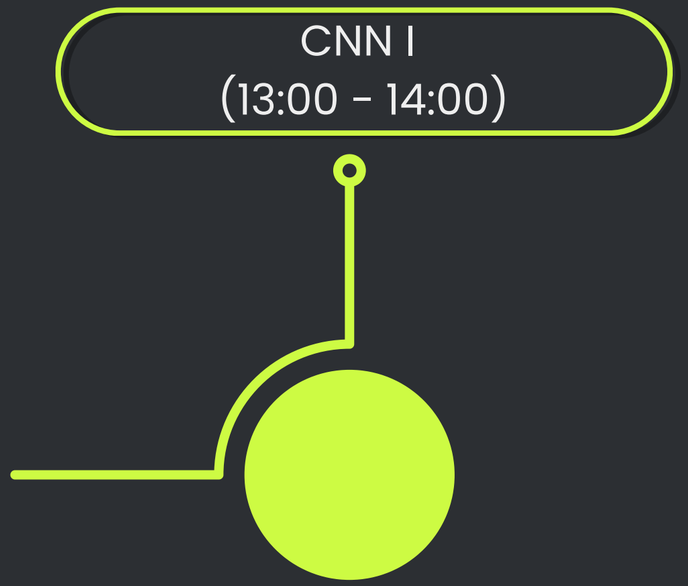
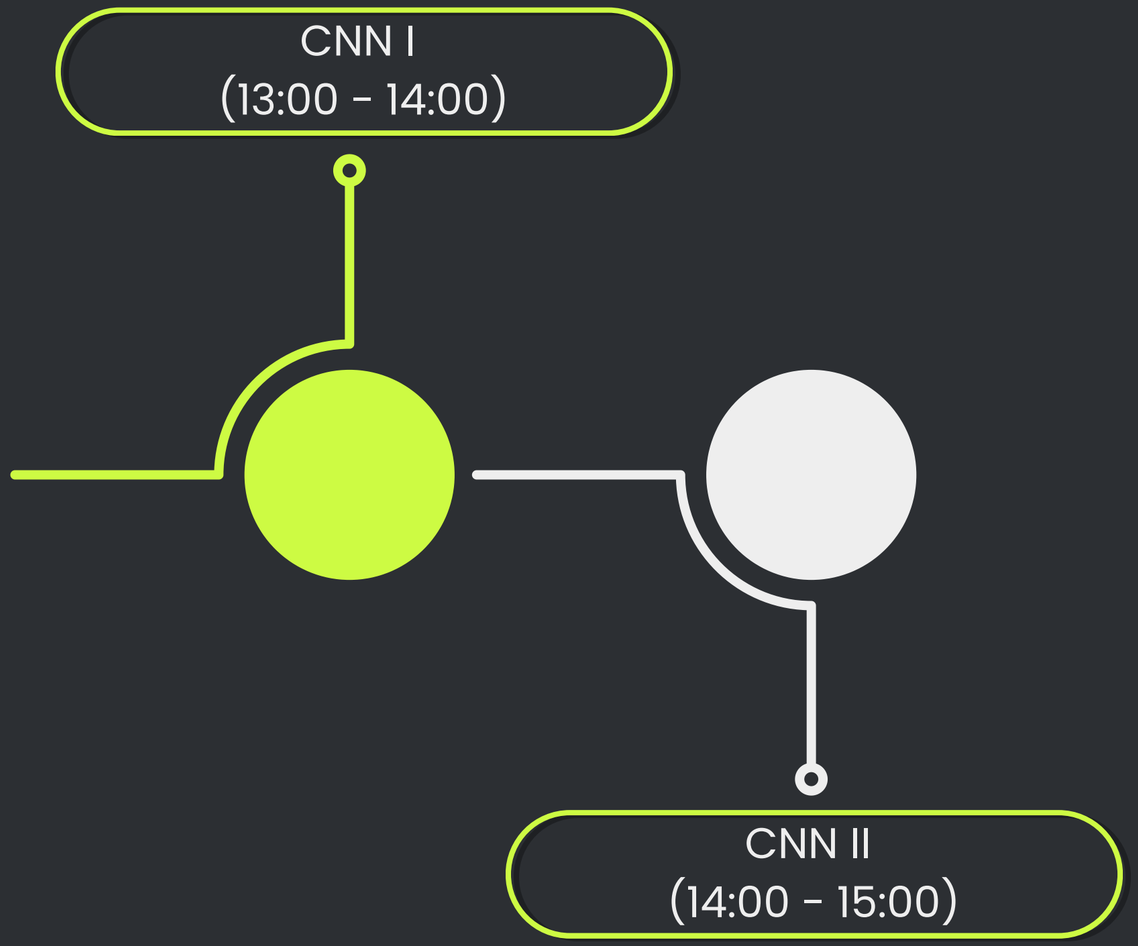
Lunch time

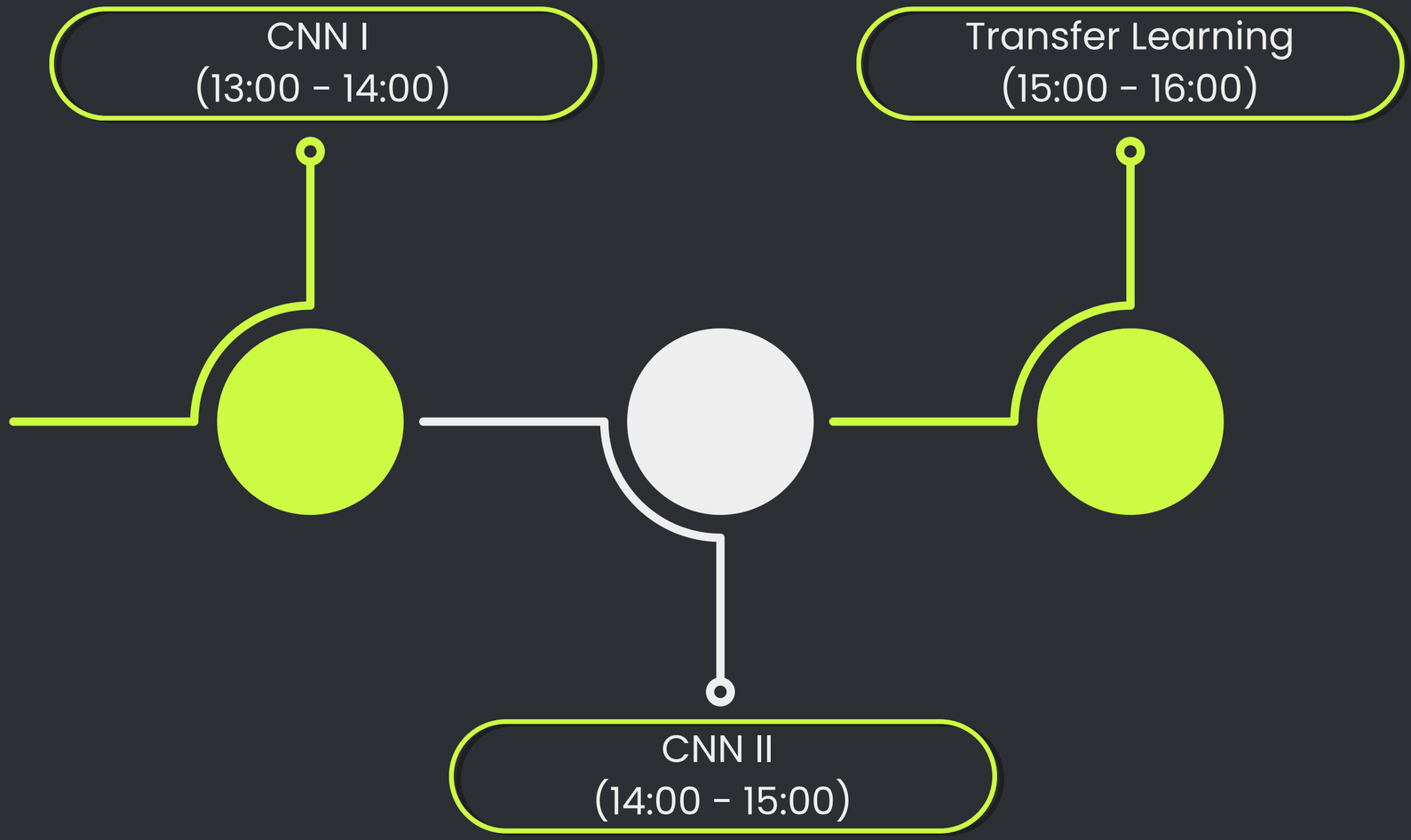
Morning Session (9:30 - 12:00)



