Tribhuvan University
Institute of Engineering
Khwopa College of Engineering
Libali, Bhaktapur-8

POSE ESTIMATION

Prepared By:
Niranjan Bekoju

Pose Estimation

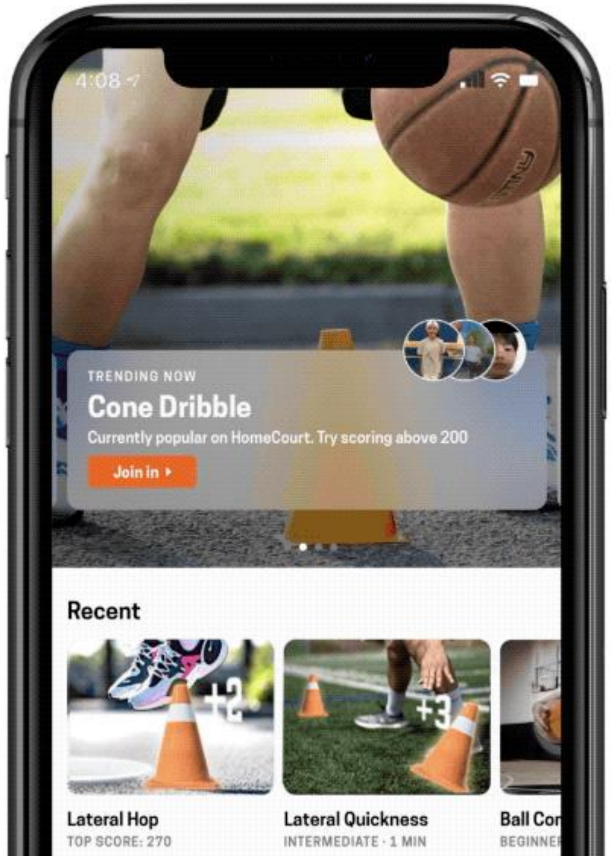
- Computer Vision Technique
- Understanding people in images and videos
- The problem of localization of human joints
- No personal identification information
- Estimates where body joints are

Why is this Exciting to begin with?

- Interactive installations that react to body to Augmented Reality
- Animation
- Fitness Uses
- Action Recognition
- Home Court

Home Court

- It is interactive, AI-powered mobile app that puts live action sports inside a game.
- It use Pose Estimation to analyze player movement



https://apps.apple.com/us/app/homecourt-the-basketball-app/id1258520424

Types of Pose Estimation

- 2D Pose Estimation
  - Estimate a 2D pose (x,y) co-ordinates for each joints from a RGB image

- 3D Pose Estimation
  - Estimate a 3D pose (x,y,z) co-ordinates for each joints from a RGB image

Let's Start with PoseNet

- Used to estimate either single pose or multiple pose

- There is two version of Pose Net:
  - One for Single Pose (That can detect only one person)

  - Second for Multiple Pose (That can detect multiple persons)

Why there are Two Version?

- The Single Person Pose Detector is faster and simpler but requires only one subject present in the image

- The Multiple Person Pose Detector is slower but can detect multiple subject present in the image

Phase of Pose Estimation

- At High level phase, there are two phase of Pose Estimation
  - An *input RGB image* is fed through a convolutional neural network.

  - Either a single-pose or multi-pose decoding algorithm.

What pose decoding Algorithm decodes?

- Pose
  - A pose object that contains a keypoints

- Pose Confidence Score
  - Determines the overall confidence in the estimation of Pose

  - Values ranges between 0 to 1

  - Used to hide pose that are not strong enough

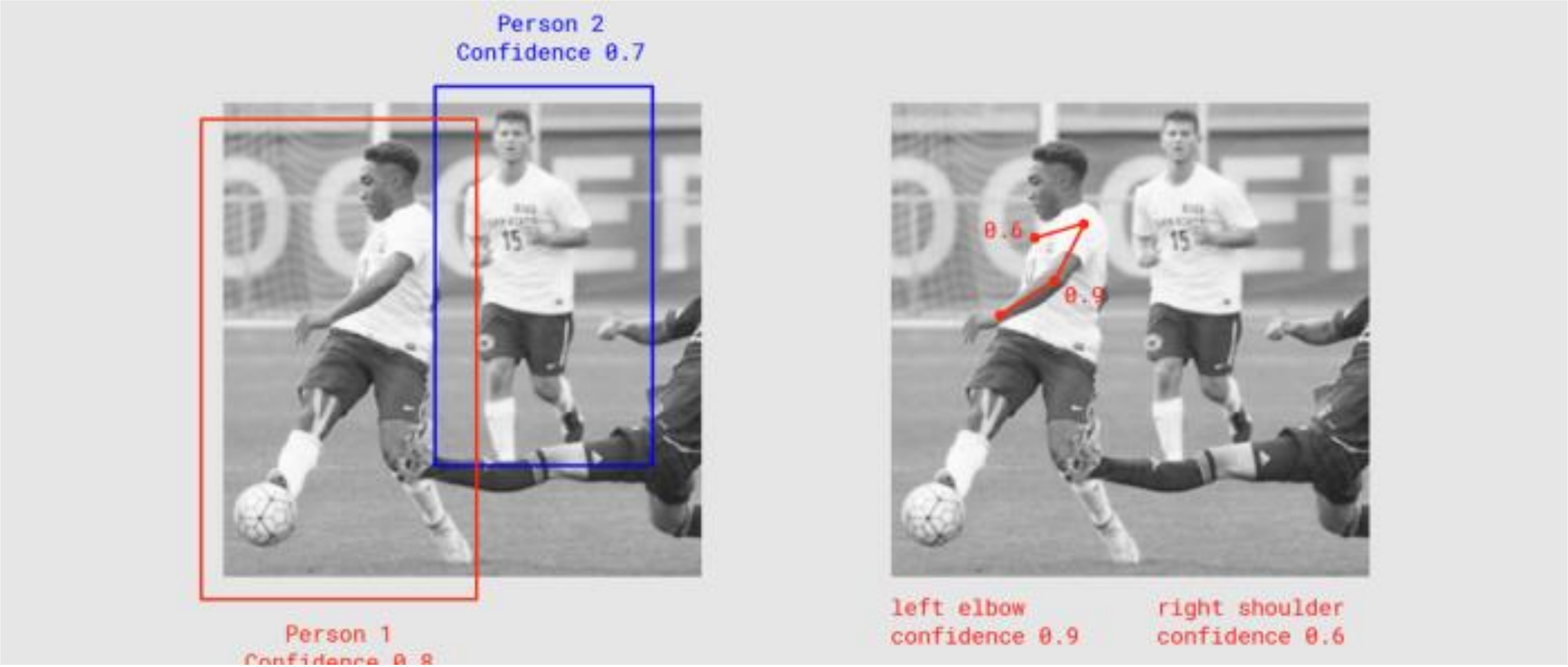
What pose decoding Algorithm decodes?

- Keypoint
  - A part of a person's pose that is estimated
    - Nose, ear, eyes, knees, ankle.

What pose decoding Algorithm decodes?

