Abnormal Behaviour Detection in CCTV Footage

BACHCHAR Ismail

Abstract

Surveillance cameras or CCTV (Closed-circuit television camera) is the most used monitoring systems in public places such as universities, streets, supermarkets, and households for the safety and security of individuals. The analysis of human behavior, for example, robbery, falling/passing out, shooting, and kids getting lost in crowded scenes, is one of the most challenging problems in CV (computer vision). It is generally done manually by humans constantly observing surveillance cameras. A task that needs to be automated, given that its cumbersomeness might lead to non-efficiency in immediate interventions. To remedy this issue, we propose as a first step, different CV algorithms to identify parts of the recording (frames) containing normal or abnormal (unusual) behavior. In this project, we examine the effectiveness of two types of models. The first model is built with CNN architectures and the latter combines CNN and LSTM architectures given the fact that both Convolutional Neural Network (CNN) and Recurrent Neural Networks (RNN) are considered as powerful class of models for applications related to video classification, specifically for anomaly detection tasks. This work examines two CNNs architectures, Resnet50 and Inception v3, with the purpose of distinguishing the best architecture for our work.

1. Introduction

In artificial intelligence, computer vision refers to the field that trains computers to analyze and understand visual information. We count among visual information digital images and videos, which are nothing but a sequence of images called frames, with or without sound. By exploiting these images and videos via deep learning methods, we achieved state-of-the-art results on computer vision's most challenging issues, such as video classification. Thus, machines nowadays can accurately recognize objects and produce a relevant label to what they see. These days, many sectors rely on visual information to accomplish their missions, for instance, the security sector. The security sector gets access to videos and images through different tools. A good case-in-point is surveillance cameras and CCTVs. Sitting ahead of a bank of up to fifteen screens, monitors spend their time perpetually watching live photos coming from surveillance cameras and watching anyone acting suspiciously, and alerting staff or police if they see an act of felony, vandalism, or other crimes. Hence, we can see visual information's usefulness in the security sector. However, the process of monitoring lasts for very long hours, which may cause tiresome and tiredness, leading to ineffectiveness in urgent intervention situations. Consequently, taking full advantage of these precious data and maximizing the rate of perfect-timing interventions implies the automation of the whole process, from observation and identification to alerting. Exclusively, the aim of this project concerns the observation and identification part. To attain that aim, we choose to apply some deep learning performant algorithms in video classification tasks, CNN and RNN. For the rest of this paper, we organize the remaining parts as follows:

- Experiments and results: we discuss in this part the algorithms used and their related experiments.
- Conclusion and future work: we sum up, site the challenges we faced, and bring up our perspective on upcoming work.

2. Materials and Methods

2.1. Databases

This project is based on the UFC Crime Dataset (web, a) which components are :
1900 long and untrimmed real-world surveillance videos. 13 anomaly classes : Abuse, Arrest, Arson, Assault, Road Accident, Burglary, Explosion, Fighting, Robbery, Shooting, Stealing, Shoplifting, and Vandalism. The dataset, when downloaded, is seen to be structured into folders, some of which contain videos belonging to the 13 anomalous classes, while some others are labeled as normal videos (not containing any anomaly), other folders contain useful information for manipulating the data, such as training and test sets.

2.1.1. Video Preprocessing

This task was divided into two parts:

- **Video framing**
  The video classification approach that we will be using will rely on image classification. As mentioned in the introduction, a video is known to be a sequence of still images, which we call frames. The classification algorithm, will then be, in broad, as follows :
  - Take a video as an input
  - Extract the frames from the video
  - Classify the frames of the video, by means of the neural network trained on the frames extracted from labeled videos of the dataset
  - Classify the video based on the label with the highest probability within the frames

  According to that, we will have to extract frames from all the videos of the dataset. To do so, FFmpeg software was used throughout the following steps :
  - Iterating through a loop containing the videos of the dataset
  - Extracting frames from the input video, and deleting the duplicate frames (frames that contain the same information, and/or are highly similar)
  - Putting the extracted frames in a folder named after the input video

- **Labeling the extracted frames**
  Now that we have the frames, we must have a label for each one. As for the videos labeled as ”normal”, it is clear that all the frames would be labeled the same, because there exists no anomaly throughout the whole video. Yet, in the case of videos labeled as ”abnormal”, we will only keep the frames containing the anomaly. A text file amongst the downloaded dataset helps with this step. It lists the sequence of frames that are labeled as ”abnormal” for each video classified as ”abnormal”. Relying on this information, we were able to keep just the abnormal frames from each folder of frames of an abnormal video.

