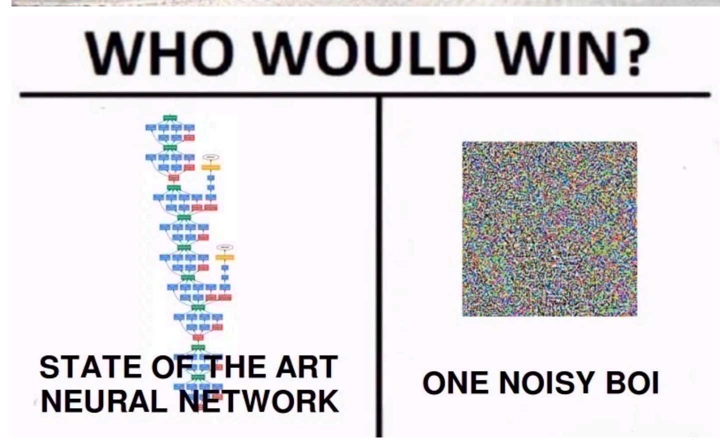
March 31, 2023

DSML: Computer Vision.

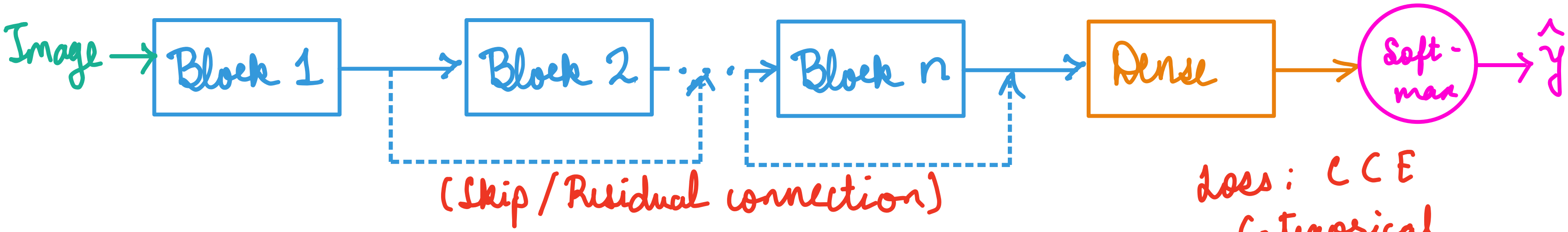
CNNs for Medical Diagnosis

Class starts @ 9:05 pm.



Recap:

* Basic CNN architecture for Image Classification:



Loss: CCE
Categorical
Cross-Entropy.

* Popular architectures and their performance:

Model Name	Number of params	Top 1 Acc	Top 5 Acc
EfficientnetB0	5.3M	77.3	93.5
MobileNet	2.3M	71.0	90.5
ResNet50	25.6M	83.2	96.5
Inception	22.9M	79.0	94.5
VGG16	138M	74.4	91.9
AlexNet	62M	63.3	84.6

Novelty

Automated scaling strategy.

Factorizing convolutions.

Residual / skip connections.

Inception module, 1x1 conv

3×3 kernels, blocks.

First method to use GPUs, SGD.

Recap:

* Major CNN related concepts:

(a) Transfer learning: Reuse an already trained CNN for image classification.

→ Helpful when we have a small dataset.

→ As long as images are similar to Imagenet, better than training from scratch.

