UWE Bristol | University of the West of England

Presentation by

Dr. Phil Legg

Associate Professor in Cyber Security

Security Data Analytics and Visualisation

10: Files, Images, Videos

Date: Autumn 2019

Recap

- We have looked at analytics techniques, and visualisation techniques.
- We have surveyed the literature to see the challenges in this area.
- Previously we may have considered data to be confined to numerical tables…

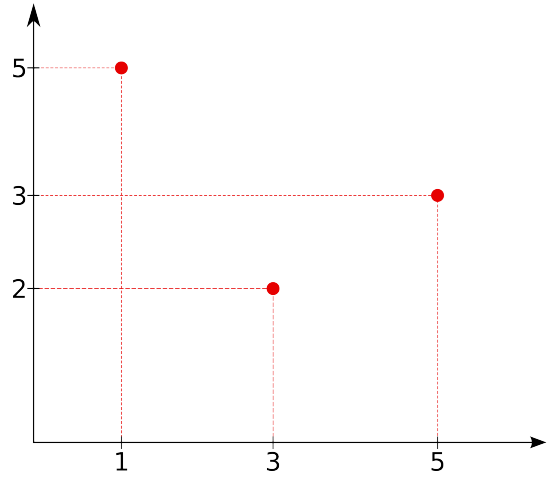
- Let's now consider different forms of data that may be useful for understanding security challenges:
  - This week we will consider **Files, Images, and Video** as a form of data.

UWE Bristol University of the West of England

Binary File Visualisation

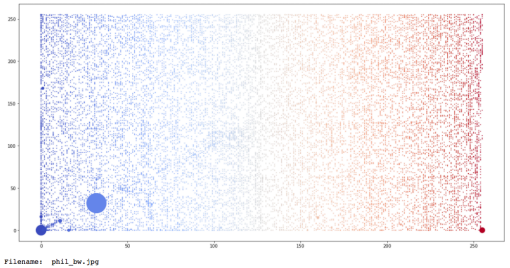
Binary File Visualisation

- How can we analyse and visualise file content?
- Let's consider a stream of bytes in a file:
  - `02 03 05 01`
- We can take pairs of bytes as co-ordinates to plot
  - "Digram Visualisation"
- Could also create a "Trigram" by plotting triplets in 3D

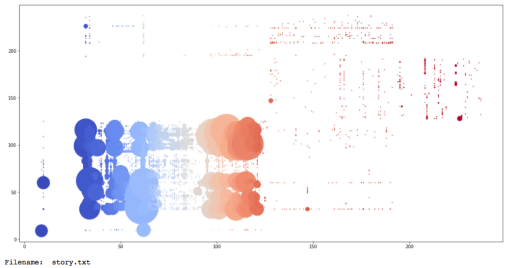


Binary File Visualisation

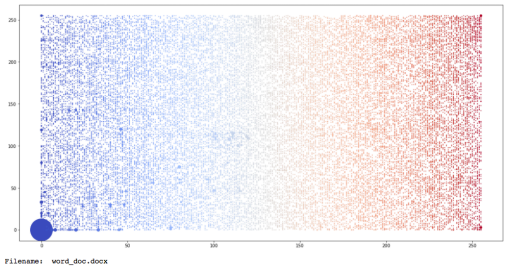
- How can we analyse and visualise file content?



