Vision-Based Approach to Senior Healthcare: Depth-Based Activity Recognition with Convolutional Neural Networks

Alisha Rege[1] Sanyam Mehra[2] Alyssa Vann[1] Zelun Luo[1]
[1]Department of Computer Science [2]Department of Electrical Engineering
Stanford University, CA 94305, USA
{amr6114,sanyam,avann,zelunluo}@stanford.edu

Abstract

Recent progress in developing cost-effective sensors and machine learning techniques have enabled new AI-assisted solutions for human behavior understanding. In this work, we aim to investigate the use of depth sensors for the detection of daily activities, lifestyle patterns, and vital signs, as well as the development of intelligent mechanisms for accurate situational assessment and rapid response. Using the dataset we collected at On Lok, a senior home in San Francisco, we propose to build and demonstrate an integrated solution for remote monitoring, assessment, and support of seniors living independently at home using computer vision techniques such as 3D convolutional neural networks and LSTMs. We also introduce a new database and annotation framework consisting of labeled activities for senior citizens.

1. Introduction

As the first baby-boomers begin to reach retirement age, the U.S. has begun to experience a shift in the age demographics of its population that has significant implications for Medicare spending and the federal budget. By 2050, the Congressional Budget office is projecting Medicare costs to rise from 3% to 5.5% of GDP [3]. This increase in spending prompts a need for a better solution to senior care. Currently, many senior citizens stay in senior homes. Although these places provide a safe haven for the seniors, in Asian cultures, it is mostly done against the wishes of the senior citizens. Some technological advances have been made in an attempt to allow seniors to live at home such as the Philips Lifeline With AutoAlert [1]; this is a personal help button that is worn around the neck or wrist and can detect if you've fallen. However, these technologies typically rely on the senior remembering to wear a device or the senior being able to reach for help.

Computer vision holds significant potential for automation and augmentation in the monitoring and recognition of health and healthcare behaviors for aging populations [4]. Recently, the field of action recognition has become a hot-topic in the world of computer vision. A successful, fully integrated action recognition model would allow senior citizens to enjoy a longer period of independent living in their homes, without the need for a full-time nurse. Such a system would classify the actions of the senior, and alert an attending nurse and/or other responsible parties, in the event of a problem or anomaly. Thus far, solutions using RGB data have been successful at classifying activities, however, these models do not conform to HIPAA (Health Insurance Portability and Accountability Act) requirements. Since depth sensors allow for personal de-identification, they are more suited for the task of remote monitoring of conditions of seniors. In addition to de-identifying the people being

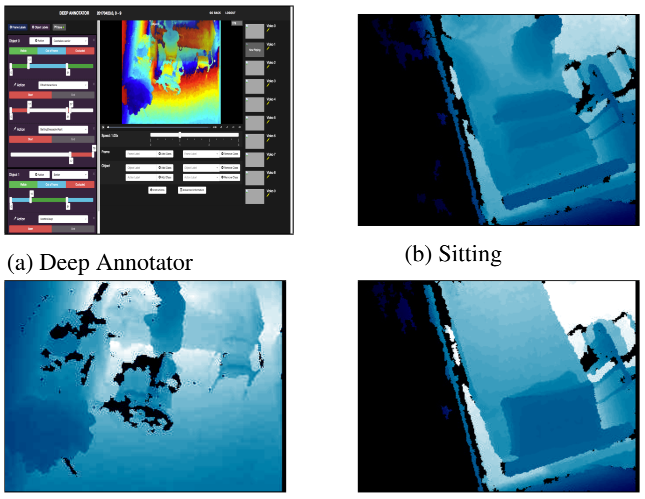

(a) Deep Annotator (b) Sitting
(c) Standing (d) Background

Figure 1: (a) Deep Annotator - labeling software we created. (b-d) These are examples of depth images in our dataset.

recorded, depth sensors also operate in a wider range of lighting conditions than RGB video cameras. Due to these reasons, we use data captured by depth sensors in the On Lok Senior Home in San Francisco to attempt to classify the actions of seniors. This is the first step toward creating a fully integrated action recognition system.

