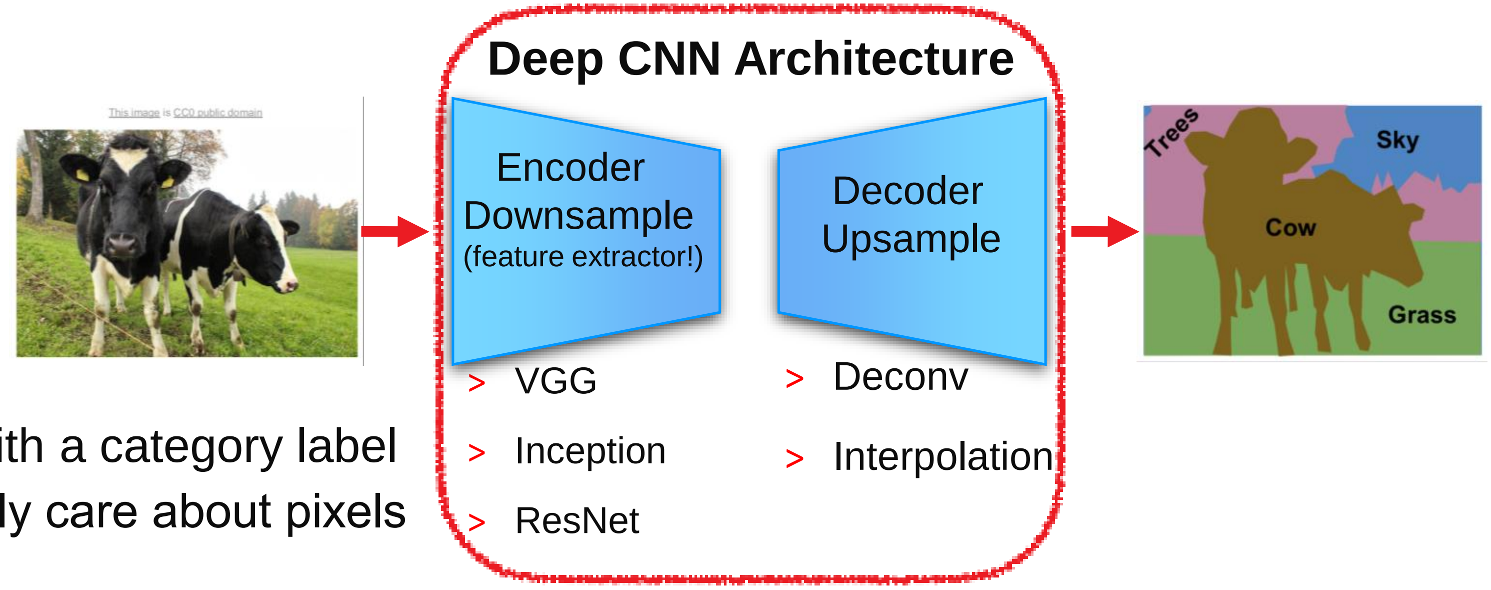
Segmentation Introduction

J. Cory – Embedded Vision Specialist

XILINX

ML Task: Segmentation

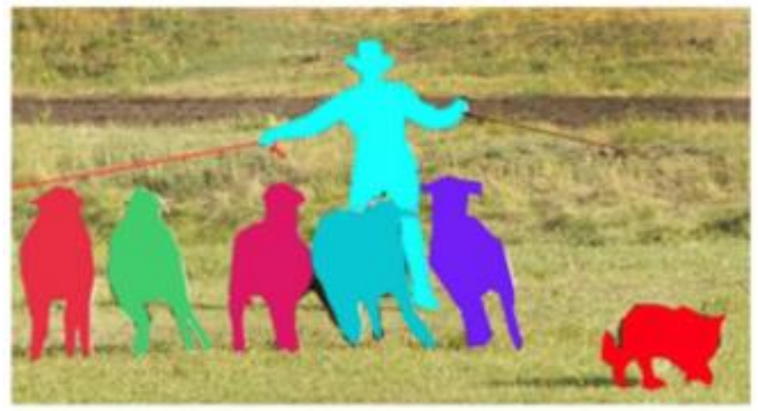
> **Task of semantic segmentation:**
>> Input: Image
>> Output: Image
>>> Label each pixel in the image with a category label
>>> Don't differentiate instances, only care about pixels
>> Evaluation metric: mIOU



Semantic Segmentation

> **Applications of semantic segmentation:**
>> Lane detection, drivable space detection, environmental perception in automotive.