JPEG File

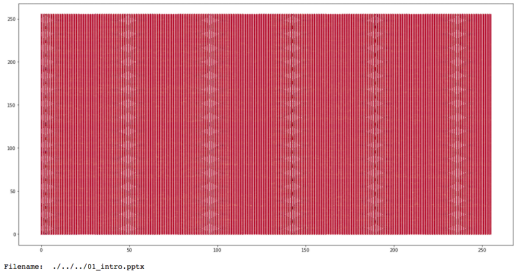


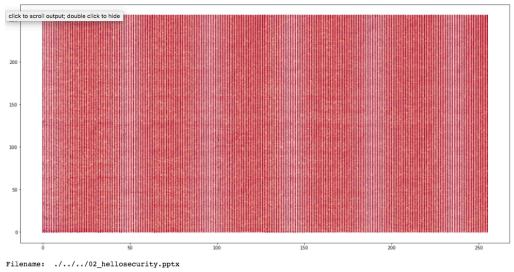
TXT File

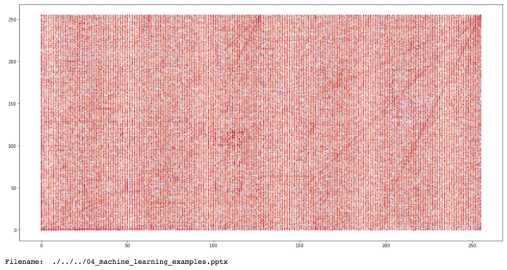


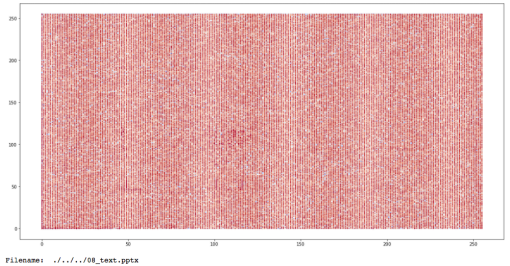
DOCX File

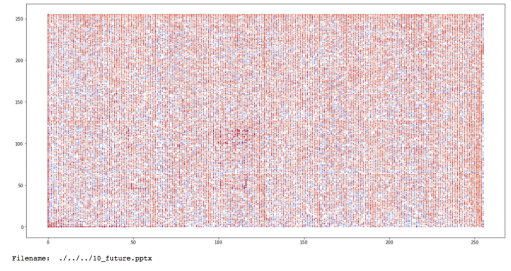
Binary File Visualisation

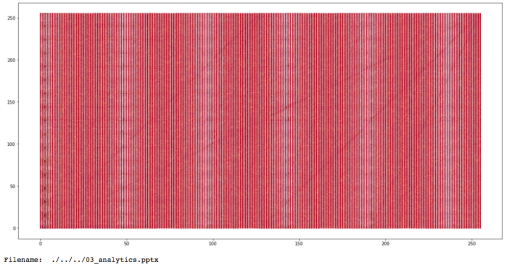








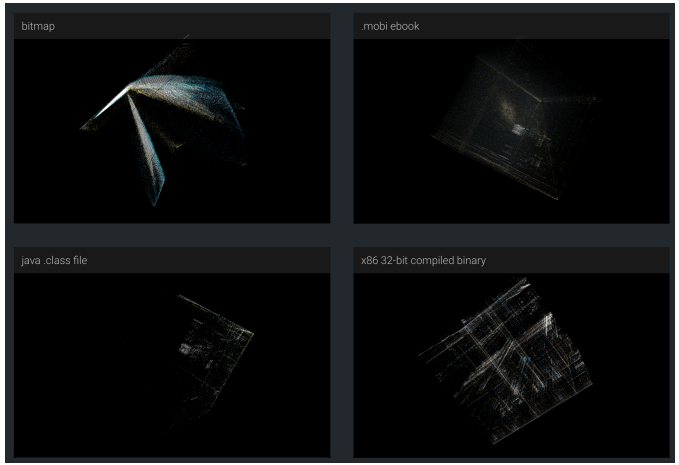


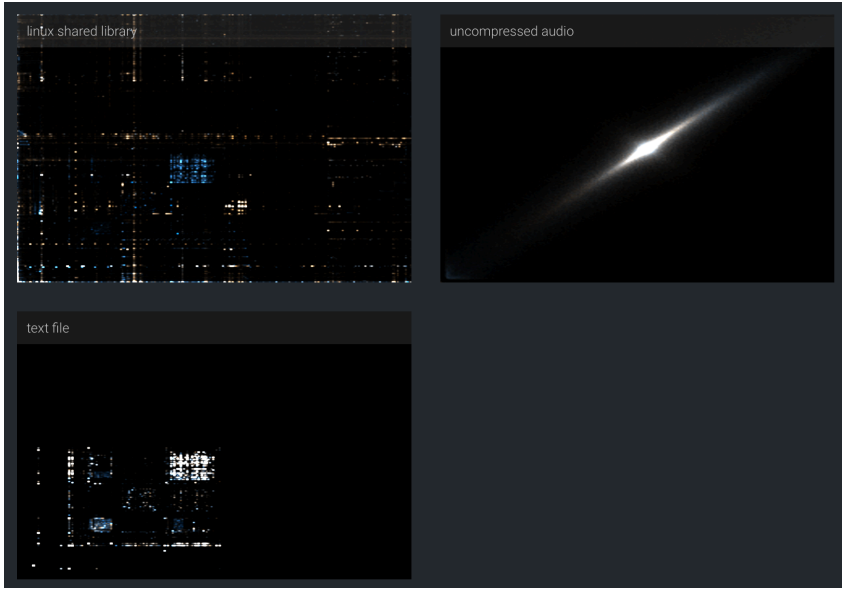


- Can we use this to compare files that are of a similar nature?
- Course Powerpoint slides shown on the left
- Any features that are shared between examples?
- Any differences between examples?

