18CN716_Deep Learning for Visual Recognition.

ASSIGNMENT-5 –CNN ARCHITECTURES FOR UERC DATASET

CEN-Biometrics

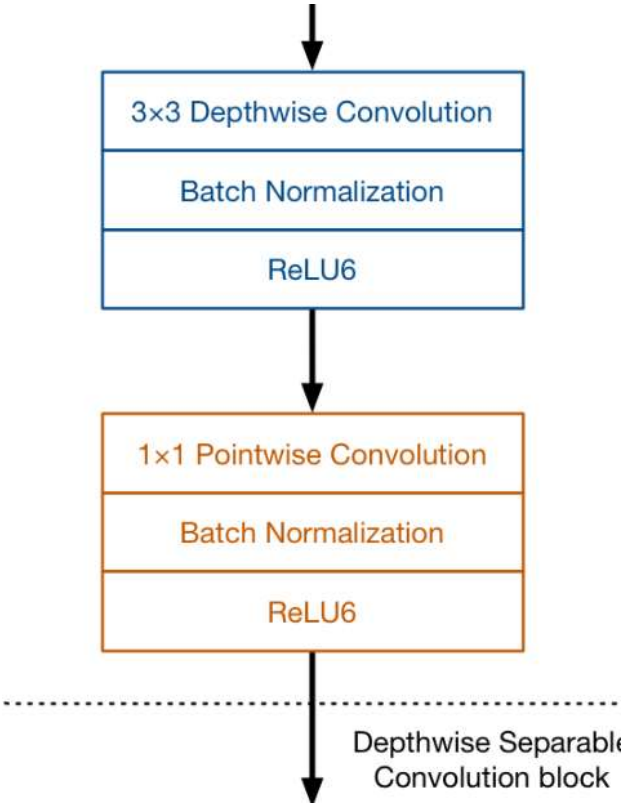
Simran CB.EN.P2CEN18016

Kiran CB.EN.P2CEN18006

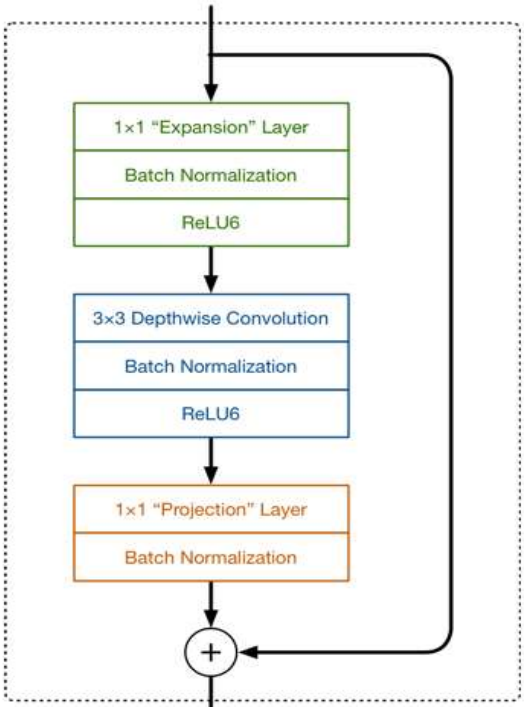
Harikrishnan CB CB.EN.P2CEN18004

Architecture details

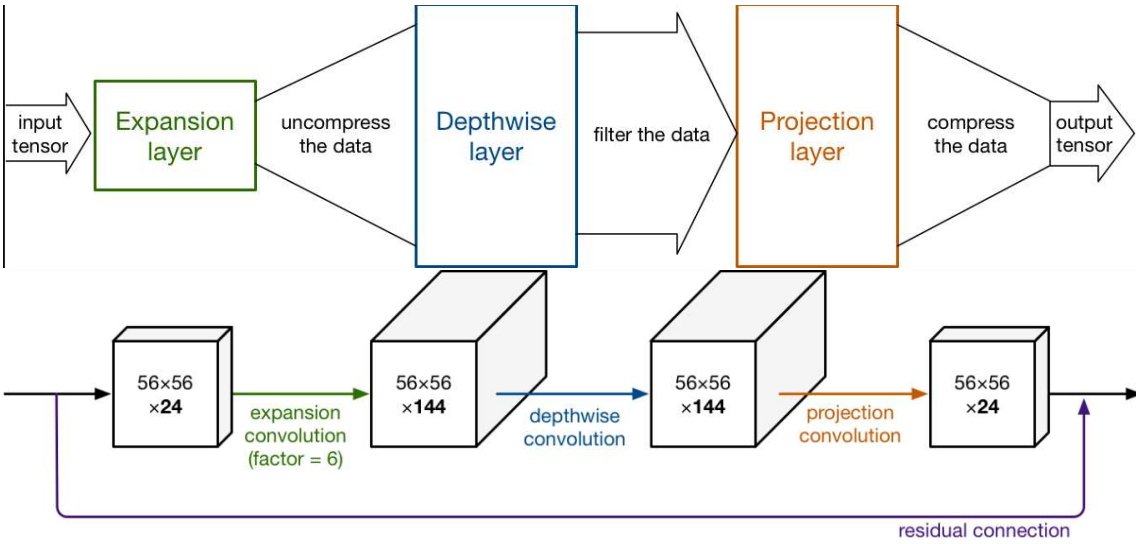
Mobilenet is a lightweight architecture developed for mobile and web applications.



