretrain a two-stream CNN model on a large-scale open source human action dataset, and then transfer the pretrained model to recognize human actions in an assembly setting by leveraging the feature extraction capability obtained from the large scale, pretraining data. Finally, t-Distributed Stochastic Neighbor Embedding (t-SNE) is investigated to evaluate the performance of the developed method, by visualizing the separation of learned features.

The rest of the paper is organized as follows. Section 2 presents the theoretical foundation for the developed method, including optical flow, CNN and t-SNE. Experimental evaluation and results discussion are presented in Section 3, and conclusions are drawn in Section 4.

2. Theoretical Foundation

In this section, theoretical background of the approaches utilized in this research is introduced. First, Section 2.1 illustrates the theory of optical flow, followed by the design of two-stream CNN structure in Section 2.2. Section 2.3 presents transfer learning and the principles of t-SNE is introduced in Section 2.4.

2.1. Optical Flow

Videos consist of a large amount of information in the form of spatial-temporal pixel intensity variation. In general, it is not straightforward to estimate the movement of every pixel in a sequence of frames. On the other hand, human workers and the object being handled are often the only moving objects in a manufacturing environment. Therefore, the ideal method to encode the temporal information in action is to eliminate the static background and only track the moving part.

Optical flow is defined as the apparent movements of pixels in a frame sequence. As shown in Fig. 2, it describes a field pointing to where each pixel can be found in the next frame. Therefore, optical flow marks out the regions of the moving object as well as the velocity [15]. Fig. 3 shows the typical color-coding scheme to represent the velocities of pixels in different directions.

(a) (b)

Fig. 2. Sample still frame (a) and optical flow (b) [16]

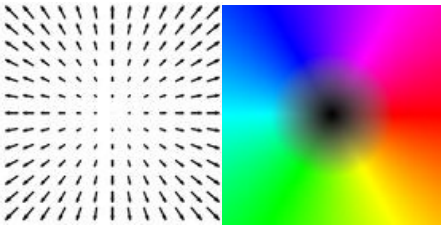


Fig. 3. Optical flow color-coding [17]

The optical flow algorithm calculates the pixel displacement vectors between two consecutive frames that are taken Δt apart. The corresponding pixels in two consecutive frames (before and after the movement) have the same intensity, and their locations are denoted as (x, y) and $(x + \Delta x, y + \Delta y)$, respectively. Mathematically, this *Brightness Constancy Constraint* is expressed as:

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t) \tag{1}$$

Assuming both the time interval Δt and movement $\Delta x, \Delta y$ are small, this constraint can be represented by *Taylor Series*:

$$I(x + \Delta x, y + \Delta y, t + \Delta t) = I(x, y, t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t + \varepsilon \tag{2}$$

where ε is a small number defined as the remainder of the series. Based on Eq. (1) and (2), the following equation can be derived:

$$\frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t = 0 \tag{3}$$

By denoting $\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}, \frac{\partial I}{\partial t}$ as I_x, I_y, I_t, Eq. (3) is rewritten as:

$$I_x \Delta x + I_y \Delta y = -I_t \Delta t \tag{4}$$

In general cases, the neighbours of a pixel can be assumed to move at the same velocity. For example, the 3 × 3 region around the target pixel can be assumed to have the same displacement between the two consecutive frames. Therefore, by writing Eq. (4) for each pixel in the 3 × 3 region, the following set of equations can be obtained:

$$\begin{bmatrix} I_x(p1) & I_y(p1) \\ I_x(p2) & I_y(p2) \\ \vdots & \vdots \\ I_x(p9) & I_y(p9) \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = -\Delta t \begin{bmatrix} I_t(p1) \\ I_t(p2) \\ \vdots \\ I_t(p) \end{bmatrix} \tag{5}$$

Equation (5) can be solved using traditional least square method. By solving Eq. (5) for all 3 × 3 regions in two consecutive frames, optical flow images can be obtained.

2.2. Two-stream CNN

Two-stream CNN was first proposed by Simonyan *et al.* [18] in which each stream consists of a series of hierarchically arranged convolutional layers for image feature extraction [19]. Specifically, the feature extraction step is achieved through sequential convolution between the kernels at each layer and feature maps produced in the preceding layer. For the l^{th} layer with M input feature maps and N kernels, the j^{th} output feature map x_j^l can be calculated as:

$$x_j^l = f\left(\sum_{i=1}^{M} x_i^{l-1} * k_{ij}^l + b_j^l\right), j=1,\ldots,N \tag{6}$$

where x_i^{l-1} represents the i^{th} input feature map, k_{ij}^l denotes the j^{th} kernel to convolve with the i^{th} input feature map, b_j^l is the bias term, f denotes a non-linear function and * denotes the convolution operation.

