Deep Learning

4.3 Putting it all together - CNN

Dr. Konda Reddy Mopuri
kmopuri@iittp.ac.in
Dept. of CSE, IIT Tirupati

Architecture of a simple CNN

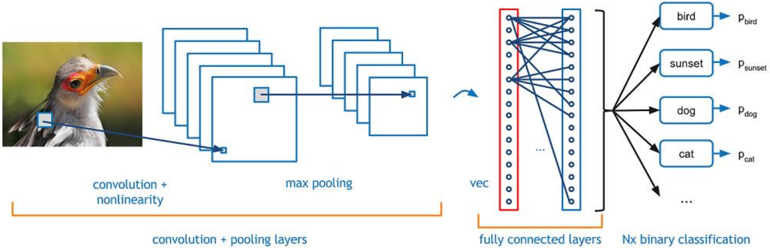
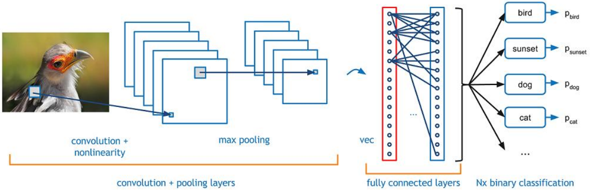


Figure credits: Adit Deshpande

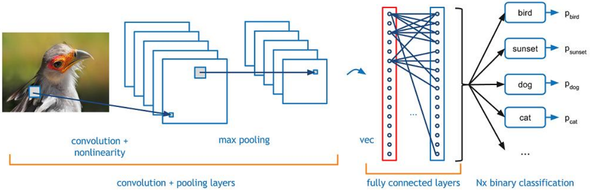
Architecture of a simple CNN



1. Initially Conv layer with nonlinearity

Figure credits: Adit Deshpande

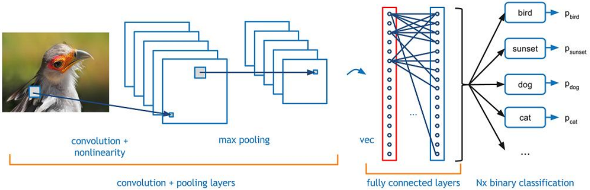
Architecture of a simple CNN



1. Initially Conv layer with nonlinearity
2. Followed by a few Conv + Nonlinearity layers

Figure credits: Adit Deshpande

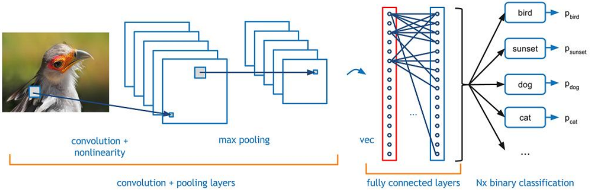
Architecture of a simple CNN



1. Initially Conv layer with nonlinearity
2. Followed by a few Conv + Nonlinearity layers
3. Have Pooling layers in between Conv layers $\rightarrow$ reduce the feature map size sufficiently

Figure credits: Adit Deshpande

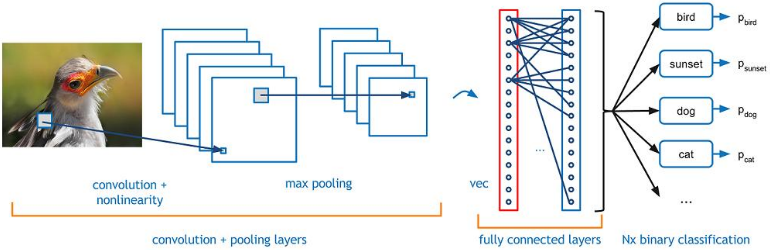
Architecture of a simple CNN



1. Initially Conv layer with nonlinearity
2. Followed by a few Conv + Nonlinearity layers
3. Have Pooling layers in between Conv layers → reduce the feature map size sufficiently
4. Vectorize and and fully connected layers

