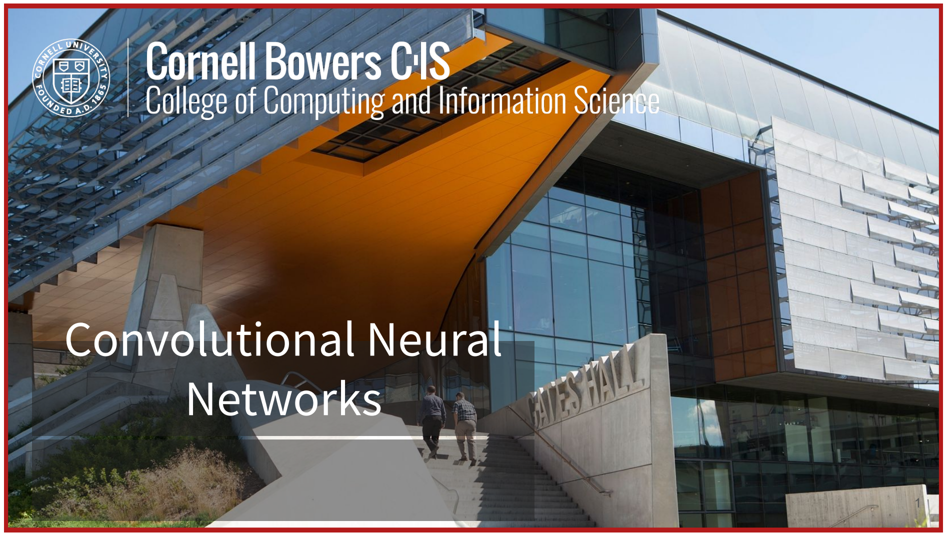


Convolutional Neural Networks

So far…

- MLPs learn complex decision boundaries
- Optimization algorithms use the gradient of the loss to find network parameters
- Different training strategies like regularization, early stopping and normalization can improve training and generalization

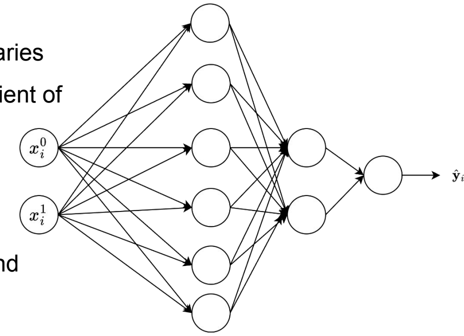
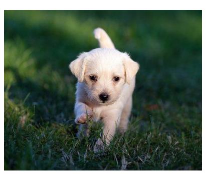


Image Classification


input image

classification →

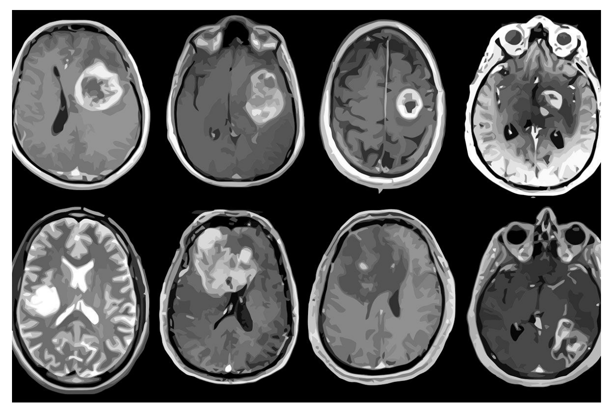
"dog"


