Signal, Image and Video Processing
https://doi.org/10.1007/s11760-020-01800-6

ORIGINAL PAPER

Human activity recognition-based path planning for autonomous vehicles

Martin Tammvee[1] · Gholamreza Anbarjafari[2,3,4]

Received: 15 August 2020 / Revised: 15 August 2020 / Accepted: 5 October 2020

Abstract
Human activity recognition (HAR) is a wide research topic in a field of computer science. Improving HAR can lead to massive breakthrough in humanoid robotics, robots used in medicine and in the field of autonomous vehicles. The system that is able to recognise human and its activity without any errors and anomalies would lead to safer and more empathetic autonomous systems. During this research work, multiple neural networks models, with different complexity, are being investigated. Each model is re-trained on the proposed unique data set, gathered on automated guided vehicle (AGV) with the latest and the modest sensors used commonly on autonomous vehicles. The best model is picked out based on the final accuracy for action recognition. Best models pipeline is fused with YOLOv3, to enhance the human detection. In addition to pipeline improvement, multiple action direction estimation methods are proposed.

Keywords Neural networks · Self-driving car · Object detection · Human detection · Human action detection · Path planning

1 Introduction

Human activity recognition (HAR) is a wide field of study dedicated on identifying the specific movement or action of a person based on acquired data. Data can be gathered by multiple different sensors, depending on field of usage for HAR [1]. Most common activities that are tracked are walking, standing and sitting. Actions can be more specific if model needs to be used in more narrow field, for example, medicine.

This research work contributes on finding the best model of HAR for self-driving cars and improving them with state-of-an-art techniques. Second part of the research work is validating a new data set gathered by the most modern sensors in the field of self-driving car. Third part of the research work is adding new features to the researched models as well as proposing various methods estimating humans movement direction in videos.

Autonomous cars can be allowed into public areas only when they are completely safe to humans. As the resources on the self-driving cars are limited, the procedure cannot be computationally expensive, while at the same time it has to run fast and maintain the high accuracy. HAR has a huge impact on the safety of autonomous vehicles [2,3].

Main objective of this research work is to find the fastest and most accurate open source HAR model for self-driving cars. Each model under view is re-trained on the proposed data set and hyper parameters fine-tuned accordingly to achieve the best performance. For better accuracy of human prediction, the best models pipeline is fused with YOLOv3 network. The rest of this paper is organised as follows: Sect. 2 describes the related works. Section 3 presents the proposed method. Section 4 shows the obtained experimental results, and Sect. 5 concludes the paper.

This work has been partially supported by the Estonian Centre of Excellence in IT (EXCITE) funded by the European Regional Development Fund. The authors also gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X Pascal GPU.

✉ Gholamreza Anbarjafari
shb@ut.ee

Martin Tammvee
martin.tammvee@cleveron.com

1 Cleveron Ltd., Reinu tee 48, 71020 Viljandi, Viljandi County, Estonia

2 iCV Research Lab, Institute of Technology, University of Tartu, 50411 Tartu, Estonia

3 PwC Advisory, Helsinki, Finland

4 Department of Electrical and Electronic Engineering, Hasan Kalyoncu University, Gaziantep, Turkey

2 Related works

HAR has a huge role in many emerging applications [4] such as autonomous vehicles, business analytics and personal assistant robots [5–7].

2.1 Human activity recognition in human–robot interaction

Robots are becoming more common in human prosaic life as, for example, domestic robots are starting to get irreplaceable in our rushy lifestyle. To make robots compatible with new situations without causing any damage to environment, human activity recognition is inevitable. Coming to self-driving car, HAR is unavoidable. Only when the car understands where are the humans and what is their activity, it is able to safely navigate through populated areas without having any external guidance. Researcher in [8] introduced convolutional neural network to classify 3D human activities for mobile robots. Developed model was trained on Vicon Physical Action data set [9]. The modal was tested on new data to validate its performance in new circumstances and environment.

In hazardous environments or tasks where high precision is required, robots outperform humans. HAR is used to make the robot move based on the human action or for mimicking humans. Research in Galatasaray University [10] proposed a method how to control a robot based on a HAR. During the experiment the human action was tracked by wearable sensors. Special neural network was conducted to classify the action from the sensor data. Based on the network output and task-based function, robot movement was performed.