2.1.2. Data Description

After extracting images (frames) with their corresponding labels, we ended up with 74,970 frames in the class ”Anomaly”, and 606,712 frames in the other class, ”normal”. These numbers are proof of a class imbalance of approximately 8.1x.

The two distributions, in Figure 1, are different in terms of the peaks, which represent the number of frames extracted from a given video. There is 79,131 maximum number of frames in the ”Normal” class compared to 6,911 in the ”Anomaly”. This is because in the first class there are videos with a longer duration, which explains the huge gap between the two classes, whereas a longer video will produce many frames.
In order to reduce that maximum gap, we decided to drop, from the ”Normal” class, the videos that produce a number of frames greater than the maximum number in the ”Anomaly” class ($> 6,911$).
As a result, the maximum number of frames in the ”Normal” class is now 5,754 which is close to 6,911 of the ”Anomaly” class. The new distribution of these ”Normal” class frames is shown in Figure 2.

Even though this maximum method did drop some frames, there is still an important gap between the total number of frames in the two classes (”Normal”:308,003 vs ”Anomaly”:74,970). Another method was used to balance the number of frames in each class while making sure that these frames came from videos with the same duration in the two classes, and across all the 13-anomalies in the ”Anomaly” class. in doing so, we kept only the ”Normal” videos that give a number of frames between 540 and 650. This results in 18,440 frames from 33 ”Normal” videos. The same approach was used for the ”Anomaly” class to get the number of frames close enough to the ”Normal” one, resulting in 18,717 frames from the 13-anomalies.

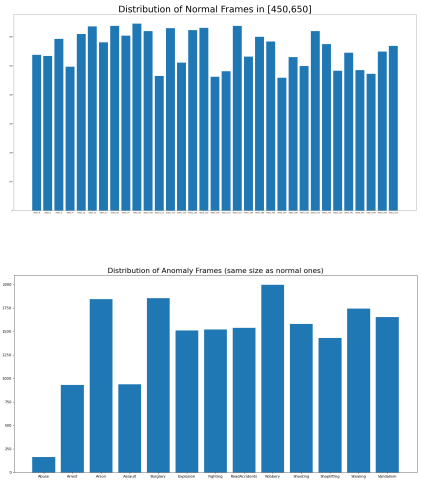

Figure 3. Anomaly and Normal Classes’ New Distributions

As it is shown in Figure 3 we managed to get a similar distribution of the number of frames in each class while making sure to use all the 13-anomalies.

