Increasing safety at smart elderly homes by Human fall detection from video using Transfer Learning approaches

Zahra Kharazian

Center for Applied Intelligent Systems Research, Halmstad University, Sweden. E-mail: zahkha20@student.hh.se

Mahmoud Rahat

Center for Applied Intelligent Systems Research, Halmstad University, Sweden. E-mail: mahmoud.rahat@hh.se

Emad Fatemizadeh

Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran. Email: fatemizadeh@sharif.edu

Ali Motie Nasrabadi

Department of Biomedical Engineering, Shahed University, Tehran, Iran. E-mail: nasrabadi@shahed.ac.ir

In this study, we investigate the problem of detecting humans fall from video images. Many of the existing methods try to solve the problem by manually defining a set of hand-crafted features for detecting fall, which is not only a suboptimal approach but also cumbersome. On the contrary, the proposed method puts the burden of feature extraction on a pre-trained deep neural network. In this way, we can extract a comprehensive set of conceptual features automatically and efficiently. An important challenge of employing deep neural networks is the need for a large collection of training data. While the available labeled data for human fall detection is very limited, we propose three approaches based on transfer learning, and we trained them on two standard RGB and depth datasets for fall detection. The pre-trained models explored in this study are VGG16, Inception V3, and ResNet50. Support vector machine and logistic regression are used to classify the extracted features from videos into two classes of fall and normal daily activities. The experimental results obtained from the proposed approach suggest that the transfer learning tactic is able to compensate for the low training data issue. It is also shown that the proposed approach can efficiently extract important features from the sequences of video and boost the accuracy of the system on the task of human fall detection.

Keywords: fall detection, machine vision, smart home, telemedicine, deep learning, transfer learning.

1. Introduction

With the recent improvements in health care, the life expectancy of humans and the average age of the society has increased in many countries. Although the phenomenon of aging is inevitable, it is possible to provide proper measures at smart homes to increase the safety of seniors. Late discovery of health-threatening emergencies happening at home is a source of potential risk for unaccompanied seniors. Human fall is one of the signals that could indicate the occurrence of such a situation. To this end, a fall detection system can be useful to improve the safety for elders.

Safety management is an essential issue in smart environments like an office, company, or household. Risks of human falls or an accident, in general, could be managed by monitoring the tools, machines, and activities of the workers or residents. (Teimourikia and Fugini 2017) have developed a dashboard named RAMIRES to control the risk in smart work environments. Also, (Fugini, Cirilli, and Locatelli 2015) introduced a risk detection project called Attiv@bili, which designed for interaction between the local health

care system and the frail person such as older people and people with disabilities.

In the context of human fall detection, researchers have used various types of sensors such as wearable (e.g., accelerometer and gyroscope), floor vibration, sound, 2D camera, and 3D depth sensors. (Igual, Medrano, and Plaza 2013) has reviewed the application of different sensors for fall detection. Most of the existing approaches focus on finding and extracting a limited number of hand-crafted features from the input signal. Subsequently, by the use of signal processing algorithms, researchers extract desired features from input data such as acceleration (Bourke et al. 2010, Kwolek and Kepski 2014), velocity (Lee and Mihailidis 2005), the distance of the body joints from the ground, the orientation of the axis of the body (Hazelhoff and Han 2008), acoustic signals (Li, Ho, and Popescu 2012), the amount of vibration on the ground, the width, height and depth of the 3D bounding box (Mastorakis and Makris 2014), and the ratio of individual's height and weight. Finally, the extracted features are fed to a classifier or

compared with some pre-defined thresholds to categorize into two classes of fall and Activities of Daily Living (ADL).

As mentioned earlier, one of the available approaches for fall detection is to use wearable devices. Typically, these devices contain several accelerometers that are attached to the human body (Bourke, O'Brien, and Lyons 2007, Cheng, Chen, and Shen 2012). These sensors measure the movement acceleration of the body parts. If the value read by the device exceeds some specific threshold, the motion is considered a fall. However, one obvious drawback of this approach is that they assume the person wears the sensors all the time. This approach not only is very inconvenient and preventive but also is not effective if the person forgets to wear the device, which is very probable for the seniors.