After convolution, a pooling layer is often implemented as a sub-sampling operation [10]. Max pooling and average pooling are the two most common types of the pooling operation. Max pooling selects the maximum feature value from each local region and discarding the rest, while average pooling computes the mean feature value within each local region. Both methods can reduce the dimensionality of the extracted features and thus improving computational efficiency. Furthermore, they have also shown to reduce feature's sensitivity to small variations, such as pixel intensity change, and improve feature robustness [10]. Mathematically, the output feature maps of the l^{th} layer after pooling can be computed as

$$x_j^l = f\left(\beta_j^l down\left(x_j^{l-1}\right) + b_j^l\right), j=1,\ldots,M \tag{7}$$

where $down()$ is the sub-sampling function.

Through sequential operations of convolution and pooling, image features are gradually distilled to reflect the most relevant information to the specific task (e.g. human action recognition) [10]. At the end of each stream, the respective features are concatenated cross-stream before progressively reducing the dimension through fully-connected layers before a softmax layer carries out the classification.

In the context of human action recognition, the two-stream CNN consists of the spatial stream and temporal stream as shown in Fig. 4. The spatial stream recognizes the spatial information from still frames, such as the appearance of workspace and human pose. The static appearance and human pose can provide useful clue for action recognition. For example, the specific position of human body in the workspace may strongly be associated with certain actions than the others, while the specific human pose may indicate the object the worker is handling. The architecture of the spatial stream is essentially a static image classifier, and it will be pretrained using a static image dataset in the presented research.

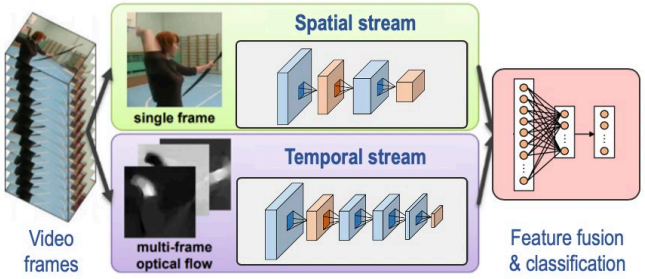


Fig. 4. Structure of two-stream Convolutional Neural Network [18]

The input of temporal stream consists of a stack of consecutive optical flow frames describing a series of movements during a time period of fixed duration. By observing the movement change, the temporal information can be extracted and complement the spatial information for more accurate human action recognition. The temporal stream will be pretrained using optical flow dataset processed from human action videos in the presented research.

To determine the network parameters of the two-stream CNN, parameter grid search is performed. By comparing the

classification accuracy of the tasks described in Section 3, the selected combinations of network parameters for each stream are shown in Table 1 and Table 2, respectively. The dimensionality of the fused feature from both streams is 30 × 40 × 64 (i.e. concatenation of two 30 × 40 × 32 features).

Table 1. Structure of spatial stream

Layer	Kernel Size	Stride	Output size
Conv1 (ReLU)	5 × 5(16)	1	120 × 160 × 16
Max Pool	-	2	60 × 80 × 16
Conv2 (ReLU)	3 × 3(32)	1	60 × 80 × 32
Average Pool	-	2	30 × 40 × 32

Table 2. Structure of temporal Stream

Layer	Kernel Size	Stride	Output size
Conv1 (ReLU)	9 × 9(64)	1	120 × 160 × 64
Max Pool	-	2	60 × 80 × 64
Conv2 (ReLU)	5 × 5(64)	1	60 × 80 × 64
Conv3 (ReLU)	3 × 3(64)	1	60 × 80 × 64
Conv4 (ReLU)	3 × 3(32)	1	60 × 80 × 32
Average Pool	-	2	30 × 40 × 32

2.3. Transfer learning

Many machine learning techniques, especially Deep Learning, work well only under the assumption that the collected training data are sufficient to optimize the large amount of network parameters (e.g. weights). However, it is generally difficult in manufacturing settings to acquire sufficient data that contain information on defects, due to the fact that defect-involved operation, once detected, will be terminated to avoid damage to the machines and products.

Transfer learning refers to the technique that is capable of transferring the learned knowledge from a source domain to a related target domain [20]. In the context of DL, it saves the need and effort to collect large amount of training data in the target domain and build a new model from scratch [21].

