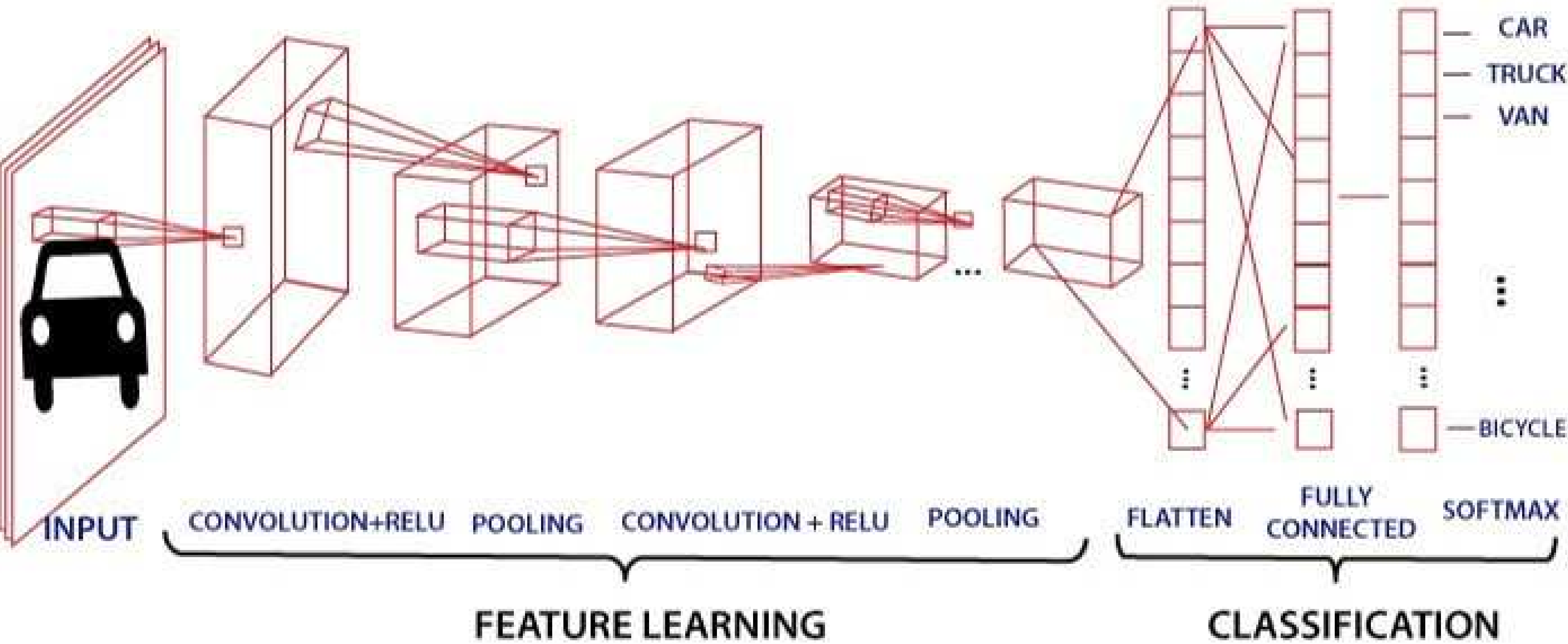
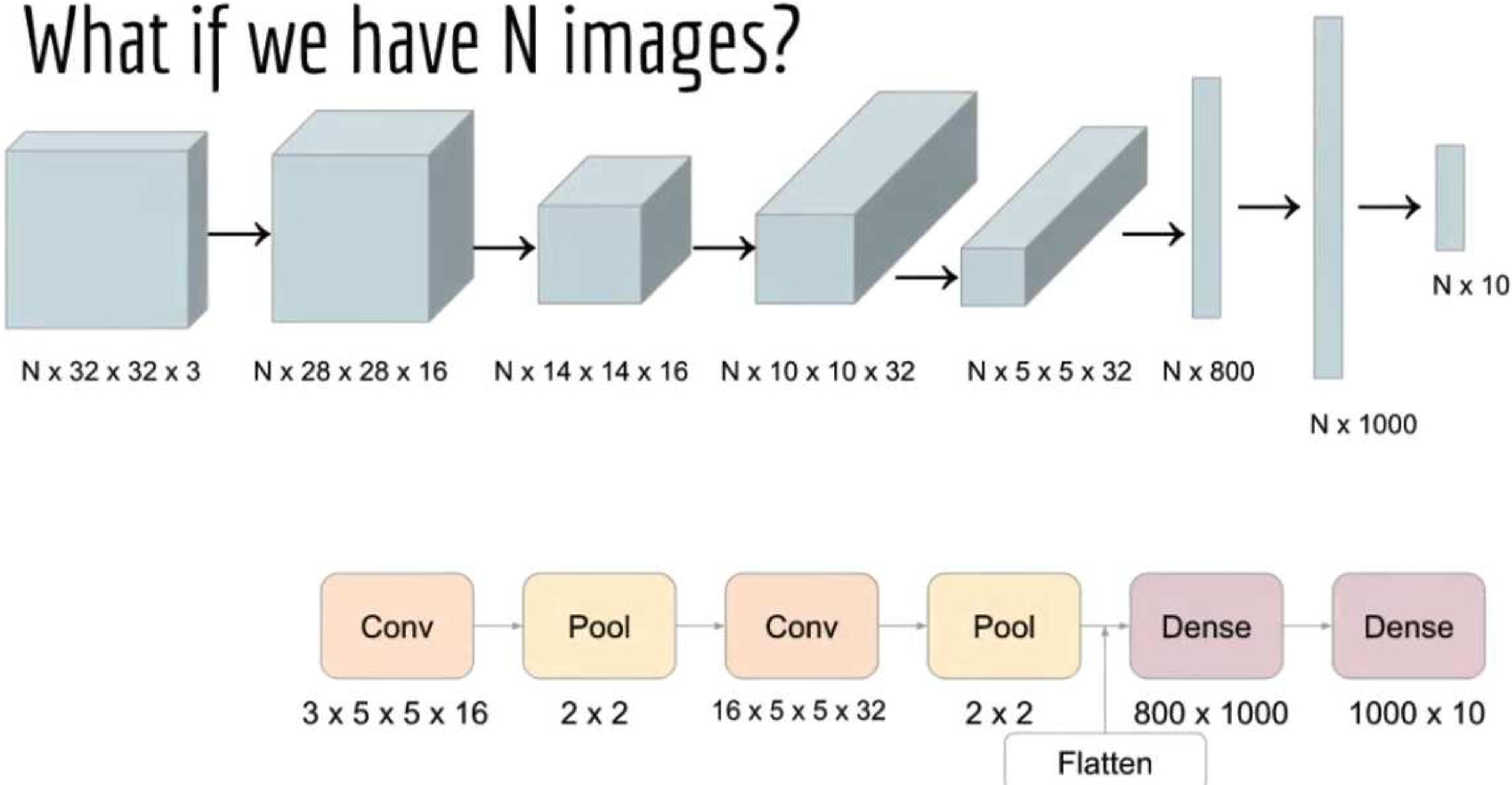
convolutional neural network

