VIDEO-BASED ACTIVITY RECOGNITION IN SURVEILLANCE FOOTAGE: A DEEP LEARNING APPROACH

Oman, Kaizer
Saint Louis University
Bakakeng Sur, Baguio City
2222613@slu.edu.ph

Del Rosario, Remo
Saint Louis University
Bakakeng Sur, Baguio City
2227393@slu.edu.ph

Gura, Kaizer Cyn
Saint Louis University
Ambiong Road, Baguio City
2221097@slu.edu.ph

Narag, Ava
Saint Louis University
Bakakeng Sur, Baguio City
2224512@slu.edu.ph

Andaya, Trisha
Saint Louis University
Ambiong, La Trinidad
2211741@slu.edu.ph

Ordoña, Elishah
Saint Louis University
San Roque Village, Baguio City
2213960@slu.edu.ph

Bato, Leovide Daniel
Saint Louis University
Bakakeng Sur, Baguio City
2223451@slu.edu.ph

ABSTRACT

The increasing importance of Human Activity Recognition (HAR) in fields such as security, healthcare, and emergency response involves effective methods for analyzing human behaviors. Despite numerous studies on action classifiers, challenges persist in surveillance footage due to the large field of vision, multiple concurrent activities, and varying object sizes and perspectives. This paper presents a video-based HAR system, highlighting the significance of the 3D Convolutional Neural Networks (3D CNN) in detecting human activities from surveillance videos. Hence, an action recognition model leveraging the 3DCNN architecture trained from scratch using the SPHAR dataset was made. The model achieved a mean average precision (mAP) of 34% with most classes reaching over 10% average precision. The results demonstrated that 3D CNNs can enhance HAR in surveillance videos, although it needed further refinements to address limitations and achieve higher mAP result.

Keywords: Human Activity Recognition, 3D Convolutional Neural Network, SPHAR dataset, Video-Based Recognition, Surveillance video action recognition, Spatio-temporal Features

I. INTRODUCTION

Human Activity Recognition (HAR) holds significant importance in today's world due to its wide range of applications across various fields, such as healthcare, sports, security, and human-computer interaction. This is due to its essence, which lies in its ability to understand, interpret, and analyze human activities using sensors, videos, data, and machine-learning algorithms [5]. Every action that is performed by humans in their daily lives is called ‘human activities,’ which encompass a wide range of behaviors and vary significantly depending on various factors like their culture, preferences, and lifestyle.

There are two ways in which HAR can be achieved. It could be sensor-based or video-based [5]. Sensor-based systems employ either on-body or ambient sensors to estimate people's motion specifics which are monitored using various types of sensors [5]. This includes ambient sensors (GNSS, Cellular, WiFi), wearable sensors (Accelerometer, Gyroscope, Magnetometer), and others (Event Camera) [5,6]. As for video-based HAR, these are models that recognize the action being performed by humans through recorded videos [5] from cameras that are placed at certain fixed predetermined locations [4]. HAR involves detecting both simple or complex actions in real-world settings, offering insights into the specific physical actions being detected by the computer. Such capabilities enable computer systems to aid users with their tasks, thereby improving the quality of life in the different sectors of the profession [2].

HAR systems exist with different types as they serve different purposes. Firstly, ambient sensor-based systems are used to capture human-environment interactions because they are mainly installed in fixed locations or in smart homes to measure different environmental parameters[5,6]. Wearable sensor-based systems, as the name implies, are portable and are usually installed in smartphones, watches, and even clothes. They are used to recognize the actions performed by the person carrying them. Finally, video-based systems are used to recognize single or multiple actions of humans that are caught on the video. This paper aims to implement a video-based HAR model to predict a single action performed in a video caught from long-shot cameras, specifically from surveillance footage.

Action recognition in surveillance videos poses significant challenges due to multiple factors, including a large field of vision, the presence of multiple concurrent activities, varying object sizes and perspectives, and the untrimmed nature of the

footage. Most existing action classifiers are designed to work with short, trimmed videos, which is impractical for recognizing actions in real-world security video scenarios[14]. This highlights a necessity for the development of more robust methods tailored to the complexities of action recognition in surveillance videos.

In relation to this gap is a recent study conducted by Jeong using a YOLO-slowfast architecture, the result highlighted significant challenges in achieving high precision across various action classes. These include dataset imbalance, misalignment of bounding boxes with activities, object sizes, and potential flaws in the YOLO loss function implementation, suggesting the need for a better architectural approach.

