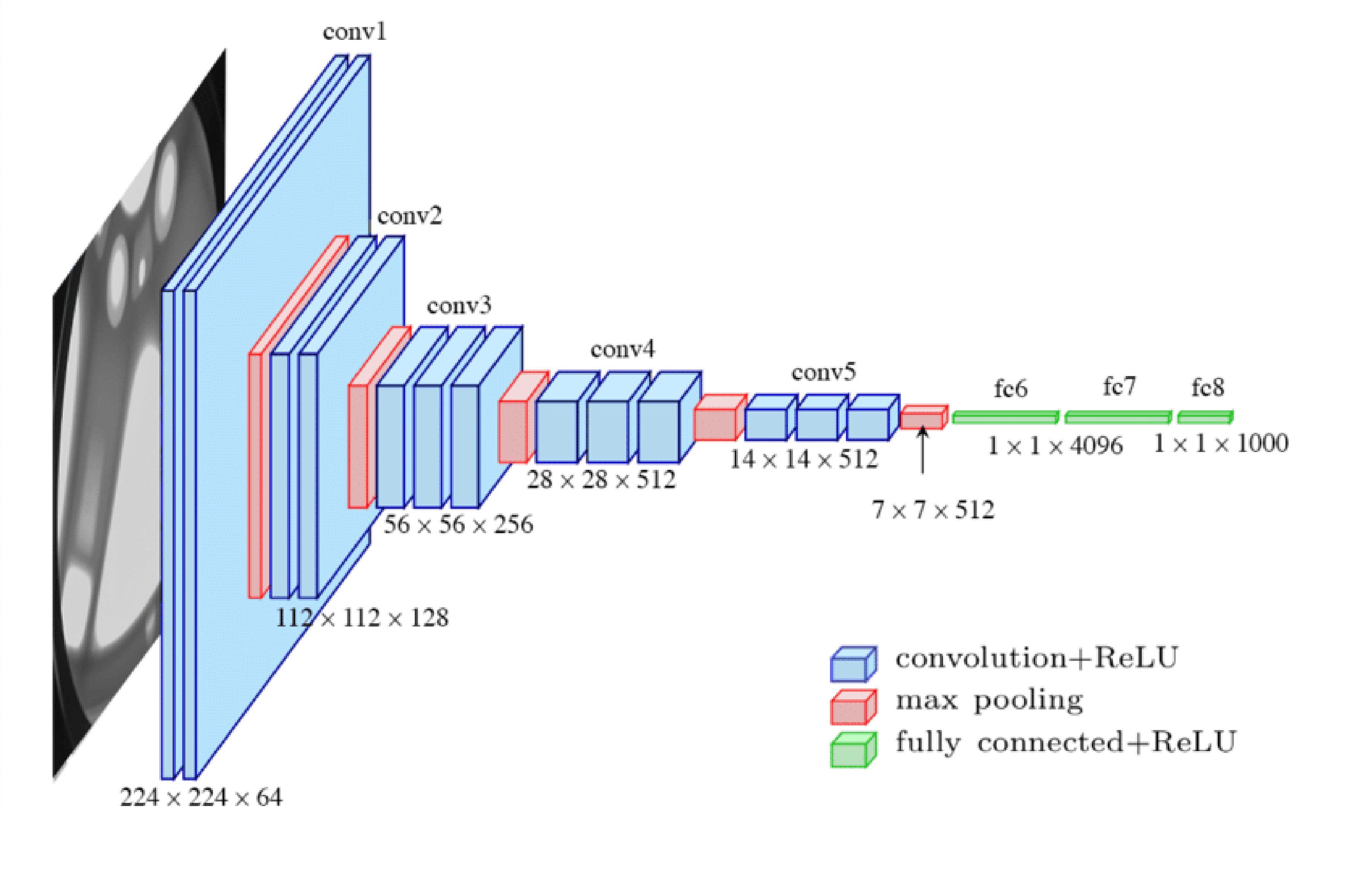
POPULAR MODELS

MOHAMMAD GHODDOSI

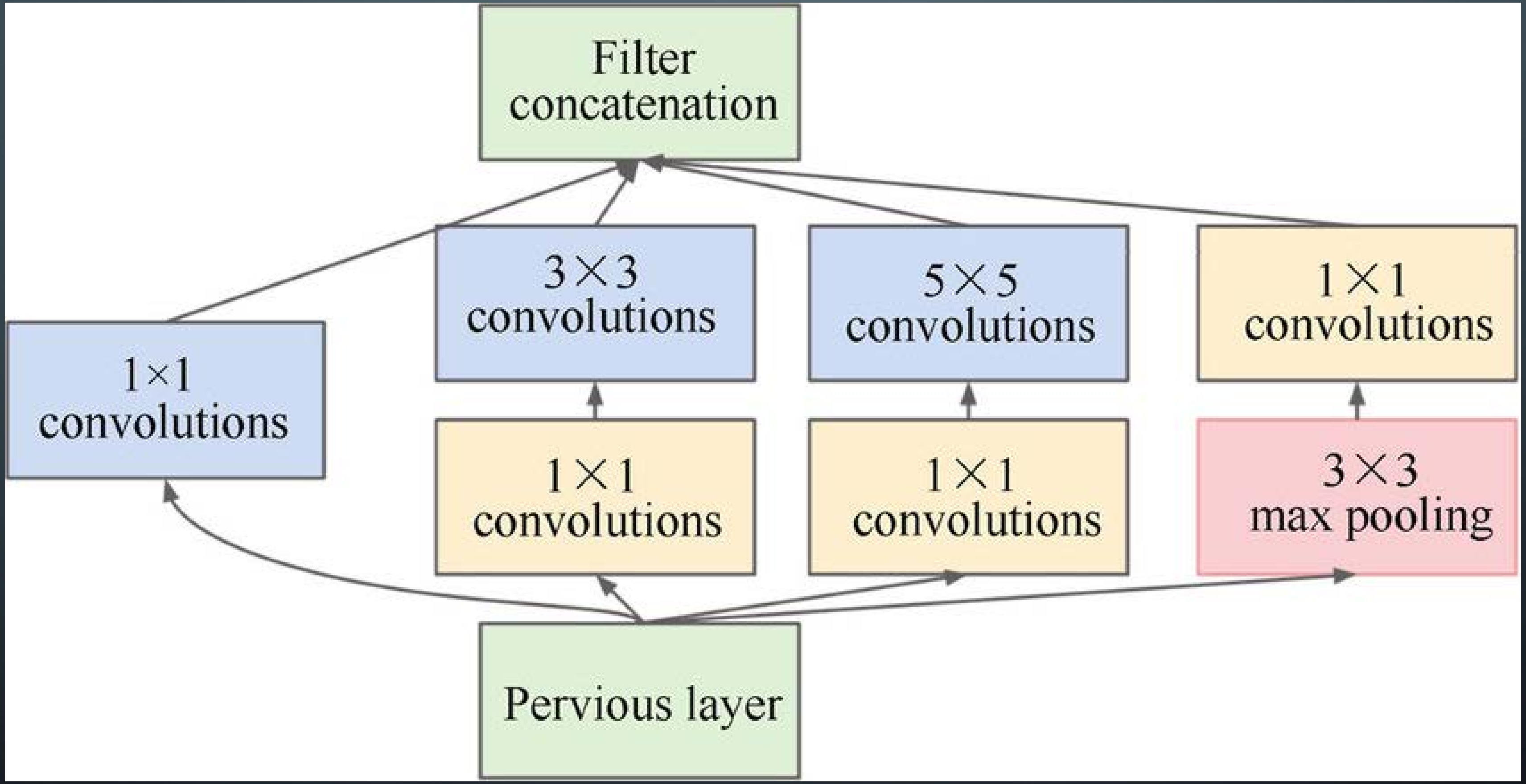
VGG16


conv1
conv2
conv3
conv4
conv5
fc6
fc7
fc8
224 × 224 × 64
112 × 112 × 128
56 × 56 × 256
28 × 28 × 512
14 × 14 × 512
7 × 7 × 512
1 × 1 × 4096
1 × 1 × 1000
convolution+ReLU
max pooling
fully connected+ReLU

