Introduction to Convolutional Neural Net

Concepts, Architecture and Applications

iDLVision Tech

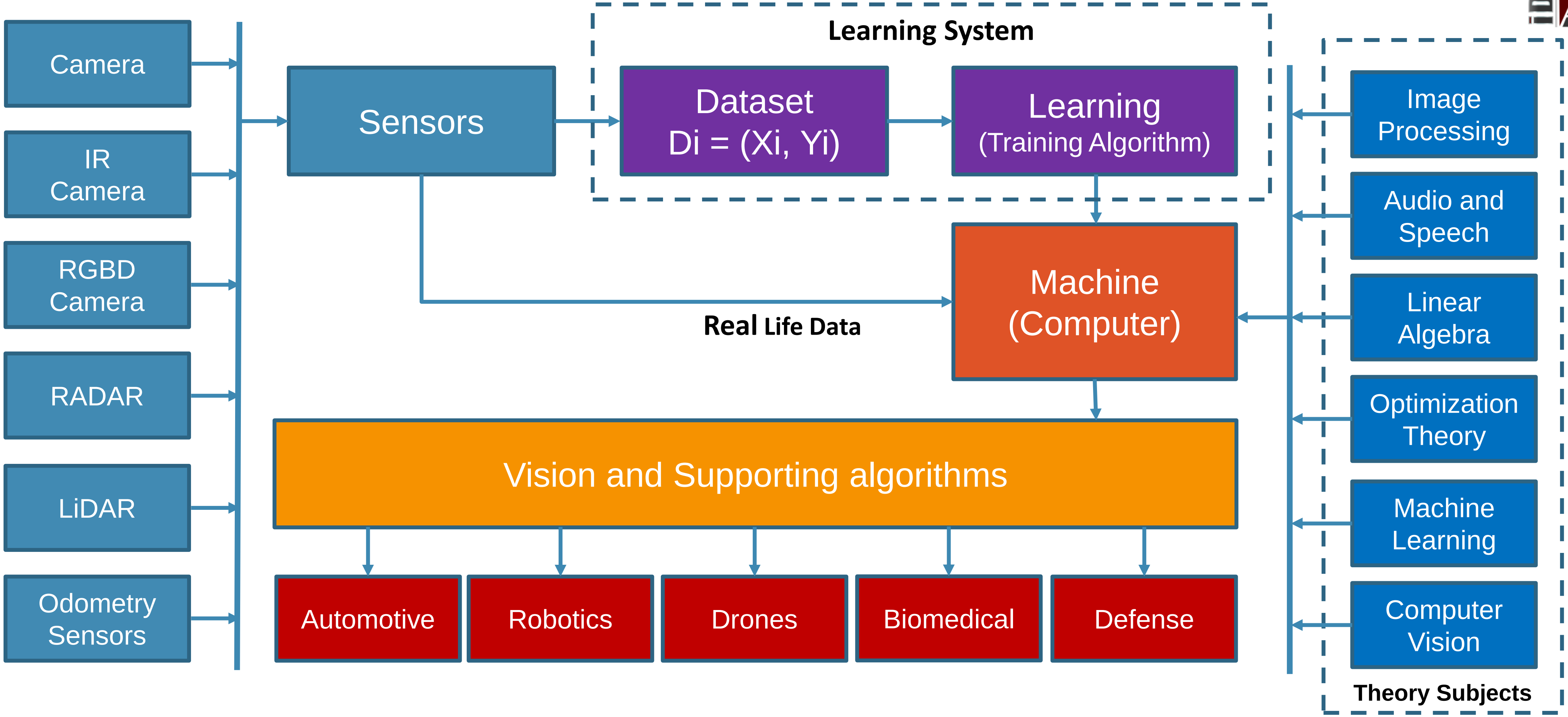
www.idlvision.com

Dr. Ganesh Bhokare

Objective: AI - ML - DL - CNN

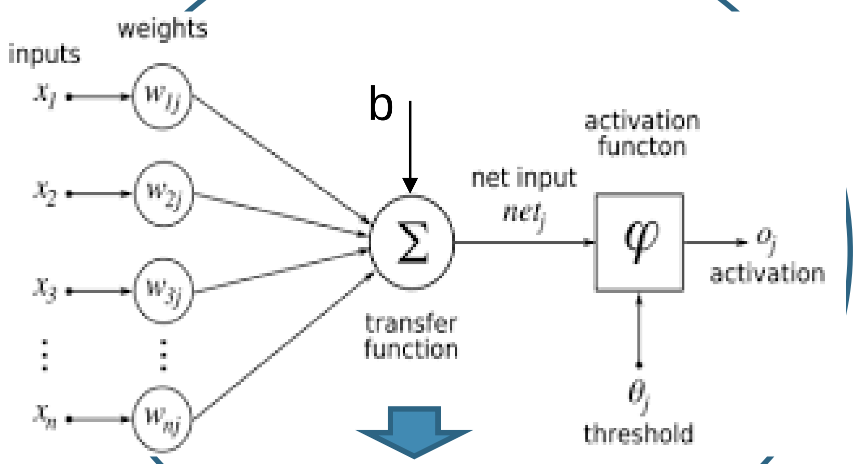
- Understand concepts: **AI - CV - ML - DL - CNN**
- Approach / Perception to learn - Methodology
- Technology – HW, SW requirements
- Applications / Use cases
- Future and Scope