پروژه محاسبات علمی

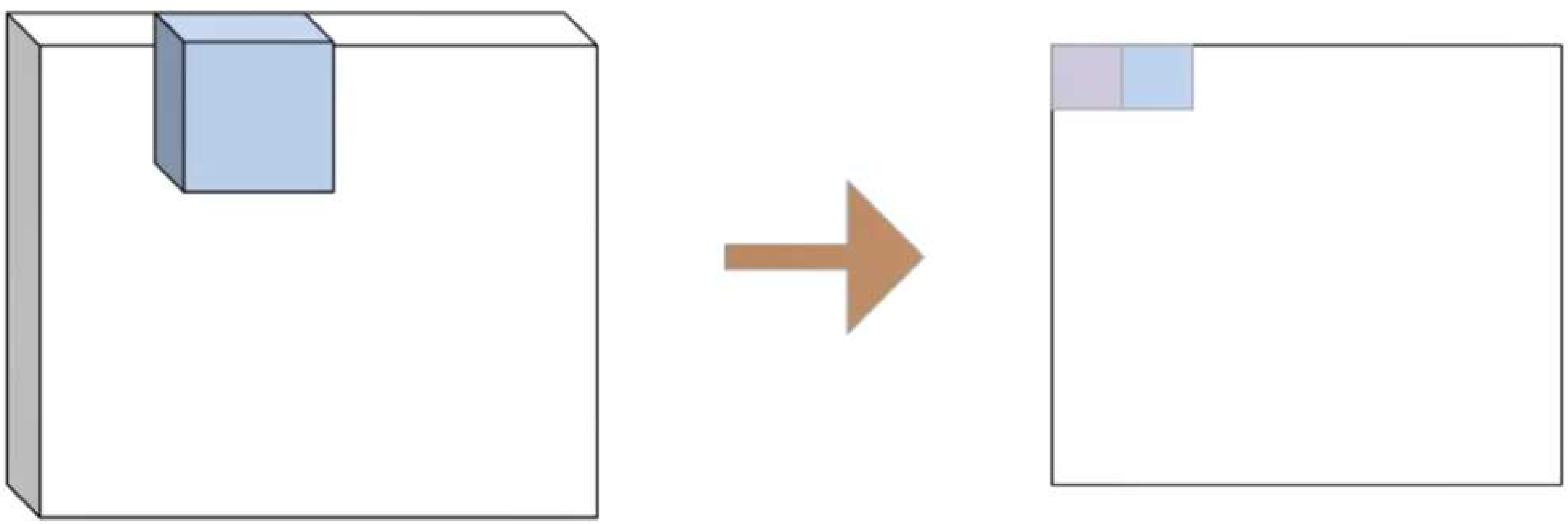
نام و نام خانوادگی : صالح زارع زاده
شماره دانشجویی : ۶۱۰۳۹۶۱۰۹


CAR
TRUCK
VAN
BICYCLE
INPUT
CONVOLUTION+RELU
POOLING
CONVOLUTION + RELU
POOLING
FLATTEN
FULLY CONNECTED
SOFTMAX
FEATURE LEARNING
CLASSIFICATION

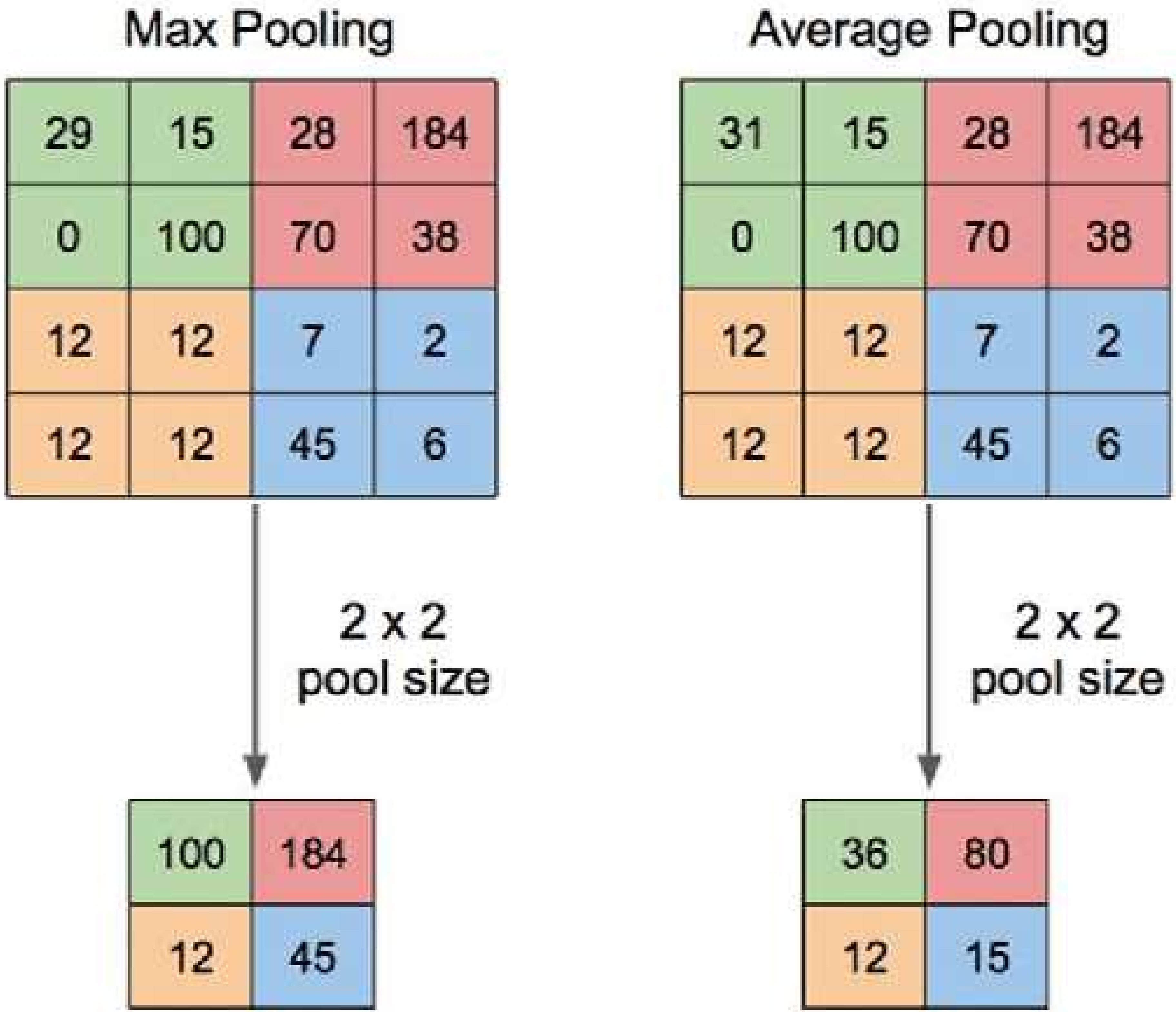
What if we have N images?



