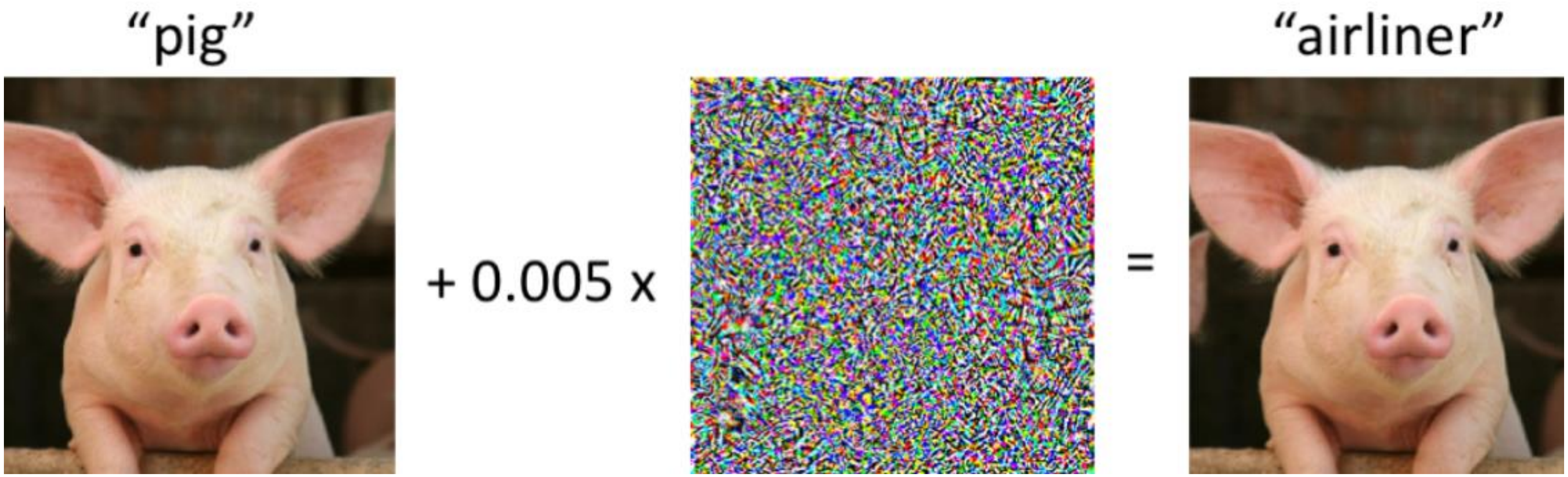
GRAD-CAM

Shingchern D. You

Fooling CNN is easy

- Known as **adversarial attack**
- Such as this one



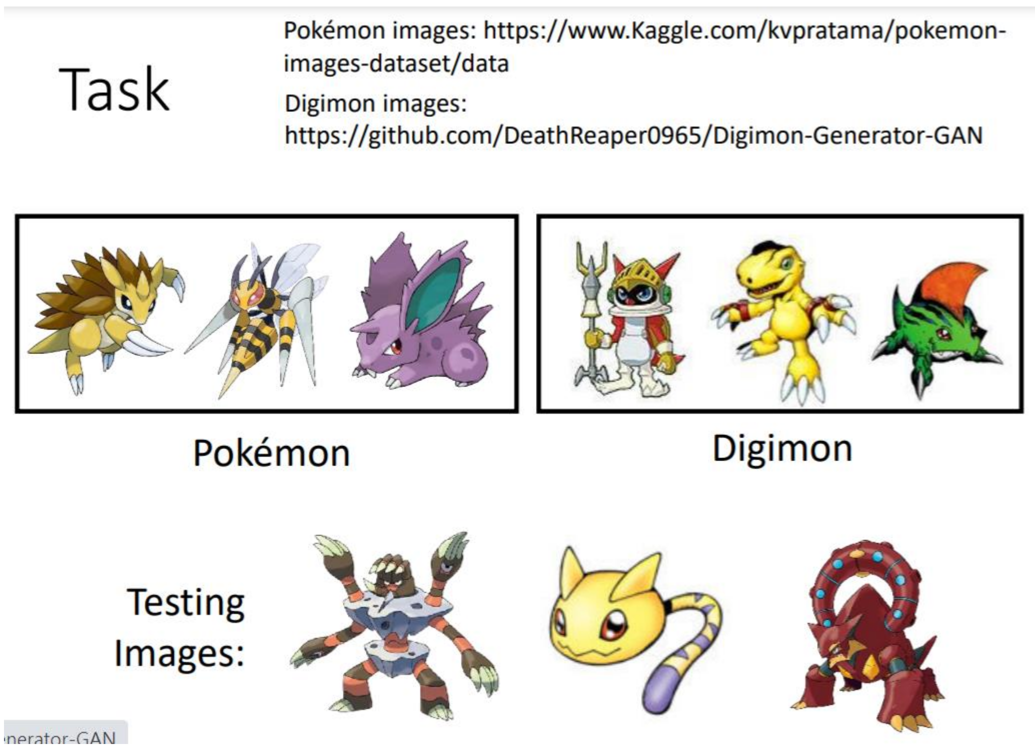
https://miro.medium.com/max/1127/1*i52xqXAc4qUn5qh7m47iKA.png

Fooling CNN is easy

- We can add noisy training samples to cope with this type of attack
  - We do not have time to discuss more about this issue (attack)
- Another problem is that CNN choosing wrong features (usually not detected by humans)
  - High test accuracy
  - But practically useless

Extracting wrong features

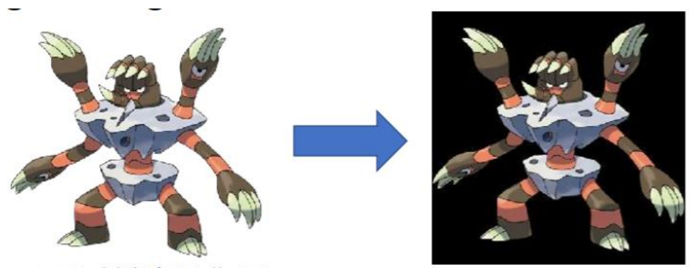
- Task: To classify characters from Pokemon or from Digimon



http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/XAI%20(v7).pdf

Extracting wrong features

- Results: Test accuracy 98.4 % (much better than human classification)