input image

classification →

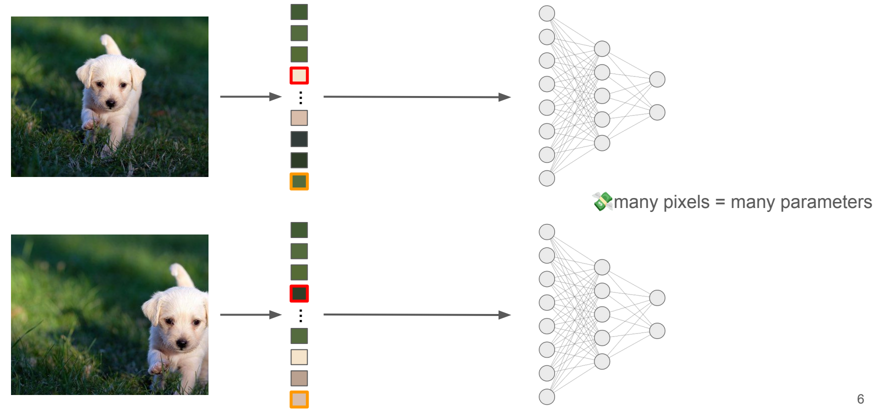
"cat"

Applications in Medicine



Applications in Autonomous Driving

Why not use a Multi-Layer Perceptron?



Convolutional Filters

$$\underbrace{\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \end{bmatrix}}_{\text{“image”}} * \underbrace{\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}}_{\text{convolutional filter}}$$

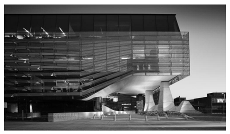
Convolutional Filters

- Aggregates information from local window around pixel
- Translational invariance
- Reduce number of parameters needed to be learned

$$\underbrace{\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \end{bmatrix}}_{\text{“image”}} * \underbrace{\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}}_{\text{convolutional filter}} = \begin{bmatrix} 3 & 2 & 2 \\ 1 & 3 & 2 \\ 2 & 1 & 2 \end{bmatrix}$$

Discuss with your Neighbor!

Match the following convolutional filters with the output they produce.

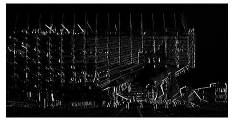


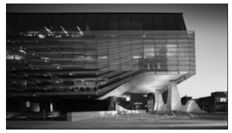
input image

-1	-1	-1
0	0	0
1	1	1

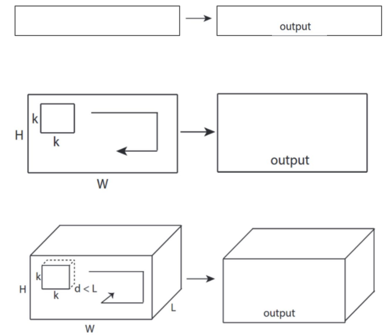
-1	0	1
-1	0	1
-1	0	1

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9





1D and 3D Convolutions



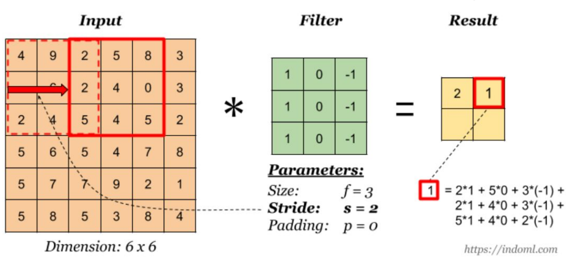
https://wandb.ai/ayush-thakur/dl-question-bank/reports/Intuitive-understanding-of-1D-2D-and-3D-convolutions-in-convolutional-neural-networks---VmlldzoxOTk2MDA

CNNs - Stride

❖ Stride controls how many units the filter / the receptive field shift at a time

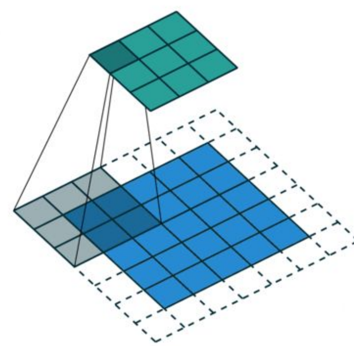
❖ The size of the output image shrinks more as the stride becomes larger

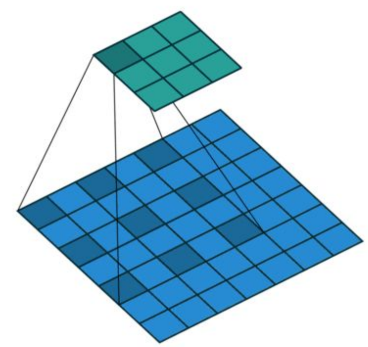
❖ The receptive fields overlap less as the stride becomes larger



