BLUR IMAGE RESTORATION WITH DEEP LEARNING

Ran Minerbi

BLUR IMAGES




CALCULATE TRANSFER FUNCTION H(U,V) FOR BLUR IMAGE CAUSED BY UNIFORM ACCELERATION FOR $0 < t < T$

$$x_{(t)} = \frac{1}{2} a_x t^2 , y_{(t)} = \frac{1}{2} a_y t^2$$

RESTORE WITH AX,AY

$$0 < t < T$$

$$x_{(t)} = \frac{1}{2}a_x t^2, y_{(t)} = \frac{1}{2}a_y t^2$$

$$h(x,y) = \delta(x - x_{(t)}, y - y_{(t)})$$

$$H(u,v) = \Im(h(x,y)) = \Im(\delta(x - x_{(t)}, y - y_{(t)})) = e^{-j2\pi x_{(t)}u} e^{-j2\pi y_{(t)}v}$$

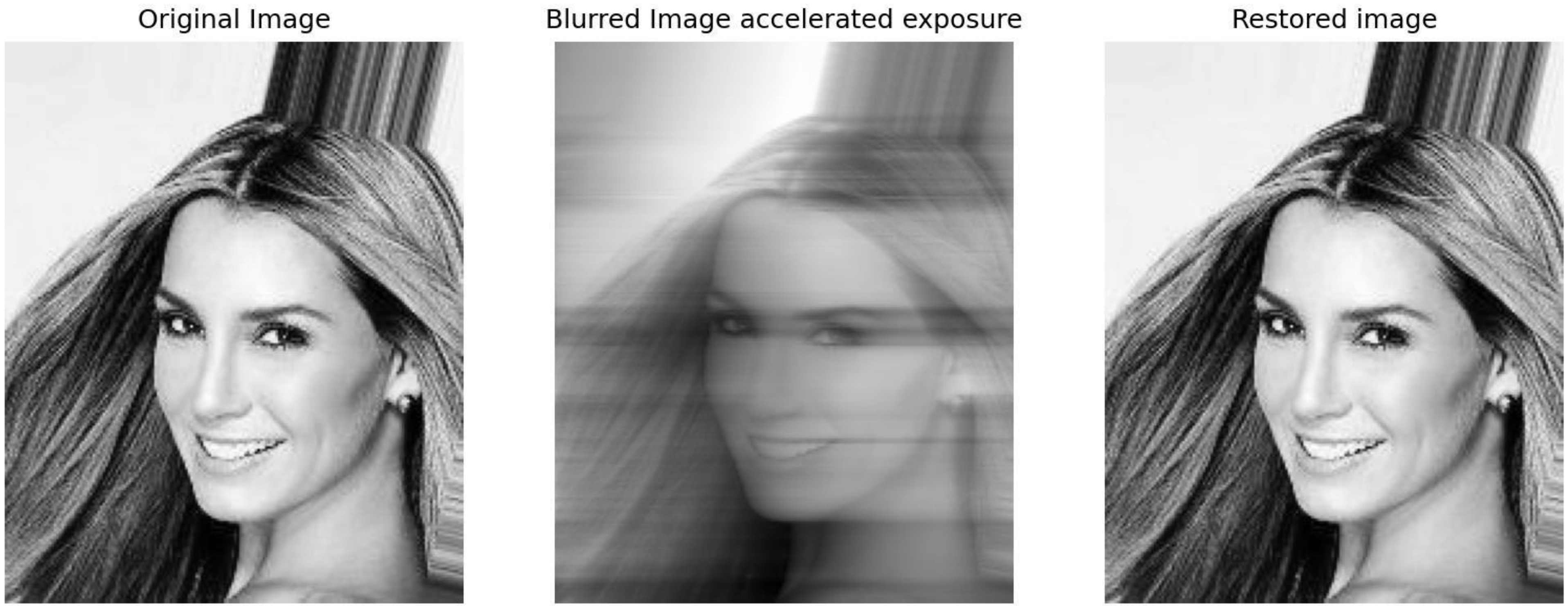
$$H(u,v) = e^{-j2\pi\frac{1}{2}a_x t^2 u} e^{-j2\pi\frac{1}{2}a_y t^2 v}$$

$$H(u,v) = \sum_{t=0}^{T} e^{-j2\pi\frac{1}{2}a_x t^2 u} \sum_{t=0}^{T} e^{-j2\pi\frac{1}{2}a_y t^2 v}$$

$$G(u,v) = F(u,v)H(u,v) = F(u,v)\sum_{t=0}^{T} e^{-j2\pi\frac{1}{2}a_x t^2 u} \sum_{t=0}^{T} e^{-j2\pi\frac{1}{2}a_y t^2 v}$$

$$f_r(x,y) = \Im^{-1}\left(\frac{G(u,v)}{\sum_{t=0}^{T} e^{-j2\pi\frac{1}{2}a_x t^2 u} \sum_{t=0}^{T} e^{-j2\pi\frac{1}{2}a_y t^2 v}}\right)$$