> **Task of instance segmentation:**
>> Input: Image
>> Output: Image
>>> Label different individuals of the same class
>>> Don't care about background
>> Evaluation metric: AP, average precision

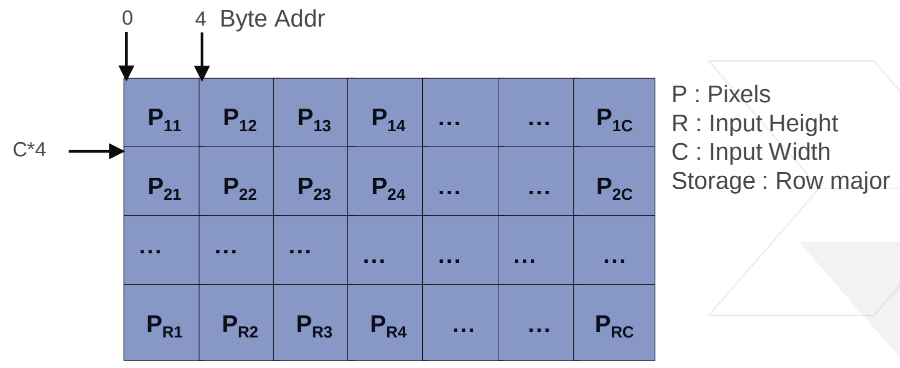


Instance Segmentation

Segmentation Output Data

> Segmentation Networks Ouptut

- For these kind of networks, the output layer is typically Crop and the size of the output layer will be same as the input size of the network.
- The output will be written sequentially in raster scan order inside output buffers.
- Each pixel represent the class ID/Label.


0
4 Byte Addr
C*4
P_{11} P_{12} P_{13} P_{14} … … P_{1C}
P_{21} P_{22} P_{23} P_{24} … … P_{2C}
… … … … … … …
P_{R1} P_{R2} P_{R3} P_{R4} … … P_{RC}
P : Pixels
R : Input Height
C : Input Width
Storage : Row major

