Depth-Based Activity Recognition in ICUs Using Convolutional and Recurrent Neural Networks

Rishab Mehra* Gabriel M. Bianconi* Serena Yeung† Li Fei-Fei†
Department of Computer Science, Stanford University
{rishab,bianconi,serena,feifeili}@cs.stanford.edu

Abstract

Currently, activity detection in Intensive Care Units (ICUs) is performed manually by trained personnel - primarily nurses - who log the activities as they happen. This process is both expensive and time consuming. Our goal is to design a system which automatically gives an annotated list of all activities that occurred in the ICU over the day. In the future, we also aim at providing a system which informs the doctor about the health status of patients and unusual activity occurrences. Overall, this system will reduce the monitoring workload of trained personnel, and lead to a quicker and safer recovery of the patient. In order to design this system, we installed depth sensors in ICUs to create a novel dataset, and performed activity recognition on it using Convolution Neural Networks and Long Short Term Memory Units. We were able to achieve high classification accuracy for a restricted dataset, particularly in networks that leveraged temporal information and multiple viewpoints.

1. Introduction

Activity recognition in hospitals is a task that has not received much attention in the past. Some of the main reasons for this gap in research are the lack of sensors installed in hospitals and the difficulty in obtaining access to the relevant data due to its sensitive nature. Thanks to our collaboration with Intermountain Healthcare, a large healthcare provider in the United States, we have access to depth sensors installed in eight intensive care unit (ICU) rooms at the LDS Hospital (Salt Lake City, Utah).

We leveraged this opportunity to create a novel dataset consisting of annotated activities happening in the ICUs. Our goal is to eventually create an automated system that is able to automatically log the activities happening in the ICU in order to alleviate the monitoring workload of nurses and other trained personnel.

Due to privacy concerns in the hospitals given the sensitive nature of the data, we do not have access to RGB cameras; instead, we are limited to depth sensors only. Most of the past research in activity recognition has focused on RGB or RGB+D datasets. Our work has focused on adapting many of these techniques to a depth-only dataset.

We built an end-to-end data pipeline that collects sensor data, streamlines labeling, and enables testing a variety of model architectures in different experiments. We created an iOS app to help nurses give approximate annotations of the data, which we used to later create precise annotations. We then created scripts to pre-process the data and generate the dataset given these annotations. Finally, we experimented with a wide variety of models, including single-frame models using CNNs [10] (e.g., variations of a ResNet-18 [7]) and temporal models using LSTM units [8] and 3D-CNNs [15].

In Section 2, we review the relevant literature for our project. In Section 3, we describe the process we used for collecting and labelling data, and how we generated the final datasets. In Section 4, we review the methodology for our experiments and the network architectures we used. In Section 5, we present the experiments we conducted and analyze the results. In Section 6, we discuss the limitations and achievements of the current state of our project and finally, in Section 7, we propose next steps for further consideration.

2. Related Work

Many traditional machine learning methods have been studied in the context of video activity detection to varying levels of success. For example, researchers at the University of Oxford used optical flow data to compute the likelihood of an action [13]. Other approaches have included using probabilistic graph models for action recognition. In a paper, researchers used methods such as Hidden Markov Models (HMMs) to estimate human body joint points [11].

More recently, there has been a strong interest in using

*Both authors contributed equally to this work.
†These authors are not enrolled in CS231N.

deep learning methods for this task. One of the approaches studied how to use the temporal information available in these tasks for classification; the authors propose a Long-term Recurrent Convolutional Network architecture, which was used for activity recognition with RGB data [5].

An alternate approach for capturing temporal information is using 3D-CNNs. These are similar to 2D-CNNs but additionally convolve across depth. Thus, information from multiple frames can be combined. Researchers at Arizona State University used this approach to perform activity recognition on RGB data [9]. Further, VoxNet [4] used 3D-CNNs for real time object detection, proving that 3D-CNNs can be used for doing recognition tasks efficiently.

Further, we studied two recent datasets for activity recognition, and techniques that have been used to perform activity recognition on them.

Activity Net [2][3] is a taxonomically divided dataset of videos for labelled for activity detection. It is one of the largest video datasets with 849 hours of videos. The researchers performed several experiments on the dataset and found that a combination of motion features (`HoG`, `HoF`, `MBH`) and deep features (`fc6`, `fc7`, `fc8`) gave the best validation accuracies. Individually, motion features outperformed the deep features.

Youtube8M [1] is a dataset by Google which contains 8 million annotated videos from Youtube. In their experiments, the researchers found that LSTMs generally give the best results for activity detection in videos. Further, researchers found that using the model they had trained on Youtube8M, the transfer learning results on Activity Net outperformed the model defined in the Activity Net paper by around 20%.

Similar to our work, researchers have analyzed a dataset of depth data to monitor hand hygiene in hospitals [16]. The authors used a CNN on single-frame data to achieve high classification accuracy on recognizing this action. Our work extends this paper by introducing multiple sensor viewpoints, temporal data, and a larger set of actions to be monitored.