Figure credits: Adit Deshpande

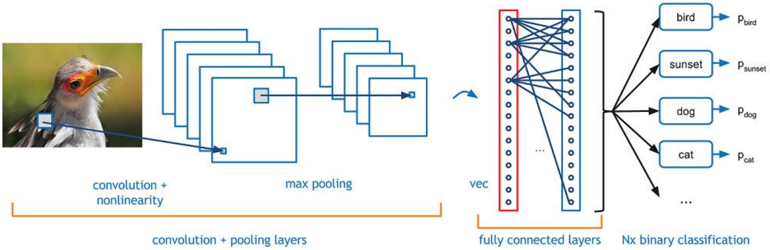
Architecture of a simple CNN



INPUT -> [[CONV -> RELU] *N -> POOL]*M -> [FC->RELU]*K -> FC

Figure credits: Adit Deshpande

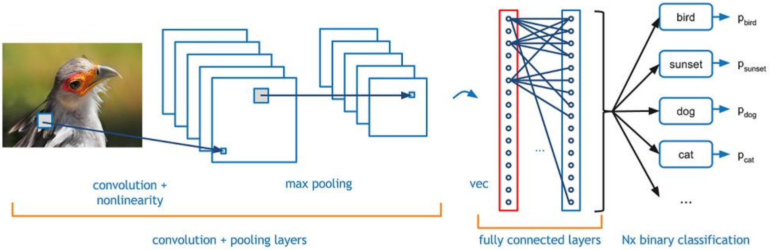
Architecture of a simple CNN



INPUT -> [[CONV -> RELU] *N -> POOL]*M -> [FC->RELU]*K -> FC

Figure credits: Adit Deshpande

Architecture of a simple CNN



INPUT -> [[CONV -> RELU] *N -> POOL]*M -> [FC->RELU]*K -> FC

Figure credits: Adit Deshpande

Case study: LeNet-like architecture

input size/ layer information	output size	# parameters	# products
$1 \times 28 \times 28$ `nn.Conv2d(1, 32, kernel_size=5)`			

Case study: LeNet-like architecture

input size/ layer information	output size	# parameters	# products
$1 \times 28 \times 28$ `nn.Conv2d(1, 32, kernel_size=5)`	$32 \times 24 \times 24$		

Case study: LeNet-like architecture

input size/ layer information	output size	# parameters	# products
$1 \times 28 \times 28$ `nn.Conv2d(1, 32, kernel_size=5)`	$32 \times 24 \times 24$	$32.(5^2 + 1) = 832$	

Case study: LeNet-like architecture

input size/ layer information	output size	# parameters	# products
$1 \times 28 \times 28$ `nn.Conv2d(1, 32, kernel_size=5)`	$32 \times 24 \times 24$	$32.(5^2 + 1) = 832$	$32.24^2.5^2 = 460800$
$32 \times 24 \times 24$ `F.max_pool2d(., kernel_size=3)`			

Case study: LeNet-like architecture

input size/ layer information	output size	# parameters	# products
$1 \times 28 \times 28$ `nn.Conv2d(1, 32, kernel_size=5)`	$32 \times 24 \times 24$	$32.(5^2 + 1) = 832$	$32.24^2.5^2 = 460800$
$32 \times 24 \times 24$ `F.max_pool2d(., kernel_size=3)`	$32 \times 8 \times 8$	0	0

Case study: LeNet-like architecture

input size/ layer information	output size	# parameters	# products
$1 \times 28 \times 28$ `nn.Conv2d(1, 32, kernel_size=5)`	$32 \times 24 \times 24$	$32.(5^2 + 1) = 832$	$32.24^2.5^2 = 460800$
$32 \times 24 \times 24$ `F.max_pool2d(., kernel_size=3)`	$32 \times 8 \times 8$	0	0
$32 \times 8 \times 8$ / `F.relu(.)`	$32 \times 8 \times 8$	0	0

