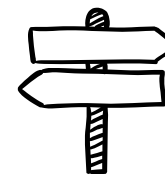


CSC 578

Neural Networks and Deep Learning

6-3. Convolutional Neural Networks (3)

Learning in CNN



Cross-correlation

$$(I \otimes K)_{ij} = \sum_{m=0}^{k_1-1} \sum_{n=0}^{k_2-1} I(i+m, j+n) K(m, n) \quad (1)$$

Convolution

$$(I * K)_{ij} = \sum_{m=0}^{k_1-1} \sum_{n=0}^{k_2-1} I(i-m, j-n) K(m, n) \quad (2)$$

$$= \sum_{m=0}^{k_1-1} \sum_{n=0}^{k_2-1} I(i+m, j+n) K(-m, -n) \quad (3)$$

Learning in CNN



Convolution Output

$$(I * K)_{ij} = \sum_{m=0}^{k_1-1} \sum_{n=0}^{k_2-1} \sum_{c=1}^C K_{m,n,c} \cdot I_{i+m,j+n,c} + b \quad (4)$$

Convolution Output Simplified

$$(I * K)_{ij} = \sum_{m=0}^{k_1-1} \sum_{n=0}^{k_2-1} K_{m,n} \cdot I_{i+m,j+n} + b \quad (5)$$

Learning in CNN formulation



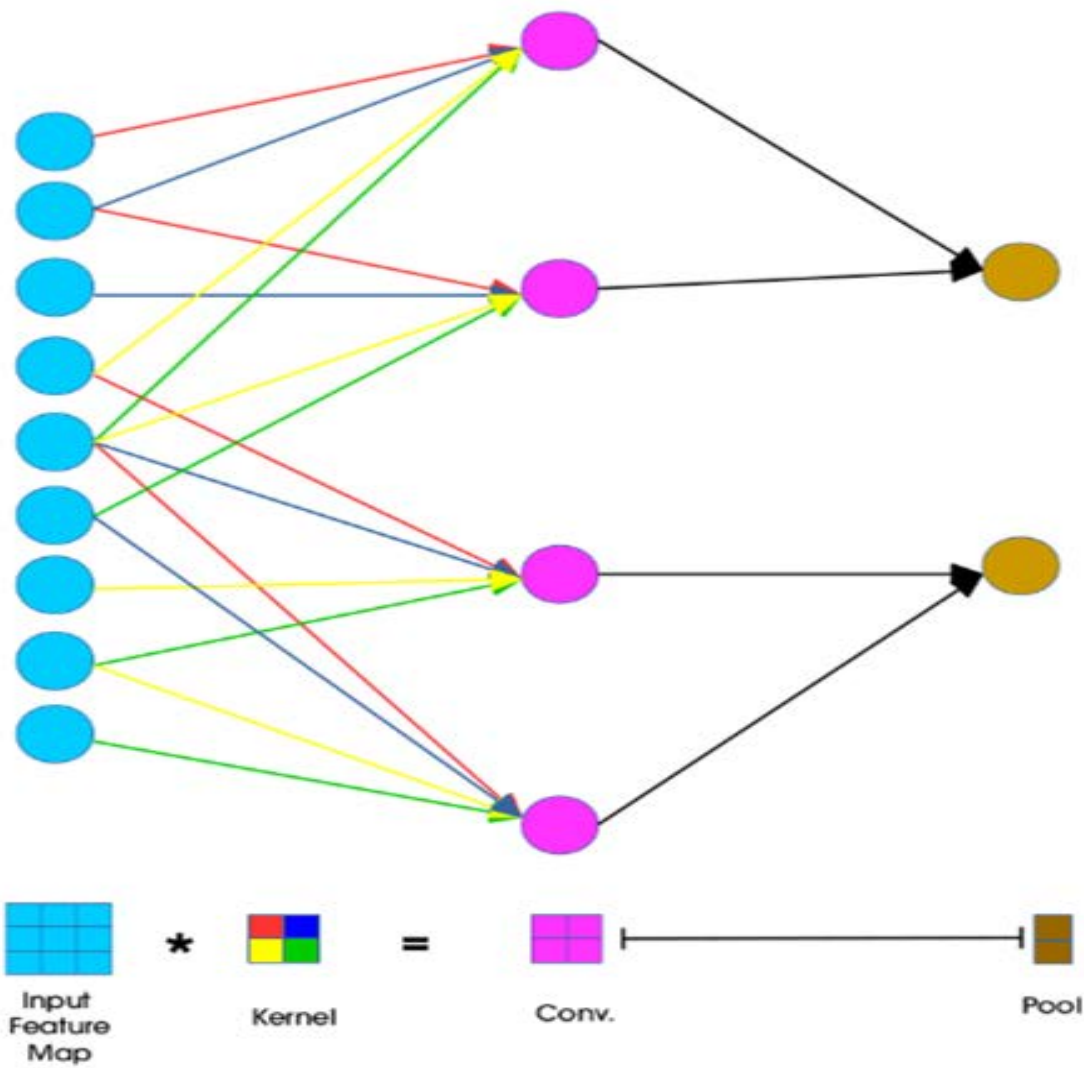
1. l is the l^{th} layer where $l = 1$ is the first layer and $l = L$ is the last layer.
2. Input x is of dimension $H \times W$ and has i by j as the iterators
3. Filter or kernel w is of dimension $k_1 \times k_2$ has m by n as the iterators
4. $w_{m,n}^l$ is the weight matrix connecting neurons of layer l with neurons of layer $l - 1$.
5. b^l is the bias unit at layer l .
6. $x_{i,j}^l$ is the convolved input vector at layer l plus the bias represented as

$$x_{i,j}^l = \sum_m \sum_n w_{m,n}^l o_{i+m,j+n}^{l-1} + b^l$$

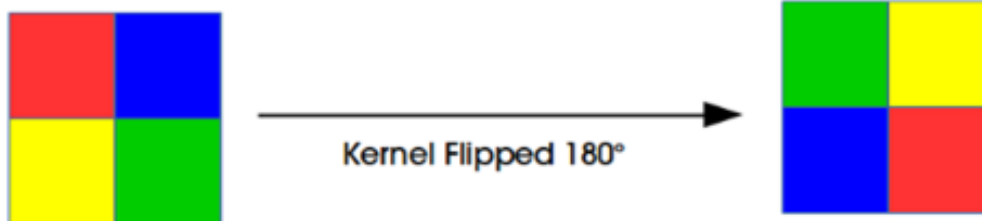
7. $o_{i,j}^l$ is the output vector at layer l given by

$$o_{i,j}^l = f(x_{i,j}^l)$$

8. $f(\cdot)$ is the activation function. Application of the activation layer to the convolved input vector at layer l is given by $f(x_{i,j}^l)$



