Limitations of MLP

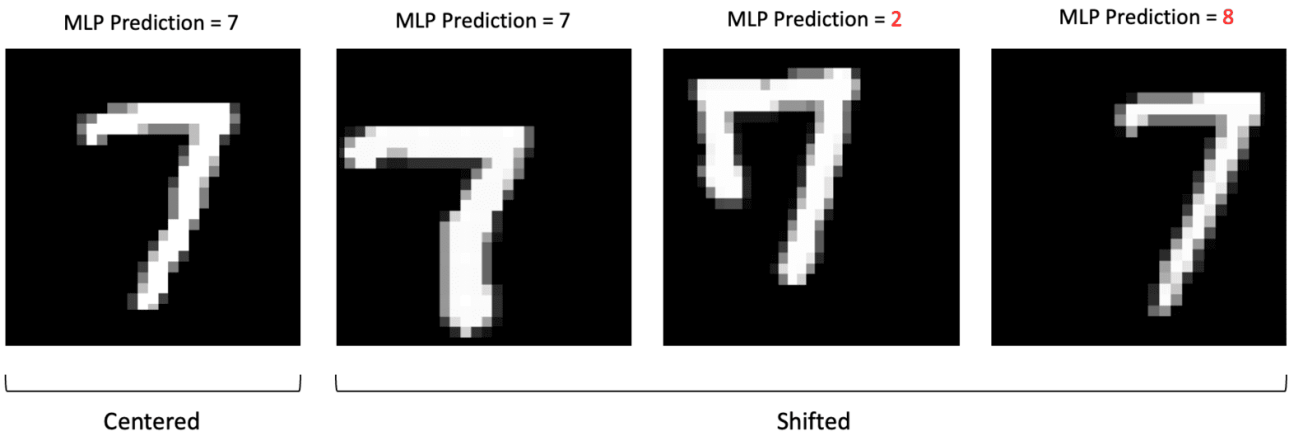


Figure: Not translation invariant

Limitations of CNN

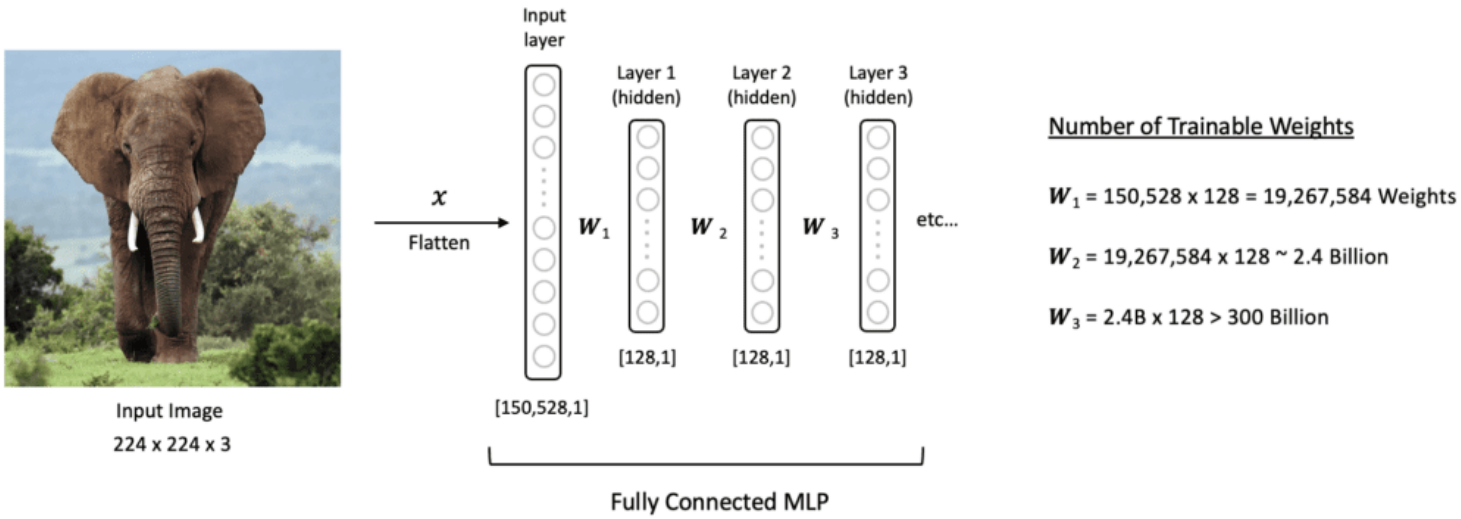
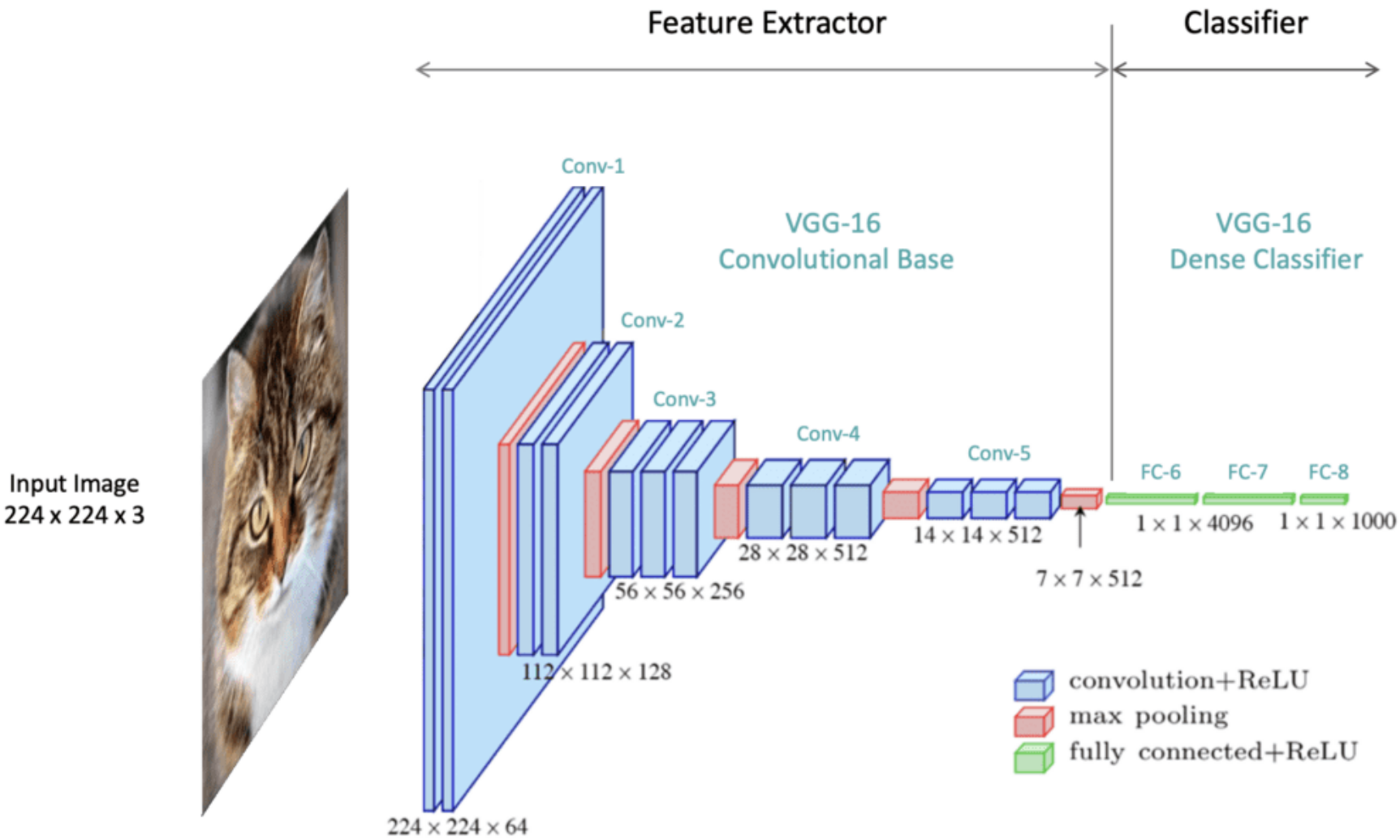


Figure: Prone to Overfitting

vgg16 architecture



The Convolution Operation in Convolutional Neural Networks

Research Presentation

March 27, 2025

The Convolution Operation: Basic Concept

Key Components

- Input: Two-dimensional data (e.g., 6x6 image)
- Filter (Kernel): Small matrix (e.g., 3x3)
- Purpose: Extract features from input data