Case study: LeNet-like architecture

input size/ layer information	output size	# parameters	# products
$1 \times 28 \times 28$ `nn.Conv2d(1, 32, kernel_size=5)`	$32 \times 24 \times 24$	$32.(5^2 + 1) = 832$	$32.24^2.5^2 = 460800$
$32 \times 24 \times 24$ `F.max_pool2d(., kernel_size=3)`	$32 \times 8 \times 8$	0	0
$32 \times 8 \times 8$ / `F.relu(.)`	$32 \times 8 \times 8$	0	0
$32 \times 8 \times 8$ `nn.conv2d(32, 64, kernel_size=5)`	$64 \times 4 \times 4$	$64.(32.4^2 + 1)$ $=51264$	$64.32.4^2.5^2 = 819200$

Case study: LeNet-like architecture

input size/ layer information	output size	# parameters	# products
$1 \times 28 \times 28$ `nn.Conv2d(1, 32, kernel_size=5)`	$32 \times 24 \times 24$	$32.(5^2 + 1) = 832$	$32.24^2.5^2$ $= 460800$
$32 \times 24 \times 24$ `F.max_pool2d(., kernel_size=3)`	$32 \times 8 \times 8$	0	0
$32 \times 8 \times 8$ / `F.relu(.)`	$32 \times 8 \times 8$	0	0
$32 \times 8 \times 8$ `nn.conv2d(32, 64, kernel_size=5)`	$64 \times 4 \times 4$	$64.(32.4^2 + 1)$ $=51264$	$64.32.4^2.5^2$ $= 819200$
$64 \times 4 \times 4$ `F.max_pool2d(., kernel_size=2)`	$64 \times 2 \times 2$	0	0

Case study: LeNet-like architecture

input size/ layer information	output size	# parameters	# products
$1 \times 28 \times 28$ `nn.Conv2d(1, 32, kernel_size=5)`	$32 \times 24 \times 24$	$32.(5^2 + 1) = 832$	$32.24^2.5^2 = 460800$
$32 \times 24 \times 24$ `F.max_pool2d(., kernel_size=3)`	$32 \times 8 \times 8$	0	0
$32 \times 8 \times 8$ / `F.relu(.)`	$32 \times 8 \times 8$	0	0
$32 \times 8 \times 8$ `nn.conv2d(32, 64, kernel_size=5)`	$64 \times 4 \times 4$	$64.(32.4^2 + 1) = 51264$	$64.32.4^2.5^2 = 819200$
$64 \times 4 \times 4$ `F.max_pool2d(., kernel_size=2)`	$64 \times 2 \times 2$	0	0
$64 \times 2 \times 2$ / `F.relu(.)`	$64 \times 2 \times 2$	0	0

Case study: LeNet-like architecture

input size/ layer information	output size	# parameters	# products
$1 \times 28 \times 28$ `nn.Conv2d(1, 32, kernel_size=5)`	$32 \times 24 \times 24$	$32.(5^2 + 1) = 832$	$32.24^2.5^2 = 460800$
$32 \times 24 \times 24$ `F.max_pool2d(., kernel_size=3)`	$32 \times 8 \times 8$	0	0
$32 \times 8 \times 8$ / `F.relu(.)`	$32 \times 8 \times 8$	0	0
$32 \times 8 \times 8$ `nn.conv2d(32, 64, kernel_size=5)`	$64 \times 4 \times 4$	$64.(32.4^2 + 1) = 51264$	$64.32.4^2.5^2 = 819200$
$64 \times 4 \times 4$ `F.max_pool2d(., kernel_size=2)`	$64 \times 2 \times 2$	0	0
$64 \times 2 \times 2$ / `F.relu(.)`	$64 \times 2 \times 2$	0	0
$64 \times 2 \times 2$ `x.view(-1,256)`	256	0	0

