Video-based Human Fall Detection in Smart Homes Using Deep Learning

Anahita Shojaei-Hashemi[1], Panos Nasiopoulos[1],
James J. Little[2]
[1]Department of Electrical & Computer Engineering
[2]Department of Computer Science
University of British Columbia
Vancouver, Canada

Mahsa T. Pourazad
TELUS Communications Inc.
Vancouver, Canada

***Abstract*—Automatic human fall detection is a challenging task of healthcare in smart homes, and video cameras have been proved to be efficient in addressing this problem. Although existing methods perform relatively well, they are all built upon "hand-crafted" features, thus constraining the performance of the model to some presumed conditions and scenarios, and making it vulnerable to any deviation from the assumed settings. In this paper, we propose a deep-learning-based approach for human fall detection, using long short-term memory neural network. Our model is not restricted to any specific circumstances, and performance evaluations show that it outperforms all the existing methods.**

Keywords—human fall detection; deep learning; long short-term memory (LSTM); smart home; depth camera

I. INTRODUCTION

The concept of a "smart home" is a major step towards wellness and improved quality of life and a hot interdisciplinary research topic bringing together artificial intelligence, cloud computing, communications and networks, psychology and healthcare [1]. Monitoring the well-being of the residents is an expected service to be provided by a smart home. Although older adults, patients newly released from hospital and chronically ill people are more vulnerable to falling, everyone is prone to it. It can be specifically dangerous for people living alone, as the incident might take hours or even days to be discovered with the person remaining injured or unconscious, while it is very important, if not vital, for the fallen person to be immediately taken care of and possibly transported to the hospital. Hence, there is a need for a monitoring system, in a smart residence, which can automatically detect fall incidents and send alarms to the emergency medical center and/or family and friends.

Devices used for fall detection are divided into three categories: wearable devices, ambient sensors, and video cameras, each having its own pros and cons, which makes them complementary to each other. Wearable devices are relatively inexpensive and can directly measure kinematic quantities. However, they cannot recognize complex physical motions and can be considered intrusive [2]. Ambient sensors, on the other hand, are versatile in application and can be installed anywhere or embedded in any object without interfering with the resident's daily life. Nevertheless, limited information can be obtained out of the data collected by each sensor, so a plenty of sensors should be installed, which is costly and difficult. Also, the data has high noise to signal ratio. Video cameras can be conveniently mounted in various places of the residence, and they collect data with rich information content that can be used for several tasks. The downside, however, is that cameras are considered intrusive and introduce privacy issues if video content is captured and stored. Nowadays, inexpensive depth cameras, such as the Microsoft Kinect, can address some of the privacy issues and under proper implementation conditions could be a promising and feasible option for human fall detection in the context of smart home.

Limited research has been done so far on depth-camera-based fall detection. In [3], Marzahl et al. apply a manual pre-segmentation to exclude areas where falling is not possible to happen, e.g., cupboards. Then, a decision tree detects any person lying on the floor by exploiting spatial characteristics of segmented objects. Out of the 55 fall samples collected in a laboratory setting, 93% of them have been classified correctly. In [4], Mastorakis and Markis use the 3D bounding box of the subject, extracted by OpenNI framework [5], and employ thresholding on the height, width, and depth of the bounding box as well as their first derivatives to detect falling incidents. All falls have been detected with no false alarms for a dataset of 48 fall samples and 112 non-fall samples, collected in a laboratory setting. Bian et al. in [6] extract the key joints of the human body using their own proposed algorithm, and apply support vector machine (SVM) on the 3D trajectory of the head joint to detect fall incidents. In evaluation on a dataset of 380 samples, collected in a laboratory setting and containing equal number of fall and non-fall samples, all falls have been detected and 9 false alarms have been generated. In [7], Rougier et al. detect the ground plane of the room, segment the person from the background, localize and track the 3D centroid of the person, and detect falls by thresholding the centroid height relative to the ground and the centroid velocity. Only one fall has not been detected out of 25 fall samples, and no false alarms have been reported on 54 non-fall samples. All the samples have been collected in a laboratory setting. In [8], Planinc and Kampel employ the skeletal data extracted by Microsoft Software Development Kit (SDK) for Kinect to calculate the orientation of the person's major axis and the height of the spine relative to the ground. These features are then used for fall detection. The

978-1-5386-4881-0/18/$31.00 ©2018 IEEE

method has been evaluated on a data set consisting of 40 fall samples and 32 non-fall samples collected in a laboratory setting. 37 falls have been detected with five false alarms.