Building upon these insights, this raises a hypothetical question about the potential outcomes of implementing a 3D CNN architecture in predicting actions in surveillance videos. Behind the architecture selection for this study, is from the analysis conducted by Pareek et al.(2021). They recommended the use of 3D RCNN for DL-based HAR models because of its ability to better exploit spatio-temporal features, which are crucial for improving model performance. The spatio-temporal feature is used when data is collected across both space(Spatial) and time(Temporal), wherein spatial information is used to capture geographical attributes of the data and temporal information for capturing motion cue overtime. Furthermore, in line with the paper's significance, this study is also valuable for public security research, as it contains some videos labeled with instances of suspicious and harmful behavior.

To further explore the efficiency of the 3D Convolutional Neural Network (3D CNN). This architecture is a powerful model that uses spatial-temporal features that can be used for gesture recognition [12]. The 3D convolutions extract the features from the temporal and spatial dimensions, which can be done by convolving three-dimensional kernels to the obtained cube by assembling more than one spatial temporal patch that is arranged in a contiguous method. The features present in the convolution layer are linked with multiple frames arranged continuously to the previous layer to gather information about the motion of the action [11]. The 3D CNN is utilized to detect complex images including videos. The advantage of 3D CNN in image classification is that it lies in their capacity to handle the temporal continuity of video frames. This is important for understanding the dynamic scene and activities that unfold over time which can not be effectively captured by 2D CNN that only analyzes individual frames [12].

The study of Vrskova et al. [12], Which implemented the 3D detection of human activity from the dataset of UCF YouTube Action, provided a better result than neural network architecture for motion, static, and hybrid features. The overall test was obtained with an accuracy of 85.2%. The precision reached 87.45, and recall with 85.6%. However, the neural network architecture for motion, static and hybrid features achieved a precision of 65.4%, 63.1%, and 71.2%. Furthermore, the modified UCF101 dataset has been added, resulting in an accuracy of 84.45%. The precision was 82.7%, and the recall was 87.7%. Based on the result it clearly shows that the 3D CNN is capable of neural network architecture for the classification of human activities [12]. Another study displays the efficiency of the model, according to Arunnehru et al. (2018), using 3D CNN on the KTH dataset shows correct prediction with an average precision, recall, and f-measures of 94.9%. Moreover, the use of the Weizmann dataset and the performance metrics that were obtained using 3D CNN resulted in a precision, recall, and f-measure of 97.2%. The 3D CNN outperformed other architectures results [11].

II. METHODOLOGY

The proposed video-based human activity recognition(HAR) system utilizes the capabilities of 3D CNN to extract spatial and temporal features from surveillance videos. This section describes the data collection, preprocessing, model architecture, training procedures, hyperparameter tuning, and evaluation metrics used in this study.

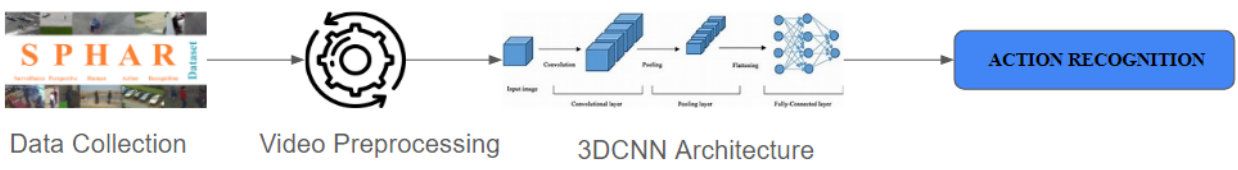


Figure 1: Overview of the video-based human activity recognition using SPHAR

II.1. Data Collection

In computer vision-based models, the first step is data collection. In this study, we utilize the SPHAR video dataset. SPHAR stands for Surveillance Perspective Human Action Recognition Dataset, which contains over 7,000 long-shot videos for 14 action classes. The dataset was retrieved from the repository of Alexander Melde wherein he mentioned that the videos that were aggregated from multiple sources and converted to a consistent file type (H265 HEVC .mp4), cutted and cropped (spatio-temporally) to contain only one action at a time and sorted into 14 action classes.

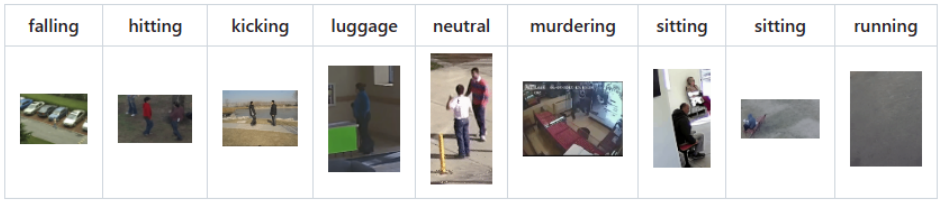


Figure 2: Samples from SPHAR dataset

Additionally, since the focus is on action recognition the ‘Luggage’ and 'Neutral' class, which composes a percentage of 0.1% and 28% of the data respectively, was excluded. This decision was able to simplify the model’s task and was made because of the perceived lack of significant actions in these classes compared to other classes.