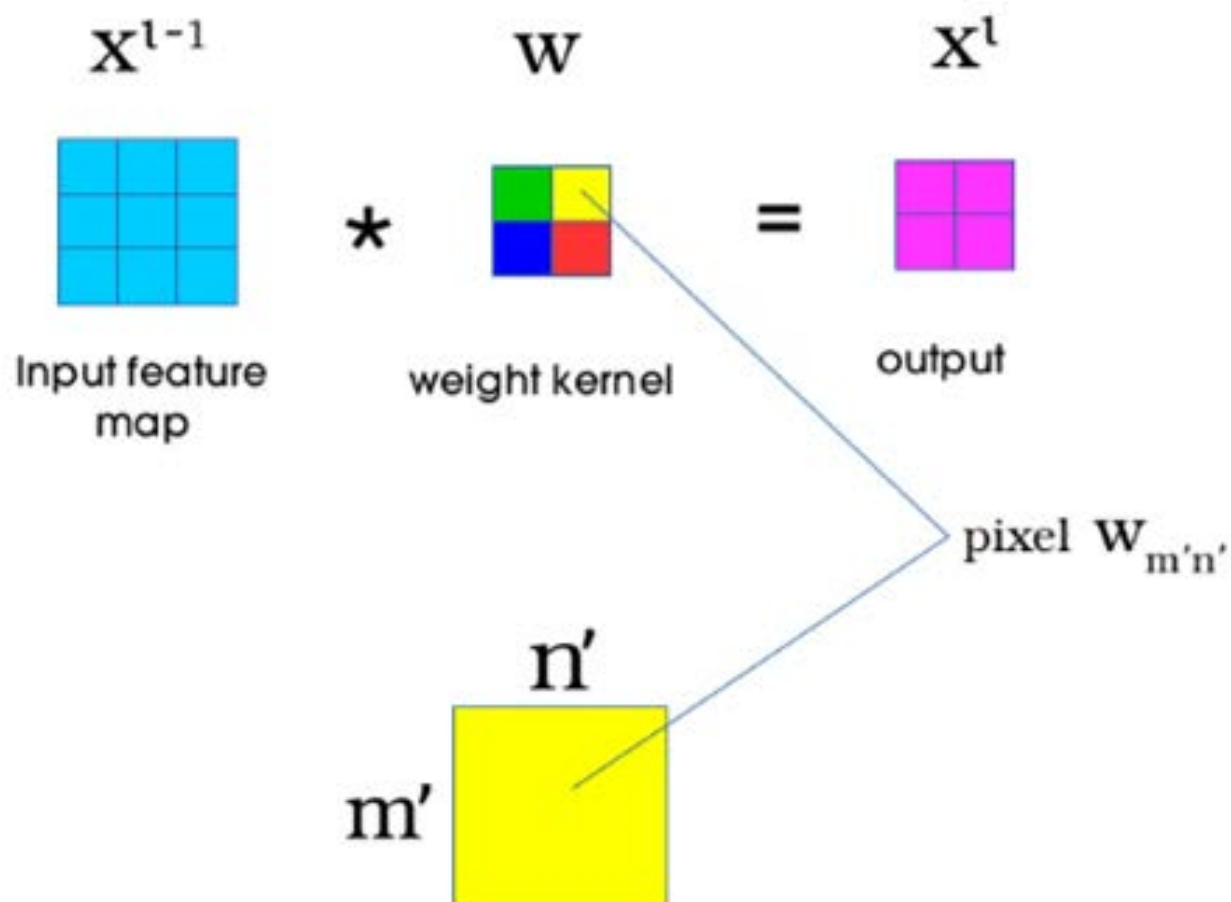
Forward Propagation



$$x_{i,j}^l = \text{rot}_{180^\circ} \{ w_{m,n}^l \} * o_{i,j}^{l-1} + b_{i,j}^l \quad (6)$$

$$x_{i,j}^l = \sum_m \sum_n w_{m,n}^l o_{i+m,j+n}^{l-1} + b_{i,j}^l \quad (7)$$

$$o_{i,j}^l = f(x_{i,j}^l) \quad (8)$$



CNN BackPropagation



$$\begin{aligned}\frac{\partial E}{\partial w_{m',n'}^l} &= \sum_{i=0}^{H-k_1} \sum_{j=0}^{W-k_2} \frac{\partial E}{\partial x_{i,j}^l} \frac{\partial x_{i,j}^l}{\partial w_{m',n'}^l} \\ &= \sum_{i=0}^{H-k_1} \sum_{j=0}^{W-k_2} \delta_{i,j}^l \frac{\partial x_{i,j}^l}{\partial w_{m',n'}^l}\end{aligned}\quad (10)$$

$$\frac{\partial x_{i,j}^l}{\partial w_{m',n'}^l} = \frac{\partial}{\partial w_{m',n'}^l} \left(\sum_m \sum_n w_{m,n}^l o_{i+m,j+n}^{l-1} + b^l \right) \quad (11)$$