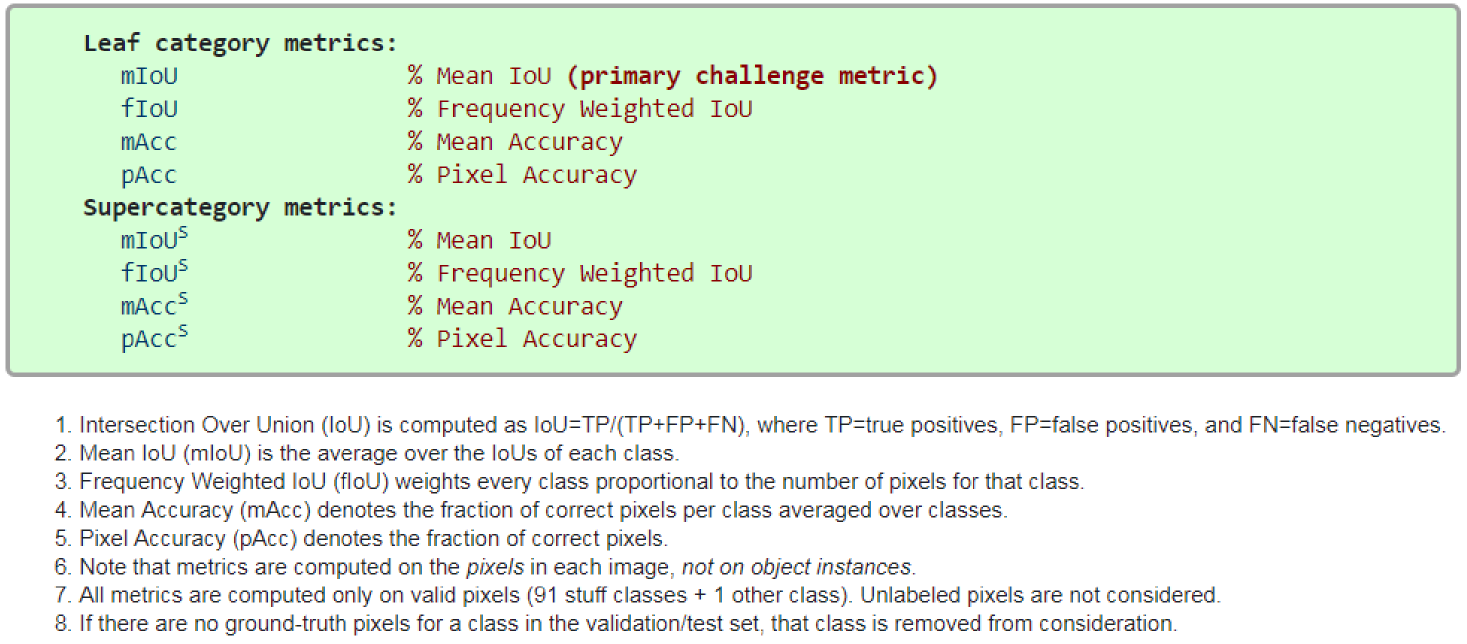
Output buffer

- Example:
  For AlexNet-FCN, the last layer is Crop and the output size is same as the input size i.e. 500 x 500.

Metrics

> **In general mean IoU is the metric of choice for segmentation**

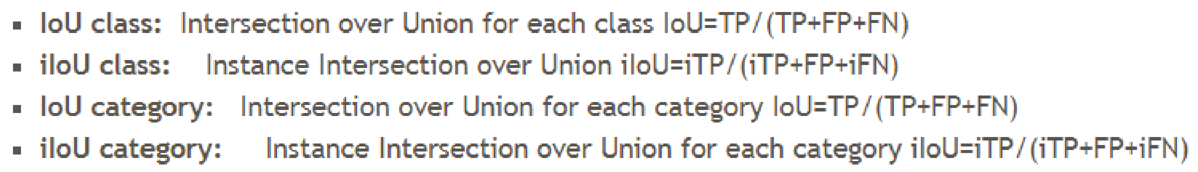
>> MS COCO uses the following metrics:



```
Leaf category metrics:
   mIoU                    % Mean IoU (primary challenge metric)
   fIoU                    % Frequency Weighted IoU
   mAcc                    % Mean Accuracy
   pAcc                    % Pixel Accuracy
Supercategory metrics:
   mIoU^S                  % Mean IoU
   fIoU^S                  % Frequency Weighted IoU
   mAcc^S                  % Mean Accuracy
   pAcc^S                  % Pixel Accuracy
```

1. Intersection Over Union (IoU) is computed as IoU=TP/(TP+FP+FN), where TP=true positives, FP=false positives, and FN=false negatives.
2. Mean IoU (mIoU) is the average over the IoUs of each class.
3. Frequency Weighted IoU (fIoU) weights every class proportional to the number of pixels for that class.
4. Mean Accuracy (mAcc) denotes the fraction of correct pixels per class averaged over classes.
5. Pixel Accuracy (pAcc) denotes the fraction of correct pixels.
6. Note that metrics are computed on the *pixels* in each image, *not on object instances*.
7. All metrics are computed only on valid pixels (91 stuff classes + 1 other class). Unlabeled pixels are not considered.
8. If there are no ground-truth pixels for a class in the validation/test set, that class is removed from consideration.

>> CityScapes and Kitti Datasets use the following metrics:



- **IoU class:** Intersection over Union for each class IoU=TP/(TP+FP+FN)
- **iIoU class:** Instance Intersection over Union iIoU=iTP/(iTP+FP+iFN)
- **IoU category:** Intersection over Union for each category IoU=TP/(TP+FP+FN)
- **iIoU category:** Instance Intersection over Union for each category iIoU=iTP/(iTP+FP+iFN)

Semantic Segmentation General Approaches

> **Fully Convolutional Networks (FCN)**
>> Uses deconvolutions to expand back up to original image size
>> Later versions also add skip connections to maintain some of input resolution

> **SegNet**
>> Similar to FCN but instead of short cut connections for features, indices from maxpooling are copied (makes it more memory efficient than FCN)