3. Dataset

We are creating a novel dataset of depth sensor data to be used for activity recognition in ICUs. We recorded depth data in eight different ICU rooms with four sensors each. For this dataset, we use three of those viewpoints. Refer to Figure 1 for a floor map of the ICU used for the data collection.

3.1. Data Collection

The depth sensors record data whenever there is a substantial change in the scene. Whenever there is an action occurring, the sensors generally record 2-5 frames per second. In each room, we have four sensors: a frontal view of the patient's bed, two facing the bed at different angles, and one facing the sink. For this iteration of the project, we disregarded the sink-facing sensors since they did not play a meaningful role in the activities under consideration.

Currently, some of our annotations are problematic. A common problem is that certain viewpoints can be partially or fully occluded at times. Another common problem is that the data from a certain viewpoint was missing while the other ones were present (for such cases, when using a combined dataset, we used empty images). Finally, the sensors occasionally skip recordings for a few minutes (e.g., sensors sometimes restart), creating gaps in the recordings. These and other issues can debilitate the model and reduce its classification accuracy.

3.2. Data Labelling

We are labeling this data with help from nurses at the hospital. The actions we annotated were: oral care; patient getting out of bed; patient getting in bed; turning patient in bed; patient getting into chair; patient getting out of bed; ultrasound; and x-ray.

We created an iOS app to streamline the labeling process (Figure 3). The app allows nurses to quickly annotate the approximate time of an activity in a specific room; we use this timestamp as a guideline when later carefully labelling the dataset.

With the depth sensor data and the approximate timestamp at hand, we proceeded to build our dataset. First, we mapped all recorded image frames from the depth sensors to their corresponding timestamps. We then rotated images as necessary, as some sensors were installed at different angles. Next, for each nurse annotation, we manually annotated the exact start and end timestamps of the given action. These are used as boundaries when extracting the frames for the action instance. We then collect unused frames as background (no activity) data for our classifier; for each positive action, we generate a negative action with the same amount of frames.

Due to a combination of sensor issues and limited time for nurse annotations, we were not able to obtain a sufficiently large number of annotations for each of the actions. Therefore, we restricted our dataset to the following three classes with at least 10 instances each: getting out of bed; oral care; and background. The final dataset had a total of 35 instances excluding background clips.

3.3. Independent Viewpoints

For part of the experiments, we treated each viewpoint as an independent instance of the action class. Our goal was to train a network that generalizes its classification capabilities to a variety of viewpoints, and does not depend on a particular stream of data. Since we have multiple viewpoints per annotation, we were further able to increase the number of

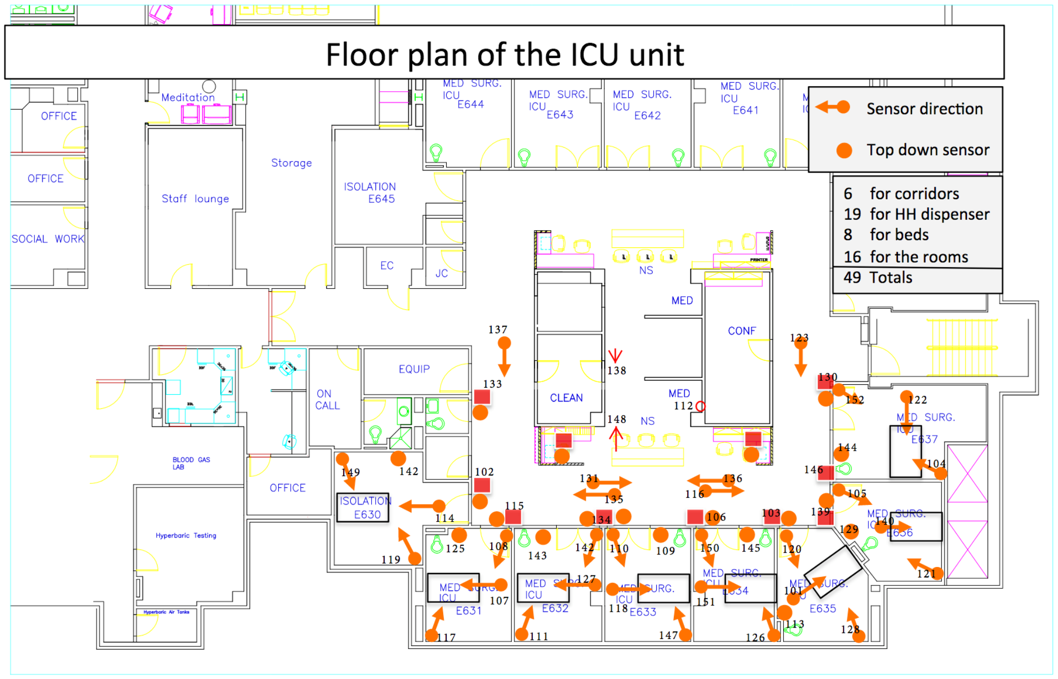

Figure 1: The floor plan of the ICU at the LDS Hospital showing the location of each depth sensor. Orange circles represent depth sensors; the arrows represent the direction they are facing, and the ones without arrows are top-down sensors used for sinks.

