```
In [ ]: # This Python 3 environment comes with many helpful analytics libraries instal
        # It is defined by the kaggle/python Docker image: https://github.com/kaggle/d
        # For example, here's several helpful packages to load

        import numpy as np # linear algebra
        import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

        # Input data files are available in the read-only "../input/" directory
        # For example, running this (by clicking run or pressing Shift+Enter) will lis

        import os
        for dirname, _, filenames in os.walk('/kaggle/input'):
            for filename in filenames:
                print(os.path.join(dirname, filename))

        # You can write up to 20GB to the current directory (/kaggle/working/) that ge
        # You can also write temporary files to /kaggle/temp/, but they won't be saved
```

## Download Caltech256 Dataset

```
In [2]: !pip install kagglehub

        import kagglehub

        # Download the dataset using kagglehub
        downloaded_dataset_root = kagglehub.dataset_download("jessicali9530/caltech256

        print(f"Dataset downloaded and extracted to: {downloaded_dataset_root}")
```

```
Requirement already satisfied: kagglehub in /usr/local/lib/python3.11/dist-pack
ages (0.3.13)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-pack
ages (from kagglehub) (25.0)
Requirement already satisfied: pyyaml in /usr/local/lib/python3.11/dist-package
s (from kagglehub) (6.0.3)
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packa
ges (from kagglehub) (2.32.5)
Requirement already satisfied: tqdm in /usr/local/lib/python3.11/dist-packages
(from kagglehub) (4.67.1)
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/pytho
n3.11/dist-packages (from requests->kagglehub) (3.4.4)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-p
ackages (from requests->kagglehub) (3.11)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/
dist-packages (from requests->kagglehub) (2.5.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/
dist-packages (from requests->kagglehub) (2025.10.5)
Dataset downloaded and extracted to: /kaggle/input/caltech256
```

## Split Dataset into Train, Validation, and Test Sets

In [3]:
```python
import os
import shutil
import random

downloaded_dataset_root = "/kaggle/input/caltech256"

root = os.path.join(downloaded_dataset_root, "256_ObjectCategories")

target_dir = "caltech256_split"

classes = sorted(os.listdir(root))

os.makedirs(target_dir, exist_ok=True)
os.makedirs(f"{target_dir}/train", exist_ok=True)
os.makedirs(f"{target_dir}/val", exist_ok=True)
os.makedirs(f"{target_dir}/test", exist_ok=True)

for c in classes:
    src = os.path.join(root, c)
    if not os.path.isdir(src):
        continue

    train_path = f"{target_dir}/train/{c}"
    val_path = f"{target_dir}/val/{c}"
    test_path = f"{target_dir}/test/{c}"

    os.makedirs(train_path, exist_ok=True)
    os.makedirs(val_path, exist_ok=True)
    os.makedirs(test_path, exist_ok=True)

    images = os.listdir(src)
    random.shuffle(images)

    n = len(images)
    train_split = int(n * 0.7)
    val_split = int(n * 0.15)

    for i, img in enumerate(images):
        src_path = os.path.join(src, img)
        if os.path.isfile(src_path):
            if i < train_split:
                shutil.copy(src_path, train_path)
            elif i < train_split + val_split:
                shutil.copy(src_path, val_path)
            else:
                shutil.copy(src_path, test_path)

print("Dataset split completed.")
```

```
Dataset split completed.
```