2.2 Classification of human activity recognition approaches

Human activity recognition can be classified into multiple research branches. Most popular branches are sensor and vision based. HAR based on sensor data can be separated into three sub-branches regarding sensor's deployment: based on wearable sensors, object tagged and dense sensing [11].

2.2.1 Sensor-based HAR

HAR based on wearable sensors is very attractive research topic mainly because of its application areas. Named method is widely tested in healthcare and smart environments. Main sensors that are being used to gather data are accelerometer, gyroscope and magnetometer. Sensors are being attached on the test object or person to log the data, while certain activities are being done. Further on, different neural network approaches are used to classify the action [12].

2.2.2 Vision-based HAR

The most popular HAR method is based on the camera footage. The reason for that lies behind the amount of data available and small cost of the sensors needed. There are a lot of open-source data that can be used for human action classification [13,14]. However, those named data sets have to be pre-processed in order to label the action for example.

Vision-based HAR can be applied using different methods on different input data. Research has been done [15] on applying HAR with uni-modal and multi-modal methods. Uni-modal methods use data from single modality, where human activities are represented as a set of visual features extracted from a video. Multi-modal approaches use input from different sources. Event of an action can be described by different types of features or even fusion of multiple features.

3 Proposed method

This section will validate different algorithms and approaches for HAR implementation on proposed data set. For that three different models, with different computational complexities, are being viewed and tested. Models were retrained on proposed data set. Data were divided into three sets: training (60%), validation (30%), and test (10%) set. Goal is to test different methods and approaches to find the model with the best performance for HAR in autonomous vehicles.

3.1 Uni-modal approach

First model under consideration would be rather simple neural network [16]. This model was tested because the author reported that he got the prediction accuracy of 91.27% on the UFC-101 data set [17]. In order to evaluate HAR, video frames are feed into LSTM model as an input. Network input image size needed to be resized to 512 pixels. LSTM model had one hidden layer of size 1024 followed by batch normalisation with appropriate linear transformations. For activation ReLU function followed by Softmax activation function was used. The output of the model was probability of each action class.

3.2 Uni-modal approach with skeleton detection

For the second model [18,19], more complicated pipeline is used. First part of the model is human pose estimation. The human pose is based on rather simple pose estimation algorithm [19] . The model is ideal for application where low latency is required, especially on self-driving cars where every millisecond of latency can be fatal for a living being. Model extracts eighteen features from human pose : left eye, nose, right eye, left ear, right ear, left shoulder, right shoulder, left elbow, right elbow, left wrist, right wrist, left hip, right hip, left knee, right knee, left ankle, right ankle and neck. Network extracting the features is based on ResNet [20], which is one of the most common backbone residual neural networks for image feature extraction. Transfer learning techniques are used for deconvolutional layers and are added to the last convolutional layers in the ResNet. All in all three deconvolutional layers with batch normalisation and ReLU activation are used. Each layer has 256 filters with 4×4 kernel with stride two. For loss, mean squared error (MSE) is used and for optimisation Adam function with learning rate of 10^{-6}.

Next part of the pipeline uses the output of the previous step as an input to a classifier, where the action is estimated. The author used [18] used logistic regression classifier. After retraining the model with proposed data set, the initial classifier performed poorly. Two other classifiers were tested: multi-layer perceptron classifier and stochastic gradient descent (SGD) classifier, where SGD outcome proved to suit the best for proposed data set.

3.3 Multi-modal approach

Third model that was researched [21] was trained on videos taken inside a room with a camera. The recorded material was with resolution of 640×480 pixels and with a frame rate of 10 (fps). Action classification was made between 9 different actions: waving, standing, punching, kicking, squatting, sitting, walking, running, jumping. The videos were from 0.8 s to 2 min long.

3.3.1 Getting the pose of the human

Algorithm to classify human action consists of multiply sub-algorithms. For the first step, algorithm detects human skeleton with OpenPose algorithm. Skeleton of the person is visualised by coordinates, where a right order of coordinates forms a specific joint. Thus, not all combinations of coordinates form a joint. The combination, that will form a specific joint, is defined by the user. Bottom-up approach detects first all human parts on the image and then groups joints belonging to individual person and estimates the pose. Top-down approach detects first all the humans on the image, followed by finding joints on each separated human and then estimating the pose for each skeleton. Top-down approach is normally easier to use, because adding person detection takes less effort than adding grouping algorithm. Performance-wise these two methods are equal [22].