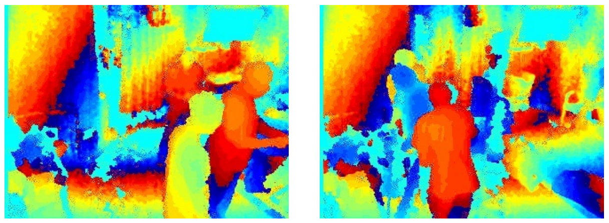

Figure 2: Two frames of depth sensors from different viewpoints in the same room. Oral Care is being performed.

instances in the dataset when analyzing each sensor's data independently.

3.4. Combined Viewpoints

For the remaining experiments, we combined the available viewpoints for a given action instance to create a single datapoint. We observed that the recordings from each sensor in a given room dont happen simultaneously. In order to match the frames for a given room, we choose one camera as the base camera - namely, the camera facing the bed frontally. For each of the frames produced by that camera, we find the closest frame from each of the other three sensors. Note that there might be repeated or skipped frames in these sensors. We group these four frames into the final dataset.

We also noticed that the time difference between frames varies, and therefore the images might not be very synchronized. In practice, most differences are in the 0.2-0.6s range, so we believe the problem should not cause large issues. Some images were missing either due to recording errors or camera problems. In those cases, we simply use an array filled with zeros as input. Finally, we also observed that there is considerable variance in the length of actions. For example, when considering getting out of bed, the action can take from a few seconds to a few minutes depending on the patient. Each activity takes between 5 seconds and 15 minutes.

One of the approaches we experimented with was combining each of the depth recordings depth-wise into a single image, and feeding it into the CNN to extract features.

With this method, the model would be able to use the information in all viewpoints to extract features at a given timestep. However, this approach also created unnatural spatial relationships between the viewpoints.

We also explored other approaches that we did not conclude at this stage of the project. Of these approaches was extracting features of each viewpoint individually and combining them at a later stage. We can merge the information deeper in the network, either before feeding into the LSTM, or by combining the outputs of multiple individual LSTMs. These approaches, however, greatly increase the complexity

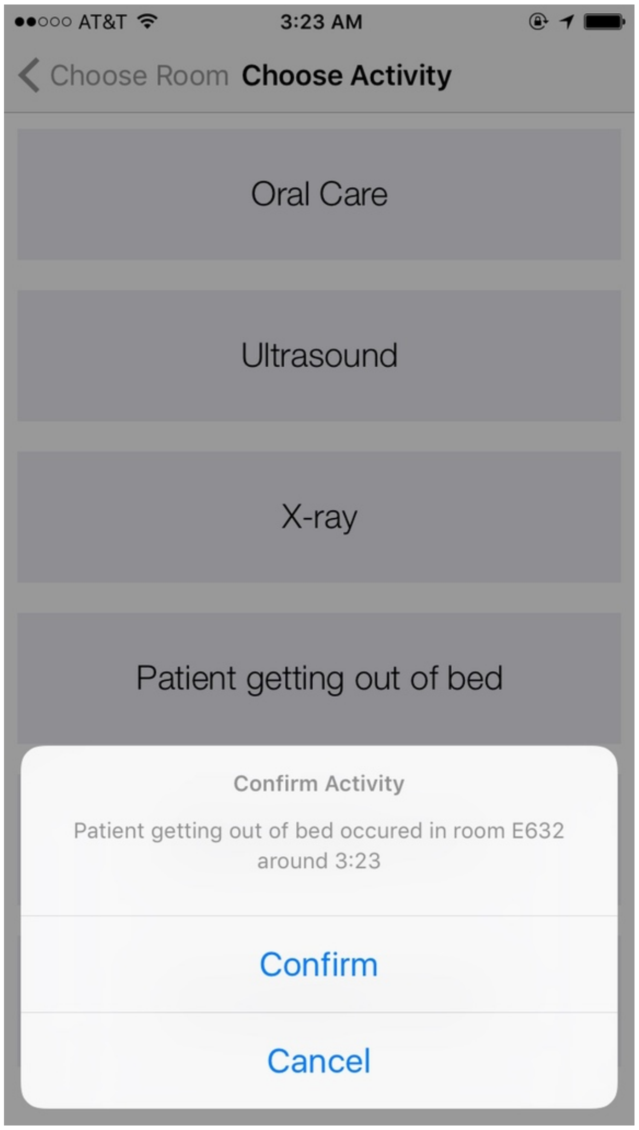

Figure 3: A screenshot of our data collection iOS app. The app allows nurses to quickly record when activities happen in a given room.

and size of the model.