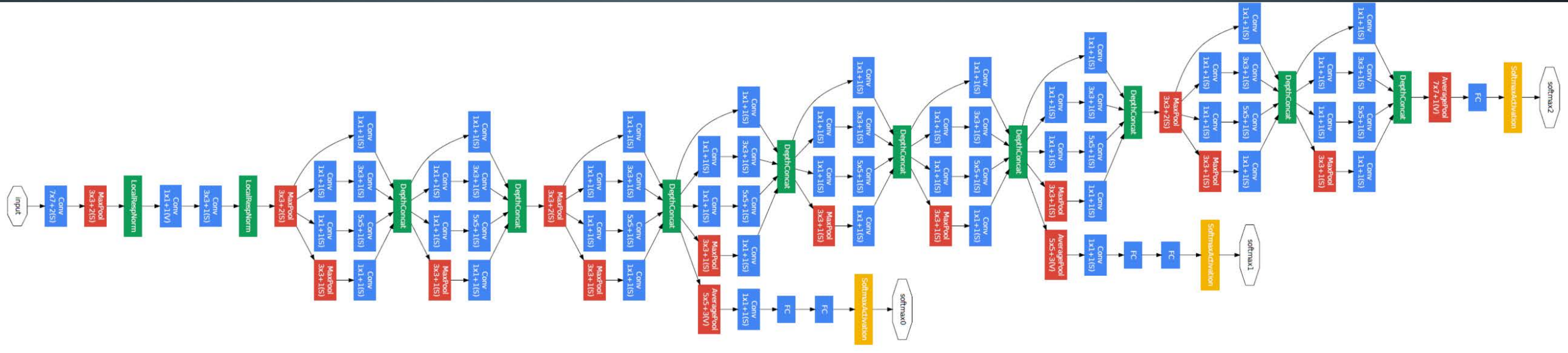
GOOLENET (IDEAS)

- We need to go deeper
  - Vanishing gradient
- 1x1 Convolution
  - Dimension reduction
- Inception module
- Global Average Pooling
- Auxiliary Classifiers

GOOGLENET (INCEPTION MODULE)

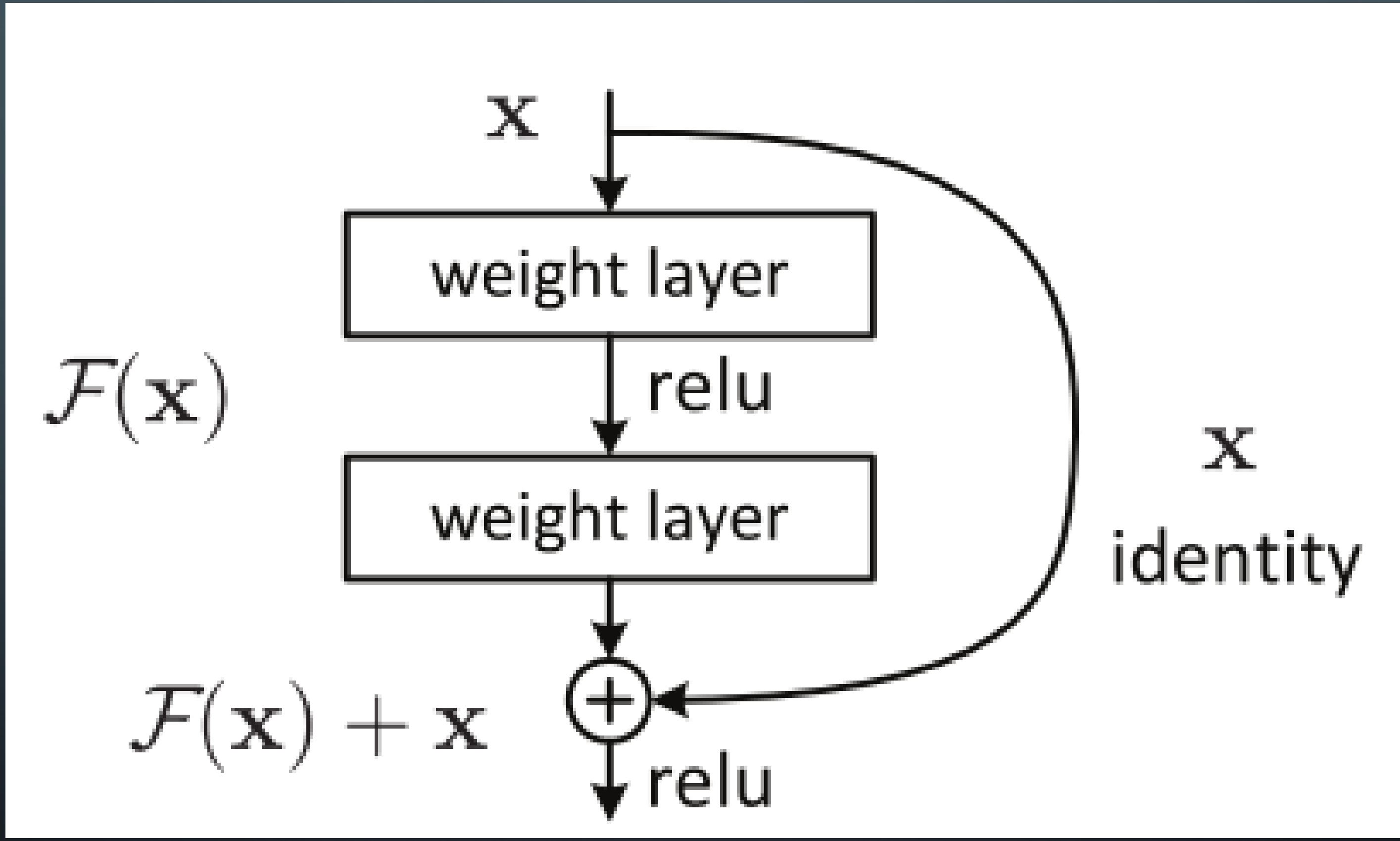


GOOGLENET



type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

RESNET (RESIDUAL BLOCK)



RESNET

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				

MOBILENET (IDEAS)

- Significant reduction in number of parameters. (same depth)
- Lightweight deep neural network
- Depthwise separable convolution
  - Depthwise convolution
  - Pointwise convolution

MOBILENET (CONVOLUTION)

- We can separate dimensions:

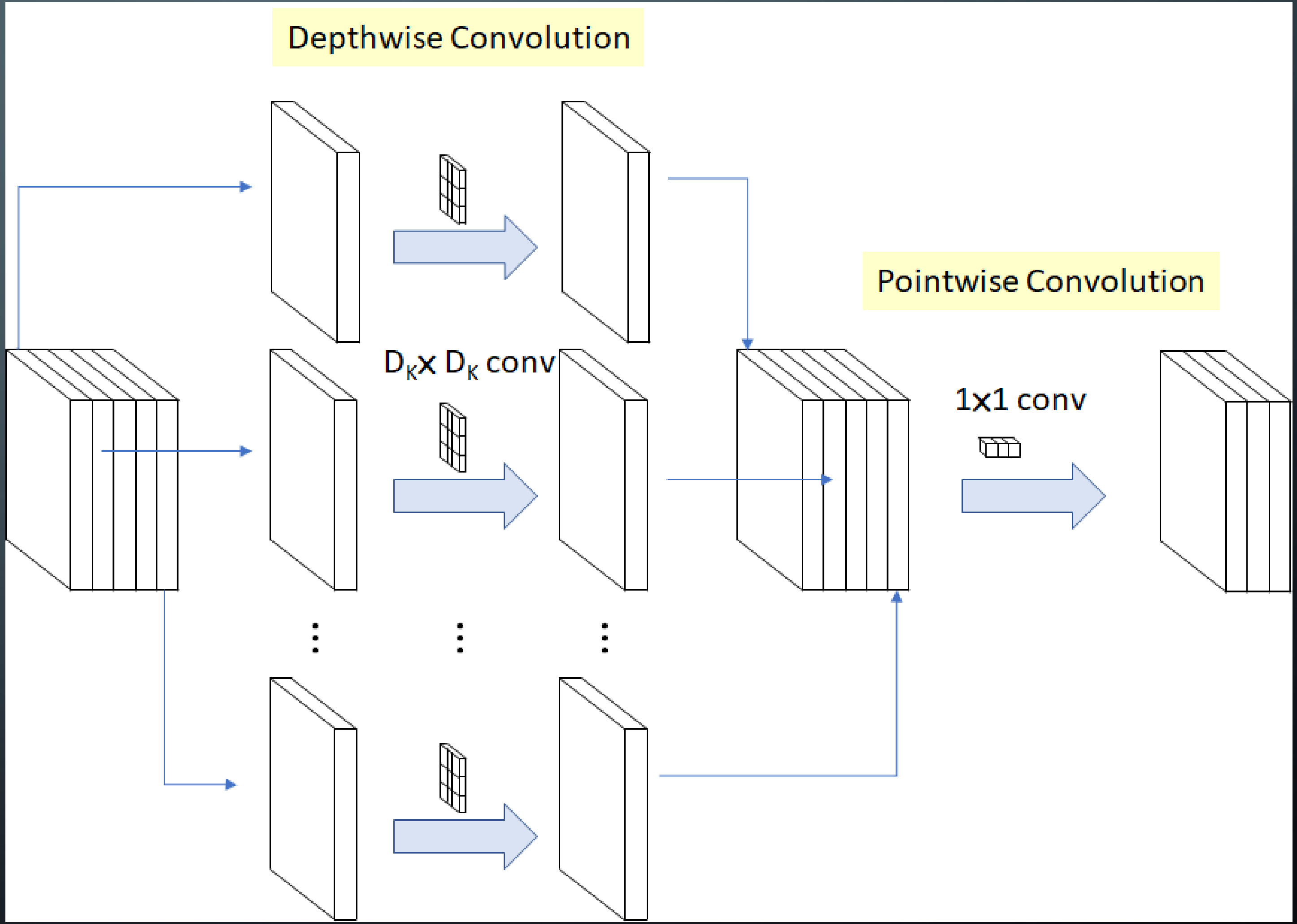
$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} . \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$$