## Count Images in Train / Val / Test Folders

In [4]:
```
import os

target_dir = "caltech256_split"

# Initialize dictionaries to store counts
train_counts = {}
val_counts = {}
test_counts = {}

# Get the list of classes (subdirectories) in the train, val, test folders
classes = os.listdir(os.path.join(target_dir, 'train'))

for c in classes:
    train_path = os.path.join(target_dir, 'train', c)
    val_path = os.path.join(target_dir, 'val', c)
    test_path = os.path.join(target_dir, 'test', c)

    if os.path.isdir(train_path):
        train_counts[c] = len(os.listdir(train_path))
    if os.path.isdir(val_path):
        val_counts[c] = len(os.listdir(val_path))
    if os.path.isdir(test_path):
        test_counts[c] = len(os.listdir(test_path))

print("--- Training Split Counts ---")
for c, count in train_counts.items():
    print(f"Class {c}: {count} images")
print(f"Total Training Images: {sum(train_counts.values())}")

print("\n--- Validation Split Counts ---")
for c, count in val_counts.items():
    print(f"Class {c}: {count} images")
print(f"Total Validation Images: {sum(val_counts.values())}")

print("\n--- Test Split Counts ---")
for c, count in test_counts.items():
    print(f"Class {c}: {count} images")
print(f"Total Test Images: {sum(test_counts.values())}")
import os
import shutil
import random

# Correctly set the root to the directory containing the actual class folders
# Assuming '256_ObjectCategories' is the main folder inside the downloaded dat
dataset_base_path = os.path.join(downloaded_dataset_root, '256_ObjectCategorie
root = dataset_base_path
```

```python
target_dir = "caltech256_split"

classes = sorted(os.listdir(root))

os.makedirs(target_dir, exist_ok=True)
os.makedirs(f"{target_dir}/train", exist_ok=True)
os.makedirs(f"{target_dir}/val", exist_ok=True)
os.makedirs(f"{target_dir}/test", exist_ok=True)

for c in classes:
    src = os.path.join(root, c)
    if not os.path.isdir(src):
        continue

    train_path = f"{target_dir}/train/{c}"
    val_path = f"{target_dir}/val/{c}"
    test_path = f"{target_dir}/test/{c}"

    os.makedirs(train_path, exist_ok=True)
    os.makedirs(val_path, exist_ok=True)
    os.makedirs(test_path, exist_ok=True)

    images = os.listdir(src)
    random.shuffle(images)

    n = len(images)
    train_split = int(n * 0.7)
    val_split = int(n * 0.15)

    for i, img in enumerate(images):
        src_path = os.path.join(src, img)
        if os.path.isfile(src_path): # Added check to ensure it's a file
            if i < train_split:
                shutil.copy(src_path, train_path)
            elif i < train_split + val_split:
                shutil.copy(src_path, val_path)
            else:
                shutil.copy(src_path, test_path)

print("Dataset split completed.")
```

```
--- Training Split Counts ---
Class 104.homer-simpson: 67 images
Class 055.dice: 68 images
Class 058.doorknob: 65 images
Class 062.eiffel-tower: 58 images
Class 119.jesus-christ: 60 images
Class 131.lightbulb: 64 images
Class 111.house-fly: 58 images
Class 257.clutter: 578 images
Class 002.american-flag: 67 images
Class 169.radio-telescope: 64 images
Class 184.sheet-music: 58 images
Class 075.floppy-disk: 58 images
Class 140.menorah-101: 62 images
Class 130.license-plate: 63 images
Class 237.vcr: 62 images
Class 241.waterfall: 66 images
Class 024.butterfly: 78 images
Class 071.fire-hydrant: 69 images
Class 004.baseball-bat: 88 images
Class 089.goose: 77 images
Class 210.syringe: 77 images
Class 164.porcupine: 70 images
Class 157.pci-card: 73 images
Class 229.tricycle: 66 images
Class 138.mattress: 134 images
Class 149.necktie: 72 images
Class 170.rainbow: 71 images
Class 126.ladder: 169 images
Class 070.fire-extinguisher: 58 images
Class 247.xylophone: 64 images
Class 086.golden-gate-bridge: 56 images
Class 135.mailbox: 65 images
Class 217.tennis-court: 73 images
Class 235.umbrella-101: 79 images
Class 224.touring-bike: 77 images
Class 001.ak47: 68 images
Class 176.saddle: 77 images
Class 118.iris: 75 images
Class 015.bonsai-101: 85 images
Class 014.blimp: 60 images
Class 032.cartman: 70 images
Class 127.laptop-101: 89 images
Class 151.ostrich: 76 images
Class 139.megaphone: 60 images
Class 179.scorpion-101: 56 images
Class 128.lathe: 73 images
Class 017.bowling-ball: 72 images
Class 057.dolphin-101: 74 images
Class 029.cannon: 72 images
Class 067.eyeglasses: 58 images
Class 061.dumb-bell: 71 images
Class 204.sunflower-101: 56 images
Class 143.minaret: 91 images
Class 234.tweezer: 85 images
Class 102.helicopter-101: 61 images
Class 137.mars: 109 images
Class 098.harp: 70 images
Class 232.t-shirt: 250 images
Class 202.steering-wheel: 67 images
Class 087.goldfish: 65 images
Class 003.backpack: 105 images
Class 183.sextant: 70 images
Class 161.photocopier: 72 images
Class 256.toad: 75 images
Class 080.frog: 81 images
Class 009.bear: 71 images
Class 245.windmill: 63 images
Class 251.airplanes-101: 560 images
Class 148.mussels: 121 images
Class 115.ice-cream-cone: 61 images
Class 013.birdbath: 68 images
Class 254.greyhound: 66 images
Class 018.bowling-pin: 70 images
Class 096.hammock: 199 images
Class 153.palm-pilot: 65 images
Class 090.gorilla: 148 images
Class 103.hibiscus: 77 images
Class 236.unicorn: 67 images
Class 011.billiards: 194 images
Class 099.harpsichord: 56 images
Class 207.swan: 80 images
Class 054.diamond-ring: 82 images
Class 129.leopards-101: 133 images
Class 097.harmonica: 62 images
Class 066.ewer-101: 58 images
Class 145.motorbikes-101: 558 images
Class 056.dog: 71 images
Class 201.starfish-101: 56 images
Class 031.car-tire: 62 images
Class 027.calculator: 70 images
Class 175.roulette-wheel: 58 images
Class 012.binoculars: 151 images
Class 186.skunk: 56 images
Class 230.trilobite-101: 65 images
Class 134.llama-101: 83 images
Class 019.boxing-glove: 86 images
Class 052.crab-101: 59 images
Class 226.traffic-light: 69 images
Class 218.tennis-racket: 56 images
Class 160.pez-dispenser: 58 images
Class 073.fireworks: 70 images
Class 240.watch-101: 140 images
Class 122.kayak: 72 images
Class 010.beer-mug: 65 images
Class 227.treadmill: 102 images
Class 155.paperclip: 64 images
Class 163.playing-card: 62 images
Class 078.fried-egg: 62 images
Class 191.sneaker: 77 images
Class 167.pyramid: 60 images
Class 043.coin: 86 images
Class 044.comet: 84 images
Class 165.pram: 61 images
Class 250.zebra: 67 images
Class 093.grasshopper: 78 images
Class 152.owl: 84 images
Class 121.kangaroo-101: 57 images
Class 132.light-house: 133 images
Class 205.superman: 60 images
Class 188.smokestack: 61 images
Class 242.watermelon: 65 images
Class 033.cd: 71 images
Class 034.centipede: 70 images
Class 106.horseshoe-crab: 60 images
Class 177.saturn: 67 images
Class 059.drinking-straw: 58 images
Class 162.picnic-table: 63 images