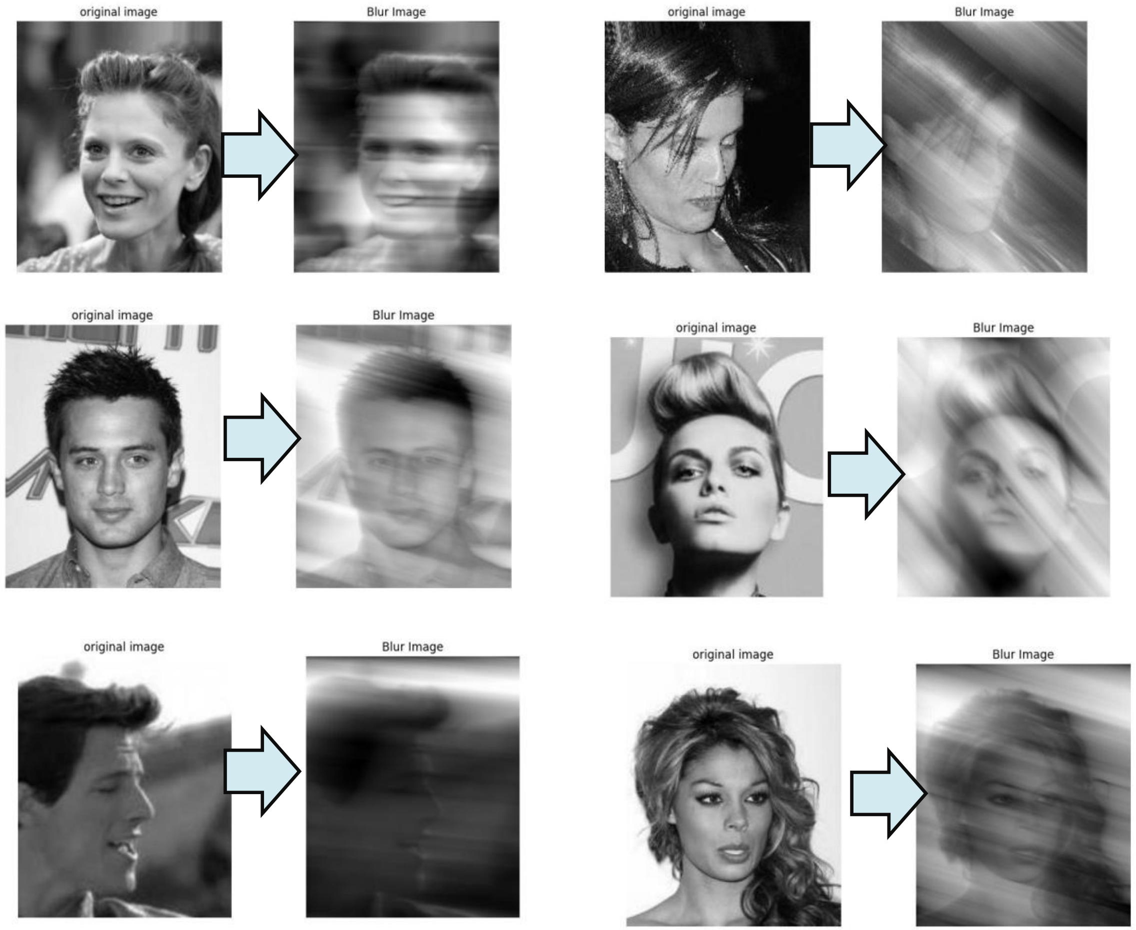
IMAGE RESTORE WITH (AX,AY)



IS IT POSSIBLE TO RESTORE IMAGES NOT KNOWING (AX,AY) ?

•Solution procedure

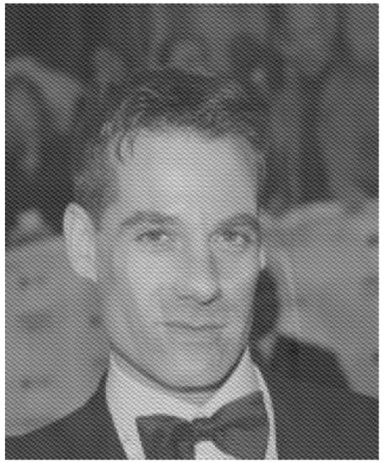
- Distort images on purpose within **known acceleration parameters (ax,ay)**
- Train set of 150 blurd images
- Train labels of 150 (ax,ay)
- Test set of 50 blurd images
- Test Labels on 50 (ax,ay)



ACCURACY IS CRITICAL

- (Ax,Ay) = [1.4724174320412484 ,0.0916516522235391]

Filtered image

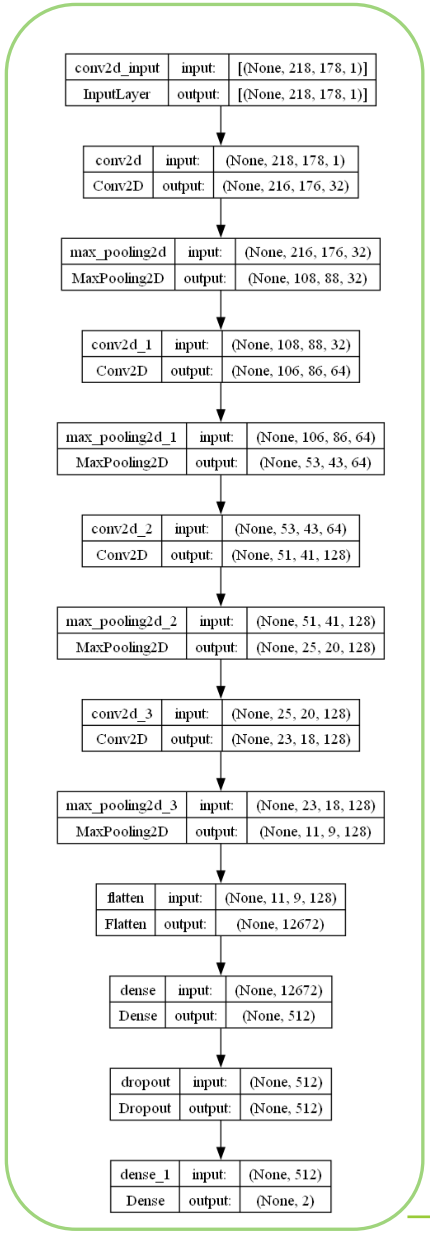


Err(Ax) = 0
Err(Ay) = 0.1

Err(Ax) = 0.1
Err(Ay) = 0.3

Err(Ax) = 0.1
Err(Ay) = 0.5

blurd image



. . .