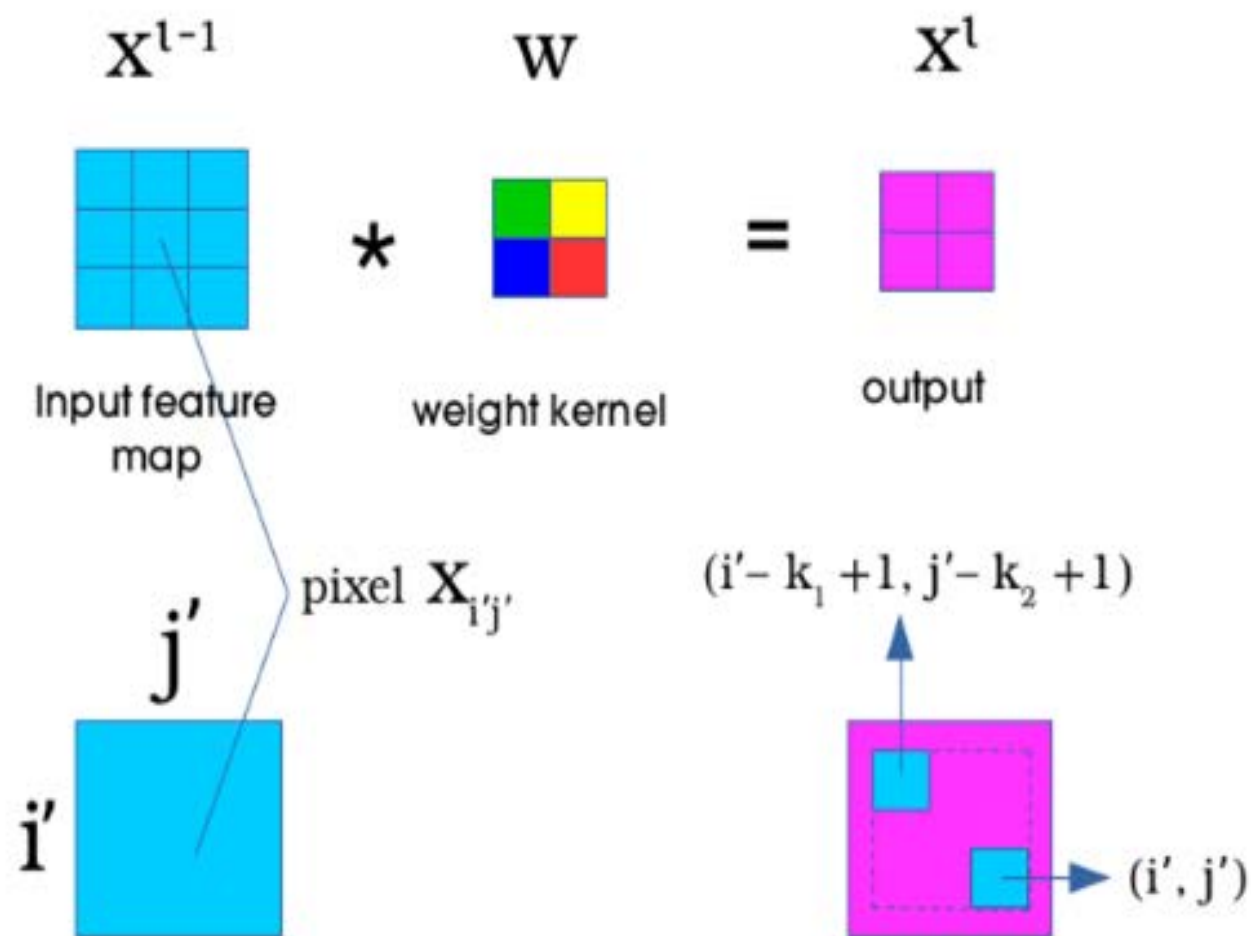
CNN BackPropagation



$$\begin{aligned}
 \frac{\partial x_{i,j}^l}{\partial w_{m',n'}^l} &= \frac{\partial}{\partial w_{m',n'}^l} \left(w_{0,0}^l o_{i+0,j+0}^{l-1} + \dots + w_{m',n'}^l o_{i+m',j+n'}^{l-1} + \dots + b^l \right) \\
 &= \frac{\partial}{\partial w_{m',n'}^l} \left(w_{m',n'}^l o_{i+m',j+n'}^{l-1} \right) \\
 &= o_{i+m',j+n'}^{l-1}
 \end{aligned} \tag{12}$$

$$\frac{\partial E}{\partial w_{m',n'}^l} = \sum_{i=0}^{H-k_1} \sum_{j=0}^{W-k_2} \delta_{i,j}^l o_{i+m',j+n'}^{l-1} \tag{13}$$

$$= \text{rot}_{180^\circ} \left\{ \delta_{i,j}^l \right\} * o_{m',n'}^{l-1} \tag{14}$$



From the diagram above, we can see that region in the output affected by pixel $x_{i',j'}$ from the input is the region in the output bounded by the dashed lines where the top left corner pixel is given by $(i' - k_1 + 1, j' - k_2 + 1)$ and the bottom right corner pixel is given by (i', j') .

Using chain rule and introducing sums give us the following equation:

$$\begin{aligned}\frac{\partial E}{\partial x_{i',j'}^l} &= \sum_{i,j \in Q} \frac{\partial E}{\partial x_Q^{l+1}} \frac{\partial x_Q^{l+1}}{\partial x_{i',j'}^l} \\ &= \sum_{i,j \in Q} \delta_Q^{l+1} \frac{\partial x_Q^{l+1}}{\partial x_{i',j'}^l}\end{aligned}\tag{16}$$

Q in the summation above represents the output region bounded by dashed lines and is composed of pixels in the output that are affected by the single pixel $x_{i',j'}$ in the input feature map. A more formal way of representing Eq. 16 is:

$$\begin{aligned}\frac{\partial E}{\partial x_{i',j'}^l} &= \sum_{m=0}^{k_1-1} \sum_{n=0}^{k_2-1} \frac{\partial E}{\partial x_{i'-m,j'-n}^{l+1}} \frac{\partial x_{i'-m,j'-n}^{l+1}}{\partial x_{i',j'}^l} \\ &= \sum_{m=0}^{k_1-1} \sum_{n=0}^{k_2-1} \delta_{i'-m,j'-n}^{l+1} \frac{\partial x_{i'-m,j'-n}^{l+1}}{\partial x_{i',j'}^l}\end{aligned}\tag{17}$$

In the region Q , the height ranges from $i' = 0$ to $i' = (k_1 - 1)$ and width $j' = 0$ to $j' = (k_2 - 1)$. These two can simply be represented by $i' = m$ and $j' = n$ in the summation since the iterators m and n exists in the following similar ranges from $0 \leq m \leq k_1 - 1$ and $0 \leq n \leq k_2 - 1$.