- all dataset: ”Normal” frames = 660,712 vs ”Anomaly” frames = 74,970

- sub-dataset: ”Normal” frames = 18,440 vs ”Anomaly” frames = 18,717

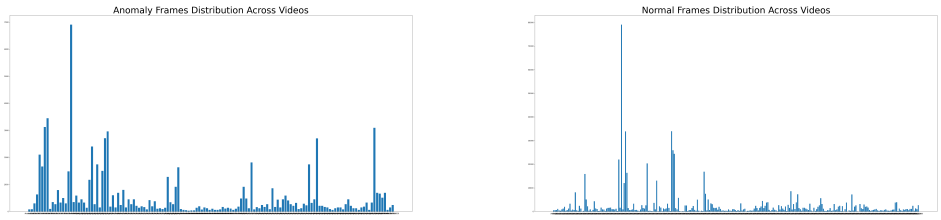

Figure 1. Anomaly and Normal Classes' Distributions

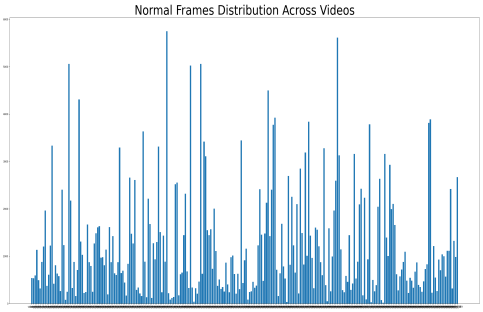

Figure 2. New Normal Frames Distribution

2.2. Algorithms

As mentioned in the Datasets' section that there are 13-classes of anomalies, but for the task at hand, which is anomaly detection, we don't consider this distinction of classes, all the 13-anomalies are supposed to be abnormal events. Consequently, the task becomes a classification problem in which an image or a set of images get classified into "Anomaly" or "Normal". To do so, a Deep Learning (DL) approach was used.

A specific type of DL algorithms that performs very well in tasks where images are presented is the Convolutional Neural Networks (CNN), but one of the challenges of CNNs is the advanced hardware resources (GPU, and high-RAM) required to train the models. Since we can not afford the required hardware we used another approach of DL called Transfer Learning, which is, briefly, a way of using pre-trained models such as ResNet50 and adapting them to the specific task. From the several models that are pre-trained on the ImageNet(web, b) dataset, we choose to work with ResNet50(web, c) and InceptionV3(web, d). We took two approaches for the anomaly-normal classification task. Firstly, we consider it as classic image classification, a video is divided into multiple images and then each image is classified with no information from the other images of the same video. Secondly, we used a Recurrent Neural Network (RNN), specifically the improved version of it which is Long Short-Term Memory (LSTM) that takes the advantage of the time-sequence ordering of a video's images, this is important because an event, such as people fighting, is composed of several actions that could only be detected in an ordered sequence of images instead of only one.

2.2.1. InceptionV3 and Fully Connected Layers

This model is a combination of the pre-trained InceptionV3 model and fully connected layers that consist of 1024 neurons in the first one connected to 512 in the second layer whose connected themselves to 128 neurons in the third layer, and finally, they are connected to two neurons that represent the "Anomaly" and "Normal" classes. As shown in Figure 4.

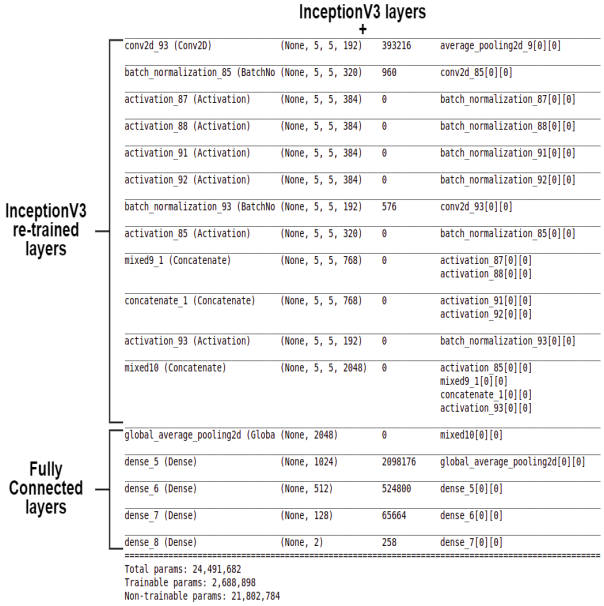

Figure 4. InceptionV3 and FCL Architecture

2.2.2. ResNet50 and Fully Connected Layers

Here we used the same method described in the previous section with one distinction of the base model. The ResNet50 was used here instead of the InceptionV3, and we changed

a bit the structure of the fully connected layers by adding another one with 256 neurons after the second layer (the one with 512 neurons), as presented in Figure 5.