Filter with stride (s) = 2

Dilated Convolutions

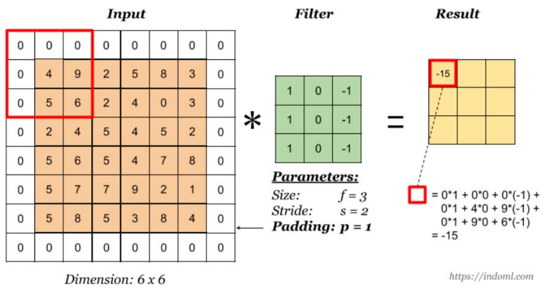




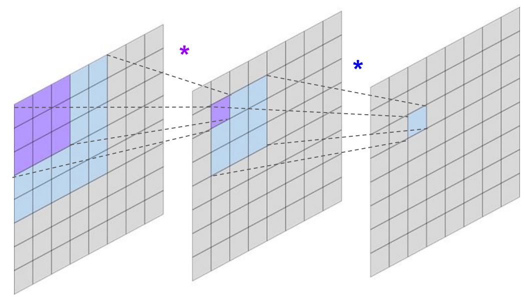
https://towardsdatascience.com/review-dilated-convolution-semantic-segmentation-9d5a5bd768f5

CNNs - Padding

- Padding adds layers of zeros (or other number) around image border
- Prevents image shrinking and loss of information from image boundary



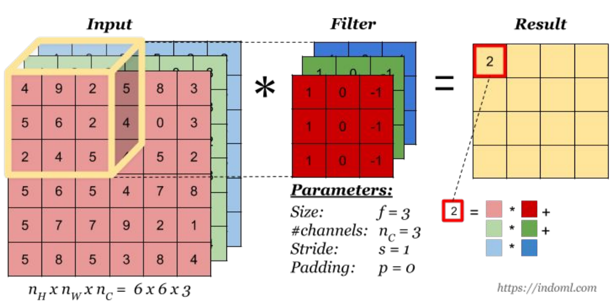
Stacking Convolutions



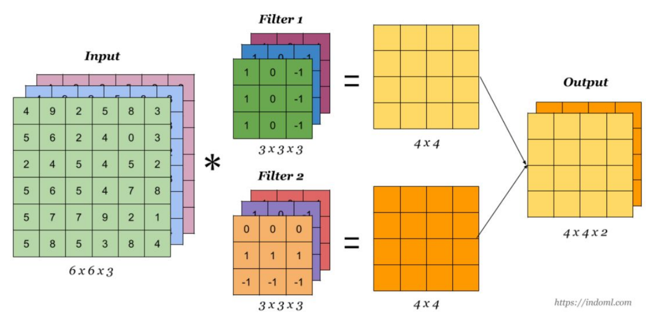
- Size of **receptive field** increases with each layer
- Capture more complex features

Convolution Over Volumes

What if our input image has more than one channel?



Convolution Operation with Multiple Filters

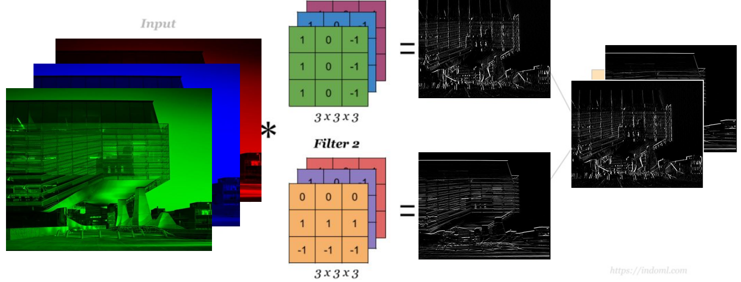


This is different from 3D convolution, in what way?

