

Image Classification using Convolutional Neural Networks to detect suspicious activity in the online proctoring space

Shravya Katukuri, Kirti Tiwari, Ritik Gupta

INTRODUCTION

Motivation for the problem:

Facial recognition and activity detection in the area of online video and image surveillance is becoming a more and more popular and accessible biometric method to enhance computer applications and detect various patterns that will help in anomaly detection in case of people and events, suspicious e-Crime, terrorist account detection, ATM and bank fraud, and intelligent video surveillance systems especially because of factors like Social Media trends, increasing number of eLearning courses and exams, and distance learning management systems to name a few.

This trend motivated us to look deep into this area and we happened to stumble upon a very interesting and useful use case in the eLearning industry – to recognise and detect suspicious activity in online exams that are remotely proctored.

Presentation of our problem:

This report focuses predominantly on the eLearning industry as it is the future of learning with the global eLearning market now capping at around \$100 billion plus. With the rapid growth, the eLearning industry has created needs for various supporting technologies, one of which is the virtual proctoring space that is gaining a lot of significance. If it weren't for eLearning, then online video proctoring wouldn't have become mainstream but given what's at stake for online course programs, the demand for such a service has grown at a fast pace.

Over the past few years, online exams have become popular because of their flexibility, usability, and user-friendliness. In term of online examinations, the abnormal behaviour monitoring of the examinee during the examination is one of the major challenges. The traditional monitoring programs mainly focus on the identity of testers and lack the effective identification of abnormal behaviours of testers. Faced with the monitoring of abnormal behaviours in online examinations, this paper proposes to obtain the information of the examinee's head posture and eye movement through the webcam and to distinguish the abnormal behaviour of the examinees into suspicious/clean activity during the online examination to facilitate smooth online/virtual proctoring with minimal human effort or intervention. Having a machine learning model in place that will detect and flag suspicious activity based on a person's hand and eye movements would help proctors review and evaluate that person's authenticity while taking the exam and improve the effectiveness of proctoring while reducing the human effort required to do the same.

METHODOLOGY

Problem Statement:

In order to detect suspicious activity based on head and eye movements, we divided our problem into 2 parts each of which has varying levels of complexity.

Part 1: We captured images of people in various positions using our laptop's webcam and label instances as clean or suspicious accordingly.

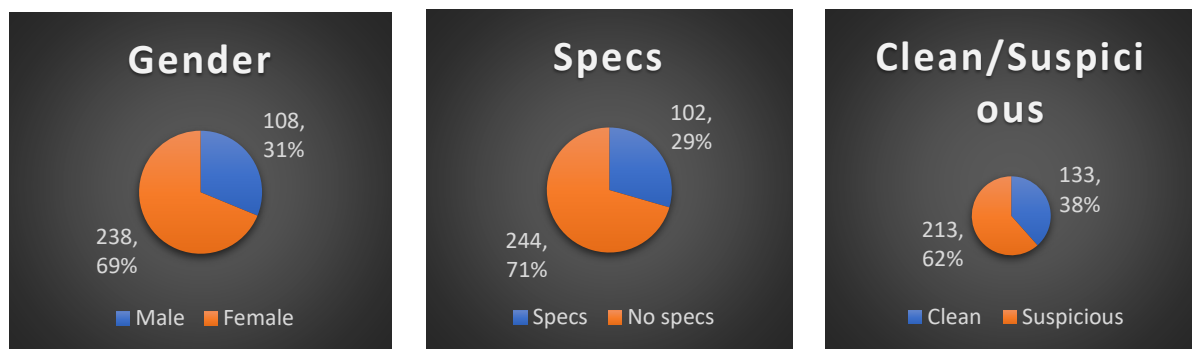
Part 2: We captured 3 to 10 seconds video segments that record a person's movements while giving an online exam and runs it through the code which labels each second of the video as suspicious or clean.

Data and Libraries used:

Data Collection:

We created our own dataset by trying to replicate the online exam scenario and captured different images and videos of friends, batchmates and family. For determining the suspicious activity based on the head movement, we have taken images of people directly looking into the computer screen and marked them as clean while for the instances where head is away from the screen/turning left or right/turning at some angle, we marked them suspicious. We tried to take images of different people, so that the model doesn't get trained on one particular person. As of now, we have 279 images in total which will be used to train our model and 25% of the same will be used for validation. For the test set, we have taken images of a person who is neither a part of training nor validation so that we can vouch for the model's unbiased predictions. Similarly, for the videos, we shot small videos ranging from 3 to 10 seconds and captured the movements of the head like we did for images. Few of them are a mixture of both the scenarios (suspicious and clean).

Demographics of our data:



- **Total Images in Training and Validation** : 279
- **Total Images in Testing** : 67
- **Total Videos for Testing** : 47

NOTE: We did not use any external datasets from Kaggle, UCI or any other repositories.

Libraries used for processing data: The libraries used in this project were Numpy, Pandas, OpenCV, Matplotlib, Scikitlearn, Tensorflow and Keras.