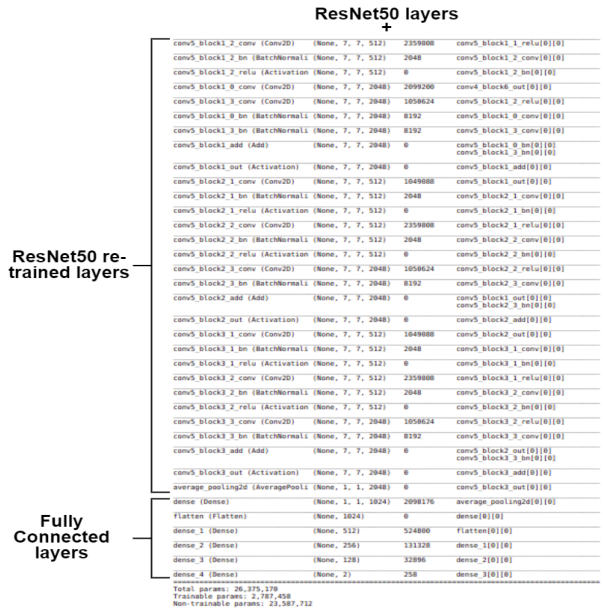

Figure 5. ResNet50 and FCL Architecture

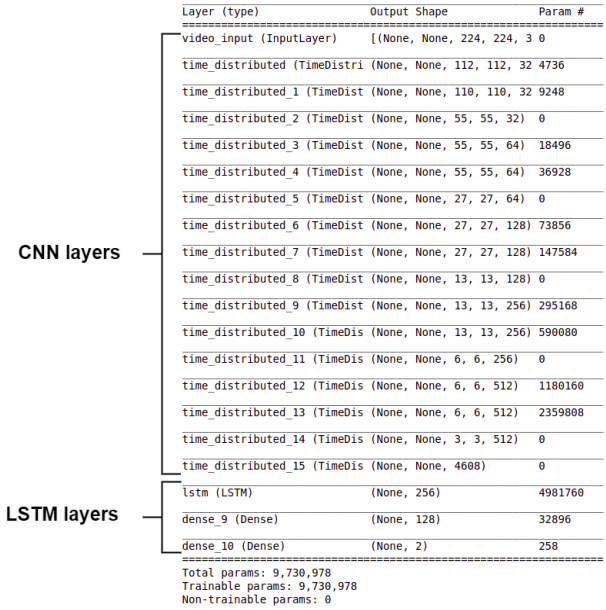

Figure 6. CNN and LSTM Architecture

These two models do not consider a video as a sequence of time-ordered images, they perform regular image classification. This is a challenging problem because an event, in our definition, could be a series of images in this case the above-described methods could not take advantage of this very important information. The following methods will try to solve this problem.

2.2.3. Convolutional Neural Network and Long Short-Term Memory

The RNN algorithms are widely used for Time Series (TS) data, data that are indexed in time order. A video, on the other hand, could be thought of as a TS, it contains time-ordered frames.For this reason, and the reason that CCTV videos are generally longer, we used the Long Short-Term Memory artificial neural network that is able to process an entire sequence of data points, in this case, the video images. To start with, we combined the power of LSTM with the standard CNN. The Figure 6 shows the architecture that was used.

2.2.4. InceptionV3 and Long Short-Term Memory

For the earlier used models (regular image classification), as a result, the InceptionV3 model performed well in comparison to ResNet50, That's why we wanted to see if a combination of InceptionV3 and LSTM will further improve the accuracy of classification. The Figure 7 shows the structure of this combination, we used InceptionV3 to extract "rich" features from a sequence of images and then feed them to LSTM, hoping that it will perform better than with the standard CNN.