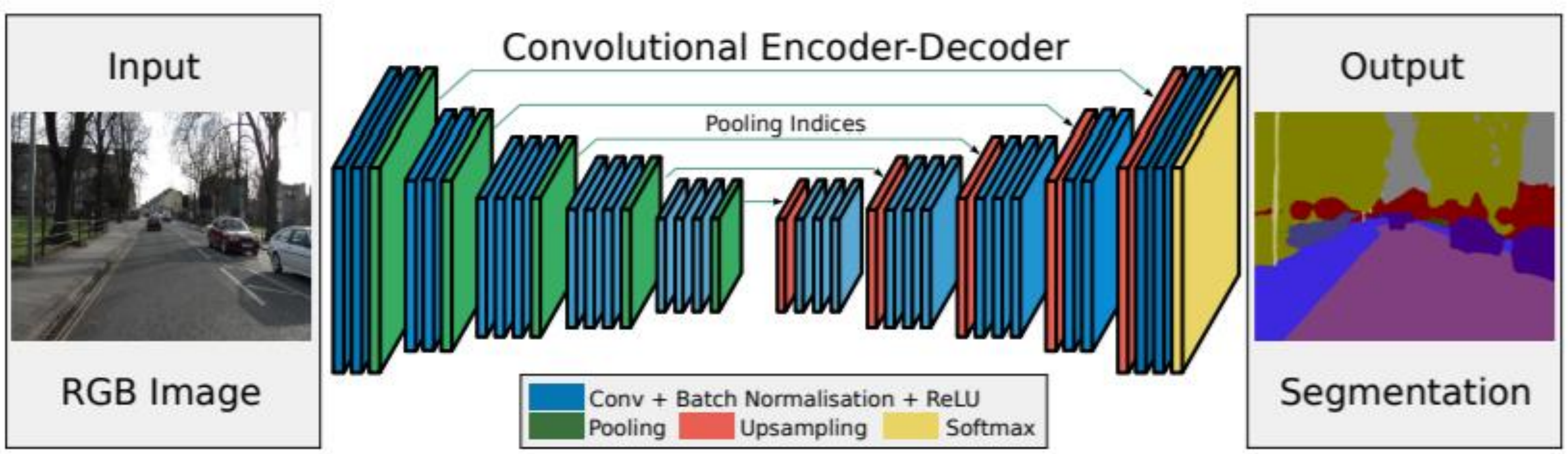
> **Dilated Convolutions**
>> Intended to increase resolution/FoV without increasing number of weights

> **DeepLab (One variant holds the leaderboard on PASCAL VOC)**
>> Starts with Dilated convolutions and adds conditional receptive fields
>> Also incorporates multiscale (image pyramid approach)

> **FPN (ResNeXt152-FPN was the winning entry for MS COCO Stuff - 2017)**
>> Creates a pyramid of feature maps during feature extraction at ½ scale for each layer, then upsamples back to original image size and provides skip connections from input maps to output

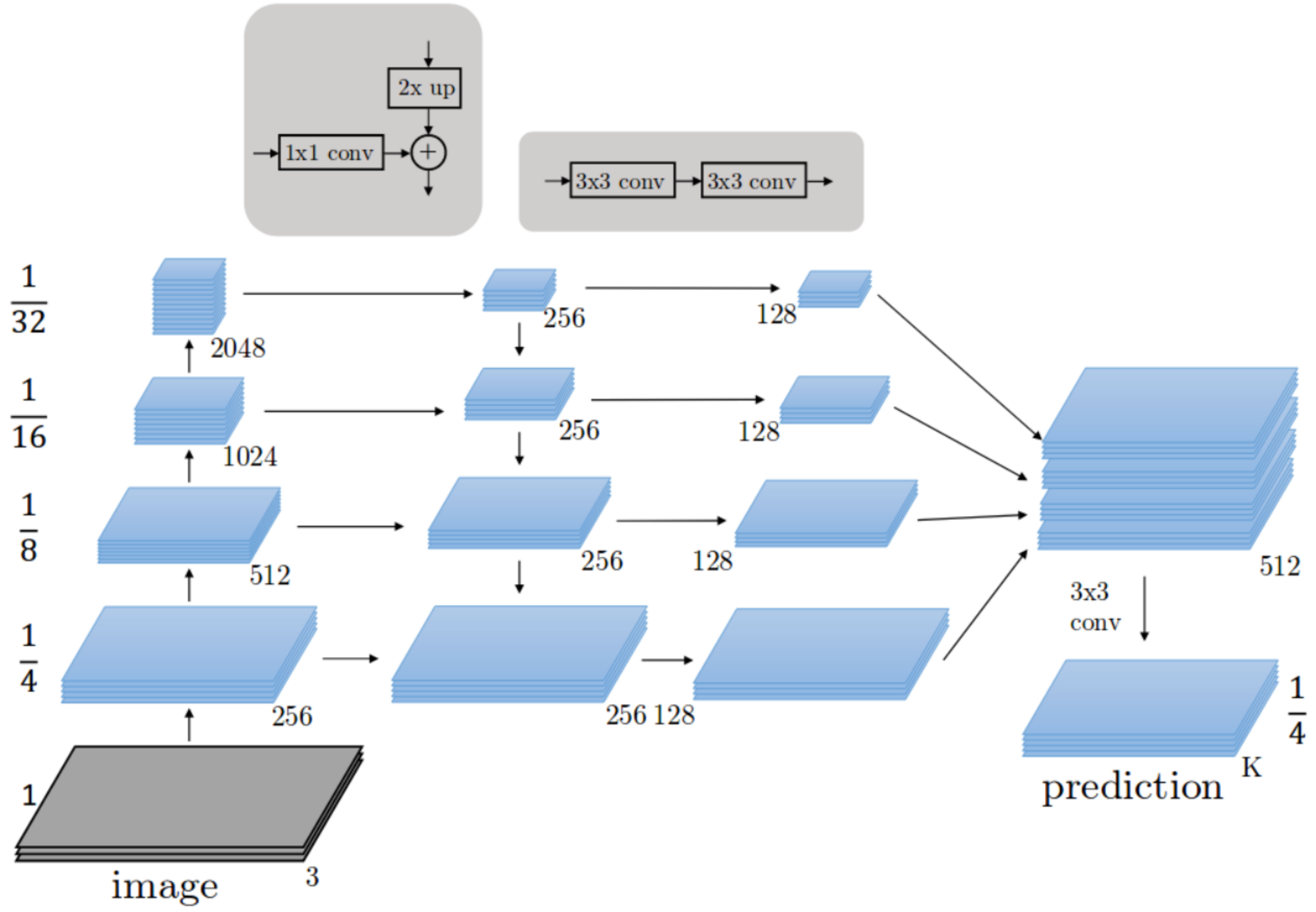
Source: http://blog.qure.ai/notes/semantic-segmentation-deep-learning-review#segnet