Although the existing methods perform relatively well, in all of them fall detection is based on video features which are manually chosen by the user. Designing these "hand-crafted" features inevitably involves making several assumptions about the problem. This not only requires background knowledge on the application field, but also constrains the performance of the model to some presumed conditions and scenarios, and makes the trained model sensitive to any deviation from the training data. Furthermore, expanding the model or modifying it for different though similar tasks is difficult and usually requires redesigning the features.

In this paper, we propose a deep-learning-based approach for human fall detection, using long short-term memory (LSTM) neural network. In contrast to classic methods whose performance is limited by the presumed conditions, the deep neural network takes care of feature design itself and extracts the most discriminative features based on the training data, and hence, covers more real-life scenarios. To overcome the inherent requirement of deep learning approaches for huge training data sets, we employ transfer learning. We trained and tested our proposed approach on the "NTU RGB+D Action Recognition Dataset" [9], which is a recently released public dataset consisting of about 50000 samples on human actions including falling. The results show that our deep-learning-based model has great performance in detecting falls, and it outperforms all the existing methods which are based on hand-crafted features.

The rest of this paper is organized as follows: section II provides a background overview on LSTM neural networks and transfer learning. Section III explains the proposed method in detail. Section IV discusses the experimental results, and section V concludes the paper.

II. Background Overview

A. Long Short-Term Memory (LSTM) Neural Networks

In machine learning, there are many scenarios where the output at each point (of time, space, etc.) depends not only on the input at the same point, but also on the inputs at other points. Such data is referred to as sequential data, and classifying such data is called sequence labeling. Data can be sequential along any continuous or discrete dimension, such as time and space. For the sake of simplicity and without loss of generality, in the rest of this section we assume the data is sequential in time. In the context of deep learning, recurrent neural network (RNN) is a prevalent option for sequence labeling [10], [11], [12]. The difference between regular neural network, which is also called feed-forward network, and RNN is the presence of a feedback loop in RNN. When unfolded, this loop produces a recurrent connection in the network, which models the correlation existing among the elements of the sequential data.

In order to have an RNN with more than one hidden layer, at each time step, the hidden state of each layer should be fed as the input to the next hidden layer of the same time step, in addition to getting passed to the hidden layer of the next time step at the same level. An RNN can be deep either layer-wise or time-wise, or both. Nevertheless, a regular RNN is not trained well if it is deep in time, because of a phenomenon called "vanishing gradient". To overcome this issue, so that the neural network can also handle long sequences, the long short-term memory (LSTM) neural network was introduced as a modified version of RNN [13].

B. Transfer Learning

Most of machine learning methods work well only when the test data is drawn from the same feature space and have the same feature distribution as the training data. Thus, for a new application where the data has a different feature space or different feature distribution, the model should be rebuilt from scratch using a new set of training data. Nevertheless, this is not always feasible in practice, since it is usually expensive, if possible, to collect the required training data, and it is time and resource consuming to retrain the model. Transfer learning, or knowledge transfer, tries to address this issue by permitting the feature distribution of the training and the test data to be different. Instead of learning the new task from scratch, transfer learning tries to transfer the knowledge learnt in the previous task to the new one. In other words, transfer learning aims to extract knowledge from one or more source tasks and to apply it to a target task [14].

Transfer learning can be of great benefit to deep models, as they need abundant data to get their plentiful parameters trained on, while the amount of data is limited in many applications. Transfer learning in deep neural networks working with visual data is based on the interesting fact that regardless of the dataset and the objective function, such networks in their first layer tend to identify basic features such as lines and corners [15]. Therefore, features generated by their first layer can be considered as "general" features. Ascending through the layers, the learnt filters become more dependent on the dataset and the objective function, so the extracted features get more "specific". Based on this fact, transfer learning for deep neural networks takes the following path: First, the source network is trained on the source dataset and task. Next, the weights of the first few layers are copied to the target network to transfer the learnt general features. The rest of the layers of the target network are then initialized and trained on the target data, like usual. Finally, the transferred layers are either fine-tuned or left unaltered. The choice depends on the size of the target dataset and the number of parameters in the transferred layers to avoid overfitting.