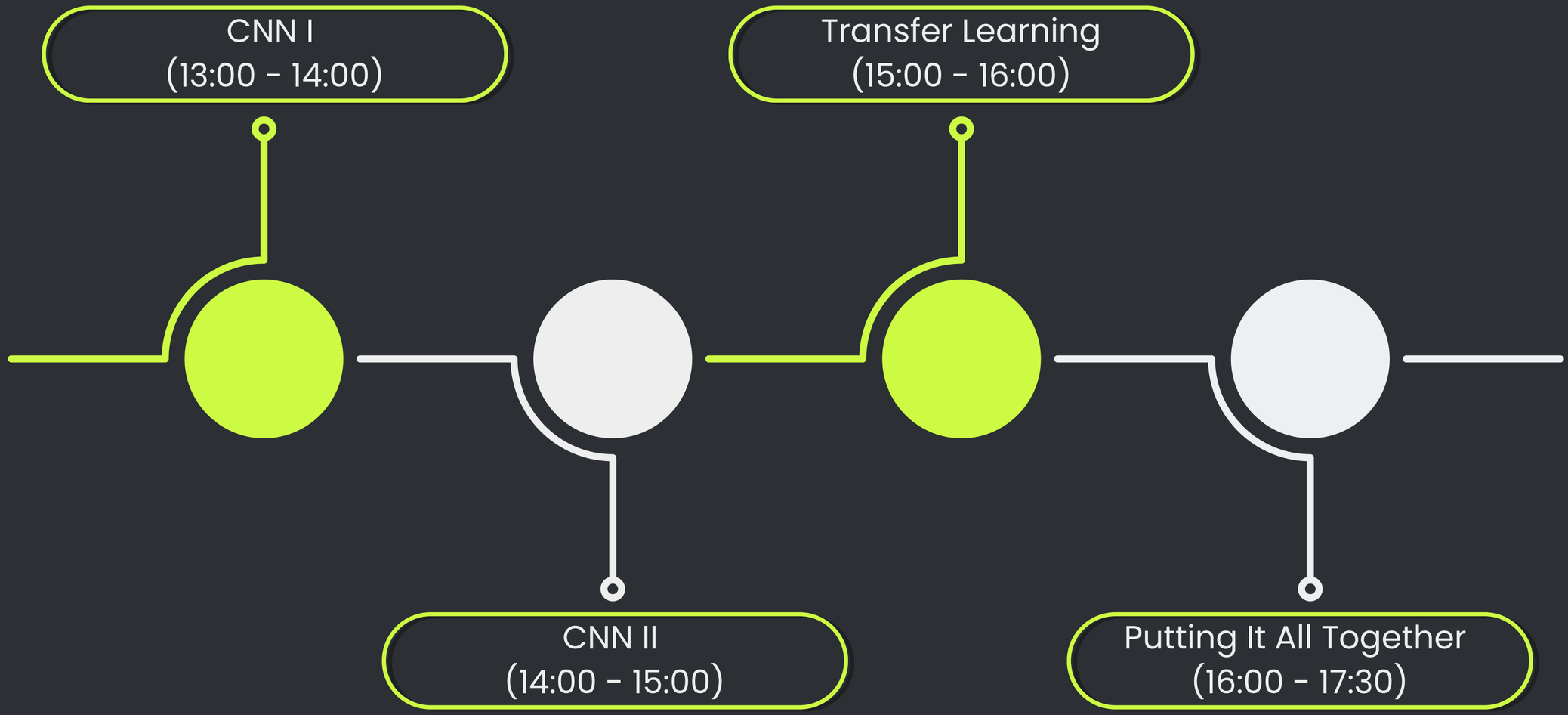
Morning Session (9:30 - 12:00)



Morning Session (9:30 - 12:00)



Afternoon Session (13:00 – 17:30)

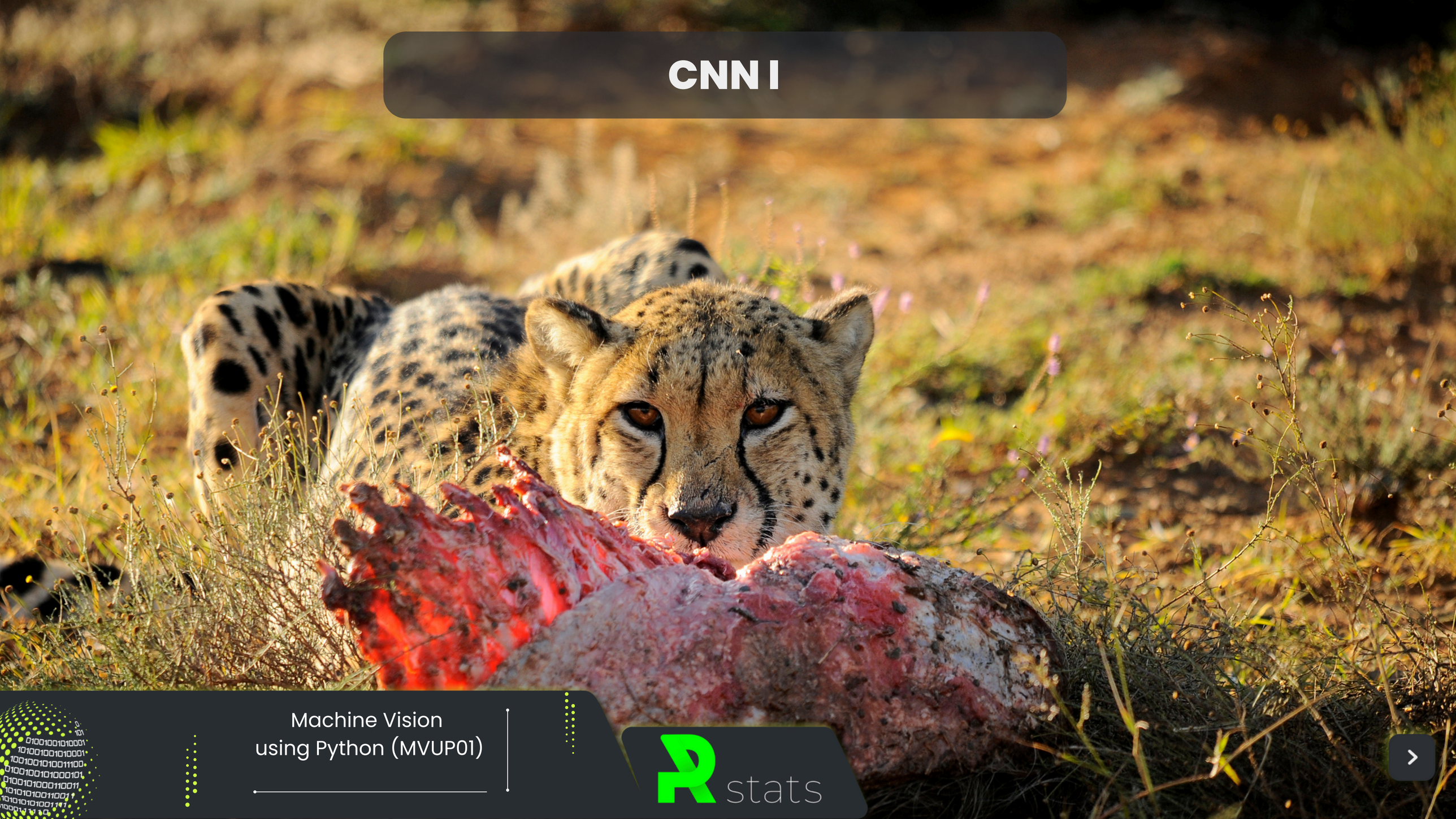


CNN I

Machine Vision
using Python (MVUP01)

Rstats

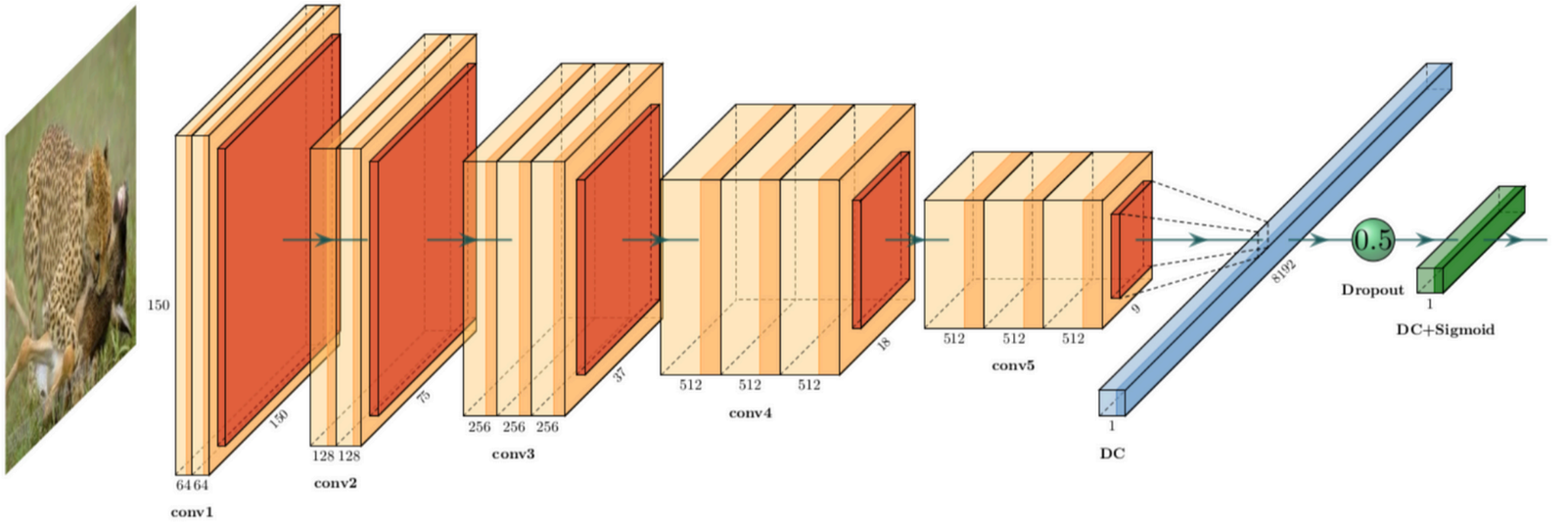
CNN I



Machine Vision
using Python (MVUP01)

stats

CNN I


150
64 64
conv1
150
128 128
conv2
75
256 256 256
conv3
37
512 512 512
conv4
18
512 512 512
conv5
9
1
DC
8192
0.5
Dropout
1
DC+Sigmoid

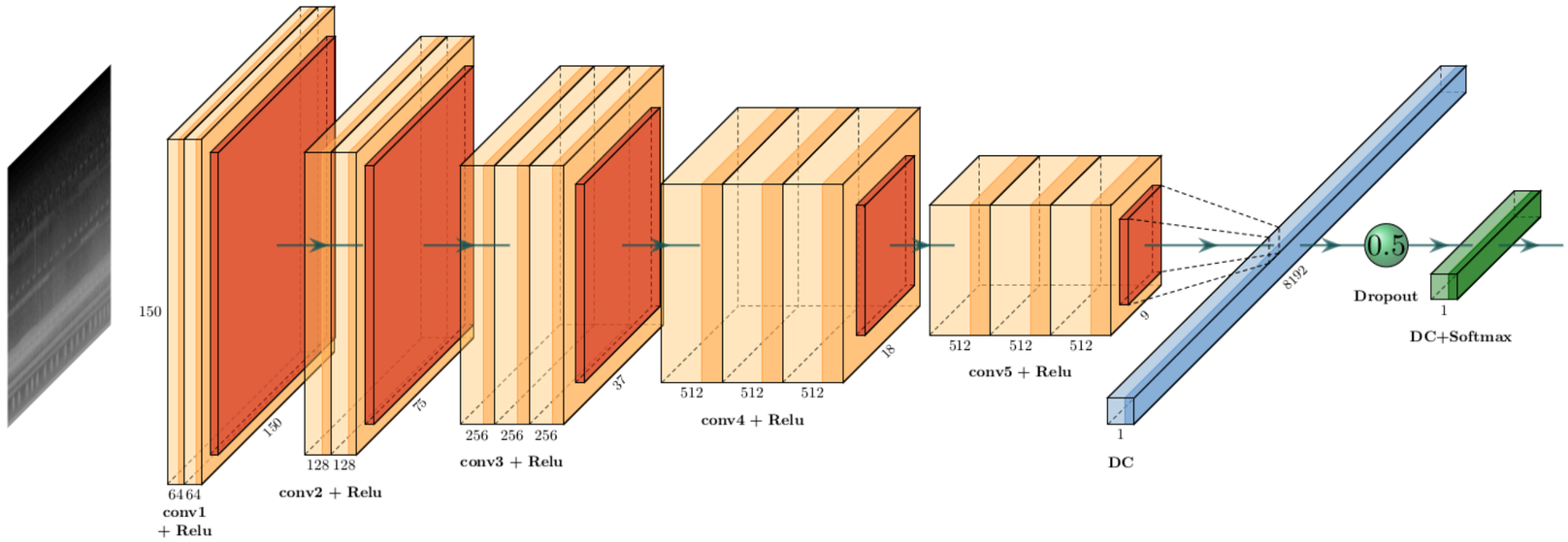
CNN I

CNN I

Machine Vision
using Python (MVUP01)