In vision-based fall detection approaches, the data is recorded by visual sensors such as 2D cameras (Mirmahboub et al. 2012) or 3D sensors like Microsoft Kinect (Mastorakis and Makris 2014). These sensors produce RGB, grayscale, and depth images. Vision-based methods usually produce superior results compared to wearable devices in fall detection. Also, these methods are user-friendly due to the passive operation of the sensors. However, methods that use 2D cameras suffer from some drawbacks such as violation of individual's privacy.

In recent studies, deep neural networks have been used to detect falls. For example, (Shojaei-Hashemi et al. 2018) use a long short-term memory network and (Feng et al. 2014) uses a Boltzmann machine along with a deep belief network. These networks are trained to extract various data features automatically. Therefore, they not only remove the need for manually defining the important features but also, they can include more abstract features in their hidden layers compared to the previous methods. However, one of the limitations of using deep neural networks is that they need to be fed by a huge number of training samples. On the other hand, the existing number of samples in publicly available fall detection databases is very limited. Moreover, collecting additional data is very laborious, costly, and time-consuming. To compensate for this challenge, we propose the transfer learning method on pre-trained deep neural networks. The idea behind transfer learning is re-using a pre-trained model to perform a different but related task.

2. Proposed method

In this study, we apply transfer learning on pre-trained deep neural networks to detect fall activity

using a video. The following three pre-trained deep neural networks are investigated:

- VGG16 (Simonyan and Zisserman 2014)
- InceptionV3 (Szegedy et al. 2016)
- ResNet50 (He et al. 2016)

These networks have been previously trained on ImageNet's (Deng et al. 2009) enormous database that contains millions of RGB images from various objects such as animals, flowers, vehicles, etc. We have investigated the application of transfer learning based on the following three scenarios:

1. Transfer learning for feature extraction
2. Transfer learning for feature fusion
3. Transfer learning with fine-tuning

2.1 Transfer learning for feature extraction

In the first approach, we employ the idea of transfer learning for feature extraction from video frames. According to the paradigm of transfer learning, in the proposed method, we use the pre-trained neural networks for feature extraction from video frames (see Fig. 1). The outputs of FC2, activation-93, and res5c-branch2b layers subsequently extracted from VGG16, InceptionV3, and ResNet50 networks are employed as feature vectors. The extracted sequence of feature vectors from consecutive frames of a video are joined together to form a matrix of features for each video. Finally, the feature matrix is fed to a classifier for labeling the entire video as ADL or Fall. The performance of two classifiers - Support Vector Machine (Cortes and Vapnik 1995), and Logistic Regression (Walker and Duncan 1967)) - are compared for this task.

Formally, one hidden layer of a neural network is a function $y: R^D \rightarrow R^L$, where D is the size of input vector x , and L is the size of the output vector $y(x)$, such that, in matrix notation:

$$y_{i,j}(x) = \sigma(Wy_{i,j-1}(x) + b) \quad (1)$$

Where i represents one of the pre-trained neural networks mentioned in section 2 i.e. $i \in \{VGG, InceptionV3, ResNet50\}$, and j represents different layers of that network. Accordingly, $y_{i,j}$ signifies the output of j^{th} layer of the network i , and σ represents the activation function of j^{th} layer of the network i applied on the input vector x to produce the output. As Eq. (1) indicates, the output of the layer $j - 1$ produces the input to layer j of the network.

The feature vector for this approach is produced according to Eq. (2). Note that the formula is presented only for the VGG16 network and the same formula is used for other networks as well.

$$f_{VGG}(X_k) = \begin{bmatrix} y_{VGG,FC2}(X_{k,1}) \\ y_{VGG,FC2}(X_{k,2}) \\ \vdots \\ y_{VGG,FC2}(X_{k,N}) \end{bmatrix} \quad (2)$$

Where f represents the feature extraction function, X_k is the k^{th} input sample video and $K \in \{1, \dots, M\}$ and M is the number of samples in the dataset. Moreover, $X_{k,l}$ represents the l^{th} frame of the video X_k , and $l \in \{1, \dots, N\}$, and N is the number of frames in the video (in our experiments $N = 35$).