AI – Machine Learning



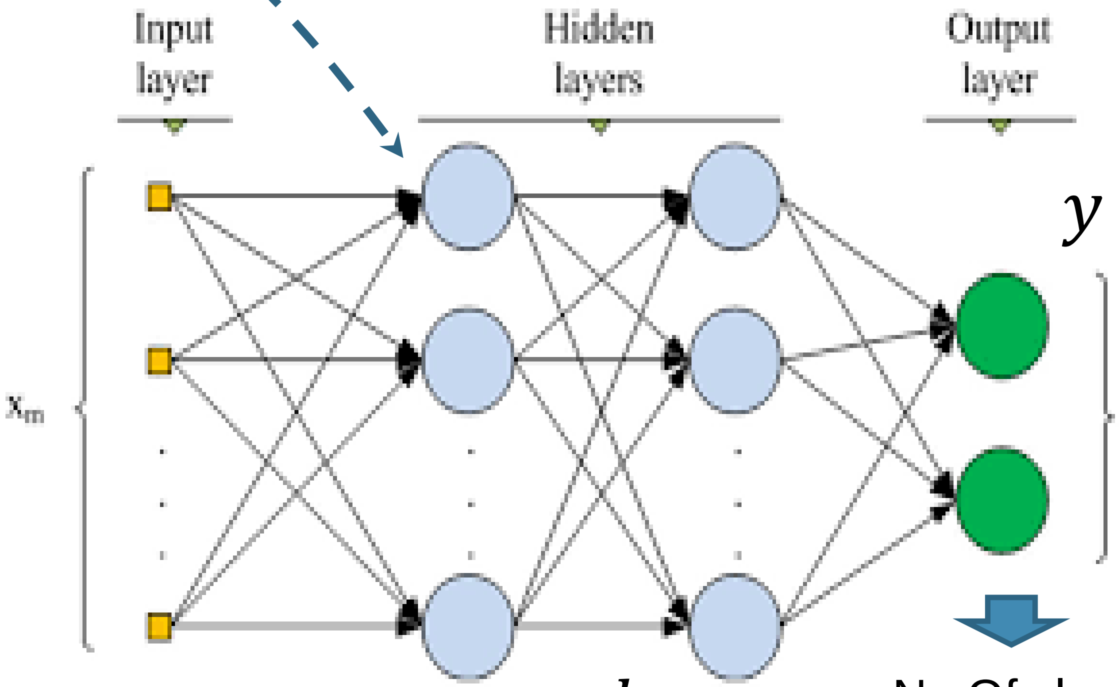
CNN – Neural Net classifier

Single Neuron architecture



Y += Wi * Xi + b :MAC operation

MLP: Fully Connected Neural Net



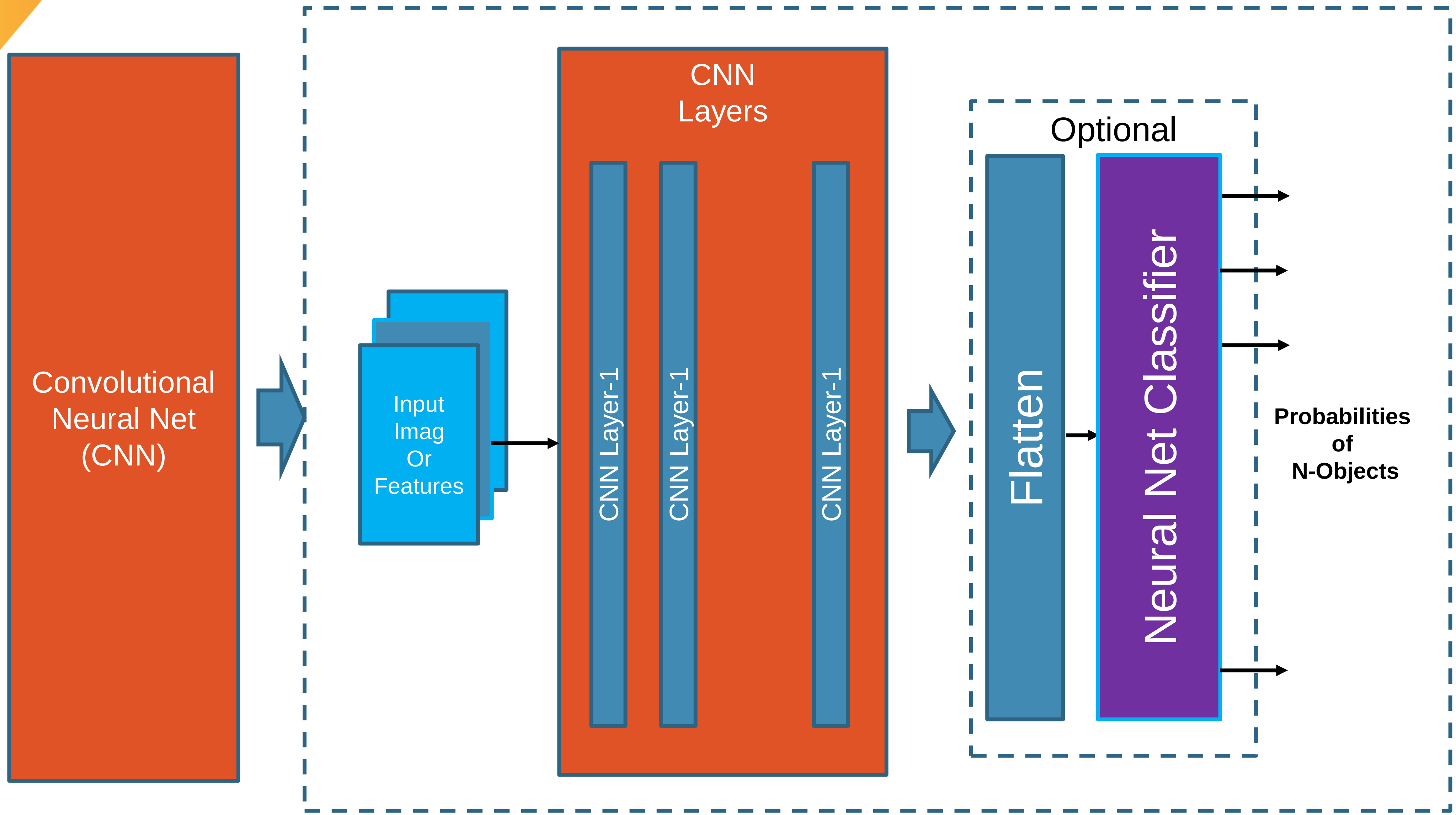
$$y = f(x)$$

l = 1 W_{ij}^{l} l = 2

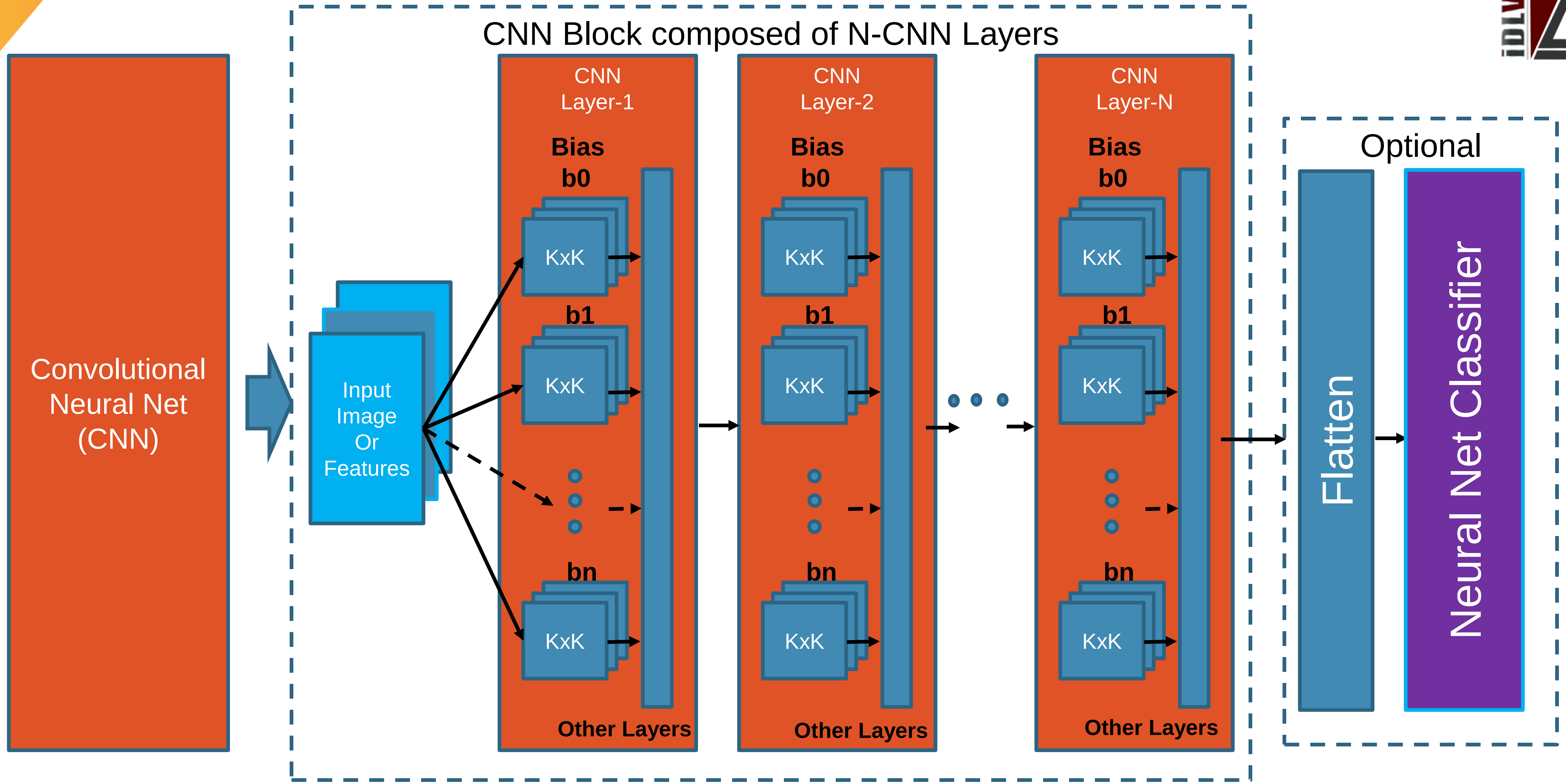