- Too good to be true
  - Pokemon figures are in PNG format
  - Digimon in JPEG format
- Using background color is enough for high accuracy



PNG 檔透明背景　　讀檔後背景是黑的!

http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/XAI%20(v7).pdf

Avoid extracting wrong features

- Need to know the attention of the network
  - Based on which part is the decision made by
- How to do it
  - Randomly alter a small portion of input image and observe the change of class probability
    - Such as multiply pixel values by 1.2 or 0.8
    - Large probability change means more important
  - Visualization (Grad-CAM)

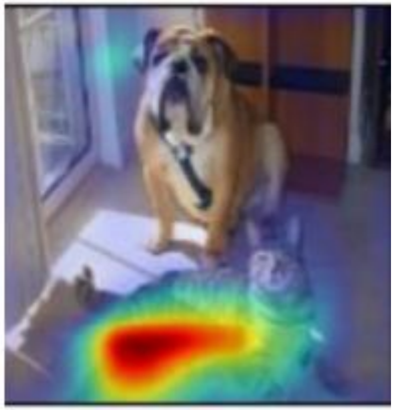
Motivation

- Ref: https://arxiv.org/abs/1610.02391, where photos are from
- This one is the input image (from ref)

- Human knows the location of the cat
- Want to know how CNN recognizes cat or dog