- Key Point Confidence Score
  - The confidence that an estimated keypoints position is accurate
  - Values ranges between 0 to 1
  - Used to hide pose that are not strong enough
- Key Point Position
  - 2D x and y co-ordinate (x, y) of each keypoints

Pose and Keypoint Confidence Score



Output of Single Pose Decoding Algorithm


```
{
  "score": 0.32371445304906,
  "keypoints": [
    { // nose
      "position": {
        "x": 301.42237830162,
        "y": 177.69162777066
      },
      "score": 0.99799561500549
    },
    { // left eye
      "position": {
        "x": 326.05302262306,
        "y": 122.9596464932
      },
      "score": 0.99766051769257
    },
      ...
  ]
}
```

Output of Multiple Pose Decoding Algorithm

```
[
 { // pose #1
   "score": 0.42985695206067,
   "keypoints": [
     { // nose
       "position": {
         "x": 126.09371757507,
         "y": 97.861720561981
        },
       "score": 0.99710708856583
     },
     ...
   ]
  },
 { // pose #2
   "score": 0.13461434583673,
   "keypoints": [
     { // nose
       "position": {
         "x": 116.58444058895,
         "y": 99.772533416748
       },
     "score": 0.9978438615799
     },
     ...
   ]
 },
 ...
]
```

Single Pose Estimation

- Simpler and Faster
- Ideal use Case (When there is only one person centered in an input image or video)
- If multiple person in an image, keypoints from both person will likely be estimated as part of same pose.
- If image contain multiple person, the multi pose estimation algorithm should be used

Technical Deep Dive

- Two model of Pose Net
  - Resnet
  - Mobile Net
- Accuracy(Resnet) > Accuracy(Mobile Net)
- Resnet is large size and have many layers
- Mobile net has small size and few layers
- Mobile net is designed to run on mobile devices
- Mobile net will be used

Revisiting Single Pose Estimation

Output Stride

- Pose Net is image size invariant(Predict Pose is same scale as original images)
- The output stride determines how much we’re scaling down the output relative to the input image size.
- Affect the size of the layers(Speed) and the model output(Accuracy)
- Higher the output stride, smaller the resolution of the layers in the network
- Values = 8, 16, 32

Revisiting Single Pose Estimation

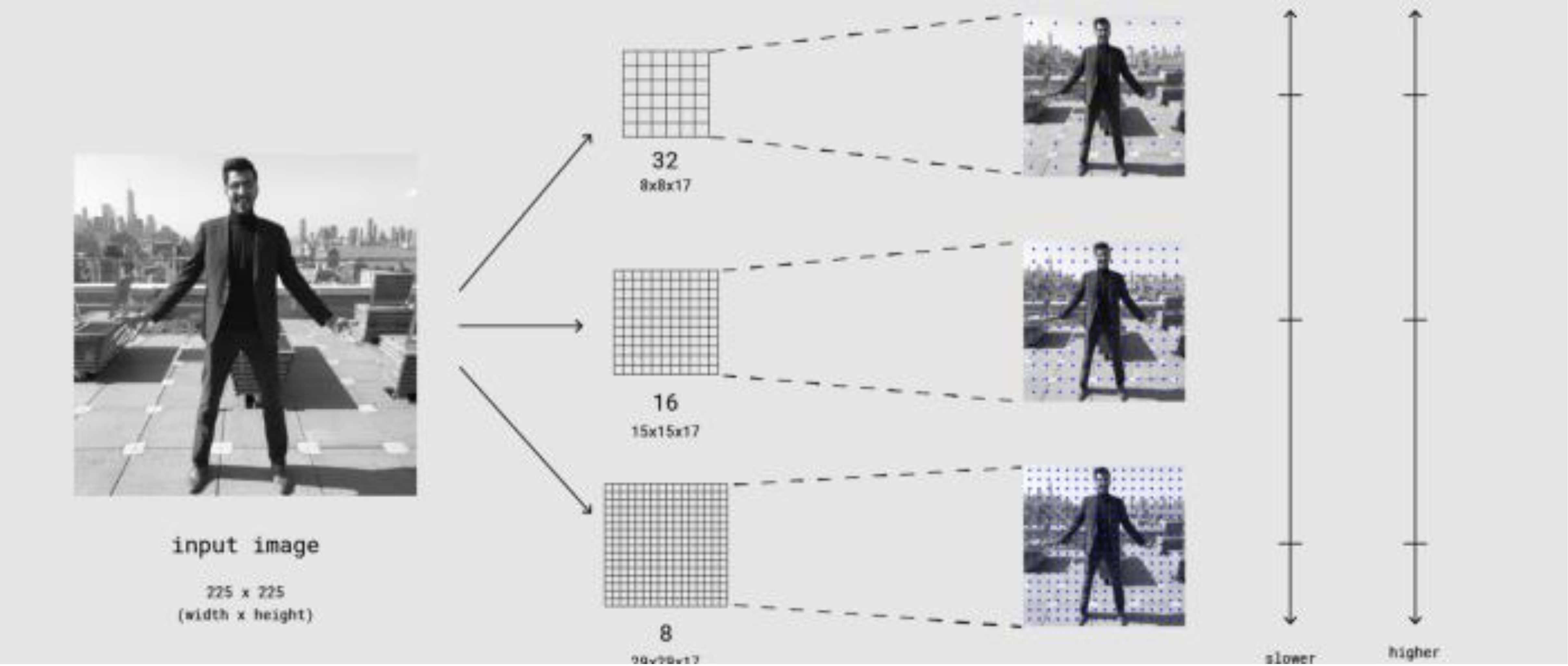
Output Stride
Formula to calculate Output relative to input

Output Resolution = ((Input Image Size –1)/Output Stride) +1

~ Output = input/stride

Note: For the output Stride of 8 and 16, Astrous Convolution is done.
It is not applied when the output stride is 32.