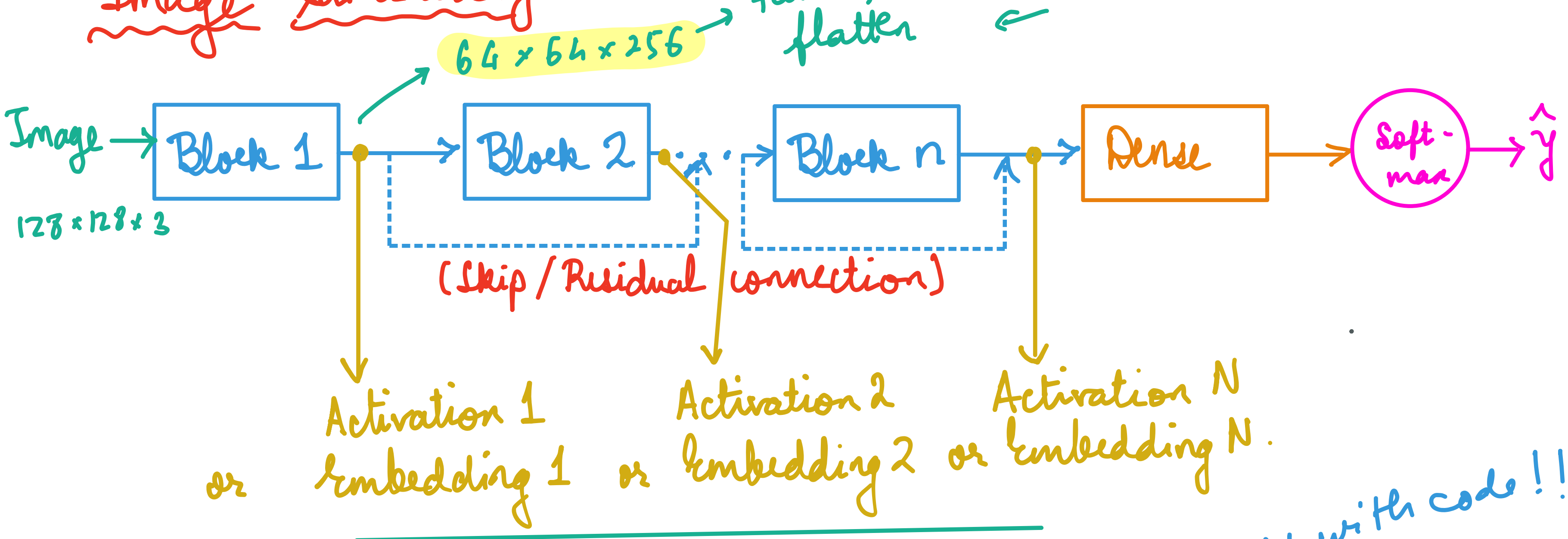
(b) Image similarity: Reuse of an already trained CNN for unsupervised similarity search.

→ Select a convolutional representation. This is referred to as "activations" or "embeddings".

→ Store these image representations in a database optimized for KNN.

→ Similar images have "activations" which are close in the Euclidean sense.

Image Similarity:



* Considerations:

(a) Which neural network to pick? → go to papers with code!!
size, accuracy, dataset it was trained on, etc.

(b) Which activation / embedding to pick?
compare all, but usually, the best is the last layer before dense.

(c) How to preprocess existing data?
Create a database containing all the activations & embeddings.

(d) How to query? Basically apply KNN (euclidean distance)

Since we have a medical application:

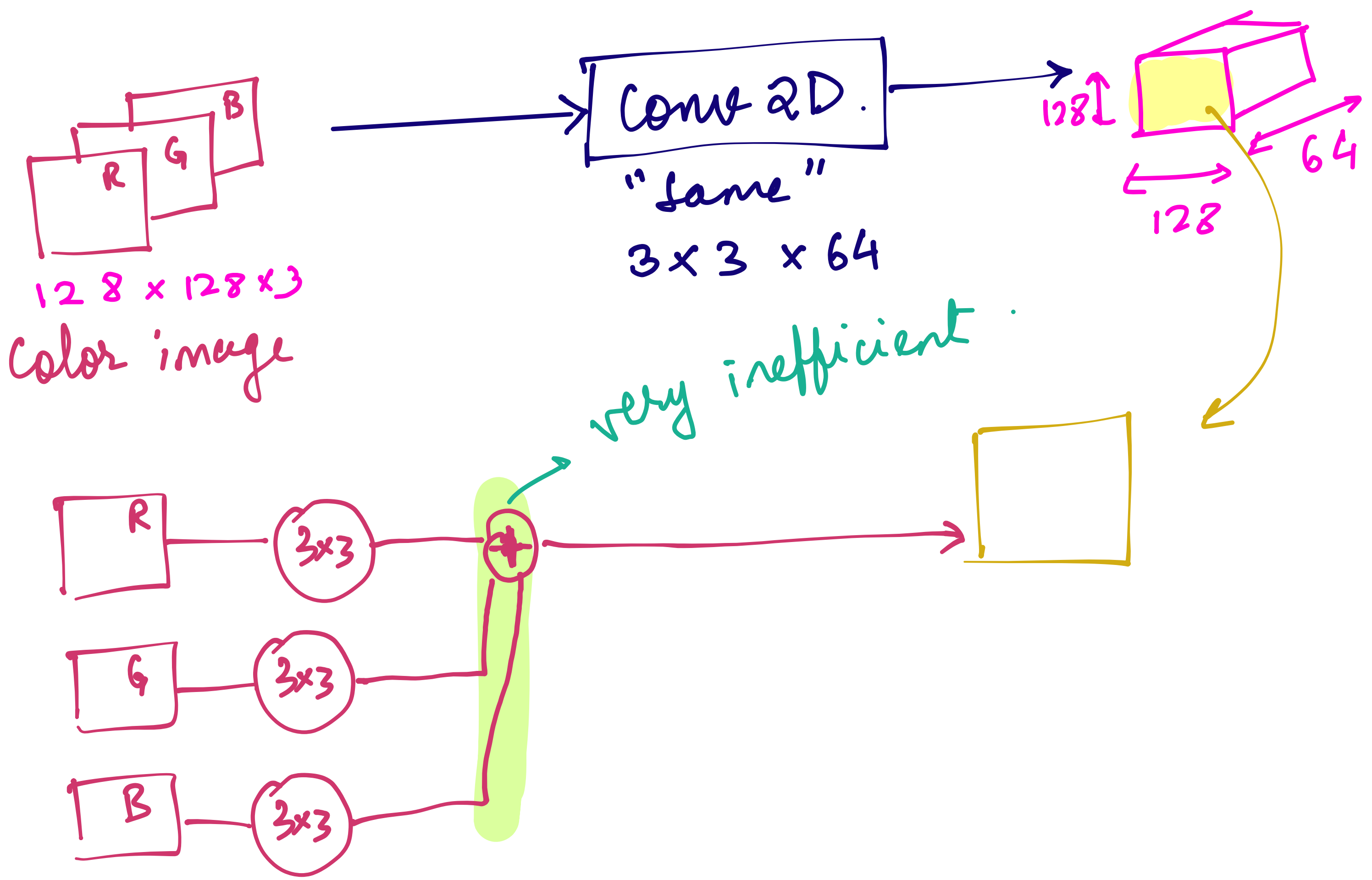
precision $\rightarrow \dfrac{T.P}{T.P. + F.P.}$

recall $\rightarrow \dfrac{TP}{TP + FN}$

This has to be high.

Mobile Net:

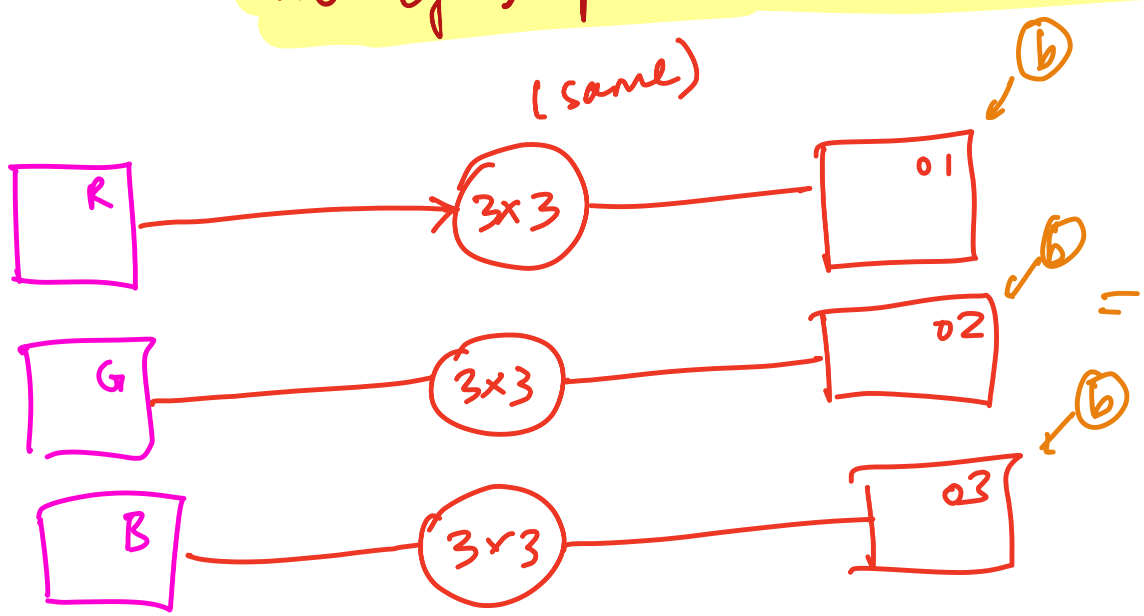
* The basic change is the introduction of Depthwise convolutions.