Case study: LeNet-like architecture

input size/ layer information	output size	# parameters	# products
$1 \times 28 \times 28$ `nn.Conv2d(1, 32, kernel_size=5)`	$32 \times 24 \times 24$	$32.(5^2 + 1) = 832$	$32.24^2.5^2$ $= 460800$
$32 \times 24 \times 24$ `F.max_pool2d(., kernel_size=3)`	$32 \times 8 \times 8$	0	0
$32 \times 8 \times 8$ / `F.relu(.)`	$32 \times 8 \times 8$	0	0
$32 \times 8 \times 8$ `nn.conv2d(32, 64, kernel_size=5)`	$64 \times 4 \times 4$	$64.(32.4^2 + 1)$ $=51264$	$64.32.4^2.5^2$ $= 819200$
$64 \times 4 \times 4$ `F.max_pool2d(., kernel_size=2)`	$64 \times 2 \times 2$	0	0
$64 \times 2 \times 2$ / `F.relu(.)`	$64 \times 2 \times 2$	0	0
$64 \times 2 \times 2$ `x.view(-1,256)`	256	0	0
256 `nn.Linear(256,200)`	0 200	0 200(256+1)=51400	0 200.256=51200

Case study: LeNet-like architecture

input size/ layer information	output size	# parameters	# products
$1 \times 28 \times 28$ `nn.Conv2d(1, 32, kernel_size=5)`	$32 \times 24 \times 24$	$32.(5^2+1) = 832$	$32.24^2.5^2$ $= 460800$
$32 \times 24 \times 24$ `F.max_pool2d(., kernel_size=3)`	$32 \times 8 \times 8$	0	0
$32 \times 8 \times 8$ / `F.relu(.)`	$32 \times 8 \times 8$	0	0
$32 \times 8 \times 8$ `nn.conv2d(32, 64, kernel_size=5)`	$64 \times 4 \times 4$	$64.(32.4^2+1)$ $=51264$	$64.32.4^2.5^2$ $= 819200$
$64 \times 4 \times 4$ `F.max_pool2d(., kernel_size=2)`	$64 \times 2 \times 2$	0	0
$64 \times 2 \times 2$ / `F.relu(.)`	$64 \times 2 \times 2$	0	0
$64 \times 2 \times 2$ `x.view(-1,256)`	256	0	0
256 `nn.Linear(256,200)`	0 200	0 200(256+1)=51400	0 200.256=51200
200 / `F.relu(.)`	200	0	0

Case study: LeNet-like architecture

input size/ layer information	output size	# parameters	# products
$1 \times 28 \times 28$ `nn.Conv2d(1, 32, kernel_size=5)`	$32 \times 24 \times 24$	$32.(5^2 + 1) = 832$	$32.24^2.5^2$ $= 460800$
$32 \times 24 \times 24$ `F.max_pool2d(., kernel_size=3)`	$32 \times 8 \times 8$	0	0
$32 \times 8 \times 8$ / `F.relu(.)`	$32 \times 8 \times 8$	0	0
$32 \times 8 \times 8$ `nn.conv2d(32, 64, kernel_size=5)`	$64 \times 4 \times 4$	$64.(32.5^2 + 1)$ $=51264$	$64.32.4^2.5^2$ $= 819200$
$64 \times 4 \times 4$ `F.max_pool2d(., kernel_size=2)`	$64 \times 2 \times 2$	0	0
$64 \times 2 \times 2$ / `F.relu(.)`	$64 \times 2 \times 2$	0	0
$64 \times 2 \times 2$ `x.view(-1,256)`	256	0	0
256 `nn.Linear(256,200)`	0 200	0 200(256+1)=51400	0 200.256=51200
200 / `F.relu(.)`	200	0	0
200 `nn.Linear(200,10)`	0 10	0 10(200+1)=2010	0 10.200=2000

Recent architectures are more sophisticated

1. Note that LeNet is a classical architecture and does not reflect the recent CNNs in complexity

Recent architectures are more sophisticated

1. Note that LeNet is a classical architecture and does not reflect the recent CNNs in complexity
2. Recent CNN architectures are far more sophisticated
   - More depth
   - Regularizers to handle the depth