Convolution with 3-D input



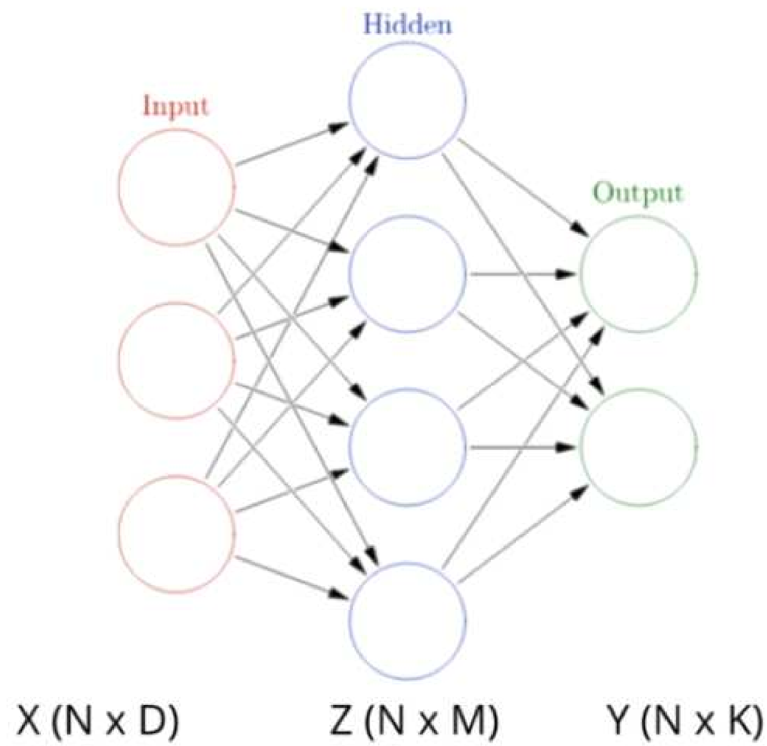
$$out[i,j] = \sum_{k_1=1}^{K} \sum_{k_2=1}^{K} \sum_{c=1}^{C} in[i-k_1, j-k_2, c]\, filter[c, k_1, k_2]$$



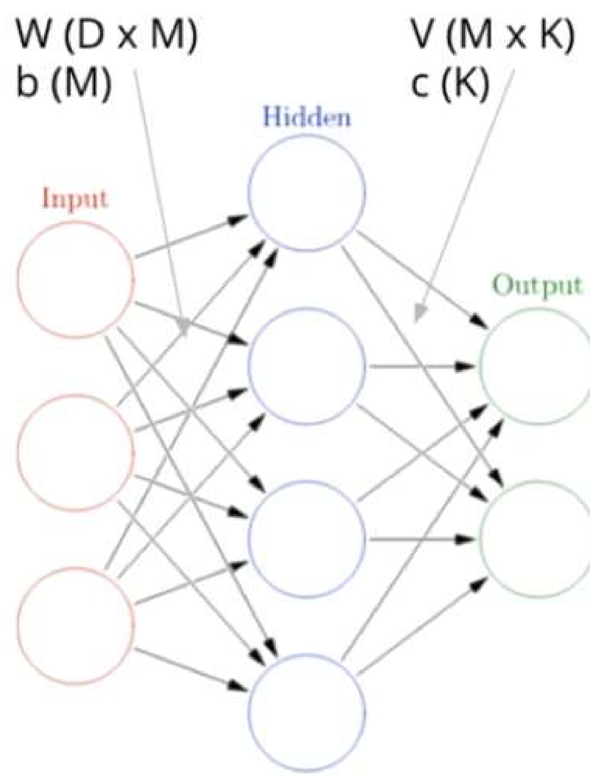
Max Pooling
29
15
28
184
0
100
70
38
12
12
7
2
12
12
45
6
2 x 2
pool size
100
184
12
45
Average Pooling
31
15
28
184
0
100
70
38
12
12
7
2
12
12
45
6
2 x 2
pool size
36
80
12
15

Backpropagation

- Let's continue our discussion on how to train a neural network



Backpropagation



Neural Network Equations

- Goal is still the same: build up our cost function, find gradients
- Gradient ascent (maximize) / Gradient descent (minimize)

Input → Hidden $$z = \sigma(W^T x + b)$$

Hidden → Output $$y = softmax(V^T z + c)$$

Output → Loss $$J = \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \log y_{nk}$$

Chain Rule

- Let's start with gradients of V and c
- As before, we split it into 3 separate derivatives using the chain rule

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \log y_{nk}$$

$$y_n = softmax(a_n)$$

$$a_{nk} = V_{:,k}^{\ T} z_n + c_k$$

$$\partial J / \partial V_{mk} = \sum_{n=1}^{N} \sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a_{nk}} \frac{\partial a_{nk}}{\partial V_{mk}}$$

$$\partial J / \partial c_k = \sum_{n=1}^{N} \sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a_{nk}} \frac{\partial a_{nk}}{\partial c_k}$$

Gradients of V and c

$$\partial J/\partial V_{mk} = \sum_{n=1}^{N} \boxed{\sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a_{nk}}} \frac{\partial a_{nk}}{\partial V_{mk}}$$

$$\partial J/\partial c_k = \sum_{n=1}^{N} \boxed{\sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a_{nk}}} \frac{\partial a_{nk}}{\partial c_k}$$

- These gradients are the ones we already found in the Logistic Regression lectures! Therefore, no need to derive them again

Gradients of V and c

- The third derivatives are easy to derive (just linear terms)

$$\partial J/\partial V_{mk} = \sum_{n=1}^{N} \sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a_{nk}} \boxed{\frac{\partial a_{nk}}{\partial V_{mk}}}$$

$$\partial J/\partial c_k = \sum_{n=1}^{N} \sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a_{nk}} \boxed{\frac{\partial a_{nk}}{\partial c_k}}$$

$$a_{nk} = V_{:,k}{}^{T} z_n + c_k \quad \Rightarrow \quad \frac{\partial a_{nk}}{\partial V_{mk}} = z_{nm}, \ \frac{\partial a_{nk}}{\partial c_k} = 1$$

Put it all together

$$\partial J / \partial V_{mk} = \sum_{n=1}^{N} (t_{nk} - y_{nk}) z_{nm}$$

$$\partial J / \partial c_k = \sum_{n=1}^{N} (t_{nk} - y_{nk})$$

Vectorize

$$\nabla_V J = Z^T(T - Y)$$

```
grad_c = np.sum(T - Y, axis=0)
```

Separate activations

$$\alpha = W^T x + b$$

$$z = \sigma(\alpha)$$

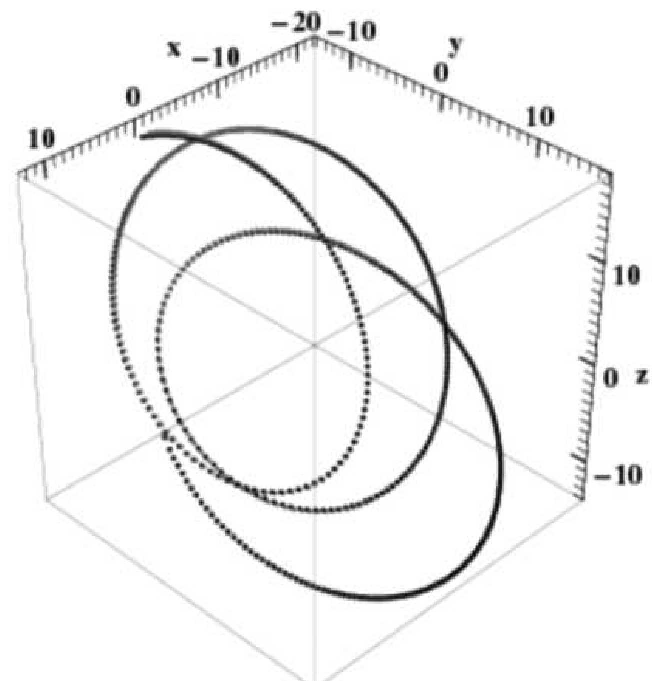
$$a = V^T z + c$$

$$y = softmax(a)$$

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \log y_{nk}$$

Law of total derivatives

- Suppose we are tracking the position of some particle in 3-D space at regular time steps t
- We have position = x(t), y(t), z(t)
- Suppose we have some function of position:
  f(x, y, z) (e.g. potential energy)