Levels of complexity:

We decided to build and train model based on the simplest level first and then carve our way up by increasing the complexities. So, we did classification on the images first and then on videos.

- **Part 1-Images:**
 - **Level-1:** We start off with classifying extreme head movements as suspicious and no head movement as clean i.e., a person would be labelled suspicious if he/she is not looking into the screen and looking away.
 - **Level-2:** For the second level, we increase complexity by trying to classify subtly angled head movements (ex. At 30°, 45°, etc) as suspicious and no movement as clean.
 - **Level-3:** For the third level, we try capturing eye movements of the person to classify their activity as suspicious or clean.
- **Part 2-Videos:**
 - **Level-1:** We classify the entire video as suspicious or clean based on a pre-defined cut-off or threshold value (in our case we used 0.5 as cutoff)
 - **Level-2:** We classify the videos by flagging or extracting parts/intervals of the video as suspicious or clean.

EXPERIMENTS AND STORIES

As already stated above, we created different versions of our code to handle the multiple levels of complexity. This section charts our journey from the initial baseline to our final model.

IMAGES

Levels-1 and 2 (Head Movement) :

Version-1:

We initially started with 101 images that comprised of images taken from different mobile phones, laptops and Google images. Since their sizes would be heterogeneous, we re-sized these images into a size of 200*200 pixels in our code.

Procedure:

- Loaded all the images in one folder (which was both our training and validation set) and labelled them as clean_xxxx and suspicious_xxxx depending on the head movements.
- Loaded a few images in a test folder which was later renamed to “predict” and added 19 images of a new person who was not part of the above set.
- In the code, we applied logic to segregate the images from the above set to 2 different folders - one each for training and validation. Both these folders were further divided into 2 subfolders namely clean and suspicious. 25% of these images are automatically moved to the validation set, as per the logic applied.
- We also applied a logic to delete and recreate the subfolders in run-time.

```

# organize dataset into a useful structure
from os import makedirs
from os import rmdir
from os import listdir
import shutil
from shutil import copyfile
from random import seed
from random import random
from random import shuffle
# create directories
dataset_home = directory+'dataset_sus_vs_clean/'
subdirs = ['train/', 'valid/']
for subdir in subdirs:
    # create label subdirectories
    labeldirs = ['sus/', 'clean/']
    for labldir in labeldirs:
        newdir = dataset_home + subdir + labldir
        makedirs(newdir, exist_ok=False)
# seed random number generator
seed(1)
# define ratio of pictures to use for validation
val_ratio = 0.25
# copy training dataset images into subdirectories
src_directory = TRAIN_DIR
for file in listdir(src_directory):
    src = src_directory + '/' + file
    dst_dir = 'train/'
    if random() < val_ratio:
        dst_dir = 'valid/'
    if file.startswith('sus'):
        dst = dataset_home + dst_dir + 'sus/' + file
        copyfile(src, dst)
    elif file.startswith('clean'):
        dst = dataset_home + dst_dir + 'clean/' + file
        copyfile(src, dst)

```

We created the below model to begin with and applied it on 101 images-76 in training and 25 in validation.

Used Number of epochs = 20, steps per epoch = 25 in training and 30 in validation; Got a training accuracy of 61-62% and a validation accuracy of 36%.

(Attaching the corresponding code and output screenshots for further reference)

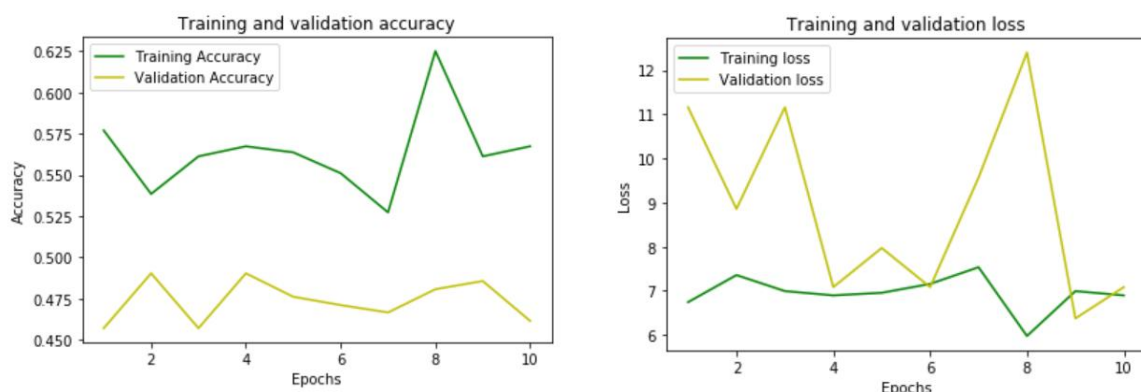


Base Version.docx

Version-2:

Added a few more images and removed few from the old set along with modifying parameters like number of epochs, batch size, steps per epoch etc. Validation Accuracy improved and rose to 47% but not too much. Also, we faced issue of overfitting here.