In this research, the transfer of feature extraction capability of the CNN is explored. It has long been understood that the working mechanism of CNN is to first extract low-level image features (such as edge and curve) at early convolutional layers and then assemble these features into high-level patterns in fully-connected layers for classification. This implies that the early layers in a CNN has a more generic feature extraction capability that can potentially be generalized across different domains. In the developed transfer learning framework, the weights of the early layers in the pretrained CNN are frozen and transferred (i.e. 2 convolutional layers and 2 pooling layers in spatial stream. 4 convolutional layers and 2 pooling layers in temporal stream). To realize human action recognition in the target domain (i.e. manufacturing assembly), the fused features through the transferred layers are fed into fully-connected layers and classification layer. The weights in these two layers are fine-tuned using the data from the target domain for domain adaptation. Given that none of the weights in convolutional layers is further adjusted, the total number of weights in the two-stream CNN to be optimized using the target domain data is considerably reduced and therefore, the limitation in training data quantity is alleviated.

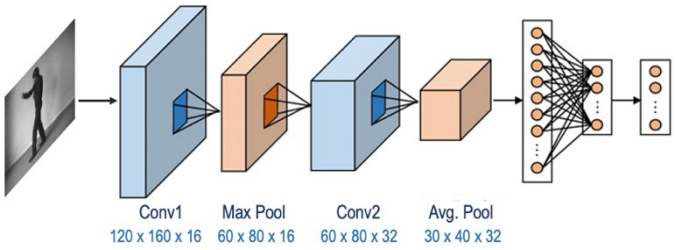


Fig. 5. Spatial stream

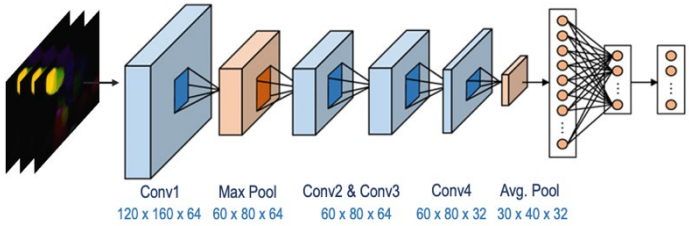


Fig. 6. Temporal stream

2.4. Feature visualization

Features extracted from CNN is presented in high dimensional space and is difficult to visualize. To facilitate the evaluation of the performance of the develop method (e.g. in terms of separability among features corresponding to different human actions), t-SNE, a method to visualize data in a high-dimensional space, is investigated [22].

t-SNE is an improved version of Stochastic Neighbor Embedding (SNE). The basic idea of SNE is to represent the similarities between data points x_j and x_i from high-dimensional space with the conditional probabilities $p_{i|j}$ as:

$$p_{j|i} = \frac{\exp\left(\frac{-\|x_i - x_j\|^2}{2\sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(\frac{-\|x_i - x_k\|^2}{2\sigma_i^2}\right)} \tag{8}$$

where σ_i is the variance of the Gaussian distribution centered on x_i. The conditional probabilities of low-dimensional counterparts y_j and y_i can be expressed in a similar manner:

$$q_{j|i} = \frac{\exp(-\|y_i - x_j\|^2)}{\sum_{k \neq i} \exp(-\|x_i - x_k\|^2)} \tag{9}$$

Intuitively, if $p_{i|j} = q_{i|j}$, the low-dimensional counterparts y_j and y_i perfectly represent the data points x_j and x_i in high-dimensional space. However, this cannot always be achieved. Therefore, to find the best visualization of x_j and x_i in low-dimensional space, the difference between $p_{i|j}$ and $q_{i|j}$ is minimized. The cost function representing the difference between $p_{i|j}$ and $q_{i|j}$ can be expressed as the summation of Kullback-Leibler divergence over all data points:

$$C = \sum_i KL(P_i \parallel Q_i) = \sum_i \sum_j p_{j|i} log \frac{p_{j|i}}{q_{j|i}} \tag{10}$$

t-SNE utilizes student t-distribution instead of Gaussian distribution when computing the conditional probabilities. Compared with SNE, execusion of t-SNE is simpler. Also, t-SNE generally produces better visualizations by reducing the tendency to cluster points together in the center of the space [22].