- From 9 parameters to 6 parameters
- Lower operations
- We can apply this for depth to

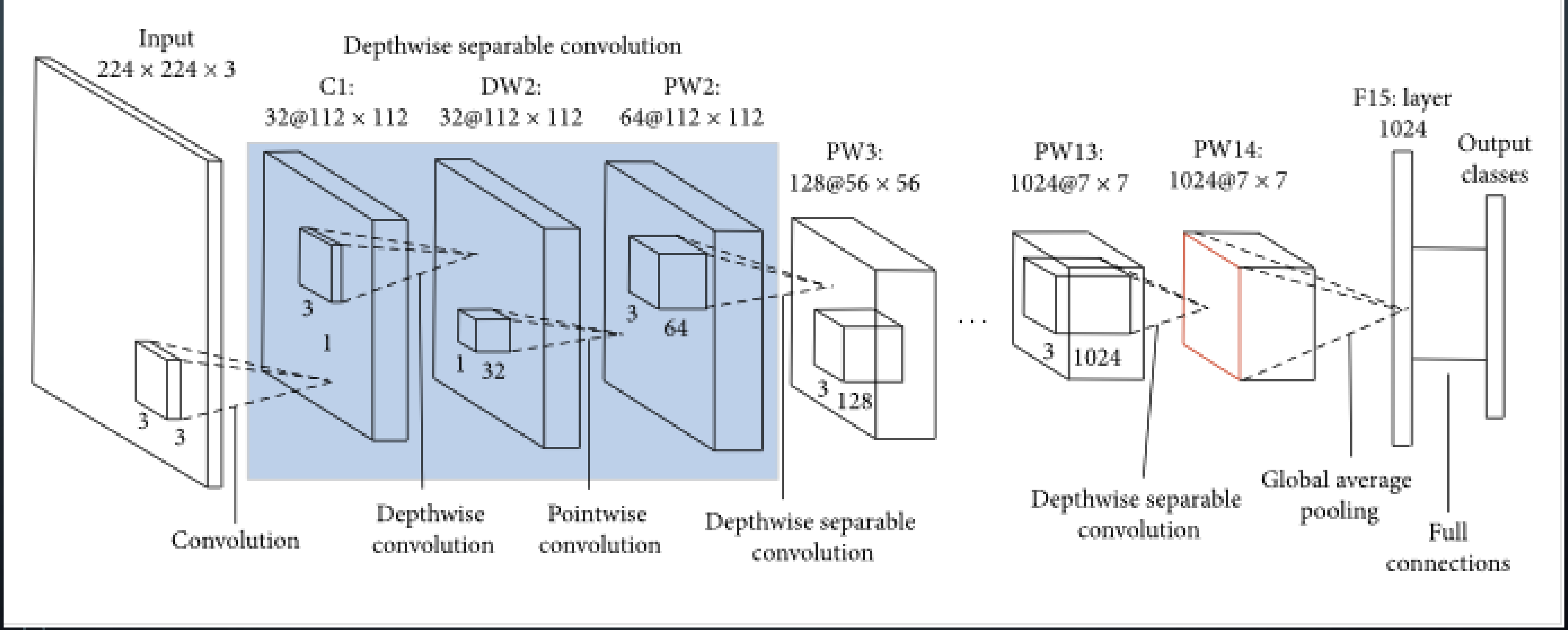
MOBILENET (CONVOLUTION)

Example:

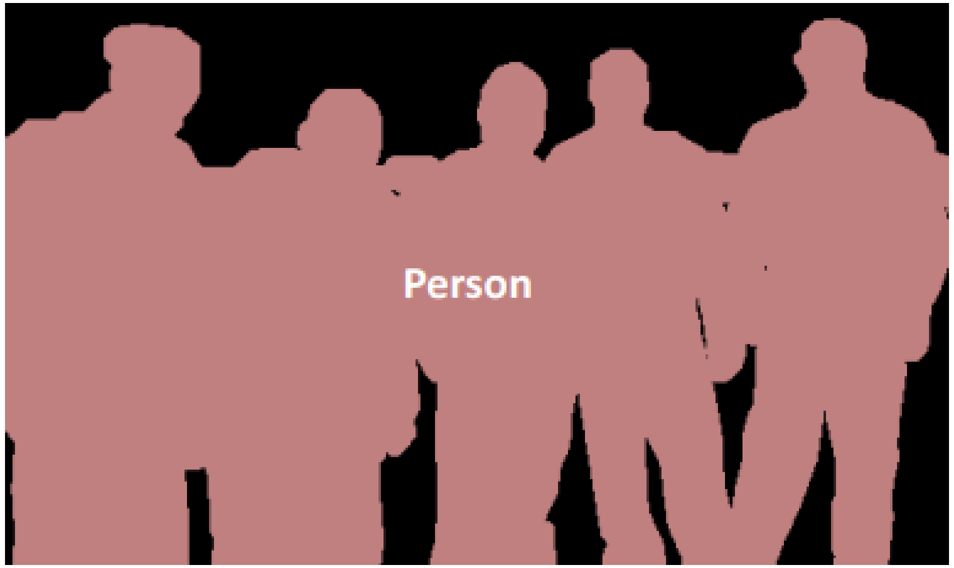
- 512*512 input image
- 3*3 filters
- 64 input depth
- 128 output depth
- Params:
  - Normal => 73,728
  - MobileNet => 704
- Operations:
  - Normal => 38.65e9
  - MobileNet => 4.59e9



MOBILENET


Input
224 × 224 × 3
Depthwise separable convolution
C1:
32@112 × 112
DW2:
32@112 × 112
PW2:
64@112 × 112
PW3:
128@56 × 56
PW13:
1024@7 × 7
PW14:
1024@7 × 7
F15: layer
1024
Output
classes
3
3
3
1
1
32
3
64
3
128
3
1024
Convolution
Depthwise
convolution
Pointwise
convolution
Depthwise separable
convolution
Depthwise separable
convolution
Global average
pooling
Full
connections