III. Proposed Method

As previously mentioned, our goal is to design a video-based deep model for human fall detection in indoor environments to be used in a smart home setting. In order for the model to be robust against changes in lighting conditions, e.g., to perform even in darkness, and for preserving the privacy of the residents, we opt depth cameras, which produce depth maps instead of regular images. Considering the fact that the 3D locations of major body joints carry most of the body kinematic information required for discriminating different actions, keeping track of the body joints, as shown by our evaluations, proves to be sufficient for action recognition and fall detection, while it is computationally much cheaper. Since the existing algorithms to extract skeletons from depth map, such as the one provided by the Microsoft Software Developer Kit (SDK), process the video

978-1-5386-4881-0/18/$31.00 ©2018 IEEE

sequence frame by frame, the use of body skeleton information by our model can be implemented in real-time.

There is correlation among the states of the body skeleton at different time steps. To exploit this sequential information intrinsically embedded in the input sequence, we choose RNN as our deep learning model. As actions usually take place within a long sequence of frames, vanilla RNN encounters the vanishing gradient issue, so we specifically take LSTM, which can go deep in time. As a deep neural network, LSTM requires abundant training data. However, the number of fall incident samples is limited, as it occasionally happens in real life and it is difficult to simulate in laboratory settings. On the other hand, large amount of data is available on ordinary actions, such as walking, drinking, picking up, etc. Thus, we propose a way to make use of this data to compensate for the lack of fall samples, by employing transfer learning in training the LSTM.

As illustrated in Fig. 1, we first train a multi-class LSTM on the abundant samples of human regular actions. Then, we transfer all the learnt weights, except for those of the last layer, to a two-class LSTM that is designed for fall detection. Finally, we train the last layer of the two-class LSTM on scarce human fall samples in combination with part of the regular action samples. The structures of the two LSTMs are the same. They only differ in the last layer, where the number of units equals the number of classes. To prevent the LSTMs from getting biased to training data, i.e. overfitting, we apply "dropout" regularization [16].

There are several parameters of the designed model which need to be set. These include the depth of the LSTM in time, the number of the layers, the number of the hidden units in each layer, and the dropout ratio. They can be selected either heuristically or through random search optimization.

IV. Experimental Results and Discussion

We evaluated the performance of our proposed method on the "NTU RGB+D Action Recognition Dataset". This dataset contains 56,880 video samples, each consisting of one action. The samples encompass 60 different actions, including falling, and are performed by 40 subjects, aging from 10 to 35. 11 of the actions involve two persons, and the rest are solo. The videos have been captured through three synchronous Microsoft Kinect v2 cameras, which were installed at the same height with three different horizontal angels: −45°, 0°, +45°. Each sample is available in 4 modalities: RGB video, depth map sequence, skeleton sequence, and infrared (IR) video. All the modalities have the speed of 30 fps. The resolution of depth map frames, RGB frames, and IR frames are 512 × 424, 1920 × 1080, and 512 × 424, respectively. The skeletons are composed of the 3D coordinates of 25 major body joints, and each frame contains up to two skeletons. The corresponding pixel to each joint is also provided for RGB frames and depth maps [9]. As explained before, the input of our proposed model is skeleton sequence, so we only used the skeleton modality of NTU dataset. Some of the skeleton sequences in this dataset have missing frames, skeletons, or joints. We removed those sequences as well as the actions involving more than one person. Consequently, the total number of samples we used was 44372, out of which 890 were falling samples. We used 75% of the samples for training, and the rest for testing. 20% of the training data was used for cross-validation. The maximum number of training iterations is 100000, while the training is halted if the value of the loss function for the validation set does not decrease for five consecutive epochs.

To design the deep neural network, considering that the lengths of the sequences were not much different, we set the depth of the LSTM in time as the average length of the training samples, which was 90, and trimmed the longer sequences and zero-padded the shorter ones. Rectified linear unit (ReLU) and Softmax were chosen as the activation function and the loss function, respectively. The other parameters of the model were set so that the performance of the method is optimized. The performance was evaluated based on the area under the curve (AUC) metric, calculated for the receiver operating characteristic (ROC) curve. To select the number of layers, we started from a single-layer LSTM and increased the number of