Convolution Operation with Multiple Filters

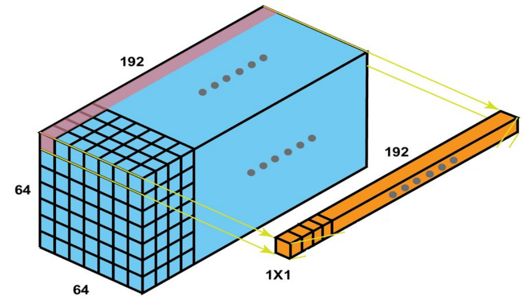
3 input channels

2 output channels

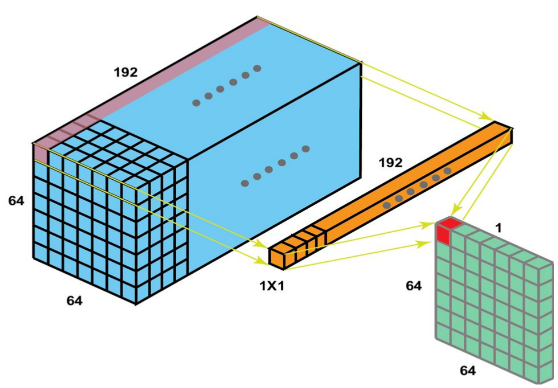


Discuss: 1x1 Convolutions

What is the result of convolving a 64x64x192 dimensional cube with a 1x1 filter?

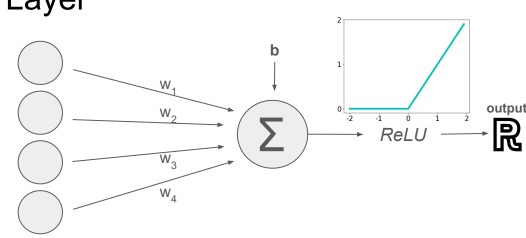


1x1 Convolutions

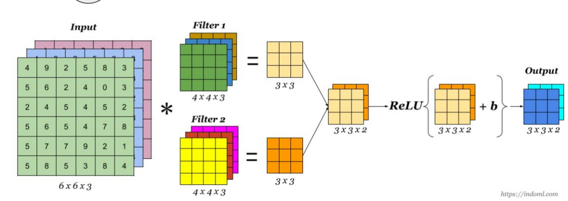


Convolution Layer

MLP Layer



Convolution Layer

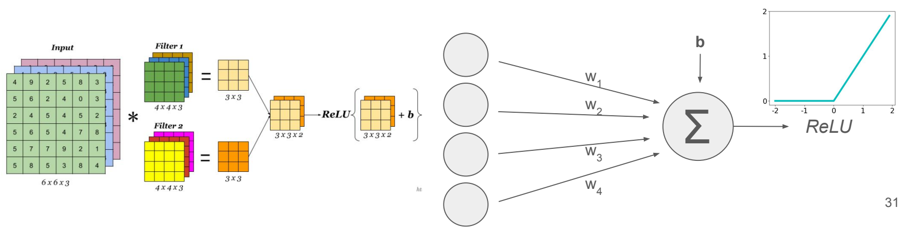


CNN/MLP Equivalence

Differences in a convolution layer:

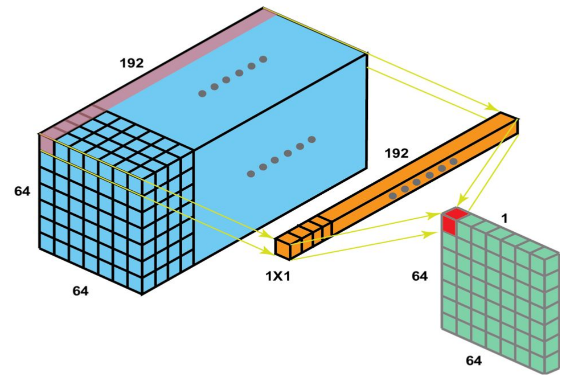
- neurons are connected to a local region
- Weights are shared across multiple parameters

CONV layers can be converted to Fully connected layers and vice versa!