SEGMENTATION



Semantic Segmentation

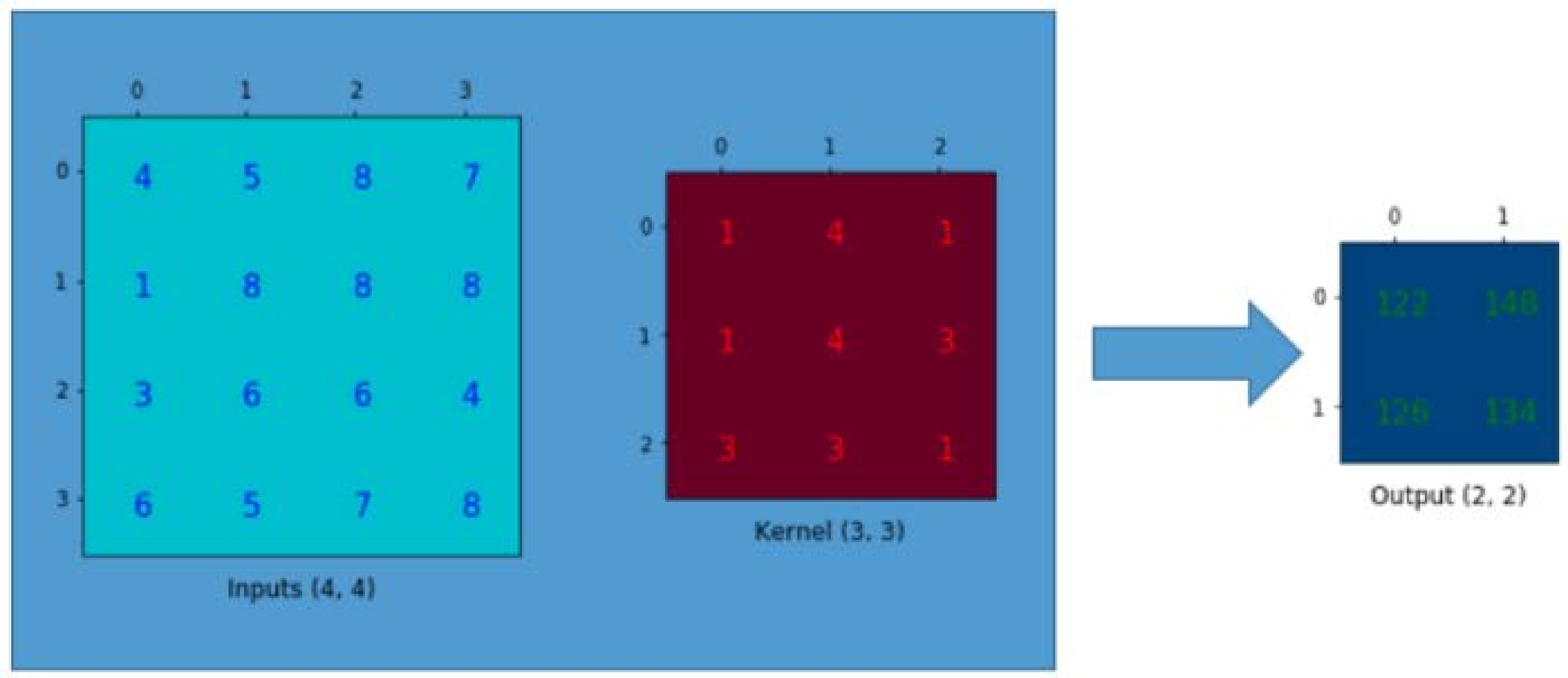
U-NET OPERATIONS

- Convolution
- Pooling (down sampling)
  - Lose “WHERE” information
  - We need “WHERE” and “WHAT” information both
- Up sampling
  - Transpose Convolution (deconvolution)
    - Convolution with fraction stride
- Skip connection

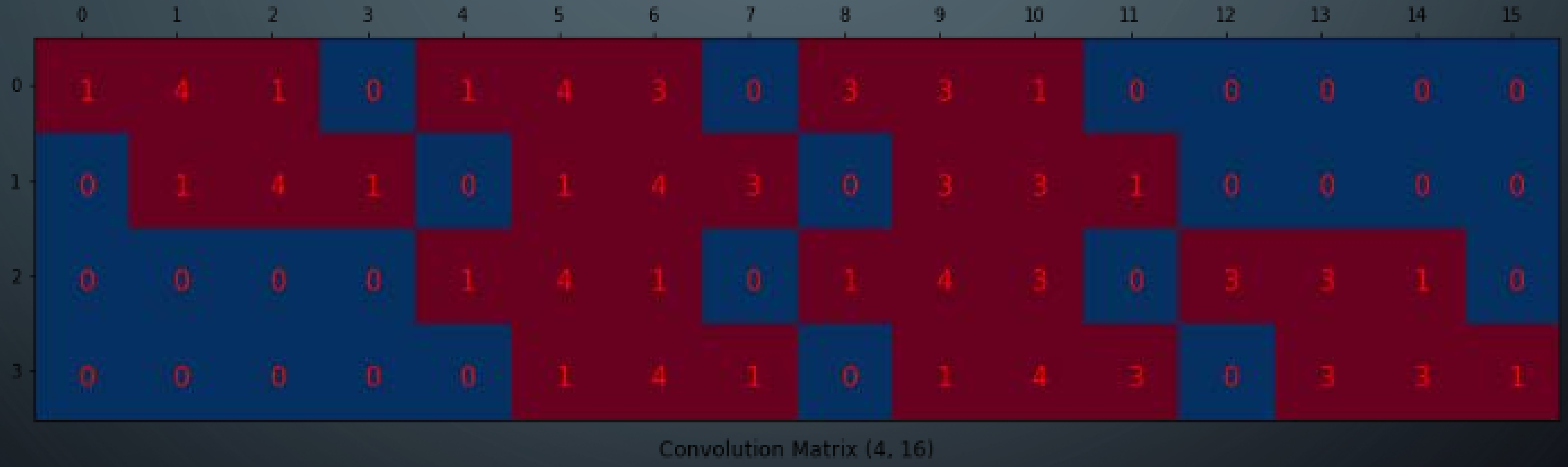
TRANSPOSE CONVOLUTION

- For matrices we know:
  - $A_{1*100}.B_{100*3} = C_{1*3}$
  - $C_{1*3}.B^T_{3*100} = D_{1*100}$
- We write convolution as dot product and
- Then write it the other way around