Revisiting Single Pose Estimation



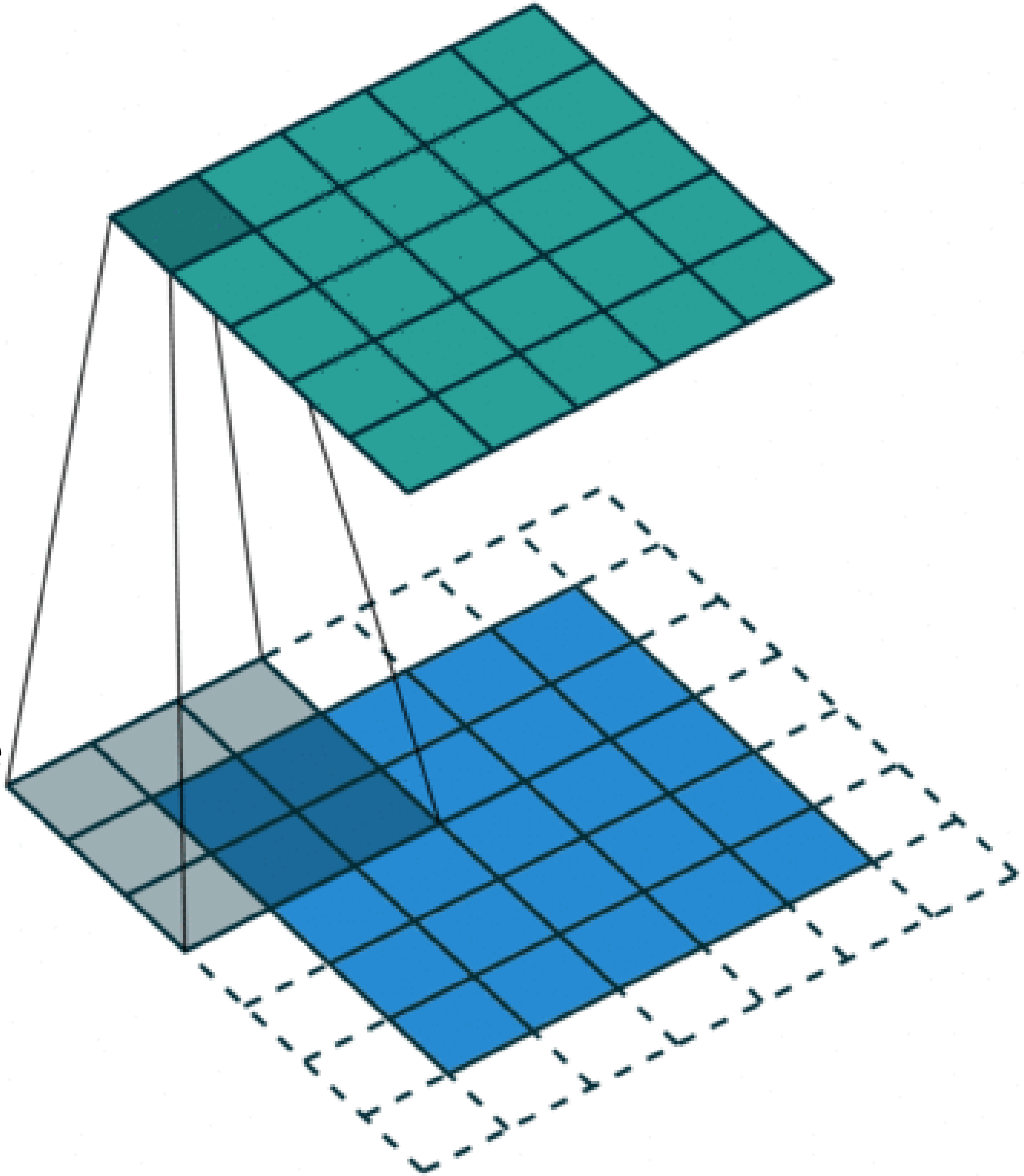
Convolution

For simplicity, we are just focusing on 2D convolution

Parameters That define the convolution layers:

1. Kernel Size
Field of view of the convolution
Generally, 3 i.e. 3 x 3 matrix

2. Stride
Defines the step size of kernel when traversing the image
Usually, 1
We can use a stride of 2 for down sampling
(Similar to Max Pooling)



Convolution

For simplicity, we are just focusing on 2D convolution

Parameters That define the convolution layers:

3. Padding

Defines how the border of a sample is handled.

-> Valid Padding

[(n x n) image] * [(f x f) filter] —> [(n – f + 1) x (n – f + 1) image]

[5,5] * [3,3] ---> [3,3]

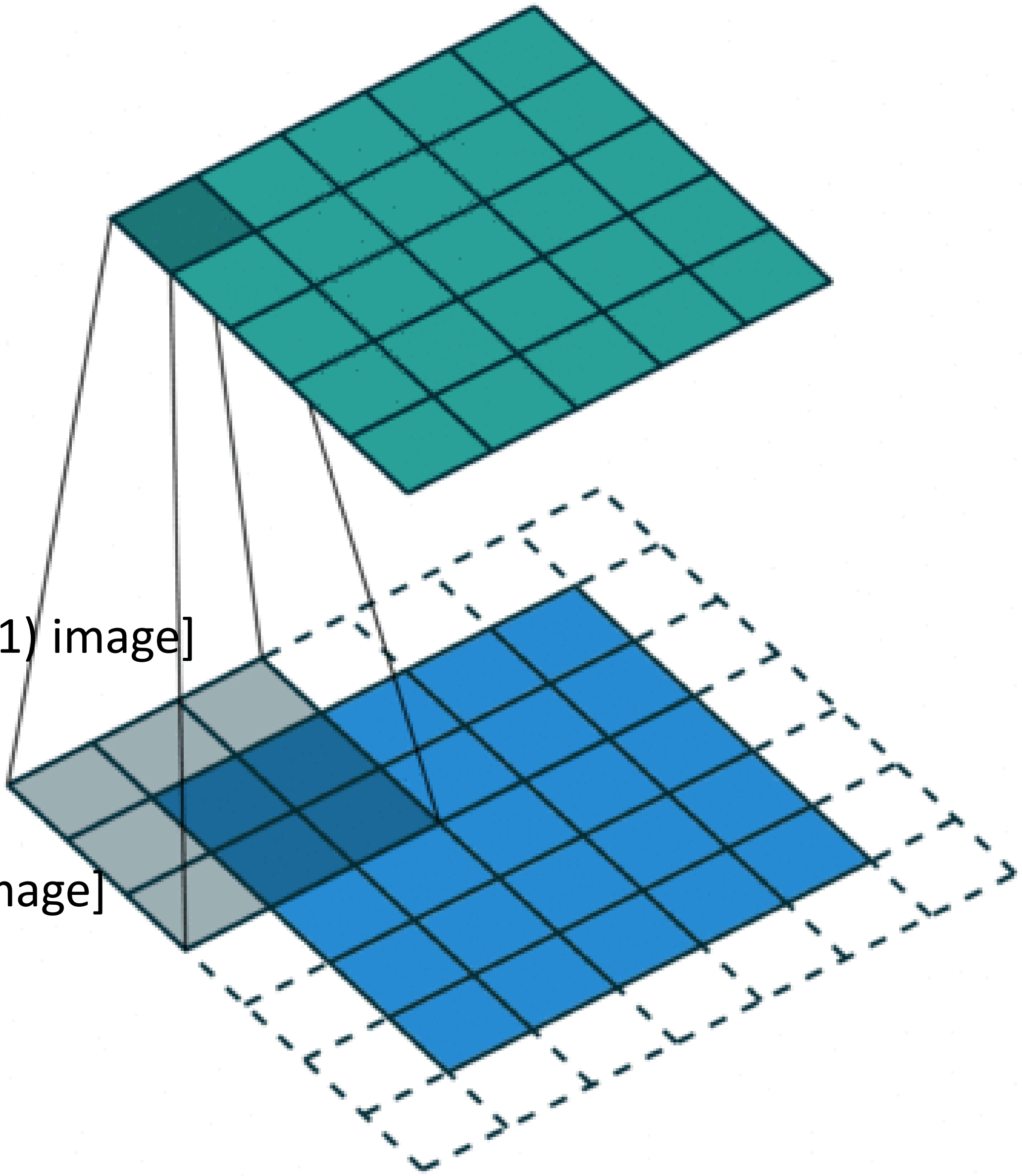
-> Same Padding

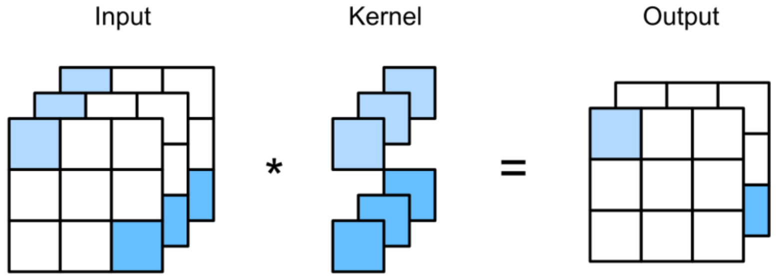
[(n + 2p) x (n + 2p) image] * [(f x f) filter] —> [(n x n) image]

P = (f-1)/2

n=5, f=3,p=1

[7,7]*[3,3]=[5,5]





Convolution

Parameters That define the convolution layers:

4. Input(I) and Output(O) Channel

Total parameter = I * O * k

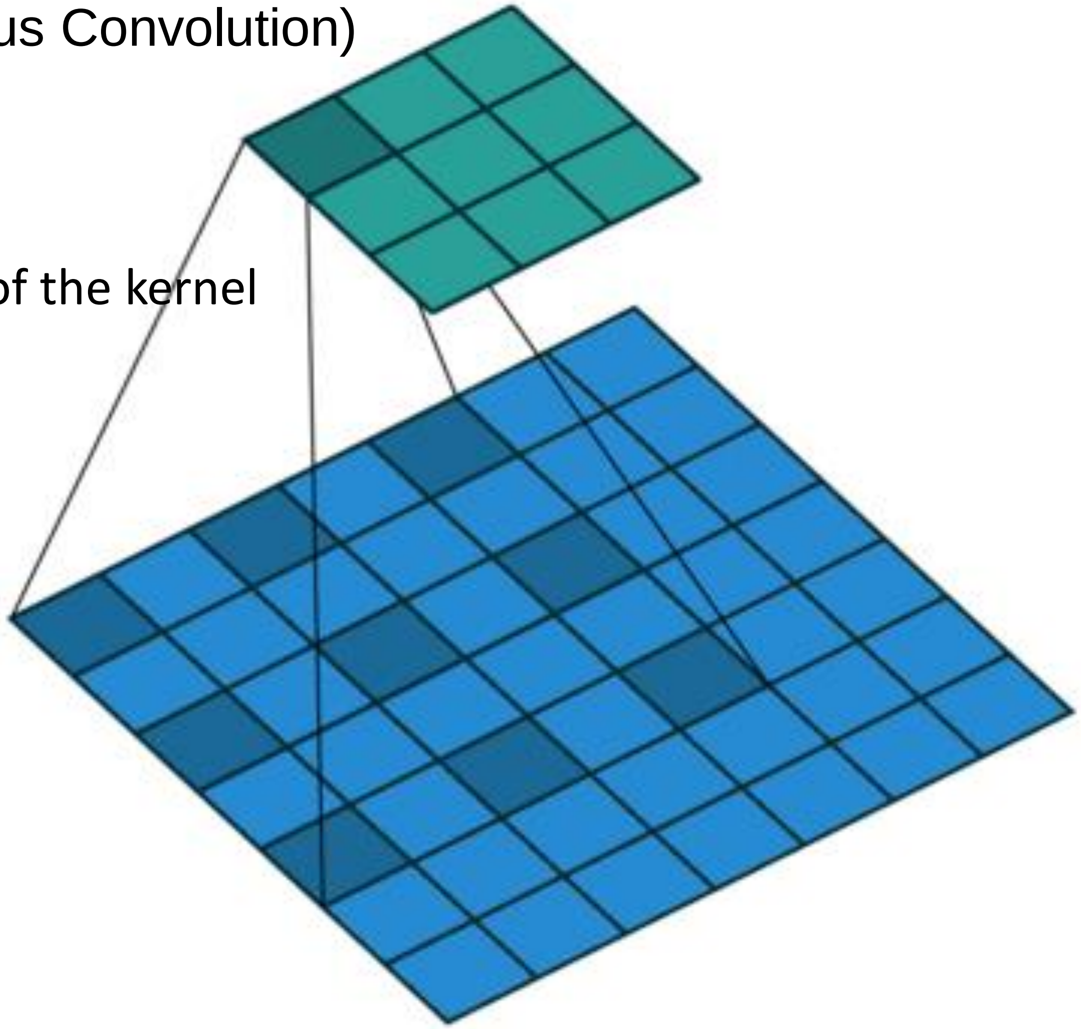
K->no. of values in the kernel

Convolution

Types of Convolution

1. Dilated Convolution
2. Transposed Convolution

Dilated Convolution(a.k.a Atrous Convolution)



Additional Parameter: Dilation rate

Dilation rate defines the spacing between the two values of the kernel

Parameters:
Kernel_size =3
Dilation = 2
Field of view is same as that of kernel_size =5
I.e. field of view = 5*5
No. Of parameter = 9

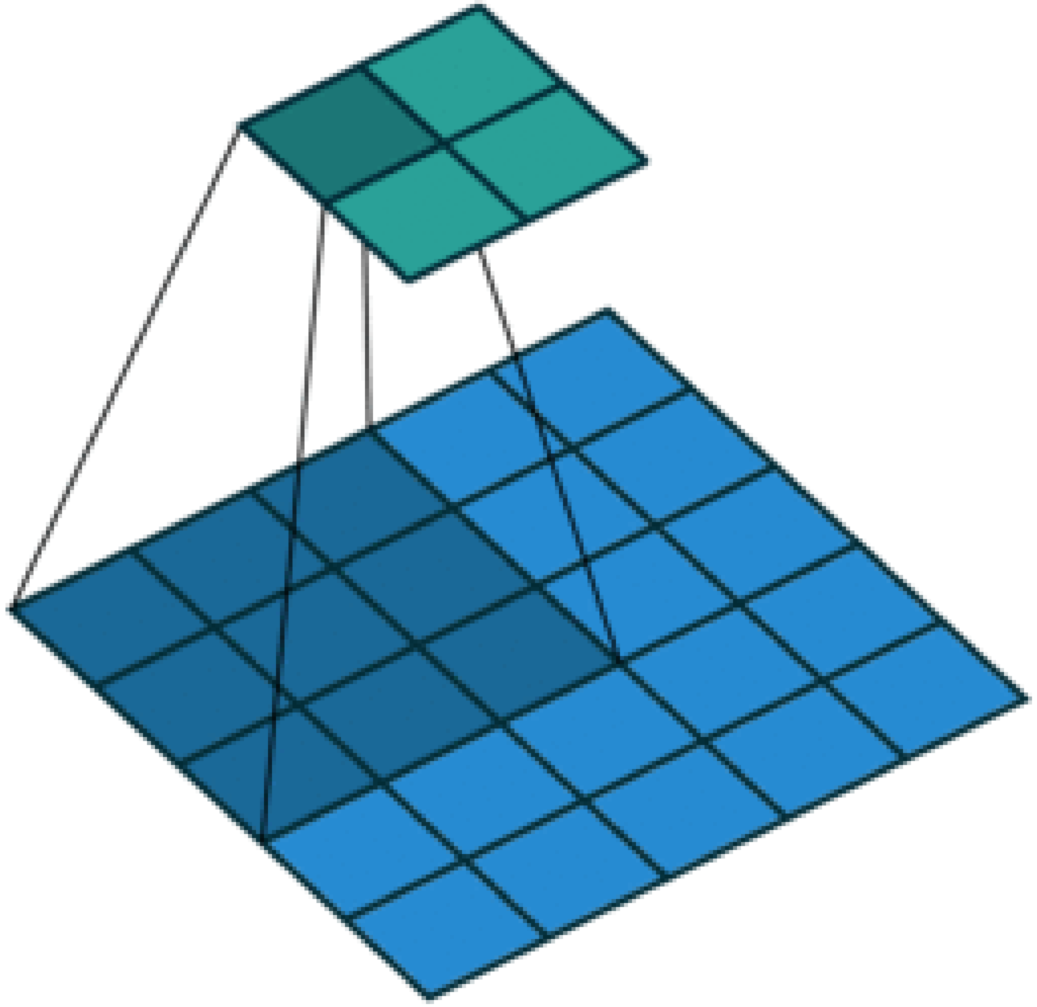
Used in Pose Net with output stride = 8 or 16

