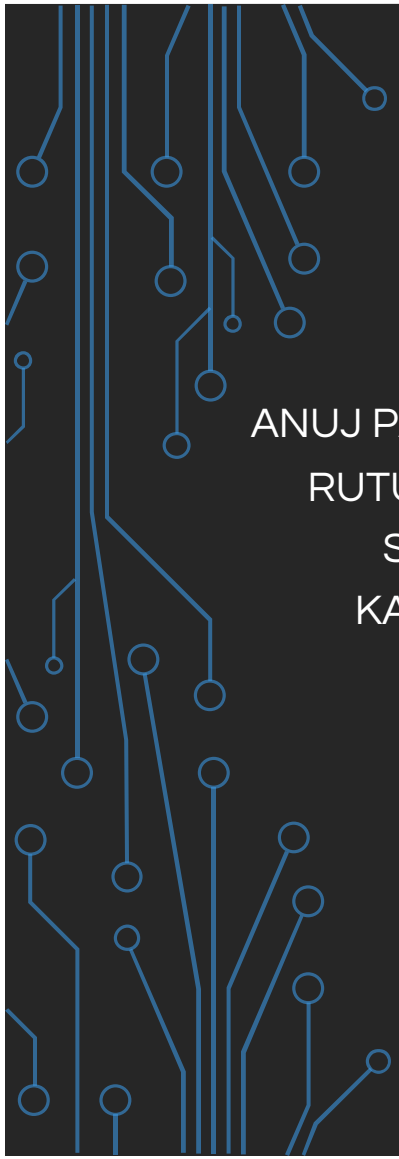


ANUJ PANCHMIA ,
RUTU GANDHI ,
SUSHANTH
KATHIRVELU

DATA SCIENCE PRACTICUM PROJECT 2 TEAM CRAGG

Type of problem

Semantic segmentation aims at grouping pixels in a semantically meaningful way.

So semantic segmentation does a pixel-wise classification

The two popular models that work well:

UNet

MaskRCNN

Dice loss

```python
smooth = 1.
def dice_coef(y_true, y_pred):
    """ The dice coef is a metric to calculate the similarilty
    (intersection) between the true values and the predictions"""
    y_true_f = K.flatten(y_true)
    y_pred_f = K.flatten(y_pred)

    intersection = K.sum(y_true_f * y_pred_f )
    return (2. * intersection + smooth) / (K.sum(y_true_f) + K.sum(y_pred_f) + smooth)
```

Table 1. The three similarity coefficients

Similarity Coefficient (X,Y)	Actual Formula
Dice Coefficient	$2\frac{\lvert X \cap Y \rvert}{\lvert X \rvert + \lvert Y \rvert}$
Cosine Coefficient	$\frac{\lvert X \cap Y \rvert}{\lvert X \rvert^{1/2} . \lvert Y \rvert^{1/2}}$
Jaccard Coefficient	$\frac{\lvert X \cap Y \rvert}{\lvert X \rvert + \lvert Y \rvert - \lvert X \cap Y \rvert}$

PRE-PPROCESSING

- Resizing
- Zero Padding
- Variance
  - Grayscale
  - FFT
- Optical Flow
  - Video2
  - Frames(1st and 50th)
- Mean of Optical Flow and Variance

Resize

(256, 256) and (512, 512)

Zero Padding

Strategy 1: Zero Pad and tile

x=128 y=128

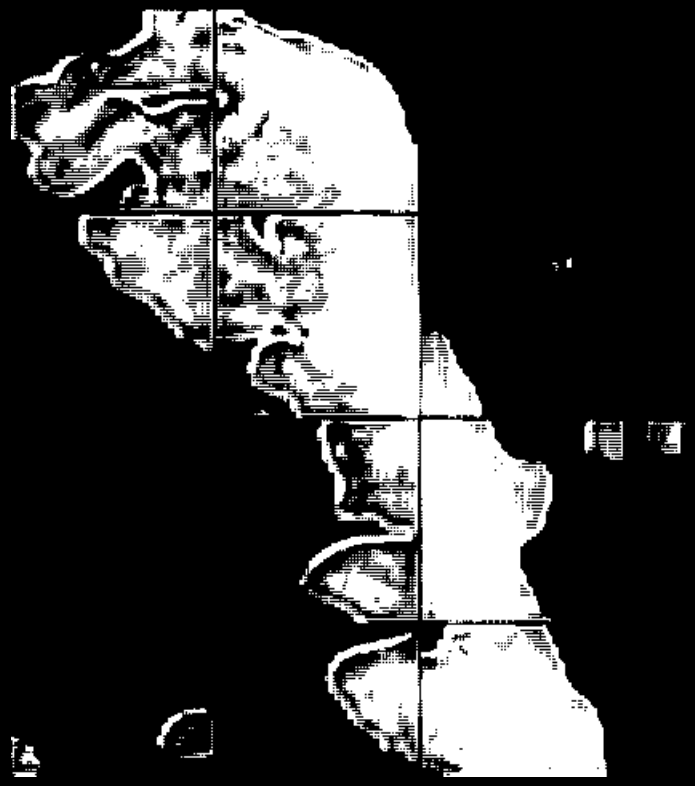
image size = (640,480)