Segnet (Encoder/Decoder) Approach

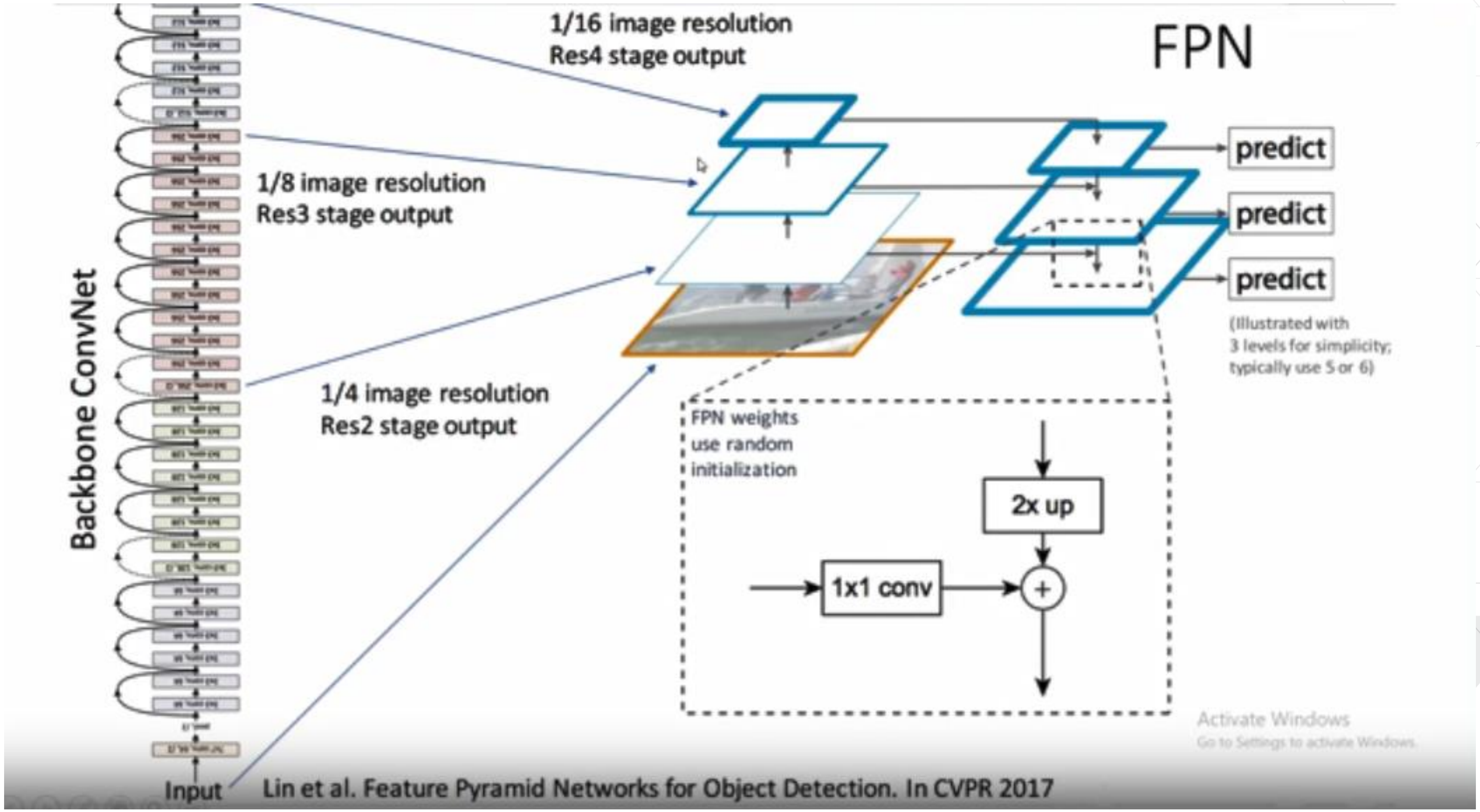


- There are different flavors of Segnet
- First 13 convolutional layers correspond to VGG16 (encoder stage)
- Decode stage consists of 13 layers which correspond to the encode layers
  - For pass through paths, convolution is performed with max pooling indices and trainable decoder filters (instead of deconvolution)

Feature Pyramid Networks



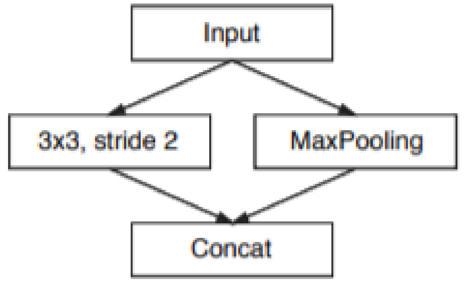
FPN Example with Resnet backbone



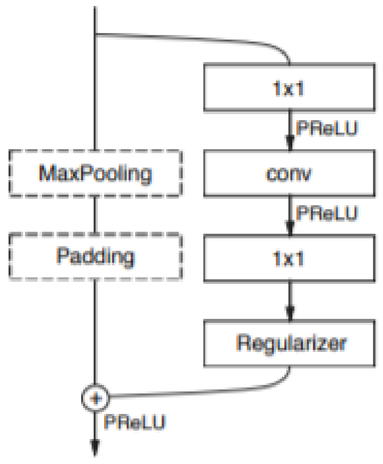