As this work is focusing mostly on self-driving cars, multi-person algorithms are taken under consideration. OpenPose is one of the most popular bottom-up methods. Algorithm first detects the joints associated to all persons on the image,

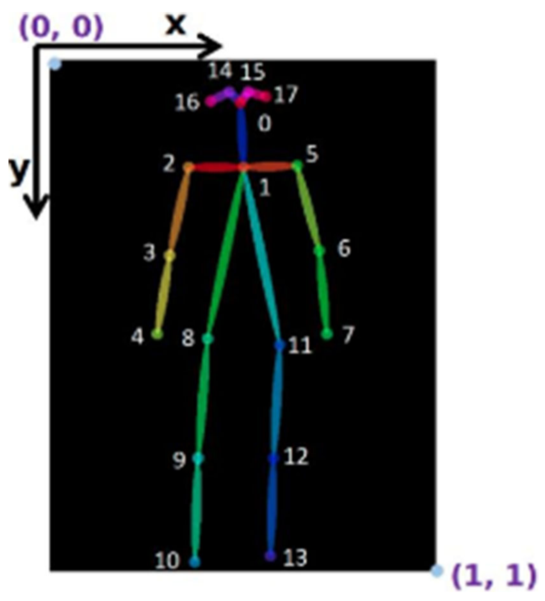


Fig. 1 OpenPose predicted skeleton parts [23]

subsequently joining joints to unique person. The video file or camera feed is given as an input into the system. Camera feed means that this algorithm can be used in real time, for example, on a self-driving car. OpenPose first extracts features from an image using the first layers (VGG-19). Repeating the procedure will lead to more coordinate points and will improve the accuracy of the predictions made by each branch. Using the part confidence maps, bipartite graphs are formed among pairs of parts. Using the part affinity fields (PAFs) values, weaker links in the bipartite graphs are neglected. From all the above steps, human pose skeletons can be estimated and assigned to every person in the image. In OpenPose algorithm, each human skeleton has 18 joints, shown in Fig. 1.

3.3.2 Feature verification

OpenPose algorithm detects in total 18 features (shown in Fig. 1), where five of the features represent the human head area. In total, 13 features are left: neck, left shoulder, left elbow, left hand palm (last three form a left hand), right shoulder, right elbow, right hand palm (last three form a right hand), left knee, left ankle (last two form a left leg), right knee, right ankle (last two form a right leg), left thigh and a right thigh.

From the human skeleton, body velocity, joint velocity and normalised joint position are extracted. Every point has x and y value, but OpenPose outputs them with a different unit. In order to work with them, coordinates are scaled to be the same unit. After the previous steps have been completed, the algorithm now verifies whether the detected skeleton has a neck and at least one thigh. If one of these components is missing, given frame becomes incompetent and no prediction is made on particular frame.

In addition for the action classification, other joints must be checked as well. If OpenPose could not predict all the joints for the current frame, missing joints have to be filled. This is necessary for the feature classification, where the input has to be fixed-size feature vector. To find the missing joints, the previous frame is taken under consideration. Algorithm compares the skeleton on the current frame to a skeleton on a previous frame. The comparison is made based on the coordinates of the neck. The missing joints can be carried forward from the previous frame, if the position difference of the neck in the consecutive frames is less than a set threshold.

3.3.3 Tracking each person

With self-driving cars, we are interested in videos or consecutive images, where the human pose must be detected and the action tracked. In order to track a human pose in consecutive frames, Euclidean distance is used. The distance is calculated between the coordinates of two skeletons from a previous and current frame. If the distance between two skeleton is lower than the threshold defined, human identifier from the previous frame is transferred to current frame. A new identifier is being set to a human skeleton when there is no match found between the skeleton coordinates in the two consecutive frames. This means that a new person has entered a frame. Numeration of the skeleton is initially given based on the human position on the image. The lowest number is acquired to the skeleton which is closest to the centre. The centre is defined by midpoint of the frame. This is import in order to be able to track the most dangerous situation for the car. The most dangerous situations are when human are just in front of the car.

3.3.4 Feature extraction for action classification

Previous body part positions are used in order to extract custom features that are used for action classification. The algorithm takes in consecutive five frames and concatenates them. This means for the first 4 frames of the video, no action is detected. Thus, if the video frame rate is ten frames per second, it takes around half a second before the action can be estimated.

Skeleton joint positions is the first feature for the action classification.

Very important feature for this model is human height. Human height is calculated from the skeleton neck position to skeleton thigh position. Height is used in order to normalise the extracted features.