CLASS	Percentage Ratio
Car Crash	3%
Falling	2%
Hitting	7%
Igniting	2%
Kicking	2%
Murdering	1%
Panicking	1%
Running	9%
Sitting	11%
Stealing	8%
Vandalizing	4%
Walking	50%

Table 1: SPHAR Dataset Classes with their corresponding percentage ratio

Table 1 shows the remaining classes of the dataset and their corresponding percentage ratio of the total number of videos present after removing the ‘Luggage’ and ‘Neutral’ classes. Notably, "Walking" dominates the dataset with the highest percentage ratio of 50.2%, indicating its prevalence among the recorded incidents, while both "Murdering" and “Panicking” classes account for the smallest proportion at 1%.

II.2. Model Architecture

Our implementation of the 3D CNN was inspired by the code from the '3d-cnn-action-recognition' repository by Dipak Kumar using UCF101. The steps include frame selection, frame resizing, gray scaling, and array conversion. Further

preprocessing was done before feeding it to the 3D CNN implementation. This includes splitting them using a ratio of 90:10 to separate the unseen dataset, then further splitting the training set into training and validation sets with a ratio of 80:20, then during frame selection we added a feature to remove corrupted videos since some of the videos were identified as such. Finally, the videos were fed into the 3D CNN implementation.

II.3. Training Procedures

During training, the following configuration parameters were applied: The model was trained with a batch size of 16 over 50 epochs. The dataset encompasses 12 distinct classes of human activities. Leveraging RGB images, the model capitalized on color information to enhance its understanding of spatial and temporal dynamics inherent in human actions. Continuous frame selection methodology was employed, enabling the model to seamlessly process frames and capture temporal nuances. Lastly, each video was sampled with a depth of 10 frames, ensuring a balance between temporal context and computational efficiency. These parameters were chosen to ensure efficient training of the 3D CNN model on the SPHAR dataset.

II.4. Evaluation Metrics

To validate the model's performance, several metrics were evaluated, including the training and validation accuracy and model loss for every 10 epochs, mean average precision(mAP) of the whole model, and the average precision(AP) for each class. These metrics provided a comprehensive assessment of how well the model was performing and allowed for the identification of any areas needing improvement. By examining both training and validation accuracy, we could determine if the model was overfitting to the training data. The model loss metric helped in understanding how well the model was optimizing, while the mAP of the whole model and the AP of each class offered insights into the model's ability to handle imbalanced classes and make accurate predictions across different classes.

III. RESULTS AND DISCUSSION

This section includes the performance metrics and their analysis interpretation of the methods and experiments employed in the 3D CNN Architecture outturn. This section also presents the model's predictions.

III.1. Training and Validation Graph Analysis (10-50 Epochs)

The training of the model was done starting from epoch 10 to epoch 50 to find the best result for the model without overfitting or to prevent the model from memorizing the dataset.

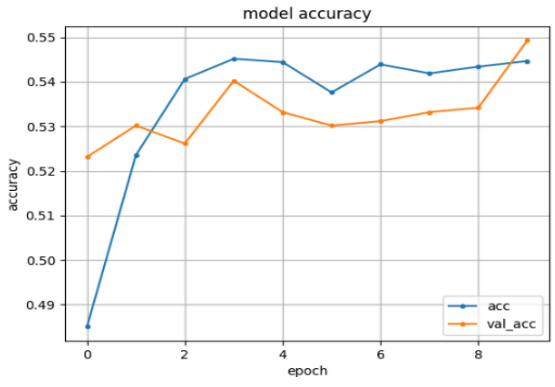


Figure 3: Model Accuracy (10 epochs)

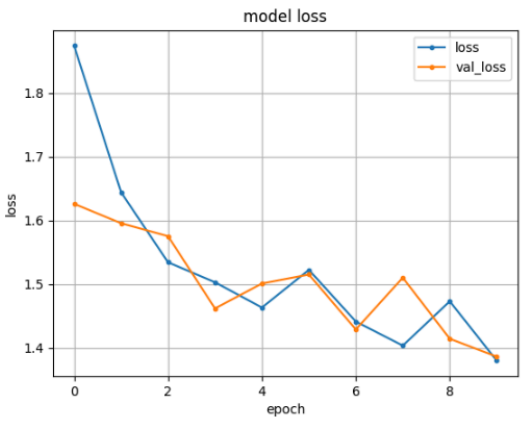


Figure 4: Model Loss (10 epochs)

The figure above on the left (Figure 3) is the result of the model with the training of 10 epochs. The model can be seen performing well due to the increase in both training accuracy and validating accuracy.The testing accuracy of the 10 epochs training resulted to 54% while the validating accuracy resulted to a 55%.The number of 10 epochs is limiting the capabilities of the model because it doesn't have enough iterations to learn more about the dataset.