zero padding : 640 = 5x, 480 = 3.75y

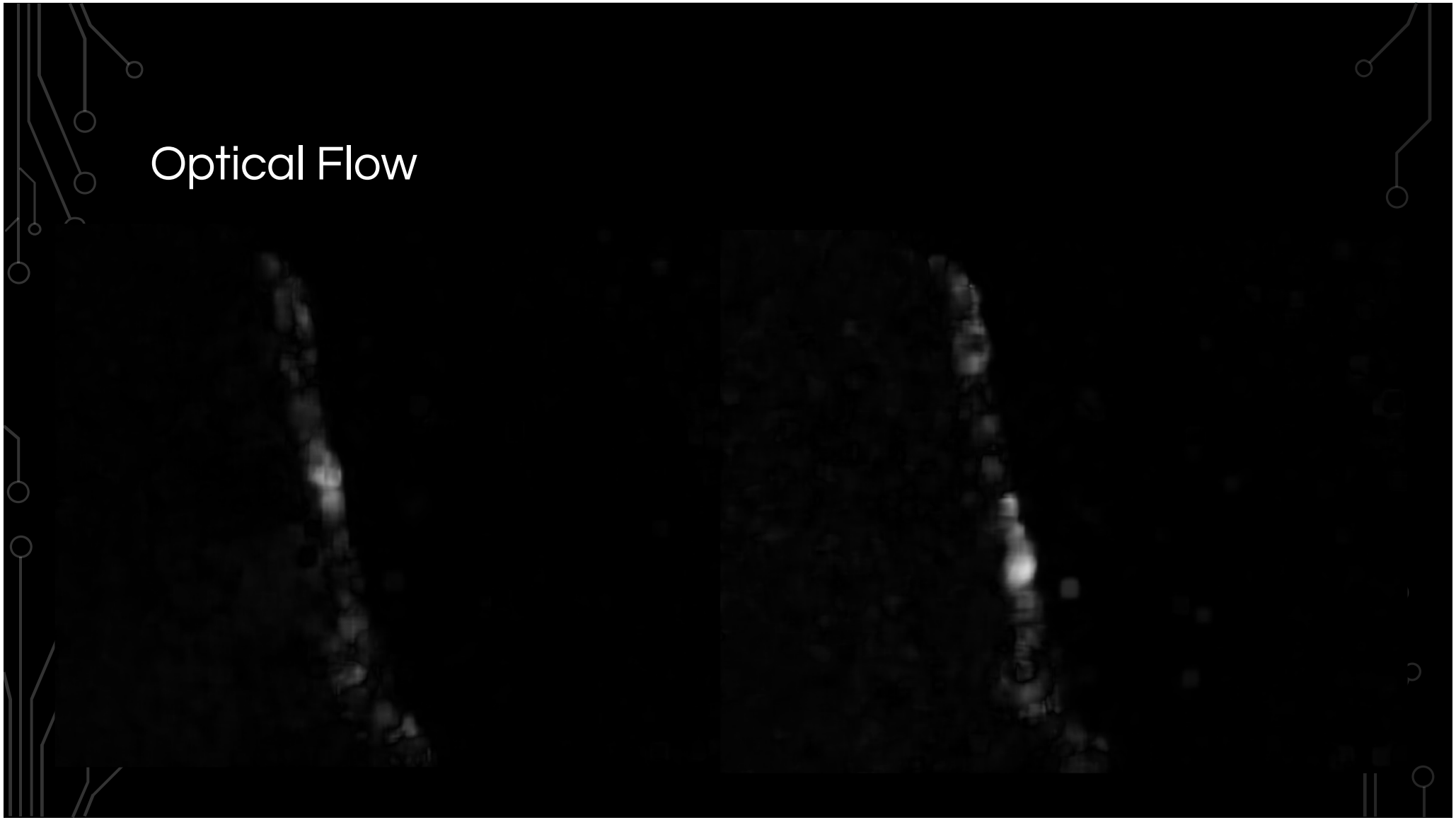
Tile

Strategy 2: Simple zero padding into (640, 640)