stats

CNN I

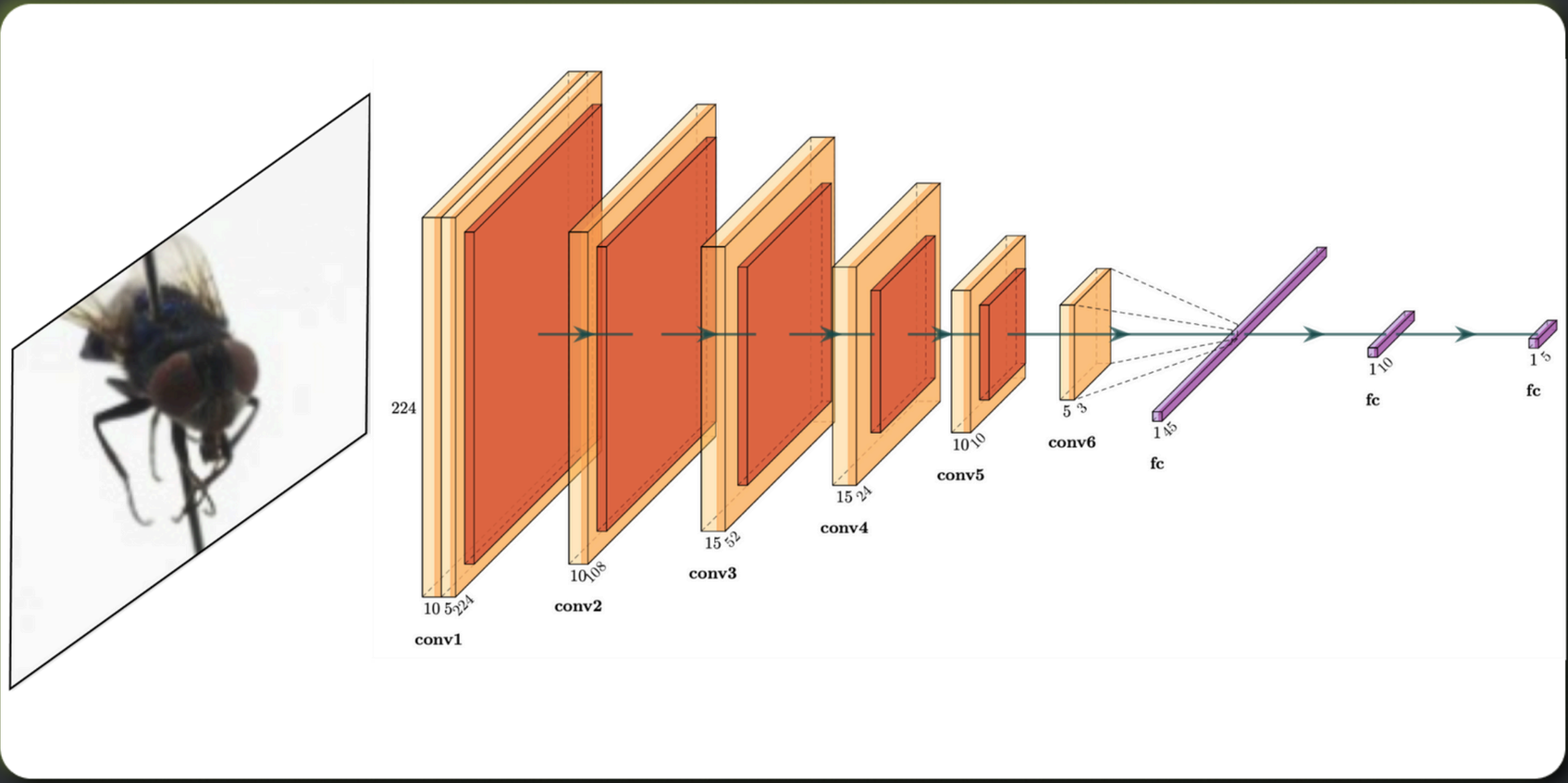

150
64 64
conv1
+ Relu
150
128 128
conv2 + Relu
75
256 256 256
conv3 + Relu
37
512 512 512
conv4 + Relu
18
512 512 512
conv5 + Relu
9
1
DC
8192
0.5
Dropout
1
DC+Softmax

CNN I

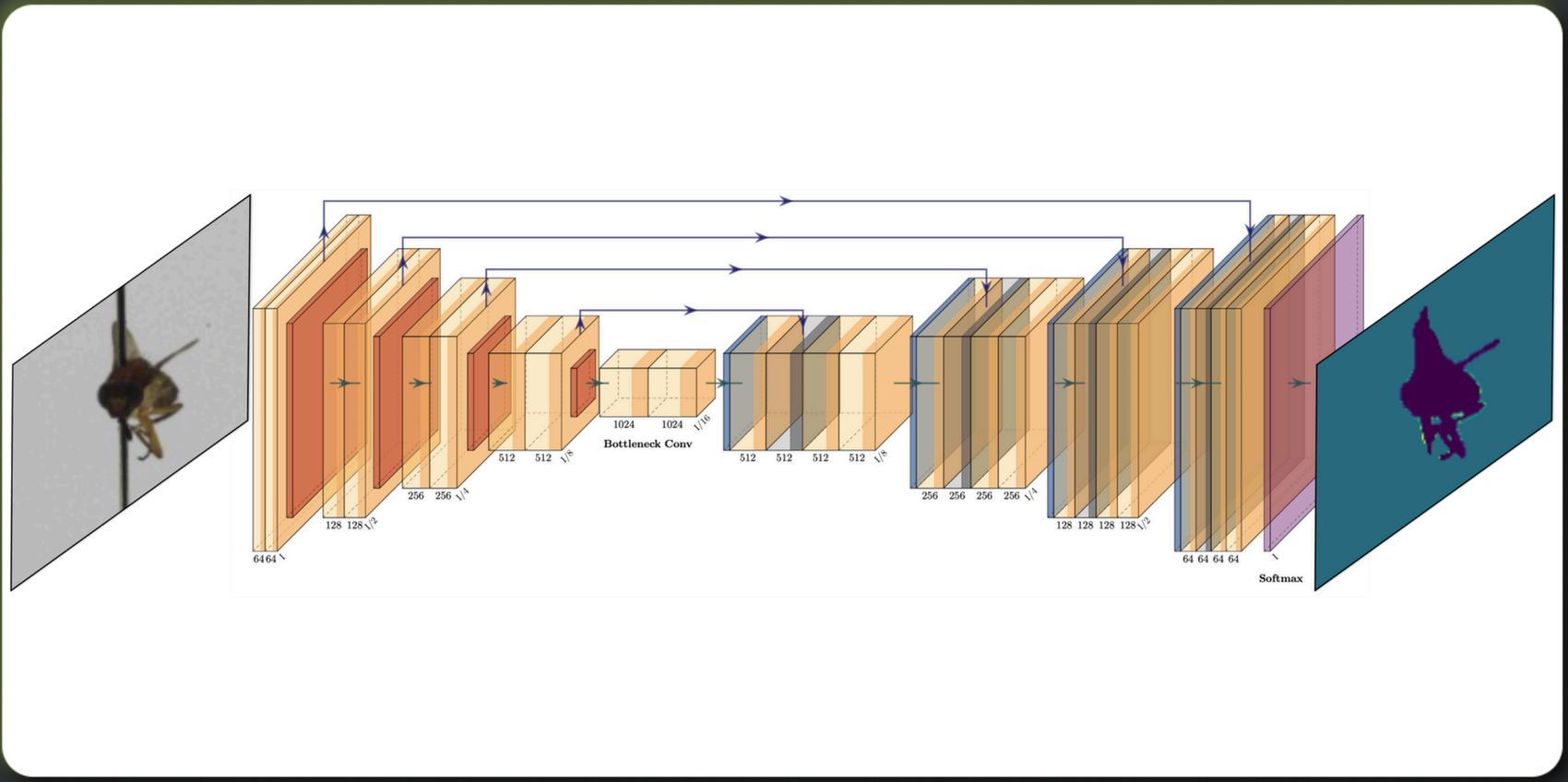
Machine Vision
using Python (MVUP01)

