Image Feature Extraction and Classification Using CNN Architectures

Dr. Meghna Kapoor

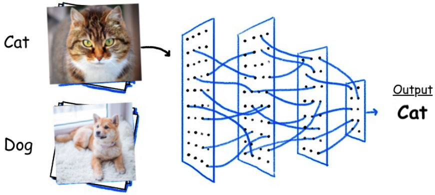
Learning Objective

- How features plays an important role in classification
- Implementation of different architectures for image classification

What is Image Classification

- Input is an image
- Output is a class label
- Example: cat, dog, car, person

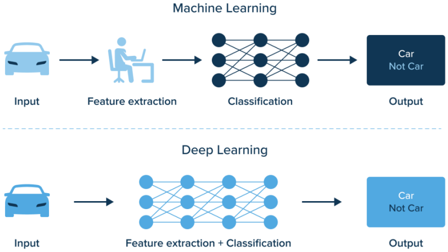
CNNs automatically learn useful patterns from images.



Why Do We Need CNNs

- Images have many pixels
- Manually designing features is difficult
- CNNs learn features directly from data

CNNs replace hand crafted features with learned features.



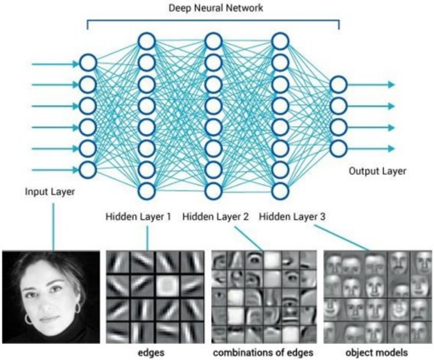
Two Main Parts of a CNN

1. **Feature Extraction**
   - Convolution layers
   - Pooling layers
2. **Classification**
   - Fully connected layers
   - Softmax output

How Feature Extraction Works

- First layers detect edges
- Middle layers detect shapes
- Deeper layers detect objects

Feature learning becomes more meaningful with depth.



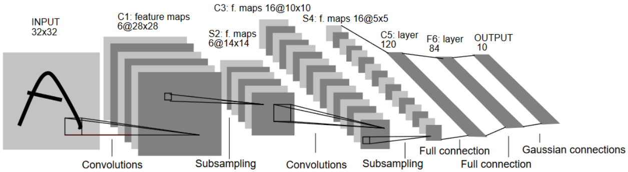
Why Count Parameters

- Parameters represent what the model learns
- More parameters means more memory usage
- More parameters increase training and inference time

Understanding parameter count helps choose the right model.

LeNet

- One of the first CNNs
- Few convolution layers
- Used for digit recognition



Simple model for simple images.

LeNet Parameters

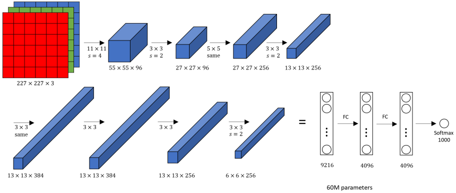
- Small filters
- Few channels
- Very low parameter count

Layer Name	Input W×H×D	Kernel W×H×D/S	Output W×H×D	Params	Mults
C1: conv2d	32×32×1	5×5×6	28×28×6	1×5×5×6+6 =156	28×28×1×5×5×6 =117,600
S2: pool/2	28×28×6	2×2/2	14×14×6	0	0
C3: conv2d	14×14×6	5×5×16	10×10×16	6×5×5×16+16 =2,416	10×10×6×5×5×16 =240,000
S4: pool/2	10×10×16	2×2/2	5×5×16	0	0
C5: conv2d	5×5×16	5×5×120	1×1×120	16×5×5×120+120 =48,120	1×1×16×5×5×120 =48,000
F6: conv2d	1×1×120	1×1×84	1×1×84	120×1×1×84+84 =10,164	120×84 =10,080
F7: conv2d	1×1×84	1×1×10	1×1×10	84×1×1×10+10 =850	84×40 =840
			Total	61,706	416,520

LeNet is lightweight and easy to train.

AlexNet

- Deeper than LeNet
- Works on larger images
- Uses ReLU and dropout



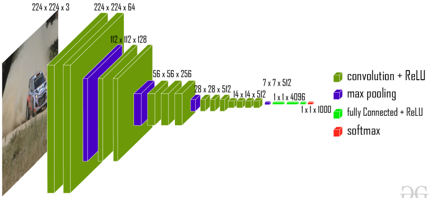
Showed that deep CNNs work well.