Reduce Computational Complexity

Transposed Convolution

No Additional Parameter

Parameters:
Kernel_size =3
field of view = 3*3
No. Of parameter = 9
Stride = 2
Padding = valid

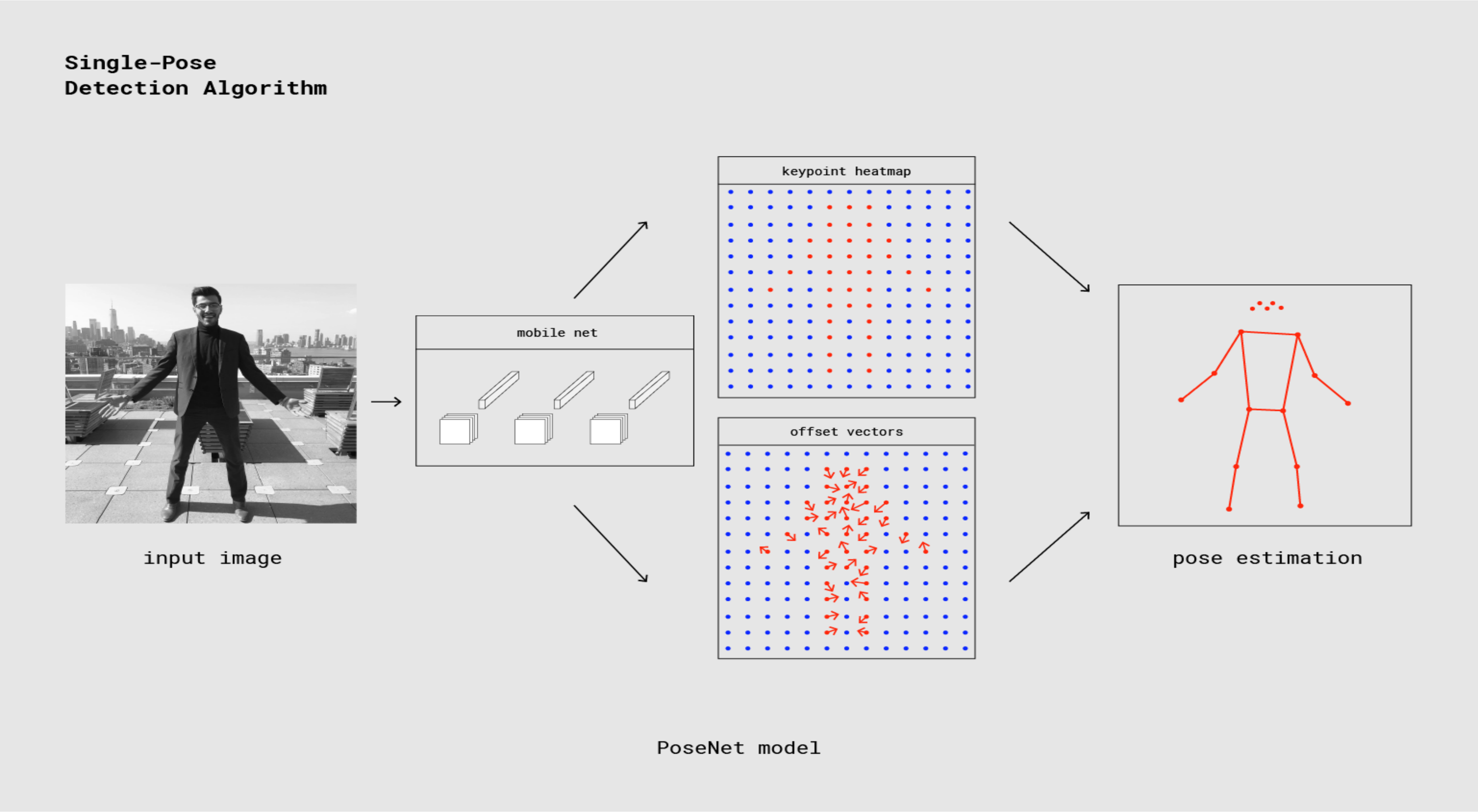


Model Outputs

When Pose net Process an Image using Mobile Net, it outputs
1. Heat Map
2. Offset Vectors

- Used to find the high confidence area of pose key points
- Position of the key points

Model Outputs



Heat Map

- 3D tensor of size (resolution * resolution * 17)

- Here, 17 represent the no. Of keypoints detected by Pose Net

For eg:

Input size = 225

Output stride = 16

Resolution = ((255-1)/16) + 1 = 15

So tensor size = (15*15*17)

Each Position in a tensor has a confidence score

Offset Vector

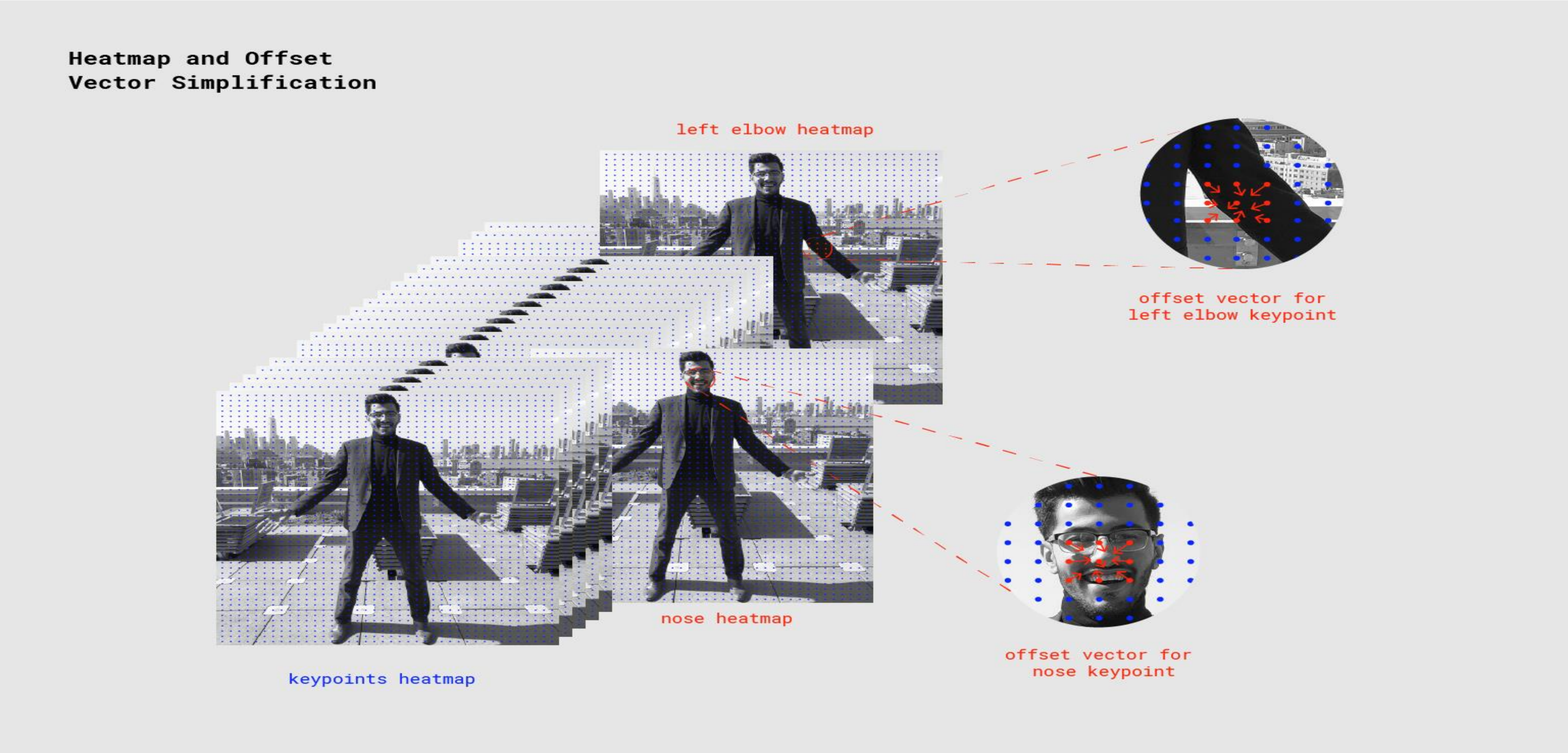
- 3D tensor of size (resolution * resolution * 34)