(Ax,Ay)

APPROACH 1
CONVOLUTION DL NETWORK

TRAINING LOSS OUTLINE PROCEDURE

- Loss should be minimized during training

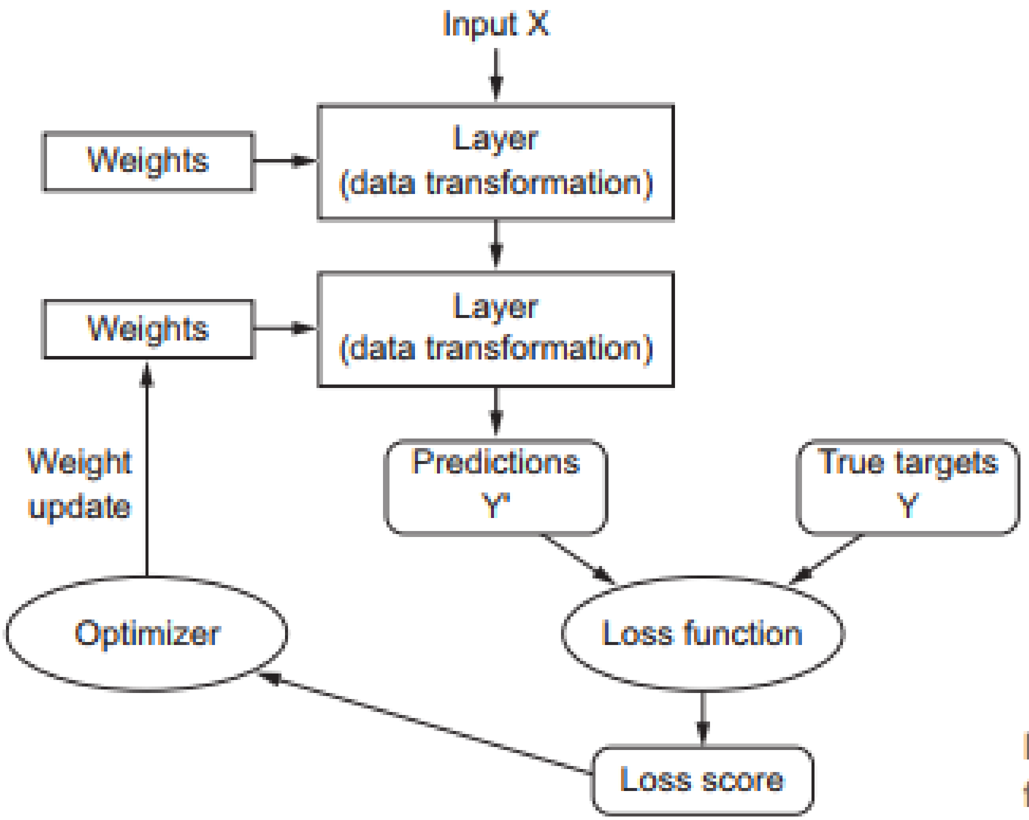
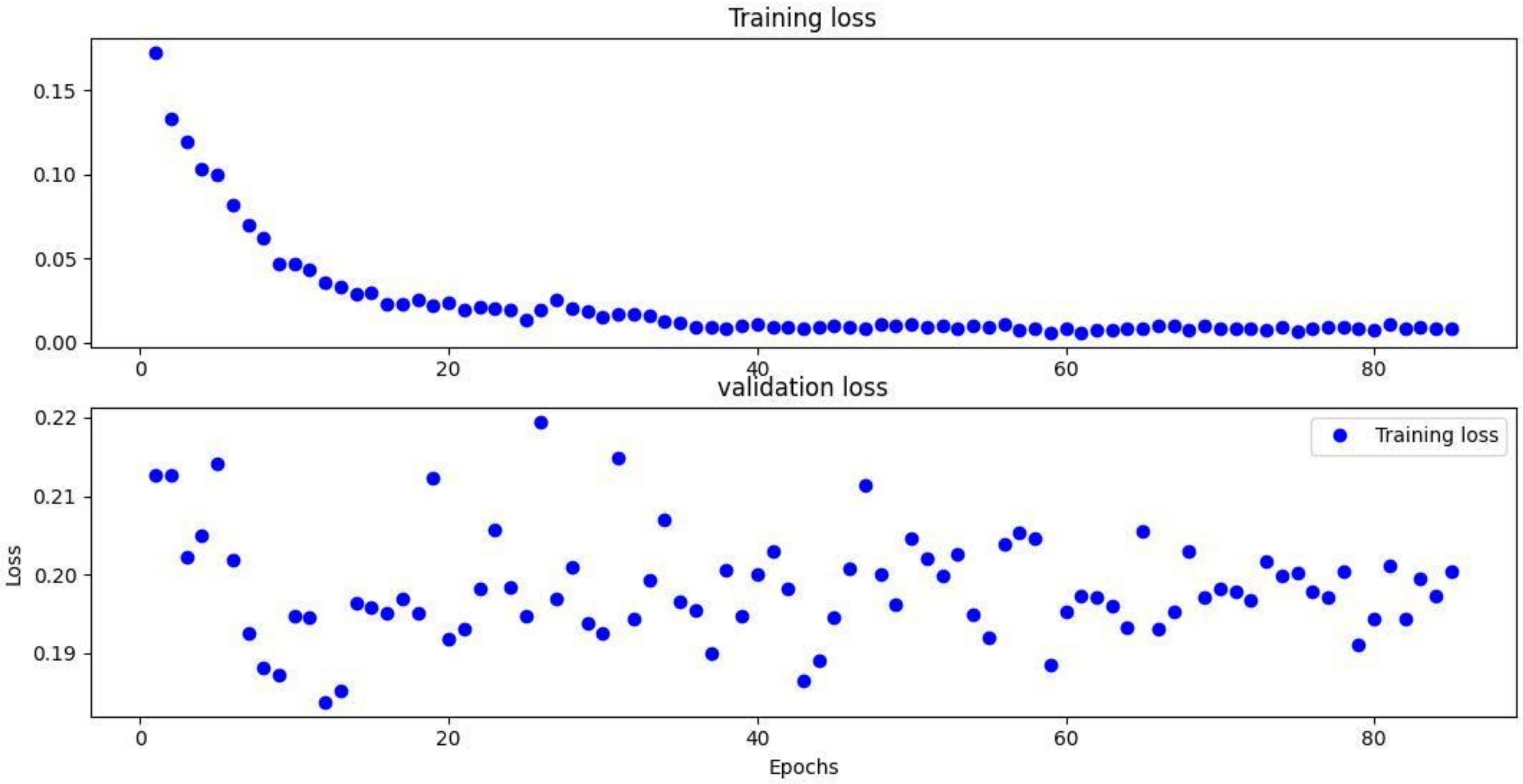