The above architecture is that of Mobilenet. It uses depthwise seperable convolution to decrease the number of learnable parameters which decreases computation.



The residual connection which is connected across the layers preserves the gradient and it eliminates the problem of vanishing gradient. This 1x1, 3x3 and 1x1 layers together constitute one bottleneck.



The pointwise expansion convolution (t) is multiplied with the depth of the input to get the number of filters to be used. Pointwise projection convolution will use c value to compute the number of filters to be used in the convolution which will determine the depth of the output.

Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

The MobileNetV2 architecture

This is the MobilenetV2 architecture. There are 17 bottleneck blocks. Total number of learnable parameters are 3.54 million. Filter size used for depthwise convolution is 3x3.

'S'-stride: when 's' is one the size remains the same,when 's' is 2 the size decreases.

't'-expansion factor: This expansion factor is multiplied with the input image depth to increase the depth of channel and are concatenated one behind other.Used by 1 *1 expansion

'c'-no of filters: The number of filters used by 1*1 projection convolution layer

'n'-no of block: The number of bottleneck blocks which are repeated.

ReLu-6:

First, we cap the units at 6, so our ReLU activation function is y = min(max(x, 0), 6).This encourages the model to learn sparse features earlier. We will refer to ReLU units capped at n as ReLU-n units.

Batch Normalization

Batch normalisation is a technique for improving the performance and stability of neural networks, and also makes more sophisticated deep learning architectures work in practice (like DCGANs). It is a method we can use to normalize the inputs of each layer, in order to fight the internal covariate shift problem. Batch normalization reduces the amount by what the hidden unit values shift around (covariance shift). It also allows each layer of a network to learn by itself a little bit more independently of other layers.

To increase the stability of a neural network, batch normalization normalizes the output of a previous activation layer by subtracting the batch mean and dividing by the batch standard deviation. The four parameters used in batch normalization are mean, standard deviation, normalization and scaling.

Input	224x224x3	0	
Zero padding	225x225x3	0	
Conv2d	112x112x32	864	3x3x3x32
Batch normalization	112x112x32	128	32x4
Relu(Activation function)	112x112x32	0	
Expanded conv (depthwise)	112x112x32	288	3x3x32
BN	112x112x32	128	32x4
Relu	112x112x32	0	
Conv project	112x112x16	512	16x32
BN	112x112x16	64	16x4
Block 1			
Expand (Conv2d)	112x112x96	1536	16x96
BN	112x112x96	384	96x4
Relu	112x112x96	0	
Zero padding	113x113x96	0	
Depthwise	56x56x96	864	3x3x96
Depthwise BN	56x56x96	384	96x4
Relu	56x56x96	0	
Project	56x56x24	2304	24x96
Project BN	56x56x24	96	24x4
Block 2			
Conv2d	56x56x144	3456	24x144
BN	56x56x144	576	144x4
Relu	56x56x144	0	
Depthwise convolution	56x56x144	1296	3x3x144
Batch normalization	56x56x144	576	144x4

Relu	56x56x144	0	0
Conv2d	56x56x24	3456	24x144
Batch normalization	56x56x24	96	24x4
Add	56x56x24		
Block 3			
Expand (conv2d)	56x56x144	3456	24x144
BN	56x56x144	576	144x4
Relu	56x56x144	0	
Zero padding	57x57x144	0	
Depthwise	28x28x144	1296	3x3x144
Depthwise bn	28x28x144	576	144x4
Relu	28x28x144	0	
Project	28x28x32	4608	32x144
Project bn	28x28x32	128	32x4
Block 4			
Expand (conv2d)	28x28x192	6144	192x32
BN	28x28x192	768	192x4
Relu	28x28x192	0	
Depthwise	28x28x192	1728	3x3x192
Depthwise bn	28x28x192	768	192x4
Relu	28x28x192	0	
Project	28x28x32	6144	32x192
Project bn	28x28x32	128	32x4
Add	28x28x32	0	
Block 5			
Expand	28x28x192	6144	192x32
BN	28x28x192	768	192x4
Relu	28x28x192	0	
Depthwise	28x28x192	1728	3x3x192
Bn	28x28x192	768	192x4
Relu	28x28x192	0	
Project	28x28x32	6144	32x192
Project bn	28x28x32	128	32x4
Add	28x28x32	0	
Block 6			
Expand	28x28x192	6144	192x32
BN	28x28x192	768	192x4
Relu	28x28x192	0	
Zero padding	28x28x192	0	
Depthwise	14x14x192	1728	3x3x192
Bn	14x14x192	768	192x4
Relu	14x14x192	768	192x4
Project	14x14x64	12288	64x192
Project bn	14x14x64	256	64x4
Block 7			
Conv2d	14x14x384	24576	64x384