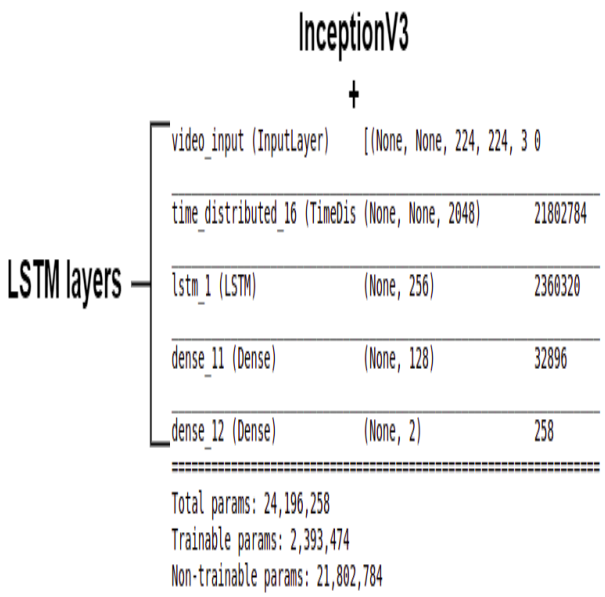

Figure 7. InceptionV3 and LSTM Architecture

In the next section, we will discuss the results of all these methods.

3. Results and Discussion

Since there are two types of datasets, as described in the Datasets' section, the all-dataset and the sub-dataset, the experiments ran on both of them while varying the model's parameters such as the number of epochs and batch size in order to improve its performance.

- sub-data: 18,440(normal-frames) 18,717(anomaly-frames)
- all-data: 660,712(normal-frames) 74,970(anomaly-frames)

For all the below experiments the images are resized into the shape (244,244,3) and normalized by 255. These two videos,"Fighting003" with its 1282 anomaly frames and "video_267" with its 1327 normal frames, were used to test the performance of each model as unseen data because they have nearly the same number of frames.

3.1. InceptionV3 and Fully Connected Layers

3.1.1. Model's Experiments

The Table 1 summarizes the experiments.
For a few epochs and small batch size (the initial learning rate could affect the performance, but it's unlikely to happen in this case) the model was not able to learn anything, by increasing both of them at the same time and still training on the sub-data the model accuracy achieved about 96% and 99.98% in the training and validation sets respectively. It seems this is the best performant model on the sub-dataset. Using the same parameters of this best-performant model we re-trained it on the all-dataset (the imbalanced one) and surprisingly the model did well by achieving 99.7% training accuracy and 99.99% validation accuracy. Another comparison metric to use in this case is the computational time required to train the model, and as it's obvious from the table the best-performant model on the sub-dataset took only half an hour, unlike the other one that took 6 hours. This is clearly caused by the data size.

3.1.2. Testing The Models on Unseen Data

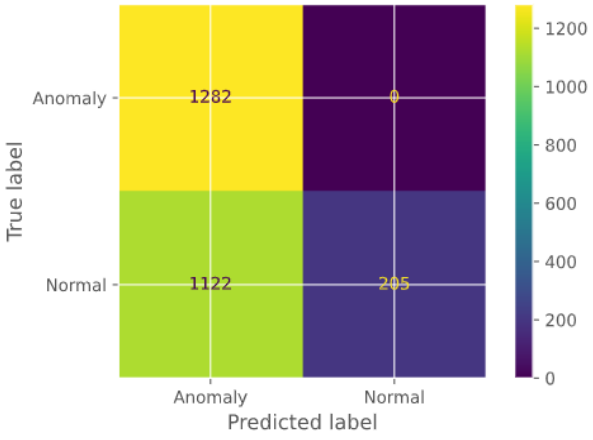

Figure 8. InceptionV3: First Best-Performant Model (sub-data) Test on Unseen Data

This model classified all anomaly images as an anomaly, as shown in Figure 8, but it had difficulties with the normal images as it considers only 205 of 1327 as normal. This could be okay as long as it does not do the opposite because we want it to detect the suspicious normal images.

Number of epochs	Initial learning rate	Batch size	Computational time (minutes)	Accuracy	Dataset
52	$1e-5$	62	31	0.412	sub-data
25	$1e-5$	32	32	0.563	sub-data
100	$1e-1$	256	28	*0.960	sub-data
100	$1e-1$	256	360(6hours)	*0.997	all-data