We propose two models for classifying actions of senior citizens: The first model is a 3D Convolutional Network and the second is a Long Term Recurrent Convolutional Network (LRCN). Both models take clips of 7 frames as input and output the activity label (action).

In this paper we walk through the steps we have taken thus far to attempt to classify the activities of seniors using depth information. These steps include data collection and annotation in section 3, building a 3D Convolutional Neural Network model and LRCN to classify activities in sections 4 and 5, experiments done to optimize the performance of these models in section 6 and conclusion and further work in section 8.

2. Related Work

2.1. Non-Neural-Network Approaches to Activity Recognition

Wong et al. [20] use thermal data to detect faints. They implement dynamic action recognition by checking if the height or width of the person has decreased. Though the model receives high accuracy scores on their data, it cannot distinguish between humans and animals, does not know how to handle scenes with multiple people, and would likely struggle to distinguish between faints and other activities where the heights and widths of people change.

Several researchers have used Hidden Markov Models (HMMs) to classify activities or actions in videos. Liouane et al. [11] proposed using a Hierarchical Hidden Markov Model to model simple to complex actions detected by different types of sensors. They propose a new grammar for activity detection, which they call the 'Home By Room Activities Language'. While their approach is novel, the model tends to be overly sensitive and relies on the development of a grammar to understand activities.

In the papers [7] and [6] Jalal et al. extract features from depth videos, including time sequence features that are meant to be translation and scaling invariant and magnitude and directional angular features from joint points of skeleton models. These features serve as the input to HMMs, which classify activities. HMMs, however, are not ideal for the complicated task of activity recognition because they are expensive in terms of memory and computing time, are trained by a set number of seed sequences, and rely on the features one extracts from input data.

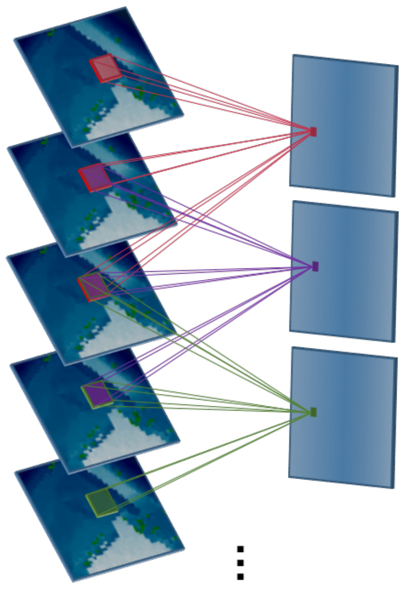


Figure 2: Example of a 3D convolution on higher dimensional inputs.

2.2. 3D Convolutional Networks for Activity Recognition

Several researchers have made use of convolutional neural networks (CNNs) to recognize activities. Karpathy et al. [10] proposes a model to classify 487 activities in 1 million YouTube videos by extending the connectivity of CNNs to the time domain (i.e. performing 3D convolutions; we show an example of a 3D convolution in figure 2). This allows for the capturing of spatial as well as temporal features through convolutions. Karpathy et al. found that classifying single frames versus multiple frames did not do much to improve their accuracies. This indicates that perhaps 3D convolutional models alone are not the best suited for capturing temporal information.

Using alternate inputs to their 3D convolutional network for action recognition, Simonyan and Zisserman use optical flow inputs to their convolutional network as a way of describing the motion in RGB video frames [17]. They used two streams, one for spatial information and the other for temporal information. They found that their network performed better on optical flow inputs than raw video inputs. While their input captured motion trajectories, their pooling layers did not account for these trajectories, and their model might have performed better with these. Tran et al. build on the work of Karpathy et al. and Simonyan and Zisserman, proposing a state of the art 3D convolutional network which learns spatial and temporal features to classify the actions in RGB videos [18]. We follow the model proposed by Tran et. al.

