Fall Detection from Thermal Camera Using Convolutional LSTM Autoencoder

Jacob Nogas, Shehroz S. Khan, Alex Mihailidis
University of Toronto, Canada
{jacob.nogas@mail., shehroz.khan@, alex.mihailidis@}utoronto.ca

Abstract

Human falls occur very rarely; this makes it difficult to employ supervised classification techniques. Moreover, the sensing modality used must preserve the identity of those being monitored. In this paper, we investigate the use of thermal camera for fall detection, since it effectively masks the identity of those being monitored. We formulate the fall detection problem as an anomaly detection problem and aim to use autoencoders to identify falls. We also present a new anomaly scoring method to combine the reconstruction score of a frame across different video sequences. Our experiments suggests that Convolutional LSTM autoencoders perform better than convolutional and deep autoencoders in detecting unseen falls.

1 Introduction

Each year, more than one out of four older people fall, and one out of five falls causes a serious injury such as broken bones or a head injury [CDC, 2017]. Among nursing home residents, on an average, 2.6 falls per person per year occur [Rubenstein *et al.*, 1990]. Detecting falls is important from the perspective of health and safety; however, due to their infrequent occurrence per person, it is difficult to collect sufficient training data for them [Khan *et al.*, 2017]. In such highly skewed situations, it is hard to employ supervised classification techniques. There are also concerns that the sensing devices for the task of fall detection may be invasive and breach the privacy of a person [Mercuri *et al.*, 2016; Yusif *et al.*, 2016]. Some non-invasive sensing devices, such as a thermal or depth camera, may not reveal the identity of a person, but it is difficult to extract discriminative features to identify unseen falls from these sensing devices [Skubic *et al.*, 2016]. It is thus desirable to find a model which can learn discriminative features from the data captured by non-invasive sensors.

In this paper, we take an alternative approach to fall detection, by formulating it as an anomaly detection problem, due to the rare occurrence of a fall. A deep autoencoder (DAE) can be used to learn features by training on the normal activities of daily living (ADL) and identify a fall as an anomaly based on the reconstruction error [Khan and Taati, 2017]. However, traditional autoencoders ignores the 2D structure of images and may force each feature to be global, and thus span the entire visual field [Masci *et al.*, 2011]. In visual recognition tasks, convolutional autoencoders (CAE) perform better because they can discover localized spatial features that repeat themselves over the input [Masci *et al.*, 2011]. Further, a video sequence embeds information in both space and time; therefore, autoencoders that can learn a representation of local spatio-temporal patterns of frames in a video can be more useful [Baccouche *et al.*, 2012]. This motivates us to use a convolutional LSTM autoenocder (ConvLSTM-AE) to learn spatio-temporal features from ADL videos. In this paper, we train DAE, CAE, DAE and ConvLSTM-AE on *only* the normal ADL. An anomaly score for video frames is then calculated, which can be used to identify an unseen fall during the testing phase.

2 Related Work

In this section, we present a brief review of the literature that uses DAE, CAE, and ConvLSTM-AE for anomaly detection in videos and other data types. Sabokrou et al. [Sabokrou *et al.*, 2016] use a sparse DAE to detect key-points in videos, and then train a non-sparse DAE on patches formed around these key points. The reconstruction error of these patches is used to determine if they are anomalous or not. Hasan et al. [Hasan *et al.*, 2016] use both hand-crafted spatio-temporal features and CAE to learn regular motion patterns in videos. They introduce a regularity score that scales the reconstruction error of a frame between 0 and 1. However, it requires minimum and maximum values of reconstruction error across all the test samples, which may not have been observed. They show competitive performance of their method to other state-of-the-art anomaly detection methods. Chong and Tay [Chong and Tay, 2017] present a method to detect anomalies in videos that consists of a spatial feature extractor and a temporal encoder-decoder framework based on convolutional long short term memory layers (more details in section 3.2). Their model is trained only on the videos with normal scenes. They use a regularity score [Hasan *et al.*, 2016] to identify anomalies. They show comparable performance in comparison to other standard methods; however, it may produce more false alarms. Medel et al. [Medel and Savakis, 2016] detect anomalies in video using a predictive convolutional long-short term memory networks. Unlike [Chong and