Finally, another approach would be leveraging the sensor data to create a 3D point clouds using techniques such as DynamicFusion [12] and 4DFusion [6]. These point clouds would be more invariant the the location of the input sensors and potentially be able to infer features that cannot be obtained from individual frames. That said, reasoning over 4D inputs (3D point cloud over time) would require considerably more resources, which might not be doable if we eventually want to create a real-time monitoring system for hospitals.

3.5. Data Augmentation & Preprocessing

Since we had a limited number of datapoints, we tried to augment our dataset with a variety of techniques. Our main approach was to sample a certain number of clips from each class. For our final implementation, we sampled 500 64-frame clips from each class to compile our dataset. The samples are random and potentially overlapping.

We then split our dataset into training, validation, testing splits. The last had a single annotation from each class; the remaining annotations were split 80/20 by the other splits.

We also ensured that the clips samples from a single annotation all were selected by the same split, in order to avoid contaminating the test and validation sets.

Finally, we built the test set by finding a longer clip around the annotation, padding the action with 5 minutes of images. Our goal was then to evaluate a sliding window of 64 frames on the network, and be able to discriminate the temporal bounds for the actions, as well as the correct category.

In order to further augment our dataset, we trained the network on random crops and horizontal flips of the images. We also normalized the dataset for a ResNet-18 model trained on ImageNet.

4. Methods

4.1. Convolution Neural Networks

Convolutional Neural Networks (CNNs) are a natural fit for visual recognition tasks such as ours since they are able to learn translation-invariant features over the input space [10]. However, CNNs do not naturally preserve temporal information, unless modifications such as 3D convolutions [15] as used, which are important to understanding video information. Thus, we only use CNN models to analyze single-frame inputs. These can be used as a baseline for classification, or instead used to extract features that are fed into another model which takes temporal information into account. In particular, we used the ResNet-18 model, which is a kind of highway CNN with 18 layers of depth [7].

Previous research has shown that knowledge transfer in CNNs from natural images to medical images made networks more robust and finetuning outperformed training models from scratch [14]. As such, we opted for a ResNet-18 model pre-trained on ImageNet which we finetune on our dataset.

4.2. Long Short Term Memory Units

Since we have depth-based videos, we need a way of capturing the temporal information, for which we will use Long Short Term Memory, which is a special kind of Recurrent Neural Network [8]. These are ideal for our model since they can capture temporal information, i.e. relation-

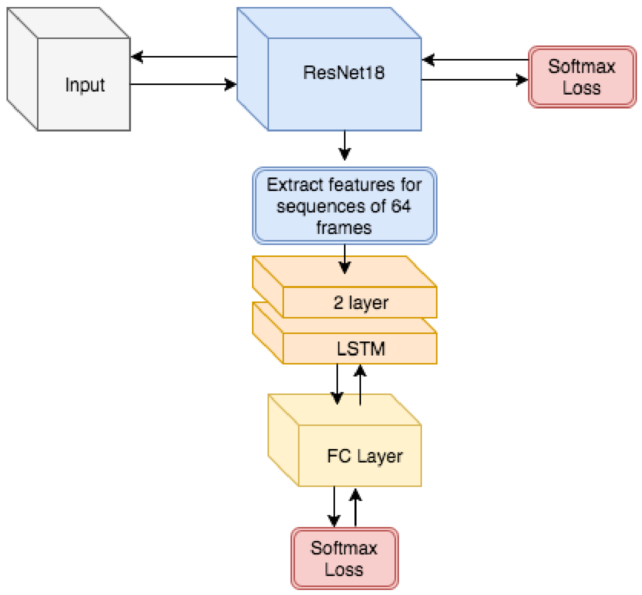

Figure 4: Our model combining a ResNet-18 and LSTM units. The ResNet is trained separately, and later frozen when used in connection with the LSTM model. Forward arrows represent data flow. Backward arrows represent gradient flow.

ship between the frames across inputs. The units in the model are characterized by:

$$
\begin{aligned}
\mathbf{g}^u &= \sigma(\mathbf{W}^u * \mathbf{h}_{t-1} + \mathbf{I}^u *_t) \\
\mathbf{g}^f &= \sigma(\mathbf{W}^f * \mathbf{h}_{t-1} + \mathbf{I}^f *_t) \\
\mathbf{g}^o &= \sigma(\mathbf{W}^o * \mathbf{h}_{t-1} + \mathbf{I}^o *_t) \\
\mathbf{g}^c &= \tanh(\mathbf{W}^c * \mathbf{h}_{t-1} + \mathbf{I}^c *_t) \\
\mathbf{m}_t &= \mathbf{g}^f \odot + \mathbf{g}^u \odot \mathbf{g}^c \\
\mathbf{h}_t &= \tanh(\mathbf{g}^o \odot \mathbf{m}_{t-1})
\end{aligned}
\tag{1}
$$

σ is the sigmoid function, $\mathbf{W}^u, \mathbf{W}^f, \mathbf{W}^o, \mathbf{W}^c$ are the weight matrices of the gates and $\mathbf{I}^u, \mathbf{I}^f, \mathbf{I}^o, \mathbf{I}^c$ are the projected matrices of the inputs into the dimensions of the hidden the state.