Kernel Characteristics

- In image processing: Predefined filters (e.g., Sobel kernel)
- In CNNs: Kernel weights are **learned during training**
- Designed to detect specific features (e.g., vertical edges)

Example (Sobel Kernel for Vertical Edge Detection)

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Convolutional operations

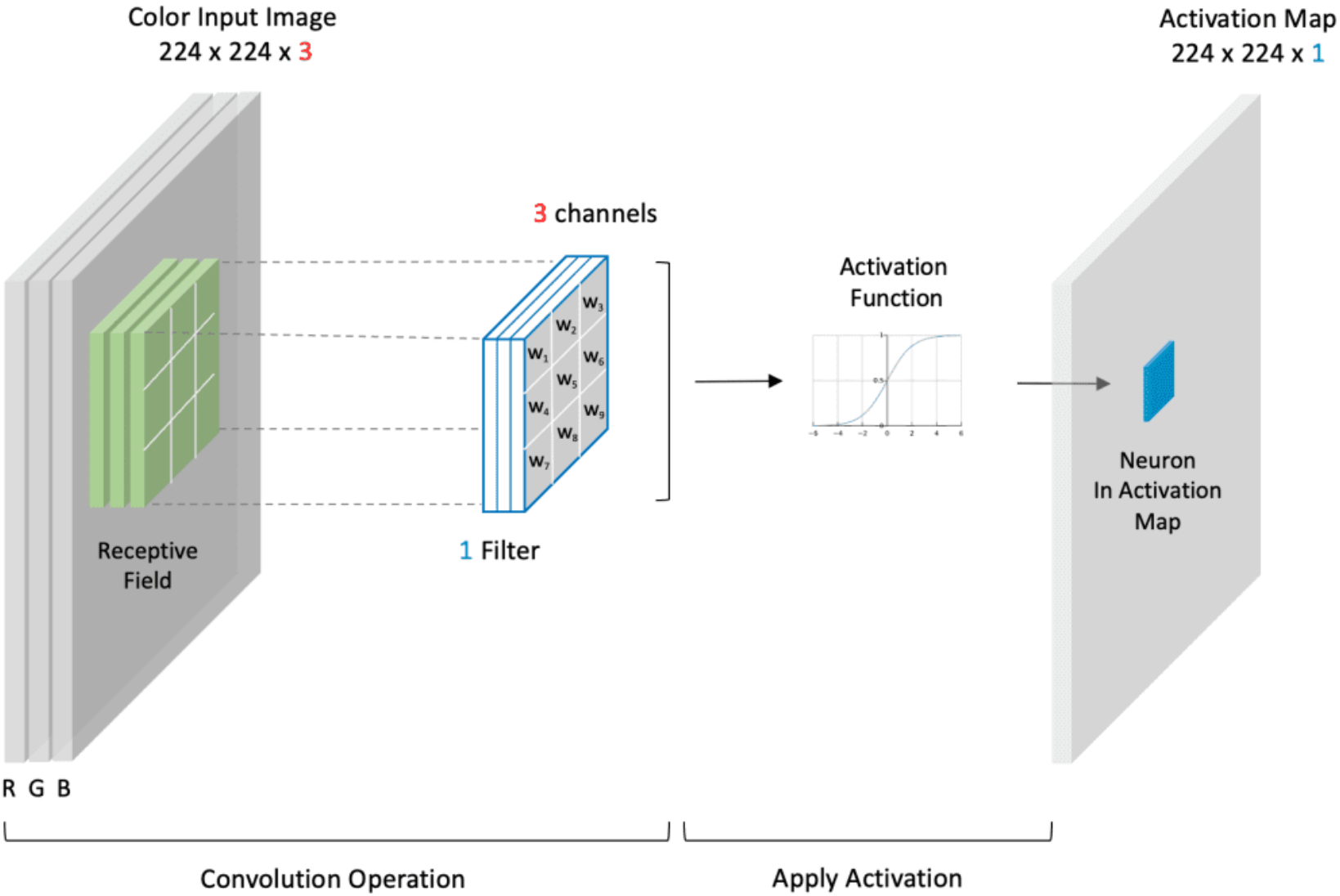


Figure: Convolution visual

Convolution Operation: Mechanism

Convolution Process

- Place kernel over a portion of the input
- Multiply kernel elements with corresponding input elements
- Compute dot product
- Produce a single output value
- Slide filter across entire input