$$\frac{df}{dt} = \frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial y}\frac{dy}{dt} + \frac{\partial f}{\partial z}\frac{dz}{dt}$$

Law of total derivatives

- We can use the same rule for any number of variables parameterized by t

$$x_k(t) \; for \; k = 1 \ldots K$$

$$\frac{df}{dt} = \sum_{k=1}^{K} \frac{\partial f}{\partial x_k} \frac{dx_k}{dt}$$

Applying the law of total derivatives

- Now it makes sense: the dummy index k goes away because we sum over it, and it doesn't appear on the LHS

$$\partial J / \partial W_{dm} = \sum_{k=1}^{K} \sum_{n=1}^{N} \sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a_{nk}} \frac{\partial a_{nk}}{\partial z_{nm}} \frac{\partial z_{nm}}{\partial \alpha_{nm}} \frac{\partial \alpha_{nm}}{\partial W_{dm}}$$

$$\partial J / \partial b_{m} = \sum_{k=1}^{K} \sum_{n=1}^{N} \sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a_{nk}} \frac{\partial a_{nk}}{\partial z_{nm}} \frac{\partial z_{nm}}{\partial \alpha_{nm}} \frac{\partial \alpha_{nm}}{\partial b_{m}}$$

We solved some of these already!

$$\partial J / \partial W_{dm} = \sum_{k=1}^{K} \sum_{n=1}^{N} \boxed{\sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a_{nk}}} \frac{\partial a_{nk}}{\partial z_{nm}} \frac{\partial z_{nm}}{\partial \alpha_{nm}} \frac{\partial \alpha_{nm}}{\partial W_{dm}}$$

$$\partial J / \partial W_{dm} = \sum_{k=1}^{K} \sum_{n=1}^{N} (t_{nk} - y_{nk}) \frac{\partial a_{nk}}{\partial z_{nm}} \frac{\partial z_{nm}}{\partial \alpha_{nm}} \frac{\partial \alpha_{nm}}{\partial W_{dm}}$$

3 more derivatives

$$\partial J / \partial W_{dm} = \sum_{k=1}^{K} \sum_{n=1}^{N} \sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a_{nk}} \boxed{\frac{\partial a_{nk}}{\partial z_{nm}} \frac{\partial z_{nm}}{\partial \alpha_{nm}} \frac{\partial \alpha_{nm}}{\partial W_{dm}}}$$

$$a_{nk} = V_{:,k}{}^{T} z_n + c_k$$

$$z_{nm} = \sigma(\alpha_{nm})$$

$$\alpha_{nm} = W_{:,m}{}^{T} x_n + b_m$$

$$\frac{\partial a_{nk}}{\partial z_{nm}} = V_{mk}, \quad \frac{\partial \alpha_{nm}}{\partial W_{dm}} = x_{nd}$$

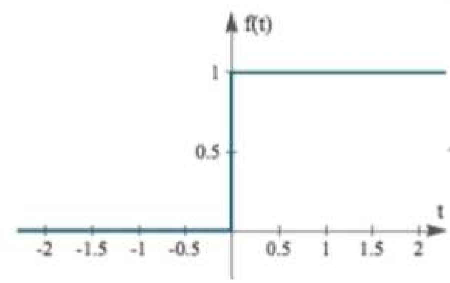
Activation function derivative

- For these lectures we'll assume sigmoid (mostly)

$$\frac{\partial z_{nm}}{\partial \alpha_{nm}} = z_{nm}(1 - z_{nm}) \quad \textit{if } \sigma(x) = \frac{1}{1+\exp(-x)}$$

$$\frac{\partial z_{nm}}{\partial \alpha_{nm}} = 1 - {z_{nm}}^2 \quad \textit{if } \sigma(x) = \tanh(x)$$

$$\frac{\partial z_{nm}}{\partial \alpha_{nm}} = u(z_{nm}) \quad \textit{if } \sigma(x) = relu(x),\ where\ u(\cdot) = step\ function$$



Put it all together

- As before, the derivative wrt bias term just replaces the input to the layer with 1

$$\partial J/\partial W_{dm} = \sum_{k=1}^{K} \sum_{n=1}^{N} (t_{nk} - y_{nk}) V_{mk} z_{nm} (1 - z_{nm}) x_{nd}$$

$$\partial J/\partial b_{m} = \sum_{k=1}^{K} \sum_{n=1}^{N} (t_{nk} - y_{nk}) V_{mk} z_{nm} (1 - z_{nm})$$

Vectorize it

- The easiest part to do first is the derivative of the activation function which is just element-wise multiplication

$$\partial J / \partial W_{dm} = \sum_{k=1}^{K} \sum_{n=1}^{N} (t_{nk} - y_{nk}) V_{mk} \boxed{z_{nm}(1 - z_{nm})} x_{nd}$$

$$Z' = Z \odot (1 - Z)$$

Consider the shapes

$$\partial J / \partial W_{dm} = \sum_{k=1}^{K} \sum_{n=1}^{N} \underbrace{(t_{nk} - y_{nk})}_{\text{N x K}} \underbrace{V_{mk}}_{\text{M x K}} z_{nm}(1 - z_{nm}) x_{nd}$$

$$(T - Y)_{N \times K} {V^T}_{K \times M} \rightarrow N \times M$$

Consider the shapes

$$\partial J / \partial W_{dm} = \sum_{k=1}^{K} \sum_{n=1}^{N} \underbrace{\boxed{(t_{nk} - y_{nk}) V_{mk}}}_{\text{N x M}} \underbrace{\boxed{z_{nm}(1 - z_{nm})}}_{\text{N x M}} x_{nd}$$

$$\left[(T - Y) V^{T} \right]_{N \times M} \odot Z'_{N \times M}$$

Consider the shapes

$$\partial J / \partial W_{dm} = \sum_{k=1}^{K} \sum_{n=1}^{N} \underbrace{(t_{nk} - y_{nk}) V_{mk} z_{nm} (1 - z_{nm})}_{\text{N x M}} \underbrace{x_{nd}}_{\text{N x D}}$$

$$\underbrace{\nabla_W J}_{\text{D x M}} = X^T \left\{ \left[(T - Y) V^T \right] \odot Z \odot (1 - Z) \right\}$$