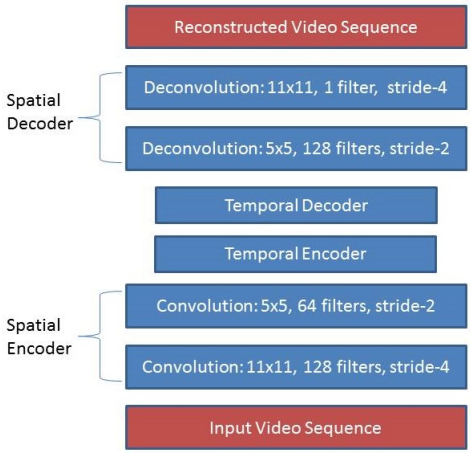


Figure 1: ConvLSTM-AE architecture [Chong and Tay, 2017]

Tay, 2017], they use only ConvLSTM layers for both encoding and decoding. Their decoding also differs in that it uses both a past decoder and future decoder. The future decoder predicts future frames, and past decoder attempts to reconstruct the original input. They also use a regularity score to identify anomalies.

The literature review suggests that using spatio-temporal CAE to learn features is a plausible approach to identify anomalies, which are falls in our case. In the next section, we briefly describe the ConvLSTM-AE.

3 Convolutional LSTM Autoencoder

The ConvLSTM-AE architecture used in our experiments is proposed by Chong and Tay [Chong and Tay, 2017], which uses convolutional layers for spatial encoding/decoding, and ConvLSTM layers for spatio-temporal encoding/decoding (see Figure 1). In particular, a window of T contiguous video frames are passed as input to the network. The spatial encoder takes each of these frames one at a time. The spatial encoder consists of two 2D convolution layers with full padding and stride 4×4, and 2×2. This results in spatial dimension reduction, and produces T encoded features, which are concatenated and fed into the temporal encoder. The decoder repeats this process in reverse to reconstruct the video window.

In order to generate windows of contiguous frames to give as input the the ConvLSTM-AE, we apply a sliding window with stride $S = 1$, and window length T. The stride represents the amount of shift by frames in subsequent windows. The sliding window process is continued until all frames are selected (we do not use any padding). If a video contains V frames, then the number of windows (D) generated is

$$D = \lfloor \frac{V - T}{S} \rfloor + 1 \tag{1}$$

For training ConvLSTM-AE, we use mean squared error loss between input windows (I) and reconstructed output windows (O), optimized on a per batch basis, giving the following cost function:

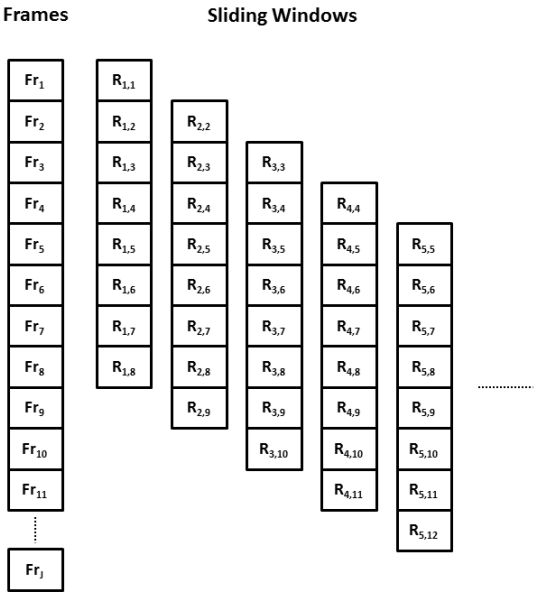


Figure 2: Temporal sliding window showing reconstruction error ($R_{i,j}$) per frame (Fr_j) .

$$C(\theta) = \frac{1}{N} \sum_{i=1}^{N} \|I_i - O_i\|_2^2 \tag{2}$$

where N is the number of training samples in a batch, θ denotes the network parameters, and $\|\cdot\|_2$ denotes the Euclidean norm.