Attached are the plots which were generated for Training/Validation Accuracy and Loss.



(Attaching the corresponding code and output screenshots for further reference)

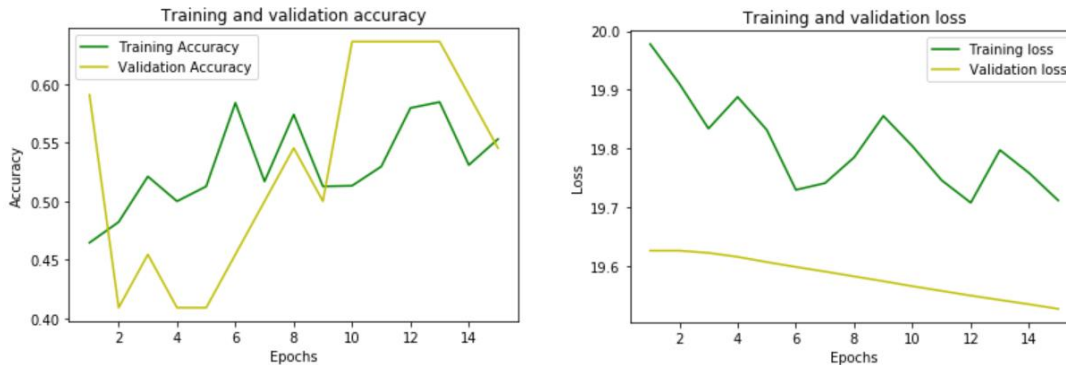


Version 2.docx

Version-3:

Tweaked the model further using different hidden layers and optimizers (Adam, nesterov_adam, sgd, etc), applied batch normalization and changed values in the ImageDataGenerator function. Using all these, we were able to increase validation accuracy up to 63%.

Attached are the plots which were generated for Training/Validation Accuracy and Loss.



(Attaching code and output screenshots for more reference or click [here](#))

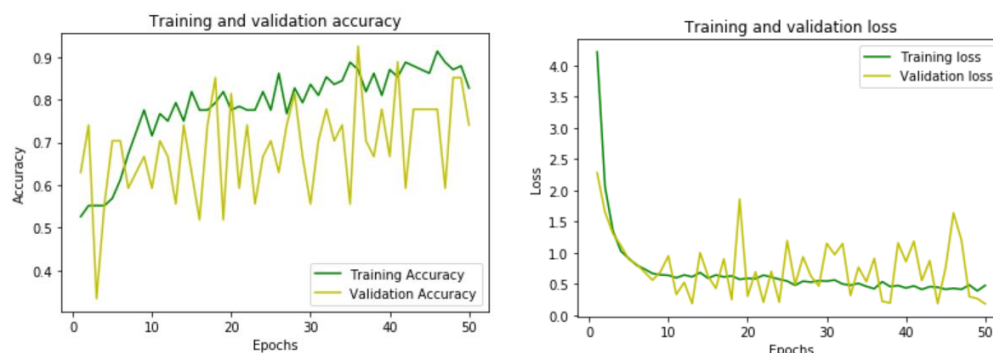


Version-4:

After a series of tweaks and subsequent testing, we removed batch normalization, retained only 4 convolutional layers and 3 dense layers. In addition, also modified the optimizer with “rmsprop” which, in our case proved to be the best. Removed the attributes - rotation_range, width_shift_range, height_shift_range from ImageDataGenerator. Revisited and updated labelling of a few images.

By changing the number of epochs to 50 and taking batch sizes of 10 and 3 for training and validation respectively, we were able to achieve training and validation accuracies of 86% - 90% in a few epochs. But there were huge variations in the validation accuracy that varied from 50- 90% and this was the best accuracy we got till that point then.

Attached are the plots which were generated for Training/ Validation Accuracy and Loss.



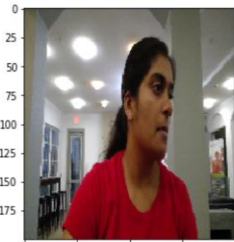
(Attaching code and output screenshots for more reference)



Version 4.docx

Additionally, in this version, we were able to properly capture extreme head movements for some of the test records; but for the slight/angled movements (our level-2), the model was not able to capture the activity accurately.