Figure 1.9 The loss score is used as a feedback signal to adjust the weights.

APPROACH 1
CONVOLUTION DL NETWORK



OVERFIT

- *Optimization* refers to the process of adjusting a model to get the best performance possible on the training data, whereas *generalization* refers to how well the trained model performs on data it has never seen before

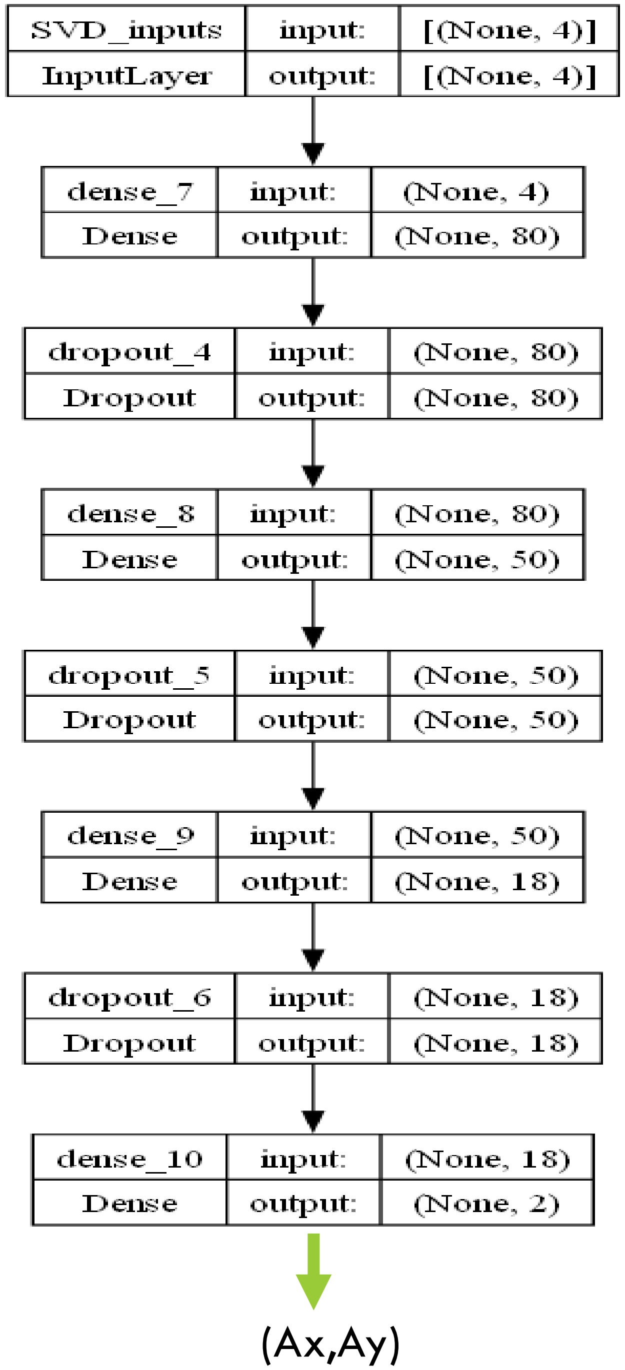
- Learning how to deal with overfitting is essential to mastering machine learning

- **Avoid overfitting :**
  - Get more training data
  - Reduce network capacity
  - Add weight regularization
  - Add dropouts