4 Detecting Unseen Falls

For CAE and DAE, the input to the network is a single frame; therefore, the reconstruction error is computed per frame. For ConvLSTM-AE, the input to the network is a video sequence. That is, given a test video sequence, we apply a sliding window as described in previous section. For the i^{th} window I_i, the network outputs a reconstruction of this window, O_i. The reconstruction error for the j^{th} frame ($R_{i,j}$) of I_i can be calculated as

$$R_{i,j} = \|I_{i,j} - O_{i,j}\|_2^2 \tag{3}$$

Figure 2 shows this sliding window approach, with $T = 8$. The first window of $T = 8$ frames, I_1 (Fr_1 to Fr_8) are reconstructed, and their corresponding reconstruction error is stored ($R_{1,1:8}$). For the next window of frames, the input window is shifted forward in time by one frame. This process continues until all frames are used.

4.1 Cross-Context Anomaly Score

To compare ConvLSTM-AE with CAE and DAE, we must get a score per frame. Notice, a frame can appear in multiple windows. For instance, frame 2 (see Figure 2) is given two reconstruction errors: $R_{1,2}$, and $R_{2,2}$. The former is attained with frame 2 as the second frame, and the latter with frame 2 as the first frame of the input window. Each window that a

frame appears in provides a different temporal *context* within which this frame can be viewed. The cross-context anomaly score gives scores on a per-frame basis, by considering all of the reconstruction errors obtained for a frame across different windows. For the j^{th} frame of the i^{th} window, an anomaly score can be computed based on the mean (C_μ^j) or standard deviation (C_σ^j) of the reconstruction error across contexts:

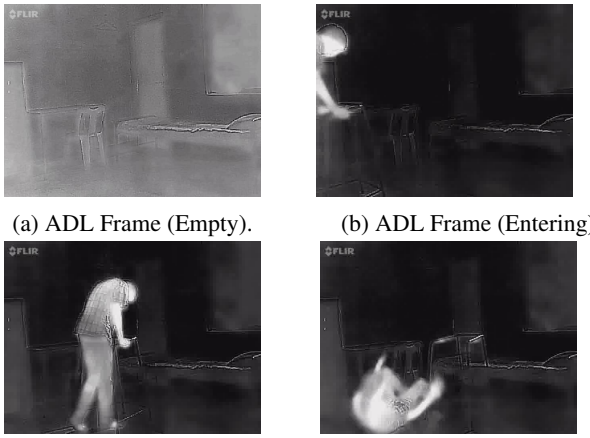
$$C_\mu^j = \begin{cases} \frac{1}{j}\sum_{i=1}^{j} R_{i,j} & j < T \\ \frac{1}{T}\sum_{i=1}^{T} R_{i,j} & j \geq T \end{cases}$$
$$C_\sigma^j = \begin{cases} \sqrt{\frac{1}{j}\sum_{i=1}^{j}(R_{i,j} - C_\mu^j)} & j < T \\ \sqrt{\frac{1}{T}\sum_{i=1}^{T}(R_{i,j} - C_\mu^j)} & j \geq T \end{cases} \quad (4)$$

A large value of C_μ^j or C_σ^j means that the j^{th} frame, when appearing at different positions in subsequent windows, is reconstructed with a high average error or highly variable error. In a normal ADL case, the reconstruction error of a frame should not vary a lot with its position in subsequent windows; however, if it does, then this may indicate anomalous behaviour, such as a fall.

5 Experiments and Results

5.1 Thermal Fall Dataset

We test our framework on the Thermal Fall Detection Activity Recognition dataset. This dataset consists of videos captured by a FLIR ONE thermal camera mounted on an Android phone in a room setting with a single view [Vadivelu *et al.*, 2016]. A total of 44 videos are collected, out of which 35 videos contain a fall event ($36,391$ frames total, 828 fall frames), and 9 videos ($22,116$ frames) contain only ADL. The resolution of the thermal images is 640×480. Some example frames of ADL of thermal dataset are shown in Figure 3.



(a) ADL Frame (Empty). (b) ADL Frame (Entering). (c) ADL Frame. (d) Fall Frame.

Figure 3: Thermal Data ADL and Fall Frames.