Key Terminology

- **Receptive Field**: The region of input covered by the kernel
- Center of the kernel (highlighted in green)
- Dark blue indicates current kernel position

Convolution Essence

Dot product of kernel values with corresponding input values at each location

Stride: Control of Feature Extraction

Stride Definition

- Number of pixels the kernel shifts in each operation
- Stride of 1: Kernel moves one pixel at a time

Impact of Stride

- Lower Stride (e.g., 1):
  - More feature extraction
  - Larger output layers
  - More detailed feature learning
- Higher Stride:
  - Reduced output layer dimensions
  - Less computational complexity
  - Fewer extracted features

Example (Stride Visualization)

Stride 1: Kernel moves 1 pixel at a time

Padding Benefits

- Conserves border information
- Maintains input spatial dimensions
- Improves model performance
- Controls output feature map size

Padding Types

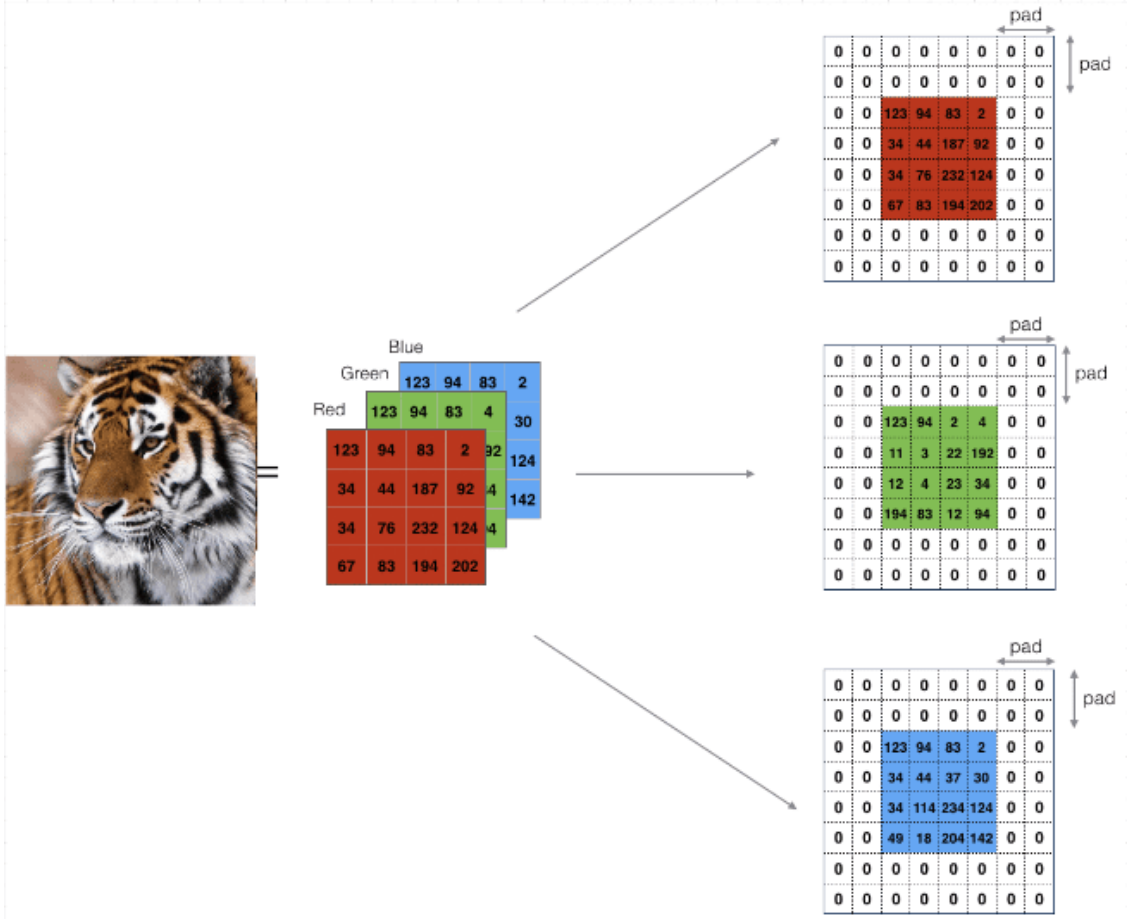
- Zero Padding: Add zeros around borders
- Reflection Padding: Reflect border pixels
- Replication Padding: Repeat border pixels