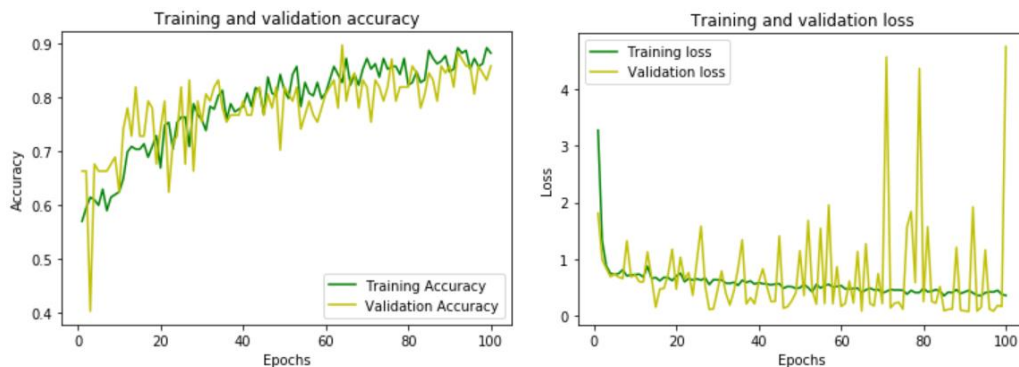
```
plt.imshow(prediction_set[i][0][0])
print(predictions[i], "This image is predicted as", "Suspicious" if predictions[i] > 0.5 else "Clean")
[0.99999297] This image is predicted as Suspicious
```



Version-5:

We added more pictures and retained the code from the previous versions, so that the model gets trained on more data. We were able to get better accuracy for validation. Also increased the number of epochs to 100.

Attached are the plots which were generated for Training/ Validation Accuracy and Loss.



(Attaching code and output screenshots for more reference)

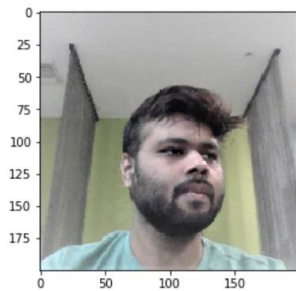


Version 5.docx

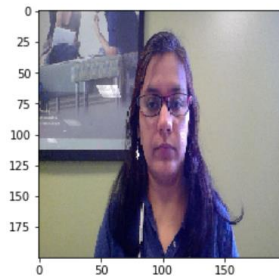
Model's Predictions for Version 5 (Our Final Version) :

```
plt.imshow(prediction_set[i][0][0])
print("This image is predicted as", "Suspicious" if predictions[i] > 0.5 else "Clean")
```

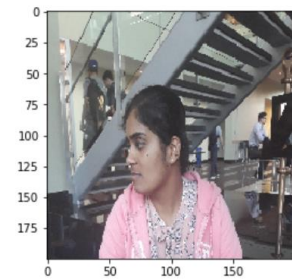
This image is predicted as Suspicious



This image is predicted as Clean



This image is predicted as Suspicious



Level-3:

Our initial plan was to incorporate this level along with the head movement. But, before doing so, we wanted to confirm that the model for eye movement was giving us the desired results, so we created separate Jupyter Notebooks, in order to test the same.

We tried 2 approaches here :

- Using our existing code for head movement
- Using an object detection algorithm

Using our existing code for head movement:

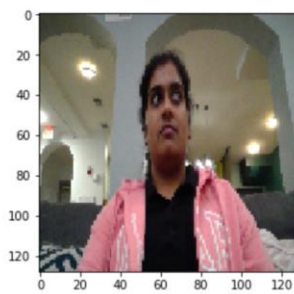
We were able to create and train the baseline model but were not able to get good accuracies due to paucity of time and data. Our model was able to predict with a training accuracy of around 60% and a validation accuracy of around 50-53% (which remained constant and was not showing any improvements). Below is the screenshot where we got the best results compared to other runs.

```
classifier.fit_generator(training_set,
                        steps_per_epoch = 30,
                        epochs = 5,
                        validation_data = valid_set,
                        validation_steps = 10)
```

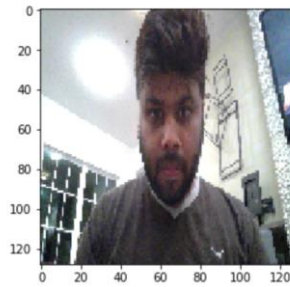
```
Epoch 1/5
30/30 [=====] - 91s 3s/step - loss: 0.6734 - accuracy: 0.5794 - val_loss: 0.6445 - val_accuracy: 0.761
9
Epoch 2/5
30/30 [=====] - 84s 3s/step - loss: 0.6288 - accuracy: 0.6635 - val_loss: 0.5944 - val_accuracy: 0.761
9
Epoch 3/5
30/30 [=====] - 75s 3s/step - loss: 0.5950 - accuracy: 0.6889 - val_loss: 0.5535 - val_accuracy: 0.714
3
Epoch 4/5
30/30 [=====] - 74s 2s/step - loss: 0.5655 - accuracy: 0.6984 - val_loss: 0.5133 - val_accuracy: 0.857
1
Epoch 5/5
30/30 [=====] - 78s 3s/step - loss: 0.5255 - accuracy: 0.7429 - val_loss: 0.4627 - val_accuracy: 0.809
5
```

```
plt.imshow(prediction_set[i][0][0])
print("This image is predicted as", "sus" if predictions1[i] > 0.5 else "clean")
```

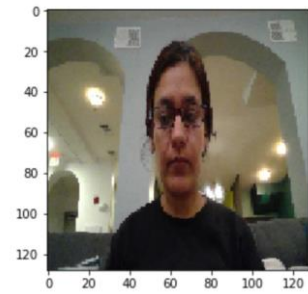

This image is predicted as sus



This image is predicted as clean

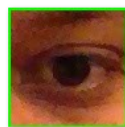


This image is predicted as sus

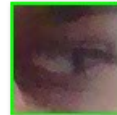


Using an Object Detection Algorithm:

In this approach, we tried extracting eyes of a person as an object using the algorithm and trained the model based on that object. We were able to extract eye part and load them into their respective folders. Below are some screenshots of eyes which our code was able to extract.