Motivation

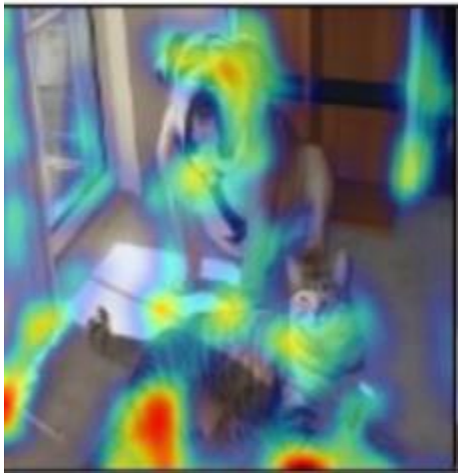
- Check which part the network is focused on (for cat)



- Left one is good, right one is not so good
- Red (or black) color means higher weights

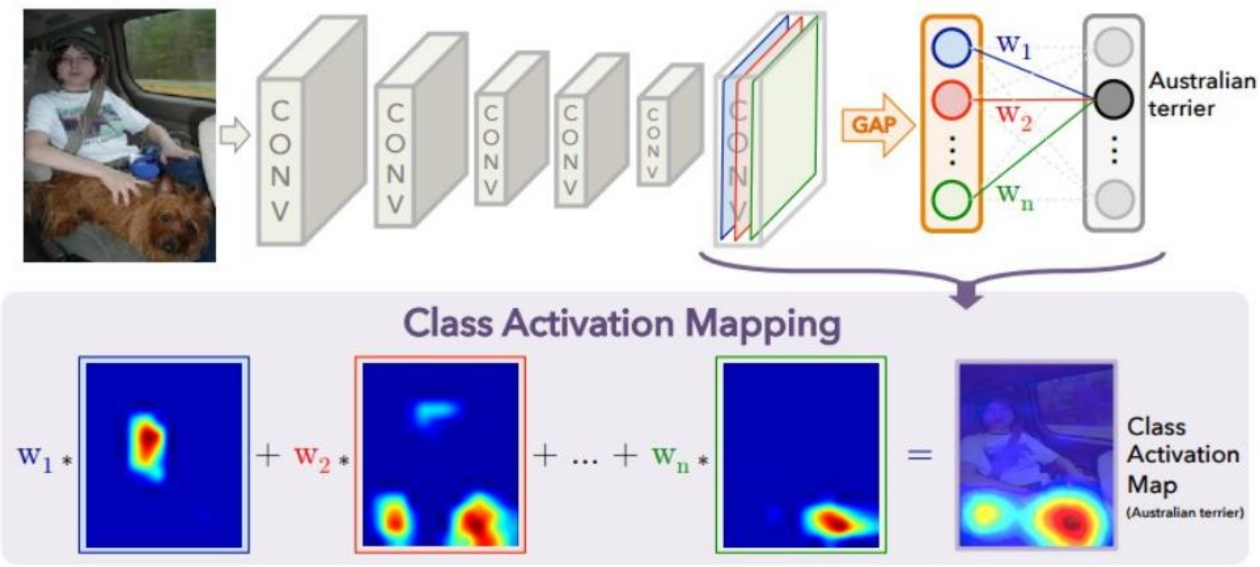
Motivation

- For dog, left is good, right is no good



CAM & global average pooling

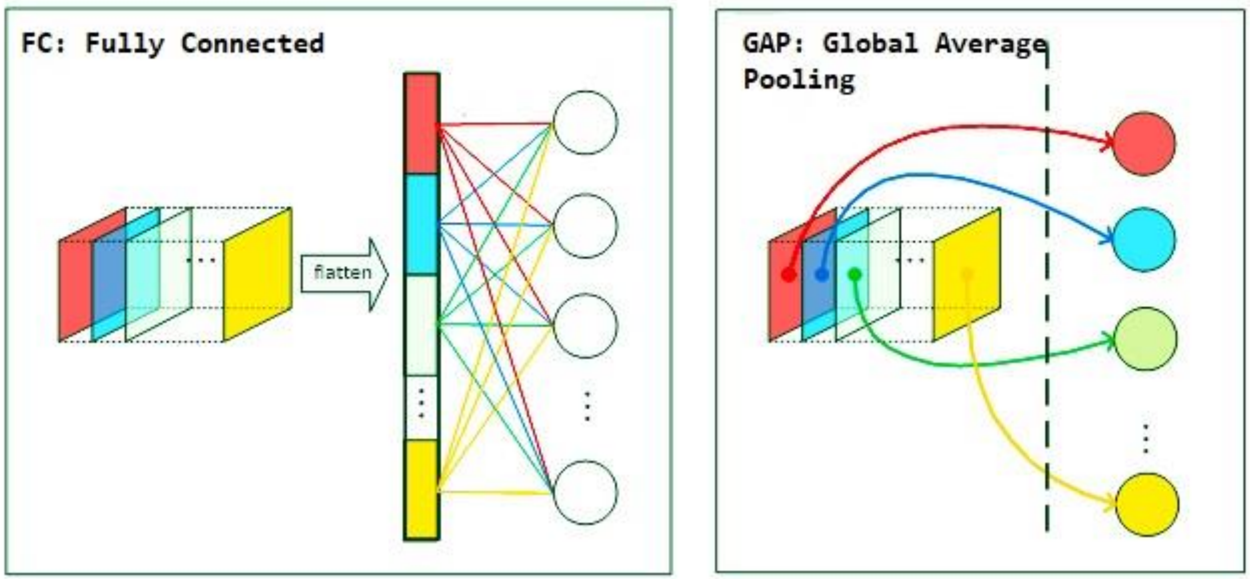