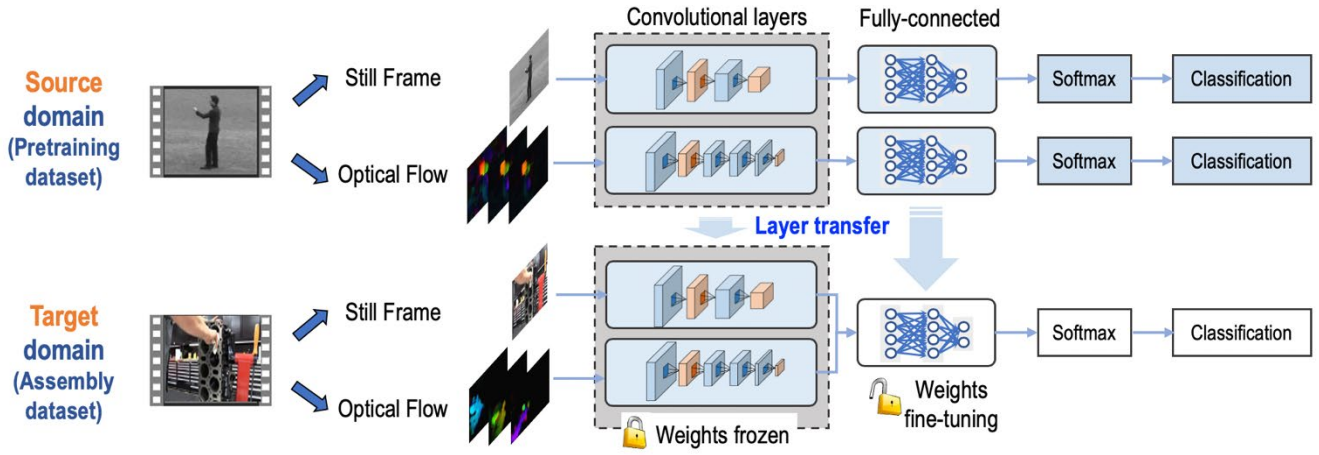


Fig. 7. Flowchart of transfer learning

3. Experimental Evaluation and Discussion

This section presents the experimental evaluation of the developed method and the result discussion.

3.1. Experimental Setup

The developed transferable two-stream CNN model is comprised of three parts: spatial stream, temporal stream, and classifier. The spatial stream and temporal stream, which consist of convolutional and pooling layers, work as feature extractors. The feature extracted by both streams will be fused before fed into the classifier, which consists of fully connected layers, and a softmax layer for classification.

Figure. 7 shows the flow chart of transfer learning. First, still frame and optical flow images are extracted from open source human action videos to build the pretraining dataset (source domain). Next, they are utilized to pretrain the spatial and temporal stream of CNN. Then, the convolutional and pooling layers in the pretrained model are transferred to capture the features from assembly dataset (target domain). Finally, the weights in fully-connected layers are fine-tuned to fuse the extracted features for action recognition in target domain.

3.1.1. Dataset

The pretraining dataset consists of images of still frames and optical flow frames chosen from two large scale open source human action datasets, KTH human action dataset and UCF101 human action dataset [23, 24]. Five human actions from KTH dataset: boxing, clapping, waving, jogging and walking and five human actions from UCF101 dataset: makeup, archery, cutting, fencing, and swimming, are chosen for pretraining and are shown in Fig. 8 and 9, respectively. In total, 11,450 still frames and optical flow frames from ten different categories are extracted from related videos.

The assembly dataset is collected from an engine block assembly video on YouTube [25]. As shown in Fig. 10, the video consists of seven assembly actions: cleaning, hammering, installing, marking, polishing, smearing and screwing. In total, 3,948 still frames and optical flow frames of the seven categories are extracted from the assembly video.

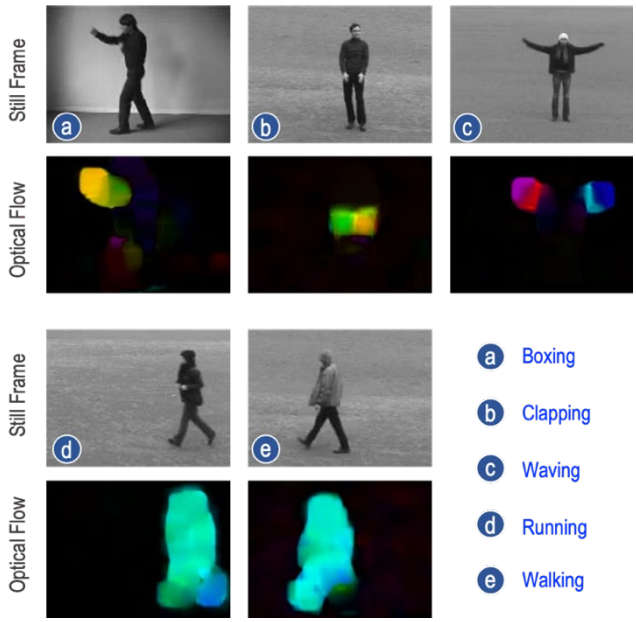


Fig. 8. Pretraining data from KTH [23]