$$sobel\_dx(f(x,y)) = \frac{\partial^2}{\partial x} f(x,y)$$

$$sobel\_dy(f(x,y)) = \frac{\partial^2}{\partial y} f(x,y)$$

$$\mathrm{cov}_{2x2}(sobel\_x, sobel\_y)$$

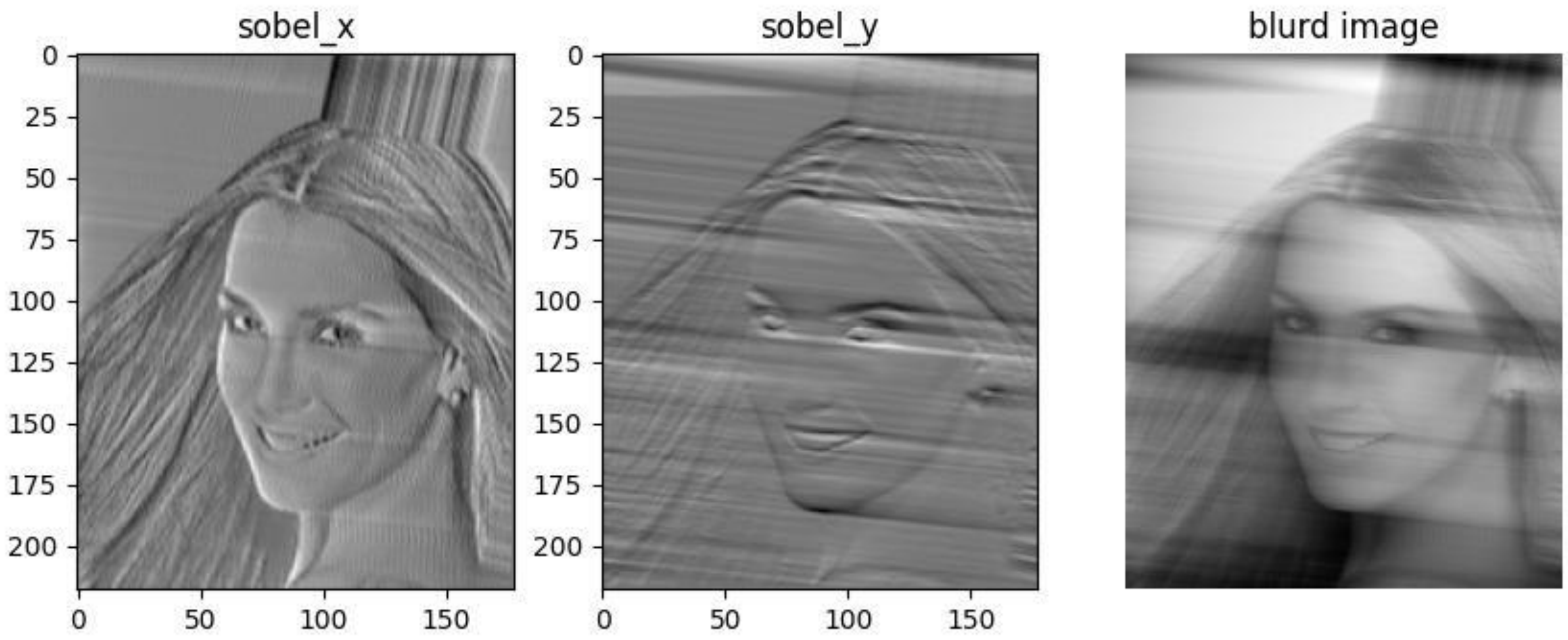


APPROACH 2

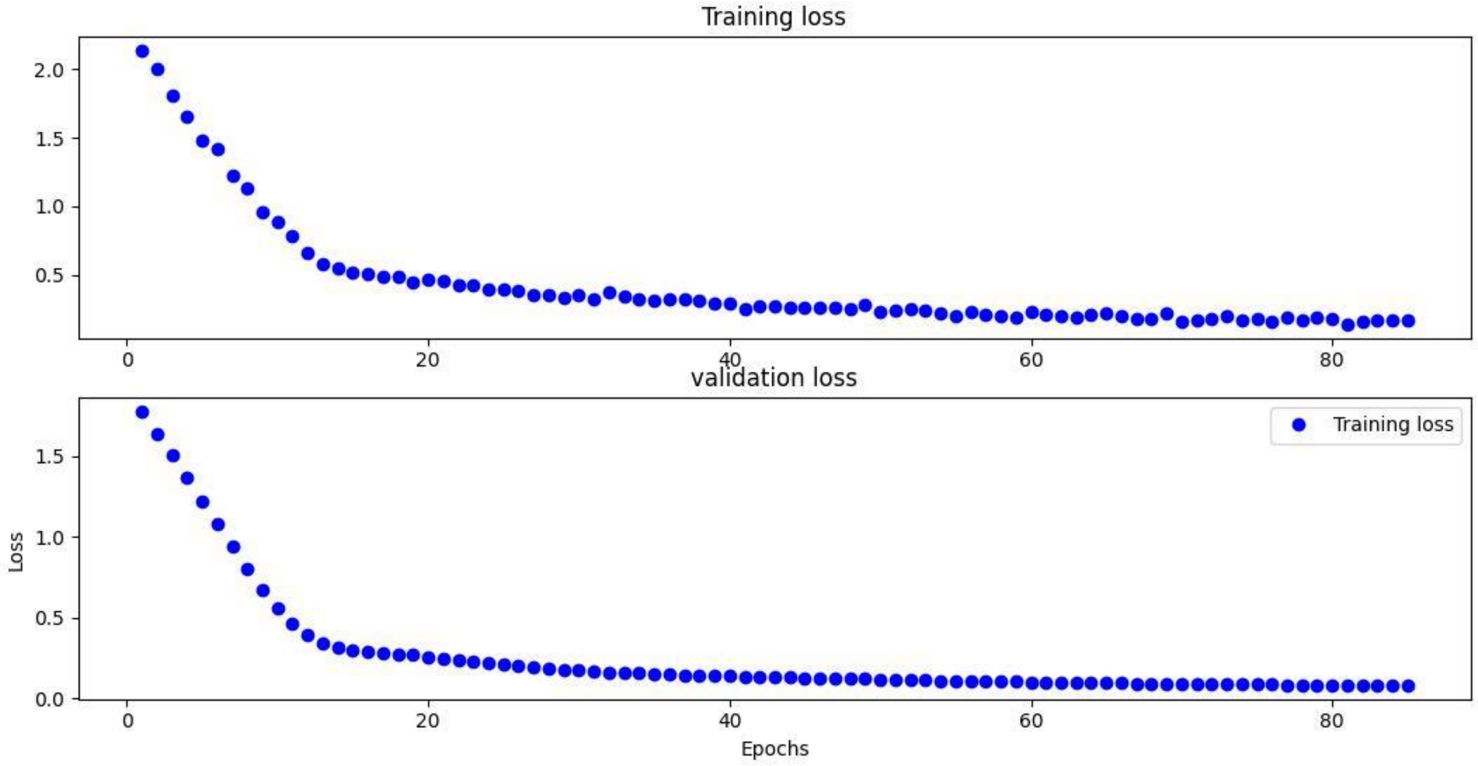