Fig. 1. Illustration of extracting features from video using the first approach.

2.2 Transfer learning for feature fusion

The pre-trained networks used in this study have their own specific structure. Therefore, each of them is specialized to extract a different set of features. The first approach considered features extracted using each network individually. The intuition of the second approach is to fuse features extracted from all three networks to create a comprehensive feature vector that covers the benefits of each individual network. The fused feature vector is called VIR. Formally, the feature vector for the second approach is produced as follows (see Eq. (3)):

$$f_{VIR}(X_k) = [f_{VGG}(X_k) \quad f_{InceptionV3}(X_k) \quad f_{ResNet50}(X_k)] \quad (3)$$

Where the extracted features from all three networks are joined together to form the final representation matrix. (see Fig. 2)

Fig. 2. Illustration of extracting features from video using the second approach.

2.3 Transfer learning with fine-tuning

This approach utilizes the same structure described in section 2.1 with one modification that is instead of freezing the pre-trained layers, we fine-tune the layer's weights using sample images from the human fall detection dataset. To fine-tune the network using this dataset, we added a fully connected classification layer on top of the loaded network with two output nodes representing two classes of ADL and fall. The Softmax activation function is employed on the added layer to estimate the probability of belonging to each class (see Fig. 3). Moreover, RMSprop^a is used as the optimizer with a learning rate equal to 1e-4 and categorical cross-entropy as the loss function. This optimizer is a gradient-based optimization technique proposed by Geoffrey Hinton. The early stopping method on a validation set is employed to find the optimal number of epochs during the fine-tuning phase. After the fine-tuning phase is finished, we again use the output of the FC2 layer of the VGG16 network for extracting features. Because of the hardware limitation that we face, we could only experiment fine-tuning approach on the VGG16 network. We leave the application of fine-tuning on the rest of the networks (Inception V3, ResNet50) as future work.

Fig. 3. Fine-tuning of VGG16 based on the third approach.

^ahttp://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf

Similarly, Eq. (4) represents the feature vector produced in this approach where $\hat{y}_{i,j}(x)$ refers to the fine-tuned network.

$$f_{fine_tuned}(X_k) = \begin{bmatrix} \hat{y}_{VGG,FC2}(X_{k,1}) \\ \hat{y}_{VGG,FC2}(X_{k,2}) \\ \vdots \\ \hat{y}_{VGG,FC2}(X_{k,N}) \end{bmatrix} \quad (4)$$

3. Experiments

We designed a series of experiments to evaluate the performance of the proposed method and compared three suggested approaches. The experiments are conducted on two standard publicly available datasets for human fall detection with varying conditions. The following two datasets have been used in this study:

- UR Fall Detection dataset^b –URFD- (Kwolek and Kepski 2014): This dataset contains 30 RGB videos of falls and 40 RGB videos of ADLs. Five actors simulated the actions. The Fall activities include fall while walking and fall from the chair which are captured by two Microsoft Kinect cameras and an accelerometer sensor simultaneously. On the other hand, ADL activities include walking, sitting, lying on the ground, and crouching down are recorded by just one camera and accelerometer. In our experiments, we only considered RGB images and ignored the accelerometer data.
- TST Fall Detection Dataset V2^c –TSTFD- (Gasparrini et al. 2015): This dataset contains 132 depth videos of falls and 132 depth videos of ADLs. All these videos are collected by a Microsoft Kinect v2 sensor and an IMU. Both fall and ADL activities are simulated by 11 young actors. Each action is repeated three times by the actors. In this dataset, fall activities contain the following subcategories: falling from the front and ends up lying, falling backward and ends up lying, falling from the side and ends up lying, and falling from backward and ends up sitting. On the other hand, ADL activities include the following subcategories: sitting on a chair, walking back and forth, grasping an object from the floor, and laying down on the floor. In our

experiments, we considered RGB and depth images and ignored the accelerometer data.