Depthwise Convolution.

→ We no longer need multiple kernels for getting the output.

→ For depthwise convolutions,
no. of input channels = = no. of output channels.



no. of kernels = no. of input channels.

Class Imbalance problem:

→ In general, we have to collect more data to balance them out.

→ If this cannot be done, we add weightage to the training examples. Assume more +ve examples than -ve examples.

if positive_class:

$$x^{(t+1)} = x^{(t)} - \eta \cdot grad$$

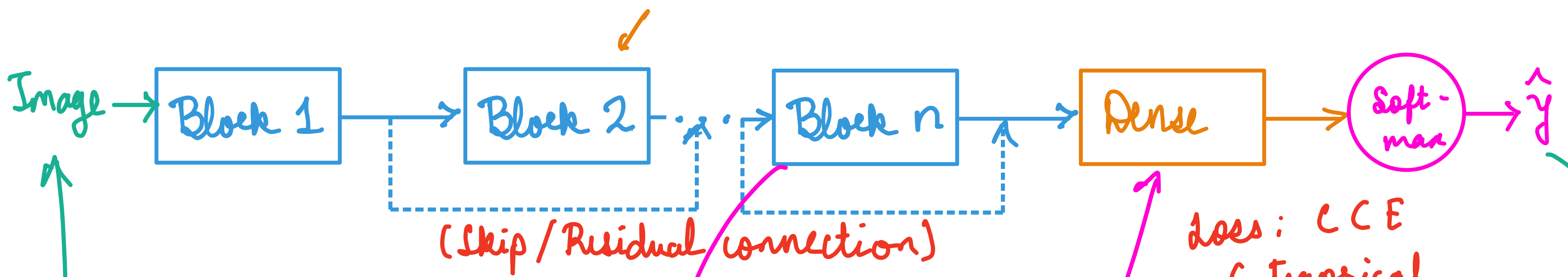
else

$$x^{(t+1)} = x^{(t)} - \underbrace{3}_{\text{weight}} \cdot \eta \cdot grad.$$

How to identify what part of the image led to the decision?

* The biggest problem with CNNs is being able to debug it.

Can I ask why the CNN is 36% confident about cat?



Loss: CCE
Categorical Cross-Entropy.