Binary File Visualisation

- More examples of Binary File Visualisation





- https://codisec.com/binary-data-visualization/
- https://codisec.com/binary-visualization-explained/

Binary File Visualisation

- More resources for binary file visualisation

- https://www.devdungeon.com/content/working-binary-data-python
- http://martin.varela.fi/post/simple-binary-data-visualization/
- https://corte.si/posts/visualisation/entropy/index.html
- http://binvis.io/#/

UWE Bristol University of the West of England

Video Analytics and Visualisation

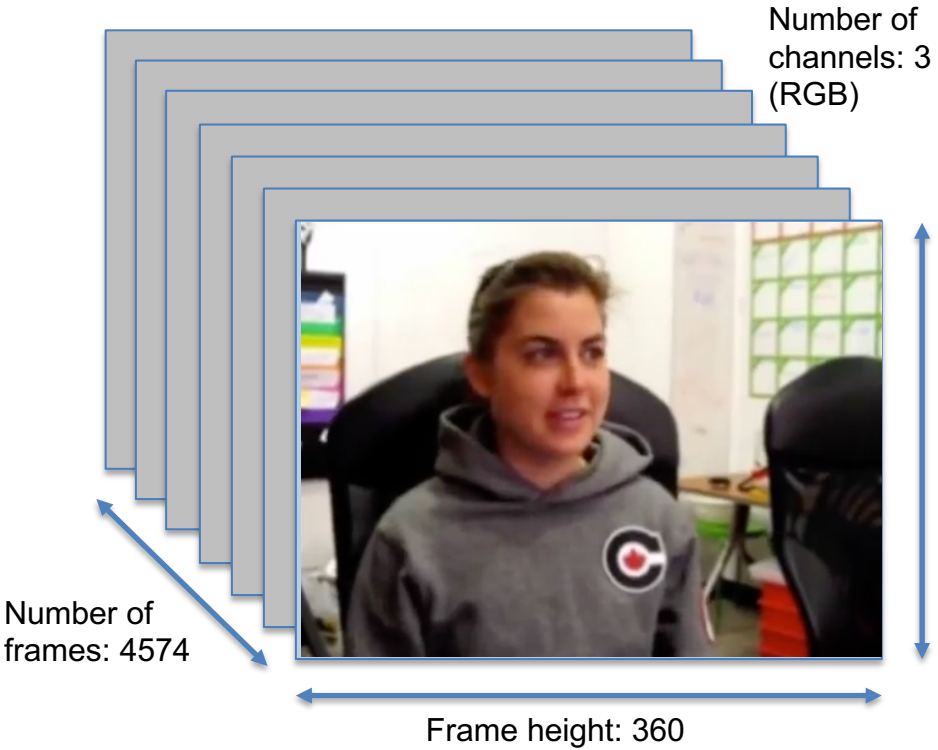
Image and Video

- There is a wealth of image and video data online!
- As like text, too much to analyse manually…
  - How can we make use of machines to do this for us?
  - "Video summarisation" or "Video Visualisation"

- What are the benefits of image and video analysis for security?
  - Facial recognition
  - Activity recognition
  - Change detection

- What sources of video may be useful?
  - E.g., CCTV – too much to analyse manually since constantly collecting

Video Analytics



- Video is essentially a stack of images
- An image is a 3D numerical matrix:
  - `width * height * channels`
- A video is a 4D numerical matrix:
  - `width * height * channels * frame`
- Our video has 4574 frames:
  - 30 frames per second
  - 4574 / 30 = 152.4 seconds
  - Video is 2 minutes 32 seconds

Video Analytics

- Let's try to solve two challenges:
  - Can we detect changes in our video stream?
  - *Imagine a CCTV example – we could identify key moments of interest if we know that something has changed*
  - Can we group (cluster) similar frames together?
  - *We can then recognise patterns of similar activity, or new patterns of activity*