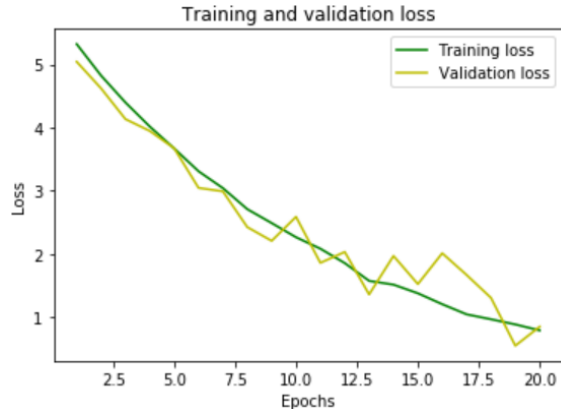
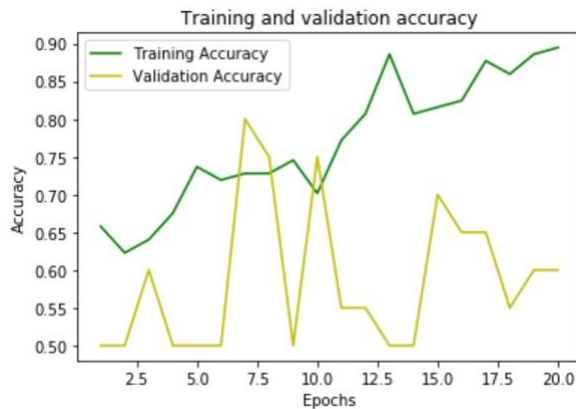


clean.22



suspicious.242

Below are the training and validation accuracies and we also used it to predict data from our test set. For some of them it gave correct results but still we wanted to spend some time to explore issues and fine tune the model further.



(Attaching code and output screenshots for more reference)



Eye_Movement.docx

VIDEOS

Once the level 1 in images was completed, we moved on to videos and predicting their outcomes. As stated earlier, we planned to do it in 2 ways:

- 1) Marking the whole video as suspicious.
- 2) Mark/ Flag the intervals in the video which were found to be suspicious

The approach taken was to split the video into frames for every 0.5 seconds using the Python code. After that, we stored the images captured in a separate folder and used the model already trained on the images to predict the classification of each frame in the videos.

We originally planned on working in 2 levels as stated above but while working on the code, we clubbed both the ideas together and generated the output which contained information about the individual frames as well as the entire video together.

A few additional steps required for videos are:

- 1) Creating an additional folder during runtime to save the images extracted from videos.
- 2) Extracting images at every 0.5 seconds from the video and renaming them by appending 1,2,3, etc at the end of the video's name, like video_xxx_1, video_xxx_2, etc.
- 3) Applying prediction model based on the previously trained data.
- 4) Applying logic so that the output of all the images pertaining to a video gets displayed as one single set.
- 5) Counting the number of frames in the video, along with the suspicious ones and marking the whole video as clean/suspicious based on the ratio (suspicious/total). If this ratio > 0.5, then it's suspicious, else it's clean.

We ran the model for 150 epochs and tested some of the videos manually to see if the results are as expected or not. Results of a few videos are tabulated as follows:

Video_021: Our conclusion was that the entire video should be treated as Suspicious and as per the logic applied in the code, it was being marked the same.

Video Frames	Model Prediction	Our Prediction	Result
Video021_1.jpg	Clean	Suspicious	FALSE
Video021_2.jpg	Suspicious	Suspicious	TRUE
Video021_3.jpg	Suspicious	Suspicious	TRUE
Video021_4.jpg	Suspicious	Suspicious	TRUE
Video021_5.jpg	Suspicious	Suspicious	TRUE
Video021_6.jpg	Suspicious	Suspicious	TRUE

Suspicious Count of the model	Total Count	Model Ratio	Suspicious Count (Our Prediction)	Expected Ratio
5	6	0.8333	6	1.0000

```
Video021_1.jpg Clean
Video021_2.jpg Suspicious
Video021_3.jpg Suspicious
Video021_4.jpg Suspicious
Video021_5.jpg Suspicious
Video021_6.jpg Suspicious
Summary: Suspicious_Count- 5 Total_Count- 6 Ratio- 0.8333 Overall Prediction- Suspicious
```

Video_007: There was no suspicious activity or any head movement in the video and the student was looking straight into the computer screen; so, we marked it Clean. Based on the below

output generated by the model, it also classified the individual frames as well as the entire video as clean.