In our experiments, the accelerometer data is ignored in both datasets because of the following two reasons. First collecting it produces a lot of restrictions for the seniors since not only they should always remember to wear the device but also wearing it all day long is not convenient. Second, we realized that passive camera images are enough for the system to produce an acceptable performance, and there is no need to employ extra sensors.

3.1 Experimental setup

In our experiments, we used two types of images to investigate which type produces better performance for fall detection. To achieve this, we used RGB videos from URFD and depth videos of TSTFD.

The first challenge is that the videos in both databases have a varying number of frames. The number of frames for all videos must be equal to create feature matrices with the same dimension. For solving this issue, by observing the data, it is clear that the majority of information about an activity happens at the end of the video. So, we calculated the average number of frames in all videos and selected this amount of frames from the end of each video. In our dataset, the average number of frames is 70. If the length of a video is less than 70 frames, we apply zero-padding (append black frames) at the beginning of the image sequence. On the other hand, we observed that information redundancy between two consecutive frames is high. To lessen this issue, we picked every second of the last 70 frames. Therefore, the final length of the selected sequence reduces to 35 frames.

The feature extraction process highly affects the performance of the final system. We studied three pre-trained neural networks (VGG16, Inception V3, and ResNet50) for this task. It is known that each of the mentioned networks has its own structure. So they can extract various features from an image and each of them focuses on a specific perception. Thus employing different networks could alter the performance of the final system.

Another parameter considered in our experiments is how to split a dataset into training and testing samples. Depending on whether the test data is easy or hard, the classification results vary. Therefore, in our experiments, we took advantage

^b<http://fenix.univ.rzeszow.pl/~mkepski/ds/uf.html>

^c<https://iee-dataport.org/documents/tst-fall-detection-dataset-v2>

of the k-fold cross-validation method with $k=4$. Eventually, we calculated the average and standard deviation of the 4-step evaluation results.

Furthermore, we compared the performance of two classifiers, Support Vector Machine (SVM) and Logistic Regression (LR). The rationale behind choosing these classifiers is that they are more robust in terms of classifying high dimensional data (Cortes and Vapnik 1995), (Walker and Duncan 1967). The code is implemented using Keras library on top of the TensorFlow backend written in Python 3.7.

3.2 Experimental results

In the first experiment, we investigated the impact of using different approaches for feature extraction and classification on the performance of the final system on the TSTFD dataset. Table 1 displays the results. As you can see, the best performance achieved by applying the logistic regression classifier on the features extracted from the fine-tuned VGG16 network. This result indicates that fine-tuning a pre-trained neural network is a powerful tool for solving the problems with a low number of samples. Furthermore, Fig. 4 visualizes the results for better comparison.

Table 1. methods and results for TSTFD dataset

Classifier	Network	Accuracy	Precision	Recall	F-score
SVM	VGG16	88.25±3.36	86.30±1.46	91.14±7.37	88.54±3.39
	Inception V3	86.74±5.85	82.87±6.41	93.48±4.85	87.75±4.48
	ResNet50	85.22±0.76	81.89±5.69	90.54±4.38	85.79±1.92
	VIR	90.15±1.95	86.78±2.78	94.75±4.35	90.52±2.17
	Fine-tuned VGG16	92.79±2.86	91.96±2.74	92.61±1.86	92.25±1.61
LR	VGG16	92.42±2.14	90.94±4.03	94.72±5.10	92.64±1.83
	Inception	90.15±5.86	82.49±5.96	95.75±5.35	92.31±5.64
	ResNet50	92.80±1.90	89.92±5.84	96.19±2.68	92.82±2.69
	VIR	93.18±1.95	90.88±2.74	96.32±5.57	93.39±1.86
	Fine-tuned VGG16	95.07±3.60	93.67±7.40	96.56±4.40	94.92±4.10

Fig. 4. results of all experiments for LR classifier on TSTFD dataset

According to the results for the LR classifier, the ResNet50 network achieved a higher F-score compared to VGG16 and Inception V3 when we used them individually. Alternatively, for SVM classifier, the best performance among individual networks achieved by VGG16. However, this score increased when we fused features from all three networks (VIR) both for SVM and LR. Ultimately,

by fine-tuning VGG 16 network, we reached the best performance.