5.2 Data-preprocessing

The thermal data set frames were extracted from video files. The videos had frame rates of either 25 fps, or 12 fps (observed from their properties). All of the frames in the dataset are normalized by dividing pixel values by 255 to keep them in the range $[0, 1]$, and subtracting the per-frame mean from each frame, resulting in pixel values to be in the range $[-1, 1]$. All frames are re-sized to 64×64. The video frames used for training the models were not annotated because all of them were considered normal ADL. In order to test the models, fall videos are used that contained both fall and non-fall frames. The fall frames were manually annotated in these videos.

In order to create windows of contiguous video frames to give as input to the ConvLSTM-AE for training and testing, we perform a sliding window on all video frames, as described in 3, with window length of $T = 8$. The thermal data contains $22,116$ ADL frames from 9 videos. After windowing the ADL videos, we generate $22,053$ (using equation 1) windows of contiguous frames, which are used for training ConvLSTM-AE.

5.3 Experiments and Results

To evaluate the performance on detecting unseen falls, we compare the DAE, CAE, and ConvLSTM-AE. The DAE and CAE specifications are shown in Table 1. This table only shows the configuration of the encoding phase of the autoencoder. The decoding configuration is the same, but using up-sampling, or deconvolution for CAE (denoted CAE Up-Sampling, and CAE Deconv. in results section), and fully connected layers for DAE. Also, dropout is applied to layer 1 for DAE. Deconvolution and up-sampling layers are used as defined in Keras [Chollet and others, 2015]. For up-sampling, we use an up-sampling factor of 2×2, and for deconvolution, a 3×3 filter is used, with padding, and stride 2×2. For Convolution, a filter size of 3×3 is used. Max-pooling uses a pool size of 2×2, stride 2×2, and padding.

CAE	DAE
Input - (64, 64, 1)	Input - (64, 64, 1)
2D Convolution - (64, 64, 16)	Fully Connected - (4096)
2D Max-pooling - (32, 32, 16)	Fully Connected - (150)
2D Convolution - (32, 32, 8)	Fully Connected - (100)
2D Max-pooling - (16, 16, 8)	Fully Connected - (50)
2D Convolution - (16, 16, 8)	-
2D Max-pooling - (8, 8, 8)	-

Table 1: Configuration of the encoding phase of CAE and DAE.

We trained CAE and DAE for 200 epochs, and trained ConvLSTM-AE for 50 epochs [Chong and Tay, 2017]. Adadelta optimizer was used in training all models. The training batch size is set to 32 for DAE and CAE, and 16 for ConvLSTM-AE, where each batch consists of windows of 8 frames. To train DAE and CAE, we augment the data by performing horizontal flipping. No data augmentation was performed when training ConvLSTM-AE, as it did not improve results.

The cross-context anomaly score gives a reconstruction error per frame across different windows (for a given video), which can be used as a score to identify a fall frame as an anomaly. For CAE and DAE, the per frame reconstruction error is computed directly, and used as an anomaly score. These

anomaly scores obtained for every frame are used to calculate ROC AUC, with fall as the class of interest. All models and training procedures are implemented in Keras with Tensorflow backend [Chollet and others, 2015].

Table 2 shows the AUC results for all models – DAE, CAE and ConvLSTM-AE. The reported AUC is the average of AUC across all test videos, with standard deviation in brackets. We observe that CAE Up-Sampling slightly performs better than CAE-Deconv, and both of them perform better than DAE. Since DAE computes features on an image globally, it may fail to exploit the spatial structure present in the data. We also observe that ConvLSTM-AE with C_μ performs equivalent to CAE; however, ConvLSTM-AE with C_σ score performs better than CAE. This may be because ConvLSTM incorporates the spatial structure of the input image similar to CAE, but also captures the temporal information of video data, which is an important component of distinguishing a fall from ADL.

Model	ROC AUC
DAE	0.64 (0.15)
CAE Deconv.	0.70 (0.16)
CAE Up-Sampling	0.75 (0.13)
ConvLSTM-AE with C_μ	0.76 (0.13)
ConvLSTM-AE with C_σ	**0.83** (0.11)

Table 2: Average AUC values for different models for Thermal dataset, with standard deviation in brackets

6 Conclusions and Future Work

Detecting falls in a non-invasive manner is a challenging problem; especially as falls occur rarely. In this paper, we formulated detecting falls as an anomaly detection problem and propose to use autoencoders to do the task. We also proposed a new method of computing anomaly scores called the cross-context anomaly score. We tested various autoencoder approaches on a dataset that captured ADL and falls in a non-invasive manner using a thermal camera. The results showed that DAE was outperformed by CAE, and ConvLSTM-AE outperformed CAE, indicating that incorporating spatial, as well as temporal information was effective in detecting unseen falls. In future, we plan to employ 3D convolutional autoencoders to advance our research in this direction.