Video Frames	Model Prediction	Our Prediction	Result
video007_1.jpg	Clean	Clean	TRUE
video007_2.jpg	Clean	Clean	TRUE
video007_3.jpg	Clean	Clean	TRUE
video007_4.jpg	Clean	Clean	TRUE
video007_5.jpg	Clean	Clean	TRUE
video007_6.jpg	Clean	Clean	TRUE
video007_7.jpg	Clean	Clean	TRUE

video007_1.jpg Clean
video007_2.jpg Clean
video007_3.jpg Clean
video007_4.jpg Clean
video007_5.jpg Clean
video007_6.jpg Clean
video007_7.jpg Clean

Summary: Suspicious_Count- 0 Total_Count- 7 Ratio- 0.0 Overall Prediction- Clean

Video_045: Our conclusion was that the video should be marked suspicious as the person was looking outside the screen for most of the time. The model also predicted accordingly.

Video Frames	Model Prediction	Our Prediction	Result
video045_1.jpg	Suspicious	Suspicious	TRUE
video045_2.jpg	Suspicious	Suspicious	TRUE
video045_3.jpg	Suspicious	Suspicious	TRUE
video045_4.jpg	Suspicious	Suspicious	TRUE
video045_5.jpg	Suspicious	Suspicious	TRUE
video045_6.jpg	Suspicious	Clean	FALSE
video045_7.jpg	Suspicious	Clean	FALSE
video045_8.jpg	Suspicious	Clean	FALSE
video045_9.jpg	Suspicious	Suspicious	TRUE
video045_10.jpg	Clean	Suspicious	FALSE
video045_11.jpg	Clean	Suspicious	FALSE
video045_12.jpg	Suspicious	Suspicious	TRUE
video045_13.jpg	Suspicious	Clean	FALSE
video045_14.jpg	Suspicious	Clean	FALSE
video045_15.jpg	Suspicious	Clean	FALSE
video045_16.jpg	Suspicious	Suspicious	TRUE
video045_17.jpg	Suspicious	Suspicious	TRUE
video045_18.jpg	Suspicious	Suspicious	TRUE
video045_19.jpg	Suspicious	Clean	FALSE
video045_20.jpg	Suspicious	Clean	FALSE
video045_21.jpg	Suspicious	Clean	FALSE

Suspicious Count of the model	Total Count	Model Ratio	Suspicious Count (Our Prediction)	Expected Ratio
19	21	0.9048	12	0.5714

video045_18.jpg Suspicious
 video045_19.jpg Suspicious
 video045_2.jpg Suspicious
 video045_20.jpg Suspicious
 video045_21.jpg Suspicious
 video045_3.jpg Suspicious
 video045_4.jpg Suspicious
 video045_5.jpg Suspicious
 video045_6.jpg Suspicious
 video045_7.jpg Suspicious
 video045_8.jpg Suspicious
 video045_9.jpg Suspicious

Summary: Suspicious_Count- 19 Total_Count- 21 Ratio- 0.9048 Overall Prediction- Suspicious

Video_003: There was no suspicious activity or any head movement in the video, so we marked it as Clean. The model also predicted accordingly.

Video Frames	Model Prediction	Our Prediction	Result
video003_1.jpg	Clean	Clean	TRUE
video003_2.jpg	Clean	Clean	TRUE
video003_3.jpg	Clean	Clean	TRUE
video003_4.jpg	Clean	Clean	TRUE
video003_5.jpg	Clean	Clean	TRUE
video003_6.jpg	Clean	Clean	TRUE
video003_7.jpg	Clean	Clean	TRUE
video003_8.jpg	Clean	Clean	TRUE
video003_9.jpg	Clean	Clean	TRUE
video003_10.jpg	Clean	Clean	TRUE
video003_11.jpg	Clean	Clean	TRUE
video003_12.jpg	Clean	Clean	TRUE

video002_9.jpg Clean

Summary: Suspicious_Count- 0 Total_Count- 14 Ratio- 0.0 Overall Prediction- Clean

Video_015: We concluded this video as Suspicious. The model also predicted accordingly.