In Eq. 17, $x_{i'-m, j'-n}^{l+1}$ is equivalent to $\sum_{m'} \sum_{n'} w_{m', n'}^{l+1} o_{i'-m+m', j'-n+n'}^l + b^{l+1}$ and expanding this part of the equation gives us:

$$\begin{aligned} \frac{\partial x_{i'-m, j'-n}^{l+1}}{\partial x_{i', j'}^l} &= \frac{\partial}{\partial x_{i', j'}^l} \left(\sum_{m'} \sum_{n'} w_{m', n'}^{l+1} o_{i'-m+m', j'-n+n'}^l + b^{l+1} \right) \\ &= \frac{\partial}{\partial x_{i', j'}^l} \left(\sum_{m'} \sum_{n'} w_{m', n'}^{l+1} f \left(x_{i'-m+m', j'-n+n'}^l \right) + b^{l+1} \right) \quad (18) \end{aligned}$$

Further expanding the summation in Eq. 17 and taking the partial derivatives for all the components results in zero values for all except the components where $m' = m$ and $n' = n$ so that $f(x_{i'-m+m',j'-n+n'}^l)$ becomes $f(x_{i',j'}^l)$ and $w_{m',n'}^{l+1}$ becomes $w_{m,n}^{l+1}$ in the relevant part of the expanded summation as follows:

$$\begin{aligned}
\frac{\partial x_{i'-m,j'-n}^{l+1}}{\partial x_{i',j'}^l} &= \frac{\partial}{\partial x_{i',j'}^l} \left(w_{m',n'}^{l+1} f(x_{0-m+m',0-n+n'}^l) + \dots + w_{m,n}^{l+1} f(x_{i',j'}^l) + \dots + b^{l+1} \right) \\
&= \frac{\partial}{\partial x_{i',j'}^l} \left(w_{m,n}^{l+1} f(x_{i',j'}^l) \right) \\
&= w_{m,n}^{l+1} \frac{\partial}{\partial x_{i',j'}^l} \left(f(x_{i',j'}^l) \right) \\
&= w_{m,n}^{l+1} f'(x_{i',j'}^l)
\end{aligned} \tag{19}$$

CNN BackPropagation



Substituting Eq. 19 in Eq. 17 gives us the following results:

$$\frac{\partial E}{\partial x_{i',j'}^l} = \sum_{m=0}^{k_1-1} \sum_{n=0}^{k_2-1} \delta_{i'-m,j'-n}^{l+1} w_{m,n}^{l+1} f' \left(x_{i',j'}^l \right) \quad (20)$$

For backpropagation, we make use of the flipped kernel and as a result we will now have a convolution that is expressed as a cross-correlation with a flipped kernel:

$$\begin{aligned} \frac{\partial E}{\partial x_{i',j'}^l} &= \sum_{m=0}^{k_1-1} \sum_{n=0}^{k_2-1} \delta_{i'-m,j'-n}^{l+1} w_{m,n}^{l+1} f' \left(x_{i',j'}^l \right) \\ &= \text{rot}_{180^\circ} \left\{ \sum_{m=0}^{k_1-1} \sum_{n=0}^{k_2-1} \delta_{i'+m,j'+n}^{l+1} w_{m,n}^{l+1} \right\} f' \left(x_{i',j'}^l \right) \end{aligned} \quad (21)$$

$$= \delta_{i',j'}^{l+1} * \text{rot}_{180^\circ} \{ w_{m,n}^{l+1} \} f' \left(x_{i',j'}^l \right) \quad (22)$$

References



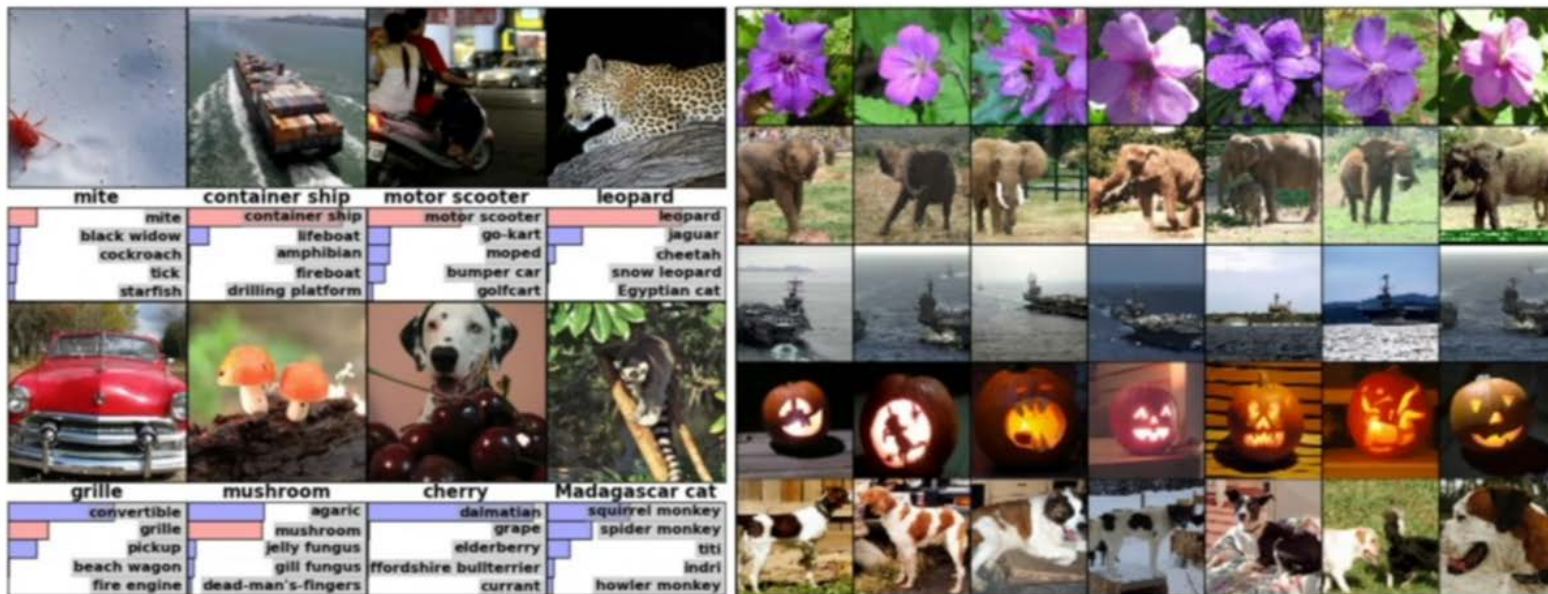
- Useful links:

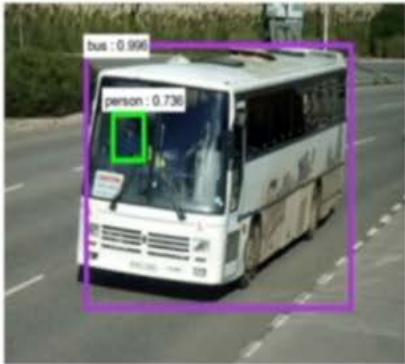
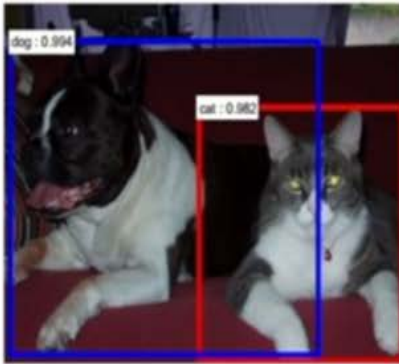
- http://courses.cs.tau.ac.il/Caffe_workshop/Bootcamp/pdf_Lectures/Lecture%203%20CNN%20-%20backpropagation.pdf
- <https://medium.com/@14prakash/back-propagation-is-very-simple-who-made-it-complicated-97b794c97e5c>
- <https://www.jefkine.com/general/2016/09/05/backpropagation-in-convolutional-neural-networks/>

CNN Applications



Image Classification





The top half of the figure shows a raw video frame from a street-level camera. The bottom half shows the same frame with a dense scene graph overlaid. The graph consists of numerous small, semi-transparent colored rectangles (nodes) placed over objects in the scene, such as buildings, trees, cars, and pedestrians. These nodes are interconnected by thin black lines representing edges, which capture spatial relationships like "near", "behind", or "on". The colors used for the nodes include shades of green, blue, red, yellow, and grey, likely representing different object classes or semantic regions. This visualization demonstrates how a computer vision system can automatically generate a structured representation of a complex visual environment.

CNN Applications



self-driving cars

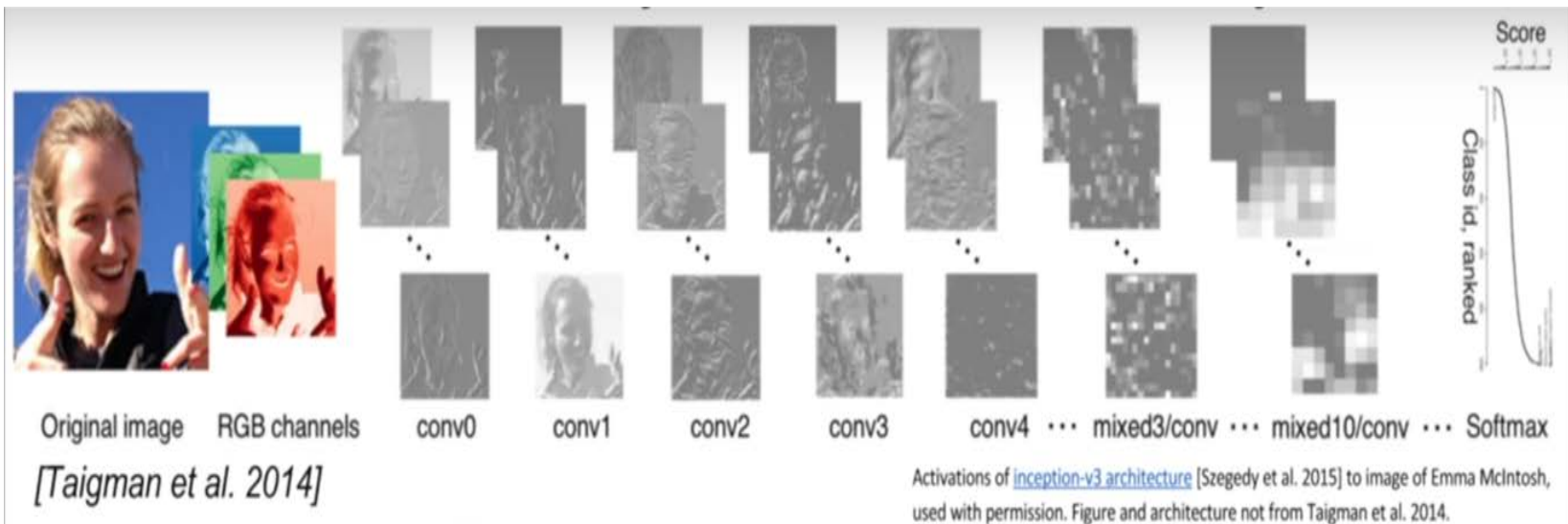
Photo by Lane McIntosh. Copyright CS231n 2017.



[This image](#) by GBPublic_PR is licensed under [CC-BY 2.0](#)

NVIDIA Tesla line
(these are the GPUs on rye01.stanford.edu)

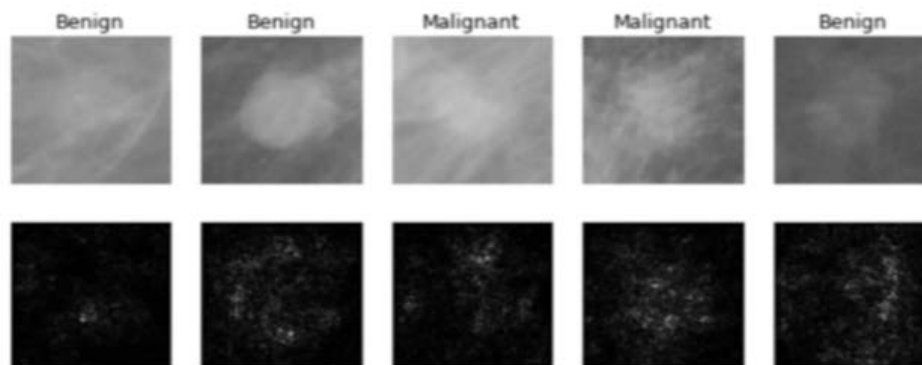
CNN Applications



[Toshev, Szegedy 2014]

Images are examples of pose estimation, not actually from Toshev & Szegedy 2014. Copyright Lane McIntosh.

