Computer Vision Ultimate Cheat Sheet

From OpenCV to Deep Learning: Concepts & Code

1 Image Fundamentals

Digital Images

- **Pixel**: Smallest unit, value 0-255.
- **Grayscale**: 2D Array $(H \times W)$.
- **RGB**: 3D Array $(H \times W \times 3)$ (Channels).
- **Resolution**: Dimensions (e.g., 1920×1080).

OpenCV Basics (cv2)

```
import cv2
# Read (Loads as BGR!)
img = cv2.imread("image.jpg")

# Convert BGR to RGB
img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

# Resize
resized = cv2.resize(img, (224, 224))

# Display
cv2.imshow("Title", img)
cv2.waitKey(0)
```

2 Image Processing

Filtering & Smoothing *Remove noise using kernels.*

```
# Gaussian Blur (Smooths image)
blur = cv2.GaussianBlur(img, (5,5), 0)
```

Edge Detection (Canny) *Finds intensity gradients.*

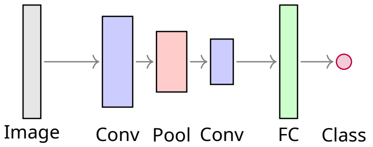
```
edges = cv2.Canny(img, 100, 200)
```

Thresholding *Binarize image (Black/White).*

```
_, binary = cv2.threshold(gray, 127, 255,
                          cv2.THRESH_BINARY)
```

3 Deep Learning: CNNs

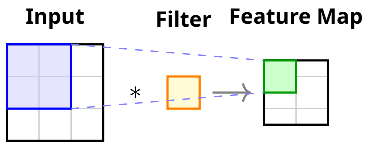
Convolutional Neural Networks Specialized for grid-like data (images).



Key Layers

- **Convolution**: Extracts features using learnable kernels/filters. Preserves spatial relationship.
- **ReLU**: Activation function $\max(0, x)$. Adds non-linearity.
- **Pooling (Max/Avg)**: Downsamples feature maps. Reduces parameters and controls overfitting.
- **Fully Connected (FC)**: Classifier at the end.

The Convolution Operation



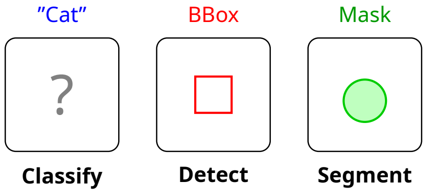
PyTorch CNN Example

```
import torch.nn as nn

class SimpleCNN(nn.Module):
    def __init__(self):
        super().__init__()
        self.conv1 = nn.Conv2d(3, 16, 3) # In, Out, K
        self.pool = nn.MaxPool2d(2, 2)
        self.fc = nn.Linear(16 * 13 * 13, 10)

    def forward(self, x):
        x = self.pool(F.relu(self.conv1(x)))
        x = torch.flatten(x, 1)
        return self.fc(x)
```

4 Core CV Tasks



1. Classification *What is in the image?*

- Output: Class Label (e.g., "Cat").
- Loss: Cross-Entropy.
- Models: ResNet, VGG, EfficientNet.

2. Object Detection *Where are the objects?*

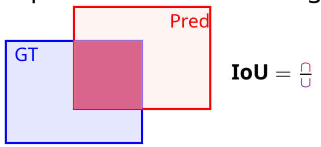
- Output: Bounding Box (x, y, w, h) + Class.
- **YOLO (You Only Look Once)**: Fast, single-stage.
- **R-CNN / Faster R-CNN**: Two-stage, accurate.

3. Segmentation *Pixel-level classification.*

- **Semantic**: All cats are same color.
- **Instance**: Each cat is a different color.
- Models: U-Net (Biomedical), Mask R-CNN.

5 Evaluation Metrics

IoU (Intersection over Union) Measures overlap between predicted box and ground truth.



mAP (Mean Average Precision) Standard metric for detection. Area under the Precision-Recall curve, averaged over all classes.
Confusion Matrix For classification (TP, FP, TN, FN).

6 Data Augmentation

Artificially increase dataset size to prevent overfitting.
Techniques

- **Geometric**: Flip, Rotate, Crop, Zoom.
- **Color**: Brightness, Contrast, Saturation.
- **Noise**: Gaussian noise, Blur.

```
from torchvision import transforms

transform = transforms.Compose([
    transforms.RandomHorizontalFlip(),
    transforms.RandomRotation(10),
    transforms.ToTensor()
])
```

7 Modern Architectures

ResNet (Residual Networks) Introduced **Skip Connections** to train very deep networks (prevents vanishing gradient).

MobileNet Optimized for mobile/edge. Uses **Depthwise Separable Convolutions**.

Vision Transformers (ViT) Splits image into patches and processes them like words in NLP. State-of-the-art on large datasets.