Video Frames	Model Prediction	Our Prediction	Result
Video015_1.jpg	Suspicious	Clean	FALSE
Video015_2.jpg	Suspicious	Clean	FALSE
Video015_3.jpg	Suspicious	Suspicious	TRUE
Video015_4.jpg	Suspicious	Suspicious	TRUE
Video015_5.jpg	Suspicious	Suspicious	TRUE
Video015_6.jpg	Suspicious	Suspicious	TRUE
Video015_7.jpg	Suspicious	Suspicious	TRUE
Video015_8.jpg	Suspicious	Suspicious	TRUE

Video015_9.jpg	Suspicious	Suspicious	TRUE
Video015_10.jpg	Suspicious	Suspicious	TRUE
Video015_11.jpg	Suspicious	Suspicious	TRUE
Video015_12.jpg	Suspicious	Suspicious	TRUE
Video015_13.jpg	Suspicious	Suspicious	TRUE
Video015_14.jpg	Suspicious	Suspicious	TRUE

Suspicious Count of the model	Total Count	Model Ratio	Suspicious Count (Our Prediction)	Expected Ratio
14	14	1.0000	12	0.8571

Summary: Suspicious_Count- 14 Total_Count- 14 Ratio- 1.0 Overall Prediction- Suspicious

Video_048: We concluded that the whole video should be Suspicious, and the model also predicted majority of the frames likewise, but overall prediction came out as Clean. It still gave decent values though.

Video Frames	Model Prediction	Our Prediction	Result
video048_1.jpg	Clean	Clean	TRUE
video048_2.jpg	Clean	Clean	TRUE
video048_3.jpg	Clean	Clean	TRUE
video048_4.jpg	Clean	Clean	TRUE
video048_5.jpg	Clean	Clean	TRUE
video048_6.jpg	Clean	Suspicious	FALSE
video048_7.jpg	Suspicious	Suspicious	TRUE
video048_8.jpg	Suspicious	Suspicious	TRUE
video048_9.jpg	Clean	Suspicious	FALSE
video048_10.jpg	Clean	Clean	TRUE
video048_11.jpg	Clean	Clean	TRUE
video048_12.jpg	Suspicious	Suspicious	TRUE
video048_13.jpg	Suspicious	Suspicious	TRUE
video048_14.jpg	Suspicious	Suspicious	TRUE
video048_15.jpg	Suspicious	Suspicious	TRUE
video048_16.jpg	Suspicious	Suspicious	TRUE
video048_17.jpg	Suspicious	Suspicious	TRUE
video048_18.jpg	Clean	Clean	TRUE
video048_19.jpg	Clean	Clean	TRUE
video048_20.jpg	Clean	Clean	TRUE

Suspicious Count of the model	Total Count	Model Ratio	Suspicious Count (Our Prediction)	Expected Ratio
8	20	0.4000	10	0.5000

Summary: Suspicious_Count- 8 Total_Count- 20 Ratio- 0.4 Overall Prediction- Clean

(Attaching code screenshots for more reference and the videos for the above examples)



Videos.docx



video_testing_results.zip

ADDITIONAL EXPERIMENTS

In this section, we list out a few things that we experimented with but did not come up with expected results. Although we were able to achieve validation accuracies of up to 80% and higher, there were some cases where we faced minor setbacks:

- **Using early stopping:** We applied early stopping in the code and initially we got good results but in the later runs it ran into a few issues. It got stuck at one validation accuracy (like .7096) and did not change even after 50 or 100 epochs. Unfortunately, we did not capture a screenshot of that issue. Also, any of restarting the Kernel, disabling early stopping function and removing it from the code altogether did not work, so we had to re-write that section of the code again to get the issue fixed. So, in the later versions, we did not use early stopping.
- **Enabling Height and width options in ImageDataGenerator:** Using this feature did not work in our favour and the accuracies achieved were not much. Performance also was slow. One possibility might be because our dataset is small, enabling these options might not have worked out for us. Maximum accuracy achieved was ranging between 64-70%. Accuracy started picking up > 70% in the last 20 of the 100 epochs.
- **Overfitting and Underfitting:** We experimented with different number of neurons, hidden layers, optimizers, number of epochs, batch sizes, etc to try and increase the accuracy but experienced problems of overfitting and underfitting in them. Hence, discarded those changes.

TESTING & EVALUATION

As part of our testing, we took different scenarios but highlighted just a few here to show the gist of the model predictions. Based on the data collected till this point and the outputs generated, we can say that our model is not showing any biasness and predicting with respect to head movements (in images and videos) solely. We have also manually labelled all the images separately and cross-checked with the model's predicted labels.

The different Scenarios we considered are:

- Added a few people in our testing data who were not part of our training
- Used backgrounds in our images/videos in testing data that were different from our training sample.
- Tested the model to check its effectiveness in predicting people with different external attributes (spectacles in our case).
- Also tested the model to check if it is biasing based on gender.

Methods of Evaluation:

We have used the following methods of evaluation:

- Checked validation accuracy and compared it with training accuracy (to check for overfitting),

- Ideally the model's predictions should be universal and independent of a few factors. To evaluate these factors, we used fairness metrics for the following:
 - If the model is biased towards gender (i.e., labelling a greater number of images of a particular gender as suspicious),
 - If the model is affected by external attributes of the person (spectacles in our case),
 - If the model is biased towards new and old testing instances (whether it is predicting new and never seen instances similar to the old ones),
 - If the model is affected by different backgrounds in the images/videos.

(Please find the attached excel sheet for the respective calculations).