CNN Applications



[Levy et al. 2016]

Figure copyright Levy et al. 2016.
Reproduced with permission.



[Dieleman et al. 2014]

From left to right: [public domain by NASA](#), usage [permitted](#) by ESA/Hubble, [public domain by NASA](#), and [public domain](#).

CNN Applications



This image by Christin Khan is in the public domain and originally came from the U.S. NOAA.



Whale recognition, Kaggle Challenge

Photo and figure by Lane McIntosh; not actual example from Mnih and Hinton, 2010 paper.



Mnih and Hinton, 2010



CNN Applications



No errors



A white teddy bear sitting in the grass

Minor errors



A man in a baseball uniform throwing a ball

Somewhat related



A woman is holding a cat in her hand

Image Captioning

[Vinyals et al., 2015]
[Karpathy and Fei-Fei, 2015]



A man riding a wave on top of a surfboard



A cat sitting on a suitcase on the floor



A woman standing on a beach holding a surfboard

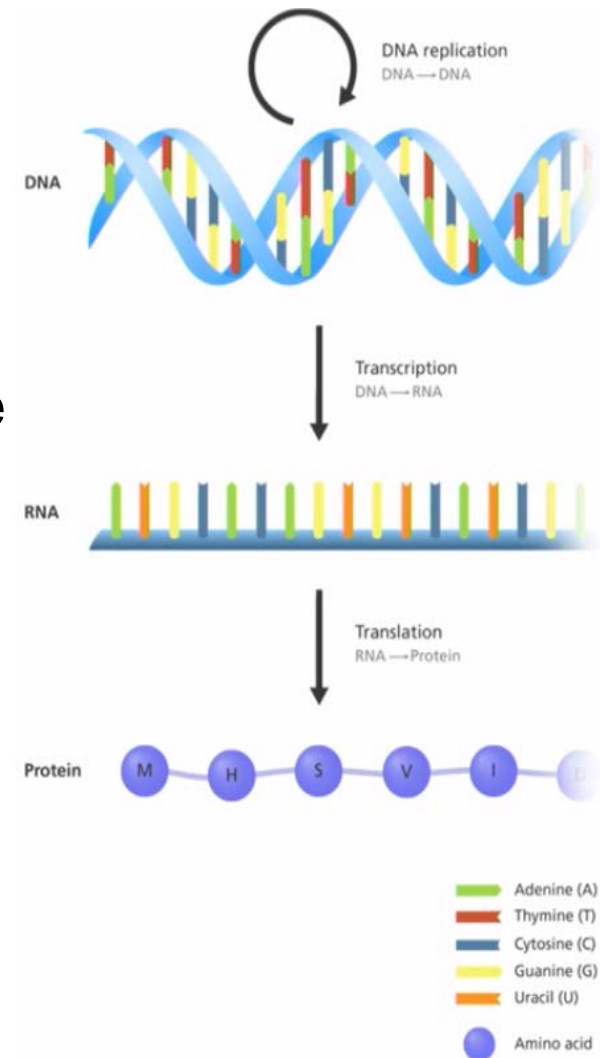
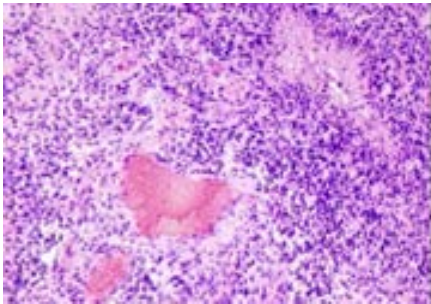
All images are CC0 Public domain:

<https://pixabay.com/en/luggage-antique-cat-1643010/>
<https://pixabay.com/en/teddy-plush-bears-cute-teddy-bear-1623436/>
<https://pixabay.com/en/surf-wave-summer-sport-litoral-1668716/>
<https://pixabay.com/en/woman-female-model-portrait-adult-983967/>
<https://pixabay.com/en/handstand-lake-meditation-496008/>
<https://pixabay.com/en/baseball-player-shortstop-infield-1045263/>

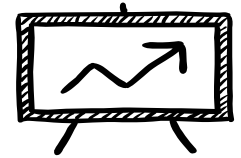
Captions generated by Justin Johnson/Cairo NeuralTalk21

CNN in Medical Imaging

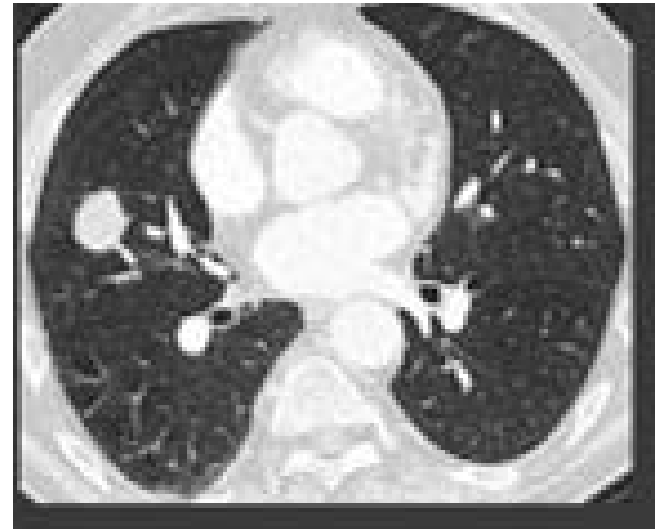
- Biomedical Data:
 - Examples of biomedical data:
 - The omics: Genome, Transcriptome
 - Imaging: Pathology, Radiology



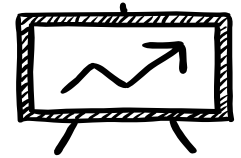
CNN in Medical Imaging



- Volumetric data => 3D images
- MRI is multimodal
- Complementary information
 - Size, location, morphology