The velocity of the body, as the next feature, is derived based on the skeleton neck. Skeleton neck velocity is normalised by diving the values by the height of the human. Third feature that is extracted is joint velocity. During testing, this feature is shown to have the biggest impact. Each point velocity is calculated by the normalised coordinate values. Velocity is derived between two consecutive frames. There were more custom feature extracted like joint angles and length of each body part. These featured did not affect the final precision of action classification, so they were left out. Moreover as the body velocity appeared as a very good feature for action classification, its weight was increased ten times. These three features are converted to one feature vector. After concatenation of all three features used, the feature vector has dimensions of 238. In order to reduce the feature vector size, principal component analysis (PCA) is used. After the feature reduction, the vector has dimension of 50.

3.3.5 Action classification

Last part of the model is estimating the action of the human. Reduced feature vector derived from three actions is fed into multi-layer perceptron (MLP) classifier. The classifier's parameters are updated iteratively at each time step based on the partial derivatives of the loss function with respect to the model parameters.

Used MLP classifier has three hidden layers. Layers size is, respectively, 20, 30 and 40. For activation function ReLU and for optimisation function, Adam is used. Learning rate during classification is constant at 0.001. The output of the classifier is a probability of an action. Special threshold is used as hyper parameter to define whether the action probability is suitable and will be estimated.

4 Results and discussion

4.1 LboroHAR data set

4.1.1 Data acquisition

During this work, an unique data set, LboroHAR [1], is used for training, validating and testing on different models of neural networks. The data were gathered in Loughborough University London. Fact that makes this particular data special is that it was recorded on the autonomous ground vehicle test bed (shown in Fig. 2) with three different sensors: RGB-D camera, LiDAR and 360-degree camera. The data represent the actual footage that today's automated guided vehicles (AVG) are capable of recording. Most of today's AGVs have either a LiDAR or RGB camera to detect humans. So this data set can give us opportunity for further research. In order to enhance proposed neural network models accuracy, different data types can be fused together, representing the same outer condition.

The proposed LboroHAR data set has sixteen different participants, doing nine different activities. All of the activi-

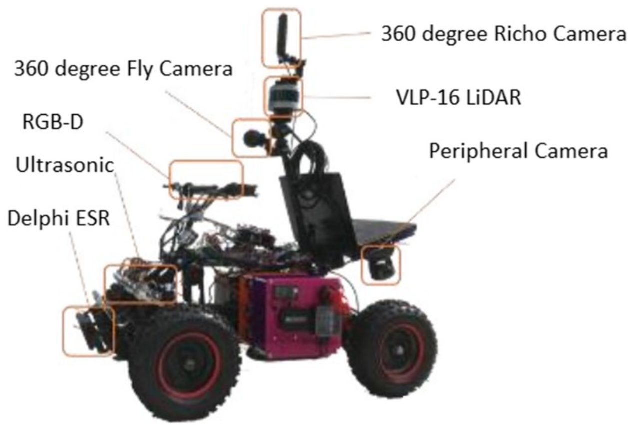


Fig. 2 Loughborough University London Autonomous driving sensor test bed

ties take place indoors. Each person carries out at least four different activities, with the maximum video length of 3 min. Each scene starts with a person coming to the centre of the frame and raising his or her hands. Then, the person who is being recorded performs one of the nine activities and finally every scene ends with a person coming back to the centre of the frame and raising his or her hands. The initial nine activities gathered are: sitting on the office chair, standing and texting on the mobile phone, sitting on the stool, lying on the couch, walking, walking and texting on the mobile phone, carrying different objects, pulling different objects, running.

4.1.2 Data pre-processing

This research is focused on the 360-degree camera footage. The raw data consisted of dual fisheye video files. In total, there are 133 videos recorded over the two days.

Main objective of the research is to detect person and classify their action; therefore, only the front view of the data is taken under consideration. In order to transform 360-degree dual fisheye camera video footage to 2D video, the rear view has to be cut out. For that FFmpeg [24] software is being used. Setback of doing the process was that the resolution dropped after the conversion.

For applying the conversion for all of the 133 initial videos, a special script was conducted that looped for a folder containing raw-videos and outputting to result folder. *Dfisheye* command was used to decompose the dual-fisheye frame with padding of 1%. To enhance the output frame quality, the chroma and luminance values were fused. To make the output more sharp and smooth, cubic interpolation was used.