- Here, 34 = no. of keypoints * 2
- 2 for x and y offset of each point in the output image

- First 17 slices of offset vector contain the x of the vector

- Last 17 slices of offset vector contain the y of the vector

Heat Map and offset Vector

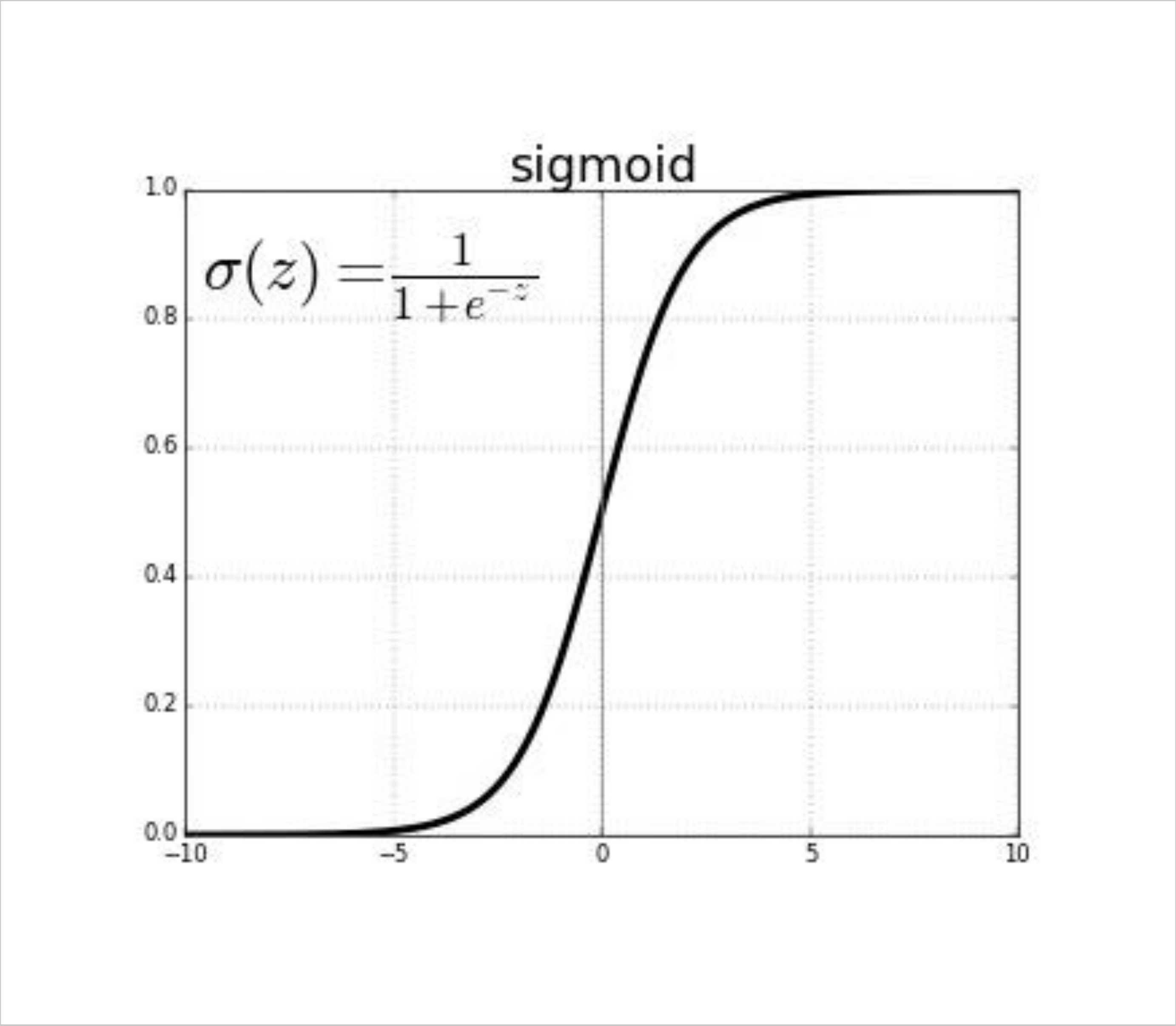


Estimating Pose From the outputs of the Model

- **Calculation**

1. Sigmoid Activation Function

Confidence Score = heatmap.sigmoid()