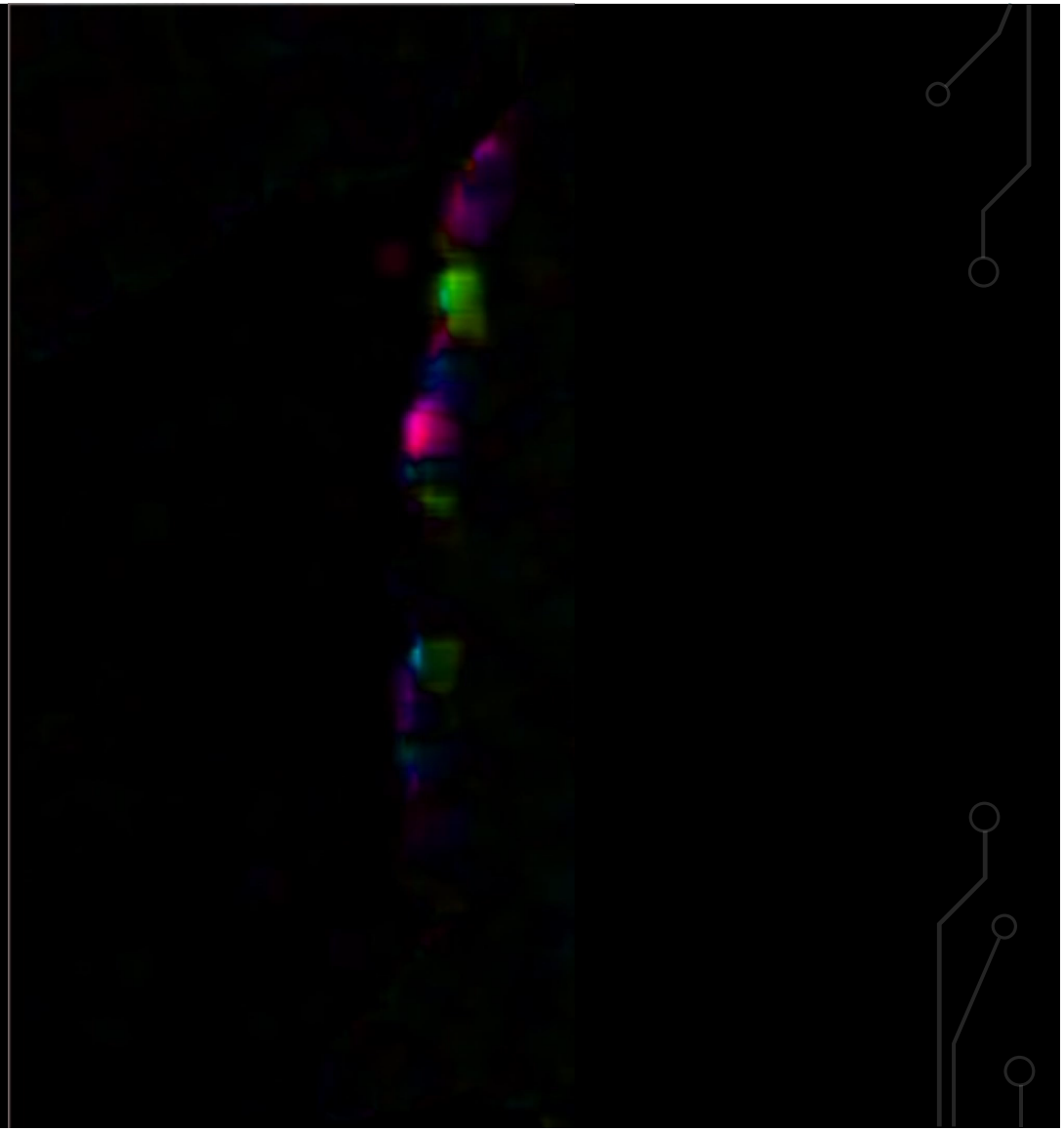
Tailor and Stitch



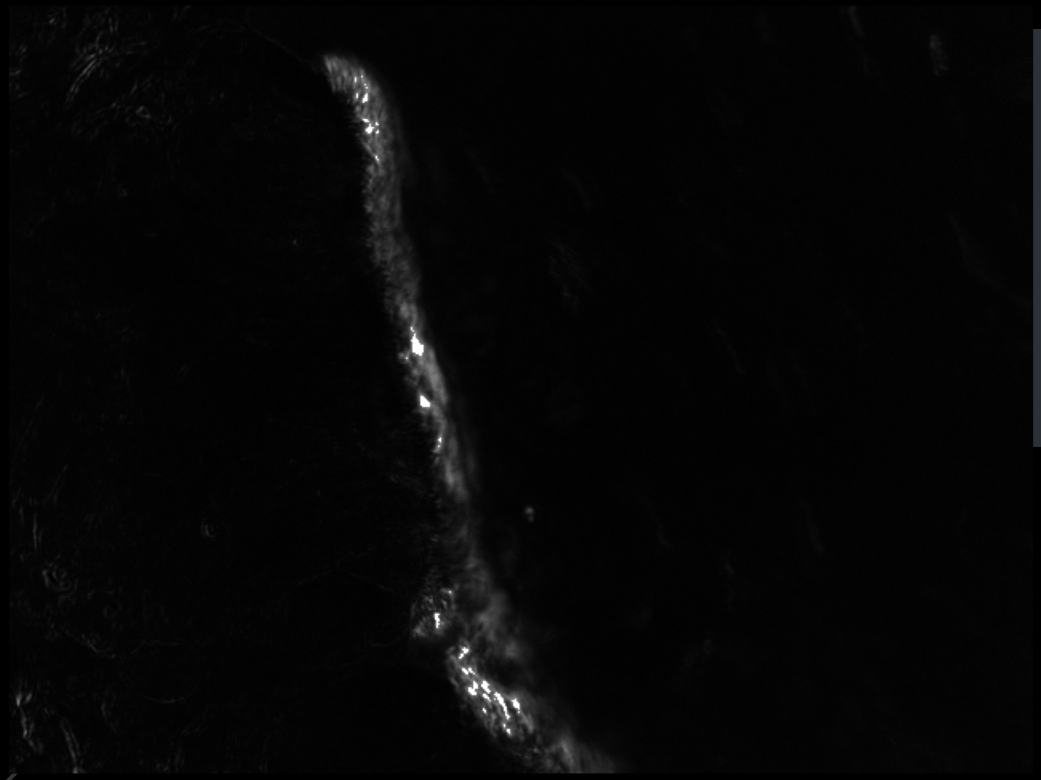
Optical Flow



Optical Flow Video

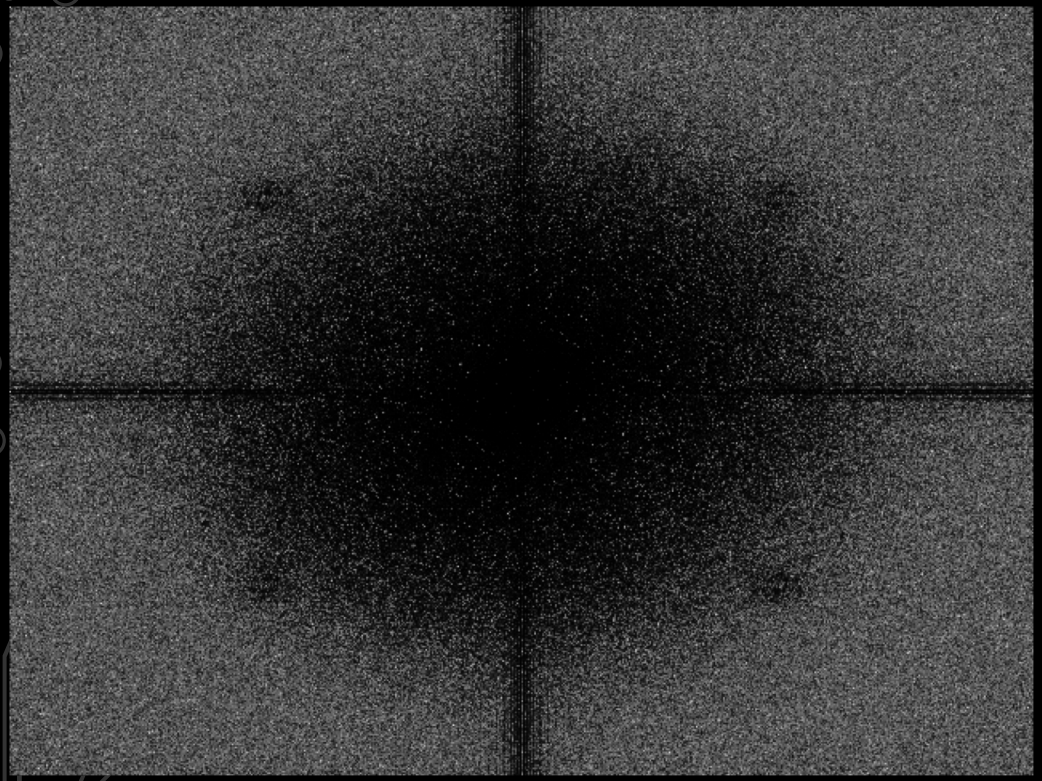


Gray Scale variance



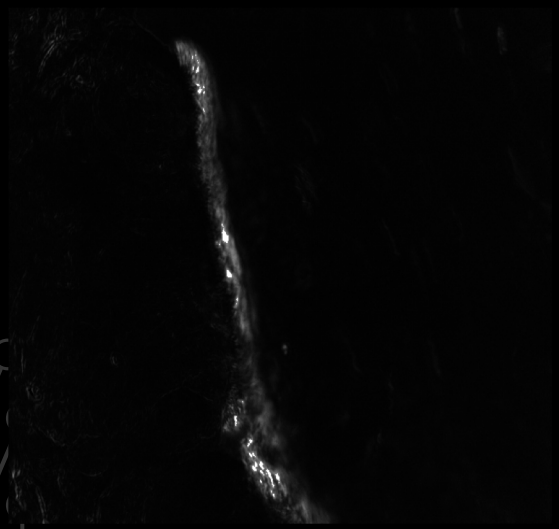
```
for i in range(len(X_train)):
    #hashName = file[i].strip()
    variances = np.var(create(i),axis=0)
    im = (variances/np.max(variances))
    for x in range(0,im.shape[0]):
        for y in range(0,im.shape[1]):
            if im[x,y] >= 0.5:
                im[x,y] = 1
            else:
                im[x,y] = im[x,y]*1
```

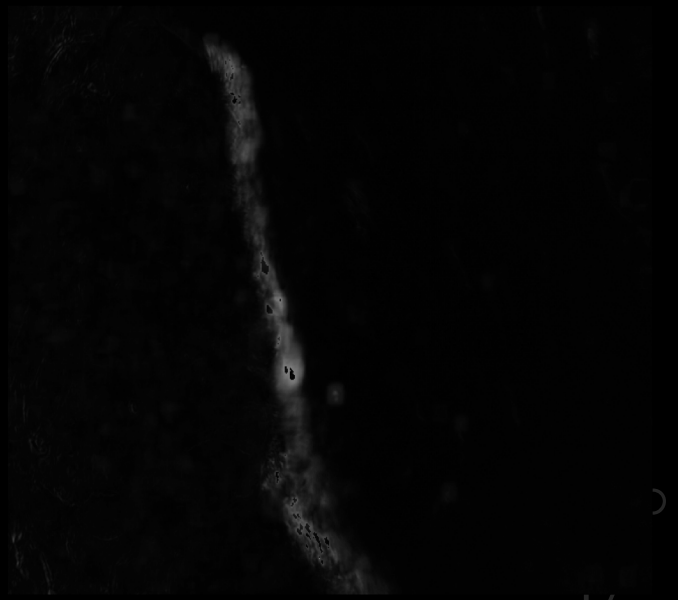
FFT + variance