Our model for the LSTM units can be seen in Figure 4. We train a ResNet-18 Model with Softmax Loss frame-by-frame similar to Section 5.1. We then extract features from the ResNet model, 64 frames at a time, and feed it into an two layer LSTM, followed by a Fully Connected Layer. We separately train the LSTM + FC Layer with a Softmax Loss. We do not use an end to end model for efficiency purposes.

4.3. 3D Convolutional Neural Networks

An alternate approach for capturing temporal information is 3D Convolutional Neural Networks [15]. 3D-CNNs are similar to 2D-CNNs, except that their filters span multiple frames, so they capture relationship of frames within the filters. Our architecture for 3D-CNNs can be seen in Figure 5. The numbers in the Conv3D layers represent filter size, num filters and stride respectively. We have a Conv3D layer with a filter size 3, stride 3, followed by Batchnorm3D, ReLU, and MaxPool3D. Then we have another Conv3D layer with filter size 3, stride 3, followed by Batchnorm3D and ReLU. Then finally, we have a Fully Connected Layer. We train the network end to end using Softmax Loss.

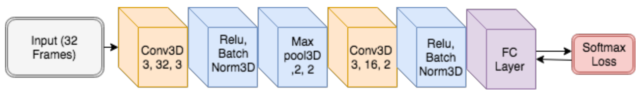

Figure 5: 3D Convolution Neural Network Model. The numbers in the Conv3D layers respectively represent filter size, the number of filters, and the stride. The numbers in Maxpool3D layer respectively represent filter size and stride. Forward arrows represent data flow. Backward arrows represent gradient flow.

5. Experiments

We conducted several experiments with our dataset. They ranged from single-frame models using 2D-CNNs to temporal models leveraging 3D-CNNs or 2D-CNNs combined with LSTMs. The task for all these experiments was classifying the frames or clips into three activities: patient getting out of bed; oral care; and no activity. For each of these classes, our dataset contained 500 samples of 64 frames each, partitioned into training, validation, and testing splits.

5.1. Baseline: ResNet-18 with Individual Frames

As a baseline model for our task, we used a ResNet-18 pre-trained on ImageNet that we finetuned on our dataset. We considered each frame in our dataset as a unique sample labeled with the correct class label. We did not distinguish the samples based on the camera (and thus, viewpoint) they originated from.

Our task was to classify each frame as Patient Getting Out of Bed, Oral Care, No Activity. We had around 40 annotations of each of these activities activity, and we subsampled 500 64 frame samples for each of these activities for our training and validation sets with an 80/20 split. Since we treated each frame as a separate training point instance, we had a total of 25600 training examples and 6400 validation examples.

For training the ResNet-18, we started from a model pretrained on Imagenet, and processed our data according as described in Section 3. We used a 1e-5 learning rate, with a batch size of 32. For our loss function, we used the cross-entropy loss, and for optimization we used the Adam opti-

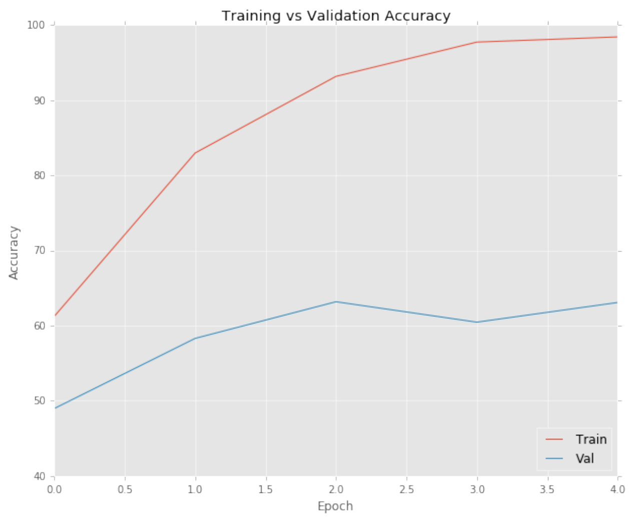

Figure 6: A comparison of the classification accuracy in the training and validation sets for the frame-by-frame ResNet-18 model. The model overfits the training set, but only achieves 60% accuracy on the validation set of the binary classifier.

