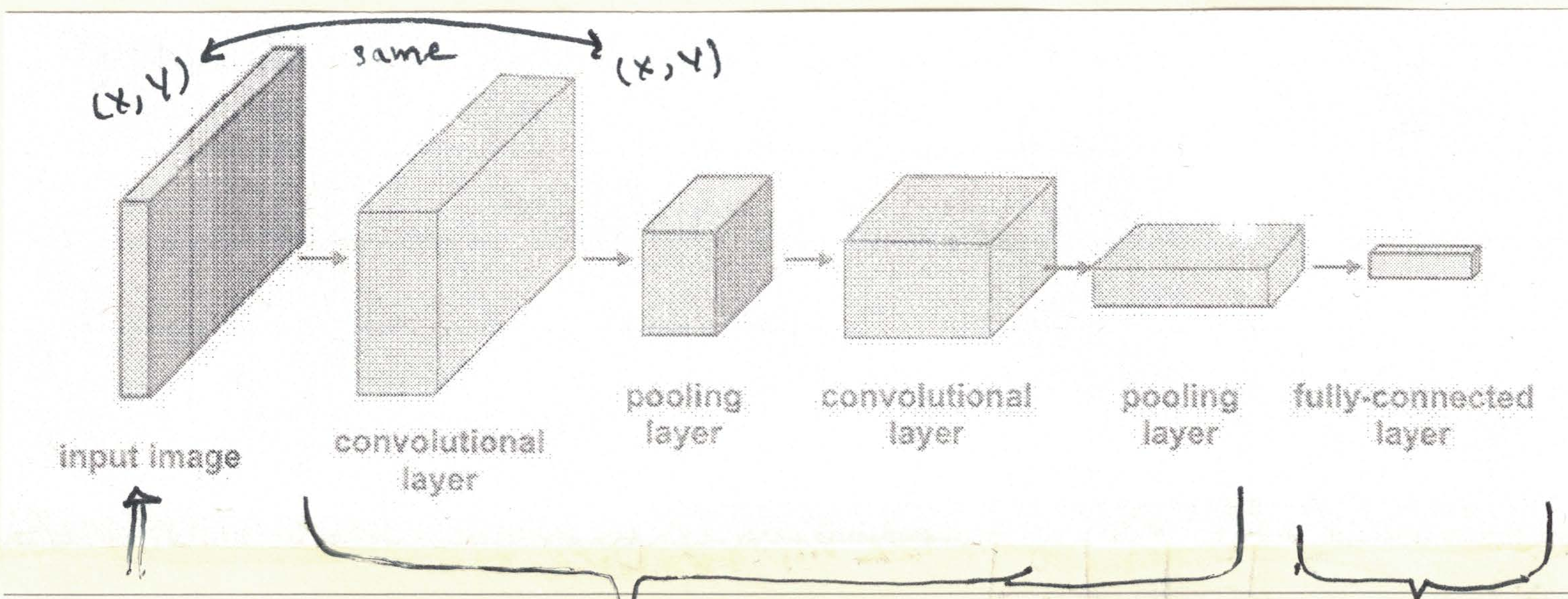


input

feature extraction

(isolation information from image)

Prediction

6/3/2020

Transfer learning: implementing feature extraction from succesful models and redining the dense/~~fu~~ fully-connected layer.

Dataset similarity

		Similar	Different
Dataset size	large	Fine-tune (2)	Fine-ture / retrain (3)
	small	End of ConvNet (1)	Start of convNet (4)

Cases:

1. Adjuct end of convNet
   - ~~remove and replace end of~~ Dense layers (add) (not remove dense)
     - randomize dense layer weights → update weights
     - freeze weights in pretrained network - don't update weights
       - ↳ prevent overfitting

Forward ~~Pass~~ are Propagation: All data travels forward through nodes to calculate the NN output

- Hidden are ~~Input~~ Layer: ~~Preprossesing~~ are non-linearity
  - Equation:
    * input: $x \cdot W$ data / weigt input to hidden
    * Output: activation fuction(input) sigmoid ~~δ~~ are σ

- output are ~~Hidden~~ Layer: non-linearity / Linear
  - Equ:
    * input: hidden output $\cdot W$ output from hidden layer / weights from hidden to output layer
    * output: activation / 1 $\cdot$ input or

Back propagation: data travels in opposite direction to fix/ajust weights & bais

- output layer:

~~Equ input:~~

Output error: $(y - \hat{y})$ target / output/prediction | label - prediction

Output error term: $(y - \hat{y}) f'(a_k)$ | $\underbrace{\text{label - prediction}}_{\text{(error)}}$ * $\underbrace{\text{derative of activation fuction of output}}_{\text{(gradient)}}$

hidden layer:

hidden error: ~~$W_{o}\delta$~~ are $W_{jk}\,\delta_k$ weigt / previous layer error term | Output layer term * weights hidden to output

hidden error term: $\Sigma[W_{jk}\,\delta_k]\,f'(h_j)$ | $\underbrace{\text{hidden layer error}}_{\text{(error)}}$ * $\underbrace{\text{derative of activation fuction of hidden output}}_{\text{(gradient)}}$