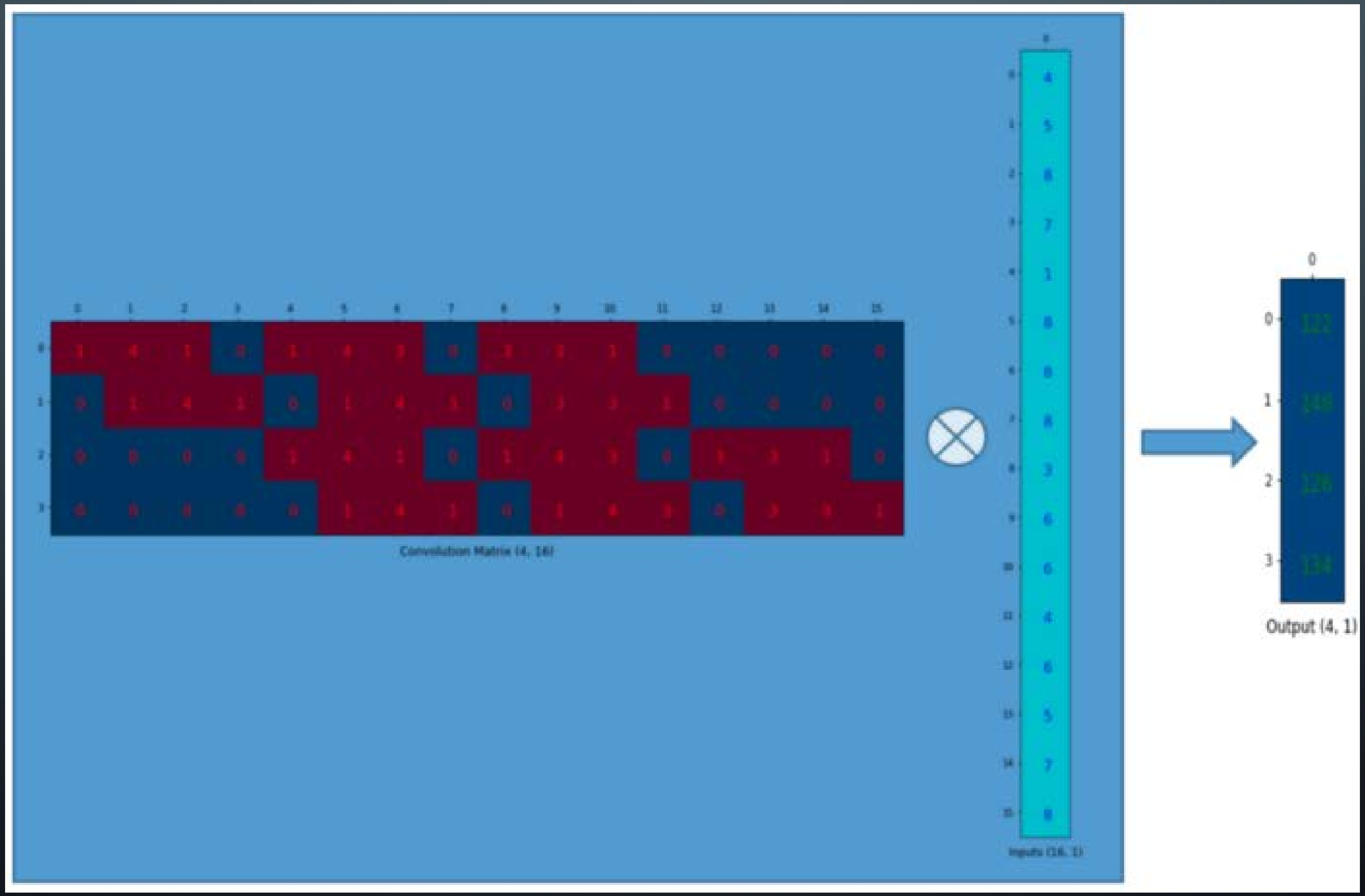
CONVOLUTION



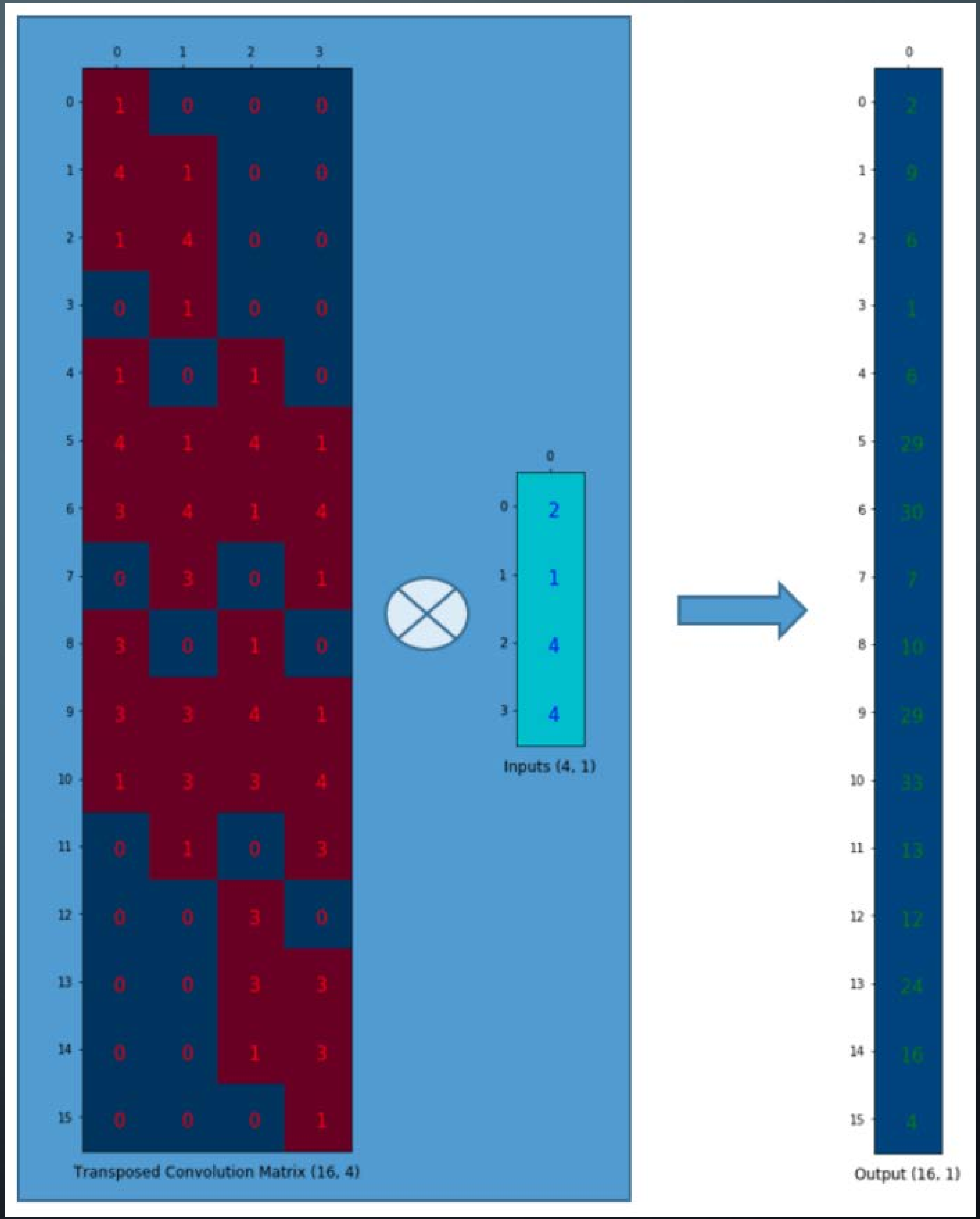
CONVOLUTION AS DOT PRODUCT



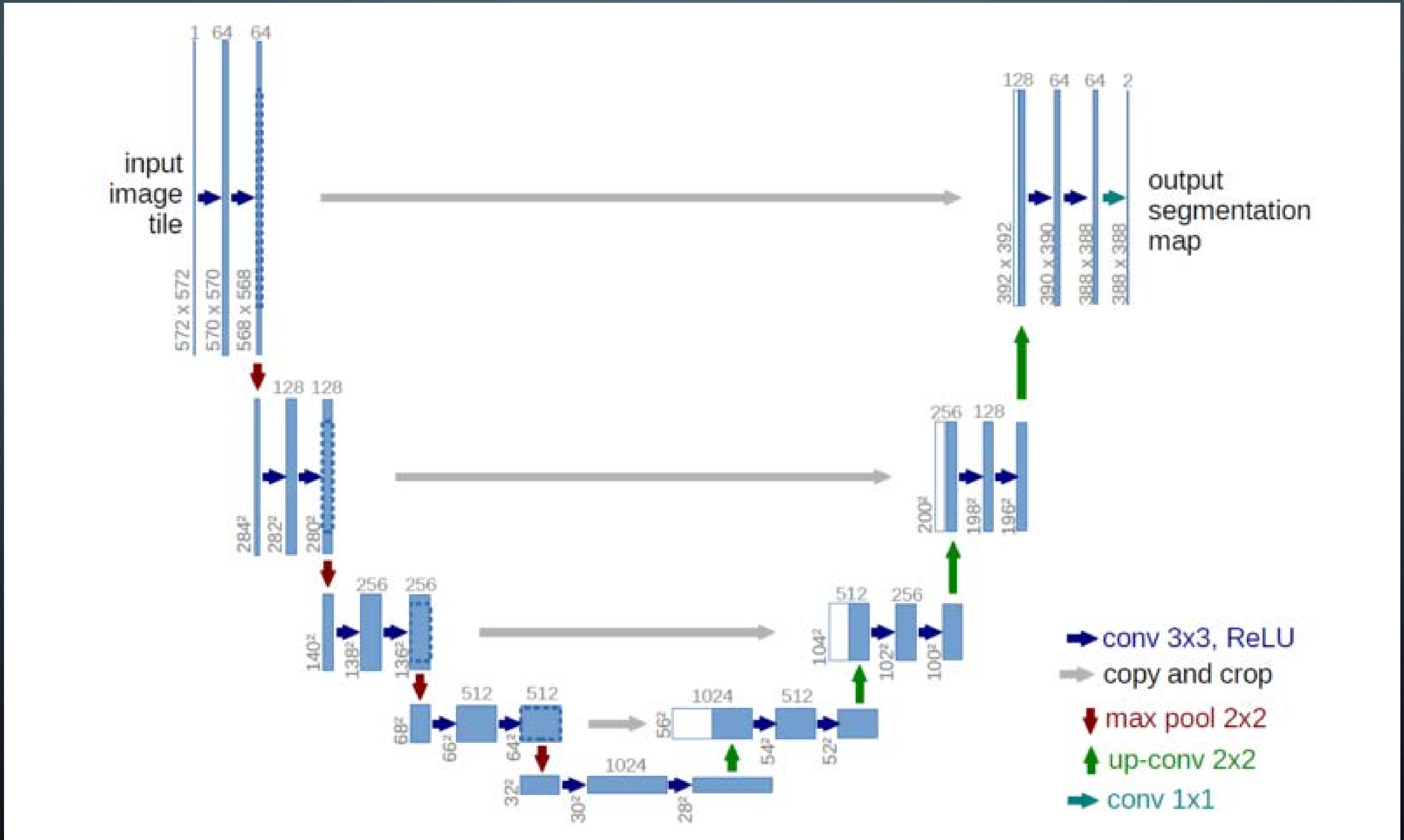
CONVOLUTION AS DOT PRODUCT



TRANSPOSE CONVOLUTION



U-NET ARCHITECTURE



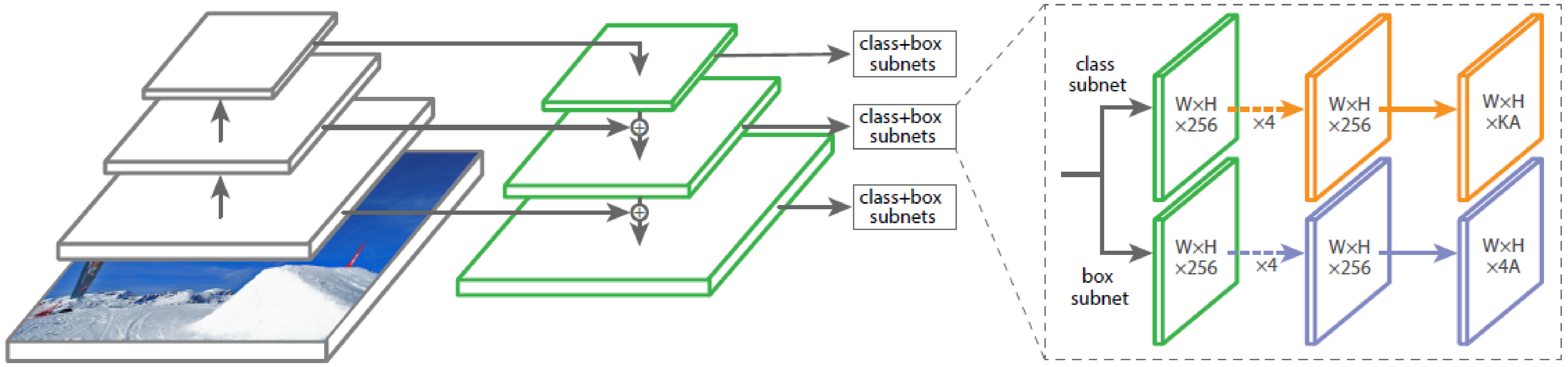
OBJECT DETECTION

RETINANET

- Focal loss
  - $FL(p_t) = -(1 - p_t)^{\gamma} \log(p_t)$
  - Lower weight to easy samples
  - Focus training on hard negatives
- Non maximum suppression

$$p_t = \begin{cases} p & if\ y = 1 \\ 1 - p & if\ y = 0 \end{cases}$$