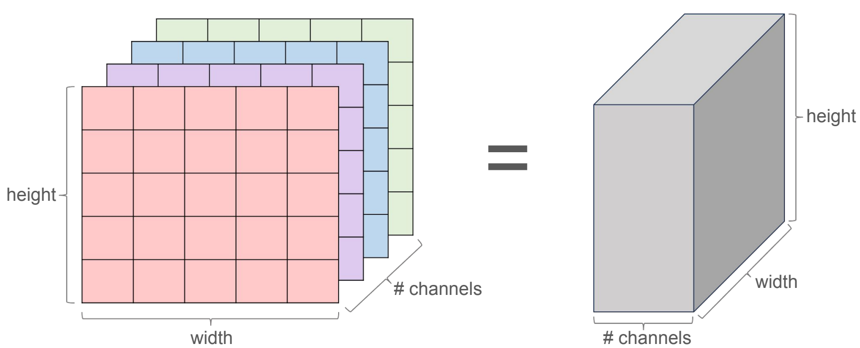


Discuss: Trade-offs between CNNs and MLPs

How would this image change if you used an MLP instead of a 1x1 convolution filter to produce a (64x64x1) feature map? Hint: think about parameter counts and feature interactions.

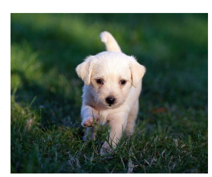


CNN Layer Output Visualization

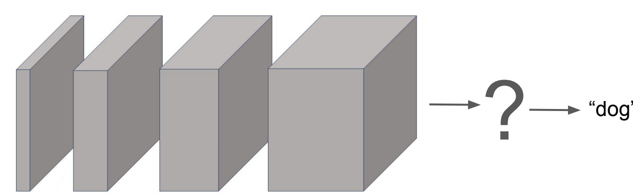


Convolutional Neural Networks (CNNs)

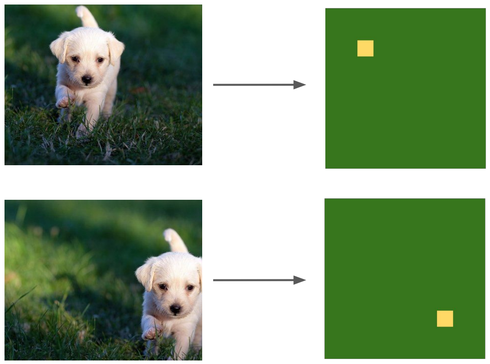
✅ **Convolutions** Maintain spatial relation between pixels
Reduce number of parameters through weight sharing



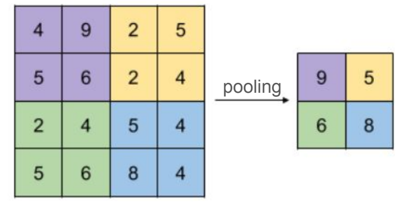
input image



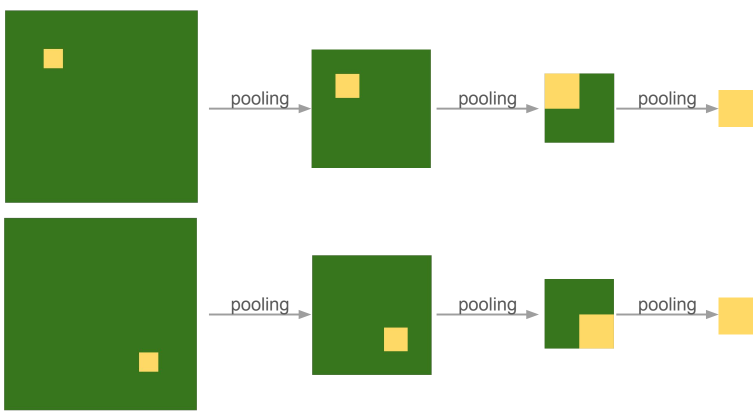
Ensuring translational invariance



Max Pooling

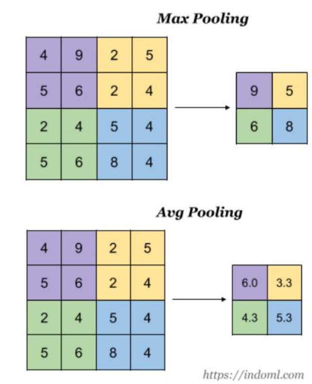


CNNs - Pooling



CNNs - Pooling

- Down sample feature maps that highlight the most present feature in the patch
- Improve efficiency by reducing computations with downsampling
- Increase receptive field size