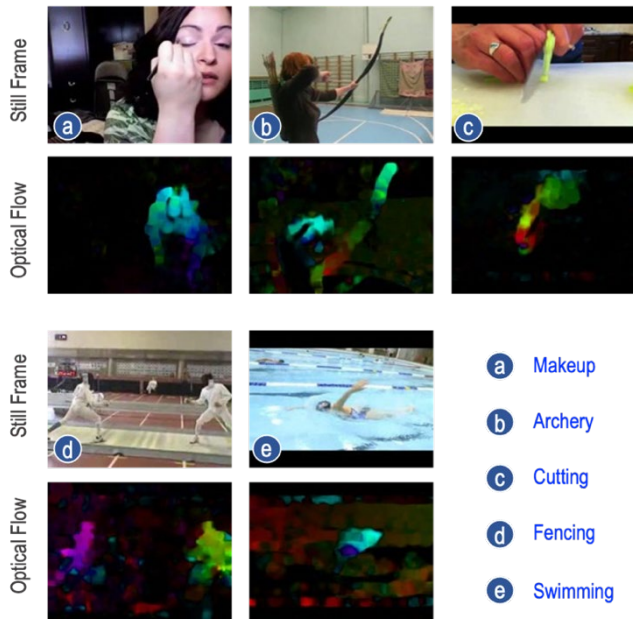


Fig. 9. Pretraining data from UCF101 [24]

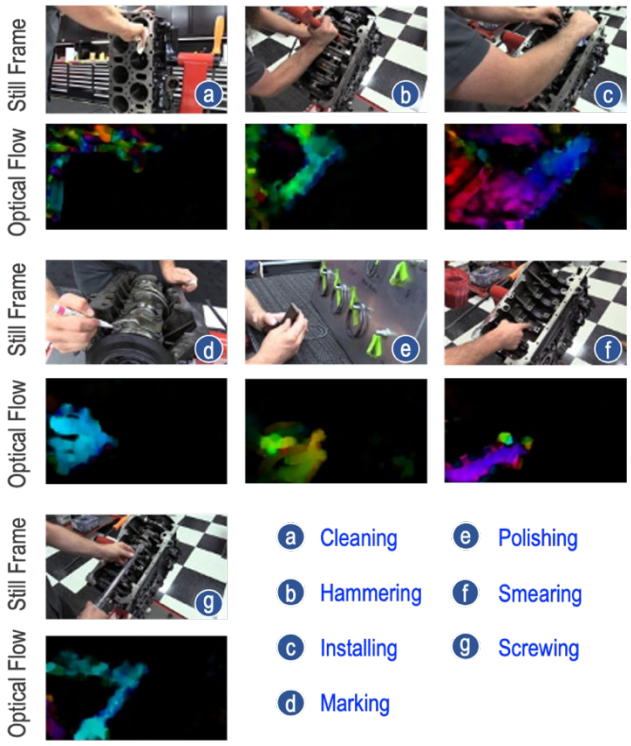


Fig. 10. Assembly data of engine block assembly [25]

3.1.2. Pretraining

The objective of pretraining is to develop a model that can recognize human actions by classifying the related images. Therefore, two CNNs, namely spatial stream and temporal stream, are constructed to classify the images into ten different categories. The spatial stream is pretrained using individual still frames while the temporal stream is pretrained using stacks of optical flow images. Detailed CNN structures of the two streams are illustrated in Fig. 5 and Fig. 6 respectively. The pretraining dataset is randomly split into two for the purpose of training and testing, respectively. 85% of the samples in the pretraining dataset are used as training set, and the remaining are used as testing set.

3.1.3. Evaluation in Target Domain

After pretraining, the convolutional and pooling layers in spatial and temporal streams are frozen and transferred to recognize human actions in the assembly videos. Similarly, the samples in assembly dataset are randomly split for training (fine-tuning) and testing. 70% of data are used for fine-tuning the new classifier such that it is able to identify seven different human actions in assembly dataset. 30% of data are used to test the performance of the adapted classifier.

3.2. Results

Table 3 and Table 4 show the action classification accuracy of pretraining dataset and assembly dataset, respectively. Specifically, each network structure is tested five times with different random split of training and testing sets. The resulting mean classification accuracy and the corresponding standard deviation are shown.