Class 252.car-side-101: 81 images
Class 219.theodolite: 58 images
Class 100.hawksbill-101: 65 images
Class 117.ipod: 84 images
Class 220.toaster: 65 images
Class 253.faces-easy-101: 304 images
Class 123.ketch-101: 77 images
Class 194.socks: 78 images
Class 041.coffee-mug: 60 images
Class 133.lightning: 95 images
Class 082.galaxy: 56 images
Class 244.wheelbarrow: 63 images
Class 047.computer-mouse: 65 images
Class 021.breadmaker: 99 images
Class 005.baseball-glove: 103 images
Class 030.canoe: 72 images
Class 144.minotaur: 57 images
Class 195.soda-can: 60 images
Class 180.screwdriver: 71 images
Class 069.fighter-jet: 69 images
Class 214.teepee: 97 images
Class 113.hummingbird: 81 images
Class 182.self-propelled-lawn-mower: 84 images
Class 038.chimp: 77 images
Class 040.cockroach: 86 images
Class 178.school-bus: 68 images
Class 023.bulldozer: 77 images
Class 248.yarmulke: 58 images
Class 063.electric-guitar-101: 85 images
Class 101.head-phones: 96 images
Class 108.hot-dog: 59 images
Class 008.bathtub: 162 images
Class 051.cowboy-hat: 79 images
Class 083.gas-pump: 66 images
Class 249.yo-yo: 70 images
Class 185.skateboard: 72 images
Class 141.microscope: 81 images
Class 042.coffin: 60 images
Class 116.iguana: 74 images
Class 084.giraffe: 58 images
Class 208.swiss-army-knife: 76 images
Class 036.chandelier-101: 74 images
Class 216.tennis-ball: 68 images
Class 016.boom-box: 63 images
Class 187.skyscraper: 66 images
Class 068.fern: 77 images
Class 085.goat: 78 images
Class 022.buddha-101: 67 images
Class 072.fire-truck: 82 images
Class 142.microwave: 74 images
Class 156.paper-shredder: 67 images
Class 074.flashlight: 80 images
Class 238.video-projector: 67 images
Class 049.cormorant: 74 images
Class 028.camel: 77 images
Class 120.joy-stick: 91 images
Class 112.human-skeleton: 58 images
Class 190.snake: 78 images
Class 105.horse: 189 images
Class 091.grand-piano-101: 66 images
Class 109.hot-tub: 109 images
Class 255.tennis-shoes: 72 images
Class 231.tripod: 78 images
Class 006.basketball-hoop: 62 images
Class 039.chopsticks: 59 images
Class 064.elephant-101: 91 images
Class 136.mandolin: 65 images
Class 166.praying-mantis: 64 images
Class 065.elk: 70 images
Class 225.tower-pisa: 62 images
Class 212.teapot: 95 images
Class 200.stained-glass: 70 images
Class 050.covered-wagon: 67 images
Class 079.frisbee: 69 images
Class 173.rifle: 74 images
Class 198.spider: 76 images
Class 007.bat: 74 images
Class 221.tomato: 72 images
Class 209.sword: 71 images
Class 035.cereal-box: 60 images
Class 088.golf-ball: 68 images
Class 215.telephone-box: 58 images
Class 081.frying-pan: 66 images
Class 206.sushi: 68 images
Class 124.killer-whale: 63 images
Class 189.snail: 83 images
Class 020.brain-101: 58 images
Class 046.computer-monitor: 93 images
Class 092.grapes: 140 images
Class 172.revolver-101: 69 images
Class 174.rotary-phone: 58 images
Class 192.snowmobile: 78 images
Class 060.duck: 60 images
Class 037.chess-board: 84 images
Class 026.cake: 74 images
Class 168.raccoon: 98 images
Class 199.spoon: 73 images
Class 203.stirrups: 63 images
Class 125.knife: 70 images
Class 223.top-hat: 56 images
Class 171.refrigerator: 58 images
Class 107.hot-air-balloon: 62 images
Class 243.welding-mask: 62 images
Class 114.ibis-101: 84 images
Class 211.tambourine: 66 images
Class 150.octopus: 77 images
Class 197.speed-boat: 70 images
Class 239.washing-machine: 58 images
Class 196.spaghetti: 72 images
Class 094.guitar-pick: 72 images
Class 159.people: 146 images
Class 246.wine-bottle: 70 images
Class 158.penguin: 104 images
Class 193.soccer-ball: 121 images
Class 233.tuning-fork: 70 images
Class 095.hamburger: 60 images
Class 154.palm-tree: 72 images
Class 213.teddy-bear: 70 images
Class 045.computer-keyboard: 59 images
Class 147.mushroom: 141 images
Class 181.segway: 70 images
Class 110.hourglass: 59 images
Class 053.desk-globe: 57 images
Class 222.tombstone: 63 images
Class 025.cactus: 79 images
Class 048.conch: 72 images
Class 076.football-helmet: 58 images
Class 077.french-horn: 64 images
Class 146.mountain-bike: 57 images
Class 228.triceratops: 66 images
Total Training Images: 21308

--- Validation Split Counts ---
Class 104.homer-simpson: 14 images
Class 055.dice: 14 images
Class 058.doorknob: 13 images
Class 062.eiffel-tower: 12 images
Class 119.jesus-christ: 13 images
Class 131.lightbulb: 13 images
Class 111.house-fly: 12 images
Class 257.clutter: 124 images
Class 002.american-flag: 14 images
Class 169.radio-telescope: 13 images
Class 184.sheet-music: 12 images
Class 075.floppy-disk: 12 images
Class 140.menorah-101: 13 images
Class 130.license-plate: 13 images
Class 237.vcr: 13 images
Class 241.waterfall: 14 images
Class 024.butterfly: 16 images
Class 071.fire-hydrant: 14 images
Class 004.baseball-bat: 19 images
Class 089.goose: 16 images
Class 210.syringe: 16 images
Class 164.porcupine: 15 images
Class 157.pci-card: 15 images
Class 229.tricycle: 14 images
Class 138.mattress: 28 images
Class 149.necktie: 15 images
Class 170.rainbow: 15 images
Class 126.ladder: 36 images
Class 070.fire-extinguisher: 12 images
Class 247.xylophone: 13 images
Class 086.golden-gate-bridge: 12 images
Class 135.mailbox: 13 images
Class 217.tennis-court: 15 images
Class 235.umbrella-101: 17 images
Class 224.touring-bike: 16 images
Class 001.ak47: 14 images
Class 176.saddle: 16 images
Class 118.iris: 16 images
Class 015.bonsai-101: 18 images
Class 014.blimp: 12 images
Class 032.cartman: 15 images
Class 127.laptop-101: 19 images
Class 151.ostrich: 16 images
Class 139.megaphone: 12 images
Class 179.scorpion-101: 12 images
Class 128.lathe: 15 images
Class 017.bowling-ball: 15 images
Class 057.dolphin-101: 15 images
Class 029.cannon: 15 images
Class 067.eyeglasses: 12 images
Class 061.dumb-bell: 15 images
Class 204.sunflower-101: 12 images
Class 143.minaret: 19 images
Class 234.tweezer: 18 images
Class 102.helicopter-101: 13 images
Class 137.mars: 23 images
Class 098.harp: 15 images
Class 232.t-shirt: 53 images
Class 202.steering-wheel: 14 images
Class 087.goldfish: 13 images
Class 003.backpack: 22 images
Class 183.sextant: 15 images
Class 161.photocopier: 15 images
Class 256.toad: 16 images
Class 080.frog: 17 images
Class 009.bear: 15 images
Class 245.windmill: 13 images
Class 251.airplanes-101: 120 images
Class 148.mussels: 26 images
Class 115.ice-cream-cone: 13 images
Class 013.birdbath: 14 images
Class 254.greyhound: 14 images
Class 018.bowling-pin: 15 images
Class 096.hammock: 42 images
Class 153.palm-pilot: 13 images
Class 090.gorilla: 31 images
Class 103.hibiscus: 16 images
Class 236.unicorn: 14 images
Class 011.billiards: 41 images
Class 099.harpsichord: 12 images
Class 207.swan: 17 images
Class 054.diamond-ring: 17 images
Class 129.leopards-101: 28 images
Class 097.harmonica: 13 images