- Grad-CAM is a generalization of CAM (Class activation mapping)
- CAM model is a CNN with global average pooling (GAP) layer



http://cnnlocalization.csail.mit.edu/

GAP layer

- One value per feature map
- Taking average, without activation function



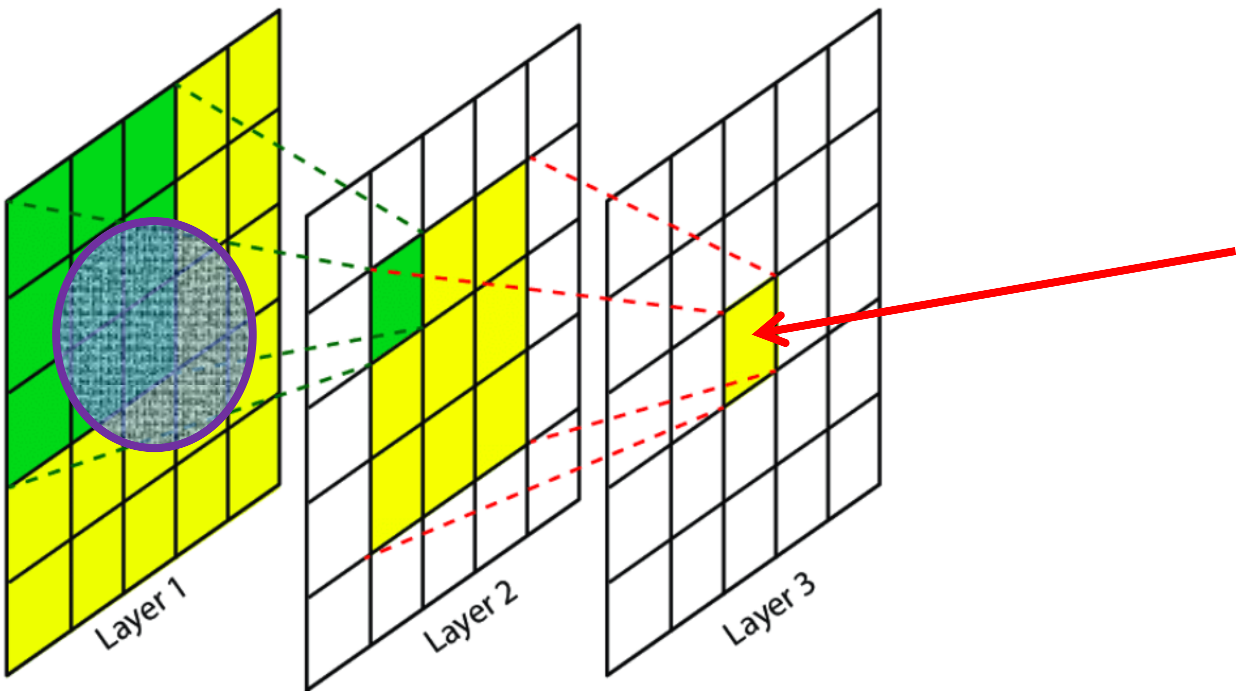
http://spytensor.com/index.php/archives/19/?palujc=xgmsb1