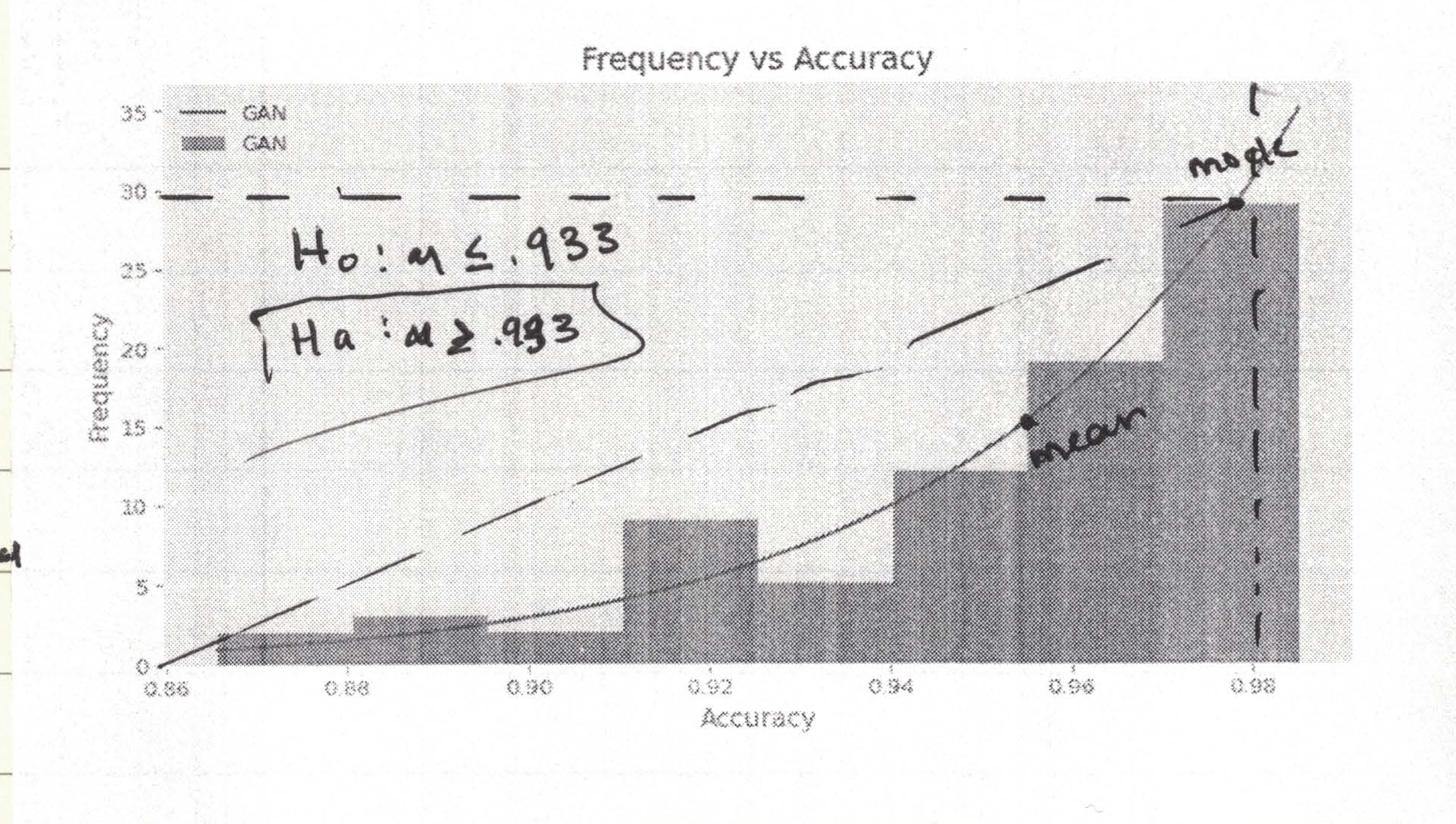
1/14/21

$\bar{x} = 0.9498$

$\sigma_x = 0.0293$

$t = \dfrac{0.9498 - 0.933}{0.0293/\sqrt{149}} = 6.99$ ave