Estimating Pose From the outputs of the Model

Argmax2D

DONE ON THE KEY POINT CONFIDENCE SCORE

GET X AND Y INDEX IN THE HEAT MAP WITH HIGHEST SCORE FOR EACH KEY POINTS

PRODUCE TENSOR OF SIZE 17*2

Offset Vector

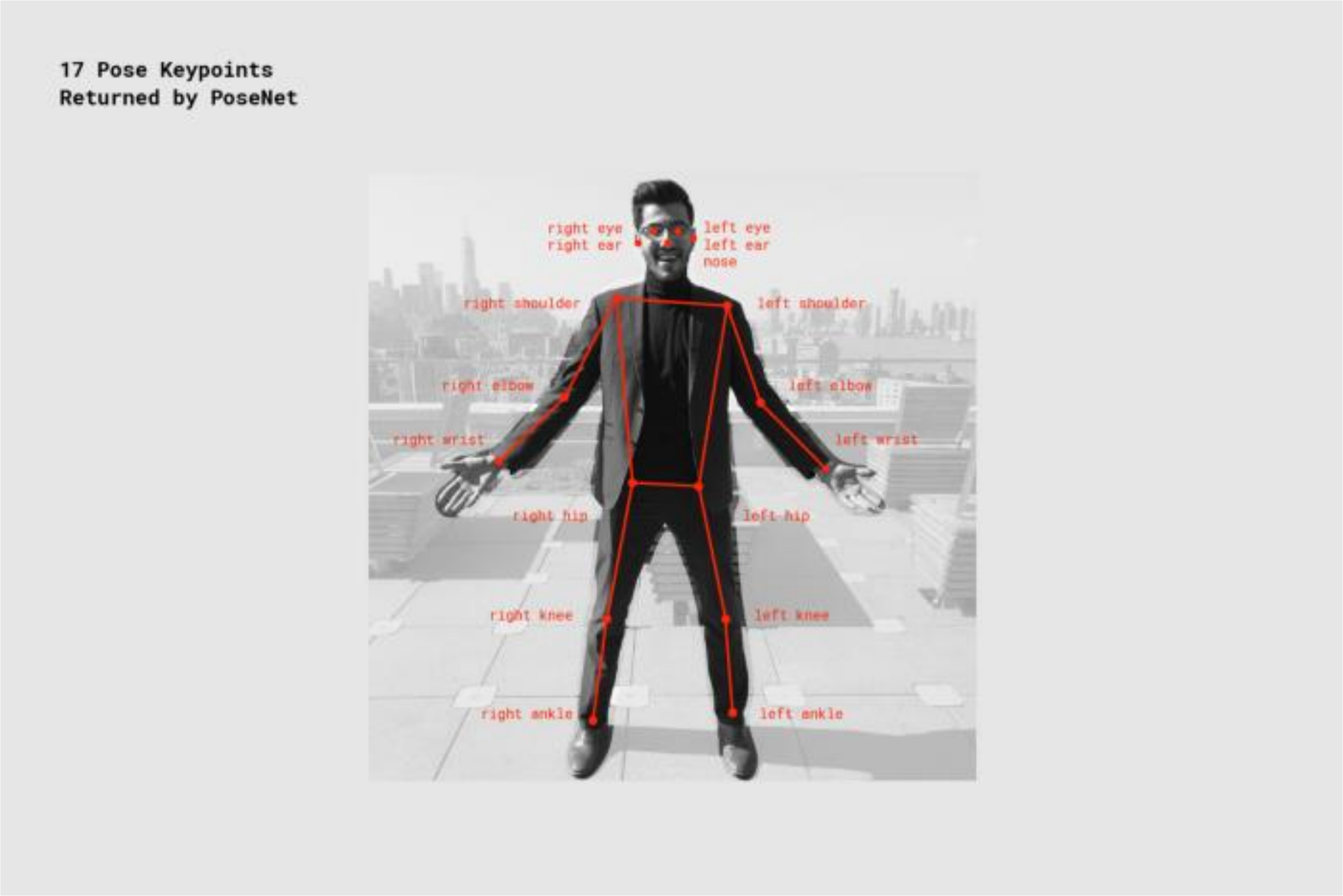
EACH PART IS RETRIVED BY GETTING X AND Y FROM THE OFFSET CORRESPONDING TO THE X AND Y INDEX IN THE HEATMAP

PRODUCE TENSOR OF SIZE (17 *2)

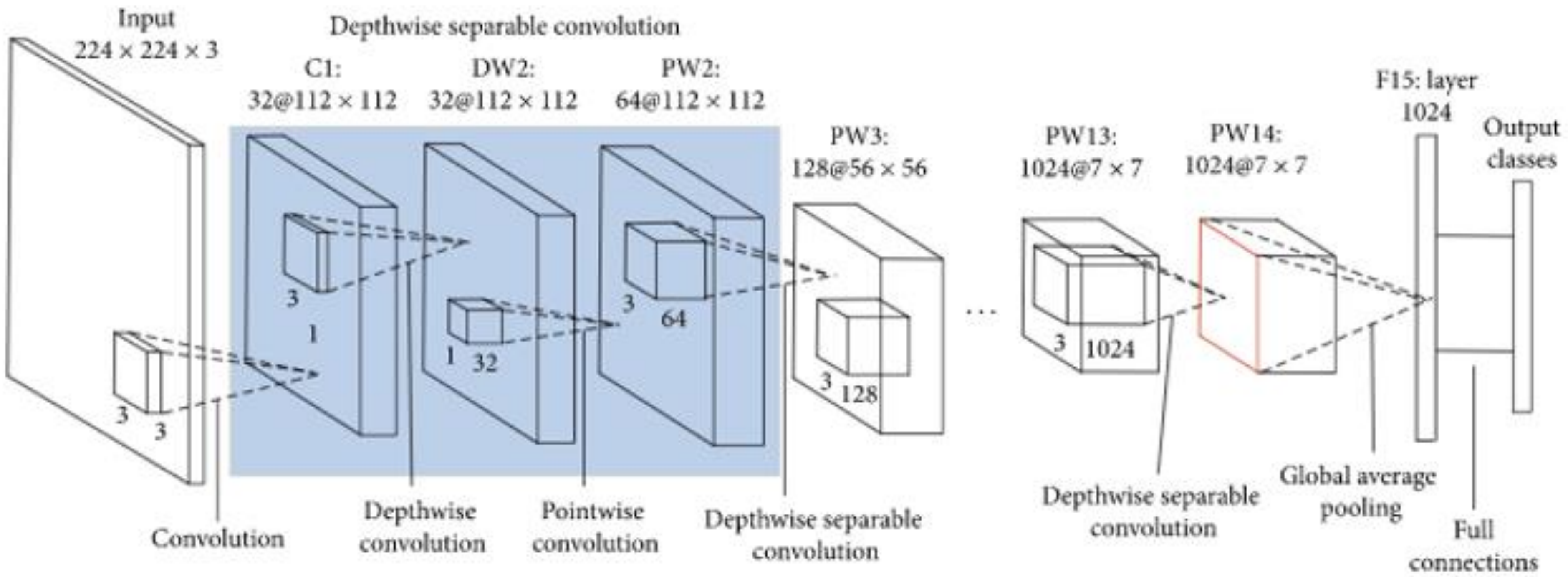
OFFSET VECTOR = [OFFSET.GET(Y,X,K),OFFSET.GET(Y,X,K+17)]