Like Simonyan and Zisserman, Jayabalan et al. do not directly classify actions in raw videos. Rather, they use joint data from peoples' motions for classification [8]. While using joint data allows for lower dimensionality, making the model simpler and faster, not all actions can be simplified into skeletal representation, making the model not applicable beyond a set number of activities.

Ji et al. also use 3D convolutions for action recognition to extract spatial and temporal features on videos from an uncontrolled environment [9]. From their input, they generate multiple channels of features that are processed in parallel and then concatenated to create a final feature representation for classification. The multiple channels in their model require a large amount of labeled data.

Unlike several of the approaches mentioned above that rely solely on RGB videos as input, Ni et al. offer a dataset with both RGB and depth information, and present a model with features that fuse information from these two modalities [12]. However, because of HIPAA regulations, requiring the de-identification of people in the recordings, we could not use this dataset or their proposed features because it would ultimately not be implementable in a real-world setting.

In contrast to the previously mentioned papers, Oreifej et al. focus exclusively on activity recognition in depth sequences [14]. They capture the distribution of surface normals in depth sequences in 4 dimensions (temporal, spatial, and depth dimensions), and use these to better classify the activities in sequences. Their use of surface normals better captures motion, as well as geometry, to show joint shapes in motion. While they describe a powerful feature for better understanding activities, they do not show how the architecture could also be leveraged to capture temporal information.

Rahmani et al. describe a different method of improving the detection of activities in depth sequences [16]. They transform their data into 3D pointclouds, making their model more robust to noise, and action speed and viewpoint variations. Similar to Oreifej et al., they describe a powerful feature for improving the robustness of the model that could be further improved by incorporating long-term information into the architecture.

2.3. Recurrent Convolutional Neural Networks

Several researchers attempt to better model temporal features for activity recognition. One attempt to capture such dynamics across an entire video was performed by Wang et al. [19] They created a temporal segment network (TSN), which sparsely sampled segments across a video to generate an ultimate prediction. Unlike other researchers, they did not utilize recurrent units. Both Ordonez et al. and Donahue et al. have combined convolutional and recurrent layers to produce state of the art models for activity recognition. Ordonez et al. show how combining convolutional and LSTM recurrent layers allows for better performance on recognizing activities from wearable sensors [13]. They describe LSTM cells as necessary for distinguishing similar actions, and for capturing various temporal scales. Donahue et al. similarly combine the benefits of convolutional and recurrent LSTM layers [2]. They describe a complex architecture called LRCN (Long Term Recurrent Convolutional Network). The algorithm consists of extracting spatial features on each frame (using RESNET [5]) which are then fed into a LSTM to capture temporal dynamics. Thus, the network is doubly deep, having both spatial and temporal layers, that can be optimized with backpropagation. This network is especially appealing because it does not require data to be preprocessed or for features to be hand-designed. We attempt to reproduce this work in one of our models. Similar to most other action recognition works, this model relies on RGB data. We propose and analyze performance of this model on data from depth sensors instead.