Class 066.ewer-101: 12 images
Class 145.motorbikes-101: 119 images
Class 056.dog: 15 images
Class 201.starfish-101: 12 images
Class 031.car-tire: 13 images
Class 027.calculator: 15 images
Class 175.roulette-wheel: 12 images
Class 012.binoculars: 32 images
Class 186.skunk: 12 images
Class 230.trilobite-101: 14 images
Class 134.llama-101: 17 images
Class 019.boxing-glove: 18 images
Class 052.crab-101: 12 images
Class 226.traffic-light: 14 images
Class 218.tennis-racket: 12 images
Class 160.pez-dispenser: 12 images
Class 073.fireworks: 15 images
Class 240.watch-101: 30 images
Class 122.kayak: 15 images
Class 010.beer-mug: 14 images
Class 227.treadmill: 22 images
Class 155.paperclip: 13 images
Class 163.playing-card: 13 images
Class 078.fried-egg: 13 images
Class 191.sneaker: 16 images
Class 167.pyramid: 12 images
Class 043.coin: 18 images
Class 044.comet: 18 images
Class 165.pram: 13 images
Class 250.zebra: 14 images
Class 093.grasshopper: 16 images
Class 152.owl: 18 images
Class 121.kangaroo-101: 12 images
Class 132.light-house: 28 images
Class 205.superman: 13 images
Class 188.smokestack: 13 images
Class 242.watermelon: 13 images
Class 033.cd: 15 images
Class 034.centipede: 15 images
Class 106.horseshoe-crab: 13 images
Class 177.saturn: 14 images
Class 059.drinking-straw: 12 images
Class 162.picnic-table: 13 images
Class 252.car-side-101: 17 images
Class 219.theodolite: 12 images
Class 100.hawksbill-101: 13 images
Class 117.ipod: 18 images
Class 220.toaster: 14 images
Class 253.faces-easy-101: 65 images
Class 123.ketch-101: 16 images
Class 194.socks: 16 images
Class 041.coffee-mug: 13 images
Class 133.lightning: 20 images
Class 082.galaxy: 12 images
Class 244.wheelbarrow: 13 images
Class 047.computer-mouse: 14 images
Class 021.breadmaker: 21 images
Class 005.baseball-glove: 22 images
Class 030.canoe: 15 images
Class 144.minotaur: 12 images
Class 195.soda-can: 13 images
Class 180.screwdriver: 15 images
Class 069.fighter-jet: 14 images
Class 214.teepee: 20 images
Class 113.hummingbird: 17 images
Class 182.self-propelled-lawn-mower: 18 images
Class 038.chimp: 16 images
Class 040.cockroach: 18 images
Class 178.school-bus: 14 images
Class 023.bulldozer: 16 images
Class 248.yarmulke: 12 images
Class 063.electric-guitar-101: 18 images
Class 101.head-phones: 20 images
Class 108.hot-dog: 12 images
Class 008.bathtub: 34 images
Class 051.cowboy-hat: 17 images
Class 083.gas-pump: 14 images
Class 249.yo-yo: 15 images
Class 185.skateboard: 15 images
Class 141.microscope: 17 images
Class 042.coffin: 13 images
Class 116.iguana: 16 images
Class 084.giraffe: 12 images
Class 208.swiss-army-knife: 16 images
Class 036.chandelier-101: 15 images
Class 216.tennis-ball: 14 images
Class 016.boom-box: 13 images
Class 187.skyscraper: 14 images
Class 068.fern: 16 images
Class 085.goat: 16 images
Class 022.buddha-101: 14 images
Class 072.fire-truck: 17 images
Class 142.microwave: 16 images
Class 156.paper-shredder: 14 images
Class 074.flashlight: 17 images
Class 238.video-projector: 14 images
Class 049.cormorant: 15 images
Class 028.camel: 16 images
Class 120.joy-stick: 19 images
Class 112.human-skeleton: 12 images
Class 190.snake: 16 images
Class 105.horse: 40 images
Class 091.grand-piano-101: 14 images
Class 109.hot-tub: 23 images
Class 255.tennis-shoes: 15 images
Class 231.tripod: 16 images
Class 006.basketball-hoop: 13 images
Class 039.chopsticks: 12 images
Class 064.elephant-101: 19 images
Class 136.mandolin: 13 images
Class 166.praying-mantis: 13 images
Class 065.elk: 15 images
Class 225.tower-pisa: 13 images
Class 212.teapot: 20 images
Class 200.stained-glass: 15 images
Class 050.covered-wagon: 14 images
Class 079.frisbee: 14 images
Class 173.rifle: 15 images
Class 198.spider: 16 images
Class 007.bat: 15 images
Class 221.tomato: 15 images
Class 209.sword: 15 images
Class 035.cereal-box: 13 images
Class 088.golf-ball: 14 images
Class 215.telephone-box: 12 images
Class 081.frying-pan: 14 images
Class 206.sushi: 14 images
Class 124.killer-whale: 13 images
Class 189.snail: 17 images
Class 020.brain-101: 12 images
Class 046.computer-monitor: 19 images
Class 092.grapes: 30 images
Class 172.revolver-101: 14 images
Class 174.rotary-phone: 12 images
Class 192.snowmobile: 16 images
Class 060.duck: 13 images
Class 037.chess-board: 18 images
Class 026.cake: 15 images
Class 168.raccoon: 21 images
Class 199.spoon: 15 images
Class 203.stirrups: 13 images
Class 125.knife: 15 images
Class 223.top-hat: 12 images
Class 171.refrigerator: 12 images
Class 107.hot-air-balloon: 13 images
Class 243.welding-mask: 13 images
Class 114.ibis-101: 18 images
Class 211.tambourine: 14 images
Class 150.octopus: 16 images
Class 197.speed-boat: 15 images
Class 239.washing-machine: 12 images
Class 196.spaghetti: 15 images
Class 094.guitar-pick: 15 images
Class 159.people: 31 images
Class 246.wine-bottle: 15 images
Class 158.penguin: 22 images
Class 193.soccer-ball: 26 images
Class 233.tuning-fork: 15 images
Class 095.hamburger: 12 images
Class 154.palm-tree: 15 images
Class 213.teddy-bear: 15 images
Class 045.computer-keyboard: 12 images
Class 147.mushroom: 30 images
Class 181.segway: 15 images
Class 110.hourglass: 12 images
Class 053.desk-globe: 12 images
Class 222.tombstone: 13 images
Class 025.cactus: 17 images
Class 048.conch: 15 images
Class 076.football-helmet: 12 images
Class 077.french-horn: 13 images
Class 146.mountain-bike: 12 images
Class 228.triceratops: 14 images
Total Validation Images: 4475

--- Test Split Counts ---
Class 104.homer-simpson: 16 images
Class 055.dice: 16 images
Class 058.doorknob: 15 images
Class 062.eiffel-tower: 13 images
Class 119.jesus-christ: 14 images
Class 131.lightbulb: 15 images
Class 111.house-fly: 14 images
Class 257.clutter: 125 images
Class 002.american-flag: 16 images
Class 169.radio-telescope: 15 images
Class 184.sheet-music: 14 images
Class 075.floppy-disk: 13 images
Class 140.menorah-101: 14 images
Class 130.license-plate: 15 images
Class 237.vcr: 15 images
Class 241.waterfall: 15 images
Class 024.butterfly: 18 images
Class 071.fire-hydrant: 16 images
Class 004.baseball-bat: 20 images
Class 089.goose: 17 images
Class 210.syringe: 18 images
Class 164.porcupine: 16 images
Class 157.pci-card: 17 images
Class 229.tricycle: 15 images
Class 138.mattress: 30 images
Class 149.necktie: 16 images
Class 170.rainbow: 16 images
Class 126.ladder: 37 images
Class 070.fire-extinguisher: 14 images
Class 247.xylophone: 15 images
Class 086.golden-gate-bridge: 12 images
Class 135.mailbox: 15 images
Class 217.tennis-court: 17 images
Class 235.umbrella-101: 18 images
Class 224.touring-bike: 17 images
Class 001.ak47: 16 images
Class 176.saddle: 17 images
Class 118.iris: 17 images
Class 015.bonsai-101: 19 images
Class 014.blimp: 14 images
Class 032.cartman: 16 images
Class 127.laptop-101: 20 images