RETINANET



(a) ResNet (b) feature pyramid net (c) class subnet (top) (d) box subnet (bottom)

FURTHER READING

- LeNet
- AlexNet
- ZFNet
- Mask R-CNN
- YOLO (You Only Look Once)
- SSD (Single Shot Detector)

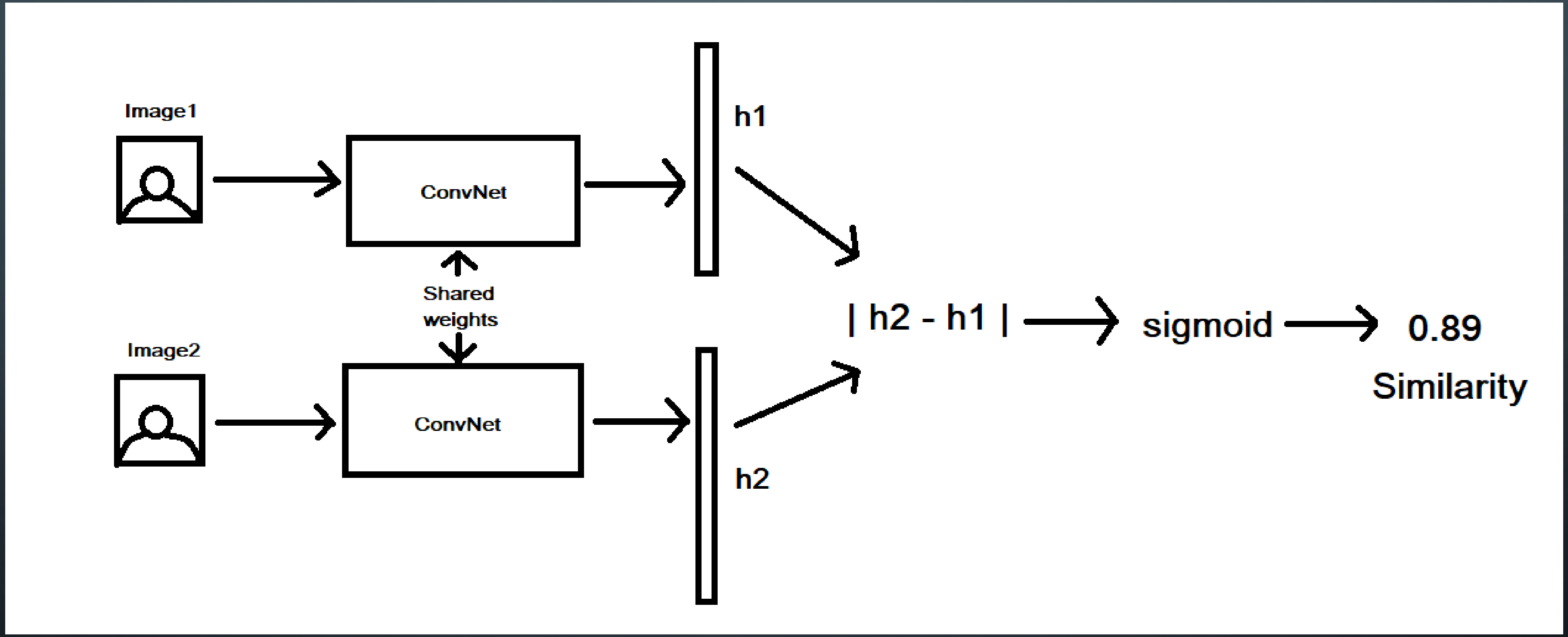
IDENTIFICATION

- Verification
- Identification

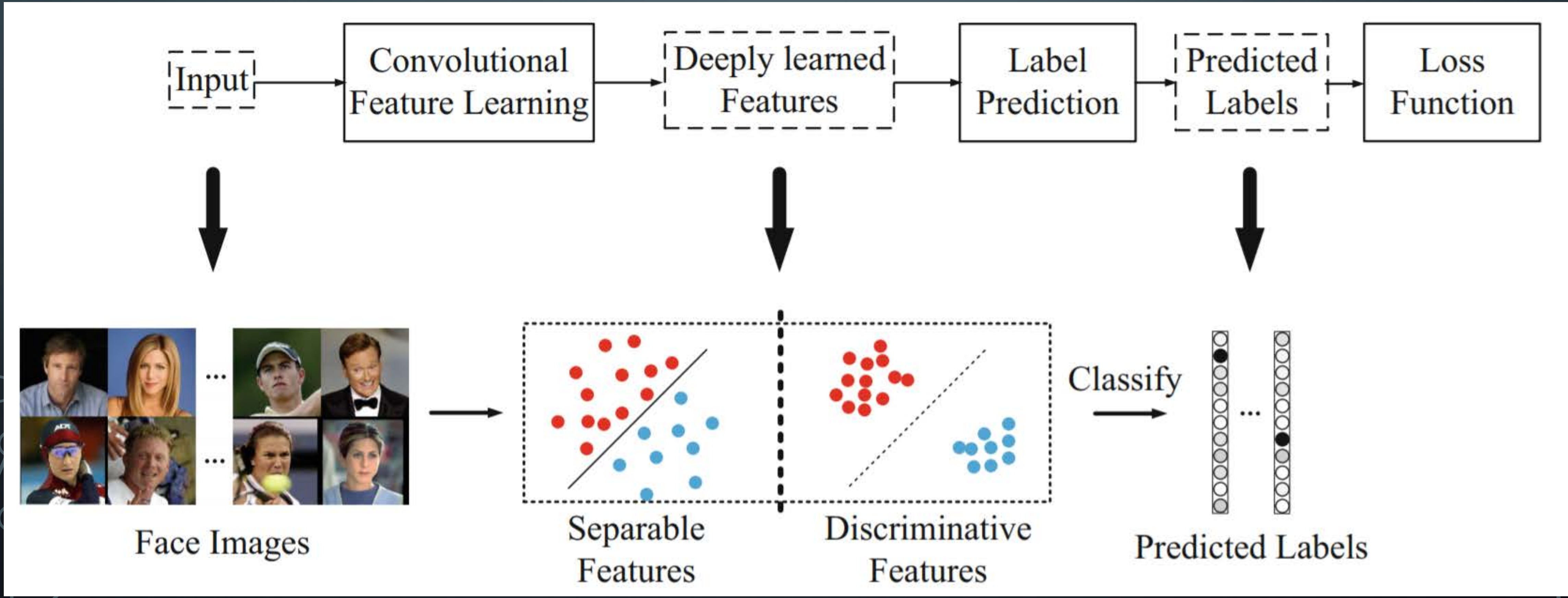
IDENTIFICATION AS CLASSIFICATION

- Each identity is a class
- We need lots of images for each person
- Its not easy to add one person
- Solution -> one shot learning