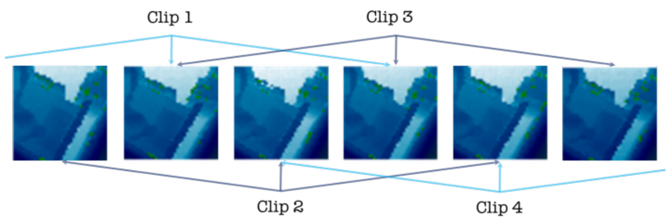


Figure 3: Example of Clip Formation

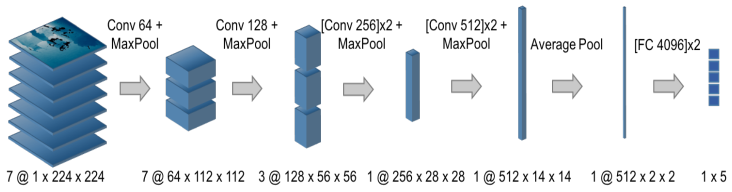
3. Dataset and Annotation

We introduce a new dataset with depth videos captured through sensors installed in On Lok Senior Home in San Francisco. We developed an annotation tool to label data called Deep Annotator[1]. Using this tool, we were able to hand annotate 7 hours of data[2] collected by 5 sensors. From these annotations, we found the distributions of actions in our data. Although we labeled around 28 actions, many of these actions are less represented in the dataset. For example, actions such as sitting down and getting up are important to the labeling process as these actions will be the most probable timing for falls; however, in the 7 hours of data annotated, we only found 3-5 instances of this action. For this experiment, we have reduced the number of classes to 5[3]. We plan to annotate more data for the other actions in order to include them in training and testing. The data used in the experiment is displayed in table 1. Figure 3 shows

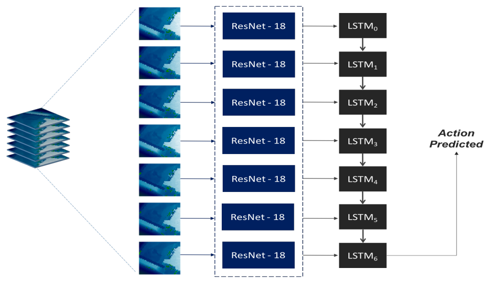
[1]Accessible to members of the lab at http://aicare.stanford.edu:8888/

[2]Across three day span

[3]These classes have at least 500 clips and are represented on all days of recordings



(a) 3D Convolutional Network Architecture



(b) LRCN Architecture

Figure 4: Architectures used

pictorially how the clips were created from instances. After each frame was annotated, clips of 7 frames each were created using alternating frames (to capture more of an activity) with majority vote labeling across frames to produce the label for each clip. Our input clips have dimensions of 7 x 240 x 320 x 1 (because depth data is captured in only one dimension). These clips are cropped to create 7 x 224 x 224 x 1 clips in the algorithm. Figure 1a shows the annotating tool. Figures 1b, 1c, 1d show examples of our dataset. We used these clips to create the training and testing sets. The split for test and train was done by day and the respective frequencies are shown in the table 1. Two days were used for training and one day for evaluating.

Table 1: Number of instances per activity class

Activity Class	**Class**	**Train**	**Dev**
AdjustingBedHeight	0	696	123
Cleaningw/Assist	1	5620	1239
GettingDressedw/Assist	2	2673	954
Rest/Sleep	3	2351	1622
Sitting	4	10231	478

4. 3D Convolutional Network

This network is altered from the work proposed by Du Tran et al. in 'Learning Spatiotemporal Features with 3D Convolutional Networks' [18]. We implemented two different 3D convolutional neural network architectures. By using 3D convolutions, our network incorporates both spatial and temporal information. Figure 2 displays how our network performs these 3D convolutions. We use a 3 x 3 x 3 kernel. With this kernel, each output of a convolutional layer corresponds to three temporal slices of input.

Our first architecture is depicted in Figure 4a. This network takes in depth video clips, and uses convolutional layers to extract features. Our network consists of the following sequence of layers: [Conv3d $\rightarrow$ MaxPool3d] x 2 $\rightarrow$[Conv3d $\rightarrow$ Conv3d $\rightarrow$ MaxPool3d] x 2 $\rightarrow$ Average Pool $\rightarrow$ [Fully Connected] x 2. The convolutional layers use 64, 128, 256, 512 filters respectively. Each convolutional layer uses a zero-padding of 1 and a stride of 1. At each convolutional, layer we also perform Batch Normalization and apply a ReLU non-linearity. Our max pooling layers alternates between kernel of sizes 2 x 2 x 2 and 1 x 2 x 2.