stats

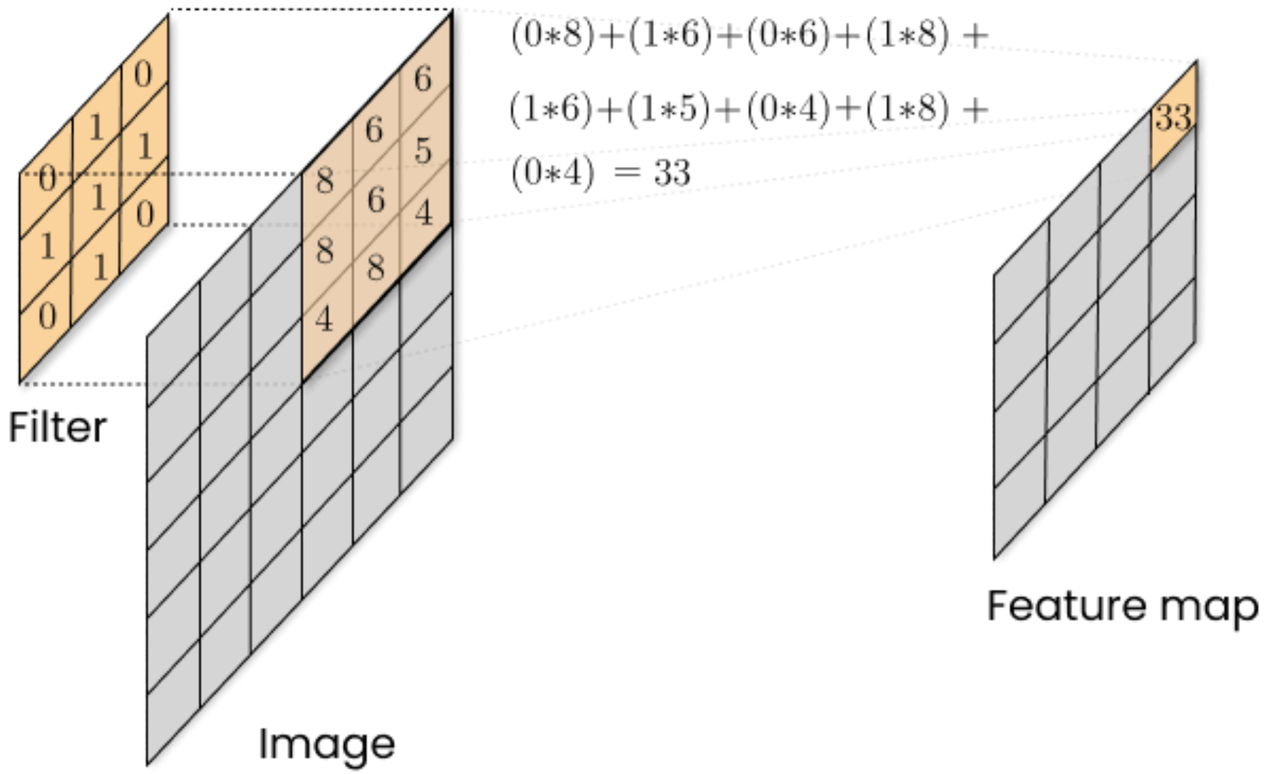
CNN I



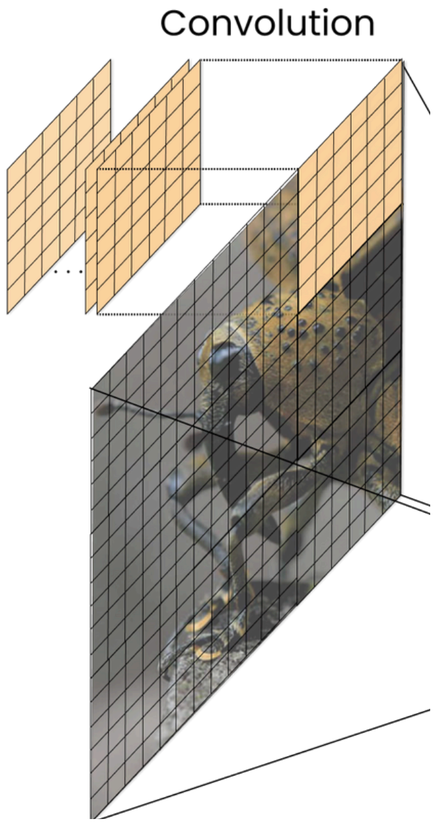
CNN I


64 64 1
128 128 1/2
256 256 1/4
512 512 1/8
1024 1024 1/16
Bottleneck Conv
512 512 512 512 1/8
256 256 256 256 1/4
128 128 128 128 1/2
64 64 64 64 1
Softmax

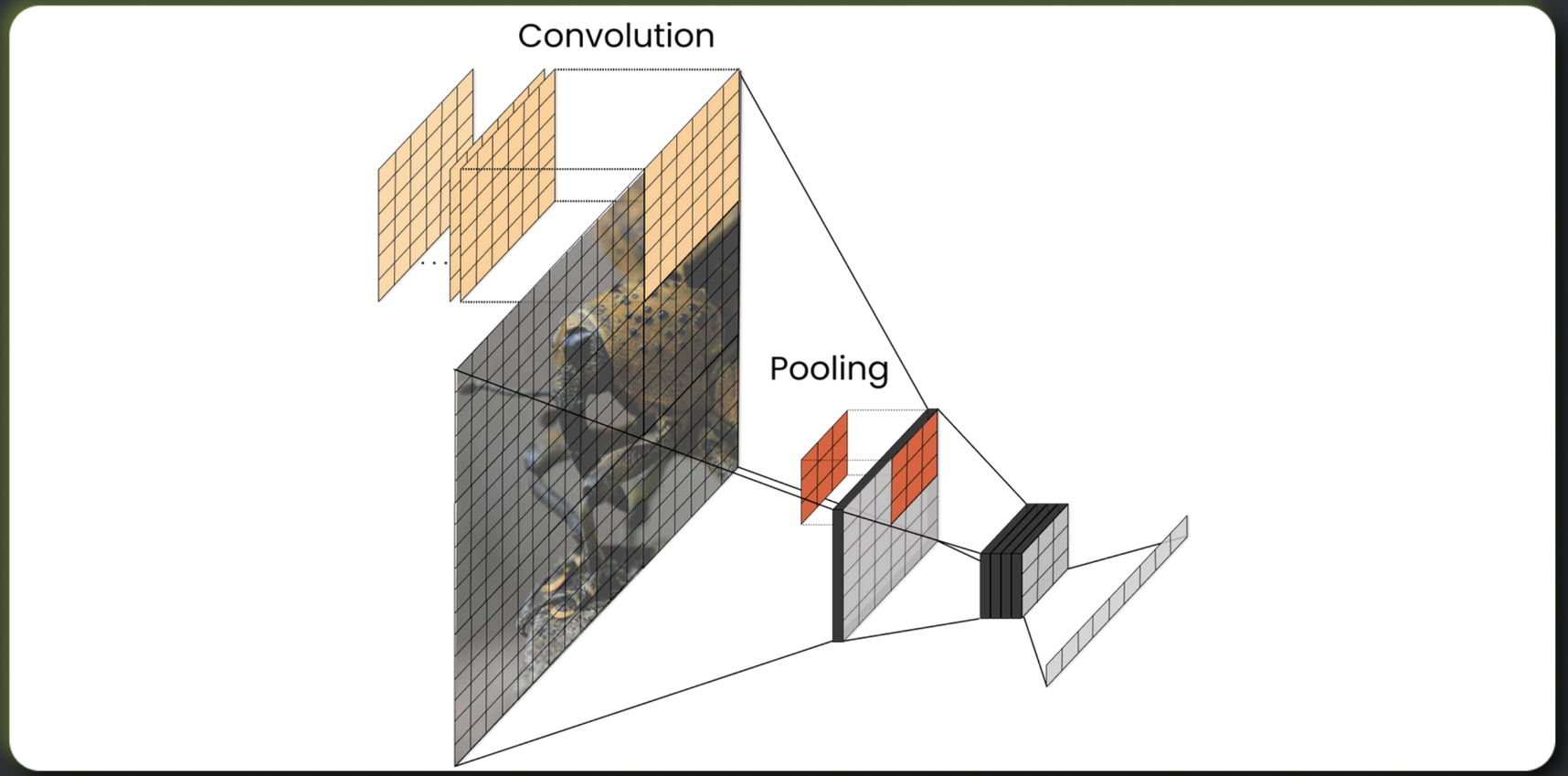
CNN I



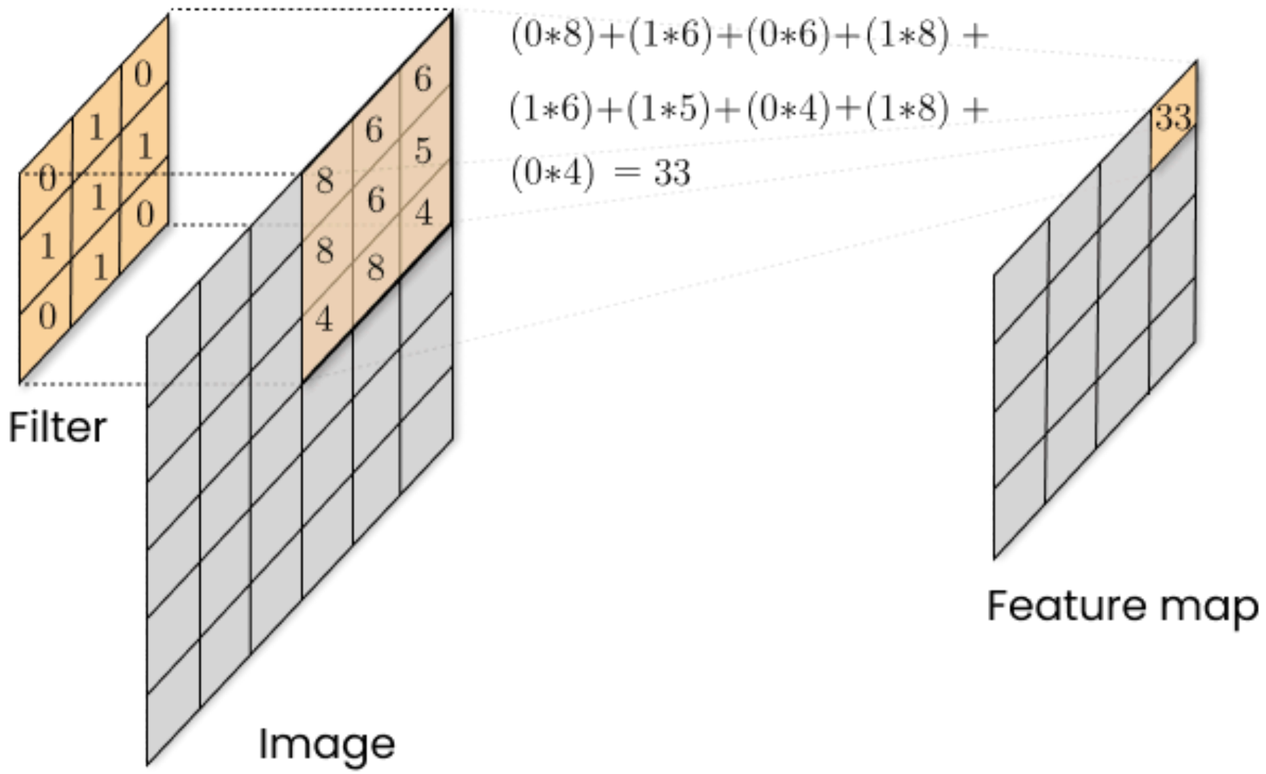
CNN I



CNN I



CNN I


0 1 0
1 1 1
0 1 0
1 1 0
0
Filter
6 6 5
8 6 4
8 8
4
Image
(0∗8)+(1∗6)+(0∗6)+(1∗8) +
(1∗6)+(1∗5)+(0∗4)+(1∗8) +
(0∗4) = 33
33
Feature map

CNN I

ConvNetJS CIFAR-10 demo

Description

This demo trains a Convolutional Neural Network on the CIFAR-10 dataset in your browser, with nothing but Javascript. The state of the art on this dataset is about 90% accuracy and human performance is at about 94% (not perfect as the dataset can be a bit ambiguous). I used this python script to parse the original files (python version) into batches of images that can be easily loaded into page DOM with img tags.