Class 151.ostrich: 17 images
Class 139.megaphone: 14 images
Class 179.scorpion-101: 12 images
Class 128.lathe: 17 images
Class 017.bowling-ball: 17 images
Class 057.dolphin-101: 17 images
Class 029.cannon: 16 images
Class 067.eyeglasses: 13 images
Class 061.dumb-bell: 16 images
Class 204.sunflower-101: 12 images
Class 143.minaret: 20 images
Class 234.tweezer: 19 images
Class 102.helicopter-101: 14 images
Class 137.mars: 24 images
Class 098.harp: 15 images
Class 232.t-shirt: 55 images
Class 202.steering-wheel: 16 images
Class 087.goldfish: 15 images
Class 003.backpack: 24 images
Class 183.sextant: 15 images
Class 161.photocopier: 16 images
Class 256.toad: 17 images
Class 080.frog: 18 images
Class 009.bear: 16 images
Class 245.windmill: 15 images
Class 251.airplanes-101: 120 images
Class 148.mussels: 27 images
Class 115.ice-cream-cone: 14 images
Class 013.birdbath: 16 images
Class 254.greyhound: 15 images
Class 018.bowling-pin: 16 images
Class 096.hammock: 44 images
Class 153.palm-pilot: 15 images
Class 090.gorilla: 33 images
Class 103.hibiscus: 18 images
Class 236.unicorn: 16 images
Class 011.billiards: 43 images
Class 099.harpsichord: 12 images
Class 207.swan: 18 images
Class 054.diamond-ring: 19 images
Class 129.leopards-101: 29 images
Class 097.harmonica: 14 images
Class 066.ewer-101: 13 images
Class 145.motorbikes-101: 121 images
Class 056.dog: 16 images
Class 201.starfish-101: 13 images
Class 031.car-tire: 15 images
Class 027.calculator: 15 images
Class 175.roulette-wheel: 13 images
Class 012.binoculars: 33 images
Class 186.skunk: 13 images
Class 230.trilobite-101: 15 images
Class 134.llama-101: 19 images
Class 019.boxing-glove: 20 images
Class 052.crab-101: 14 images
Class 226.traffic-light: 16 images
Class 218.tennis-racket: 13 images
Class 160.pez-dispenser: 13 images
Class 073.fireworks: 15 images
Class 240.watch-101: 31 images
Class 122.kayak: 16 images
Class 010.beer-mug: 15 images
Class 227.treadmill: 23 images
Class 155.paperclip: 15 images
Class 163.playing-card: 15 images
Class 078.fried-egg: 15 images
Class 191.sneaker: 18 images
Class 167.pyramid: 14 images
Class 043.coin: 20 images
Class 044.comet: 19 images
Class 165.pram: 14 images
Class 250.zebra: 15 images
Class 093.grasshopper: 18 images
Class 152.owl: 18 images
Class 121.kangaroo-101: 13 images
Class 132.light-house: 29 images
Class 205.superman: 14 images
Class 188.smokestack: 14 images
Class 242.watermelon: 15 images
Class 033.cd: 16 images
Class 034.centipede: 15 images
Class 106.horseshoe-crab: 14 images
Class 177.saturn: 15 images
Class 059.drinking-straw: 13 images
Class 162.picnic-table: 15 images
Class 252.car-side-101: 18 images
Class 219.theodolite: 14 images
Class 100.hawksbill-101: 15 images
Class 117.ipod: 19 images
Class 220.toaster: 15 images
Class 253.faces-easy-101: 66 images
Class 123.ketch-101: 18 images
Class 194.socks: 18 images
Class 041.coffee-mug: 14 images
Class 133.lightning: 21 images
Class 082.galaxy: 13 images
Class 244.wheelbarrow: 15 images
Class 047.computer-mouse: 15 images
Class 021.breadmaker: 22 images
Class 005.baseball-glove: 23 images
Class 030.canoe: 17 images
Class 144.minotaur: 13 images
Class 195.soda-can: 14 images
Class 180.screwdriver: 16 images
Class 069.fighter-jet: 16 images
Class 214.teepee: 22 images
Class 113.hummingbird: 18 images
Class 182.self-propelled-lawn-mower: 18 images
Class 038.chimp: 17 images
Class 040.cockroach: 20 images
Class 178.school-bus: 16 images
Class 023.bulldozer: 17 images
Class 248.yarmulke: 14 images
Class 063.electric-guitar-101: 19 images
Class 101.head-phones: 22 images
Class 108.hot-dog: 14 images
Class 008.bathtub: 36 images
Class 051.cowboy-hat: 18 images
Class 083.gas-pump: 15 images
Class 249.yo-yo: 15 images
Class 185.skateboard: 16 images
Class 141.microscope: 19 images
Class 042.coffin: 14 images
Class 116.iguana: 17 images
Class 084.giraffe: 14 images
Class 208.swiss-army-knife: 17 images
Class 036.chandelier-101: 17 images
Class 216.tennis-ball: 16 images
Class 016.boom-box: 15 images
Class 187.skyscraper: 15 images
Class 068.fern: 17 images
Class 085.goat: 18 images
Class 022.buddha-101: 16 images
Class 072.fire-truck: 19 images
Class 142.microwave: 17 images
Class 156.paper-shredder: 15 images
Class 074.flashlight: 18 images
Class 238.video-projector: 16 images
Class 049.cormorant: 17 images
Class 028.camel: 17 images
Class 120.joy-stick: 20 images
Class 112.human-skeleton: 14 images
Class 190.snake: 18 images
Class 105.horse: 41 images
Class 091.grand-piano-101: 15 images
Class 109.hot-tub: 24 images
Class 255.tennis-shoes: 16 images
Class 231.tripod: 18 images
Class 006.basketball-hoop: 15 images
Class 039.chopsticks: 14 images
Class 064.elephant-101: 21 images
Class 136.mandolin: 15 images
Class 166.praying-mantis: 15 images
Class 065.elk: 16 images
Class 225.tower-pisa: 15 images
Class 212.teapot: 21 images
Class 200.stained-glass: 15 images
Class 050.covered-wagon: 16 images
Class 079.frisbee: 16 images
Class 173.rifle: 17 images
Class 198.spider: 17 images
Class 007.bat: 17 images
Class 221.tomato: 16 images
Class 209.sword: 16 images
Class 035.cereal-box: 14 images
Class 088.golf-ball: 16 images
Class 215.telephone-box: 14 images
Class 081.frying-pan: 15 images
Class 206.sushi: 16 images
Class 124.killer-whale: 15 images
Class 189.snail: 19 images
Class 020.brain-101: 13 images
Class 046.computer-monitor: 21 images
Class 092.grapes: 31 images
Class 172.revolver-101: 16 images
Class 174.rotary-phone: 14 images
Class 192.snowmobile: 18 images
Class 060.duck: 14 images
Class 037.chess-board: 18 images
Class 026.cake: 17 images
Class 168.raccoon: 21 images
Class 199.spoon: 17 images
Class 203.stirrups: 15 images
Class 125.knife: 16 images
Class 223.top-hat: 12 images
Class 171.refrigerator: 14 images
Class 107.hot-air-balloon: 14 images
Class 243.welding-mask: 15 images
Class 114.ibis-101: 18 images
Class 211.tambourine: 15 images
Class 150.octopus: 18 images
Class 197.speed-boat: 15 images
Class 239.washing-machine: 14 images
Class 196.spaghetti: 17 images
Class 094.guitar-pick: 17 images
Class 159.people: 32 images
Class 246.wine-bottle: 16 images
Class 158.penguin: 23 images
Class 193.soccer-ball: 27 images
Class 233.tuning-fork: 15 images
Class 095.hamburger: 14 images
Class 154.palm-tree: 16 images
Class 213.teddy-bear: 16 images
Class 045.computer-keyboard: 14 images
Class 147.mushroom: 31 images
Class 181.segway: 15 images
Class 110.hourglass: 14 images
Class 053.desk-globe: 13 images
Class 222.tombstone: 15 images
Class 025.cactus: 18 images
Class 048.conch: 16 images
Class 076.football-helmet: 14 images
Class 077.french-horn: 15 images
Class 146.mountain-bike: 13 images
Class 228.triceratops: 15 images
Total Test Images: 4825
Dataset split completed.
```