We then used Cross Entropy Loss 1 and Adam Optimizer to learn and optimize the function.

$$\mathcal{L}(X,Y) = -\frac{1}{n}\sum_{i=1}^{n} y^{(i)} \ln a(x^{(i)}) + \left(1 - y^{(i)}\right) \ln\left(1 - a(x^{(i)})\right) \tag{1}$$

Here, X is the set of input examples in the training dataset, and Y is the corresponding set of labels for those input examples. The a(x) represents the output of the neural network given input X.

5. LRCN

This network was created by altering the work proposed by [2]. We used a RESNET-18 with pretrained weights (from ImageNet) to extract features from the individual frames, then these frames were fed into a LSTM that correlated the features to the time dimension. Since our dataset is very different from Imagenet, in order to let the CNN learn these different features, we backpropagate the gradients through the pretrained RESNET-18 with a learning rate equal to 1% of the learning rate used for other layers in the model. 4b is the depiction of the model.Firstly, the 7 images go into a Resnet-18 with the last fully connected layer not included. The representations of the images output by the CNN are then fused into one clip. This clip is then fed as input (each frame is one time step) into a multi-layer LSTM. The last time step of the LSTM then outputs a label. The entire network is updated using Adam Optimizer. The equations used to create the LSTM are detailed by equations 2-6.

$$i_t = \sigma(W_i h_{t-1} + U_i x_t + b_i) \tag{2}$$
$$f_t = \sigma(W_f h_{t-1} + U_f x_t + b_f) \tag{3}$$
$$\tilde{c_t} = \phi(W h_{t-1} + U x_t + b) \tag{4}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c_t} \tag{5}$$
$$o_t = \sigma(W_o h_{t-1} + U_o x_t + b_o) \tag{6}$$
$$h_t = o_t \odot \phi(c_t) \tag{7}$$

Here equation 2 calculates the value for the input gate i_t by multiplying a weight matrix W_i with output h_{t-1} from the previous cell, plus another weight matrix U_i times the input at that time step x_t, plus the bias b_i. Similarly, equations 3-6 calculate the value for the forget gate f_t, new memory cell $\tilde{c_t}$, final memory cell c_t , and Output/Exposure gate c_t respectively.

6. Experiments

The results are displayed in table 2. Multiple experiments were done on the dataset from altering the number of LSTM layers to the learning rates. LRCN is faster in training than 3D Convolution, and has more parameters, more experiments were done with LRCN.

6.1. Number of classes

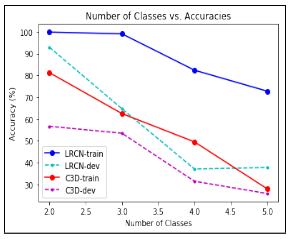
A number of experiments were conducted on the class distributions to test the model. The distributions were as follows:
2 classes - Sitting and Rest/Sleep
3 classes - Sitting, Rest/Sleep, Cleaningw/Assist
4 classes - Sitting, Rest/Sleep, Cleaningw/Assist, Adjusting Bed Height
5 classes - AdjustingBedHeight, Cleaningw/Assist, GettingDressedw/Assist, Rest/Sleep, Sitting

We also ran experiments in which we varied the number of classes. Classes were added in based on the number of instances of the class and how dissimilar they were to other classes being tested.

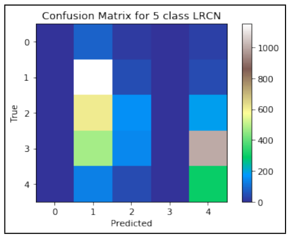
Figure 5a displays how the number of classes affects the accuracies. As we increase the number of classes, our accuracy decreases for training and dev sets. This may occur as a result of the complexity of the problem. As we add more classes, the differences between classes such as Resting/Sleeping and added classes such as Adjusting Bed Height starts to get more finegrained. As a result, requiring more and more data for the model to learn how to distinguish between such classes.

6.2. Pretrained

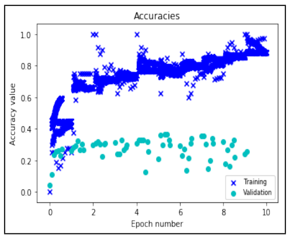
For the LRCN, we attempted to increase the accuracy by using pretrained weights. The pretrained weight for Resnet-18 did not affect the results very much. This may be because the weights were calculated for RGB images, and the pretrained weights were off of the mark by an equal amount. This is observable when comparing the 5 class results for LRCN.