With a generic activation function

- Sigmoid

$$\nabla_W J = X^T \left\{ \left[(T - Y) V^T \right] \odot Z \odot (1 - Z) \right\}$$

- Generic

$$\nabla_W J = X^T \left\{ \left[(T - Y) V^T \right] \odot Z' \right\}$$

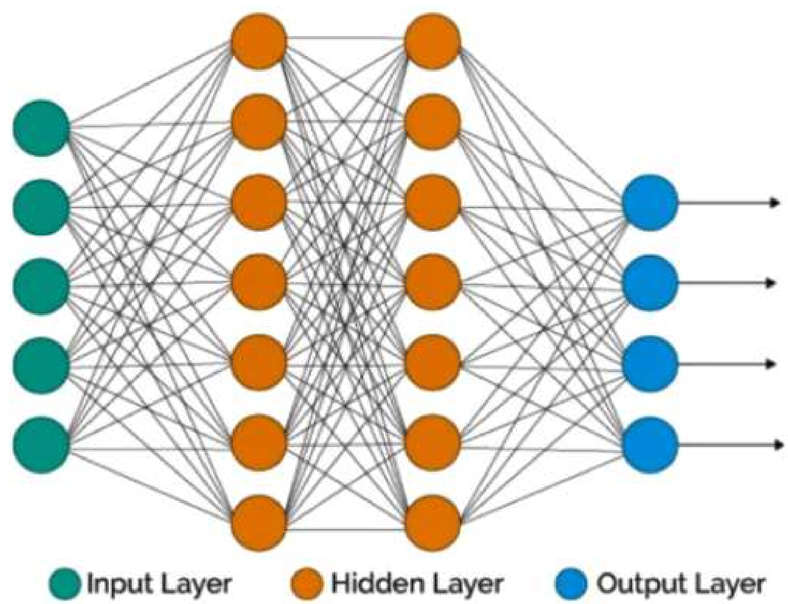
Vectorized bias term gradient

```
grad_b = np.sum((T-Y).dot(V.T) * Z * (1 - Z), axis=0)
```

- Note: the same thing we had for W, except without X (or equivalently, if X was just 1)

More Layers!

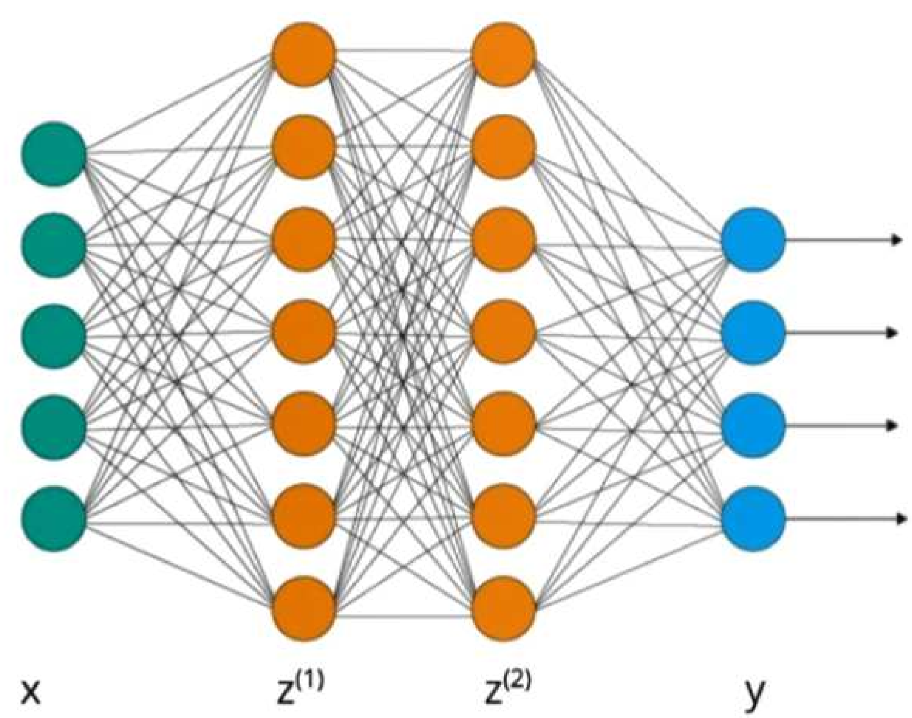
- Can we work our way up to an arbitrary number of layers?
- In order to help us more clearly see the patterns, let's first consider a neural network with 3 layers (of weights)



New Symbols

- We are running out of letters, let's use numbered superscripts instead

$$a^{(1)} = W^{(1)T}x + b^{(1)}$$
$$z^{(1)} = \sigma(a^{(1)})$$
$$a^{(2)} = W^{(2)T}z^{(1)} + b^{(2)}$$
$$z^{(2)} = \sigma(a^{(2)})$$
$$a^{(3)} = W^{(3)T}z^{(2)} + b^{(3)}$$
$$y = softmax(a^{(3)})$$
$$Think\ of\ x = z^{(0)},\ y = z^{(3)}$$



New Symbols

- Superscript weights also - the general pattern is:
  size($W^{(l)}$) = $M^{(l-1)}$ x $M^{(l)}$

$$size(W^{(1)}) = D \times M^{(1)},\ size(b^{(1)}) = M^{(1)}$$
$$size(W^{(2)}) = M^{(1)} \times M^{(2)},\ size(b^{(2)}) = M^{(2)}$$
$$size(W^{(3)}) = M^{(2)} \times K,\ size(b^{(3)}) = K$$
$$Think\ of\ D = M^{(0)},\ K = M^{(3)}$$

Gradients in last layer

- No different than what we saw before - just with new symbols!

$$\partial J/\partial W^{(3)}{}_{m^{(2)}k} = \sum_{n=1}^{N} \sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a^{(3)}{}_{nk}} \frac{\partial a^{(3)}{}_{nk}}{\partial W^{(3)}{}_{m^{(2)}k}}$$

$$\partial J/\partial b^{(3)}{}_{k} = \sum_{n=1}^{N} \sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a^{(3)}{}_{nk}} \frac{\partial a^{(3)}{}_{nk}}{\partial b^{(3)}{}_{k}}$$

Gradients in last layer

- Important: the answer hasn't changed, only the symbols are different

$$\partial J / \partial {W^{(3)}}_{m^{(2)}k} = \sum_{n=1}^{N} (t_{nk} - y_{nk}) {z^{(2)}}_{nm^{(2)}}$$

$$\partial J / \partial {b^{(3)}}_{k} = \sum_{n=1}^{N} (t_{nk} - y_{nk})$$

Gradients in the 2nd layer

- More terms (due to the chain rule) - but still: the answer hasn't changed
- Remember: law of total derivatives

$$\partial J / \partial W^{(2)}{}_{m^{(1)} m^{(2)}} = \sum_{k=1}^{K} \sum_{n=1}^{N} \sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a^{(3)}{}_{nk}} \frac{\partial a^{(3)}{}_{nk}}{\partial z^{(2)}{}_{nm^{(2)}}} \frac{\partial z^{(2)}{}_{nm^{(2)}}}{\partial a^{(2)}{}_{nm^{(2)}}} \frac{\partial a^{(2)}{}_{nm^{(2)}}}{\partial W^{(2)}{}_{m^{(1)} m^{(2)}}}$$

$$\partial J / \partial b^{(2)}{}_{m^{(2)}} = \sum_{k=1}^{K} \sum_{n=1}^{N} \sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a^{(3)}{}_{nk}} \frac{\partial a^{(3)}{}_{nk}}{\partial z^{(2)}{}_{nm^{(2)}}} \frac{\partial z^{(2)}{}_{nm^{(2)}}}{\partial a^{(2)}{}_{nm^{(2)}}} \frac{\partial a^{(2)}{}_{nm^{(2)}}}{\partial b^{(2)}{}_{m^{(2)}}}$$