Key Insight

Padding allows flexible control of feature map dimensions and information preservation

Padding Definition

- Adding extra pixels around input image borders
- Helps preserve spatial dimensions



widthheightfigure

Caption

Conclusion: Convolution Operation Essentials

Key Takeaways

- Convolution is a fundamental operation in CNNs
- Kernel learns to extract meaningful features
- Stride and padding provide flexibility in feature extraction
- Critical for understanding how CNNs process image data

Convolution Output Spatial Size Calculation

Output Size Formula

$$O = \left\lfloor \frac{n - f + 2p}{s} \right\rfloor + 1$$

Formula Parameters

- O: Output size (height or width)
- n: Input size (height or width)
- f: Kernel (filter) size
- p: Padding
- s: Stride

Example Calculation

- Input size: 32×32
- Kernel size: 3×3
- Padding: 1 (same padding)

Sobel Kernel: Vertical Edge Detection

Sobel Kernel Structure

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Edge Detection Mechanism

- Positive values on left column
- Zeros in center column
- Negative values on right column

Derivative Approximation

- Kernel slides across image
- Convolution operation calculates weighted sum
- Produces numerical approximation of horizontal derivative
- Effectively detects vertical edges

Sobel Kernel
(3 x 3)

1	0	-1
2	0	-2
1	0	-1

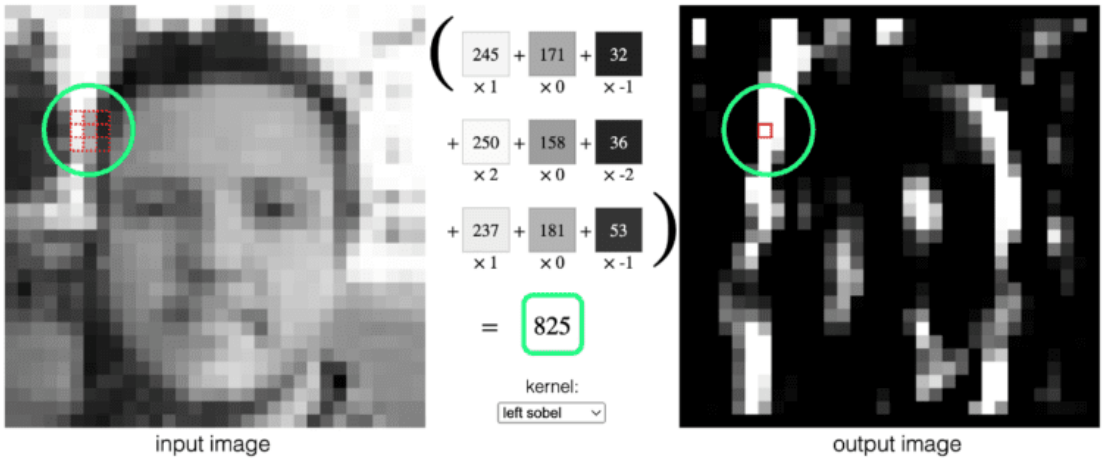


Figure: Caption

Convolutional Layer: Key Properties

Filter-Input Compatibility

- Filter depth must match input data depth
- Channels in filter = Channels in input

Filter Spatial Size

- Typical sizes: 3x3 or 5x5
- Size is a design choice
- Impacts feature extraction capability

Number of Filters

- Design choice determines output complexity
- Controls number of activation maps
- More filters = more feature extraction

Activation Maps and Kernels

Activation Map Terminology

- Multiple activation maps = Multi-channel activation map
- Each channel can be considered a separate activation map

Filter and Kernel Relationship

- Filter: Container for kernels
- Single-channel filter = Filter is the kernel
- Multi-channel filter contains multiple kernels

Weight Initialization and Learning

- Initial weights: Small random values
- Weights are learned during network training
- Enables adaptive feature detection