(a) Effect of class size on Accuracy



(b) Confusion Matrix



(c) Accuracy of five class LRCN

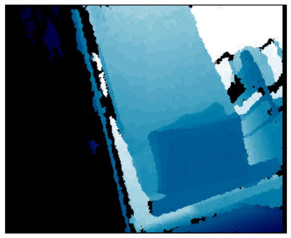

(d) Faulty Sensor angle leading to data loss

Figure 5: Experimental Results

6.3. Learning Rate

Each experiment we conducted was fine tuned to have the optimal learning rate parameter. We found that increasing the learning rate caused the 3D convolutional model to have very little changes; whereas, the LRCN model was heavily affected by changes in learning rate. This may be because the LRCN was loaded with pretrained weights; whereas the 3D convolution needed to be trained from scratch. This meant that even though the learning rate was being adjusted by the Adam optimizer, the 3D convolutional model could not fine tune the parameters enough. However, in the case of the pretrained weights, since they were pretrained on RGB data, we need a large learning rate to boost it out of the local minima. This is observable in the results of experiments done for 3 classes in LRCN.

6.4. Number of Layers

Varying the number of layers for the LSTM affected the results significantly. Increasing the number of layers from 2 to 3 increased accuracy in the 3, 4, and 5 class experiments. However, we can see that increasing the number of layers hits a peak at 3 from the experiments on 5 classes. The

Table 2: Results

Model	Pretrained	lr	Hidden Size	Num Layers	Num Classes	Train Acc.	Dev Acc.
3DConv	N	0.001			2	81.31	**56.69**
3DConv	N	0.0001			2	80.23	**56.60**
3DConv	N	0.00001			2	80.54	**56.63**
3DConv	N	0.0001			3	62.51	**53.5**
3DConv	N	0.0001			4	49.53	**31.46**
3DConv	N	0.0001			5	28.03	**25.86**
LRCN	N	0.001	500	2	5	80.32	10.82
LRCN	N	0.0001	1000	2	5	79.8	29
LRCN	N	0.001	**2000**	2	5	88.21	**37.64**
LRCN	Y	0.0001	2000	2	5	68.06	28.92
LRCN	**Y**	0.0001	2000	**3**	5	72.81	**37.77**
LRCN	Y	0.0001	2000	4	5	58.57	21.26
LRCN	**Y**	0.0001	2000	3	5*	80.85	31.46
LRCN	Y	0.0001	2000	2	4	67.06	20.57
LRCN	Y	0.0001	2000	**3**	4	82.41	**37.09**
LRCN	Y	0.0001	2000	2	3	72	37
LRCN	Y	0.001	2000	2	3	65	58.5
LRCN	Y	**0.0005**	2000	**3**	3	99.09	**64.76**
LRCN	Y	0.0001	2000	2	2	99.89	**92.95**

*Class balancing on train and dev

accuracy drops by approximately 6% when we increase the number of layers to 4.

6.5. Hidden Size

Increasing the hidden size for LRCN also increased the accuracy of the model. This can be attributed to the complex nature of the data. The increase in number of neurons allows for more complexity and more connections to be made. This is observable in the experiments with 5 classes in LRCN.

6.6. Class Balancing

Although class balancing is meant to help models learn representations better, in this case, class balancing hampered the results. We believe this to be true because we don't have enough data for the complexity of the task. The class balancing is now removing relevant examples from train.