The figure above on the right (Figure 4) is showing the loss for the model with 10 epochs of training. The model showed great performance due to the evident decrease in the value of the loss in both training and validation. The loss for the 10 epochs is showing the issue with the 10 epoch accuracy; they both didn't have enough time to train with the dataset that is the limiting factor for the both metrics in the 10 epochs.

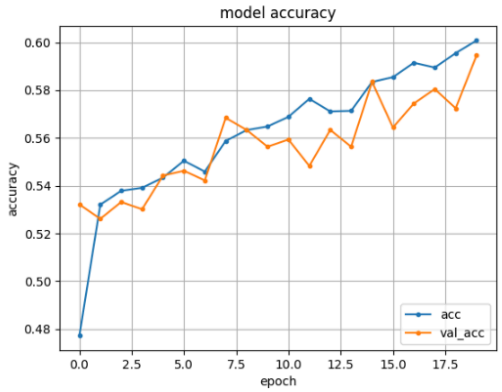


Figure 5: Model Accuracy (20 epochs)

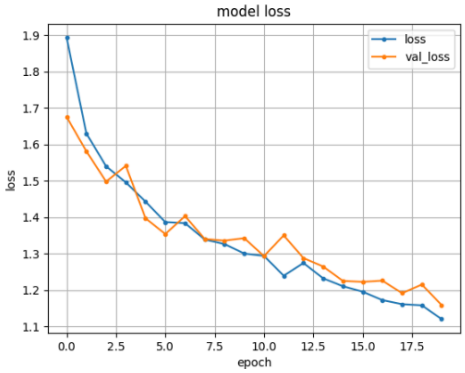


Figure 6 : Model Loss (20 epochs)

In Figure 5, the model is trained for 20 epochs. The accuracy of the model has shown a steady increase compared to the 10 epochs meaning that the model is performing better compared to the model that was only trained for 10 epochs.The model achieved a test accuracy of 60% and a validation accuracy of 59%.. Based on the figure above the increasing accuracy up until the very last epoch indicates that the model can still have improvement with the increase of the iteration.

The model metrics in terms of loss with respect to training with 20 epochs yielded the same result with the accuracy as shown in Figure 6, because they both decrease steadily. The model loss resulted in around 1.15 in training loss and 1.75 in validation loss. The training of the model with 20 epochs proved to be insufficient because the model is still showing improvement up until the last epoch meaning the model can still have improvement with the increase of number in iteration.

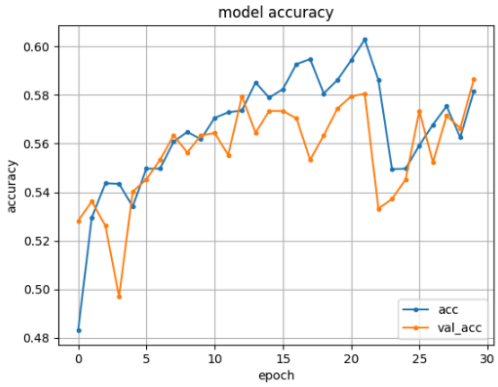


Figure 7 : Model Accuracy (30 epochs)

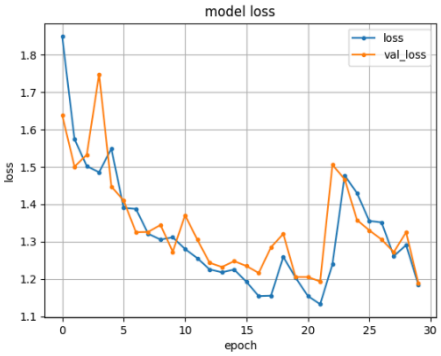


Figure 8: Model Loss (30 epochs)

Figure 7 above shows the result of the model with the training of 30 epochs. The model showed great improvement compared to the 20 epochs but it had a dip in the accuracy around 20 epochs meaning the model is starting to become overfitted or the model is starting to memorize the dataset. This means that the model will have to be stopped in training around 40-50 epochs. The accuracy for the 30 epochs resulted in 58% in testing accuracy and 59% in validating accuracy.

Figure 8 shows the training model loss for 30 epochs of training. The model shared the similar result compared to the 30 epochs of accuracy. The model is starting to overfit because it is training for too much iteration. The model loss resulted in around 1.2 in both testing loss and validation loss.