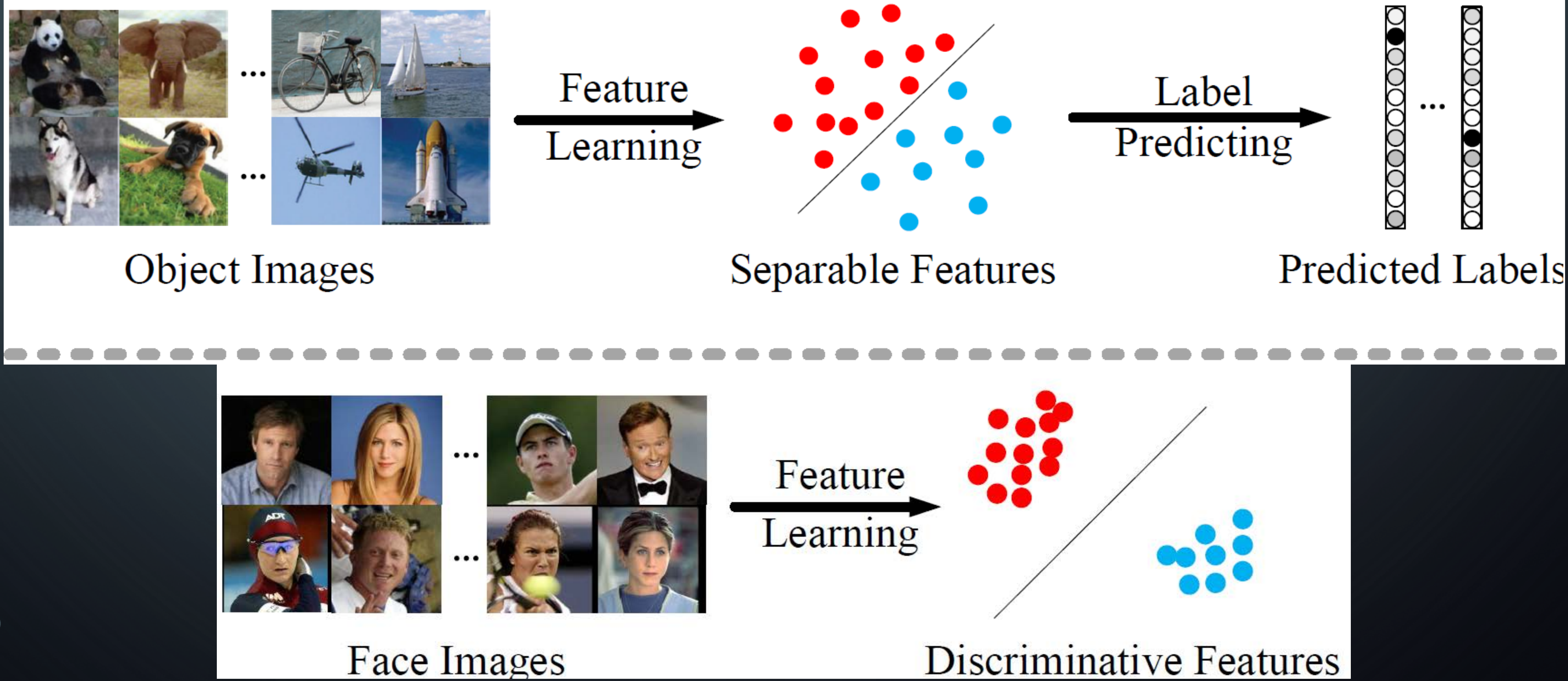
SIAMESE NETWORK



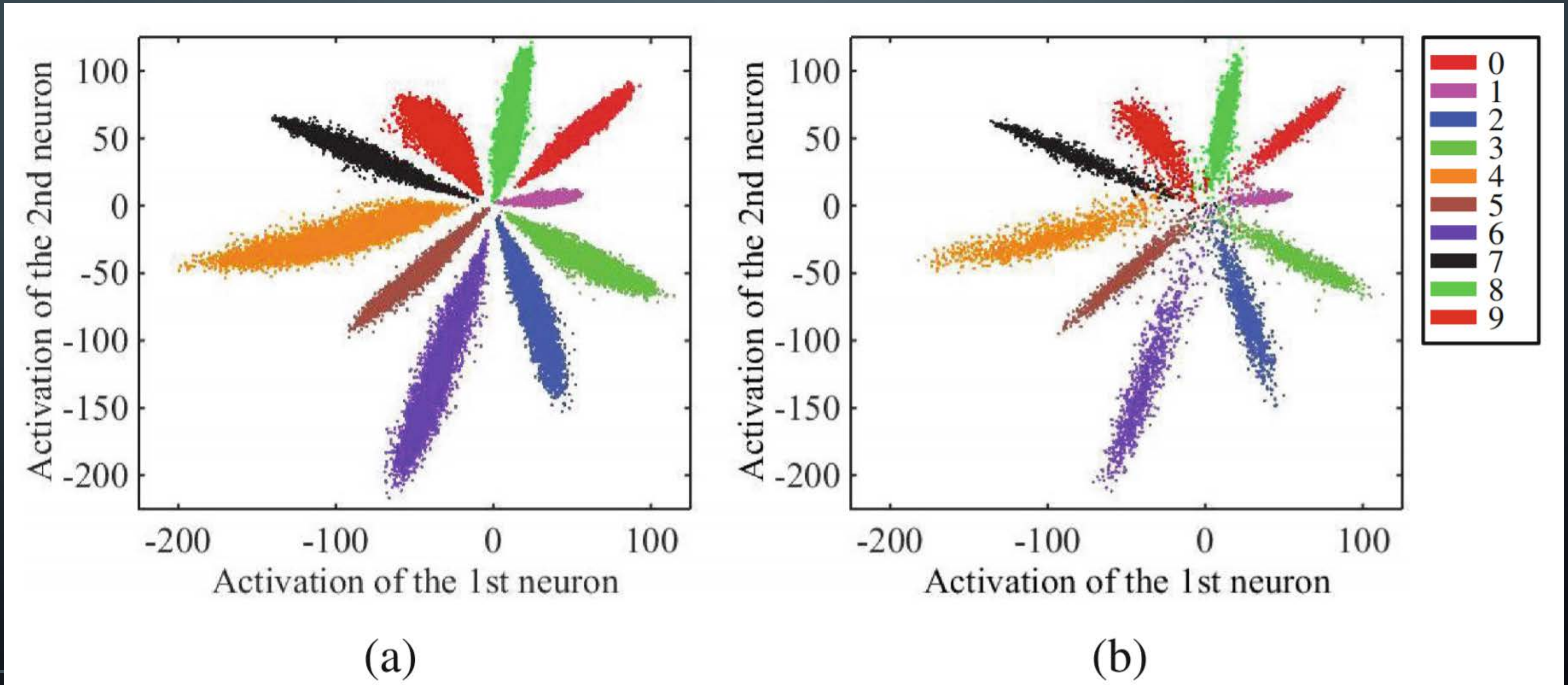
DISCRIMINATIVE VS SEPARABLE



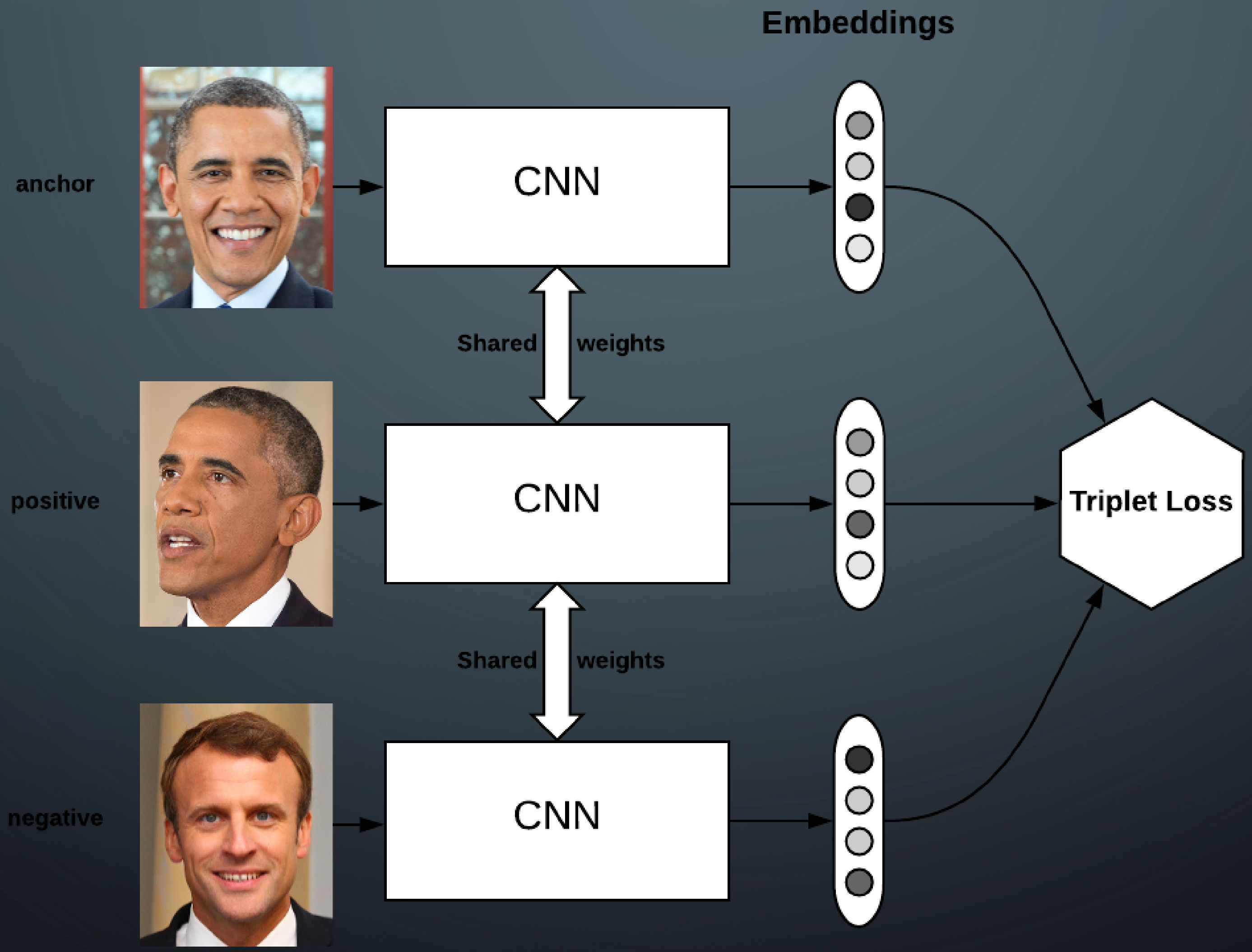
DISCRIMINATIVE VS SEPARABLE



CROSS ENTROPY 2D FEATURE SPACE



TRIPLET LOSS



TRIPLET LOSS

- $L(a, p, n) = \max(0, D(a, p) - D(a, n) + margin)$
- D is distance
- D could be L2 distance
- D could be (1− cosine similarity)

PROBLEMS WITH TRIPLET LOSS

- Most triplet has no information
- We can choose hard negative
- Noisy data and mislabeled data
- Computation is time consuming
- We can choose farthest positive and nearest negative from batch
- We need large batch-size (for example 1000)
- Memory bottleneck

CENTER LOSS

- We can use classification to train model
- Try to hold feature vector of each class near each other
- Force model to shape feature vectors as sphere
- This feature space will be discriminative