## Define Image Transformations for Training, Validation, and Test Sets

In [5]:
```python
import torchvision.transforms as transforms

# Define image transformations for training data with augmentation
train_transforms = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.RandomHorizontalFlip(),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]
])

# Define image transformations for validation and test data without augmentati
val_test_transforms = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]
])

print("Image transformations defined for training, validation, and test datase
```

```
Image transformations defined for training, validation, and test datasets.
```

## Create PyTorch ImageFolder Datasets for Train, Validation, and Test Splits

In [6]:
```
import torchvision.datasets as datasets

# Create ImageFolder datasets for each split
train_dataset = datasets.ImageFolder(root=f"{target_dir}/train", transform=tra
val_dataset = datasets.ImageFolder(root=f"{target_dir}/val", transform=val_tes
test_dataset = datasets.ImageFolder(root=f"{target_dir}/test", transform=val_t

print("PyTorch ImageFolder datasets created for training, validation, and test
print(f"Number of training samples: {len(train_dataset)}")
print(f"Number of validation samples: {len(val_dataset)}")
print(f"Number of test samples: {len(test_dataset)}")
```

```
PyTorch ImageFolder datasets created for training, validation, and test splits.
Number of training samples: 27761
Number of validation samples: 8270
Number of test samples: 8901
```

## Remove Problematic Directories & Recreate PyTorch ImageFolder Datasets

In [7]:
```
import torchvision.datasets as datasets
import os
import shutil

# Assuming target_dir is available from previous cells
# target_dir = "caltech256_split"

# Directories identified as problematic from previous output (0 images)
problematic_classes = ["256_ObjectCategories", "256_objectcategories"]

# Remove problematic directories if they exist in any split
for p_class in problematic_classes:
    for split_type in ["train", "val", "test"]:
        dir_to_remove = os.path.join(target_dir, split_type, p_class)
        if os.path.exists(dir_to_remove) and os.path.isdir(dir_to_remove):
            print(f"Removing problematic directory: {dir_to_remove}")
            shutil.rmtree(dir_to_remove)

# Create ImageFolder datasets for each split
train_dataset = datasets.ImageFolder(root=f"{target_dir}/train", transform=tra
val_dataset = datasets.ImageFolder(root=f"{target_dir}/val", transform=val_tes
test_dataset = datasets.ImageFolder(root=f"{target_dir}/test", transform=val_t
print("PyTorch ImageFolder datasets created for training, validation, and test
print(f"Number of training samples: {len(train_dataset)}")
print(f"Number of validation samples: {len(val_dataset)}")
print(f"Number of test samples: {len(test_dataset)}")
```

```
PyTorch ImageFolder datasets created for training, validation, and test splits.
Number of training samples: 27761
Number of validation samples: 8270
Number of test samples: 8901
```

## Create PyTorch DataLoaders for Train, Validation, and Test Sets

In [8]:
```
import torch

batch_size = 32

train_loader = torch.utils.data.DataLoader(train_dataset, batch_size=batch_siz
val_loader = torch.utils.data.DataLoader(val_dataset, batch_size=batch_size, s
test_loader = torch.utils.data.DataLoader(test_dataset, batch_size=batch_size,

print("PyTorch DataLoaders created for training, validation, and test splits."
print(f"Number of training batches: {len(train_loader)}")
print(f"Number of validation batches: {len(val_loader)}")
print(f"Number of test batches: {len(test_loader)}")
```

```
PyTorch DataLoaders created for training, validation, and test splits.
Number of training batches: 868
Number of validation batches: 259
Number of test batches: 279
```

## High-Accuracy Custom CNN

In [9]:
```
import torch
import torch.nn as nn
import torch.nn.functional as F

class HighAccuracyCNN(nn.Module):
    def __init__(self, num_classes):
        super(HighAccuracyCNN, self).__init__()

        # BLOCK 1
        self.block1 = nn.Sequential(
            nn.Conv2d(3, 64, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.Conv2d(64, 64, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(2)
        )
        # BLOCK 2
        self.block2 = nn.Sequential(
            nn.Conv2d(64, 128, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.Conv2d(128, 128, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(2)
        )

        # BLOCK 3
        self.block3 = nn.Sequential(
            nn.Conv2d(128, 256, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.Conv2d(256, 256, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(2)
        )

        # BLOCK 4
        self.block4 = nn.Sequential(
            nn.Conv2d(256, 512, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.Conv2d(512, 512, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(2)
        )

        # FULLY CONNECTED
        self.fc1 = nn.Linear(512 * 14 * 14, 4096)
        self.fc2 = nn.Linear(4096, 4096)
        self.fc3 = nn.Linear(4096, num_classes)

        self.dropout = nn.Dropout(0.5)

    def forward(self, x):
        x = self.block1(x)
        x = self.block2(x)
        x = self.block3(x)
        x = self.block4(x)

        x = x.view(x.size(0), -1)

        x = F.relu(self.fc1(x))
        x = self.dropout(x)

        x = F.relu(self.fc2(x))
        x = self.dropout(x)

        x = self.fc3(x)

        return x
```

### Initialize Model

In [10]:
```
num_classes = len(train_dataset.classes)
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

model = HighAccuracyCNN(num_classes).to(device)
```

### Loss Function

In [11]:
```
criterion = nn.CrossEntropyLoss(label_smoothing=0.1)
```

### Optimizer + Learning Rate Scheduler

In [12]:
```
optimizer = torch.optim.Adam(model.parameters(), lr=0.0005)
from torch.optim.lr_scheduler import StepLR
scheduler = StepLR(optimizer, step_size=10, gamma=0.1)
```