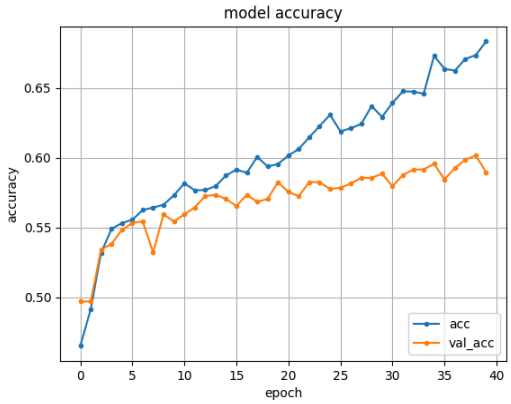


Figure 9: Model Accuracy (Best Result)

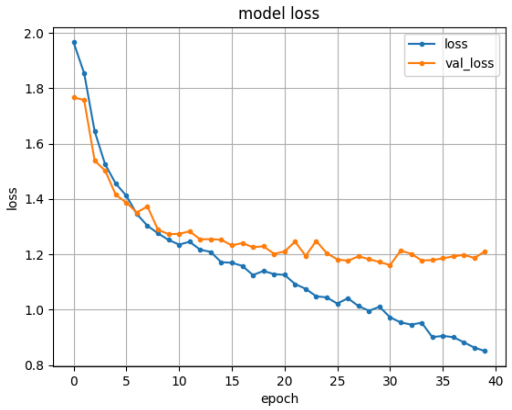


Figure 10: Model Loss (Best Result)

Figure 9 presents the accuracy of the 3D CNN model on both the training and validation sets. It can be seen that the training accuracy increases steadily throughout the epochs from below 50% to above 65% accuracy, which indicates the increasing model performance on the data it was trained on. Similarly, the validation accuracy increases throughout the epochs but not as much as the training accuracy, resulting in a widening gap between the training and validation accuracy, which might result in more overfitting as the epochs progress further.

Figure 10, on the other hand, presents the model loss. Loss measures how well a model's predictions match the actual labels. A lower loss indicates a better model. Similar to the accuracy graph of the model, its loss graph indicates a sign of overfitting as there is a widening gap between the training and validation loss as the epochs progress further. By epoch 40, the training loss is around 0.9, while the validation loss is around 1.2

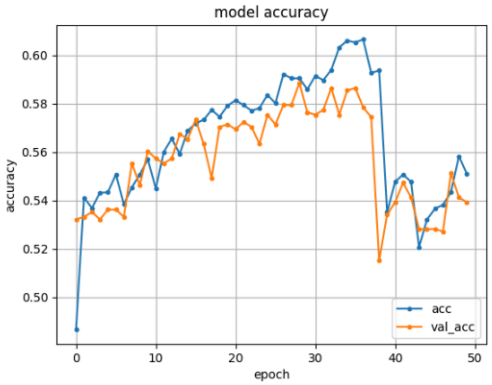


Figure 11: Model Accuracy (50 epochs)

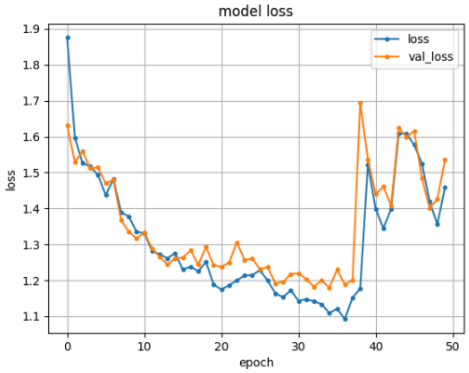


Figure 12 : Model loss (50 Epochs)

The figure on the left (Figure 11) shows the result of the training with 50 epochs. The model shows a very evident case of overfitting meaning the model no longer predicts the output but instead memorizes the answer based on the training dataset. The graph also shows there is a dip around 45 epoch meaning that it is overfitted. The takeaway from the graph of this model is that the training should be halted at an earlier iteration to prevent the overfitting of training.

The figure on the right (Figure 12) shows the result of model loss with a training of 50 epochs. The model shows a very similar result to the accuracy of the model with the same epoch. Based on the 2 graphs given by training with 50 epochs the number of iteration proved that the model is overfitted with these many iterations. The loss for the model resulted in 1.5 training loss and 1.55 validation loss which meant that the model performed worse compared to the previous number of iterations of 40 epochs.

Based on the detailed analysis provided for the models trained with different numbers of epochs, it is evident that the model trained with 40 epochs outperformed the others. The analysis highlights that training for 40 epochs strikes a balance between learning from the dataset and preventing overfitting. The accuracy and loss metrics at 40 epochs demonstrate better performance compared to both shorter and longer training durations. Therefore, it's reasonable to conclude that the model trained with 40 epochs is the most suitable choice for the system.