No Of classes

Activation function = Nonlinear function => exp, tanh etc

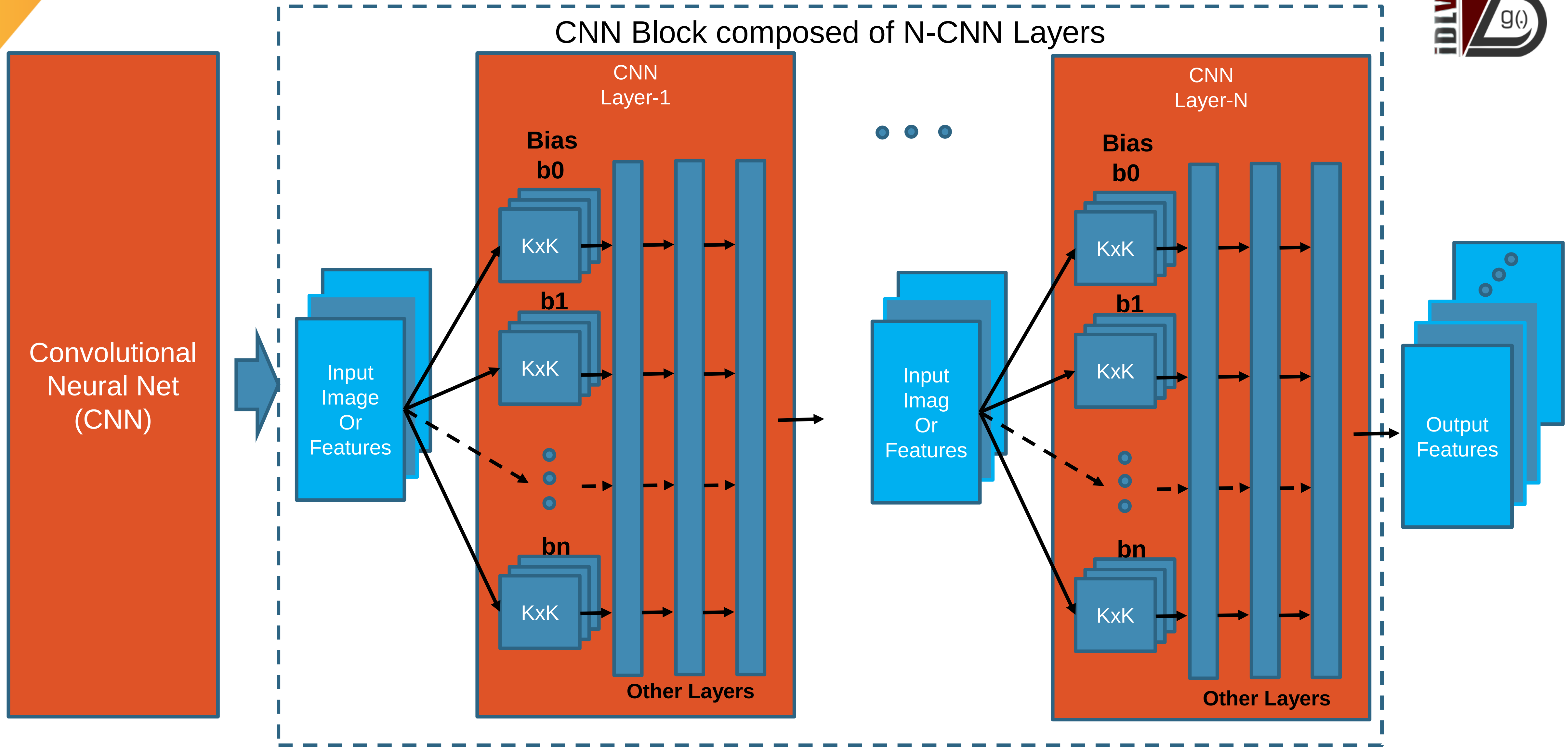
Convolutional Neural Net (CNN) Model



CNN Layers

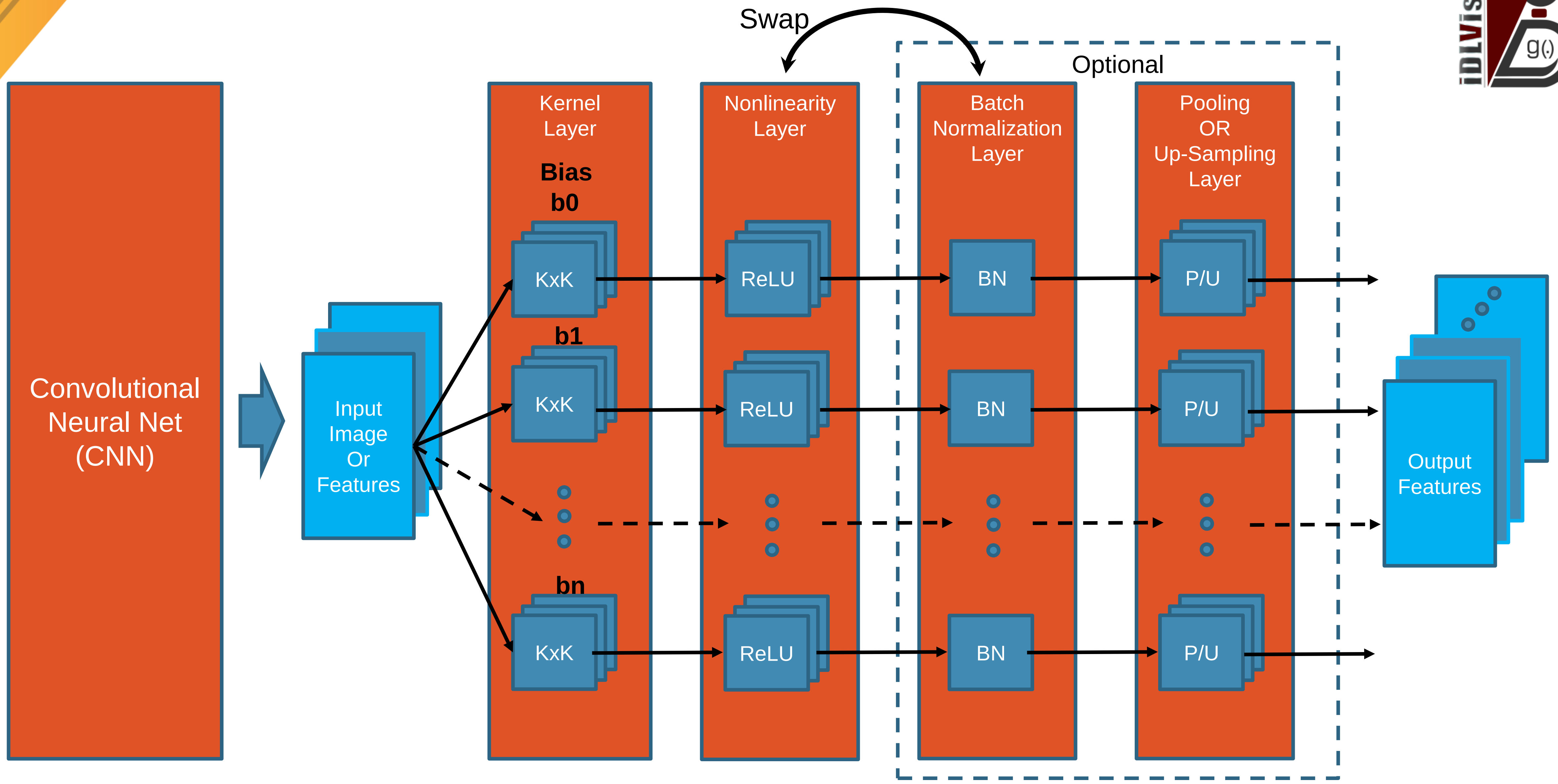

Convolutional Neural Net (CNN)
CNN Block composed of N-CNN Layers
Input Image Or Features
CNN Layer-1