Data.xlsx

- **Confusion matrix:**

Shown below is the confusion matrix computed for images.

Confusion Matrix			
		Predicted	
		Clean	Suspicious
Actual	Clean	11	18
	Suspicious	10	28
		Accuracy	58%

We did not compute any confusion matrix as such for videos. We cross-checked the predictions using our manual labels.

- **Varying cut-off values:** We should choose our cut-off value such that correct predictions of suspicious instances (True Positives) are maximized, and accuracy is also maximized. The table below shows the corresponding values for different cut-off values that we experimented with. We have used 0.5 as our baseline cut-off value in our code based on which all the predicted values were obtained by the model. But, for evaluating these metrics further, we have manually changed the cut-off and got the below accuracies and suspicious counts. The total count of Suspicious instances classified by us is 38.

Cut-off value	Number of Suspicious instances predicted by the model	Accuracy of the model
0.1	30	55%
0.2	30	57%
0.3	28	55%
0.4	28	58%
0.5	28	58%
0.6	28	61%
0.7	28	61%
0.8	25	57%
0.9	23	60%

FUTURE WORK

We managed to build the baseline model for eye movement at present but would like to improve it furthermore so that it can be used in conjunction with head movements.

We also want to further train and improve the model to detect a wide variety of attributes, in the future, that have historically proved to be reliable indicators of suspicious activity given we have more time, higher computing resources and large amounts of data. Since these are highly complex ideas given the level that we are at now, we are listing only the “what” part as of now and are not sure of the “how” part of these ideas. Here is a list of a few of them:

- Being able to detect suspicious activity in online exams that have a full lockdown of the web browser to prevent sourcing of information from online resources like Google, Gmail, WhatsApp, etc.
- Being able to detect suspicious activity based on emotions and facial expressions of the person.
- Being able to detect suspicious activity when there are multiple people in the same room for the entire duration of the exam or for a smaller duration when they enter, stay for a while and leave,
- Being able to detect suspicious activity using the audio of the screen recording and movements of the mouth to differentiate instances of a person talking to himself and talking/asking others for information,
- After incorporating all the above successfully, being able to implement this model to detect suspicious activity for vendors like Coursera, Edx, Udemy, etc. This can also be extended to accreditation/certification exams like GRE, TOEFL, AWS certifications, etc majority of which are still taken in-person by visiting specialized exam centers.
- Most of the images and videos taken for the purpose of this project were from a single device as there were instances where we had to manually edit images taken from other devices for our model to work properly. In future, we would like to incorporate the capability to handle images taken from any device in our model.

BUSINESS IMPLICATIONS

After the successful implementation of our existing model, it can be used to flag instances of suspicious activity in online based exams based on the examinee’s head and eye movements to assist proctors in intervening in real-time and evaluate the degree of suspicion which would result in effective proctoring than usual but with reduced human effort.

Additionally, our model, when implemented with an acceptable amount of accuracy can be used in a wide variety of cases starting with online exams/assessments and going all the way up to image detection in surveillance videos and Social Media.

RELATED WORK (Academic and Industry)

- [Suspicious Activity Detection in Surveillance Video using Discriminative Deep Belief Network](#)
- [Social Media Image Retrieval Using Distilled CNN for suspicious e-Crime and Terrorist Account Detection](#)
- [Abnormal behaviour recognition for intelligent video surveillance systems](#)

- [Learning Deep Representations of Appearance and Motion for Anomalous Event Detection](#)
- [An intelligent system to detect human suspicious activity using deep neural networks](#)
- [Abnormal Event Detection Using Convolutional Neural Networks and 1-Class SVM classifier](#)
- [Unusual Event Detection via Multi-camera Video Mining](#)

REFERENCES

- <https://medium.com/nybles/create-your-first-image-recognition-classifier-using-cnn-keras-and-tensorflow-backend-6eaab98d14dd>
- <https://towardsdatascience.com/image-recognition-with-keras-convolutional-neural-networks-e2af10a10114>
- <https://www.geeksforgeeks.org/python-image-classification-using-keras/>
- https://medium.com/@jonathan_hui/improve-deep-learning-models-performance-network-tuning-part-6-29bf90df6d2d
- <https://towardsdatascience.com/deep-learning-3-more-on-cnns-handling-overfitting-2bd5d99abe5d>
- <https://www.learnopencv.com/batch-normalization-in-deep-networks/>
- <https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/>
- <https://machinelearningmastery.com/image-augmentation-deep-learning-keras/>
- <https://machinelearningmastery.com/improve-deep-learning-performance/>
- <https://www.pyimagesearch.com/2019/07/08/keras-imagedatagenerator-and-data-augmentation/>
- <https://towardsdatascience.com/image-detection-from-scratch-in-keras-f314872006c9>
- <https://medium.com/@linda0511ny/tensorflow-train-dataset-by-epochs-or-steps-3839705f307d>
- <https://blog.talview.com/a-complete-guide-to-online-remote-proctoring>