Lin et al. Feature Pyramid Networks for Object Detection. In CVPR 2017

ENet Overview

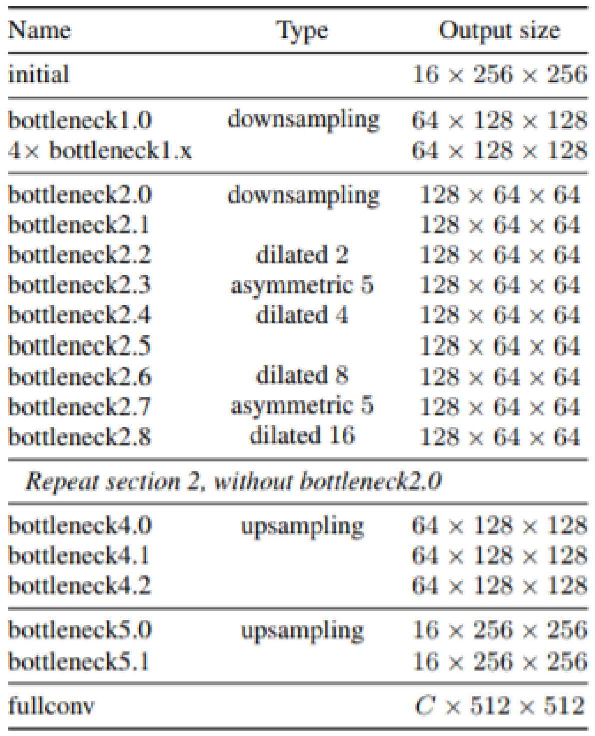
> **ENet builds on the encoder/decoder network approach similar to Segnet**
>> In general, it implements figure a for the initial stage, then figure b for the bottleneck stages as are shown in the table
>> The latter stages of ENet (decoder stage) expand back up to the original image size
– Pooling is also used to forward layers from the encoder through to the decoder stage to help maintain spatial coherence



(a)



(b)