COVARIANCE MATRIX WITH FULLY CONNECTED DL NETWORK

APPROACH 2

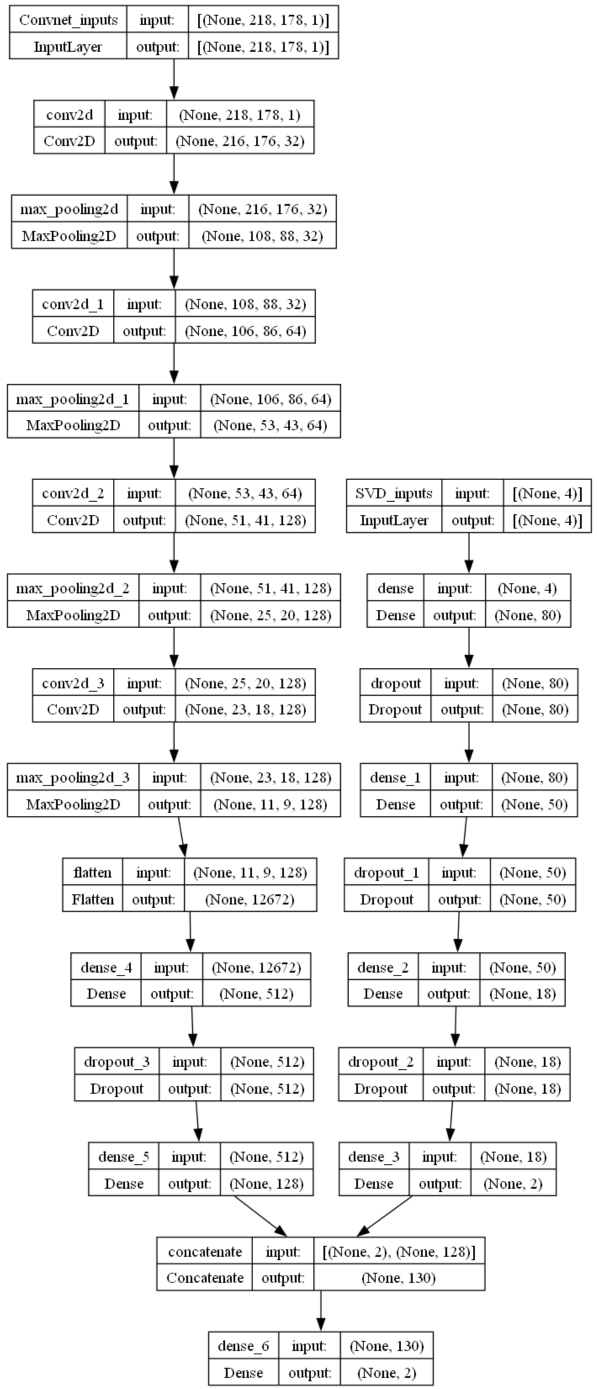
COVARIANCE MATRIX WITH FULLY CONNECTED DL NETWORK



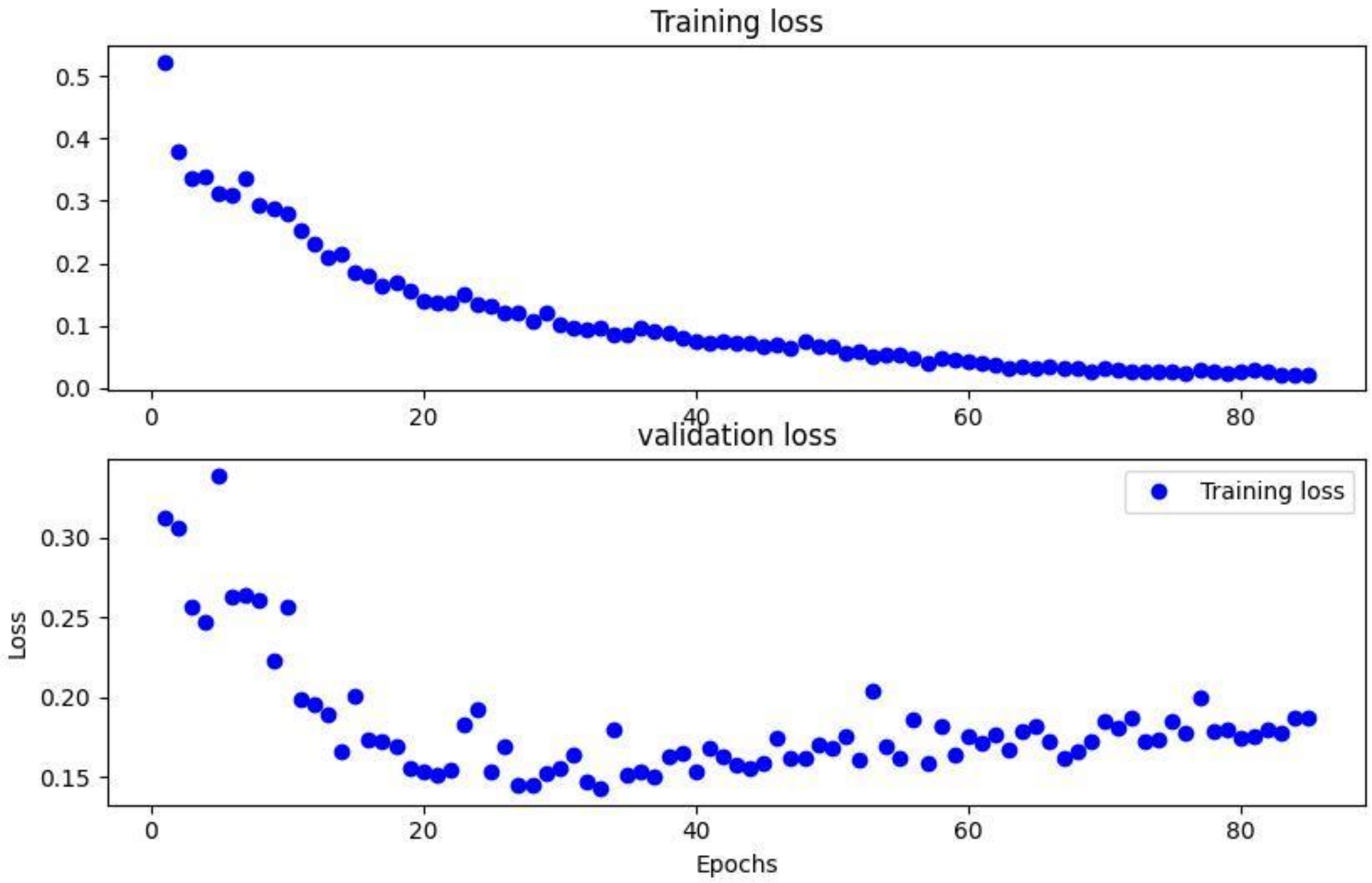
APPROACH 2
COVARIANCE



APPROACH 3
CONVNET- COVARIANCE



APPROACH 3 CONVNET-COVARIANCE



APPROACH 4 CONVNET OVER SOBEL

$$sobel\_dx(f(x,y)) = \frac{\partial^2}{\partial x} f(x,y)$$

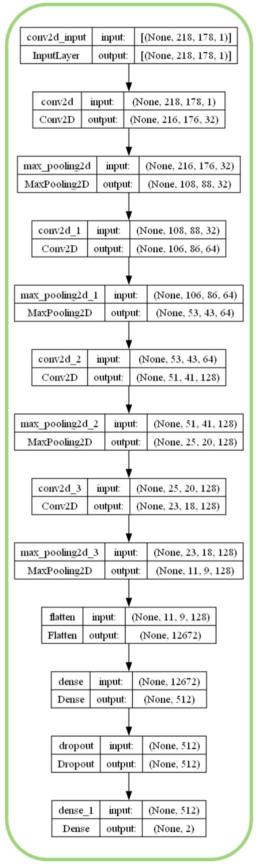
$$sobel\_dy(f(x,y)) = \frac{\partial^2}{\partial y} f(x,y)$$