Video Analytics - Dataset

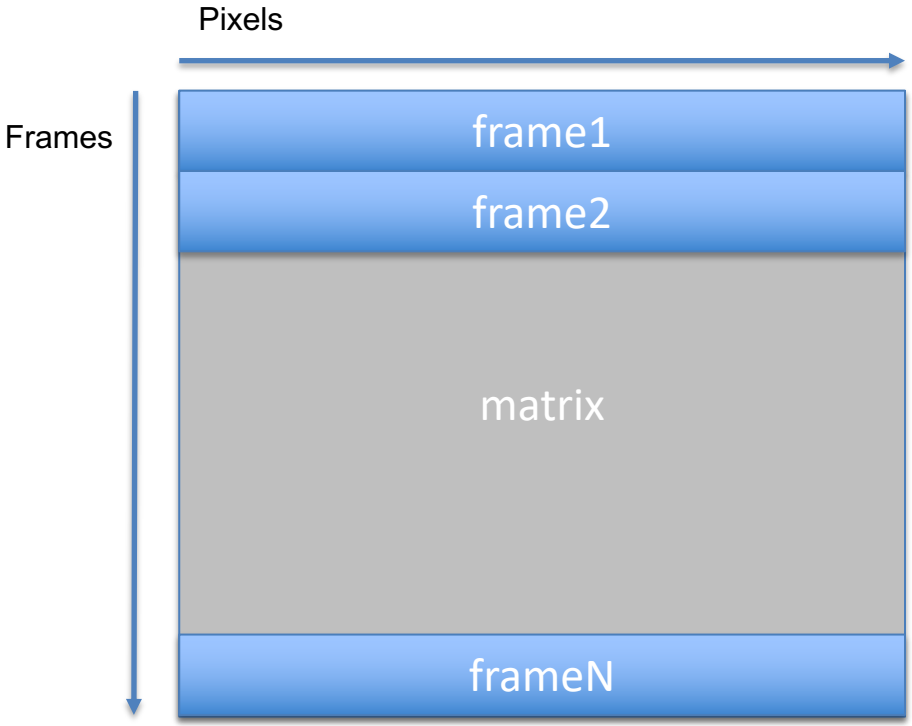
- "Faces" video consists of short video clips that show the faces of 20 unique users
- "McGill Real-World Face Video Dataset"

Video Analytics – Reading data

```
import matplotlib.pyplot as plt
import skvideo.io
import numpy as np

videodata = skvideo.io.vread("09_faces.mp4")
print(videodata.shape)

# rather than processing every video frame, lets grab every 30 frames (as we know the video is 30fps)
frame_rate = 30
new_video = videodata[0::frame_rate, :, :, :]
print (new_video.shape)

matrix = np.zeros([new_video.shape[1] * new_video.shape[2], new_video.shape[0]])

framecount = 0

for frame in new_video:
    print (framecount, frame.shape)
    pixelcount = 0
    for row in range(frame.shape[0]):
        for col in range(frame.shape[1]):
            pixel = frame[row,col,:]
            greyscale = np.mean(pixel)
            matrix[pixelcount, framecount] = greyscale
            pixelcount += 1
    framecount += 1

matrix = matrix.T
```

- For our example, we will extract one frame every second
- This radically reduces our dataset to something more manageable, 153 frames compared to 4574!
- We calculate grayscale pixel values and store video as a 2D matrix

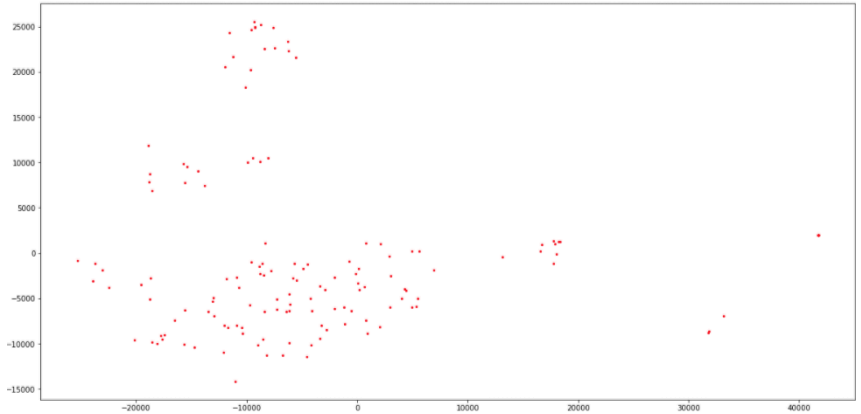
Video Analytics – Storing data



- In the 2D matrix, each row is a frame from the video, that lists all the pixels from that frame