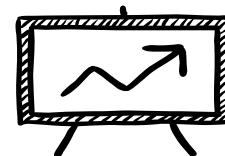


CNN in Medical Imaging



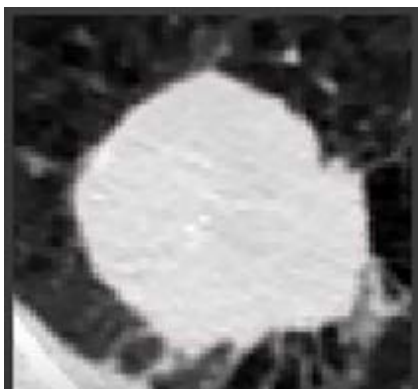
- Two types of image features
 - Semantic features
 - Image features manually annotated by radiologist
 - Radiologist-dependent interpretation of an image
 - Most likely qualitative
 - Computational features
 - (semi)-automatically extracted from an image
 - Most likely quantitative

CNN in Medical Imaging



- Semantic features:
 - Edge shape

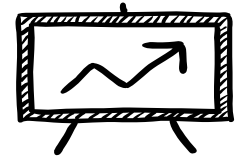
Smooth



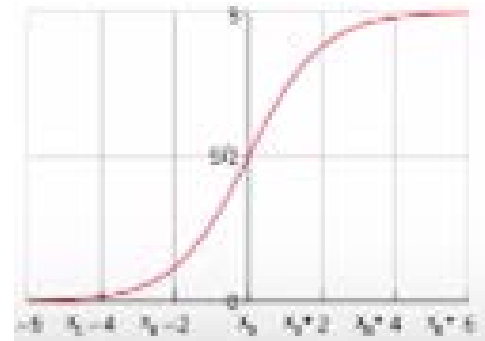
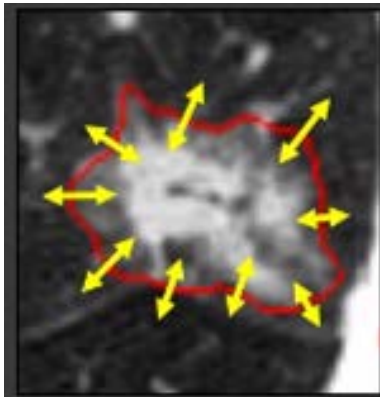
Lobulated



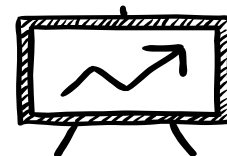
CNN in Medical Imaging



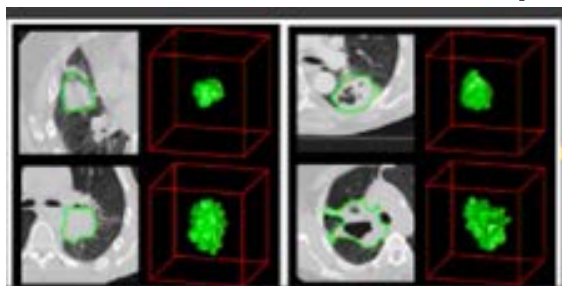
- Computational Features:
 - Texture Features
 - Shape Features
 - Edge Sharpness Features



CNN in Medical Imaging

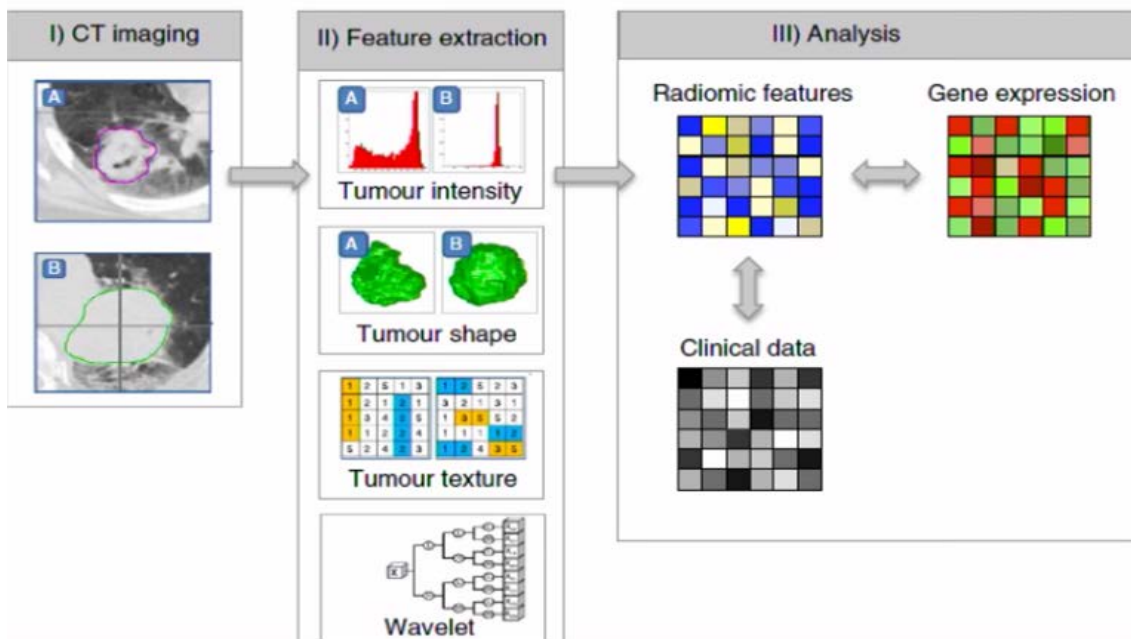


Given CT scan of a patient can we predict survival?