Bias
b0
KxK
b1
KxK
bn
KxK
Other Layers
CNN Layer-2
Bias
b0
KxK
b1
KxK
bn
KxK
Other Layers
CNN Layer-N
Bias
b0
KxK
b1
KxK
bn
KxK
Other Layers
Optional
Flatten
Neural Net Classifier

CNN Layers



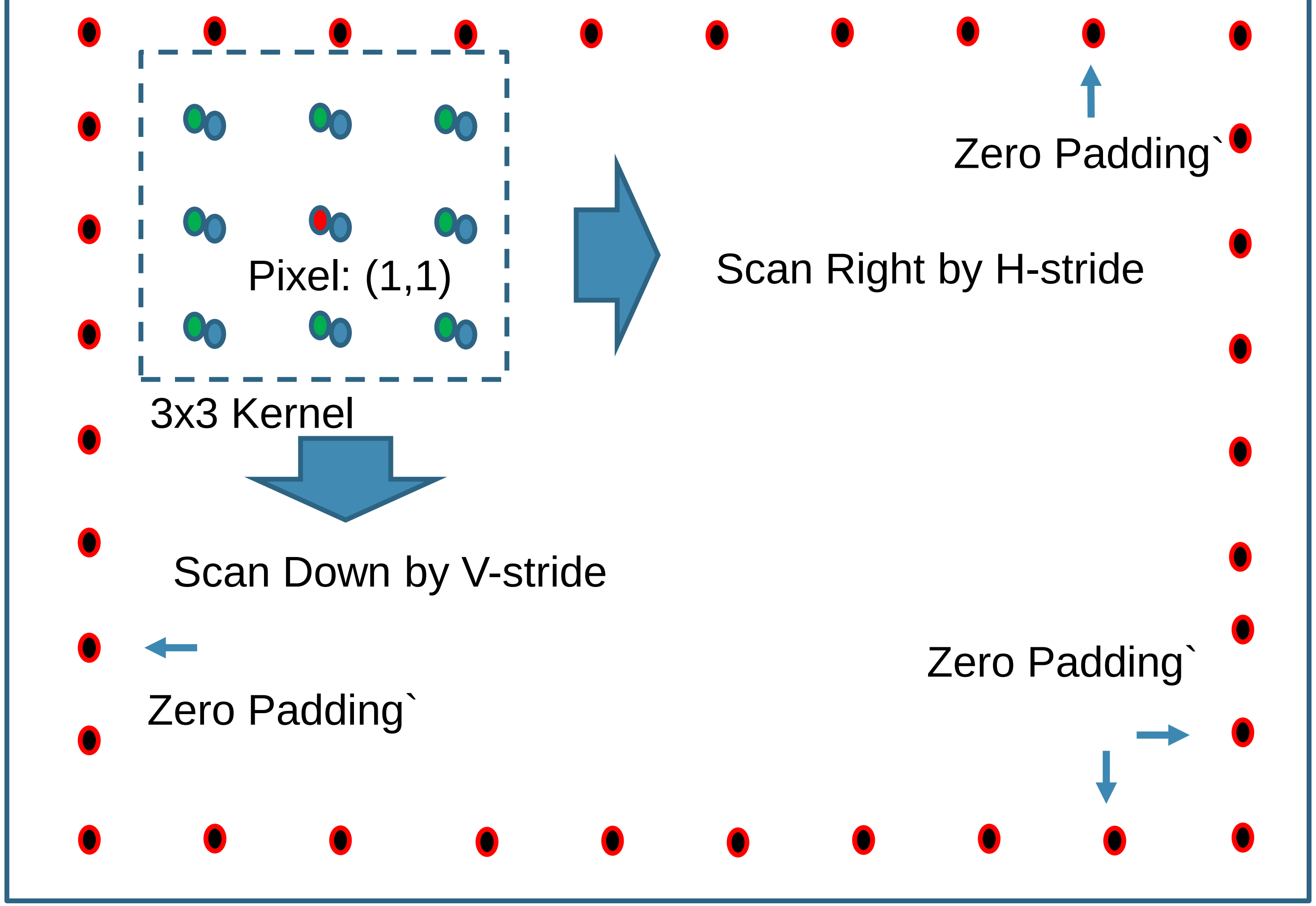
Convolutional Neural Net (CNN)
CNN Block composed of N-CNN Layers
Input Image Or Features
CNN Layer-1
Bias
b0
KxK
b1
KxK
bn
KxK
Other Layers
Input Imag Or Features
CNN Layer-N
Bias
b0
KxK
b1
KxK
bn
KxK
Other Layers
Output Features

CNN – One Layer

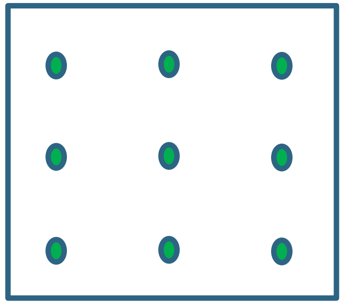


CNN – Kernel Filtering operation

3x3 Dot Product Kernel computation for pixel: (1,1)