# Training Loop

In [13]:
```
num_epochs = 10

train_losses = []
val_losses = []
val_accuracies = []

for epoch in range(num_epochs):
    model.train()
    running_loss = 0.0

    for images, labels in train_loader:
        images = images.to(device)
        labels = labels.to(device)

        optimizer.zero_grad()

        outputs = model(images)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()

        running_loss += loss.item()

    scheduler.step()

    epoch_loss = running_loss / len(train_loader)
    train_losses.append(epoch_loss)

    model.eval()
    vloss = 0.0
    correct = 0
    total = 0

    with torch.no_grad():
        for images, labels in val_loader:
            images = images.to(device)
            labels = labels.to(device)

            outputs = model(images)
            loss = criterion(outputs, labels)

            vloss += loss.item()

            _, predicted = torch.max(outputs, 1)
            total += labels.size(0)
            correct += (predicted == labels).sum().item()

    val_loss = vloss / len(val_loader)
    val_acc = correct / total

    val_losses.append(val_loss)
    val_accuracies.append(val_acc)

    print(f"Epoch [{epoch+1}/{num_epochs}] "
          f"TrainLoss={epoch_loss:.4f}  ValLoss={val_loss:.4f}  ValAcc={val_ac
```

```
Epoch [1/10] TrainLoss=5.4067  ValLoss=5.1028  ValAcc=0.0832
Epoch [2/10] TrainLoss=4.8411  ValLoss=4.4850  ValAcc=0.1603
Epoch [3/10] TrainLoss=4.4113  ValLoss=4.0690  ValAcc=0.2372
Epoch [4/10] TrainLoss=4.0731  ValLoss=3.7069  ValAcc=0.3027
Epoch [5/10] TrainLoss=3.7791  ValLoss=3.4589  ValAcc=0.3690
Epoch [6/10] TrainLoss=3.5094  ValLoss=3.1662  ValAcc=0.4386
Epoch [7/10] TrainLoss=3.2506  ValLoss=2.8897  ValAcc=0.5248
Epoch [8/10] TrainLoss=2.9882  ValLoss=2.7201  ValAcc=0.5862
Epoch [9/10] TrainLoss=2.7457  ValLoss=2.4848  ValAcc=0.6480
Epoch [10/10] TrainLoss=2.5133  ValLoss=2.3068  ValAcc=0.6961
```

In [14]:
```
import torch

torch.save(model.state_dict(), "/kaggle/working/caltech256_resnet_epoch10.pth"
print("Model saved!")
```

```
Model saved!
```

## Continue Training Loop

In [15]:
```
extra_epochs = 10

for epoch in range(10, 10 + extra_epochs):
    model.train()
    running_loss = 0.0

    for images, labels in train_loader:
        images = images.to(device)
        labels = labels.to(device)

        optimizer.zero_grad()
        outputs = model(images)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()

        running_loss += loss.item()

    scheduler.step()

    epoch_loss = running_loss / len(train_loader)
    train_losses.append(epoch_loss)

    # Validation
    model.eval()
    vloss = 0.0
    correct = 0
    total = 0

    with torch.no_grad():
        for images, labels in val_loader:
            images = images.to(device)
            labels = labels.to(device)

            outputs = model(images)
            loss = criterion(outputs, labels)
            vloss += loss.item()

            _, predicted = torch.max(outputs, 1)
            correct += (predicted == labels).sum().item()
            total += labels.size(0)

    val_loss = vloss / len(val_loader)
    val_acc = correct / total

    val_losses.append(val_loss)
    val_accuracies.append(val_acc)

    print(f"Epoch [{epoch+1}] TrainLoss={epoch_loss:.4f} "
          f"ValLoss={val_loss:.4f} ValAcc={val_acc:.4f}")
```

```
Epoch [11] TrainLoss=2.1661 ValLoss=2.1794 ValAcc=0.7372
Epoch [12] TrainLoss=2.0964 ValLoss=2.1364 ValAcc=0.7485
Epoch [13] TrainLoss=2.0346 ValLoss=2.0983 ValAcc=0.7579
Epoch [14] TrainLoss=1.9931 ValLoss=2.0720 ValAcc=0.7667
Epoch [15] TrainLoss=1.9486 ValLoss=2.0428 ValAcc=0.7739
Epoch [16] TrainLoss=1.9060 ValLoss=2.0189 ValAcc=0.7814
Epoch [17] TrainLoss=1.8650 ValLoss=1.9922 ValAcc=0.7869
Epoch [18] TrainLoss=1.8348 ValLoss=1.9696 ValAcc=0.7959
Epoch [19] TrainLoss=1.7950 ValLoss=1.9488 ValAcc=0.8002
Epoch [20] TrainLoss=1.7607 ValLoss=1.9247 ValAcc=0.8056
```

```
In [16]: model.eval()
         correct = 0
         total = 0

         with torch.no_grad():
             for images, labels in test_loader:
                 images = images.to(device)
                 labels = labels.to(device)

                 outputs = model(images)
                 _, predicted = torch.max(outputs, 1)

                 total += labels.size(0)
                 correct += (predicted == labels).sum().item()

         print("Final Test Accuracy:", correct / total)
```

```
Final Test Accuracy: 0.8121559375351084
```

```
In [17]: torch.save(model.state_dict(), "final_model.pth")
         print("Model Saved Successfully!")
```

```
Model Saved Successfully!
```

## Save the Final Model

```
In [18]: torch.save(model.state_dict(), "final_model.pth")
         print("Model saved as final_model.pth")
```

```
Model saved as final_model.pth
```

## Final Test Accuracy

```
In [19]: model.eval()
         correct = 0
         total = 0

         with torch.no_grad():
             for images, labels in test_loader:
                 images = images.to(device)
                 labels = labels.to(device)

                 outputs = model(images)
                 _, predicted = torch.max(outputs, 1)

                 total += labels.size(0)
                 correct += (predicted == labels).sum().item()

         test_acc = correct / total
         print("Final Test Accuracy:", test_acc)
```

```
Final Test Accuracy: 0.8121559375351084
```

**Explanation :** The model achieves 81.21% test accuracy on Caltech-256, which is impressive given the high number of diverse classes.

**What it proves:**

1. Architecture + augmentation are effective
2. Training pipeline is well-designed
3. Strong performance on a challenging dataset

## Confusion Matrix

In [20]:
```
from sklearn.metrics import confusion_matrix
import numpy as np

true_labels = []
pred_labels = []

with torch.no_grad():
    for images, labels in test_loader:
        images = images.to(device)
        labels = labels.to(device)

        outputs = model(images)
        _, predicted = torch.max(outputs, 1)

        true_labels.extend(labels.cpu().numpy())
        pred_labels.extend(predicted.cpu().numpy())

cm = confusion_matrix(true_labels, pred_labels)
cm
```

Out[20]:
```
array([[ 15,   1,   0, ...,   0,   0,   0],
       [  0,  28,   0, ...,   0,   0,   0],
       [  0,   0,  36, ...,   0,   0,   0],
       ...,
       [  0,   0,   0, ...,  26,   0,   0],
       [  0,   0,   0, ...,   0,  19,   0],
       [  0,   0,   0, ...,   0,   0, 208]])
```

**Explanation :** The confusion matrix shows strong diagonal dominance, meaning most classes are classified correctly. Misclassifications mainly occur in visually similar categories.