```
def create(j):
    mag_spec_2 = []
    for i in range(100):
        f = np.fft.fft2(X_train[j][i])
        fshift = np.fft.fftshift(f)
        magnitude_spectrum1 = 20*np.log(np.abs(fshift))
        mag_spec_2.append(magnitude_spectrum1)
        mag_spec_new = np.array(mag_spec_2)
    return mag_spec_new

if not os.path.exist(savePath):
    os.mkdir(savePath)

## path for the traing file

file = open(path)


for i in range(len(X_train)):
    #hashName = file[i].strip()
    variances = np.var(create(i),axis=0)
    im = (variances/np.max(variances))
    for x in range(0,im.shape[0]):
        for y in range(0,im.shape[1]):
            if im[x,y] >= 0.5:
                im[x,y] = 1
            else:
                im[x,y] = im[x,y]*1

    cv2.imwrite(savePath +"/"+ hashName +".png", im)
```

Gray scale variance + Optical Flow averaging





RESULTS

Model	Pre-Processing	Accuracy on Test
Unet	Resize(256,256)	9.0
Unet	Resize(512,512)	14.8
Unet	Zero Padding(640,640)	10.3
Unet	Zero Padding(640,640),Optical Flow of 2 Images	24.7
Unet	Zero Padding(640,640),Optical Flow of Video	13.1
Unet	Zero Padding(640,640),Grey Scale Variance	36.8
Unet	Zero Padding(640,640),FFT Variance	11.1
Unet	Zero Padding(640,640),Mean of Optical Flow and Variance	21.2

THANK YOU

Questions??