This dataset is more difficult and it takes longer to train a network. Data augmentation includes random flipping and random image shifts by up to 2px horizontally and verically.

By default, in this demo we're using Adadelta which is one of per-parameter adaptive step size methods, so we don't have to worry about changing learning rates or momentum over time. However, I still included the text fields for changing these if you'd like to play around with SGD+Momentum trainer.

Report questions/bugs/suggestions to @karpathy.

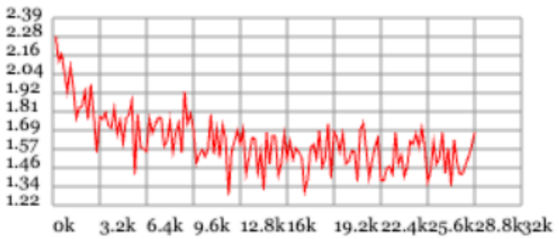
Training Stats

pause

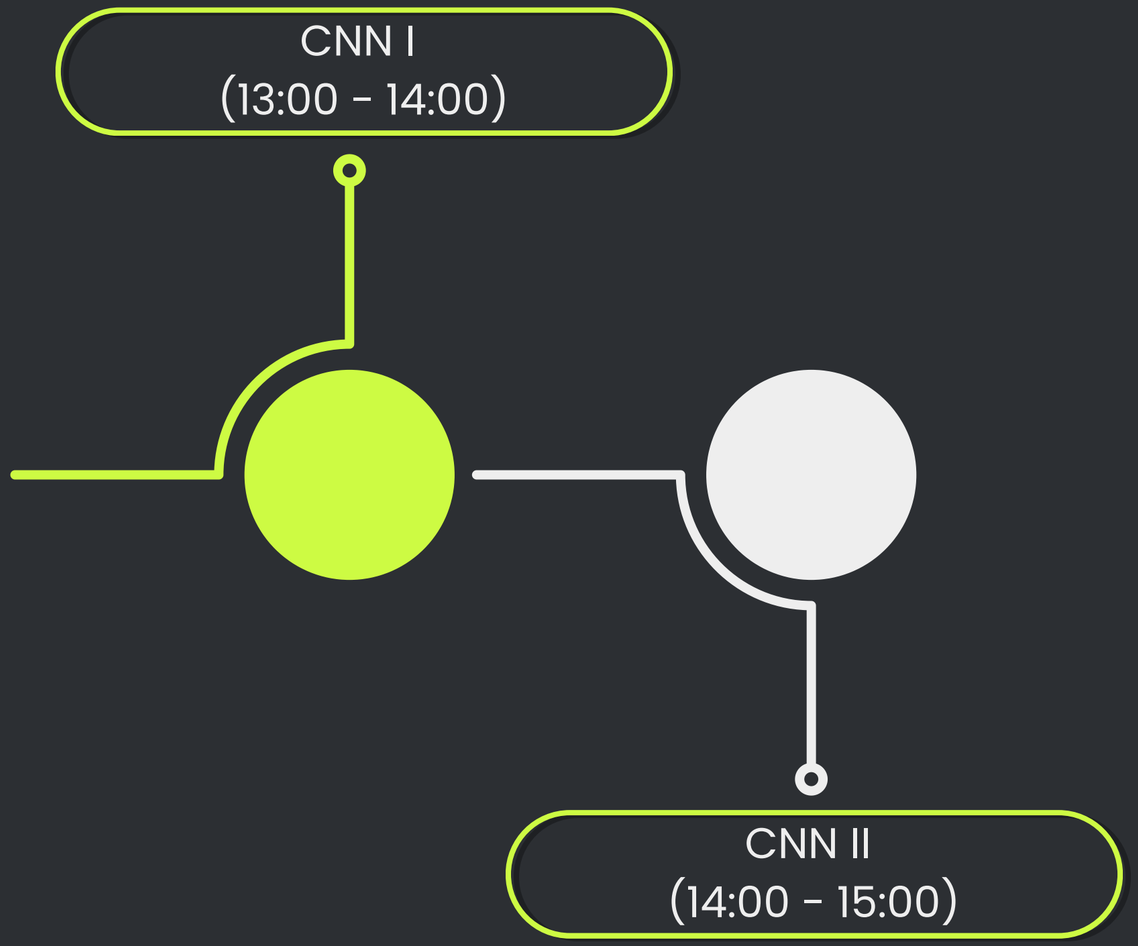
Forward time per example: 5ms
Backprop time per example: 9ms
Classification loss: 1.57914
L2 Weight decay loss: 0.01882
Training accuracy: 0.45
Validation accuracy: 0.51
Examples seen: 28957

Learning rate: 0.01 change
Momentum: 0.9 change
Batch size: 4 change

Loss:



Morning Session (9:30 - 12:00)



CNN II

Machine Vision
using Python (MVUP01)

stats

CNN II

Machine Vision
using Python (MVUP01)

stats

CNN I

The CNN architecture contains more parameters to estimate than a DNN, mainly because of the convolution layers. This layer uses an operation called convolution that is defined by the equation below:

CNN I

The CNN architecture contains more parameters to estimate than a DNN, mainly because of the convolution layers. This layer uses an operation called convolution that is defined by the equation below:

$$C_{r,q}^{d} = f\left(\sum_{m=1}^{M}\sum_{n=1}^{N} w_{m,n}^{c} x_{r+m-1,q+n-1} + \theta^{c}\right),$$

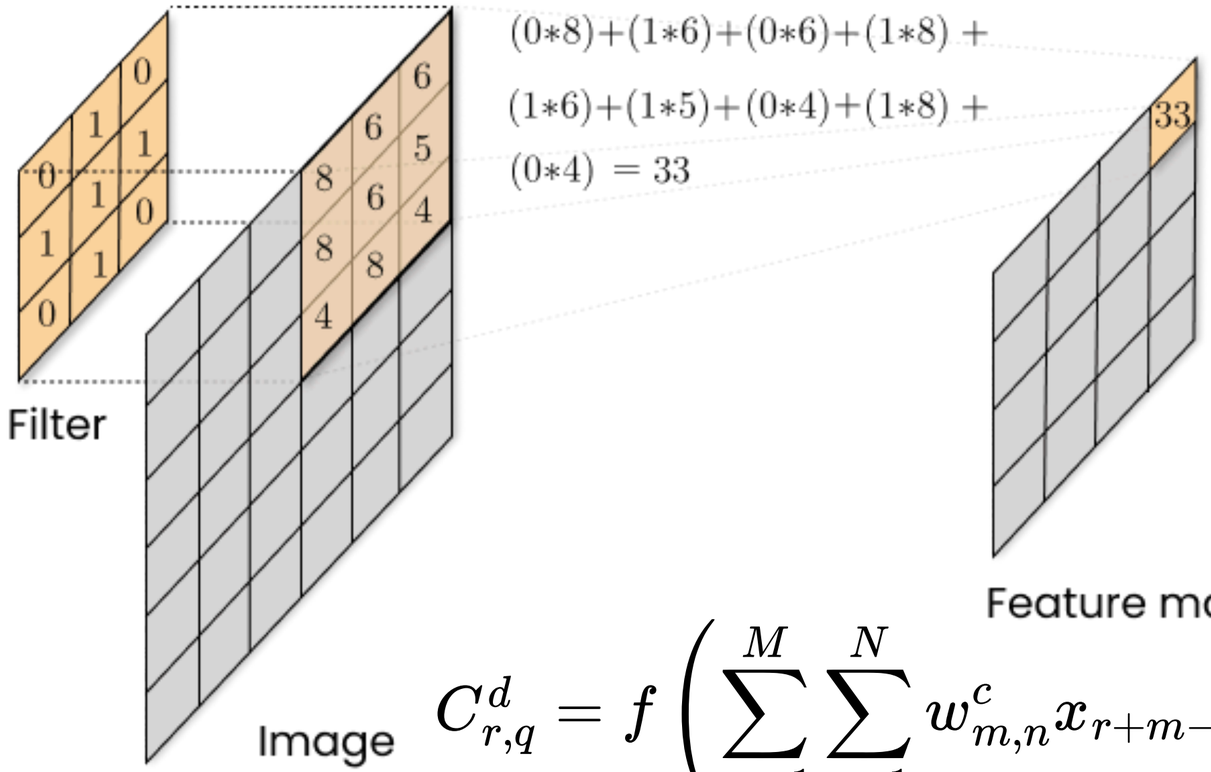
CNN I

The CNN architecture contains more parameters to estimate than a DNN, mainly because of the convolution layers. This layer uses an operation called convolution that is defined by the equation below:

$$C_{r,q}^{d} = f\left(\sum_{m=1}^{M}\sum_{n=1}^{N} w_{m,n}^{c} x_{r+m-1,q+n-1} + \theta^{c}\right),$$