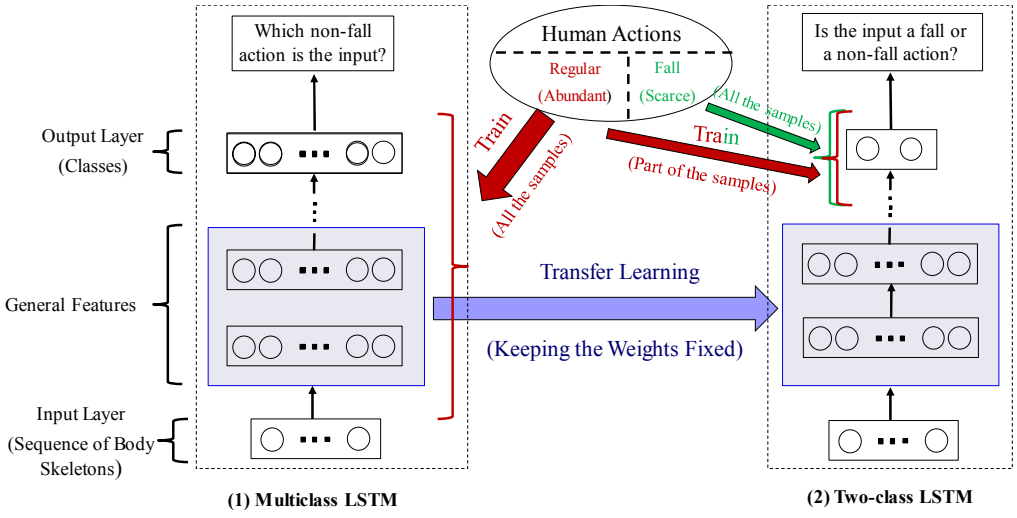

Fig. 1. The overall structure of our proposed method

978-1-5386-4881-0/18/$31.00 ©2018 IEEE

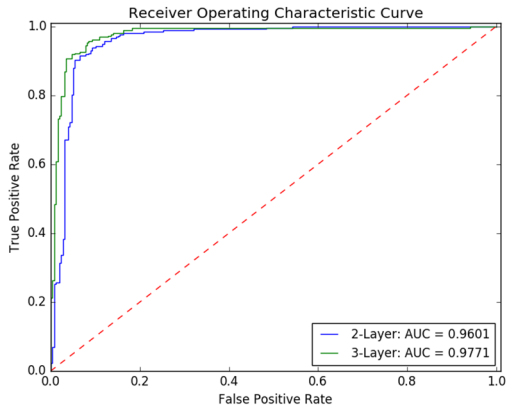

Fig. 2. The ROC curves and their corresponding AUCs for different numbers of the layers

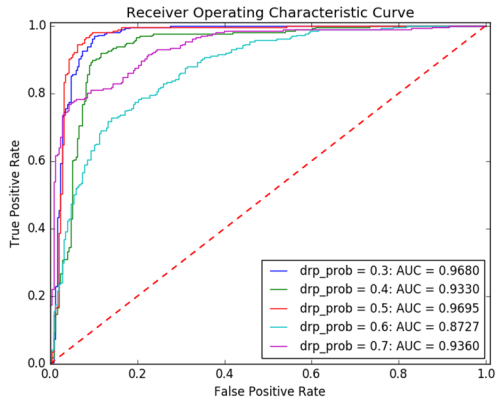

Fig. 3. The ROC curves and their corresponding AUCs for different dropout ratios

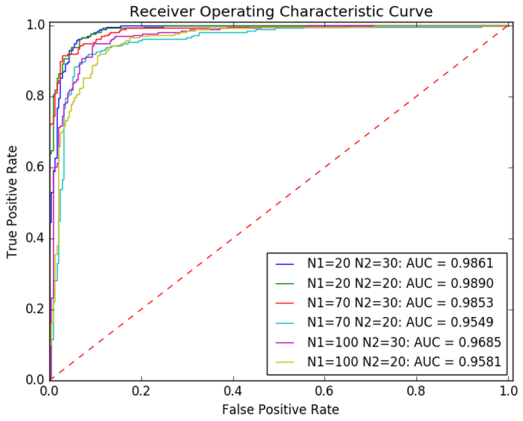

Fig. 4. The ROC curves and their corresponding AUCs for different numbers of the hidden units in both layers

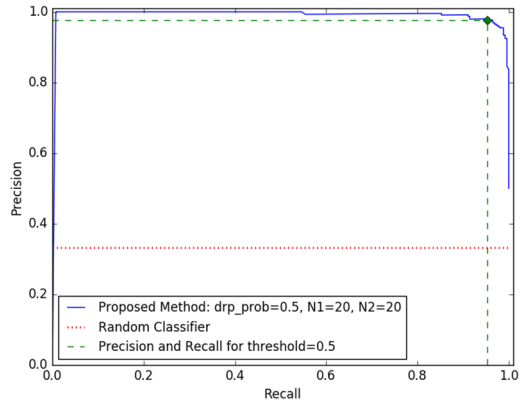

Fig. 5. The predision-recall curve for the optimal parameters

layers up to three. As shown in Fig. 2, the performance of the two-layer LSTM was good enough and adding the third layer did not improve it much, so we did not go above three layers, and set the number of the layers at two. The dropout ratio and the number of the hidden units were selected through random search. According to Fig. 3, the performance of the model is the best at the dropout ratio of 0.5. Fig. 4 shows that the best performance is obtained with 20 hidden units for both layers, though the combination of 70 units for the first layer and 30 units for the second layer is also close to optimum.