II.2. Evaluation Metrics

Class	Average Precision(Validation Set)	Correct Predictions(Unseen set)
Car Crash	21%	6/16
Falling	20%	3/12
Hitting	25%	19/39
Igniting	18%	8/10
Kicking	13%	7/12
Murdering	1%	2/5
Panicking	10%	0/6
Running	30%	24/52
Sitting	47%	15/61
Stealing	58%	32/42
Vandalizing	17%	7/21
Walking	77%	247/280

Table 2: Model's AP in validation set and correct predictions in unseen set for each class

The overall mean average precision of the model is 34%. Shown in Table 2 is the list of the average precisions for each class. Interestingly, the Average Precision (AP) of each class revealed that Classes with higher percentage ratios in the dataset, such as "Walking" and "Sitting," also tend to have higher average precision values, indicating that the model performs well on these frequently occurring incidents. Conversely, classes with lower percentage ratios, like "Luggage" and "Murdering," often exhibit lower average precision values, suggesting that the model's performance may be less consistent for these less common incidents. Additionally, the correct predictions of the model in predicting the unseen dataset for each class is represented by x/y in column 3; where x is the number of correct video predictions, and y is the total number of videos stored in the unseen data for that class. The model tends to perform well on classes with higher frequency in the dataset and those with higher average precision. Although some of the classes were underrepresented below, like falling, kicking, and car crash, the model was still able to smoothly predict some of its unseen data except for the panicking class, which might be due to the ambiguity and variability of the visual features associated with the unseen set of the class.

III.3. Model Predictions

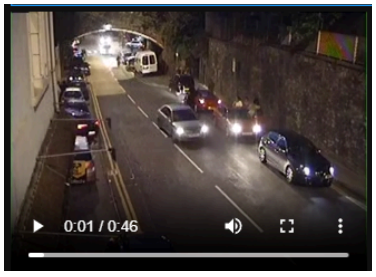
```
print("Top 5 Predictions:")
for i, (class_name, probability) in enumerate(zip(top_5_classes, top_5_probabilities), 1):
    print(f"{i}. Class: {class_name}, Probability: {probability:.4f}")
```

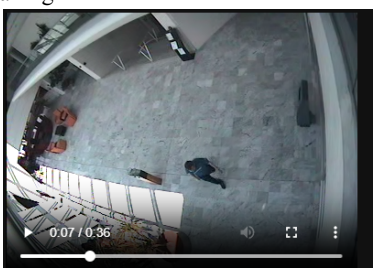
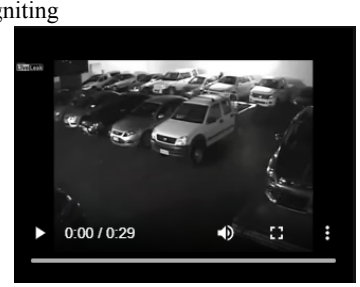
Code Snippet 1: Display predicted classes with their corresponding probabilities

```
from IPython.display import HTML
from base64 import b64encode

# Function to display a frame from the video
def display_video_frame(filepath, frame_number):
    video_tag = '<video controls src="data:video/mp4;base64,{0}" type="video/mp4">'.format(
        b64encode(open(filepath, "rb").read()).decode('utf-8'))
    display(HTML(video_tag))

# Display frame number 100 from the video
display_video_frame(filepath, 10)
```

Code Snippet 2: Display a frame from the video

Code Snippet 1 is used to display the predicted classes by the model that is recognized in the video. It is further coupled with their corresponding probabilities. Code Snippet 2 displays a specific frame from the video (i.e. at the 10-second mark), still analyzing the whole video. This is beneficial for the proponents to easily visualize and analyze if the result predicted the correct class.

Sample Videos per Activity	Top 5 Predictions
1. Stealing 	Top 5 Predictions: 1. Class: stealing, Probability: 0.8209 2. Class: vandalizing, Probability: 0.0916 3. Class: igniting, Probability: 0.0547 4. Class: hitting, Probability: 0.0176 5. Class: carcrash, Probability: 0.0084
2. Car Crash 	Top 5 Predictions: 1. Class: stealing, Probability: 0.5156 2. Class: walking, Probability: 0.3620 3. Class: hitting, Probability: 0.0590 4. Class: murdering, Probability: 0.0211 5. Class: carcrash, Probability: 0.0144