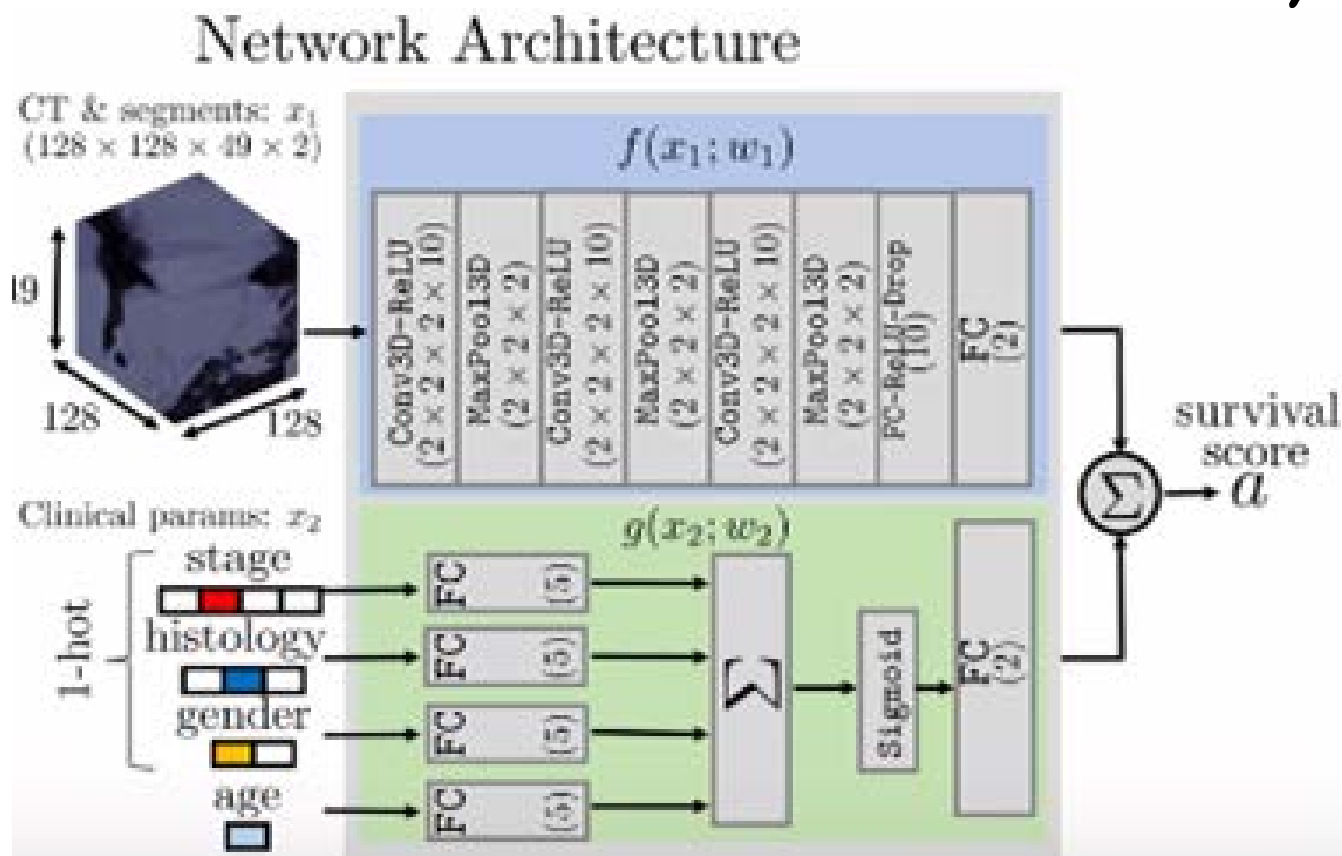
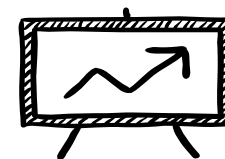


Survival score?

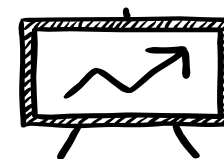
Conventional
Approach



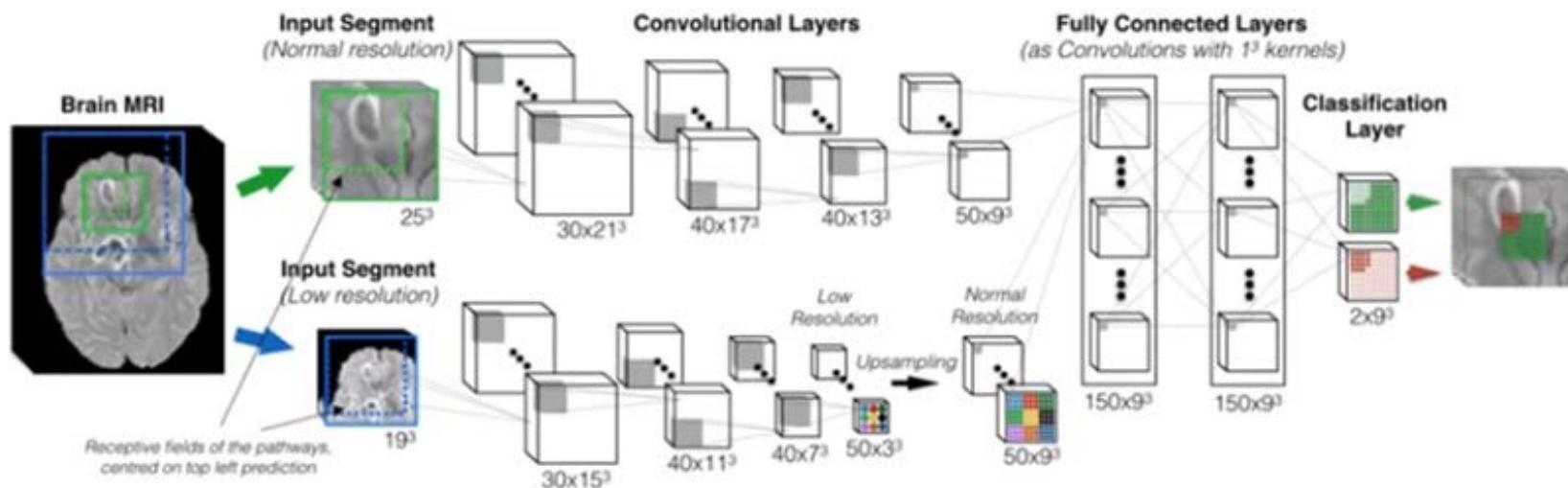
CNN in Medical Imaging



CNN in Medical Imaging



brain lesions



[Kamnitsas et al. 2016]

CNN in Medical Imaging