Name	Type	Output size
initial		16 × 256 × 256
bottleneck1.0	downsampling	64 × 128 × 128
4× bottleneck1.x		64 × 128 × 128
bottleneck2.0	downsampling	128 × 64 × 64
bottleneck2.1		128 × 64 × 64
bottleneck2.2	dilated 2	128 × 64 × 64
bottleneck2.3	asymmetric 5	128 × 64 × 64
bottleneck2.4	dilated 4	128 × 64 × 64
bottleneck2.5		128 × 64 × 64
bottleneck2.6	dilated 8	128 × 64 × 64
bottleneck2.7	asymmetric 5	128 × 64 × 64
bottleneck2.8	dilated 16	128 × 64 × 64
Repeat section 2, without bottleneck2.0		
bottleneck4.0	upsampling	64 × 128 × 128
bottleneck4.1		64 × 128 × 128
bottleneck4.2		64 × 128 × 128
bottleneck5.0	upsampling	16 × 256 × 256
bottleneck5.1		16 × 256 × 256
fullconv		C × 512 × 512

Reference: https://arxiv.org/pdf/1606.02147.pdf

ESPNet Overview (1)

- ESPNet (Efficient Spatial Pyramid Network) has several flavors
  - Types a-c produce reduced size feature map outputs and type d produces a segmentation map that is the same size as the input image
  - The flavors implemented in the following tutorial are ESPNet-C (c) and the full ESPNet (d)
    - These two types are a sort of feature pyramid type of network
    - Pass through concepts of Segnet are also applied in types b, c, and d

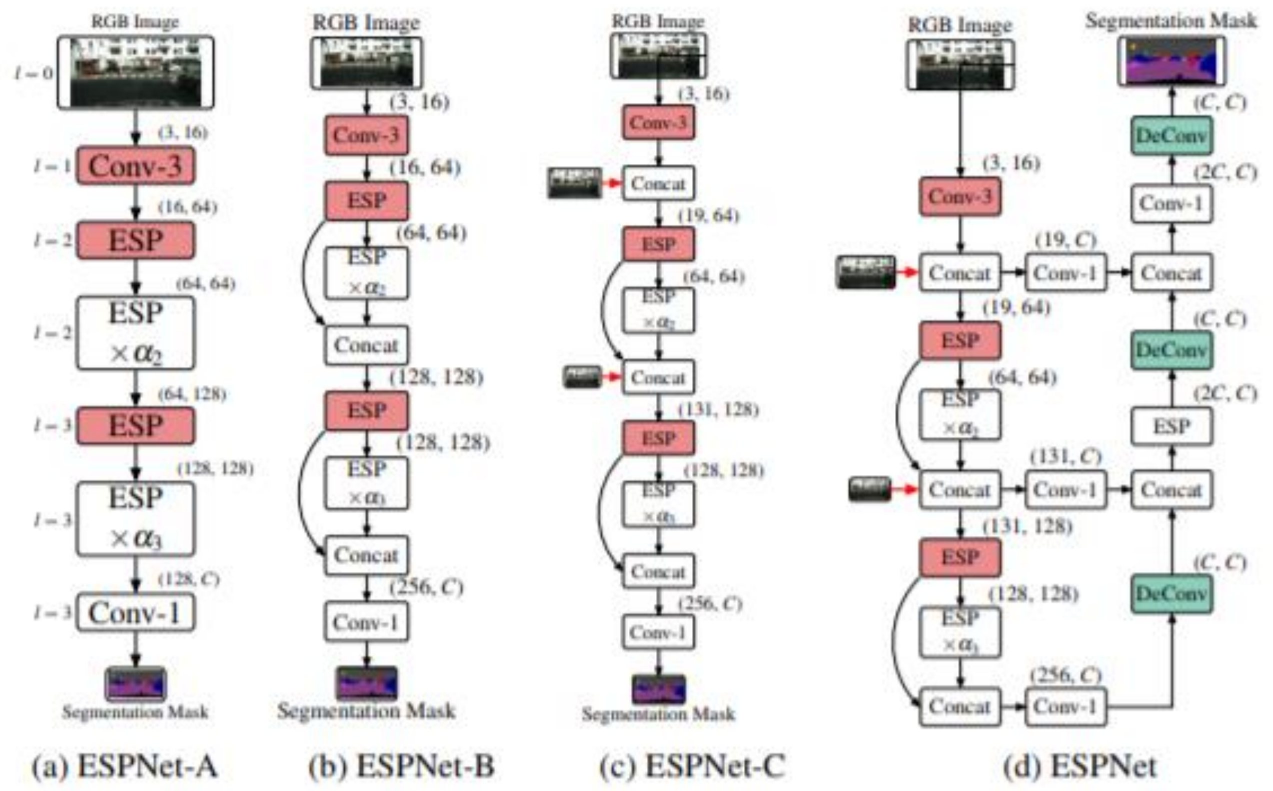
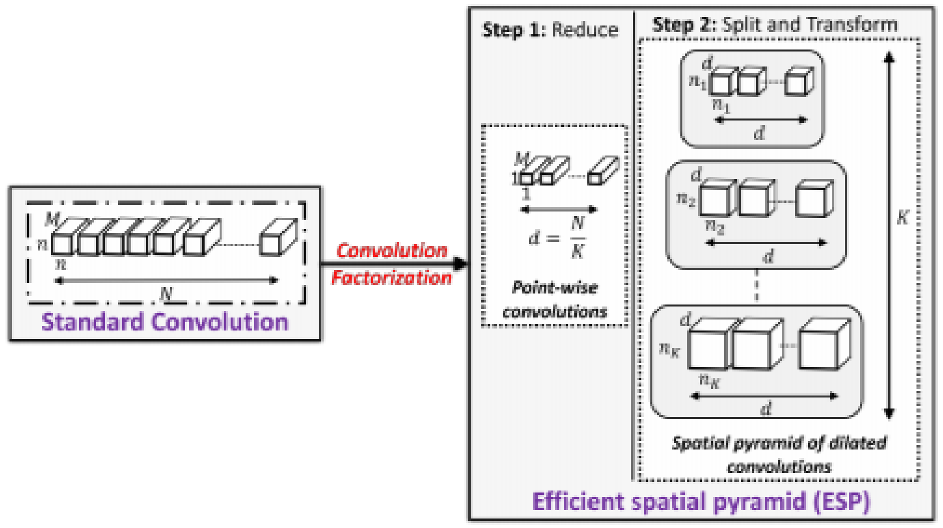