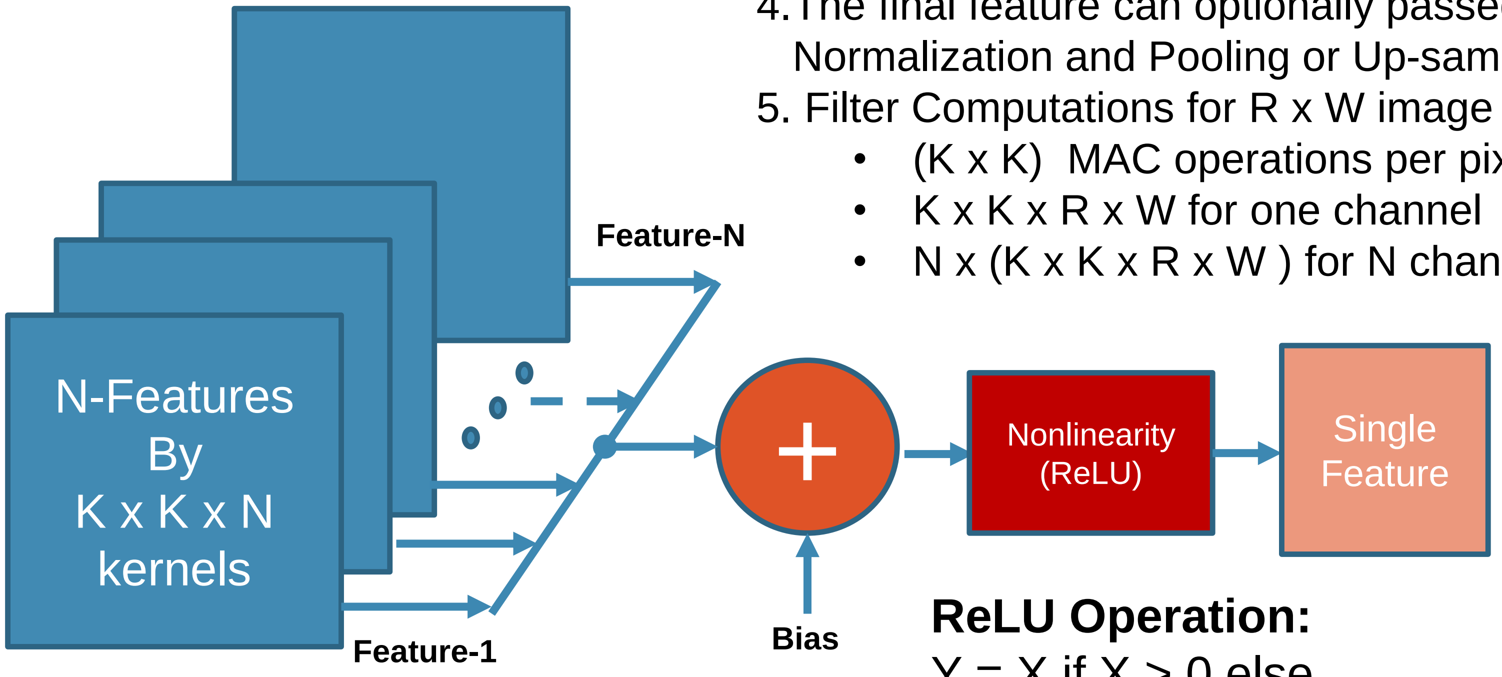
3x3 Kernel



1. Padding is optional but used 99%
2. It keeps size of filtered output feature image same
3. Channels of kernel = number of input features
4. Number of kernels is defined by user. Typically, gets doubled to next layer
5. In kernel convolution operation
   - Dot product is calculated for pixel with kernel center is position on pixel
   - Kernel is moved in Raster scan fashion in horizontal and then next row by stride amount
   - Stride is typically(1,1)
   - Higher value of stride is used to reduce the size of feature output in orde to reduce computations in DL model