$p = 1.08e^{-10}$



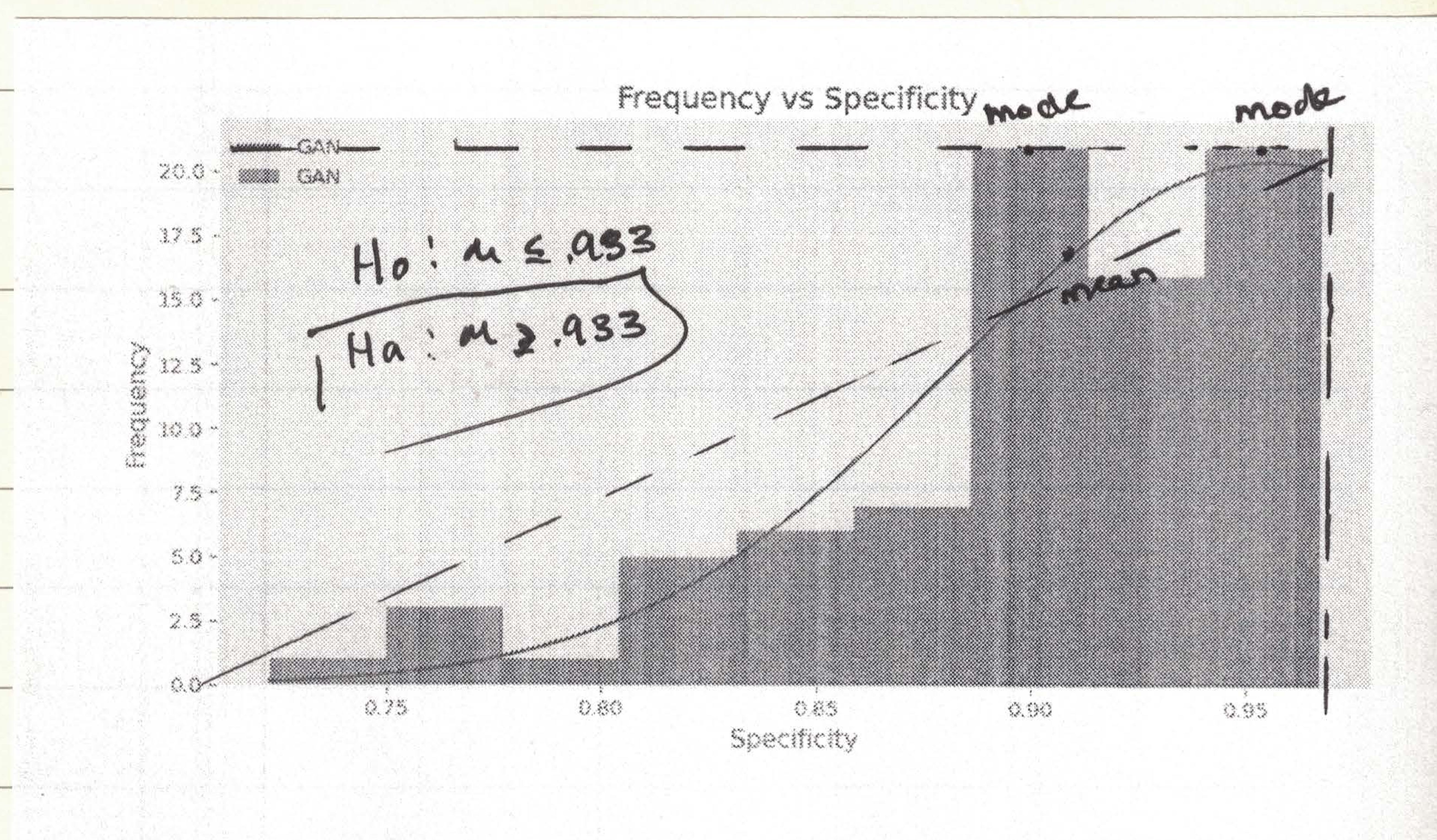
The distributions are skewed to the left, could ~~mean~~ that it is stable in performance.