AlexNet Parameters

	Activation shape	Activation size	# parameters
Input image	227 x 227 x 3	154587	0
Conv 1	55 x 55 x 96 (f=11 s = 4 p = 0)	290400	34944
Pool 1	27 x 27 x 96 (f=3 s = 2)	69984	0
Conv 2	27 x 27 x 256 (f=5 s = 1 p = 2)	186624	614,656
Pool 2	13 x 13 x 256 (f=3 s = 2)	43264	0
Conv 3	13 x 13 x 384 (f=3 s = 1 p = 1)	64896	885,120
Conv 4	13 x 13 x 384 (f=3 s = 1 p = 1)	64896	1,327,488
Conv 5	13 x 13 x 256 (f=3 s = 1 p = 1)	43264	884,992
Pool 5	6 x 6 x 256 (f=3 s = 2)	9216	0
FC 3	4096 x 1	4096	37,748,737
FC 4	4096 x 1	4096	16,777,217
Softmax	1000 x 1	1000	4096001

VGGNet

- Many layers
- Uses only small 3×3 filters
- Easy to understand structure



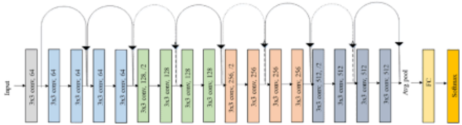
Deeper networks learn better features.

Problem with Very Deep Networks

- Training becomes difficult
- Gradients can vanish
- Accuracy may stop improving

ResNet

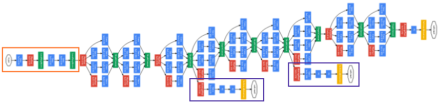
- Uses shortcut connections
- Skips some layers
- Makes training easier



Allows very deep CNNs.

InceptionNet

- Uses multiple filter sizes together
- Looks at image at different scales
- Efficient and accurate

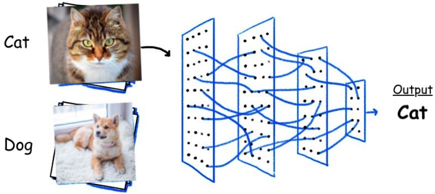


GoogLeNet. The orange box is the **stem**, which has some preliminary convolutions. The purple boxes are **auxiliary classifiers**. The wide parts are the inception modules. (Source: Inception v1)

Combines multiple views of the image.

Dog Cat classification

- Binary image classification problem
- Real-world dataset with large visual variability
- More complex than digit recognition
- Tests generalization capability of CNNs



Problem Statement

- Input: RGB images of dogs and cats
- Output: Class label (Dog or Cat)
- Dataset organized using folder structure
- Supervised learning setup

Dataset Organization

- Images stored in Folder
- Two folders: Train and Test
- Each folder contains two classes
- Used ImageFolder for automatic labeling

CNN Architectures Used

- LeNet-5
- AlexNet
- VGG16
- ResNet18
- Inception Network

Why Compare Multiple Architectures

- Different depth and complexity
- Different feature learning capacity
- Different computational cost
- Observe performance trends

Preprocessing and Augmentation

- Images resized to required input size
- Normalization applied
- Data augmentation used only during training
- Improves robustness and generalization

Training Setup

- Loss function: Cross-Entropy Loss
- Optimizer: Adam
- Batch size fixed across models
- Same train-test split for fair comparison

Evaluation Metrics: Mathematical Definitions

Let:

- TP = True Positives
- TN = True Negatives
- FP = False Positives
- FN = False Negatives

- **Accuracy**

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Precision**

$$\text{Precision} = \frac{TP}{TP + FP}$$

- **Recall**

$$\text{Recall} = \frac{TP}{TP + FN}$$

- **F1-score**

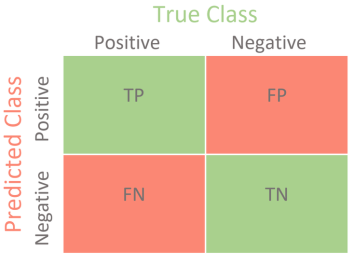
$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Accuracy Comparison

- LeNet shows lowest accuracy
- AlexNet improves feature learning
- VGG16 benefits from deeper structure
- ResNet18 converges faster
- Inception achieves strong generalization

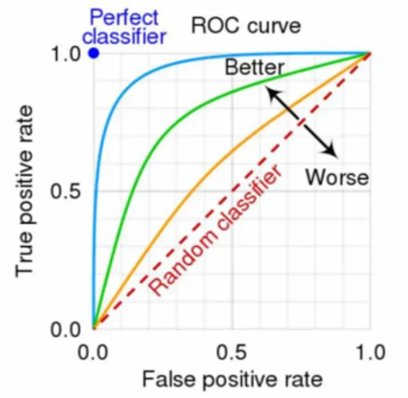
Confusion Matrix Analysis

- Shows class-wise prediction errors
- Highlights false positives and false negatives
- Helps understand model bias
- Deeper models reduce misclassification



ROC Curve and AUC

- Threshold independent evaluation
- Measures discrimination ability
- Higher AUC indicates stronger classifier
- Inception and ResNet show higher AUC



Precision, Recall, and F1 Score

- Precision measures prediction reliability
- Recall measures detection capability
- F1 score balances both
- Important when class imbalance exists

Qualitative Results

- Visual inspection of predictions
- True label vs predicted label displayed
- Confidence score indicates certainty
- Misclassifications analyzed visually

Key Observations

- Deeper models learn richer features
- Residual connections improve stability
- Inception captures multi-scale patterns
- Model complexity improves performance

Limitations

- Training time increases with depth
- Large models require more memory
- Small dataset may cause overfitting

Conclusion

- CNNs outperform traditional methods
- Architecture choice impacts accuracy
- Deeper and multi-scale models perform better