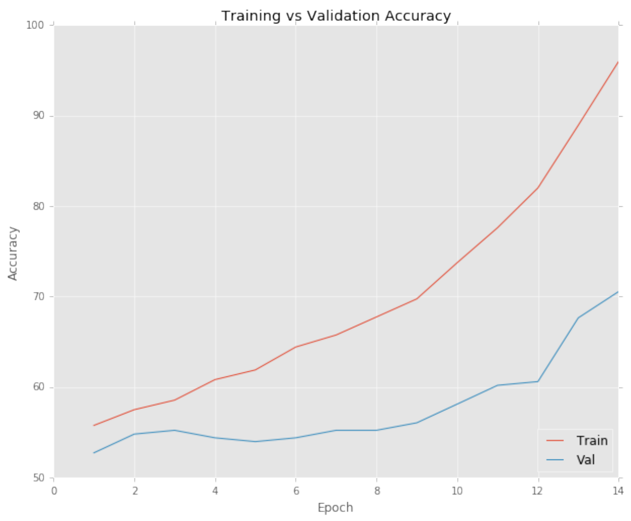

Figure 7: A comparison of the classification accuracy in the training and validation sets for the Depth-Combined Frames ResNet-18 model. We note that the validation accuracy outperforms that of the frame-by-frame model.

mization function with betas 0.9 and 0.99 respectively. We ran this setup for 5 epochs, and we were able to achieve the accuracies as seen the Figure 6.

The validation accuracy peaks at 63.18%, while the model is able to overtrain on the training data. Further, this approach was able to reach an accuracy of 60.12% on the test set. The main reason for the low accuracies this is that we are not using any sort of temporal information, and we are not using the information from different viewpoints. Since our dataset is relatively small, the model does not generalize well to different viewpoints.

5.2. ResNet-18 with Depth-Combined Frames

One limitation with our baseline is that the model did not leverage the information from different viewpoints at each timestep. That is, we are not using this data collectively; we are treating the viewpoints as individual instances despite referring to the same annotation. In an attempt to solve this issue, we modified our dataset such that every frame was combined depth-wise. The first three channels were from the first viewpoint, the next three channels were from the second viewpoint, and the last three channels were from the third viewpoint. Thus, our input had nine channels. Since not all viewpoints were recorded at the same time, we use the nearest neighbors of the timestamps as described in 2.2. We then sub-sampled 200 clips for each of the activities, and pre-processed the data as described.

We did not use a pre-trained ResNet for this approach since they were trained on inputs with three channels. Instead, we created a modified ResNet-18 model which accepts input images of nine channels that we train from scratch. We used a learning rate of 1e-4. Similar to our last experiment, we use batch sizes of 32, a cross-entropy loss, and the Adam optimization function with betas of 0.9 and 0.99 respectively. We train the model for five epochs, and were able to achieve the accuracies reported in Figure 7.

As seen with this approach we were able to reach validation accuracies of up to 71.76%, and it reached an accuracy of 59.28% on the test set. The two main reasons for the failure of this approach is that the validation accuracies were very dependent on the initialization, and we are finding the nearest frame, which may actually be quite far away in real time. We did not pursue further modifications to this approach mainly because it is not generalizable - 3 viewpoints in one ICU room may be very different, and possibly unrelated from 3 viewpoints in an other ICU room. Thus, a viewpoint invariant approach or a 3D point cloud approach is better served for our task.

5.3. Long Short Term Memory Unit Viewpoint Invariant

The next experiment we tried was to use temporal information. For our data we subsampled 500 clips of 64 frames of each of the activities (Oral Care, Getting Out of Bed, No Activity) and split them into the training/validation set with an 80-20 split. Further we had a single long clip as our test as described in Section 3. For our model, we decided to use a ResNet-18 model followed by an LSTM as

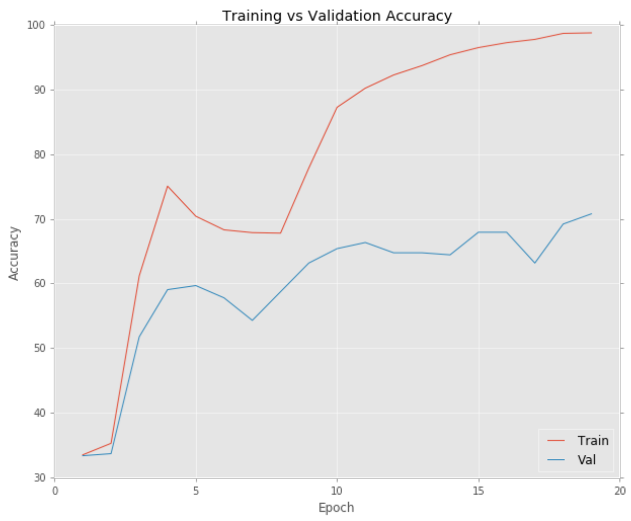

Figure 8: A comparison of the classification accuracy in the training and validation sets for our combined ResNet-18 and LSTM model across all viewpoints on ternary classification.