Video Analytics – Analysing data

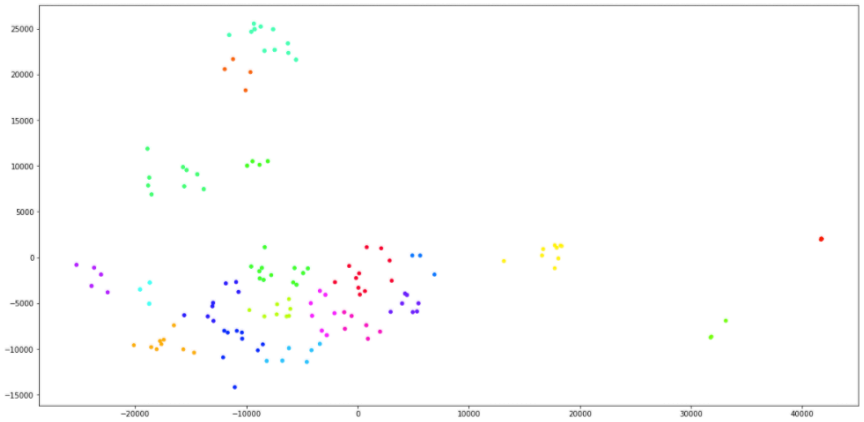
- How can we make sense of this high dimensional matrix that we now have?
- *Principal Component Analysis could be used*
- Let's project (embed) our high dimensional data into something we can plot in 2D.




```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
data = pca.fit_transform(matrix)
fig = plt.figure(figsize=(20,10))
plot = fig.add_subplot(111)
plot.scatter(data[:,0],data[:,1], s=5, c='red', alpha=1)
plt.show()
```

Video Analytics – Analysing data

- We know that we have 20 users to identify
- We could try using k-Means clustering on our data to identify 20 clusters?

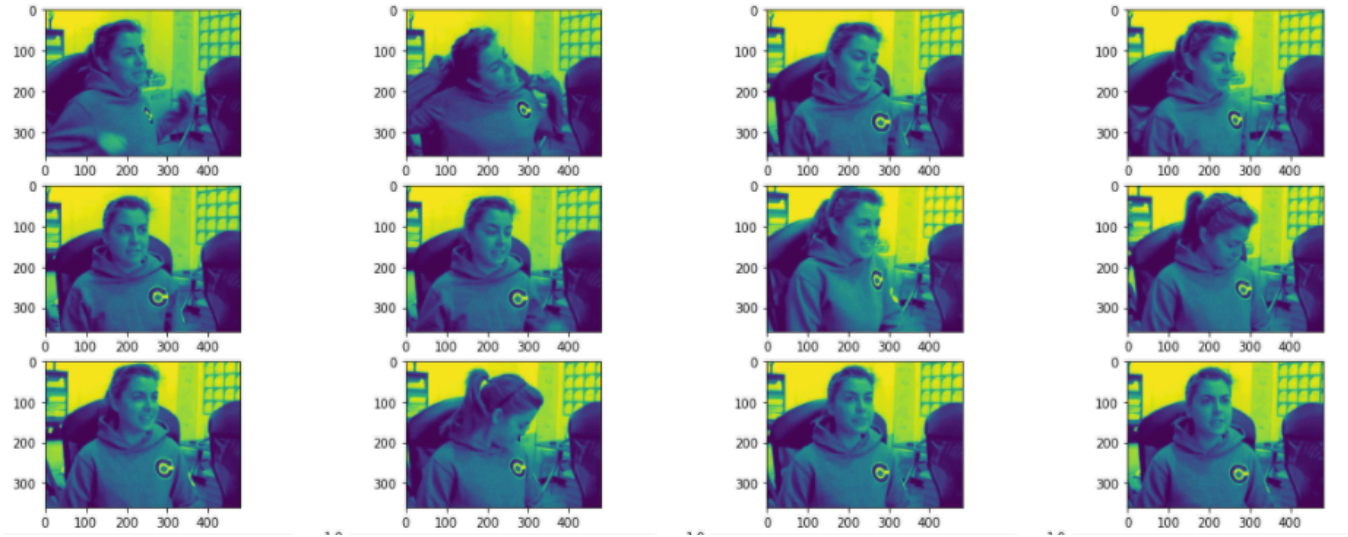



```
1 from sklearn.cluster import KMeans
2 kmeans = KMeans(n_clusters=20, random_state=0).fit(data)
3 kmeans.labels_
```

```
array([ 1,  6,  6,  6,  1, 14, 12, 14, 14,  3,  1, 17, 17, 17, 17, 17,  1,
       16, 16, 16, 16,  1,  9,  9,  9,  9,  9,  9,  1, 10, 10, 10, 10,
       10, 10,  2, 10,  2,  2,  2, 10, 10, 10, 10, 10,  1, 15, 15, 14, 15,
        1, 15, 14, 14, 14,  5, 18,  5, 14,  1,  5, 18, 19,  0,  0, 18,  0,
       18,  5,  5, 14, 18, 12, 12, 18, 19,  5,  5,  1,  9,  7,  7,  7,  7,
        3,  9,  1, 15, 11,  3,  3, 11,  3,  3,  3, 11,  3,  3,  1, 13, 16,
       16, 19, 12, 12, 18,  8,  0,  8, 15,  8,  8,  8, 15,  0,  8,  0,  1,
       19, 19, 19,  1,  8,  8,  8,  8,  5,  8,  8,  1,  0,  0,  0,  0,  1,
        4,  4,  4,  4,  4,  4,  4,  4,  1, 15, 12,  1, 16,  4, 13, 13,  0], dtype=int32)
```