Gradients in the 2nd layer

- Again assuming sigmoid activation function
- Since there's another layer behind us, where we saw x before we now see $z^{(1)}$

$$\partial J / \partial W^{(2)}{}_{m^{(1)} m^{(2)}} = \sum_{k=1}^{K} \sum_{n=1}^{N} (t_{nk} - y_{nk}) W^{(3)}{}_{m^{(2)} k} z^{(2)}{}_{n m^{(2)}} (1 - z^{(2)}{}_{n m^{(2)}}) z^{(1)}{}_{n m^{(1)}}$$

$$\partial J / \partial b^{(2)}{}_{m^{(2)}} = \sum_{k=1}^{K} \sum_{n=1}^{N} (t_{nk} - y_{nk}) W^{(3)}{}_{m^{(2)} k} z^{(2)}{}_{n m^{(2)}} (1 - z^{(2)}{}_{n m^{(2)}})$$

Gradients for the 1st layer

- The largest # of terms from the chain rule we've seen so far
- Law of total derivatives: must sum over k and $m^{(2)}$
- "Sum over any index that doesn't appear on LHS"

$$\partial J / \partial W^{(1)}{}_{dm^{(1)}} = \sum_{m^{(2)}=1}^{M^{(2)}} \sum_{k=1}^{K} \sum_{n=1}^{N} \sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a^{(3)}{}_{nk}} \frac{\partial a^{(3)}{}_{nk}}{\partial z^{(2)}{}_{nm^{(2)}}} \frac{\partial z^{(2)}{}_{nm^{(2)}}}{\partial a^{(2)}{}_{nm^{(2)}}} \frac{\partial a^{(2)}{}_{nm^{(2)}}}{\partial z^{(1)}{}_{nm^{(1)}}} \frac{\partial z^{(1)}{}_{nm^{(1)}}}{\partial a^{(1)}{}_{nm^{(1)}}} \frac{\partial a^{(1)}{}_{nm^{(1)}}}{\partial W^{(1)}{}_{dm^{(1)}}}$$

$$\partial J / \partial b^{(1)}{}_{m^{(1)}} = \sum_{m^{(2)}=1}^{M^{(2)}} \sum_{k=1}^{K} \sum_{n=1}^{N} \sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a^{(3)}{}_{nk}} \frac{\partial a^{(3)}{}_{nk}}{\partial z^{(2)}{}_{nm^{(2)}}} \frac{\partial z^{(2)}{}_{nm^{(2)}}}{\partial a^{(2)}{}_{nm^{(2)}}} \frac{\partial a^{(2)}{}_{nm^{(2)}}}{\partial z^{(1)}{}_{nm^{(1)}}} \frac{\partial z^{(1)}{}_{nm^{(1)}}}{\partial a^{(1)}{}_{nm^{(1)}}} \frac{\partial a^{(1)}{}_{nm^{(1)}}}{\partial b^{(1)}{}_{m^{(1)}}}$$

Gradients for the 1st layer

- 7 functions → 7 derivatives - go through this slowly by yourself

$$a^{(1)} = W^{(1)T}x + b^{(1)}$$
$$z^{(1)} = \sigma(a^{(1)})$$
$$a^{(2)} = W^{(2)T}z^{(1)} + b^{(2)}$$
$$z^{(2)} = \sigma(a^{(2)})$$
$$a^{(3)} = W^{(3)T}z^{(2)} + b^{(3)}$$
$$y = softmax(a^{(3)})$$
$$J = \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\log y_{nk}$$

$$\partial J/\partial W^{(1)}{}_{dm^{(1)}} = \sum_{m^{(2)}=1}^{M^{(2)}}\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a^{(3)}{}_{nk}} \frac{\partial a^{(3)}{}_{nk}}{\partial z^{(2)}{}_{nm^{(2)}}} \frac{\partial z^{(2)}{}_{nm^{(2)}}}{\partial a^{(2)}{}_{nm^{(2)}}} \frac{\partial a^{(2)}{}_{nm^{(2)}}}{\partial z^{(1)}{}_{nm^{(1)}}} \frac{\partial z^{(1)}{}_{nm^{(1)}}}{\partial a^{(1)}{}_{nm^{(1)}}} \frac{\partial a^{(1)}{}_{nm^{(1)}}}{\partial W^{(1)}{}_{dm^{(1)}}}$$

$$\partial J/\partial b^{(1)}{}_{m^{(1)}} = \sum_{m^{(2)}=1}^{M^{(2)}}\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{k'=1}^{K} \frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a^{(3)}{}_{nk}} \frac{\partial a^{(3)}{}_{nk}}{\partial z^{(2)}{}_{nm^{(2)}}} \frac{\partial z^{(2)}{}_{nm^{(2)}}}{\partial a^{(2)}{}_{nm^{(2)}}} \frac{\partial a^{(2)}{}_{nm^{(2)}}}{\partial z^{(1)}{}_{nm^{(1)}}} \frac{\partial z^{(1)}{}_{nm^{(1)}}}{\partial a^{(1)}{}_{nm^{(1)}}} \frac{\partial a^{(1)}{}_{nm^{(1)}}}{\partial b^{(1)}{}_{m^{(1)}}}$$

Gradients for the 1st layer

- We've solved all of these derivatives already
- Same form but different subscripts
- E.g. $W^{(1)T}x$ and $W^{(2)T}z^{(1)}$ have the same form
- For compactness, I've replaced activation function derivative with z'

$$\partial J/\partial W^{(1)}{}_{dm^{(1)}} = \sum_{m^{(2)}=1}^{M^{(2)}} \sum_{k=1}^{K} \sum_{n=1}^{N} (t_{nk} - y_{nk}) W^{(3)}{}_{m^{(2)}k} z^{(2)\prime}{}_{nm^{(2)}} W^{(2)}{}_{m^{(1)}m^{(2)}} z^{(1)\prime}{}_{nm^{(1)}} x_{nd}$$

$$\partial J/\partial b^{(1)}{}_{m^{(1)}} = \sum_{m^{(2)}=1}^{M^{(2)}} \sum_{k=1}^{K} \sum_{n=1}^{N} (t_{nk} - y_{nk}) W^{(3)}{}_{m^{(2)}k} z^{(2)\prime}{}_{nm^{(2)}} W^{(2)}{}_{m^{(1)}m^{(2)}} z^{(1)\prime}{}_{nm^{(1)}}$$