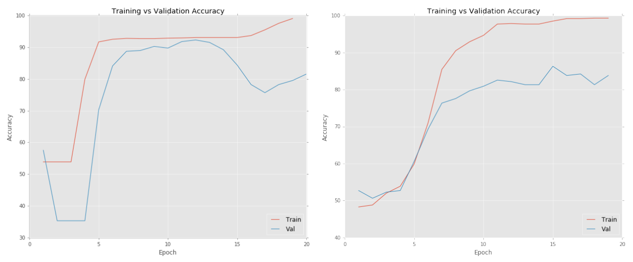

Figure 9: A comparison of the classification accuracy in the training and validation sets for our combined ResNet-18 and LSTM model across all viewpoints on binary classification. The first graph represents the binary classification of "Patient Getting out of Bed" and the second one represents "Oral Care".

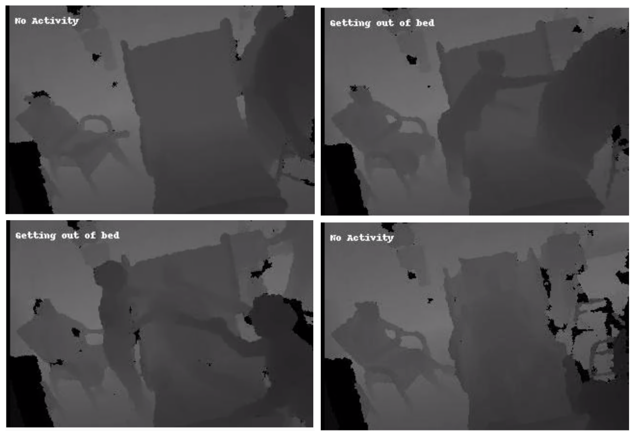

Figure 10: Test time results on a long unseen clip. For the entire video sequence please refer to supplementary materials.

shown in Figure 4. For efficiency, we trained the ResNet-18 separately from the LSTM and FC layer. Thus, we first trained ResNet-18 as a single frame model similarly to Section 5.1. Then we extracted features of 64 frames at a time from this ResNet-18 model and fed them into the LSTM. For the LSTM we used a batch size of 10 (10 inputs of 64 frames each), a learning rate of 2e-5, 2 layers, the softmax loss function and an Adam optimizer with betas 0.5 and 0.99. The LSTM has a hidden size of 20. We initialized our hidden state and cell state to be all 0s. With this model, we were able to achieve validation accuracy of 71.28% as seen in Figure 8. However, on the test set we were only able to reach an accuracy of 62.22%. The main reason for this is that the validation accuracy was very much based on the initialization. Over multiple runs, on the same hyper-parameters validation accuracies would not go up at all. We tried different forms of initialization including Xavier Initialization, but it did not help. After looking at our test data where it was going wrong, we realized it went wrong mostly when the viewpoint was occluded or the activity was barely visible in the viewpoint. Thus, we decided to try out activity recognition from a single viewpoint - facing the bed - that is not occluded in any room.

5.4. Long Short Term Memory Unit Single Viewpoint

As we saw in the previous approach our model was very prone to initialization, so we decided to change our dataset to train more easily. Specifically, we used data from only one view point (camera facing directly at the patients bed). Then we sub-sampled from this new dataset. Further, instead of training a single ternary classifier, we decided to train two binary classifiers: one for Patient Getting out of Bed (true or false), and one for Oral Care (true or false). If both were false it meant there was No Activity. For our model, we used a very similar approach to Section 5.3.

The binary classifier for Oral Care reached a validation accuracy of 87.55% as seen in Figure 9, and was able to reach an accuracy of 83.28% on the Oral Care test clip, and for Patient getting out of bed the binary classifier achieved a 92.3% accuracy on the training set, and 86.92% accuracy on the test set. The models are still prone to initialization, but they vary much less with different initializations. The best binary models are consistently able to achieve validation accuracies of over 70% over multiple runs. Sample frames from the run on the test set can be seen in Figure 10.

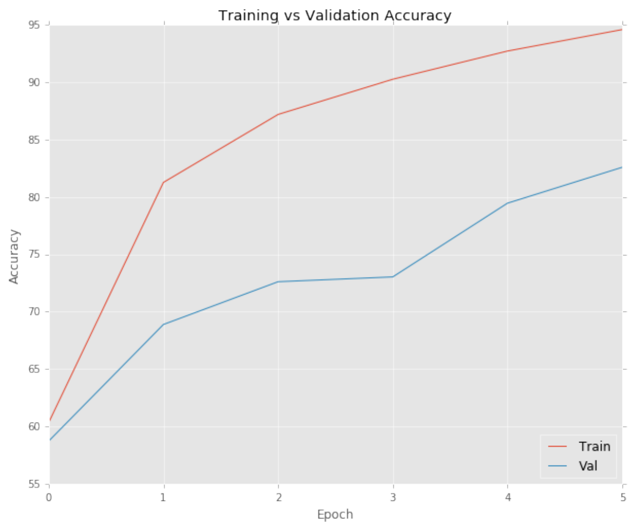

Figure 11: A comparison of the classification accuracy in the training and validation sets for our 3D-CNN model across a single viewpoint (frontally facing the bed) on ternary classification.