In Table 3, the mean classification accuracy of the two-stream model is 88.31%, which is 5% higher than the mean classification accuracy when using the spatial stream alone (83.06%). It is also considerably higher than the accuracy using only the temporal stream (88.31% vs. 66.37%). This confirms the importance of using both spatial and temporal information for improved action recognition performance. In addition, the standard deviation of two-stream model results (0.0327) is lower than those from the two single-stream models (0.0419 and 0.0448), suggesting that the two-stream model is more robust to data variation.

Table 3. Accuracy of pretraining dataset

	Mean	Std. Dev.
Spatial Stream	83.06%	0.0419
Temporal Stream	66.37%	0.0448
Two Stream	88.31%	0.0327

Table 4. Accuracy of assembly dataset

	Mean	Std. Dev.
Spatial Stream	99.95%	0.0002
Temporal Stream	72.88%	0.0448
Two Stream	100.00%	0.0000

In Table 4, it is seen that the mean classification accuracy of the two-stream model has reached 100%, indicating that the transferred model has effectively captured the action-related image patterns from the assembly dataset, even though the feature extraction capability is obtained from the pretraining dataset. This suggests that the low-level feature extraction mechanism in CNN is indeed generic and can be generalized among different action recognition tasks. It is also seen that the two-stream model has the best performance as compared to two single-stream models after transfer, although the spatial stream also achieved near perfect recognition accuracy (99.95%).

To evaluate the performance of the models beyond classification accuracy, t-SNE is deployed to map the extracted high-dimensional features into a two-dimensional space and visualize the feature separability. The larger the separation, the better the effectiveness of the model in distinguishing different human actions. Specifically, features extracted from the fully-connected layer in each model are chosen to be visualized.

The features from the pretraining dataset extracted by the spatial stream, temporal stream and the two-stream models are visualized in Fig. 11. By comparing the visualization of two-stream model with that of single-stream model, it is seen that the clusters of data points from two-stream model are more separated from each other. Furthermore, the spatial and temporal information do seem to complement each other. For instance, the spatial stream model cannot distinguish hand waving from hand clapping. However, the difference is much clearer in the temporal stream, as shown by the two clusters with a more obvious border. This separation is the most obvious in the two-stream model, as the two clusters are completely separated. As another example, the temporal stream model failed to distinguish boxing from archery as both actions are finished with the movement of arm. However, by considering the additional spatial information, boxing and archery are successfully separated in both the spatial stream and the two-stream models.

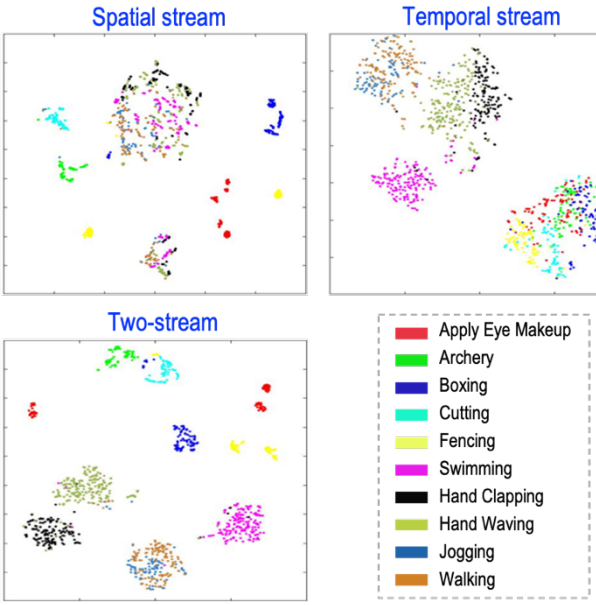


Fig. 11. t-SNE results of pretraining dataset among spatial stream, temporal stream and two-stream network models

Fig. 12 shows the visualization of features of the assembly dataset. Similarly, the performance of two-stream model is better than both the spatial stream and temporal stream models as the clusters of different human assembly actions are clearly separated from each other in two-stream model, while the separation in the spatial stream model is less obvious and poor separation is observed in the temporal stream model, which is consistent with the low classification accuracy achieved.