BN	14x14x384	1536	384x4
Relu	14x14x384	0	
Depthwise	14x14x384	3456	3x3x384
BN	14x14x384	1536	384x4
Relu	14x14x384	0	
Conv	14x14x64	24576	384x64
Bn	14x14x64	256	64x4
Add	14x14x64	0	
Block 8			
Conv2d	14x14x384	24576	64x384
BN	14x14x384	1536	384x4
Relu	14x14x384	0	
Depthwise	14x14x384	3456	3x3x384
BN	14x14x384	1536	384x4
Relu	14x14x384	0	
Conv	14x14x64	24576	384x64
Bn	14x14x64	256	64x4
Add	14x14x64	0	
Block 9			
Conv	14x14x384	24576	64x384
BN	14x14x384	1536	384x4
Relu	14x14x384	0	
Depthwise	14x14x384	3456	3x3x384
BN	14x14x384	1536	384x4
Relu	14x14x384	0	
Conv2d	14x14x64	24576	384x64
Bn	14x14x64	256	64x4
Add	14x14x64	0	
Block 10			
Conv	14x14x384	24576	64x384
BN	14x14x384	1536	384x4
Relu	14x14x384	0	
Depthwise	14x14x384	3456	3x3x384
BN	14x14x384	1536	384x4
Relu	14x14x384	0	
Conv2d	14x14x96	36864	384x96
Bn	14x14x96	384	96x4
Block 11			
Conv	14x14x576	55296	96x576
BN	14x14x576	2304	576x4
Relu	14x14x576	0	
Depthwise	14x14x576	5184	3x3x576
BN	14x14x576	2304	576x4
Relu	14x14x576	0	
Conv	14x14x96	55296	576x96
Bn	14x14x96	384	96x4
Add	14x14x96	0	

Block 12			
Conv	14x14x576	55296	96x576
BN	14x14x576	2304	576x4
Relu	14x14x576	0	
Depthwise	14x14x576	5184	3x3x576
BN	14x14x576	2304	576x4
Relu	14x14x576	0	
Conv2d	14x14x96	55296	576x96
Bn	14x14x96	384	96x4
Add	14x14x96	0	
Block 13			
Conv2d	14x14x576	55296	96x576
BN	14x14x576	2304	576x4
Relu	14x14x576	0	
Zero padding	15x15x576	0	
Depthwise conv	7x7x576	5184	3x3x576
BN	7x7x576	2304	576x4
Relu	7x7x576	0	
Conv2d	7x7x160	92160	576x160
Bn	7x7x160	690	160x4
Block 14			
Expand (Conv2d)	7x7x960	153600	160x960
BN	7x7x960	3840	960x4
Relu	7x7x960	0	
Depthwise	7x7x960	8640	3x3x960
BN	7x7x960	3840	960x4
Relu	7x7x960	0	
Project	7x7x160	153600	160x960
BN	7x7x160	640	160x4
Add	7x7x160	0	
Block 15			
Expand (Conv2d)	7x7x960	153600	160x960
BN	7x7x960	3840	960x4
Relu	7x7x960	0	
Depthwise	7x7x960	8640	3x3x960
BN	7x7x960	3840	960x4
Relu	7x7x960	0	
Project	7x7x160	153600	160x960
BN	7x7x160	640	160x4
Add	7x7x160	0	
Block 16			
Expand (Conv2d)	7x7x960	153600	160x960
BN	7x7x960	3840	960x4
Relu	7x7x960	0	
Depthwise	7x7x960	8640	3x3x960
BN	7x7x960	3840	960x4
Relu	7x7x960	0	
Project	7x7x320	307200	320x960

BN	7x7x320	1280	320x4
Conv2d	7x7x1280	409600	1280x320
BN	7x7x1280	5120	1280x4
Relu	7x7x1280	0	
Global average pooling	1280	0	
Logits (Dense)	1000	1281000	1280x1000+1000
Total	3,538,984		

All the filters used for depthwise convolution is 3x3 and pointwise convolution is 1x1.

Reference

Sandler, Mark, et al. "MobileNetV2: Inverted Residuals and Linear Bottlenecks." *arXiv preprint arXiv:1801.04381* (2018).