CNN – Single Kernel Filtering Operation

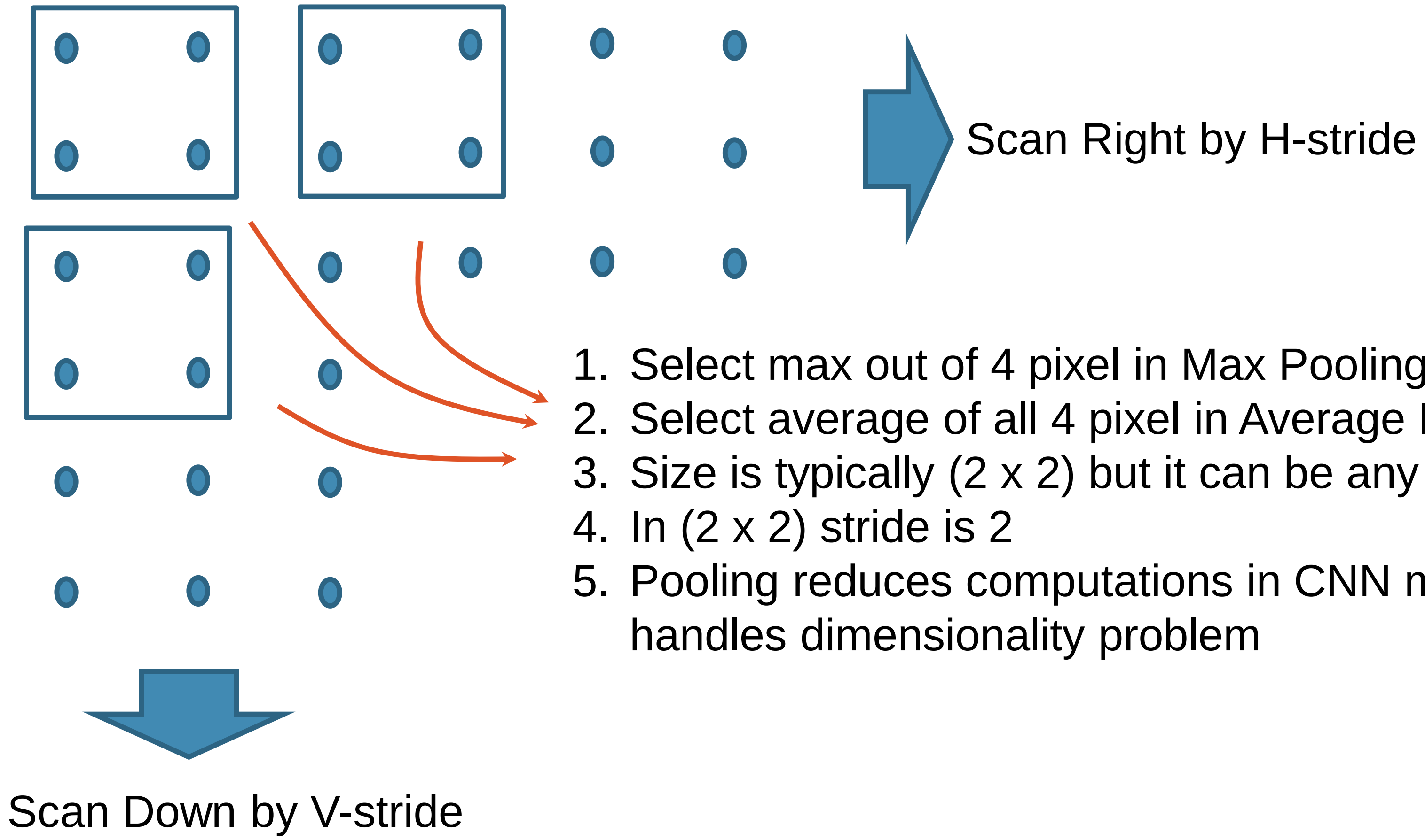
1. N channels of (K x K) size kernel produces N filtered images / features
2. All filtered images are added at pixel level with bias to generate single image / feature
3. Output of adder is passed through Nonlinear operation for each pixel and generates final feature
4. The final feature can optionally passed through Batch Normalization and Pooling or Up-sampling
5. Filter Computations for R x W image / feature
   - (K x K) MAC operations per pixels
   - K x K x R x W for one channel
   - N x (K x K x R x W) for N channels



ReLU Operation:
Y = X if X > 0 else
Y = 0

CNN – Pooling Operation

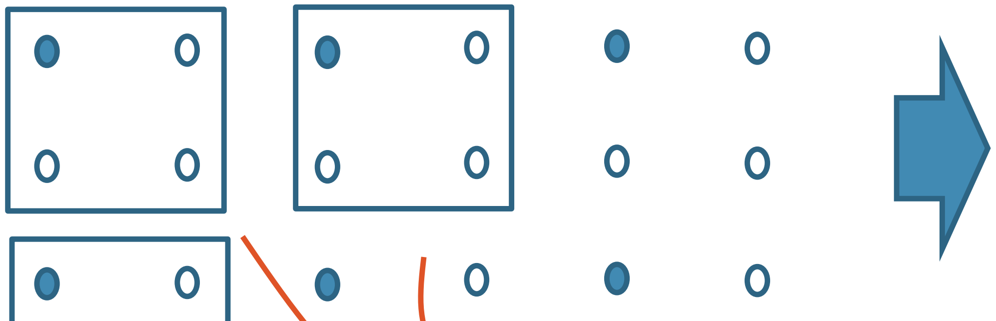
2x2 Pooling size



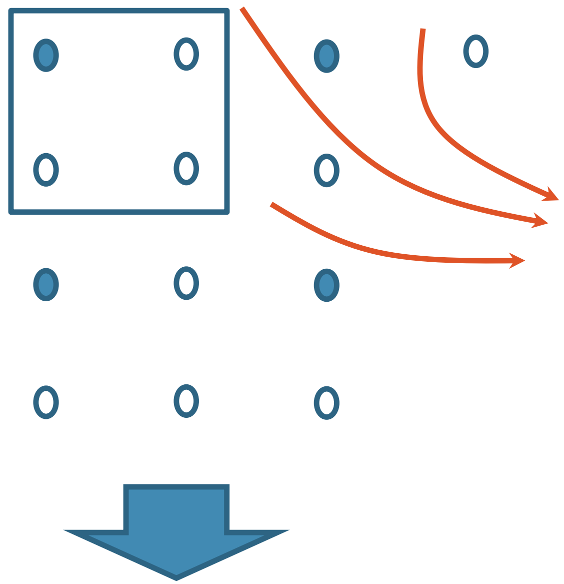
1. Select max out of 4 pixel in Max Pooling
2. Select average of all 4 pixel in Average Pooling
3. Size is typically (2 x 2) but it can be any integer
4. In (2 x 2) stride is 2
5. Pooling reduces computations in CNN model and handles dimensionality problem

CNN – Nearest Up-sampling Operation

2x2 Nearest Up-sampling



Scan Right by H-stride



1. Replicate 3 zero pixel by original pixel
2. Size is typically (2 x 2) but it can be any integer
3. Up-sampling:
    - Improves resolution for better features
    - Required in adder and concatenate blocks to add from previous outputs of higher resolutions
4. Bilinear up-sampling can be used for better quality at the cost of increase computations