CAM activation map

- As given previously, activation map is a weighted sum of all feature maps (from the last layer)
- Usually activation map is smaller than input image
  - Need to do **up-sampling** to rescale the map size
    - Will discuss how to do it later (in DCGAN)

CNN receptive fields

- If target cell is assigned, we know its corresponding pixels in the input image (spatial info is preserved)



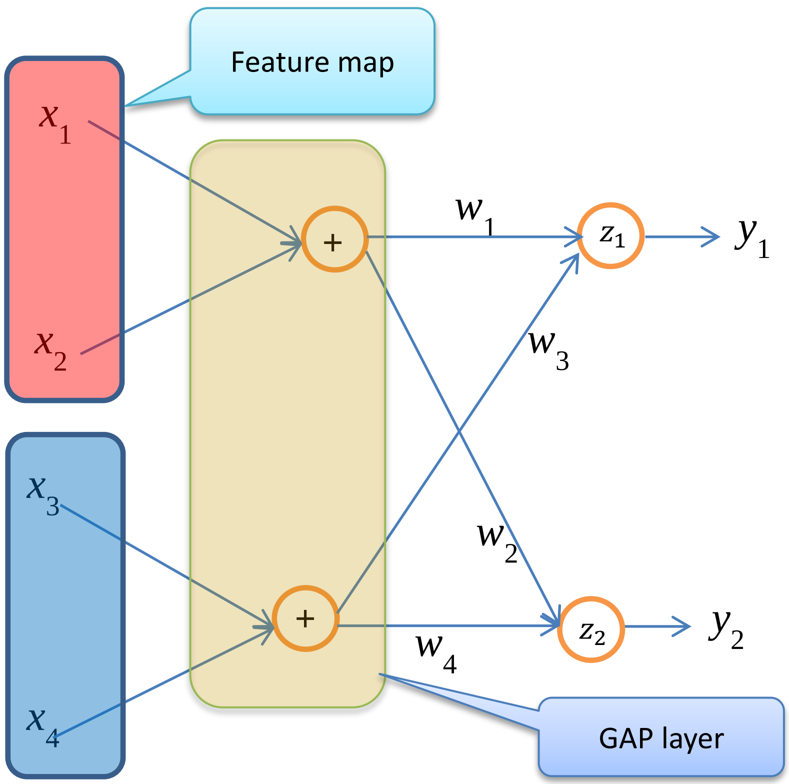
https://theaisummer.com/assets/img/posts/receptive-field/receptive-field-in-convolutional-networks.png

Problems with CAM model

- Need to train CNN with GAP layer
  - Not working with other types of layers (such as fully-connected layer, called dense in keras)
- We don’t want to re-train CNN models just for visualization
- Need a way to indirectly compute necessary weights for same visual effects