In order to retrain machine learning models, the data needed to be labelled. The nine initial classes would be too narrow for action classification. Instead, new more general classes were created, which are walking, standing, sitting, running and lying.

All the data needed to be labelled manually. The labelling process for this research meant giving a label to each frame. In order to do that a special script was done that would separate all the frames from 133 raw 2D videos. In the end, total of 136,710 frames were gained. Output image size was 1080 $\times$ 1920 pixels.

From the last column of the table 3.1, it can be seen that the classes are not distributed equally. This fact can make a big impact on the accuracy of correct action classification. Neural network models can perform poorly on classes that are under represented because it will not have enough data to learn specific features of that particular class. For research purposes, all those classes are maintained to see the actual outcome on different machine learning models with given data set and classes.

Table 1 Presented models average accuracy

Model	Average (%) accuracy
Multi-modal approach	70.91
Uni-modal approach with skeleton detection	63.02
Uni-modal approach	2.4

4.2 Results

Each model was evaluated based on its accuracy of action prediction with the proposed data set. The accuracy was calculated over the test data set, which was 10% of all the provided data. Table 1 shows each models accuracy.

Frames where the skeleton was not predicted and the action was not estimated were counted as false positives. False positives were not taken into account while calculating the accuracy.

The most accurate class was standing, having average accuracy of 81.75% in the network where multiple neural networks were used. Class run had the lowest accuracy among five classes. The reason behind this is purely the data distribution, as the standing had the most samples among training and run had the least samples.

Figure 3 describes the training and validation loss of the first approach, uni-modal approach. From the graph, it can be seen that the validation loss is very high and does not follow the training loss, as it should be for the ideal case. After changing the following hyperparameters: hidden layer size, hidden layer dimension, batch and epoch size, the result did not change much.

Figure 4 describes the training score and validation score of the uni-modal approach with skeleton detection. Cross-validation is used with split parameter of 5. The result is

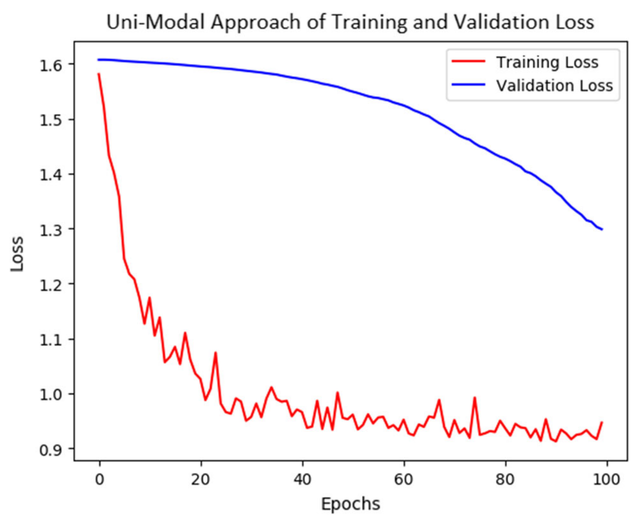


Fig. 3 Uni-modal approach plot of training and validation loss

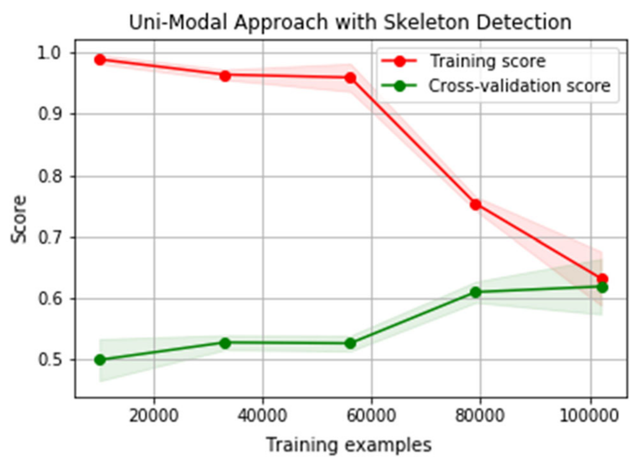


Fig. 4 Uni-modal approach with skeleton detection plot of training and validation loss