Fig. 4: The path from ESPNet-A to ESPNet. Red and green color boxes represent the modules responsible for down-sampling and up-sampling operations, respectively. Spatial-level l is indicated on the left of every module in (a). We denote each module as (# input channels, # output channels). Here, Conv-n represents $n \times n$ convolution.

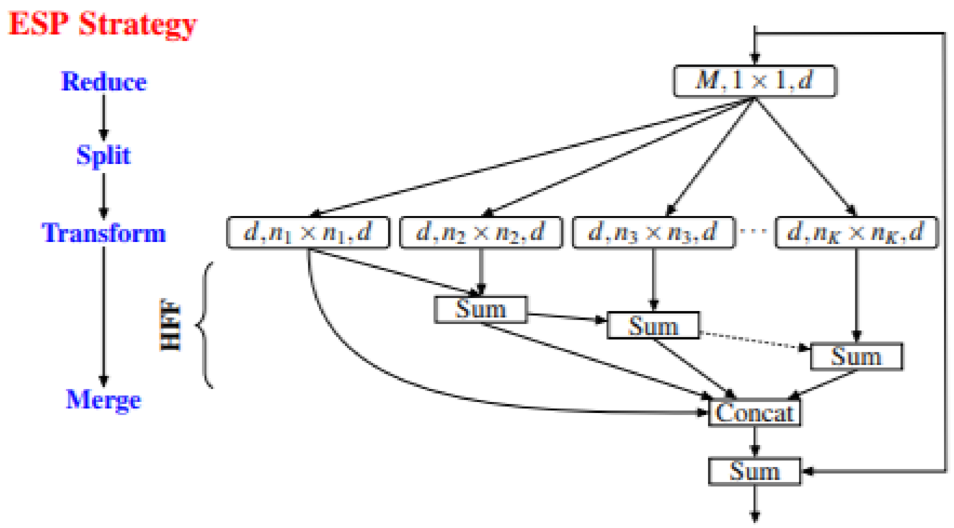
ESPNet Overview (2)

- **One of the novel features of ESPNet are ESP modules which are a factorized form of convolutions**
  - Factorization is a way to optimize the convolution operation by decomposing it into a spatial pyramid of point-wise and dilated convolutions
  - This reduces the computational complexity of the convolution operation



Step 1: Reduce
Step 2: Split and Transform
Standard Convolution
Convolution Factorization
Point-wise convolutions
Spatial pyramid of dilated convolutions
Efficient spatial pyramid (ESP)

(a)



(b)

Reference: https://arxiv.org/pdf/1803.06815.pdf

Adaptable.
Intelligent.

XILINX®