In general, the logistic regression classifier achieved better performance compared to the support vector machine. We think one reason could be related to the use of the LBFGS algorithm in optimizing the parameters of this classifier. Another reason might be related to the huge

number of features in comparison to the number of training samples. The SVM usually underperforms when the number of features are very high compared to the number of training samples or the target classes overlap.

The results of our experiments on the URFD dataset are shown in Table 2. This dataset contains RGB images. As you can see, for this dataset, all of the proposed approaches achieved 100% accuracy. In other words, all the test samples were classified

correctly. This result illustrates the high performance of deep pre-trained neural networks to extract comprehensive features from RGB images. We know these networks are pre-trained on RGB images of the ImageNet dataset. We believe one reason for achieving better performance on RGB compared to depth images is related to the fact that both ImageNet and URFD datasets benefit the same image type (RGB).

Table 2. methods and results for URFD dataset

Classifier	Network	Accuracy	Precision	Recall	F-score
SVM	VGG16	100±0	100±0	100±0	100±0
	Inception V3	100±0	100±0	100±0	100±0
	ResNet50	100±0	100±0	100±0	100±0
	VIR	100±0	100±0	100±0	100±0
	Fine-tuned VGG16	100±0	100±0	100±0	100±0
LR	VGG16	100±0	100±0	100±0	100±0
	Inception	100±0	100±0	100±0	100±0
	ResNet50	100±0	100±0	100±0	100±0
	VIR	100±0	100±0	100±0	100±0
	Fine-tuned VGG16	100±0	100±0	100±0	100±0

All the results in both Table 1 and Table 2 are produced by averaging over four rounds of cross-validation. We have also included the standard deviation. To better understand the impact of using deep learning and transfer learning in human fall detection, we compared our experimental results with the final results of other studies that have used the same datasets. In our experiments, the highest accuracy for fall detection on the TSTFD dataset is 95.07%, while (Ghojogh, Mohammadzade, and Mokari 2017) reported the accuracy of 90.15% for fall detection on the same database. Also, (Manzi, Dario, and Cavallo 2017) gained 93.3% accuracy on the TSTFD dataset.

Although in the approach proposed by (Gasparrini et al. 2015), the accuracy of 99% is reported, it is not possible to compare their result with the results achieved in this study since Gasparrini et al. fused TSTFD depth images and accelerometer data. In the introduction section, we discussed that we had removed accelerometer data since gathering it produces many restrictions for the elderly. Last but not least, we obtained 100% accuracy on RGB images of the URFD dataset that effectively eliminates the need for comparing with other similar methods on this dataset.

4. Conclusions

In this paper, we proposed three transfer learning-based methods for human fall detection. We evaluated the methods on two standard datasets (TSTFD and URFD). The best results for the TSTFD dataset achieved by fine-tuning VGG16 network and logistic regression classifier. For the URFD dataset, all the methods could achieve 100 percent accuracy. The experiments indicate that the transfer learning approach could produce very promising results both on RGB and depth videos for detecting human falls. Moreover, it is shown that the results achieved by fine-tuning a pre-trained neural network could outperform deep individual networks and feature fusion approaches. Finally, we have compared the results with some of the reported results on the same standard datasets as the baseline to illustrate the effectiveness of the proposed transfer learning approach for solving the human fall detection problem.

Eventually, due to the hardware limitation, we left the application of fine-tuning on the rest of the networks (Inception V3, ResNet50) as future work. It would also be interesting to see how the performance of our system changes on the depth images if we use neural networks that are pre-trained on depth images instead of RGB images.