CENTER LOSS

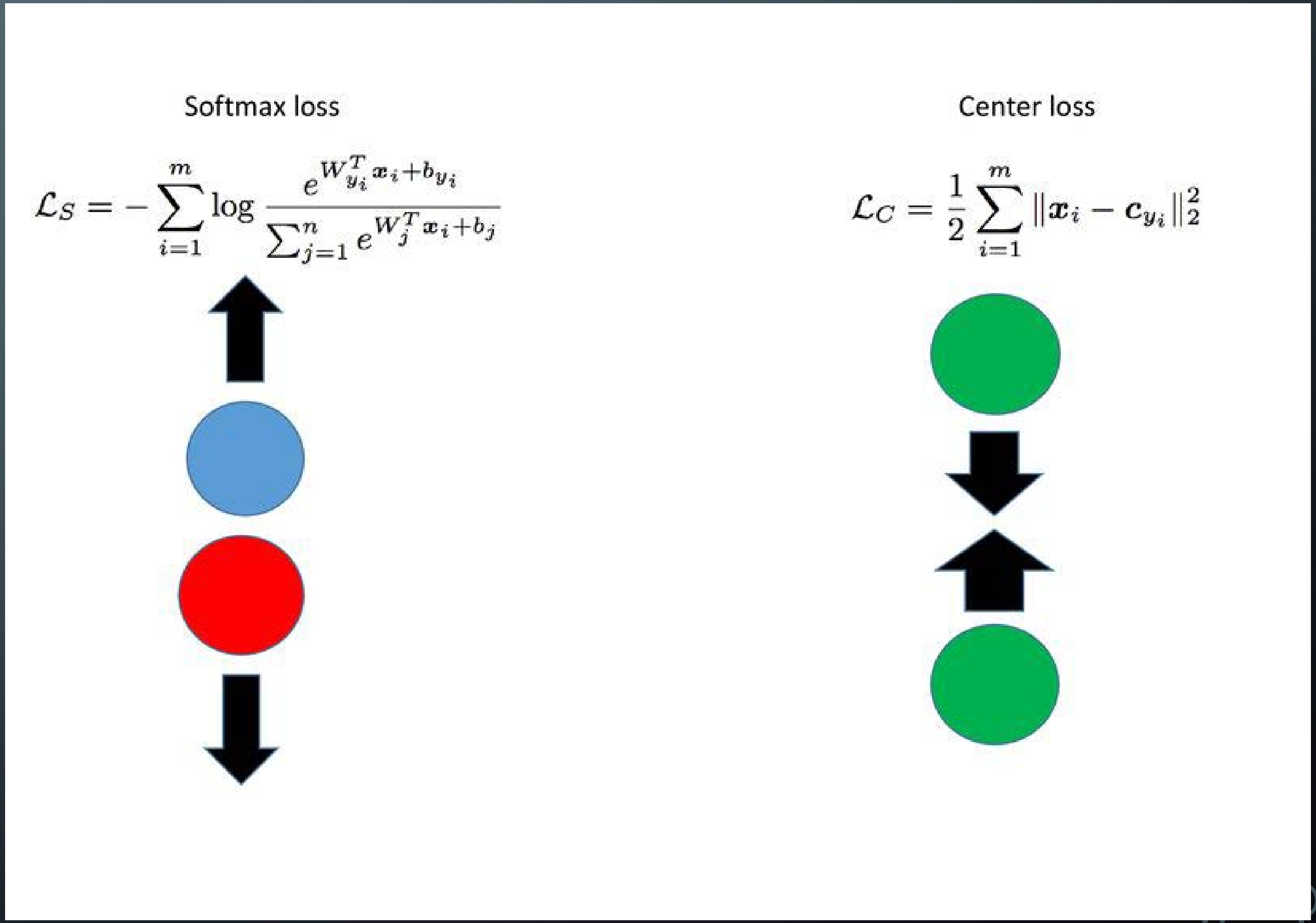
$$\mathcal{L}_s = -\sum_{i=1}^{m} \log \left(\frac{e^{W_{y_i}^T x_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T x_i + b_j}} \right)$$

$$\mathcal{L}_C = \frac{1}{2} \sum_{i=1}^{m} \|\boldsymbol{x}_i - \boldsymbol{c}_{y_i}\|_2^2$$

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_S + \lambda \mathcal{L}_C \\ &= -\sum_{i=1}^{m} \log \frac{e^{W_{y_i}^T \boldsymbol{x}_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T \boldsymbol{x}_i + b_j}} + \frac{\lambda}{2} \sum_{i=1}^{m} \|\boldsymbol{x}_i - \boldsymbol{c}_{y_i}\|_2^2 \end{aligned}$$

SOFTMAX LOSS VS CENTER LOSS

- Just Softmax loss
  - Separable
  - not discriminative
- Just Center loss
  - all feature vectors will be same
- Softmax loss =>
  between-class variation
- Center loss =>
  inter-class similarity



CENTER LOSS + CROSS ENTROPY FEATURE SPACE