3. Falling 	Top 5 Predictions: 1. Class: walking, Probability: 0.5020 2. Class: luggage, Probability: 0.2035 3. Class: hitting, Probability: 0.1976 4. Class: sitting, Probability: 0.0439 5. Class: falling, Probability: 0.0317
4. Hitting 	Top 5 Predictions: 1. Class: hitting, Probability: 0.8246 2. Class: kicking, Probability: 0.1432 3. Class: sitting, Probability: 0.0241 4. Class: walking, Probability: 0.0065 5. Class: running, Probability: 0.0007
5. Igniting 	Top 5 Predictions: 1. Class: igniting, Probability: 0.3579 2. Class: hitting, Probability: 0.1755 3. Class: stealing, Probability: 0.1471 4. Class: carcrash, Probability: 0.0946 5. Class: kicking, Probability: 0.0720
6. Kicking 	Top 5 Predictions: 1. Class: kicking, Probability: 0.5309 2. Class: hitting, Probability: 0.4491 3. Class: walking, Probability: 0.0073 4. Class: carcrash, Probability: 0.0054 5. Class: sitting, Probability: 0.0052
7. Murdering 	Top 5 Predictions: 1. Class: murdering, Probability: 0.4748 2. Class: stealing, Probability: 0.2929 3. Class: hitting, Probability: 0.1292 4. Class: vandalizing, Probability: 0.0637 5. Class: igniting, Probability: 0.0280

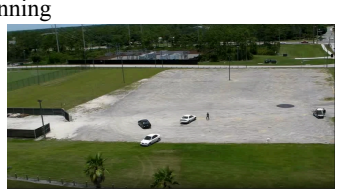
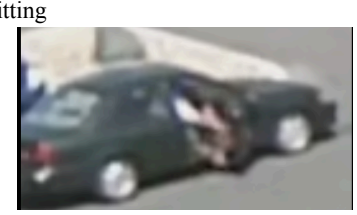
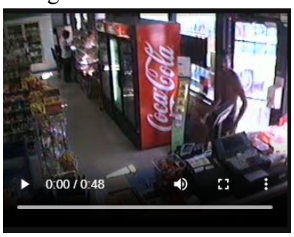
8. Panicking 	Top 5 Predictions: 1. Class: walking, Probability: 0.4157 2. Class: running, Probability: 0.2945 3. Class: panicking, Probability: 0.1604 4. Class: falling, Probability: 0.0531 5. Class: hitting, Probability: 0.0340
9. Running 	Top 5 Predictions: 1. Class: running, Probability: 0.4140 2. Class: walking, Probability: 0.2573 3. Class: hitting, Probability: 0.1664 4. Class: vandalizing, Probability: 0.1601 5. Class: stealing, Probability: 0.0007
10. Sitting 	Top 5 Predictions: 1. Class: sitting, Probability: 0.6160 2. Class: hitting, Probability: 0.1469 3. Class: falling, Probability: 0.1397 4. Class: walking, Probability: 0.0575 5. Class: kicking, Probability: 0.0194
11. Vandalizing 	Top 5 Predictions: 1. Class: vandalizing, Probability: 0.4427 2. Class: stealing, Probability: 0.3986 3. Class: hitting, Probability: 0.1195 4. Class: igniting, Probability: 0.0118 5. Class: kicking, Probability: 0.0080
12. Walking 	Top 5 Predictions: 1. Class: walking, Probability: 0.7451 2. Class: carcrash, Probability: 0.0927 3. Class: igniting, Probability: 0.0554 4. Class: hitting, Probability: 0.0390 5. Class: running, Probability: 0.0313

Table 3: Sample Videos of each Activity and its Top 5 Predictions

Table 3 showcases the model's predictions on two sample videos from the SPHAR dataset. In the first example, the model has successfully identified the video content, ranking the classes in descending order of probability as follows: (a) stealing, (b) vandalizing, (c) igniting, (d) hitting, and (e) car crash. Noteworthy is the class 'stealing,' which stands out with the highest probability of 0.8209. This prediction aligns with the actual scenario depicted in the video, where a masked individual is observed stealing from a convenience store. The model's accurate recognition underscores its proficiency in discerning complex actions within video content.

However, in the second example, the model fails to correctly predict the action as "car crash." Instead, it assigns the highest probability to "stealing," followed by "walking," "hitting," "murdering," and "car crash" with the lowest probability of

0.0144. This discrepancy suggests that the model may struggle with accurately identifying certain types of actions, particularly those that are underrepresented in the training data or visually similar to other classes.