Patterns to notice

- First layer gradient had 7 derivatives
- Second layer gradient had 5 derivatives
- Output layer gradient had 3 derivatives
- At each layer, we add 2 functions:
  - Linear transformation
  - Activation function

$$a^{(l)} = W^{(l)T} z^{(l-1)} + b^{(l)}$$

$$z^{(l)} = \sigma(a^{(l)})$$

Patterns to notice

- The further we go back, the more common terms we encounter

$$\partial J/\partial W^{(3)}{}_{m^{(2)}k} = \sum_{n=1}^{N}\sum_{k'=1}^{K} \boxed{\frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a^{(3)}{}_{nk}}} \frac{\partial a^{(3)}{}_{nk}}{\partial W^{(3)}{}_{m^{(2)}k}}$$

$$\partial J/\partial W^{(2)}{}_{m^{(1)}m^{(2)}} = \sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{k'=1}^{K} \boxed{\boxed{\frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a^{(3)}{}_{nk}}} \frac{\partial a^{(3)}{}_{nk}}{\partial z^{(2)}{}_{nm^{(2)}}} \frac{\partial z^{(2)}{}_{nm^{(2)}}}{\partial a^{(2)}{}_{nm^{(2)}}}} \frac{\partial a^{(2)}{}_{nm^{(2)}}}{\partial W^{(2)}{}_{m^{(1)}m^{(2)}}}$$

$$\partial J/\partial W^{(1)}{}_{dm^{(1)}} = \sum_{m^{(2)}=1}^{M^{(2)}}\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{k'=1}^{K} \boxed{\boxed{\frac{\partial J_{nk'}}{\partial y_{nk'}} \frac{\partial y_{nk'}}{\partial a^{(3)}{}_{nk}}} \frac{\partial a^{(3)}{}_{nk}}{\partial z^{(2)}{}_{nm^{(2)}}} \frac{\partial z^{(2)}{}_{nm^{(2)}}}{\partial a^{(2)}{}_{nm^{(2)}}}} \frac{\partial a^{(2)}{}_{nm^{(2)}}}{\partial z^{(1)}{}_{nm^{(1)}}} \frac{\partial z^{(1)}{}_{nm^{(1)}}}{\partial a^{(1)}{}_{nm^{(1)}}} \frac{\partial a^{(1)}{}_{nm^{(1)}}}{\partial W^{(1)}{}_{dm^{(1)}}}$$

Patterns to notice

- The same pattern seems to be followed: input to the layer multiplied by some other term
- Let's call it "delta" for convenience

$$\nabla_{W^{(l)}} J = Z^{(l-1)T} \delta^{(l)}$$

Patterns to notice

- Let's consider the weight updates for a 3-layer neural network to "guess" what δ should be

$$\partial J / \partial W^{(3)}{}_{m^{(2)}k} = \sum_{n=1}^{N} (t_{nk} - y_{nk}) z^{(2)}{}_{nm^{(2)}}$$

$$\partial J / \partial W^{(2)}{}_{m^{(1)}m^{(2)}} = \sum_{k=1}^{K} \sum_{n=1}^{N} (t_{nk} - y_{nk}) W^{(3)}{}_{m^{(2)}k} z^{(2)\prime}{}_{nm^{(2)}} z^{(1)}{}_{nm^{(1)}}$$

$$\partial J / \partial W^{(1)}{}_{dm^{(1)}} = \sum_{m^{(2)}=1}^{M^{(2)}} \sum_{k=1}^{K} \sum_{n=1}^{N} (t_{nk} - y_{nk}) W^{(3)}{}_{m^{(2)}k} z^{(2)\prime}{}_{nm^{(2)}} W^{(2)}{}_{m^{(1)}m^{(2)}} z^{(1)\prime}{}_{nm^{(1)}} x_{nd}$$

Delta

$$\partial J / \partial W^{(3)}{}_{m^{(2)}k} = \sum_{n=1}^{N} \boxed{(t_{nk} - y_{nk})} z^{(2)}{}_{nm^{(2)}}$$

$$\delta^{(3)}{}_{nk} = t_{nk} - y_{nk}$$

$$\partial J / \partial W^{(2)}{}_{m^{(1)}m^{(2)}} = \sum_{k=1}^{K} \sum_{n=1}^{N} (t_{nk} - y_{nk}) W^{(3)}{}_{m^{(2)}k} z^{(2)\prime}{}_{nm^{(2)}} z^{(1)}{}_{nm^{(1)}}$$

$$\partial J / \partial W^{(1)}{}_{dm^{(1)}} = \sum_{m^{(2)}=1}^{M^{(2)}} \sum_{k=1}^{K} \sum_{n=1}^{N} (t_{nk} - y_{nk}) W^{(3)}{}_{m^{(2)}k} z^{(2)\prime}{}_{nm^{(2)}} W^{(2)}{}_{m^{(1)}m^{(2)}} z^{(1)\prime}{}_{nm^{(1)}} x_{nd}$$

Delta

$$\delta^{(2)}{}_{nm^{(2)}} = \sum_{k=1}^{K} (t_{nk} - y_{nk}) W^{(3)}{}_{m^{(2)}k} z^{(2)\prime}{}_{nm^{(2)}}$$

$$\partial J / \partial W^{(3)}{}_{m^{(2)}k} = \sum_{n=1}^{N} (t_{nk} - y_{nk}) z^{(2)}{}_{nm^{(2)}}$$

$$\partial J / \partial W^{(2)}{}_{m^{(1)}m^{(2)}} = \sum_{k=1}^{K} \sum_{n=1}^{N} \boxed{(t_{nk} - y_{nk}) W^{(3)}{}_{m^{(2)}k} z^{(2)\prime}{}_{nm^{(2)}}} z^{(1)}{}_{nm^{(1)}}$$

$$\partial J / \partial W^{(1)}{}_{dm^{(1)}} = \sum_{m^{(2)}=1}^{M^{(2)}} \sum_{k=1}^{K} \sum_{n=1}^{N} (t_{nk} - y_{nk}) W^{(3)}{}_{m^{(2)}k} z^{(2)\prime}{}_{nm^{(2)}} W^{(2)}{}_{m^{(1)}m^{(2)}} z^{(1)\prime}{}_{nm^{(1)}} x_{nd}$$

Delta

$$\delta^{(1)}{}_{nm^{(1)}} = \sum_{m^{(2)}=1}^{M^{(2)}} \sum_{k=1}^{K} (t_{nk} - y_{nk}) W^{(3)}{}_{m^{(2)}k} z^{(2)\prime}{}_{nm^{(2)}} W^{(2)}{}_{m^{(1)}m^{(2)}} z^{(1)\prime}{}_{nm^{(1)}}$$

$$\partial J / \partial W^{(3)}{}_{m^{(2)}k} = \sum_{n=1}^{N} (t_{nk} - y_{nk}) z^{(2)}{}_{nm^{(2)}}$$

$$\partial J / \partial W^{(2)}{}_{m^{(1)}m^{(2)}} = \sum_{k=1}^{K} \sum_{n=1}^{N} (t_{nk} - y_{nk}) W^{(3)}{}_{m^{(2)}k} z^{(2)\prime}{}_{nm^{(2)}} z^{(1)}{}_{nm^{(1)}}$$

$$\partial J / \partial W^{(1)}{}_{dm^{(1)}} = \sum_{m^{(2)}=1}^{M^{(2)}} \sum_{k=1}^{K} \sum_{n=1}^{N} \boxed{(t_{nk} - y_{nk}) W^{(3)}{}_{m^{(2)}k} z^{(2)\prime}{}_{nm^{(2)}} W^{(2)}{}_{m^{(1)}m^{(2)}} z^{(1)\prime}{}_{nm^{(1)}}} x_{nd}$$