Grad-CAM math

- Want to compute w_1 to w_4 in terms of z_1, z_2 and $x_1, \cdots, x_4$



Grad-CAM math

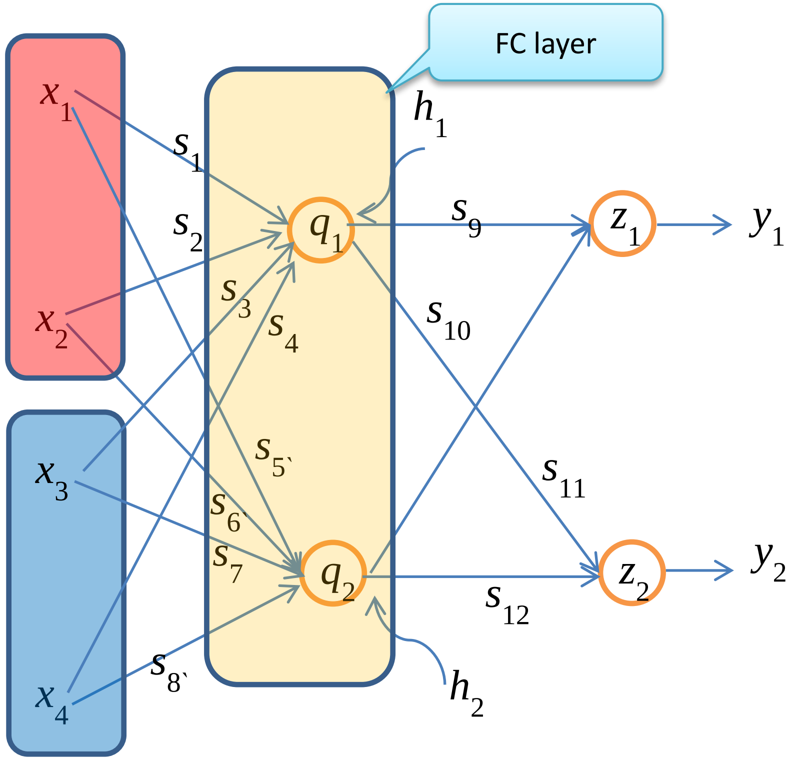
- Recall $y_1 = \text{softmax}(z_1, \text{given } z_1 \text{ and } z_2)$

where $z_1 = \frac{1}{2}(x_1 + x_2) \cdot w_1 + \frac{1}{2}(x_3 + x_4) \cdot w_3$

- With a little bit of math, we have
- $w_1 = \frac{1}{2}\left(\frac{\partial z_1}{\partial x_1} + \frac{\partial z_1}{\partial x_2}\right)$
- That means, if we can find $\frac{1}{2}\left(\frac{\partial z_1}{\partial x_1} + \frac{\partial z_1}{\partial x_2}\right)$, we can compute GAP layer weights
- All other weights can be computed in a similar way

Grad-CAM math

- If we have the following fully connected layer, then how to convert it



Grad-CAM math

- Want to find $w_1 = \frac{1}{2}\left(\frac{\partial z_1}{\partial x_1} + \frac{\partial z_1}{\partial x_2}\right)$
- But, we know (from the figure)

$$\frac{\partial z_1}{\partial x_1} = \frac{\partial z_1}{\partial h_1}\frac{\partial h_1}{\partial q_1}\frac{\partial q_1}{\partial x_1} + \frac{\partial z_1}{\partial h_2}\frac{\partial h_2}{\partial q_2}\frac{\partial q_2}{\partial x_1}$$