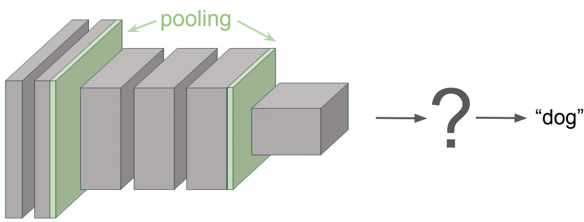


Convolutional Neural Networks (CNNs)

✅ Convolutions — Maintain spatial relation between pixels
Reduce number of parameters through weight sharing

✅ **Pooling** — Captures key information from across different areas of the feature maps
Together with convolutions allows for translational invariance


input image



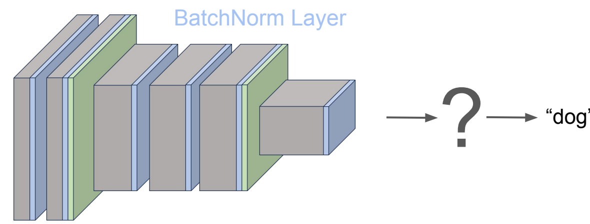
Convolutional Neural Networks (CNNs)

✅ Convolutions — Maintain spatial relation between pixels
Reduce number of parameters through weight sharing

✅ Pooling — Captures key information from across different areas of the feature maps
Together with convolutions allows for translational invariance

✅ **BatchNorm** — Increases speed and stability of training


input image



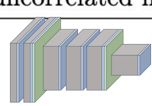
Batch Normalization

1. Efficient BackProp 17

[Efficient Backprop, LeCun et al., 1998]

Transforming the Inputs

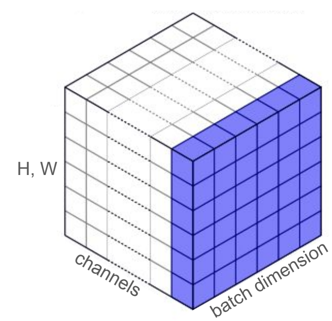
1. The average of each input variable over the training set should be close to zero.
2. Scale input variables so that their covariances are about the same.
3. Input variables should be uncorrelated if possible.



BatchNorm: Transform features throughout layers [Ioffe and Szegedy, 2015]
However this would be very slow, why?

Batch Normalization [Ioffe and Szegedy, 2015]

- Normalize channels to mean 0 and variance 1 across each training batch
- Increases speed of training by enabling the use of larger learning rates
- Improves stability of training



The Batch Normalization Algorithm

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

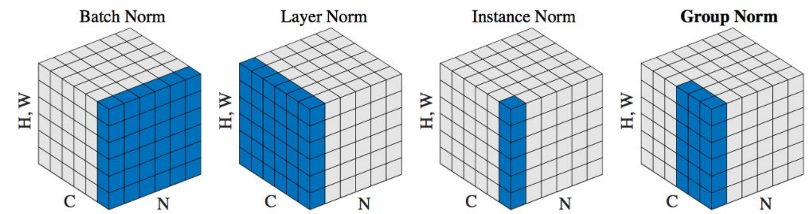
$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

Cornell Bowers C·IS

Many Kinds of Normalization Layers

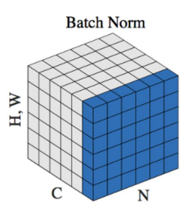


Normalization Methods

"Group Normalization" by Wu et al., 2018

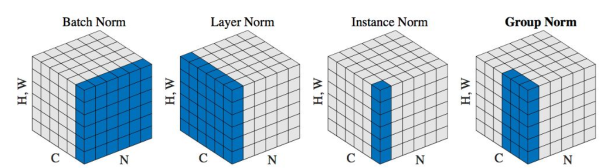
Discuss!

What is the dimension of the mean when you compute the batch norm of a volume of dimension (b x c x h x w)?