Delta

Can you see the pattern?

$$\delta^{(3)}{}_{nk} = t_{nk} - y_{nk}$$

$$\delta^{(2)}{}_{nm^{(2)}} = \sum_{k=1}^{K} (t_{nk} - y_{nk}) W^{(3)}{}_{m^{(2)}k} z^{(2)\prime}{}_{nm^{(2)}}$$

$$\delta^{(1)}{}_{nm^{(1)}} = \sum_{m^{(2)}=1}^{M^{(2)}} \sum_{k=1}^{K} (t_{nk} - y_{nk}) W^{(3)}{}_{m^{(2)}k} z^{(2)\prime}{}_{nm^{(2)}} W^{(2)}{}_{m^{(1)}m^{(2)}} z^{(1)\prime}{}_{nm^{(1)}}$$

Delta

- Define delta in terms of delta

$$\delta^{(3)}{}_{nk} = t_{nk} - y_{nk}$$

$$\delta^{(2)}{}_{nm^{(2)}} = \sum_{k=1}^{K} \delta^{(3)}{}_{nk} W^{(3)}{}_{m^{(2)}k} z^{(2)\prime}{}_{nm^{(2)}}$$

$$\delta^{(1)}{}_{nm^{(1)}} = \sum_{m^{(2)}=1}^{M^{(2)}} \delta^{(2)}{}_{nm^{(2)}} W^{(2)}{}_{m^{(1)}m^{(2)}} z^{(1)\prime}{}_{nm^{(1)}}$$

Delta Recursion

$$\delta^{(L)}{}_{nk} = t_{nk} - y_{nk}$$

$$\delta^{(l)}{}_{nm^{(l)}} = \sum_{m^{(l+1)}=1}^{M^{(l+1)}} \delta^{(l+1)}{}_{nm^{(l+1)}} W^{(l+1)}{}_{m^{(l)}m^{(l+1)}} z^{(l)\prime}{}_{nm^{(l)}}, \; for\ l = 1 \ldots L-1$$

Vectorized Delta

$$\delta^{(L)} = T - Y$$

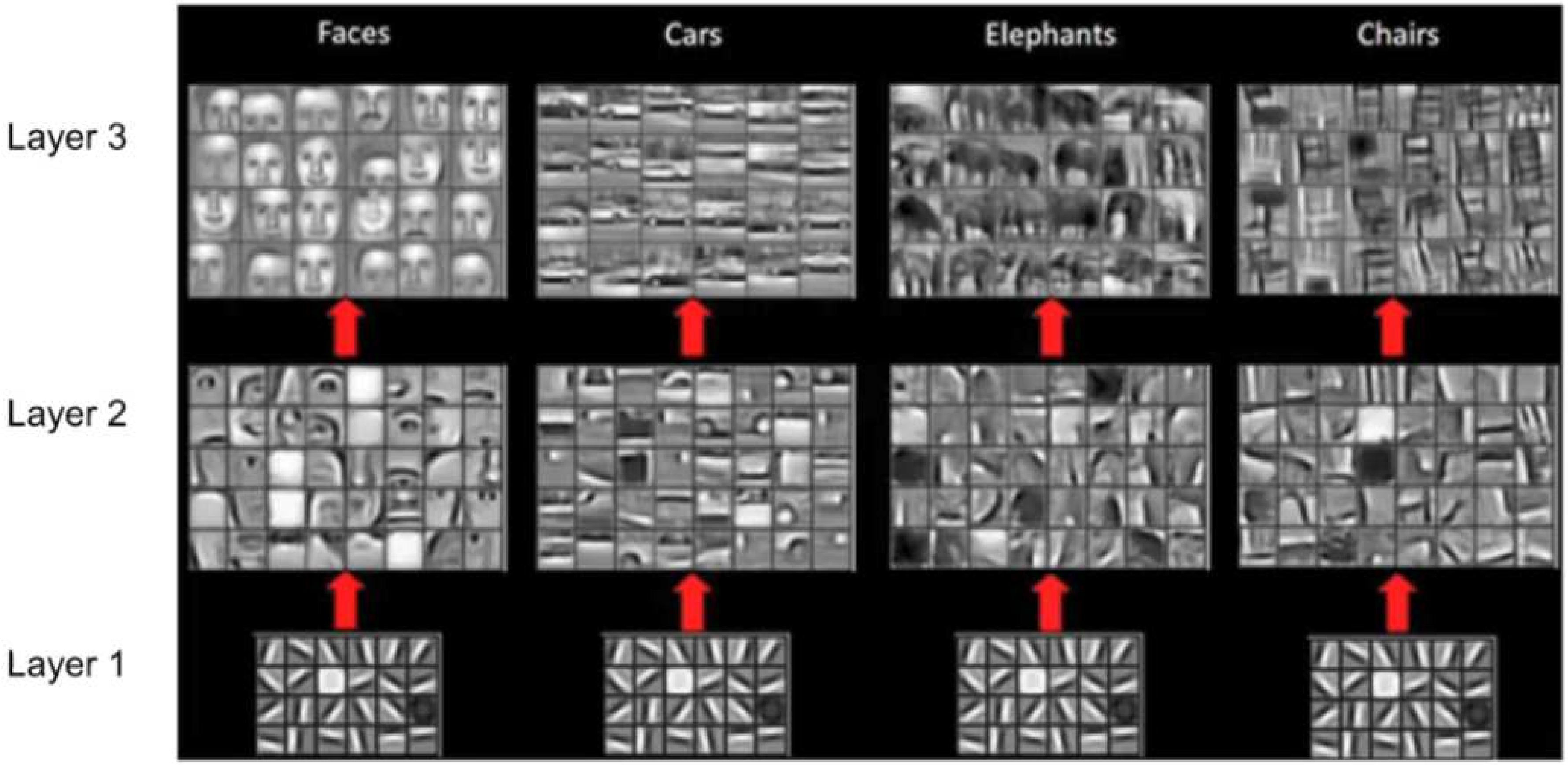
$$\delta^{(l)} = (\delta^{(l+1)} W^{(l+1)T}) \odot Z^{(l)\prime}, \ for \ l = 1 \ldots L - 1$$

Generic Weight Update

$$\nabla_{W^{(l)}} J = Z^{(l-1)T} \delta^{(l)}$$

$$\nabla_{b^{(l)}} J = sum(\delta^{(l)},\ axis = 0) = \delta^{(l)T} 1_N$$

What has a CNN learned?



منابع

- An efcient and improved scheme for handwritten digit recognition based on convolutional neural network Saqib Ali1 · Zeeshan Shaukat1 · Muhammad Azeem1 · Zareen Sakhawat1 · Tariq Mahmood2 · Khalil ur Rehman1 © Springer Nature Switzerland AG 2019
- Back Propagation Neural Networks Article in Substance Use & Misuse · February 1998