Table 1. InceptionV3 and FCL Experiments

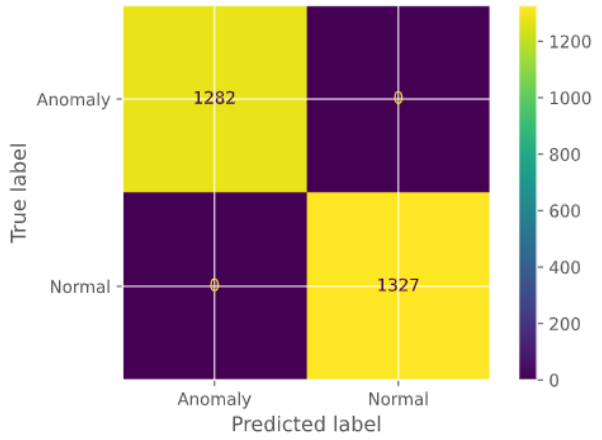

Figure 9. InceptionV3: Second Best-Performant Model (all-data) Test on Unseen Data

When we see the model that was trained on all-data, there is no mistake as depicted in Figure 9, it was able to classify every image to its corresponding class. But this could not be true in all cases, because as we mentioned we want to detect some suspicious normal images so that we are sure we will not miss any abnormal event happening in real life.

3.2. ResNet50 and Fully Connected Layers

3.2.1. Model's Experiments

The Table 2 summarizes the experiments.
Same as inception v3 model, reaching a high accuracy implies increasing both of number of epochs and batch size. For sub-data, the model accuracy achieved about 82.65% and 30.24% in the training and validation sets, respectively. Using the same parameters of the previous model, we retrained it on the all-dataset and the model achieved 98.53% in training accuracy and 99.97% in validation accuracy. As for the computational time, the best-performant model on the sub-data took 2h33min, being few minutes apart from other models if not the same timing, while the computational time of the best-performant model on the all-data reached 24h23min.

3.2.2. Testing The Models on Unseen Data

Testing the two "best" performant models on the sub-data and all-data using the above-mentioned two videos.

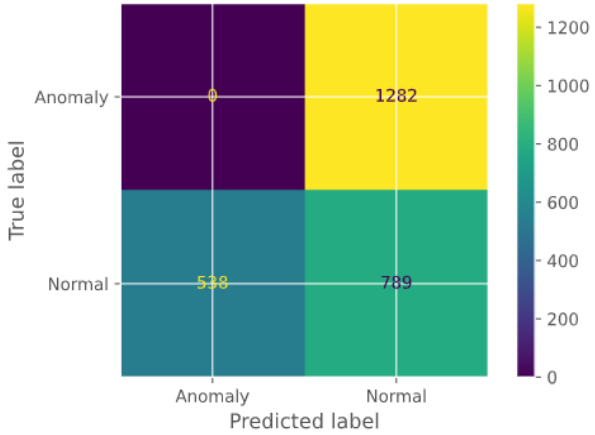

Figure 10. ResNet50: First Best-Performant Model (sub-data) Test on Unseen Data

As shown in Figure 10, this model classified all the anomaly images into the "Normal" class which is totally false, and also was not able to classify the normal images, they are divided by 538 images as "Anomaly" and 789 as "Normal". So basically, this model, ResNet50+FC on sub-data, was not good in any way.

Number of epochs	Initial learning rate	Batch size	Computational time (minutes)	Accuracy	Dataset
52	$1e-6$	64	140(2.33h)	*0.8265	sub-data
100	$1e-1$	256	61	0.5013	sub-data
100	$1e-6$	256	135(2.25h)	0.6496	sub-data
100	$1e-6$	512	134(2.24h)	0.6281	sub-data
200	$1e-6$	64	47	0.5951	sub-data
200	$1e-6$	128	140(2.33h)	0.7990	sub-data
52	$1e-6$	64	1455(24.23h)	*0.9853	all-data
100	$1e-6$	256	1353(22.5h)	0.9641	all-data