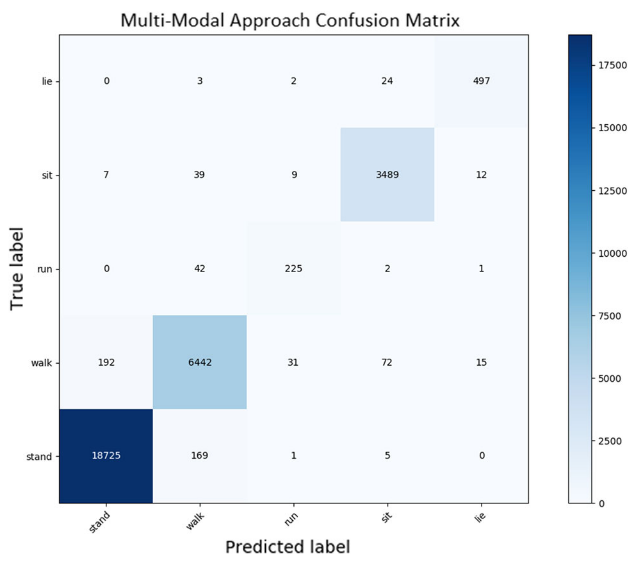


Fig. 5 Multi-modal approach confusion matrix

good as the training and validation scores are getting close to each other towards the final epochs. Overall accuracy could be improved with better quality of input data and more data samples. Having more data samples provides better data distribution, and the network would become more sensitive. After having better data distribution, network could require new parameter tuning, to improve the accuracy.

Third method, multi-modal approach, had similar learning curve as uni-modal approach with skeleton detection but with higher overall accuracy. The final accuracy could be improved by re-training the modal with larger data set and with equal data distribution. From Fig. 5, it can be seen that the classes that had the most samples (stand) had the best accuracy and the classes that had the least samples (run) had the worst accuracy. It can be observed from the image that the walking and standing classes had the most false-positives. Running was often miss-classified with walking. From a single image, it can be hard to classify whether the person is running or walking.

4.3 Further development

Model using multiple neural networks in its pipeline showed the best performance for this particular research and for self-driving car concept. The data were most similar to proposed data set, with similar data distribution.

4.3.1 Human action direction estimation

None of analysed models had any information about the movement direction of the human. For autonomous vehicles, this is a crucial aspect. Having the information about the humans action without a direction will not help to improve the autonomy. Humans movement prediction can be very useful factor while planning self-driving car path. In order to predict the humans movement direction, multiply methods were experimented.

First approach was implementing optical flow on the whole detected human skeleton from OpenPose algorithm. The centre point of the human was tracked with the optical flow. Centre of the human was calculated by minimum and maximum coordinates acquired from the skeleton joints.

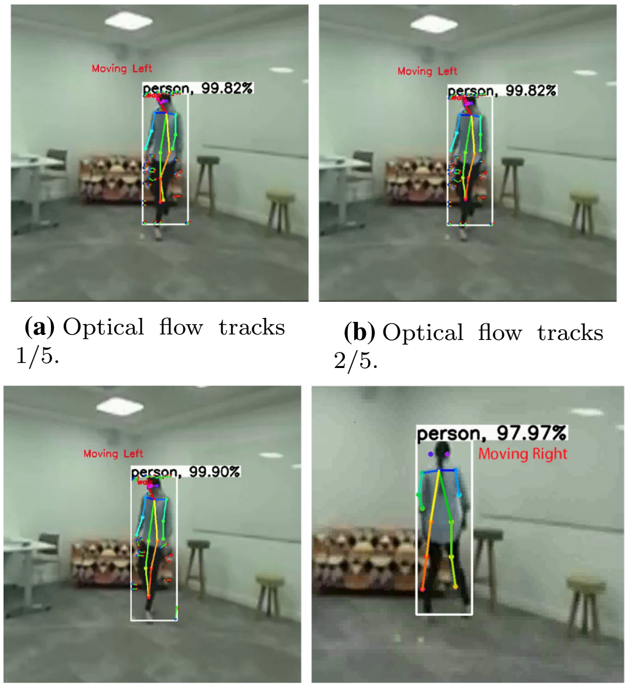
Drawback that occurred while validating given method was that human limbs make this approach unstable. While human walks or stands (waving her/his hands for example), the midpoint of the skeleton shifts heavily. Shift is tracked by the optical flow and will cause the system to predict that the person is moving.