5. References

- Bourke, AK, JV O'Brien, and GM Lyons. 2007. "Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm." *Gait & posture* 26 (2):194-199.
- Bourke, Alan K, Pepijn Van De Ven, Mary Gamble, Raymond O'Connor, Kieran Murphy, Elizabeth Bogan, Eamonn McQuade, Paul Finucane, Gearóid ÓLaighin, and John Nelson. 2010. "Assessment of waist-worn tri-axial accelerometer based fall-detection algorithms using continuous unsupervised activities." 2010 Annual International Conference of the IEEE Engineering in Medicine and Biology.
- Cheng, Juan, Xiang Chen, and Minfen Shen. 2012. "A framework for daily activity monitoring and fall detection based on surface electromyography and accelerometer signals." *IEEE journal of biomedical and health informatics* 17 (1):38-45.
- Cortes, Corinna, and Vladimir Vapnik. 1995. "Support-vector networks." *Machine learning* 20 (3):273-297.
- Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. "Imagenet: A large-scale hierarchical image database." 2009 IEEE conference on computer vision and pattern recognition.
- Feng, Pengming, Miao Yu, Syed Mohsen Naqvi, and Jonathon A Chambers. 2014. "Deep learning for posture analysis in fall detection." 2014 19th International Conference on Digital Signal Processing.
- Fugini, Mariagrazia, Federica Cirilli, and Paolo Locatelli. 2015. "Integrated care solutions." *ERCIM News* 102:42.
- Gasparrini, Samuele, Enea Cippitelli, Ennio Gambi, Susanna Spinsante, Jonas Wåhslén, Ibrahim Orhan, and Thomas Lindh. 2015. "Proposal and experimental evaluation of fall detection solution based on wearable and depth data fusion." International Conference on ICT Innovations.
- Ghojogh, Benyamin, Hoda Mohammadzade, and Mozghan Mokari. 2017. "Fisherposes for human action recognition using Kinect sensor data." *IEEE Sensors Journal* 18 (4):1612-1627.
- Hazelhoff, Lykele, and Jungong Han. 2008. "Video-based fall detection in the home using principal component analysis." International Conference on Advanced Concepts for Intelligent Vision Systems.
- He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. "Identity mappings in deep residual networks." European conference on computer vision.
- Igual, Raul, Carlos Medrano, and Inmaculada Plaza. 2013. "Challenges, issues and trends in fall detection systems." *Biomedical engineering online* 12 (1):66.
- Kwalek, Bogdan, and Michal Kepski. 2014. "Human fall detection on embedded platform using depth maps and wireless accelerometer." *Computer methods and programs in biomedicine* 117 (3):489-501.
- Lee, Tracy, and Alex Mihailidis. 2005. "An intelligent emergency response system: preliminary development and testing of automated fall detection." *Journal of telemedicine and telecare* 11 (4):194-198.
- Li, Yun, KC Ho, and Mihail Popescu. 2012. "A microphone array system for automatic fall detection." *IEEE Transactions on Biomedical Engineering* 59 (5):1291-1301.
- Manzi, Alessandro, Paolo Dario, and Filippo Cavallo. 2017. "A human activity recognition system based on dynamic clustering of skeleton data." *Sensors* 17 (5):1100.
- Mastorakis, Georgios, and Dimitrios Makris. 2014. "Fall detection system using Kinect's infrared sensor." *Journal of Real-Time Image Processing* 9 (4):635-646.
- Mirmahboub, Behzad, Shadrokh Samavi, Nader Karimi, and Shahram Shirani. 2012. "Automatic monocular system for human fall detection based on variations in silhouette area." *IEEE Transactions on Biomedical Engineering* 60 (2):427-436.
- Shojaei-Hashemi, Anahita, Panos Nasiopoulos, James J Little, and Mahsa T Pourazad. 2018. "Video-based Human Fall Detection in Smart Homes Using Deep Learning." 2018 IEEE International Symposium on Circuits and Systems (ISCAS).
- Simonyan, Karen, and Andrew Zisserman. 2014. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556*.
- Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. "Rethinking the inception architecture for computer vision." Proceedings of the IEEE conference on computer vision and pattern recognition.
- Teimourikia, Mahsa, and Mariagrazia Fugini. 2017. "Ontology development for run-time safety management methodology in Smart Work Environments using ambient knowledge." *Future Generation Computer Systems* 68:428-441.
- Walker, Strother H, and David B Duncan. 1967. "Estimation of the probability of an event as a function of several independent variables." *Biometrika* 54 (1-2):167-179.