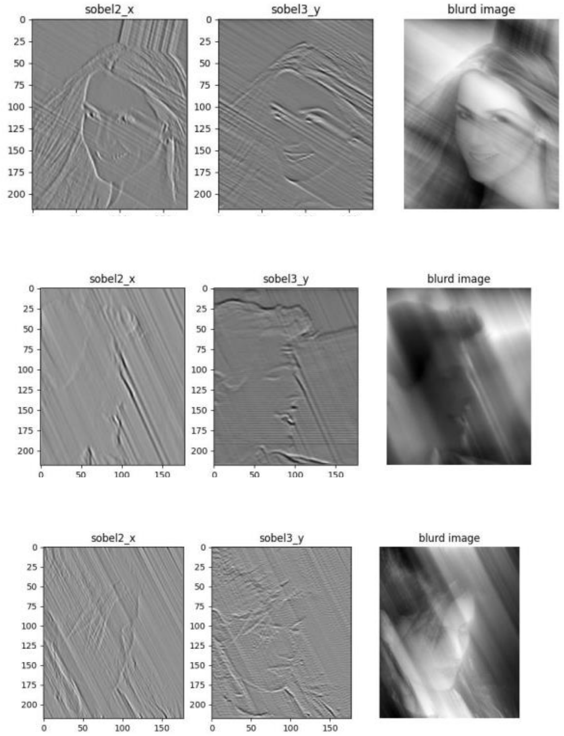
APPROACH 4

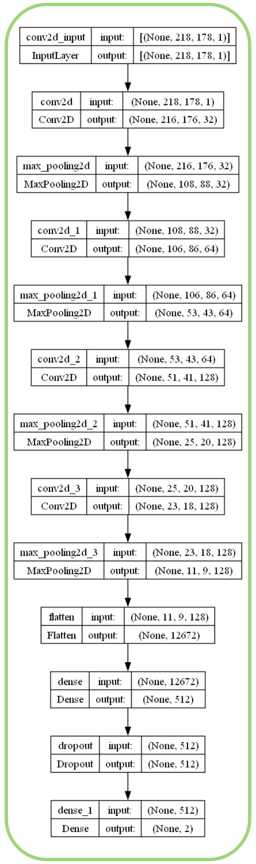
X DL path

Y DL path

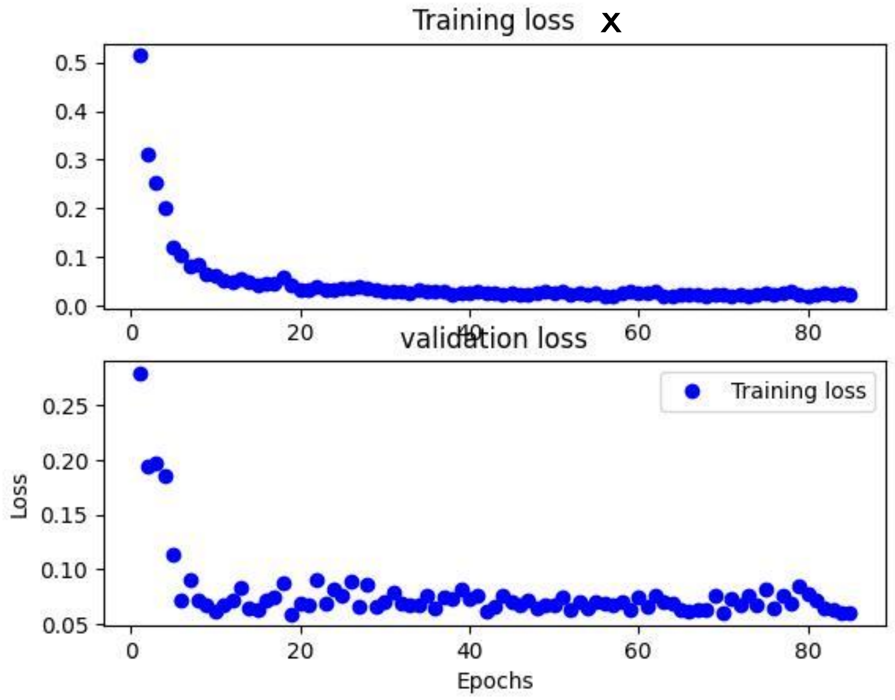
Ax

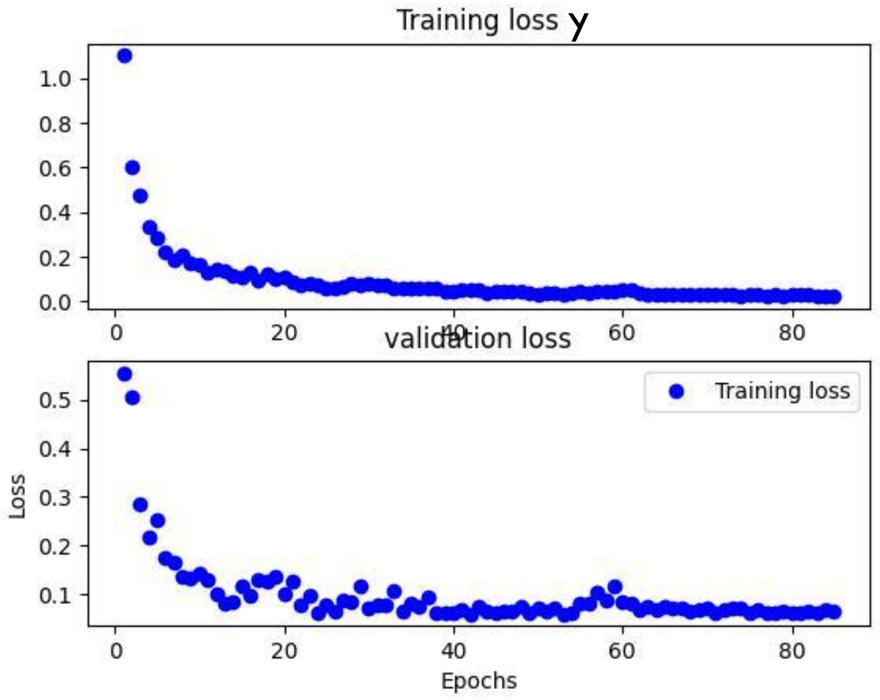






Ay





APPROACH 4 TRAIN AND VALIDATION LOSS

Restored image

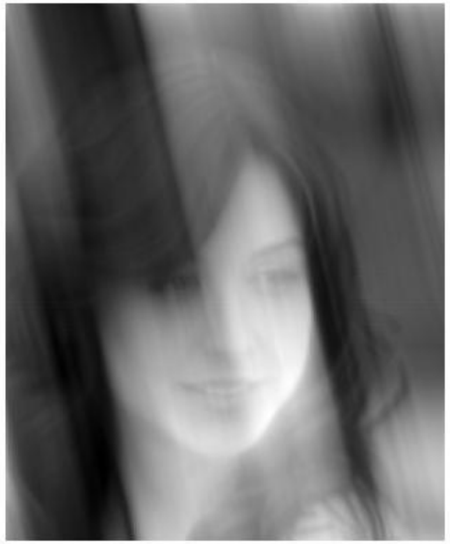
Blur Image

Restored image

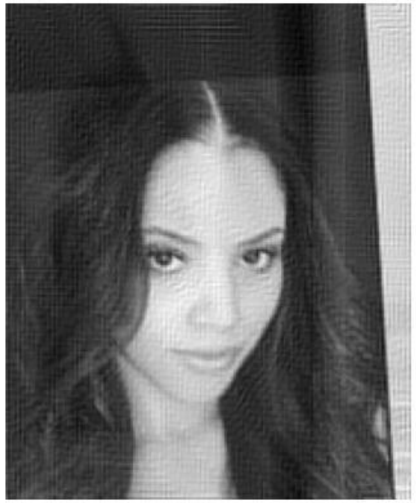
Blur Image

Restored image

Blur Image



Restored image



Blur Image