Trainable Parameters in Convolutional Layers

Parameter Calculation Factors

- Number of filters
- Filter size
- Input channel count

Example (Parameter Calculation)

- Input: 32x32x3 image
- Filter: 3x3 with 16 filters
- Calculation: $(3 \times 3 \times 3 + 1) \times 16$
- Bias term added for each filter

Computational Complexity

Parameter count impacts model complexity, training time, and potential overfitting

Convolutional Layer: Single Filter Example

Input Image Specifications

- Input Size: $224 \times 224 \times 3$
- Color Image (RGB Channels)
- Filter Size: $3 \times 3 \times 3$

Filter Composition

- Single Filter with 3 Kernels
- Each Kernel: 9 Trainable Weights
- Total Trainable Weights: 27
- Bias Term: 1
- Total Trainable Parameters: 28

Convolution Process

- Convolution performed separately for each channel
- Weighted sum across channels

Multiple Filters and Channel Dynamics

Key Channel Principles

- Filter Channels Must Match Input Channels
- Number of Filters Determines Output Depth
- Each Filter Produces One Activation Map

Practical Example: 32-Filter Layer

- Input: $224 \times 224 \times 3$
- 32 Filters of Size $3 \times 3 \times 3$
- Output: $224 \times 224 \times 32$
- 32 Activation Maps Produced

Layer Connectivity

- Subsequent Layer Filters Must Match Previous Layer's Depth
- Filter Count is a Design Choice
- Enables Hierarchical Feature Extraction

Max Pooling Layers

Purpose of Pooling

- Reduce Spatial Dimensions
- Summarize Activation Map Features
- Decrease Network Parameters
- Mitigate Overfitting

Pooling Characteristics

- 2D Sliding Filter
- Configurable Stride
- No Trainable Parameters
- Two Common Types: Max Pooling, Average Pooling

Max Pooling Operation

- Slide Filter Over Input
- Select Maximum Value in Filter Window

Convolutional Block Structure

Typical Block Composition

- 2-3 Consecutive Convolutional Layers
- Minimum 32 Filters per Layer
- Followed by Max Pooling Layer
- Reduces Spatial Dimensions

VGG-16 Block Example

- Input: Color Image ($224 \times 224 \times 3$)
- Convolutional Layers
- Max Pooling Layer
- Repeating Block Structure

Block Purpose

- Hierarchical Feature Extraction
- Progressively Abstract Representations

Conclusion: CNN Layer Dynamics

Key Takeaways

- Convolutional Layers Extract Hierarchical Features
- Filter Design Crucial for Performance
- Pooling Layers Reduce Spatial Complexity
- Network Architecture Impacts Learning

Design Considerations

- Balance Between Feature Extraction and Computational Efficiency
- Careful Selection of Filters and Pooling Parameters

Feature Map Characteristics

- Final convolutional layer contains meaningful image information
- Each spatial location relates to original input image
- Preserves spatial relationships during processing

Fully Connected Classifier

- Allows processing content from entire image
- Flattens activation maps from last convolutional layer
- Maintains original spatial interpretation

Flattening Note

Flattening is a data repacking method

- Does not alter spatial data interpretation
- Prepares data for classifier processing

Feature Map Processing Mechanism

Trained Model Scenario

- Assume all network weights are fixed
- Consider different input images (e.g., car, elephant)
- Feature maps differ based on input content

Classification Process

- Feature maps processed through classifier
- Combine feature maps with trained weights
- Produce higher activations for specific output neurons

Connection Complexity

- Over 100 Million connections between flattened maps and first fully connected layer
- Weights remain constant across different inputs
- Activation patterns vary based on input

Conceptual Visualization

Network Processing

- Same trained network weights
- Different input images
- Varied activation patterns

Activation Pathways

- Dotted lines represent high-weight connections
- Indicate pathways to highest output neuron activations
- Demonstrate network's learned feature recognition

Learning Mechanism

- Network learns to recognize distinctive features
- Weights capture complex image characteristics
- Enables accurate image classification

Conclusion: CNN Working Principles

Key Takeaways

- Spatial information is crucial in CNNs
- Fully connected layers leverage learned features
- Network weights enable sophisticated image understanding

Fundamental Concept

CNNs transform raw image data into meaningful, hierarchical representations