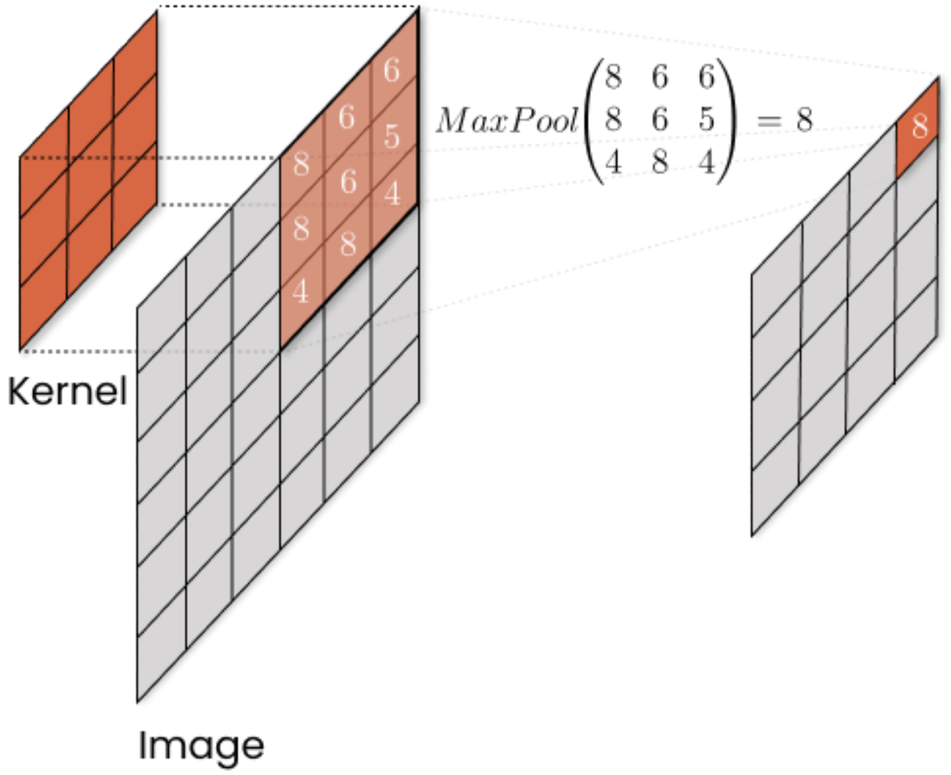
where $x_{r,q}$ are pixels of a greyscale image $\mathbf{X}$ of size $R \times Q$, $w_{m,n}^{c}$ and θ^{c} are hyperparameters of a filter $\mathbf{W}$ with size $M \times N$, f can be any activation function, and (r, q) are the indices of the output, also known as a *feature map*. This operation can be repeated d times per convolution layer.

CNN I

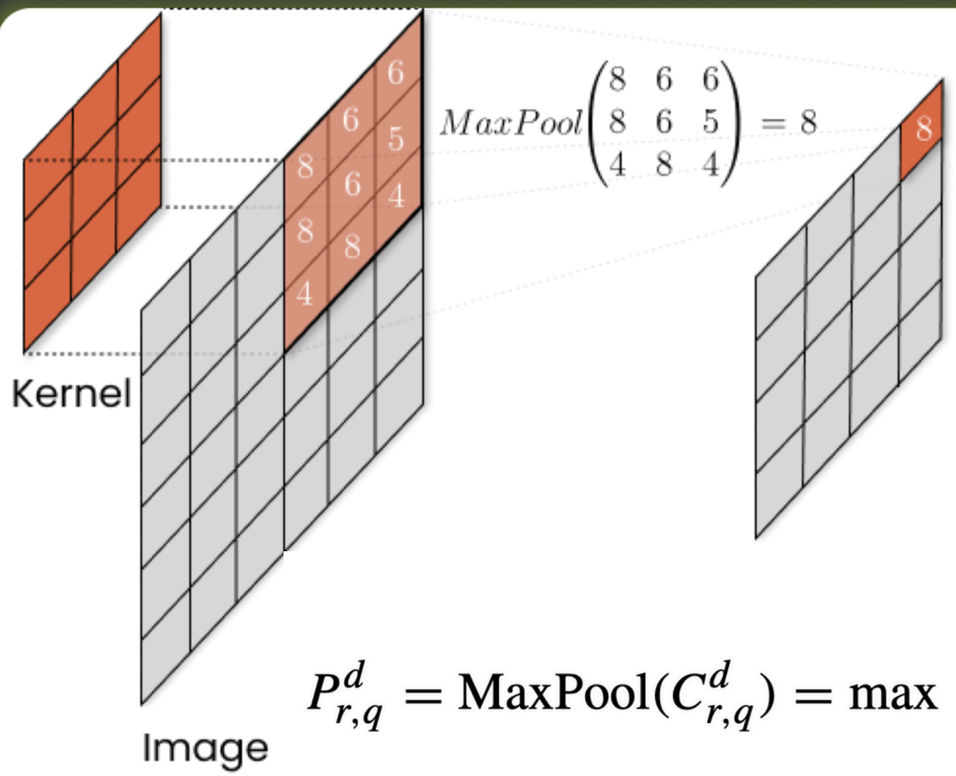


$$C^d_{r,q} = f\left(\sum_{m=1}^{M}\sum_{n=1}^{N} w^c_{m,n} x_{r+m-1,q+n-1} + \theta^c\right)$$

CNN I


$MaxPool\begin{pmatrix} 8 & 6 & 6 \\ 8 & 6 & 5 \\ 4 & 8 & 4 \end{pmatrix} = 8$
Kernel
Image

CNN I



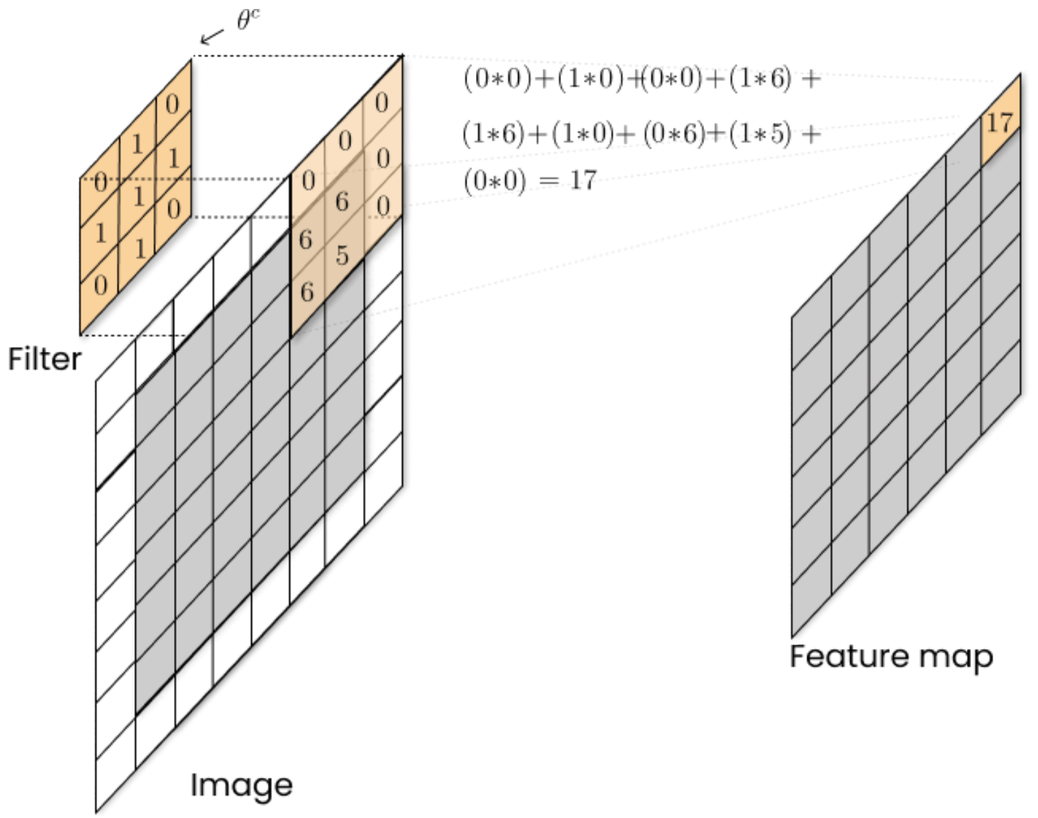
$$MaxPool\begin{pmatrix} 8 & 6 & 6 \\ 8 & 6 & 5 \\ 4 & 8 & 4 \end{pmatrix} = 8$$

$$P^d_{r,q} = \text{MaxPool}(C^d_{r,q}) = \max \begin{pmatrix} C^d_{r,q} & C^d_{r+1,q} & C^d_{r+2,q} \\ C^d_{r+1,q} & C^d_{r+1,q+1} & C^d_{r+2,q} \\ C^d_{r+2,q} & C^d_{r+2,q+1} & C^d_{r+2,q+2}, \end{pmatrix}$$

CNN I

Typically since small filters are used in convolution operations, a few pixels may be lost. A solution for this problem is the padding approach, which consists of adding extra pixels around the boundary of the input image, increasing the effective size of the image [70]. These extra values are typically zero.

CNN I



CNN I

ConvNetJS CIFAR-10 demo

Description

This demo trains a Convolutional Neural Network on the CIFAR-10 dataset in your browser, with nothing but Javascript. The state of the art on this dataset is about 90% accuracy and human performance is at about 94% (not perfect as the dataset can be a bit ambiguous). I used this python script to parse the original files (python version) into batches of images that can be easily loaded into page DOM with img tags.

This dataset is more difficult and it takes longer to train a network. Data augmentation includes random flipping and random image shifts by up to 2px horizontally and verically.

By default, in this demo we're using Adadelta which is one of per-parameter adaptive step size methods, so we don't have to worry about changing learning rates or momentum over time. However, I still included the text fields for changing these if you'd like to play around with SGD+Momentum trainer.

Report questions/bugs/suggestions to @karpathy.