To overcome this problem, instead of tracking the whole skeleton, only one certain point is tracked. The main joint of the skeleton for the action detection is neck, and if the OpenPose algorithm cannot detect the human neck, the whole skeleton will not be predicted on the given frame. That gives us one check for false negatives, as the movement would not be detected if the skeleton has not been detected. Considering that aspect for the second method, human neck is tracked with optical flow. The proposed approach worked perfectly on consecutive frames where the pose was estimated. The problems accrued when the pose was not being estimated due to OpenPose algorithm. Not having the skeleton, means that optical flow does not have the neck coordinates and direction estimation cannot be computed.

Next method introduces a new optical flow algorithm that follows YOLOv3 human prediction. A Shi-Tomasi corner detector and good features to track algorithm are used that detects strong corners from an image. Instead of following one point, the neck, now multiple points are being tracked. The points that are being determined are all in the YOLOv3 human region. Points detected go to optical flow function, using the Lucas–Kanade method. Function tracks where the points have shifted between consecutive frames. Points predicted by the optical flow are iteratively tracked, and the mean shift of all the points is computed. Then, the direction can be estimated when the mean shift goes above or below a predefined threshold.

To deal with optical flow detection noise, a back-tracking method is introduced, where the detected points are fed back into the optical flow function to find the original points. If the error between the two predictions is too high (usually due to a level of noise), those predicted points are ignored for the tracks.

Figure 6 shows the output after applying the new method of optical flow. On the first three images (Fig. 6a–c) the points and tracks are shown. Blue points correspond to the oldest tracks that are memorised, and the red points correspond to the newest shifted points. Tracks are drawn in green showing the path how the oldest and newest point has shifted. The movement direction is shown upon the human prediction bounding box. The last image (Fig. 6d) shows the output without the predicted point and tracks.



(a) Optical flow tracks 1/5.

(b) Optical flow tracks 2/5.

(c) Optical flow tracks 3/5.

(d) Frame with out the tracks.

Fig. 6 Result of mean optical flow

4.3.2 Anomalies

During the validation of the most successful model, some problems occurred. Namely OpenPose algorithm was detecting anomalies that false the whole direction estimation approach proposed in this work. Detected anomalies can be seen in Fig. 7. OpenPose detects human in the region of an image, where there is actually no human.

Anomalies keep appearing for different reasons. After proper investigation, two reasons can be brought out: the poor quality of training data and not enough data during training.

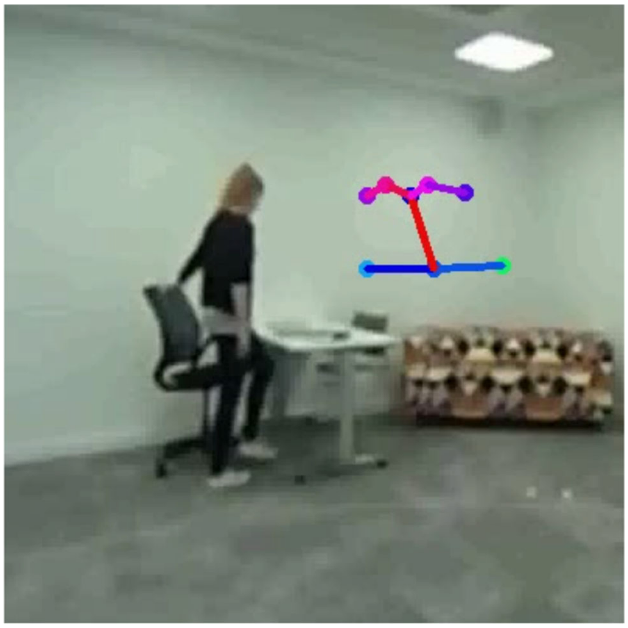


Fig. 7 Anomalies on video frames

The poor quality was caused after converting the raw 360-degree footage to 2D.

First method to overcome anomalies was to pre-process the input data going to the action estimation model. Initial video is passed through the YOLOv3 network, where human is detected. On each frame, the human is masked out. Mask is made 5 pixels wider and higher than the original YOLOv3 prediction. Enlarging the YOLOv3 mask will confirm that new mask covers the whole human on the frame. Mask is finally fed into action recognition network.

The outcome of the experiment was not successful. More anomalies were detected throughout the whole video. After analysing, it was found that the mask size of the human differs throughout the frames. It is caused by the human body movement. While the human is facing sideways, the mask is smaller and while she/he is facing the camera with hands stretched out wide, mask is bigger. Additionally, while the human walks or runs, her or his hands and legs move, which will affect the mask size. Neural network algorithm needs a fixed-size image as an input; thus, resizing is done.