In order to compare our proposed model with the existing methods, we had to use a metric other than AUC, since the other papers do not provide AUC for their methods. This metric, which is not as comprehensive as AUC in evaluating the performance, consists of precision and recall values at the fixed threshold of 0.5. Fig. 5 presents the precision-recall curve of our optimized model, showing the precision and recall values at the threshold of 0.5. Table 2 compares the performance of our model with that of the best depth-map-based fall detection methods, which are all built using hand-crafted features. With 93% precision and 96% recall, our deep model outperforms Rougier's [7] and Plannic's [8] with considerable margins.

TABLE I. THE PERFORMANCE OF OUR PROPOSED MODEL COMPARED TO THE STATE-OF-THE-ART- METHODS

Method	Precision	Recall
Rougier's [7]	0.6861	0.9894
Plannic's [8]	0.8177	0.9210
Ours	0.9323	0.9612

V. CONCLUSION

In this work, we proposed a deep learning model for human fall detection in videos captured by depth cameras, with potential application in smart homes. Our approach takes depth map sequences and determines if a falling incident has happened, so that an alarm can be generated and sent to family/friends or medical service staff. This is the first deep-learning-based method in the field of human fall detection. It has area under the ROC curve of 0.99, and it outperforms all the existing fall detection algorithms, which are based on hand-crafted features.

REFERENCES

[1] R. Harper, Inside the smart home, Springer Science & Business Media, 2006.

[2] L. a. K. I. Chen, "Activity recognition: Approaches, practices and trends," *Activity Recognition in Pervasive Intelligent Environments,* pp. 1-31, 2011.

978-1-5386-4881-0/18/$31.00 ©2018 IEEE

[3] C. Marzahl, P. Penndorf, I. Bruder and M. Staemmler, "Unobtrusive fall detection using 3D images of a gaming console: Concept and first results," *Ambient Assisted Living,* pp. 135-146, 2012.

[4] G. Mastorakis and D. Makris, "Fall detection system using Kinect's infrared sensor," *Journal of Real-Time Image Processing,* vol. 9, no. 4, pp. 635-646, 2014.

[5] "OpenNI 2 Downloads and Documentation | The Structure Sensor," 2017. [Online]. Available: https://structure.io/openni.

[6] Z.-P. Bian, J. Hou, L.-P. Chau and N. Magnenat-Thalmann, "Fall detection based on body part tracking using a depth camera," *IEEE journal of biomedical and health informatics,* vol. 19, no. 2, pp. 430-439, 2015.

[7] C. Rougier, E. Auvinet, J. Rousseau, M. Mignotte and J. Meunier, "Fall detection from depth map video sequences," *Toward useful services for elderly and people with disabilities,* pp. 121-128, 2011.

[8] R. Planinc and M. Kampel, "Introducing the use of depth data for fall detection," *Personal and ubiquitous computing,* vol. 17, no. 6, pp. 1063-1072, 2013.

[9] A. Shahroudy, J. Liu, T.-T. Ng and G. Wang, "NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[10] D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning representations by back-propagating errors," *nature,* vol. 323, no. 6088, p. 533, 1986.

[11] J. L. Elman, "Finding structure in time," *Cognitive science,* vol. 14, no. 2, pp. 179-211, 1990.

[12] P. J. Werbos, "Generalization of backpropagation with application to a recurrent gas market model," *Neural networks,* vol. 1, no. 4, pp. 339-356, 1988.

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

[14] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering,* vol. 22, no. 10, pp. 1345-1359, 2010.

[15] J. Yosinski, J. Clune, Y. Bengio and H. Lipson, "How transferable are features in deep neural networks?," in *Advances in neural information processing systems*, 2014.

[16] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research,* vol. 15, no. 1, pp. 1929-1958, 2014.

978-1-5386-4881-0/18/$31.00 ©2018 IEEE