$\bar{x} = \cancel{0.9952}\ 0.9030$ ave

$\sigma_x = 0.0547$

$t = \dfrac{0.903 - 0.933}{\frac{0.0547}{\sqrt{149}}} = -6.69$

$P = 1.65e^{-10}$



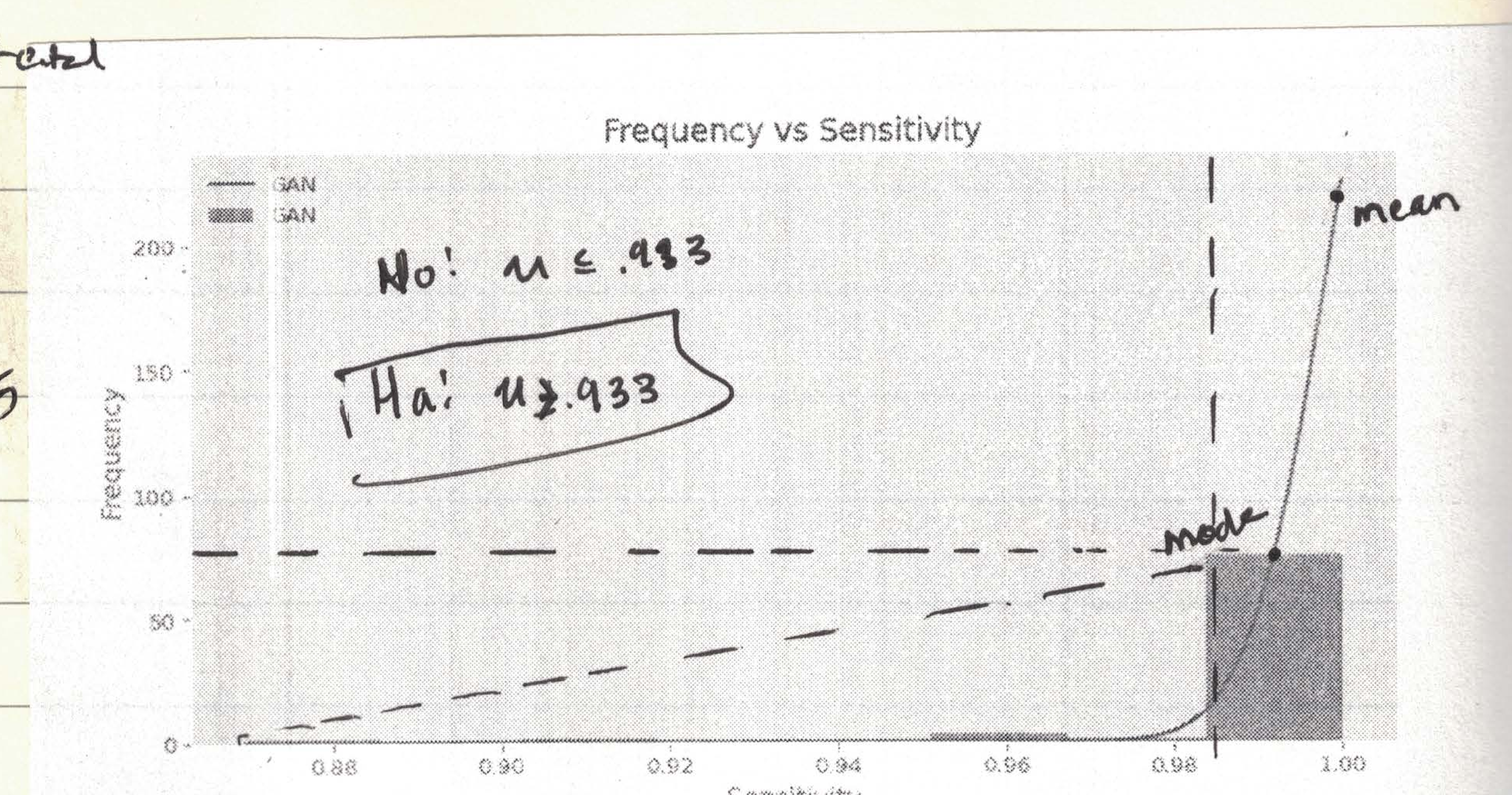
Sensitivity is maximize b/c we want to have all potential anomalies detected

$\bar{x} = 0.9952$

$\sigma_x = 0.0197$

$t = \dfrac{0.9952 - 0.933}{\frac{0.0197}{\sqrt{149}}} = 38.5$

$P = 1.54e^{-77}$



Filters & Edges in CNNs:

- low pass filters : block high frequencies (v/c versa)
- using high pass filters - can block low pass freqs in image
  - used for edge detection
- convolutional kernal : edge detection filter

weights

" #'s

ex

n_{ij}

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

$\sum n_{ij} = 0$

$(-1) + (-1) + (-1) + (-1) + (+4) + 0 + 0 + 0 + 0 = 0$

~~ya~~ process $\Downarrow$

Kernel $\Rightarrow$ $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ $*$ Image data $\Rightarrow$ $\begin{bmatrix} 3 & 4 & 7 \\ 8 & 5 & 6 \\ 9 & 11 & 10 \end{bmatrix}$

convolution (astrik)

numbers multiplied

* main same ~~depth~~ volume

Final $\Rightarrow$ $\begin{bmatrix} 0 & -4 & 0 \\ -8 & 20 & -6 \\ 0 & -11 & 0 \end{bmatrix}$

$\sum n_{ij} = -9$

$(-4)+(-6)+(-8)+(-11)+20 = -9$

Convolution Layer:

- applies many convolutional kernels
- stride the distance from pixel to pixel

- -9 becomes center pixel value (5) in data image

Max pooling - used for decreasing complexity b/c overfitting

Avg Pooling

* capsel network