Video Analytics – Result

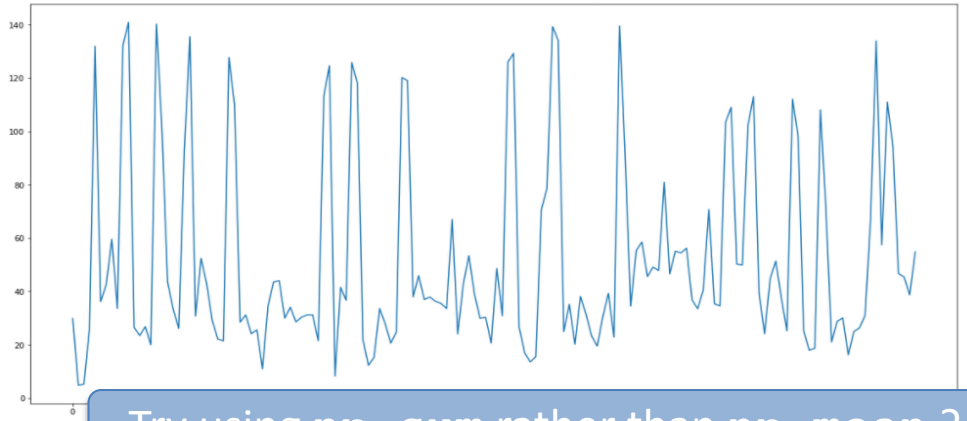
- We now have groups of similar video frames that we can inspect further



- Our clustering found all frame with our original lady on!
- Should be noted – not all clusters work as well…!

Video Analytics – Result

```
diffs = []
frame_number_diffs = []
for frame_number in range(matrix.shape[0] - 1):
    diff = np.mean(np.abs(matrix[frame_number,:] - matrix[frame_number+1,:]))
    diffs.append(diff)
    if diff > 100:
        frame_number_diffs.append(frame_number)
plt.figure(figsize=(20,10))
plt.plot(diffs)
plt.show()
```



Try using `np.sum` rather than `np.mean` ?

- We can use the same matrix configuration to perform video change detection between frames.
- Essentially, we just calculate the pixel values differences from one frame to the next.
- A peak in our time series shows that there was a signficant change in the video (i.e., a scene change).

Summary

- We can analyse and visualisation raw file content
  - File inspection and comparison with similar file types
- We can analyse and visualisation image and video content
  - Can provides a means of understanding faster than viewing the video content (summarisation)
- Image and video are important forms of data for physical security
  - Facial / Activity Recognition, and Change detection