Scan Down by V-stride

CNN – Design Frameworks

1. Caffe – Opensource from Yann Lecun

2. Caffe-2 – Opensource improved version of Caffe

3. Theano

4. CNTK – Microsoft

5. Torch – Facebook

6. Tensorflow – Google – Difficult to understand

7. Keras – Opensource: Simple to use and very popular

CNN – MNIST example

```
# Include keras libraries to build a DL CNN model

from __future__ import print_function

import keras
from keras.datasets import mnist
from keras.models import Sequential

from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D
from keras import backend as K
```

CNN – MNIST example

```
# Define parameters for DL CNN model
batch_size = 128
num_classes = 10
epochs = 12

# Input image dimensions
img_rows, img_cols = 28, 28

# Data, segregate for train and test sets
(x_train, y_train), (x_test, y_test) = mnist.load_data()

if K.image_data_format() == 'channels_first':
    x_train = x_train.reshape(x_train.shape[0], 1, img_rows, img_cols)
    x_test = x_test.reshape(x_test.shape[0], 1, img_rows, img_cols)
    input_shape = (1, img_rows, img_cols)
else:
    x_train = x_train.reshape(x_train.shape[0], img_rows, img_cols, 1)
    x_test = x_test.reshape(x_test.shape[0], img_rows, img_cols, 1)
    input_shape = (img_rows, img_cols, 1)
```

CNN – MNIST example

```
# Normalize data
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
x_train /= 255
x_test /= 255

# Print info
print('x_train shape:', x_train.shape)
print(x_train.shape[0], 'train samples')
print(x_test.shape[0], 'test samples')

# Convert class vectors to binary class matrices
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)
```

CNN – MNIST example

```
# Design CNN Model

model = Sequential()

# Add CNN layers
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu’, input_shape=input_shape))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

# Flatten data before feed it to NN layer
model.add(Flatten())

# Define NN layer
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax’))
```

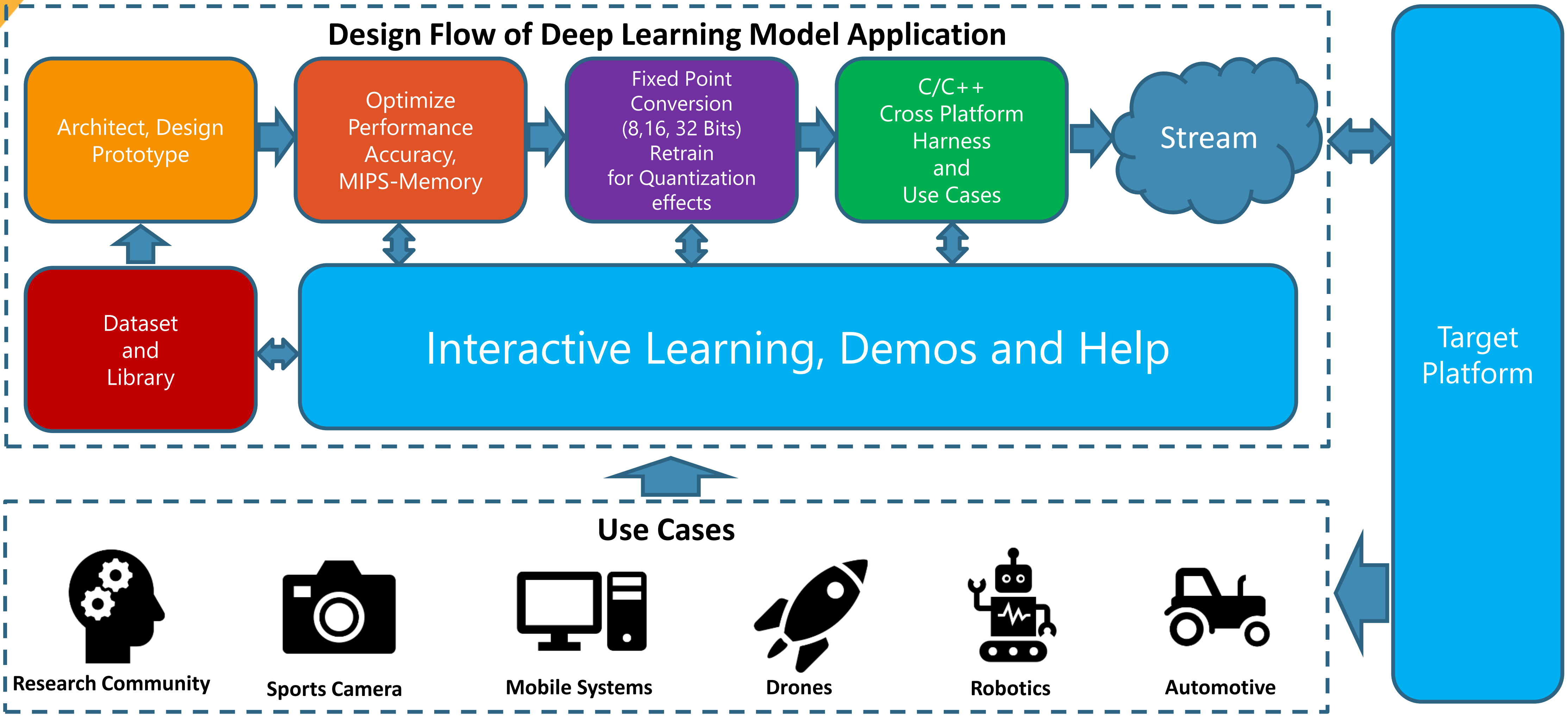
CNN – MNIST example

```
# Generate the CNN model
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adadelta(),
              metrics=['accuracy'])

# Train the DL CNN model
model.fit(x_train, y_train,
          batch_size=batch_size,
          epochs=epochs,
          verbose=1,
          validation_data=(x_test, y_test))

# Evaluate the DL CNN model
score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

DL Model CNN - Design Flow



Thank You !!!

iDLVision Tech

www.idlvision.com

Dr. Ganesh Bhokare

Q & A