5.5. 3D Convolution Neural Network Single Viewpoint

For our final experiment we decided to classify all three activities (Oral Care, Patient Getting out of bed and No Activity) using a 3D-CNN. We used only a single viewpoint and we fed in 32 frames at a time to the 3D-CNN. Since we used 32 frame inputs instead of 64, we had 1000 samples of each class instead of 500. The model can be seen in Figure 5. We used a batch size of 6, a learning rate of 5e-5 and an Adam optimizer with betas 0.9 and 0.99. With this setup we were able to reach a validation accuracy of 82.59% as seen in Figure 11. Further, the model reached a test accuracy of 81.36%. Also, this approach was much less prone to initialization and gave very similar accuracies over multiple runs using the same hyperparameters.

6. Conclusion

In this paper we presented a novel dataset for action recognition in ICUs. Using multiple depth sensors in ICU rooms, we tried to build a dataset for seven relevant activities. However, due to limitations in data collection and issues with the sensors, we currently restricted our final dataset to two activities and background clips. A particular challenge of the dataset is learning using only depth data instead of the more common task of analyzing RGB or even RGB+D data.

We experimented with a variety of models that benefitted from multiple characteristics of the dataset. As expected, we see that models leveraging temporal information outperform those that classify using individual datapoints. Additionally, we observe that models leveraging simultaneous information from multiple viewpoints outperform those that treat each viewpoint as a separate instance. The models tend to not generalize well and are sensitive to initialization parameters since the dataset is relatively small. Despite our constrained dataset, we were able to achieve high accuracy for a classification task with three activity types.

7. Next Steps

An natural continuation to our work is expanding our dataset. We plan on collecting more data and annotations in order to be able to classify clips into the original seven activity types. In addition, a larger number of datapoints would help the models generalize better, both in terms of differences in viewpoints and in how activities are performed.

Once our system can successfully log the basic activities we noted, we plan to expand it to detect anomalies such as emergency situations. To do so, we could potentially use a dataset of simulations of different emergencies (e.g., patient falls on the floor).

We also plan to investigate pre-processing the data with joint detection. This could help the model learn more generalized actions for different activities, as well as for the detection of emergencies (e.g., figure out when a patient falls on the floor from the movement of joints).

Finally, we previously hinted at the possibility of creating a 3D point cloud using data from multiple viewpoints. With this approach, the model could potentially leverage information that would not otherwise be available, as well as generalized better to different combinations viewpoints. We note that the vast majority of previous research in 3D point cloud formation has focused on RGB-D data. We would like to expand this line of work into depth-only inputs.

Finally, we want to explore the possibility of using weakly labelled data. Currently, once we receive the approximate timestamps from the nurse annotations, we carefully annotate the exact time boundaries for each activities. With this alternative approach, however, would would be able to predict those values, which would allow us to more easily scale the size of our dataset. In order to achieve this, we plan on creating a model that will be able to find the boundaries of each activity given the nurse annotation.

8. Acknowledgements

We would like to thank Intermountain Healthcare for providing us with data and support for this project. We would also like to thank the Stanford SVL Healthcare team, particularly Albert Haque and Alexandre Alahi, for valuable feedback.

References

[1] S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*, 2016.

[2] F. Caba Heilbron, J. Carlos Niebles, and B. Ghanem. Fast temporal activity proposals for efficient detection of human actions in untrimmed videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1914–1923, 2016.

[3] F. Caba Heilbron, V. Escorcia, B. Ghanem, and J. Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 961–970, 2015.

[4] Daniel, S. Maturana, and Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. 2013.

[5] 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.

[6] M. Dou, S. Khamis, Y. Degtyarev, P. Davidson, S. R. Fanello, A. Kowdle, S. O. Escolano, C. Rhemann, D. Kim, J. Taylor, et al. Fusion4d: Real-time performance capture of challenging scenes. *ACM Transactions on Graphics (TOG)*, 35(4):114, 2016.

[7] 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.

[8] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[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.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[11] D. Maturana and S. Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 922–928. IEEE, 2015.

[12] R. A. Newcombe, D. Fox, and S. M. Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 343–352, 2015.

[13] N. Robertson and I. Reid. A general method for human activity recognition in video. *Computer Vision and Image Understanding*, 104(2):232–248, 2006.

[14] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang. Convolutional neural networks for medical image analysis: full training or fine tuning? *IEEE transactions on medical imaging*, 35(5):1299–1312, 2016.

[15] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. *arXiv preprint arXiv:1412.0767*, 2014.

[16] S. Yeung, A. Alahi, Z. Luo, B. Peng, A. Haque, A. Singh, T. Platchek, A. Milstein, and L. Fei-Fei. Vision-based hand hygiene monitoring in hospitals. In *NIPS Machine Learning for Health Care Workshop*, 2015.