Normalization Layers

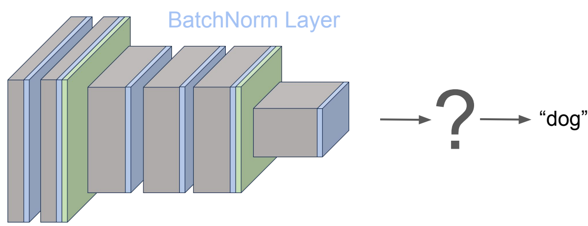
- Normalization layers improve training stability
- Can train with larger learning rates
  - Faster training
- A large learning rate acts as an implicit regularizer
  - Better generalization



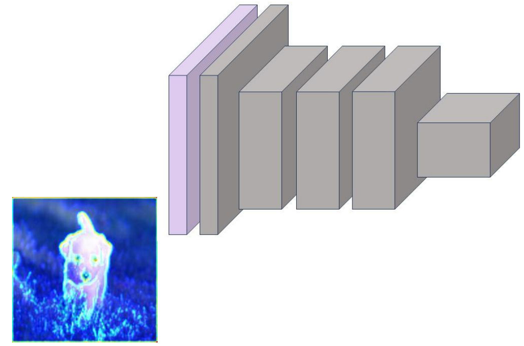
Convolutional Neural Networks (CNNs)

✅ Convolutions	Maintain spatial relation between pixels Reduce number of parameters through weight sharing
✅ Pooling	Captures key information from across different areas of the feature maps Together with convolutions allows for translational invariance
✅ **BatchNorm**	Increases speed and stability of training

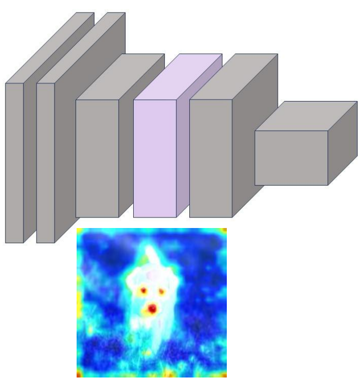
input image



Convolutional Neural Networks (CNNs)



Convolutional Neural Networks (CNNs)



Convolutional Neural Networks (CNNs)

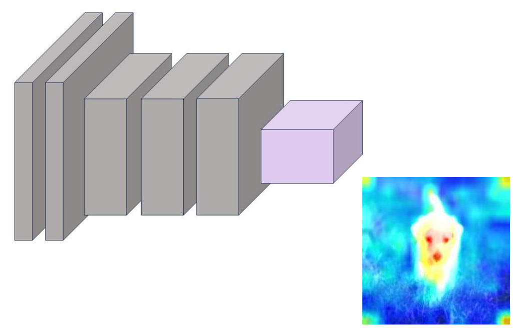
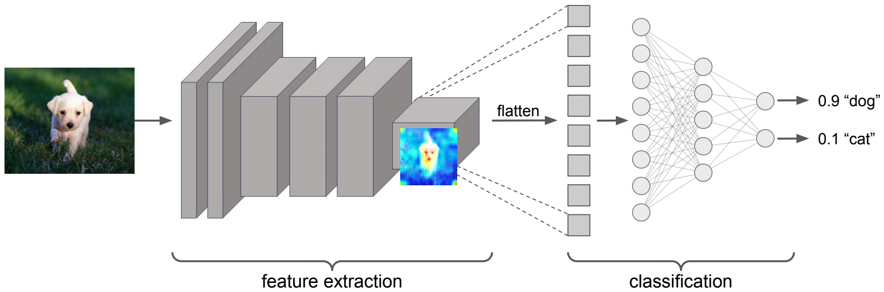


Image Classification



Best Practices

- Input image dimensions is divisible by 2
- Small conv filters (3x3 or 5x5)
- Zero padding is used to maintain spatial resolution
- Max pooling for downsampling
- Pooling layers have a receptive field of 2 and stride of 2

Summary

- CNNs are primarily designed to process and analyze visual data, such as images and videos.
- Key components: convolution layers, pooling layers, activation functions, normalization layers
- Advantages:
  - Translational Invariance
  - Parameter sharing
  - Feature learning
- Can be trained with backprop
- Used for tasks such as segmentation, classification, object detection, etc.