Table 2. ResNet50 and FCL Experiments

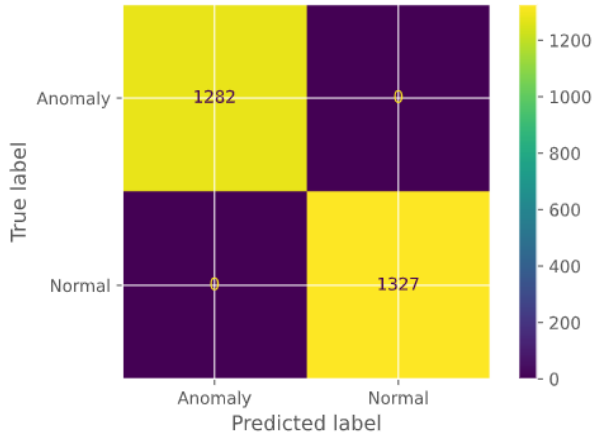

Figure 11. ResNet50: Second Best-Performant Model (all-data) Test on Unseen Data

As with the InceptionV3 the trained model on all-data that classified every image correctly, this ResNet50 trained on all-data also classified normal images and anomaly images correctly, as presented in Figure 11.

For the remaining algorithms, because they consider a sequence of images as a data point, another parameter was introduced called "Sequence length" that defines the number of samples in a sequence. The "Sequence length" must not exceed 150sample because of the hardware limitations.

3.3. Convolutional Neural Network and Long Short-Term Memory

The Table 3 below summarizes the experiments.

3.4. InceptionV3 and Long Short-Term Memory

The Table 4 summarizes the experiments.

The performance tests of the models, CNN and Inception with LSTM, were not displayed as their accuracies were very low. In fact, we intended to work with Recurrent Neural Networks to take advantage of their input type; a sequence of images (frames), instead of one single frame. Yet, due to the high number of frames in our disposal, and the hardware limitations, we couldn't upload the whole sequence as an input. The latter, could be a probable reason behind the low performance of models that involve LST.

Number of epochs	Initial learning rate	Batch size	Sequence length	Computational time (minutes)	Accuracy	Dataset
10	$1e-6$	16	10	143(2.37h)	0.4747	sub-data
10	$1e-6$	16	100	79(1.31h)	0.5360	sub-data
200	$1e-6$	8	150	75(1.24h)	0.5338	sub-data
100	$1e-6$	32	30	81(1.34h)	0.4889	sub-data
200	$1e-6$	25	50	81(1.34h)	0.5540	sub-data

Table 3. CNN and LSTM Experiments

Number of epochs	Initial learning rate	Batch size	Sequence length	Computational time (minutes)	Accuracy	Dataset
10	$1e-6$	16	10	105(1.75h)	0.4936	sub-data
10	$1e-6$	16	100	65(1.07h)	0.5710	sub-data
25	$1e-6$	32	100	65(1.07h)	0.4991	sub-data
100	$1e-6$	16	100	65(1.07h)	0.5398	sub-data
100	$1e-6$	8	150	63(1.04h)	0.4721	sub-data
200	$1e-6$	25	50	63(1.10h)	0.4225	sub-data

Table 4. InceptionV3 and LSTM Experiments

4. Conclusion and Future Work

The process of monitoring CCTV footage and alerting staffers or officers is usually done by humans, which might cause a non-efficiency in immediate interventions due to its cumbersomeness. To tackle the problem, we aim to build an AI-based system to alert security officers if an abnormality was detected. Hence, we started by implementing computer vision algorithms to detect abnormal behavior.

As for now, we have run only a couple of experiments on the available dataset, and some of them haven't shown great performance. Thus, we intend to improve these experiments, from preparing the data to evaluating the models, in order to achieve better results and reliable classification. Expressly, we will improve the hardware used to be able to run RNN based models on bigger volumes of data and with a variation of parameters. Then, we aim to move on to the next steps of the project that rely on sending automatic alerts to complement authorities after the anomaly detection. In sum, we are devoted to the implementation of all details that will grant more fluidity and better performance of this project.

References

Ufc crime dataset, a. URL https://webpages.uncc.edu/cchen62/dataset.html.

Imagenet, b. URL https://www.image-net.org/.

Resnet50: Deep residual learning for image recognition, c. URL https://arxiv.org/abs/1512.03385.

Inceptionv3: Rethinking the inception architecture for computer vision, d. URL https://arxiv.org/abs/1512.00567.