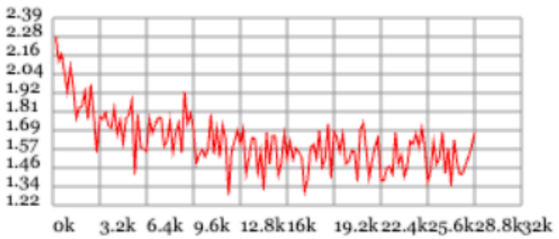
Training Stats

pause

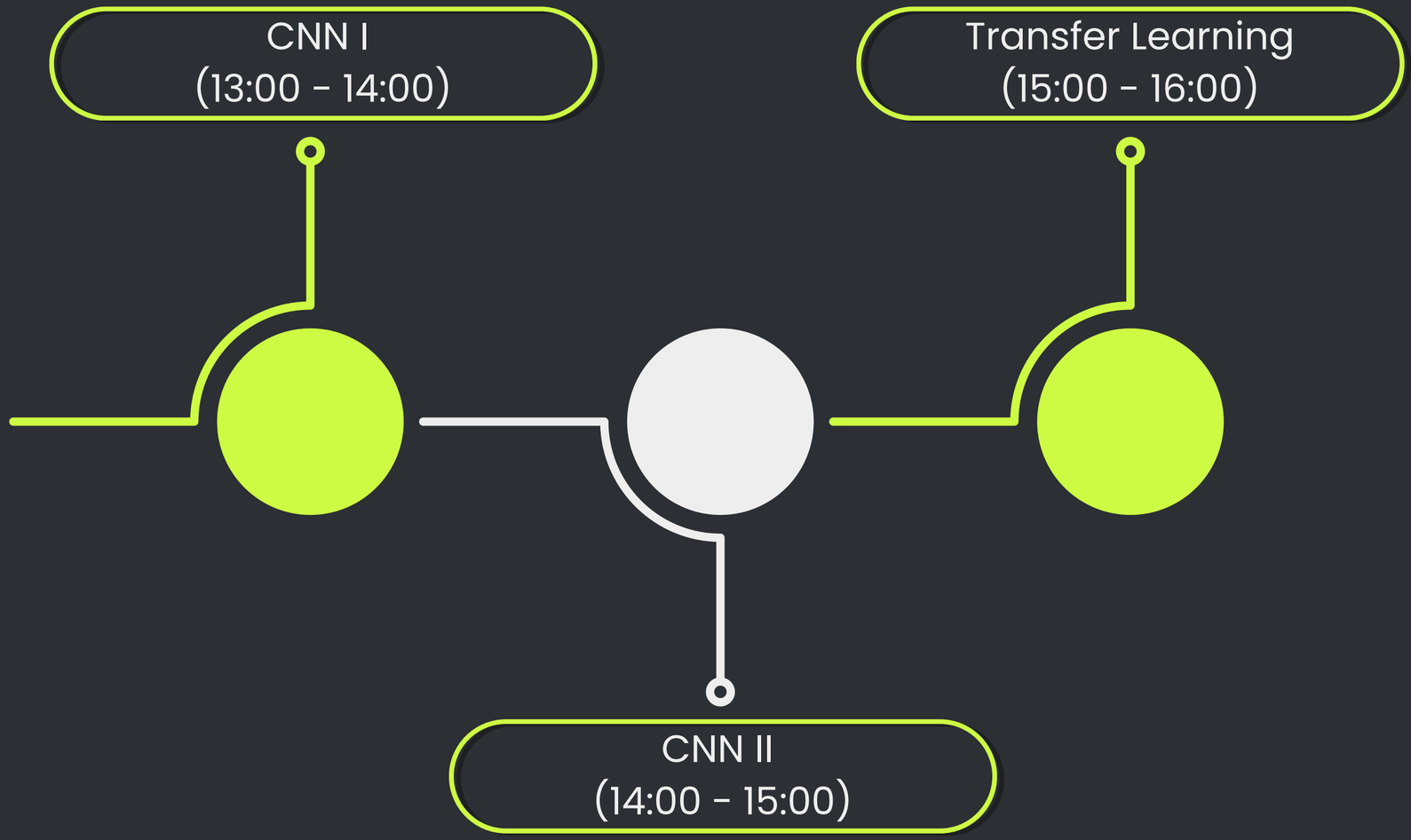
Forward time per example: 5ms
Backprop time per example: 9ms
Classification loss: 1.57914
L2 Weight decay loss: 0.01882
Training accuracy: 0.45
Validation accuracy: 0.51
Examples seen: 28957

Learning rate: 0.01 change
Momentum: 0.9 change
Batch size: 4 change

Loss:



Morning Session (9:30 - 12:00)

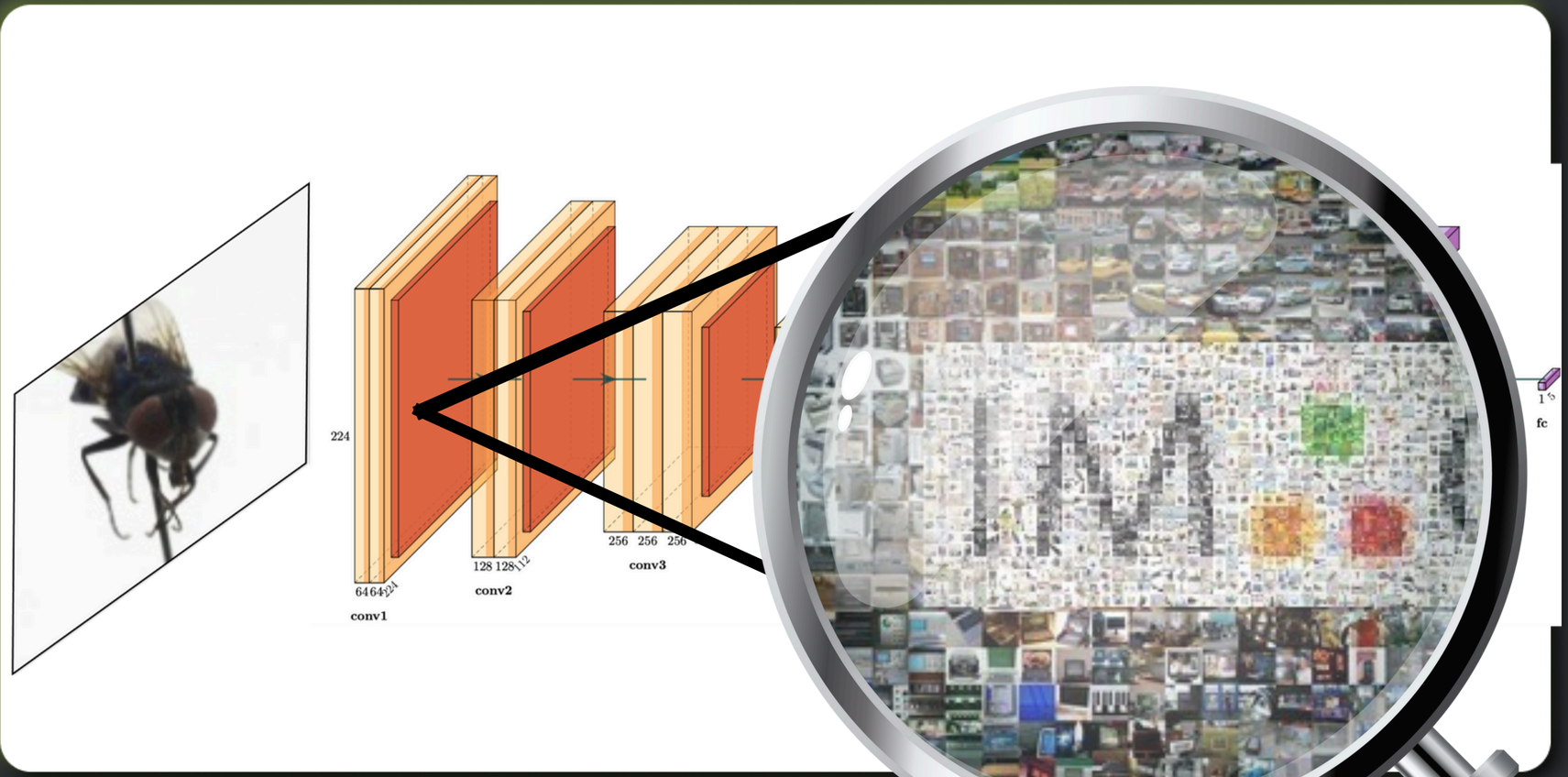


Transfer Learning

Machine Vision
using Python (MVUP01)

stats

CNN II



CNN II

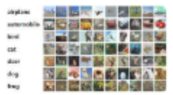
Search for datasets

Best match

Filter by Modality (clear)

- Images ×
- Texts 2931
- Videos 959
- Audio 457
- Medical 376
- 3D 358

3058 dataset results for Images ×



CIFAR-10 (Canadian Institute for Advanced Research, 10 classes)

The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. The dataset is di-...

15,538 PAPERS • 108 BENCHMARKS

ImageNet

The ImageNet dataset contains 14,197,122 annotated images according to the WordNet hierarchy. Since 2010 the dataset is used in the ImageNet Large Scale Visual Recognition...

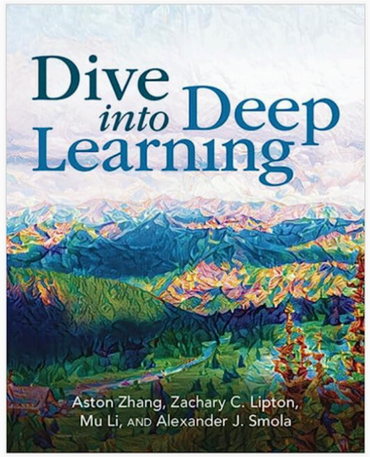
14,860 PAPERS • 52 BENCHMARKS

MS COCO (Microsoft Common Objects in Context)

The MS COCO (Microsoft Common Objects in Context) dataset is a large-scale object detection, segmentation, key-point detection, and captioning dataset. The dataset consists of...