$$= s_9 \frac{\partial h_1}{\partial q_1} s_1 + \cdots$$

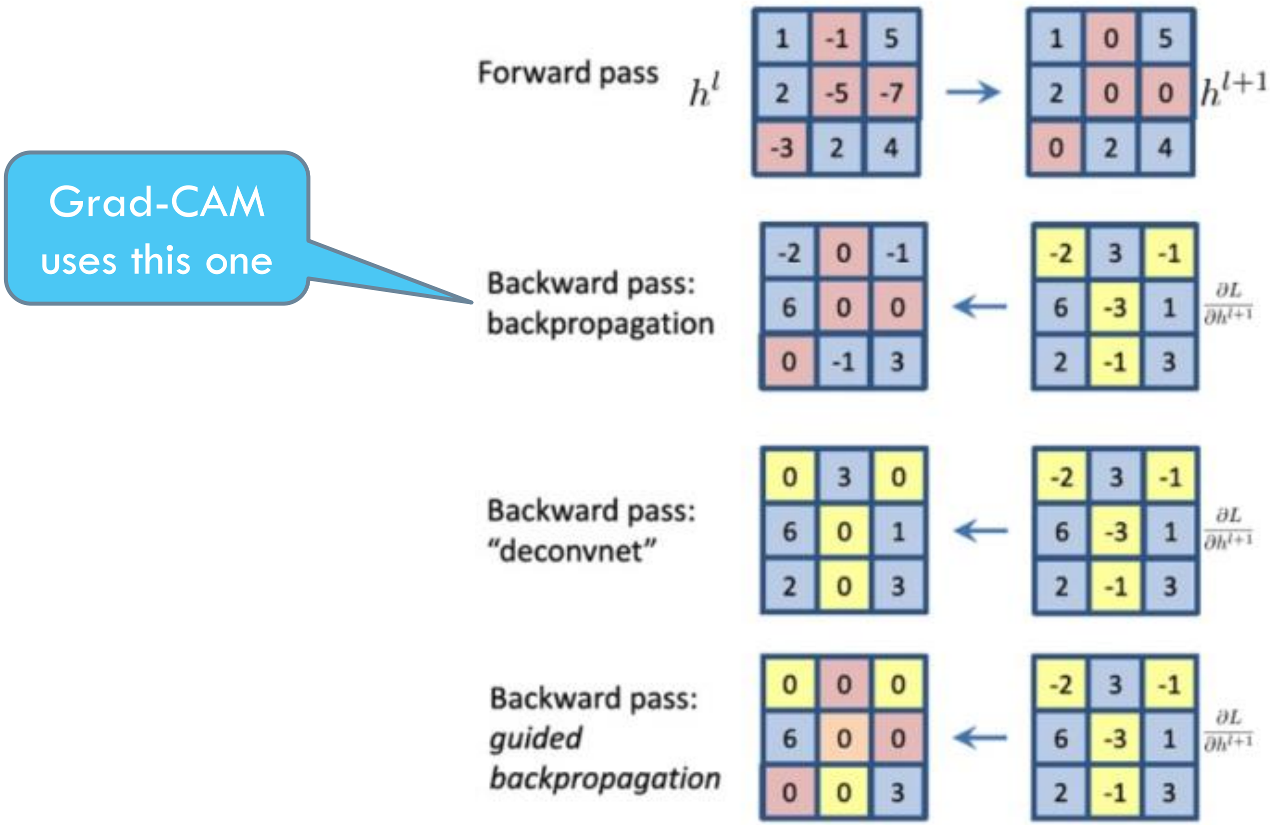
where $\frac{\partial h_1}{\partial q_1} = 1 \text{ or } 0$ if ReLU is used

Grad-CAM math

- What did we do?
  - Let both have same **gradients** only (not same equations)
  - That is why it is called Grad-CAM
- In actual applications, we do not need derivatives
  - Just use existing network weights to compute (equivalent) weights in a CAM network
- With the basic math given previously, you should have no problem to read the original paper

ReLU backpropagation

- There are three possible backprop operations



https://pythonzeal.files.wordpress.com/2018/06/screen-shot-2018-06-24-at-11-40-11-am.png?w=433&h=468

Using Grad-CAM

- You can find lots of resources for Grad-CAM implementation over Internet
  - Number of feature maps to average
- Need to be careful about the orientation of maps
  - Check horizontal and vertical directions
  - Use some well-known pictures to confirm you correctly use it