IV. CONCLUSION AND RECOMMENDATIONS

Iterating the research question about the potential outcomes of implementing a 3D CNN architecture in predicting actions in surveillance videos. By leveraging spatio-temporal information from the data and the 3D CNN architecture trained from scratch, the study was able to show a model with an average precision of 34%. This performance, while not exceedingly high, demonstrates the potential of 3D CNNs in capturing complex human activities in surveillance videos

The findings of this study suggest that 3D CNNs can indeed contribute to the field of video-based human activity recognition, particularly in surveillance footage which includes a large field of vision. However, the moderately low performance of the model and its fair accuracy for both training and validation highlights the need for further refinement. These challenges may be due to the imbalance in the representations of the classes wherein most of the classes were significantly underrepresented relative to the overrepresented ‘walking’ class. This may also be attributed to the lack of further video preprocessing and data augmentation, as well as early stopping mechanisms for training to reduce model overfitting.

With the existing limitations of the model, future works could focus on several areas: collecting a larger and more balanced dataset to ensure all action classes are well-represented; applying more thorough preprocessing steps to clean and normalize the video data; perform data augmentation techniques like flipping and rotation to enhance vision flexibility; adding bounding boxes on each human object on the video to focus on a specific action to prevent misinterpretation due to multiple actions being represented; adding early stopping mechanisms to prevent model overfitting, and using pre-trained models for faster training and better model performance.

REFERENCES

[1] S. Zhang et al. 2017. A Review on Human Activity Recognition Using Vision-Based Method. J Healthc Eng. 2017, 3090343. Retrieved from https://doi.org/10.1155/2017/3090343

[2] N. Golestani, and M. Moghaddam. 2020. Human activity recognition using magnetic induction-based motion signals and deep recurrent neural networks. Nat Commun. 11, 1551. Retrieved from https://doi.org/10.1038/s41467-020-15086-

[3] V. Aruna, S. Aruna Deepthi, and R. Leelavathi. 2022. Human Activity Recognition Using Single Frame CNN. In Applications of Artificial Intelligence and Machine Learning. Lecture Notes in Electrical Engineering, vol 925. B. Unhelker, H.M. Pandey, and G. Raj, Eds. Springer, Singapore. Retrieved from https://doi.org/10.1007/978-981-19-4831-2_17

[4] S. Wan et al. 2020. Deep Learning Models for Real-time Human Activity Recognition with Smartphones. Mobile Netw Appl. 25, 743-755. Retrieved from https://doi.org/10.1007/s11036-019-01445-x

[5] K. Chen et al. 2018. Deep Learning for Sensor based Human Activity Recognition: Overview, Challenges and Opportunities. J. ACM. 37, 4, Article 111 (August 2018), 40 pages. Retrieved from https://dl.acm.org/doi/10.1145/3447744

[6] F. Gu et al. 2021. A Survey on Deep Learning for Human Activity Recognition. ACM Comput. Surv. 54, 8, Article 177 (November 2022), 34 pages. Retrieved from https://doi.org/10.1145/3472290

[7] C. Feichtenhofer et al. 2019. SlowFast Networks for Video Recognition. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), 6201-6210. Retrieved from https://doi.org/10.1109/ICCV.2019.00630

[8] X. Xu et al. 2013. Exploring techniques for vision based human activity recognition: methods, systems, and evaluation. Sensors (Basel). 13, 2, 1635-1650. Retrieved from https://doi.org/10.3390/s130201635

[9] Jeong-O Jeong. 2020. Human Activity Recognition with Computer Vision. Retrieved from https://cs230.stanford.edu/projects_fall_2020/reports/55772236.pdf

[10] A. Melde. 2020. SPHAR-Dataset: Surveillance Perspective Human Action Recognition Dataset: 7759 Videos from 14 Action Classes, aggregated from multiple sources, all cropped spatio-temporally and filmed from a surveillance-camera like position. Retrieved from https://github.com/AlexanderMelde/SPHAR-Dataset

[11] J. Arunnehru, G. Chamundeeswari, and S. Prasanna Bharathi. 2018. Human Action Recognition using 3D Convolutional Neural Networks with 3D Motion Cuboids in Surveillance Videos. ResearchGate. Retrieved from https://www.sciencedirect.com/science/article/pii/S1877050918310044

[12] R. Vrskova. 2022. Human activity classification using the 3DCNN architecture. MDPI. Retrieved from https://www.mdpi.com/2076-3417/12/2/931

[13] P. Pareek and A. Thakkar. 2021. A survey on video-based Human Action Recognition: recent updates, datasets, challenges, and applications. Artif Intell Rev. 54, 2259-2322. Retrieved from https://doi.org/10.1007/s10462-020-09904-8

[14] A. Karbalaie, F. Abtahi, and M. Sjöström. 2022. Event detection in surveillance videos: a review. Multimed Tools Appl. 81, 35463-35501. Retrieved from https://link.springer.com/article/10.1007/s11042-021-11864-2

[15] D. Kumar and Ventakesh. 2019. 3d-cnn-action-recognition. GitHub repository. Retrieved from https://github.com/dipakkr/3d-cnn-action-recognition