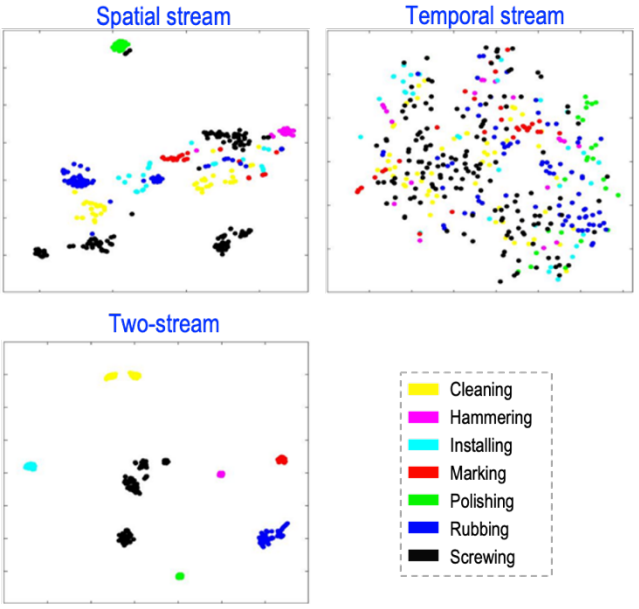
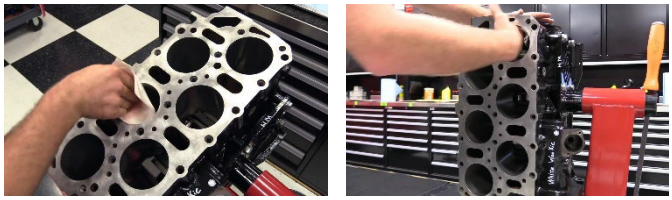


Fig. 12. t-SNE results of assembly dataset among spatial stream, temporal stream and two-stream network models

It is further noted in Fig. 12 that the cleaning and screwing actions exhibit clustering behavior within themselves. The reason is that, for the cleaning action, the engine block is presented with two configurations (i.e. standing and lying as shown in Fig. 13), and for the screwing action, the angle and distance of the camera from the engine block are both varying. The clustering behavior within the same activity shown in Fig. 12 is the direct reflection of these variations. The capture of the in-class clustering and the accurate classification of the actions from different classes confirm the robustness of proposed method to the variations in the position and location of the camera and engine block.



(a) (b)

Fig. 13. Cleaning with (a) lying engine block; (b) standing engine block

To gain insight into the robustness of the method to the variation in room light conditions, noise with different densities is progressively added into the raw images of all seven assembly actions. Noise density refers to the ratio of the number of the affected image pixels to the total number of pixels. Digital camera commonly adapts to the dimming light condition by increasing the “brightness level”, which generates image noise as trade-off. Therefore, progressive addition of noise serves as simulation of the varying light conditions. The peak signal-to-noise ratio (PSNR) values of the images with different noise densities are also calculated. PSNR is expressed as the ratio of the maximum possible value of a signal to the power of distorting noise that affects the quality of its representation [26] and is computed as:

$$PSNR = 10 \times \log_{10}\left(\frac{\left(2^{n}-1\right)^{2}}{MSE}\right) \tag{11}$$

where MSE is the deviation of the noise image from the raw image computed as mean square error, n is determined by the image datatype (e.g. for uint8, n is 8).

Table 5. Accuracy of assembly dataset under different noise densities

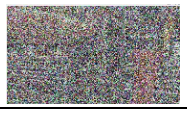
Noise Density	Sample Images (Cleaning)	PSNR (dB)	Mean	Std. Dev
0.0		-	100.00%	0.0000
0.2		15.45	100.00%	0.0000
0.4		12.41	100.00%	0.0000
0.6		10.55	99.66%	0.0046
0.8		9.20	97.12%	0.0142

Table 5 shows the sample images (cleaning operation) with different noise densities and the classification results under the corresponding noise contamination. It is seen that, the developed method has been able to correctly identify all seven

human assembly actions until the noise density is raised to 0.6 when the mean classification accuracy drops to 99.66%. With a noise density of 0.8, the two-stream CNN still achieved a classification rate of 97.12%. These observations indicate that the developed method is robust to the image noise and consequently, the varying light conditions.

4. Conclusion

This paper presented an integrated method to advance human action recognition in HRC. First, optical flow images have been investigated to add temporal information to complement the spatial information from static images for CNN-based action recognition, for which a two-stream CNN structure has been designed. Then, to improve the generalization ability of the model and alleviate the limitation of training data quantity in manufacturing, transfer learning, an emerging learning paradigm that allows the models trained over a large-scale dataset to be effectively adapted to new domains, has been investigated. Experimental study has shown that the developed method based on optical flow and transfer learning is capable of achieving high human action recognition accuracy in a realistic assembly scenario. In addition, the developed method has shown to be robust under modest variation of assembly configuration and noisy video footage. Future work includes the theoretical analysis of data transferability, which would advance the developed method towards the broad acceptance as a trustworthy technique in HRC.

Acknowledgment

This work is partially supported by the National Science Foundation under award CMMI-1830295.