Key Points Position

- Key points position = heatmap Position * output Stride + offset Vector



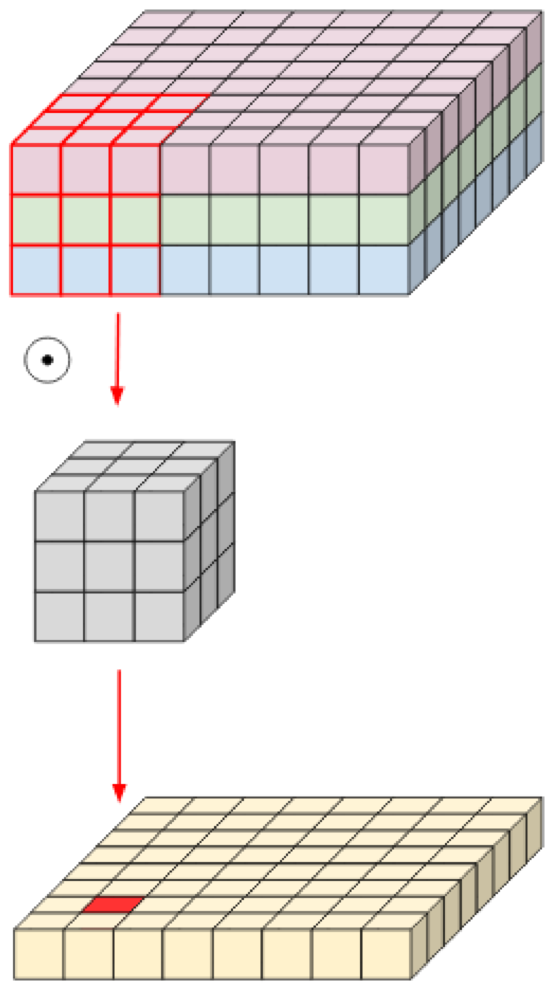
Mobile Net –v2 Architecture



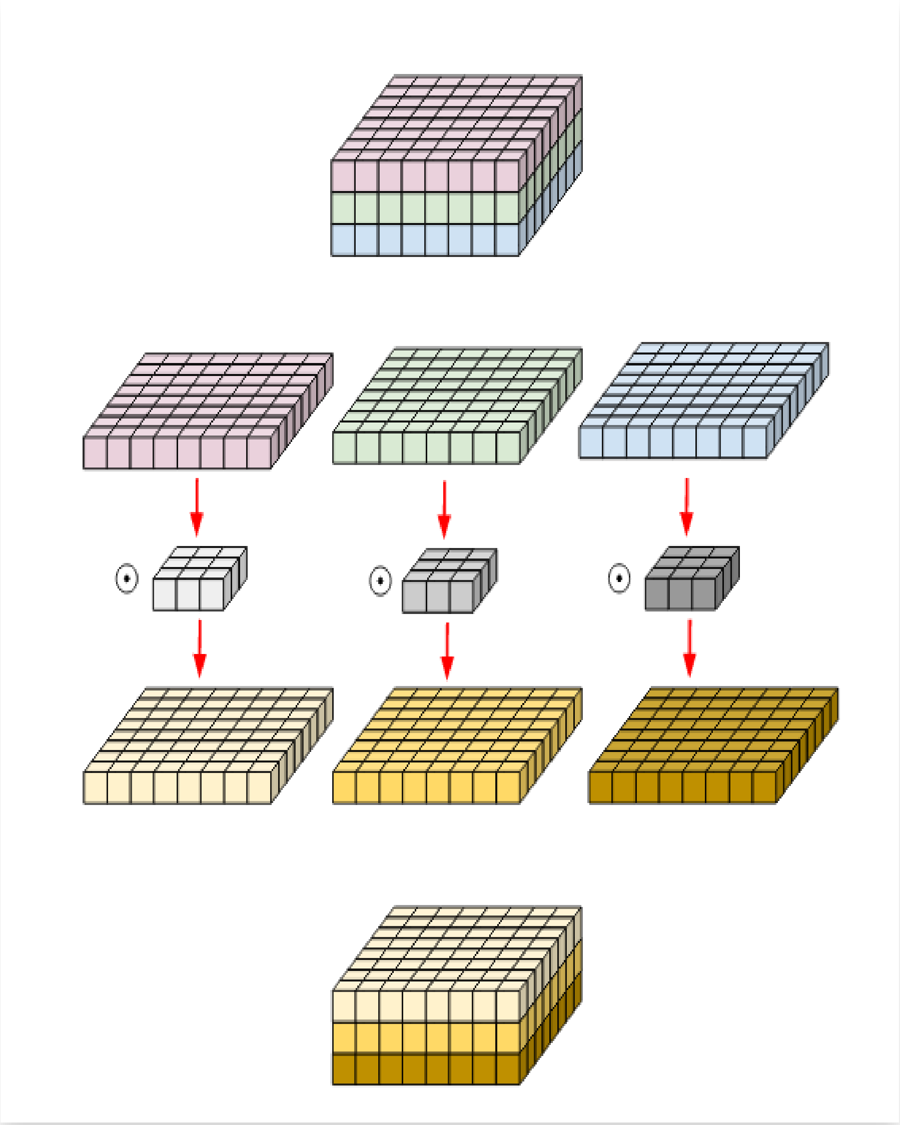
MobileNet-V2 Architecture

Chiung-Yu Chen

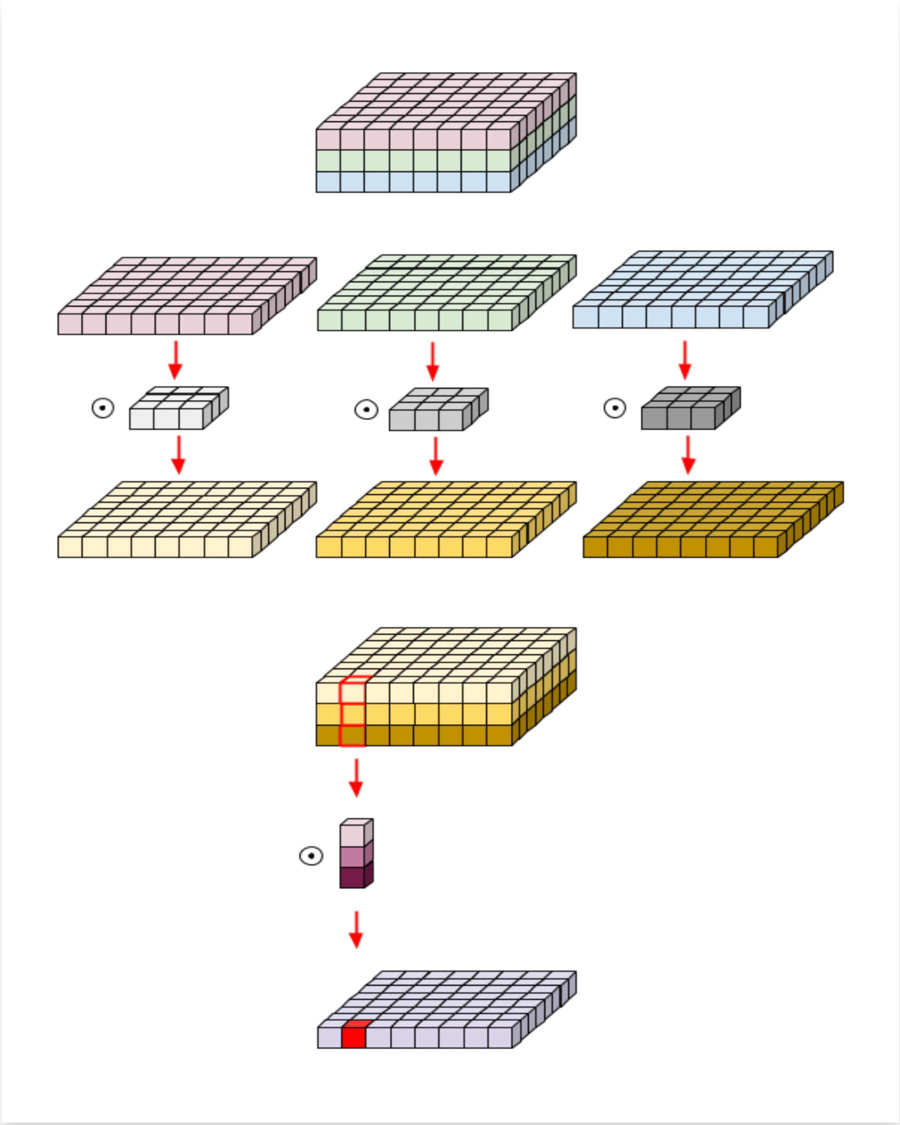
Normal Convolution



Depth Wise Convolution



Depth Wise Separable Convolution



PoseNet using Tensorflow.js

- With PoseNet running on TensorFlow.js anyone with a descent webcam can experience this technology right from a web browser

https://storage.googleapis.com/tfjs-models/demos/posenet/camera.html

References

- https://medium.com/tensorflow/real-time-human-pose-estimation-in-the-browser-with-tensorflow-js-7dd0bc881cd5
- https://medium.com/@zurister/depth-wise-convolution-and-depth-wise-separable-convolution-37346565d4ec
- https://d2l.ai/chapter_convolutional-neural-networks/channels.html
- https://www.geeksforgeeks.org/cnn-introduction-to-padding/
- https://nanonets.com/blog/human-pose-estimation-2d-guide/
- https://towardsdatascience.com/types-of-convolutions-in-deep-learning-717013397f4d
- https://www.hindawi.com/journals/misy/2020/7602384/

Thank You