References

[Baccouche *et al.*, 2012] Moez Baccouche, Franck Mamalet, Christian Wolf, Christophe Garcia, and Atilla Baskurt. Spatio-temporal convolutional sparse autoencoder for sequence classification. In *In BMVC*. Citeseer, 2012.

[CDC, 2017] CDC. enter of Disease Control and Prevention – Important Facts about Falls. `https://www.cdc.gov/homeandrecreationalsafety/falls/adultfalls.html`, 2017. [Online accessed 22-December-2017].

[Chollet and others, 2015] François Chollet et al. Keras: The python deep learning library. `https://github.com/fchollet/keras`, 2015. Online accessed 20-January-2018.

[Chong and Tay, 2017] Yong Shean Chong and Yong Haur Tay. Abnormal event detection in videos using spatiotemporal autoencoder. In *International Symposium on Neural Networks*, pages 189–196. Springer, 2017.

[Hasan *et al.*, 2016] Mahmudul Hasan, Jonghyun Choi, Jan Neumann, Amit K Roy-Chowdhury, and Larry S Davis. Learning temporal regularity in video sequences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 733–742, 2016.

[Khan and Taati, 2017] Shehroz S Khan and Babak Taati. Detecting unseen falls from wearable devices using channel-wise ensemble of autoencoders. *Expert Systems with Applications*, 2017.

[Khan *et al.*, 2017] Shehroz S Khan, Michelle E Karg, Dana Kulić, and Jesse Hoey. Detecting falls with x-factor hidden markov models. *Applied Soft Computing*, 55:168–177, 2017.

[Masci *et al.*, 2011] Jonathan Masci, Ueli Meier, Dan Cireşan, and Jürgen Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. In *International Conference on Artificial Neural Networks*, pages 52–59. Springer, 2011.

[Medel and Savakis, 2016] J. R. Medel and A. Savakis. Anomaly Detection in Video Using Predictive Convolutional Long Short-Term Memory Networks. *ArXiv e-prints*, December 2016.

[Mercuri *et al.*, 2016] Marco Mercuri, Carmine Garripoli, Peter Karsmakers, Ping Jack Soh, Guy AE Vandenbosch, Calogero Pace, Paul Leroux, and Dominique Schreurs. Healthcare system for non-invasive fall detection in indoor environment. In *Applications in Electronics Pervading Industry, Environment and Society*, pages 145–152. Springer, 2016.

[Rubenstein *et al.*, 1990] Laurence Z Rubenstein, Alan S Robbins, Karen R Josephson, Barbara L Schulman, and Dan Osterweil. The value of assessing falls in an elderly population: a randomized clinical trial. *Annals of internal medicine*, 113(4):308–316, 1990.

[Sabokrou *et al.*, 2016] M Sabokrou, M Fathy, and M Hoseini. Video anomaly detection and localisation based on the sparsity and reconstruction error of auto-encoder. *Electronics Letters*, 52(13):1122–1124, 2016.

[Skubic *et al.*, 2016] Marjorie Skubic, Bradford H Harris, Erik Stone, KC Ho, Bo-Yu Su, and Marilyn Rantz. Testing non-wearable fall detection methods in the homes of older adults. In *Engineering in Medicine and Biology Society (EMBC), 2016 IEEE 38th Annual International Conference of the*, pages 557–560. IEEE, 2016.

[Vadivelu *et al.*, 2016] Somasundaram Vadivelu, Sudakshin Ganesan, OV Ramana Murthy, and Abhinav Dhall. Thermal imaging based elderly fall detection. In *ACCV Workshop*, pages 541–553. Springer, 2016.

[Yusif *et al.*, 2016] Salifu Yusif, Jeffrey Soar, and Abdul Hafeez-Baig. Older people, assistive technologies, and the barriers to adoption: A systematic review. *International journal of medical informatics*, 94:112–116, 2016.