5 Conclusion

This paper focused on applying the HAR on 360-degree camera footage. It compared multiply neural networks models. Each model was retrained with the LboroHAR data set, and respective hyperparameters were fine-tuned, to achieve a better performance. The best model, multi-modal approach, was enhanced with more accurate human detection by fusing YOLOV3 human prediction. As the path planning is very crucial aspect on self-driving car, the research work introduced methods to estimate the human movement direction on videos. The best method to track human direction was to implement a Shi-Tomasi corner detector and good features function. Detected points were tracked with optical flow.

References

1. Moencks, M., De Silva, V., Roche, J., Kondoz, A: Adaptive feature processing for robust human activity recognition on a novel multimodal dataset. ArXiv preprint arXiv:1901.02858 (2019)
2. Koopman, P., Wagner, M.: Challenges in autonomous vehicle testing and validation. SAE Int. J. Transp. Saf. **4**(1), 15–24 (2016)
3. Mordue, G., Yeung, A., Wu, F.: The looming challenges of regulating high level autonomous vehicles. Transp. Res. Part A Policy Pract. **132**, 174–187 (2020)
4. Jordao, A., Torres, L.A.B., Schwartz, W.R.: Novel approaches to human activity recognition based on accelerometer data. Signal Image Video Process. **12**(7), 1387–1394 (2018)
5. Sapiński, T., Kamińska, D., Pelikant, A., Anbarjafari, G.: Emotion recognition from skeletal movements. Entropy **21**(7), 646 (2019)
6. Nan, M., Ghiţă, A.S., Gavril, A.-F., Trascau, M., Sorici, A., Cramariuc, B., Florea, A.M.: Human action recognition for social robots. In: 2019 22nd International Conference on Control Systems and Computer Science (CSCS). IEEE, pp. 675–681 (2019)
7. Batziakoudi, K., Griva, A., Karagiannaki, A., Pramatari, K.: Human computer interaction in business analytics: the case of a retail analytics platform. In: ECIS (2020)
8. Olatunji, I.: Human activity recognition for mobile robot. J. Phys. Conf. Ser. **1069**, 01 (2018)
9. Dua, D., Graff, C.: UCI machine learning repository. http://archive.ics.uci.edu/ml. Accessed 05 April 2020
10. Uzunovic, T., Golubovic, E., Tucakovic, Z., Acikmese, Y., Sabanovic, A.: Task-based control and human activity recognition for human-robot collaboration. In: IECON 2018—44th Annual Conference of the IEEE Industrial Electronics Society. IEEE, pp. 5110–5115 (2018)
11. Shukri, S., Kamarudin, L.M., Rahiman, M.H.F.: Device-free localization for human activity monitoring. In: Intelligent Video Surveillance. IntechOpen (2018)
12. Jordao, A., Nazare Jr, A.C., Sena, J., Schwartz, W.R.: Human activity recognition based on wearable sensor data: a standardization of the state-of-the-art (2018)
13. Openimage dataset. https://storage.googleapis.com/openimages/web/index.html. Accessed 19 March 2020
14. Kaggle dataset on human. https://www.kaggle.com/dasmehdixtr/human-action-recognition-dataset. Accessed 19 March 2020
15. Vrigkas, M., Nikou, C., Kakadiaris, I.A.: A review of human activity recognition methods. Front. Robot. AI **2**, 28 (2015)
16. Lstm har. https://github.com/eriklindernoren/Action-Recognition. Accessed 22 Jan 2020
17. Ufc-101 dataset. https://www.crcv.ucf.edu/data/UCF101.php. Accessed 22 Jan 2020
18. Action recognition based on human skeleton. https://github.com/smellslikeml/ActionAI. Accessed 21 Jan 2020
19. Action recognition based on human skeleton. https://github.com/NVIDIA-AI-IOT/trt_pose. Accessed 22 Jan 2020
20. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition (2015)
21. Action recognition based on human skeleton. https://tinyurl.com/y4e74b9v. Accessed 04 Jan 2020
22. Cao, Z., Simon, T., Wei, S.-E., Sheikh, Y.: Realtime multi-person 2D pose estimation using part affinity fields. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7291–7299 (2017)
23. Qiao, S., Wang, Y., Li, J.: Real-time human gesture grading based on OpenPose. In: 2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), pp. 1–6 (2017)
24. FFmpeg. https://www.ffmpeg.org/. Accessed 12 March 2020