pneumonia

pneumonia

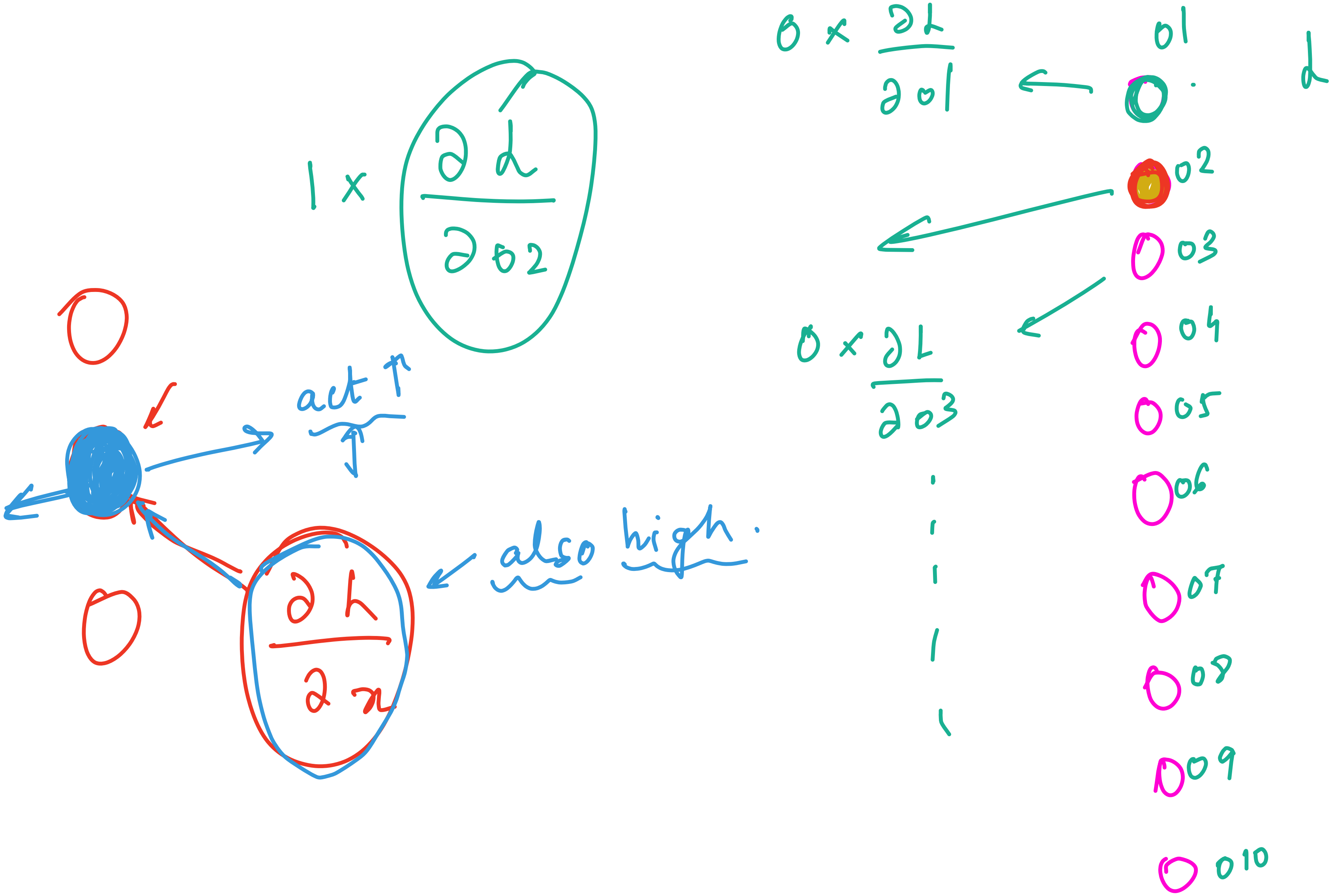
0.99.

which feature maps here contributed the most to those 5

which neurons here had the major contribution?

(Select top 5 output neurons).

Backpropagation already does this!!



$0 \times \frac{\partial L}{\partial o1}$
o1
L
o2
o3
o4
o5
o6
o7
o8
o9
o10
$1 \times \frac{\partial L}{\partial o2}$
$0 \times \frac{\partial L}{\partial o3}$
act ↑
$\frac{\partial L}{\partial x}$
also high.

Customize VGG-16

Complete the given code snippet by filling **1)**, **2)** and **3)**. You have a dataset containing images of shape 100x100x3. The model you are building will be used to classify those images into 7 classes. You are using a pre-trained VGG-16 model for this purpose.

```
pretrained_model = tf.keras.applications.VGG16(weights='imagenet', include_top=False , input_shape=1)  )

pretrained_model.trainable=False

vgg16_model = tf.keras.Sequential([

   pretrained_model,

   2) ,

   tf.keras.layers.Dense(7, activation=3) )

])
```