References

[1] Bauer, A., Wollherr, D. and Buss, M., 2008. Human–robot collaboration: a survey. International Journal of Humanoid Robotics, 5(1), pp.47-66..

[2] Lowe, D.G., 1999, September. Object recognition from local scale-invariant features. International Conference on Computer Vision, 99(2), pp. 1150-1157.

[3] Bay, H., Ess, A., Tuytelaars, T. and Van Gool, L., 2008. Speeded-up robust features (SURF). Computer Vision and Image Understanding, 110(3), pp.346-359.

[4] Rublee, E., Rabaud, V., Konolige, K. and Bradski, G.R., 2011, November. ORB: An efficient alternative to SIFT or SURF. International Conference on Computer Vision, 11(1), p. 2.

[5] Belongie, S., Malik, J. and Puzicha, J., 2002. Shape matching and object recognition using shape contexts. IEEE Transactions on Pattern Analysis and Machine Intelligence, 24(4), pp.509-522.

[6] Han, J., Shao, L., Xu, D. and Shotton, J., 2013. Enhanced computer vision with microsoft kinect sensor: A review. IEEE Transactions on Cybernetics, 43(5), pp.1318-1334.

[7] Wilson, A.D. and Bobick, A.F., 1999. Parametric hidden markov models for gesture recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 21(9), pp.884-900.

[8] Feng, K.P. and Yuan, F., 2013, December. Static hand gesture recognition based on HOG characters and support vector machines. 2nd International Symposium on Instrumentation and Measurement, Sensor Network and Automation (IMSNA). pp. 936-938.

[9] Bobick, A.F. and Davis, J.W., 2001. The recognition of human movement using temporal templates. IEEE Transactions on Pattern Analysis and Machine Intelligence, 23(3), pp.257-267.

[10] LeCun, Y., Bengio, Y. and Hinton, G., 2015. Deep learning. Nature, 521(7553), pp.436-444.

[11] Wang, P., Liu, H., Wang, L. and Gao, R.X., 2018. Deep learning-based human motion recognition for predictive context-aware human-robot collaboration. CIRP Annals, 67(1), pp.17-20.

[12] Chaudhary, S. and Murala, S., 2019. Deep network for human action recognition using Weber motion. Neurocomputing, 367, pp.207-216.

[13] Ijjina, E.P. and Chalavadi, K.M., 2017. Human action recognition in RGB-D videos using motion sequence information and deep learning. Pattern Recognition, 72, pp.504-516.

[14] Ullah, A., Muhammad, K., Haq, I.U. and Baik, S.W., 2019. Action recognition using optimized deep autoencoder and CNN for surveillance data streams of non-stationary environments. Future Generation Computer Systems, 96, pp.386-397.

[15] Turaga, P., Chellappa, R. and Veeraraghavan, A., 2010. Advances in video-based human activity analysis: challenges and approaches. Advances in Computers, 80, pp. 237-290.

[16] Optical flow, docs.opencv.org/3.4/d4/dee/tutorial_optical_flow.html.

[17] Sánchez, J., Salgado, A. and Monzón, N., 2015. Computing inverse optical flow. Pattern Recognition Letters, 52, pp.32-39.

[18] Simonyan, K. and Zisserman, A., 2014. Two-stream convolutional networks for action recognition in videos. Advances in neural information processing systems, pp. 568-576.

[19] Wu, H. and Zhao, J., 2018. Deep convolutional neural network model based chemical process fault diagnosis, Computers and Chemical Engineering, 115, pp. 185–197.

[20] Torrey, L. and Shavlik, J., 2010. Transfer learning. Handbook of research on machine learning applications and trends: algorithms, methods, and techniques, pp. 242-264.

[21] Pan, S.J. and Yang, Q., 2009. A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering, 22(10), pp.1345-1359.

[22] Maaten, L.V.D. and Hinton, G., 2008. Visualizing data using t-SNE. Journal of Machine Learning Research, 9, pp.2579-2605.

[23] Recognition of human actions, www.nada.kth.se/cvap/actions/

[24] Soomro, K., Zamir, A.R. and Shah, M., 2012. UCF101: A dataset of 101 human actions classes from videos in the wild. arXiv:1212.0402.

[25] How to assemble an engine block,Youtube, www.youtube.com/watch?v=zPAElcQH0YY.

[26] Peak Signal-to-Noise Ratio as an image quality metric, National Instruments, www.ni.com/en-us/innovations/white-papers/11/peak-signal-to-noise-ratio-as-an-image-quality-metric.html.