11,420 PAPERS • 96 BENCHMARKS

CNN I



Dive into Deep Learning

Interactive deep learning book with code, math, and discussions

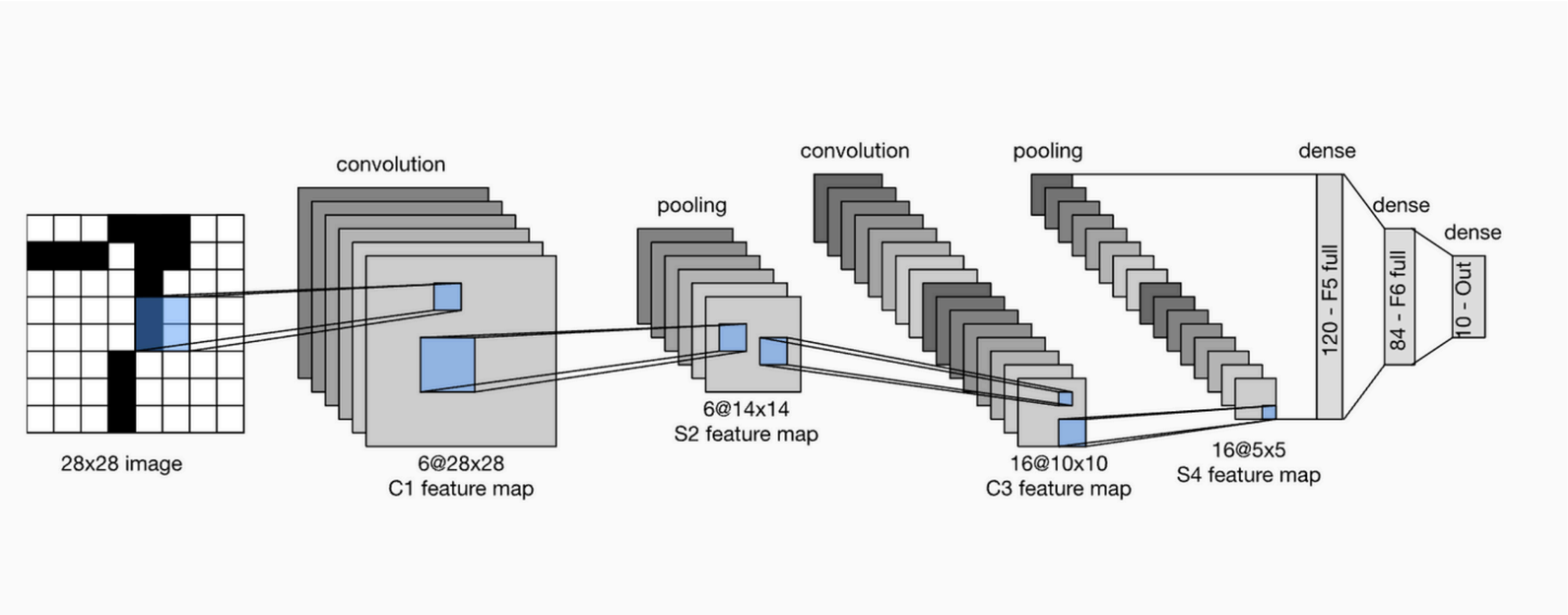
Implemented with PyTorch, NumPy/MXNet, JAX, and TensorFlow

Adopted at 500 universities from 70 countries

☆ Star 24,823

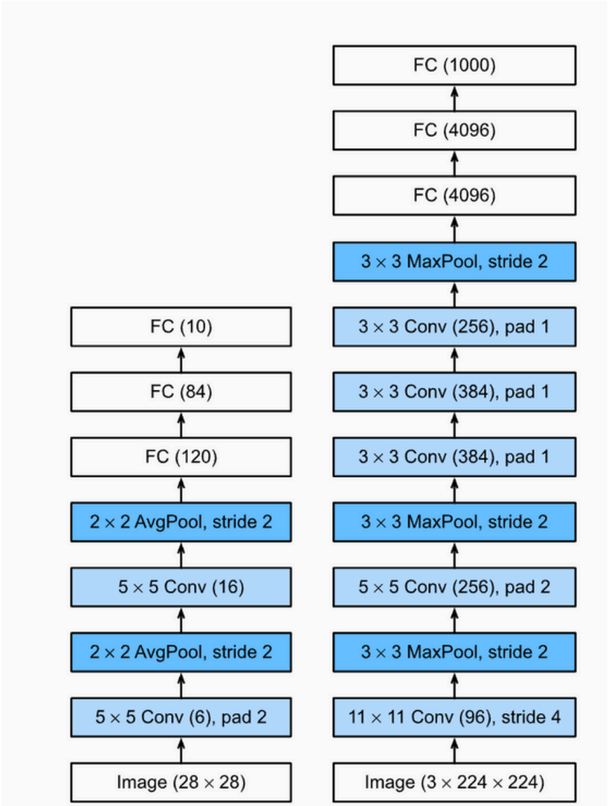
CNN I

LeNet


convolution
pooling
convolution
pooling
dense
dense
dense
120 - F5 full
84 - F6 full
10 - Out
28x28 image
6@28x28
C1 feature map
6@14x14
S2 feature map
16@10x10
C3 feature map
16@5x5
S4 feature map

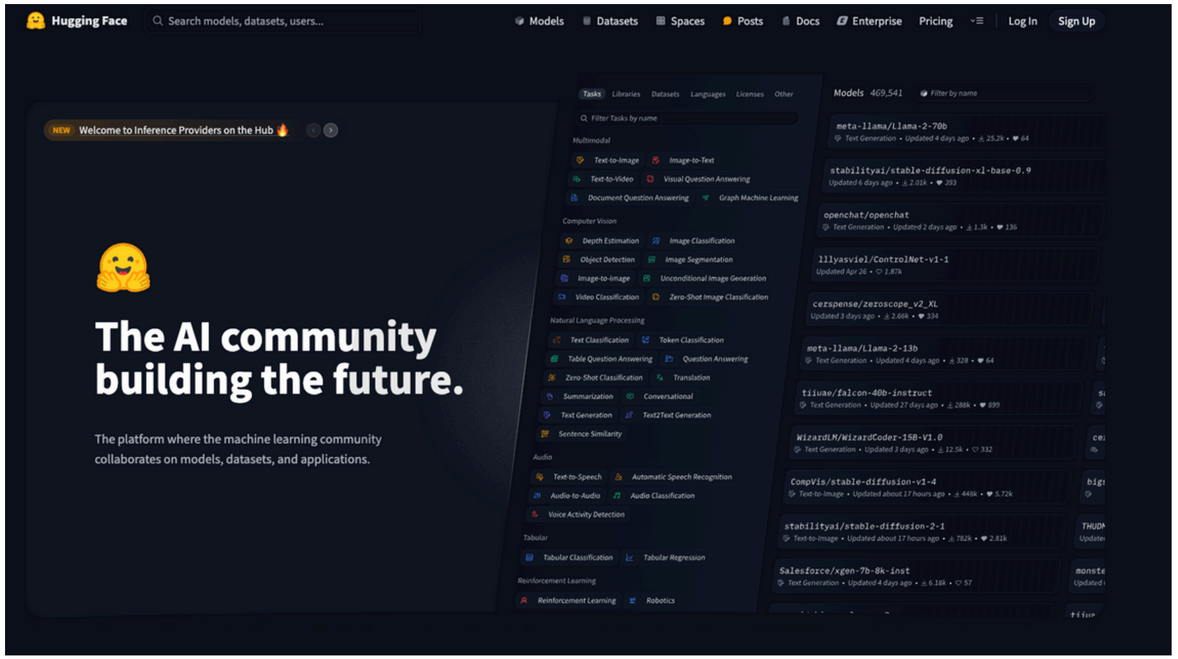
CNN I

LeNet VS AlexNet

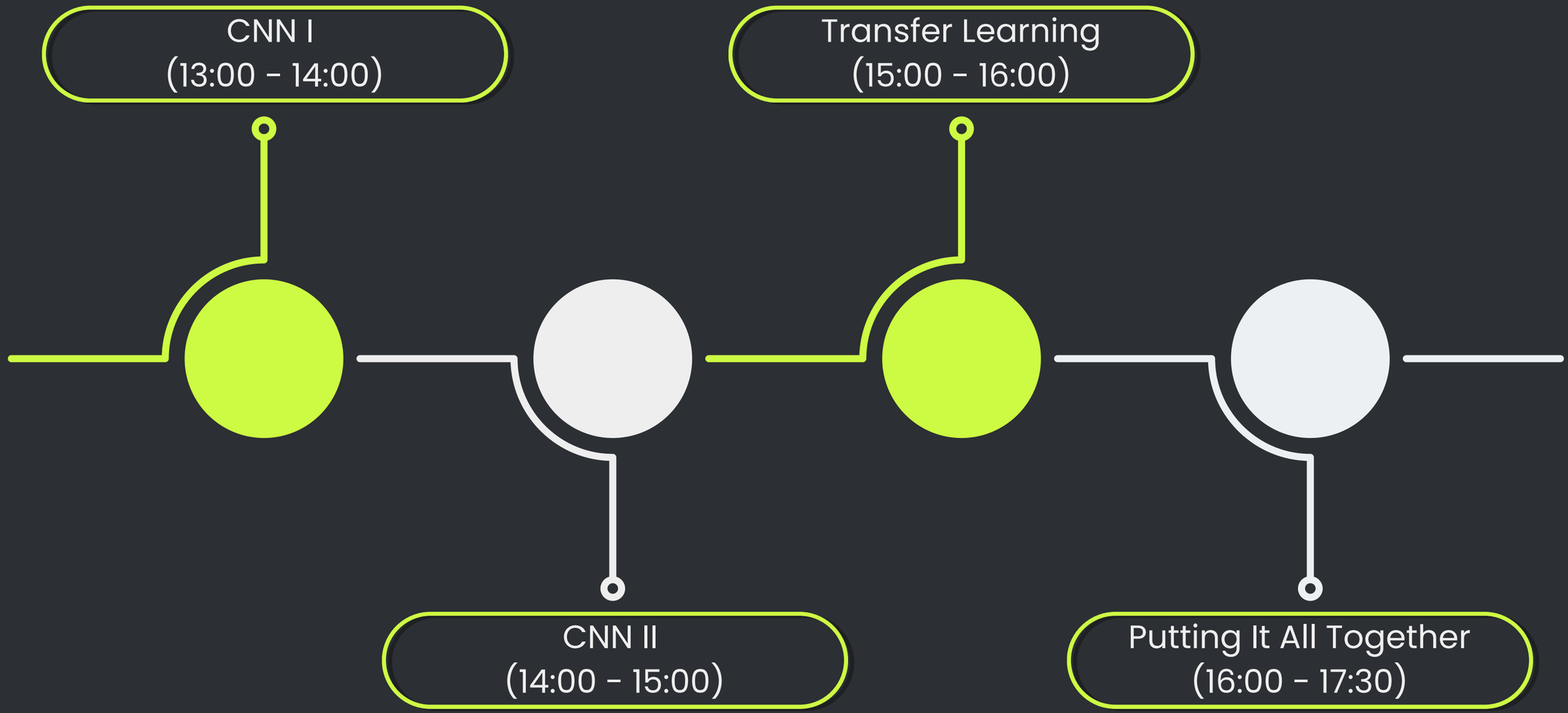


CNN I

huggingface.co



Afternoon Session (13:00 – 17:30)



Putting It All Together

Machine Vision
using Python (MVUP01)

stats