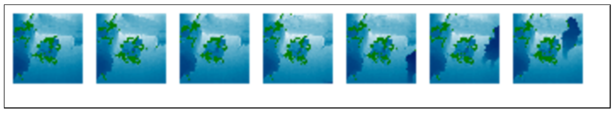
7. Analysis

7.1. Issues with Cameras

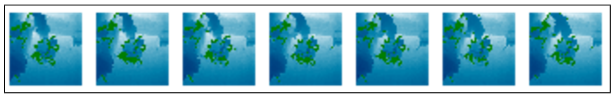
Several issues with the collected data played a role in the classification errors being reported. At the time we were able to collect data, 2 of the 7 sensors currently at On Lok were not functional, limiting the size of our dataset. 3 of the working 5 sensors were not optimally placed, as in field of view did not capture the region of interest properly. This more often than not led to missing out capturing essential activity information. For example, figure **??** illustrates and example where the sensor was placed at an angle that prevented it from seeing most of the senior's room, and instead mostly output images of the wall, which captures very little about the activity going on in the captured frames. Though we attempted to collect more data for our models, we were unable to because of administrative issues, including the need to renew IRB (Institutional Review Board) approval. We believe our models' performance will improve with more data that better captures and represents activities.

7.2. Confusion Matrix

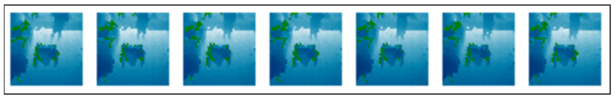
The confusion matrix as a result of the 5 class LRCN is shown in figure 5b. Here we can see that 0 or 3 (Adjusting Bed Height or Rest/Sleep) are very rarely predicted. These two classes were the least represented in the training set, and they were commonly mis-classified as Cleaning with Assist. This classification occurs because of two major problems: the first is described in the previous section. The orientation of one of the sensors was such that we could not distinguish between these three classes and since Cleaning with Assist had more examples in the dataset, it was deemed most fit for these examples. The second issue was that Cleaning with Assist included a label we had originally deemed to be changing adult diaper. This was usually performed on the bed and during this process, the nurse would walk away for a few seconds to retrieve oint-



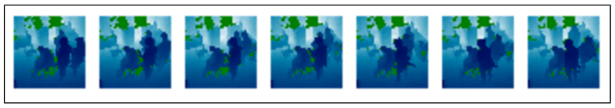
(a) Rest no sleep classified as sitting



(b) Sitting classified as clean with assist



(c) Sitting correctly classified



(d) Clean with assist correctly classified

Figure 6: Examples from Results

ments or the other diaper. Our relatively low clip length of 7 frames (approximately 1 second) caused some clips in Cleaning with Assist to become very similar to Rest/Sleep or Adjusting Bed Height. We plan to combat this issue by using the updated sensors, once we get approval, and temporally down-sampling the incoming stream even further, resulting in capturing a longer sequence of events.

7.3. Train/Validation Curves

Figure 5c shows that the LRCN network trains correctly. The training accuracies continue to increase as the validation curve stays constant after approximately 2 epochs. After this point, we face over-fitting as train accuracy continues to increase while the gap between train and validation grows.

7.4. Qualitative Analysis

Though one can see that the senior was lying in bed in figure 6a, this clip was misclassified as Sitting. This points to the fact that our model learned to not only look at the people in the room, but at the background, including the wheelchair in the clip. Likely because the wheelchair is present, this clip was classified as Sitting. Similarly, in figure 6c a wheelchair is present, and this image was classified as Sitting, though in this case this was in fact the correct label. We plan to improve the model's accuracy by incorporating background removal, in order to allow the model to learn the most salient features for every activity.

In figure 6b, we see an example of a Sitting clip that was misclassified as Cleaning with Assist. This is likely because the model learned that Cleaning with Assist involves both a senior and caretaker, so upon observing two figures in the scene, labeled this clip as Cleaning with Assist. Potentially based on a similar assumption, that Cleaning with Assist involved both a senior and caretaker, the model was able to correctly classify Figure 6d as Cleaning with Assist.

From these examples, we can see that our models rely on context to deduce the activity taking place. Such context includes everything from how many people are in the scene and whether a wheelchair or bed frame is present.

8. Conclusion

As a step toward enabling seniors to live for longer periods at home independently, we collected data from depth sensors at the On Lok home in San Francisco. We labeled the activities taking place in this data using the annotation tool we developed, called DeepAnnotator. To automatically recognize the activities in this data, we built two convolutional neural network models. Our first model is a 3D Convolutional Network, and the second is a Long-Term Recurrent Convolutional Network. Both show improved performance with fewer classes. We believe this is in large part because of scarcity of data for some classes in our dataset and issues with data collection, causing the data from some sensors to not convey enough information.

In the future we plan to focus on acquiring and labeling more data. We also plan to expand from depth into other modalities that also conform to HIPAA requirements, including thermal data collected by [15]. These cameras will be installed at the On Lok homes in later May 2017. We would like better account for the motion in our clips by using optical flow, and increase the robustness of our models to viewpoint variance by using pointcloud representations.

9. Acknowledgments

The DeepAnnotator was previously created by another member of our lab in Summer 2016. We have rebooted the website and redone many of it's features. We have worked with the Vision Lab in getting approval, funding, and resources for this project. Thanks to Zelun Luo for providing direction, Guido Puisol for setting up the architecture to gather data, and CERC team for providing funding and financial support.

References

[1] S. Abrahms. Technology, gadgets for seniors aging in place, Mar 2014. 1

[2] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual

recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015. 3, 4

[3] D. W. Elmendorf. The 2014 long-term budget outlook. DTIC Document, 2014. 1

[4] V. Gulshan, L. Peng, M. Coram, M. C. Stumpe, D. Wu, A. Narayanaswamy, S. Venugopalan, K. Widner, T. Madams, J. Cuadros, et al. Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. *JAMA*, 316(22):2402–2410, 2016. 1

[5] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 3

[6] A. Jalal, S. Kamal, and D. Kim. A depth video sensor-based life-logging human activity recognition system for elderly care in smart indoor environments. *Sensors*, 14(7):11735–11759, 2014. 2

[7] A. Jalal, M. Z. Uddin, and T.-S. Kim. Depth video-based human activity recognition system using translation and scaling invariant features for life logging at smart home. *IEEE Transactions on Consumer Electronics*, 58(3), 2012. 2

[8] A. Jayabalan, H. Karunakaran, S. Murlidharan, and T. Shizume. Dynamic action recognition: A convolutional neural network model for temporally organized joint location data. *arXiv preprint arXiv:1612.06703*, 2016. 3

[9] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2013. 3

[10] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014. 2

[11] Z. Liouane, T. Lemlouma, P. Roose, F. Weis, and M. Hassani. A markovian-based approach for daily living activities recognition. *arXiv preprint arXiv:1603.03251*, 2016. 2

[12] B. Ni, G. Wang, and P. Moulin. Rgbd-hudaact: A color-depth video database for human daily activity recognition. In *Consumer Depth Cameras for Computer Vision*, pages 193–208. Springer, 2013. 3

[13] F. J. Ordóñez and D. Roggen. Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors*, 16(1):115, 2016. 3

[14] O. Oreifej and Z. Liu. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 716–723, 2013. 3

[15] G. Pusiol. Thermset data. https://github.com/blusa/research/raw/master/Thermset 7

[16] H. Rahmani, A. Mahmood, D. Q. Huynh, and A. Mian. Hopc: Histogram of oriented principal components of 3d pointclouds for action recognition. In *European Conference on Computer Vision*, pages 742–757. Springer, 2014. 3

[17] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014. 2

[18] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4489–4497, 2015. 2, 4

[19] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. Temporal segment networks: towards good practices for deep action recognition. In *European Conference on Computer Vision*, pages 20–36. Springer, 2016. 3

[20] W. K. Wong, H. L. Lim, C. K. Loo, and W. S. Lim. Home alone faint detection surveillance system using thermal camera. In *Computer Research and Development, 2010 Second International Conference on*, pages 747–751. IEEE, 2010. 2