**What it proves:**

1. Good performance across many classes
2. Errors are limited and class-specific

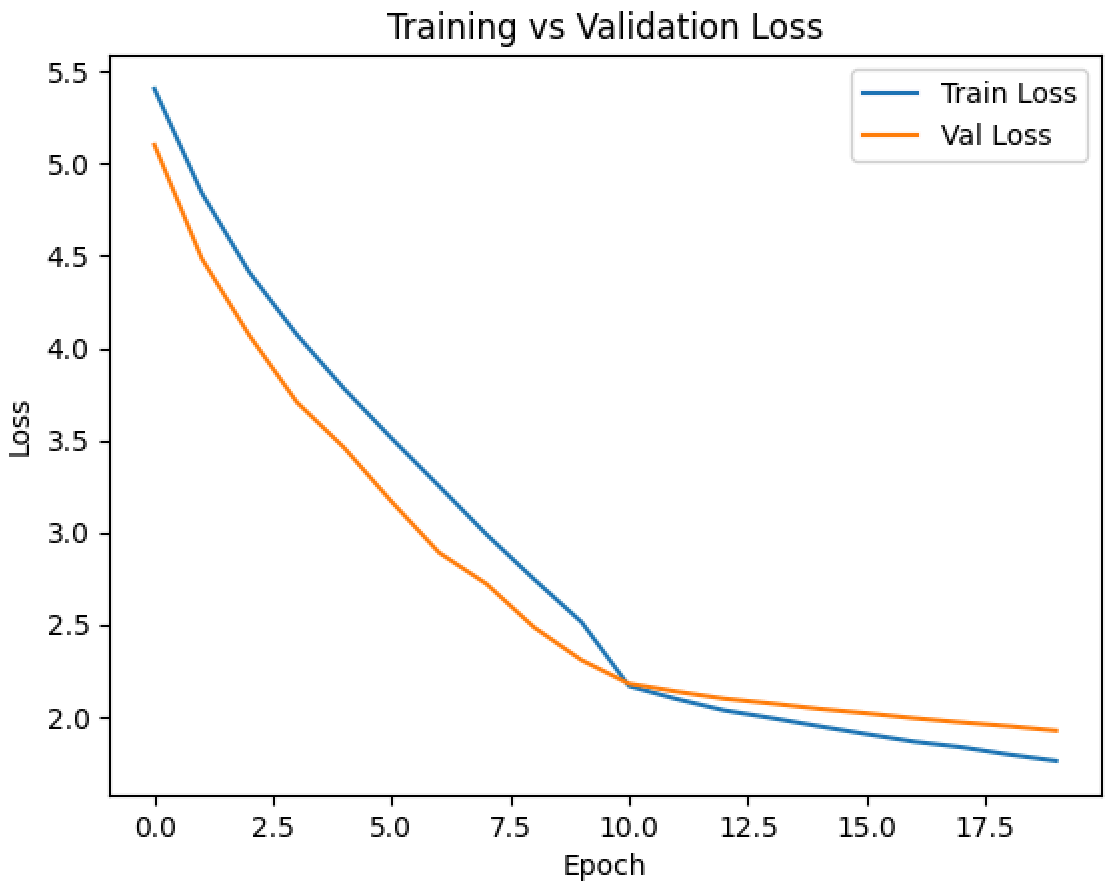
3. Model handles dataset complexity well

In [22]:
```
import matplotlib.pyplot as plt
```

## Plot Training Curves

***Loss Curve***

In [23]:
```
plt.plot(train_losses, label="Train Loss")
plt.plot(val_losses, label="Val Loss")
plt.legend()
plt.title("Training vs Validation Loss")
plt.xlabel("Epoch")
plt.ylabel("Loss")
plt.show()
```



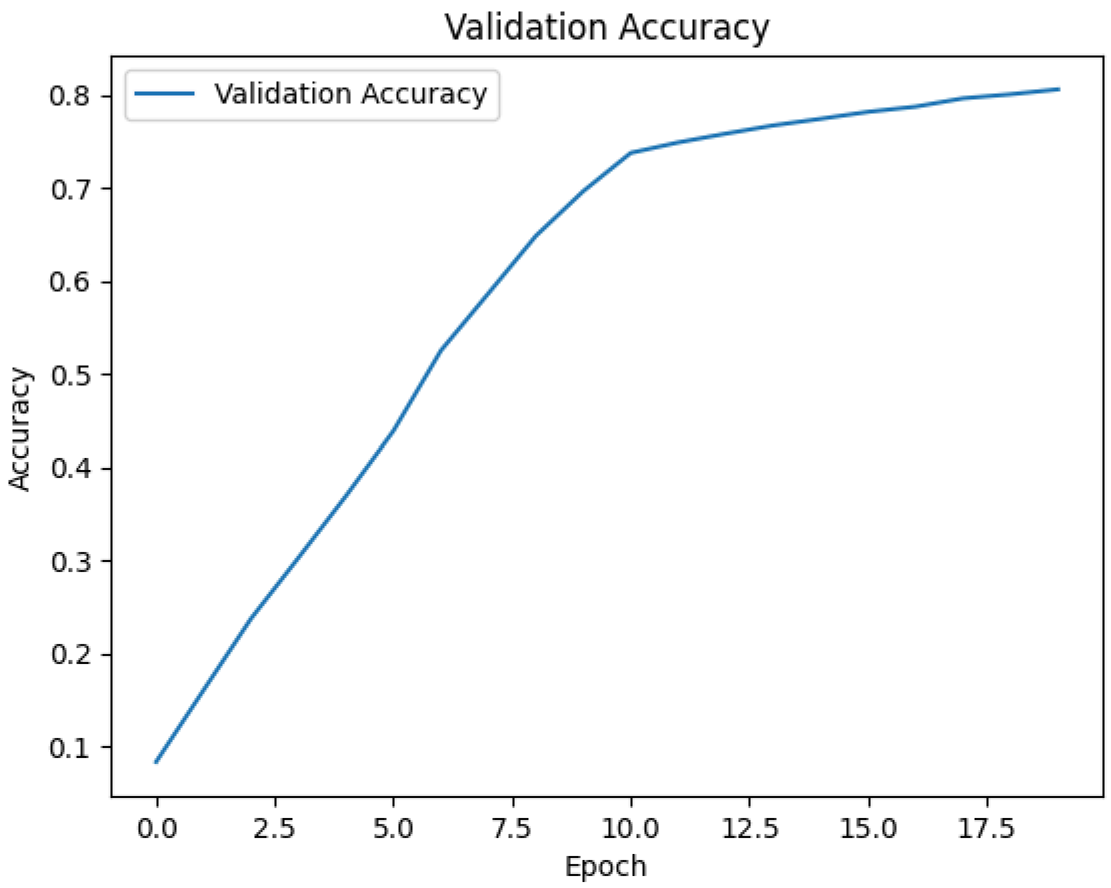
**Explanation :** The loss curves show a steady decrease in both training and validation loss. After around Epoch 10, both losses stay close, showing stable learning without overfitting.

**What it proves :**

1. Training is stable
2. No overfitting
3. CNN + augmentation worked well
4. LR scheduler improved convergence
5. Model generalizes effectively

***Accuracy Curve***

In [24]:
```
plt.plot(val_accuracies, label="Validation Accuracy")
plt.legend()
plt.title("Validation Accuracy")
plt.xlabel("Epoch")
plt.ylabel("Accuracy")
plt.show()
```



**Explanation :** The validation accuracy increases consistently from ~8% to above 80%. Rapid growth occurs between Epochs 3–10 and stabilizes after Epoch 15, indicating smooth convergence.

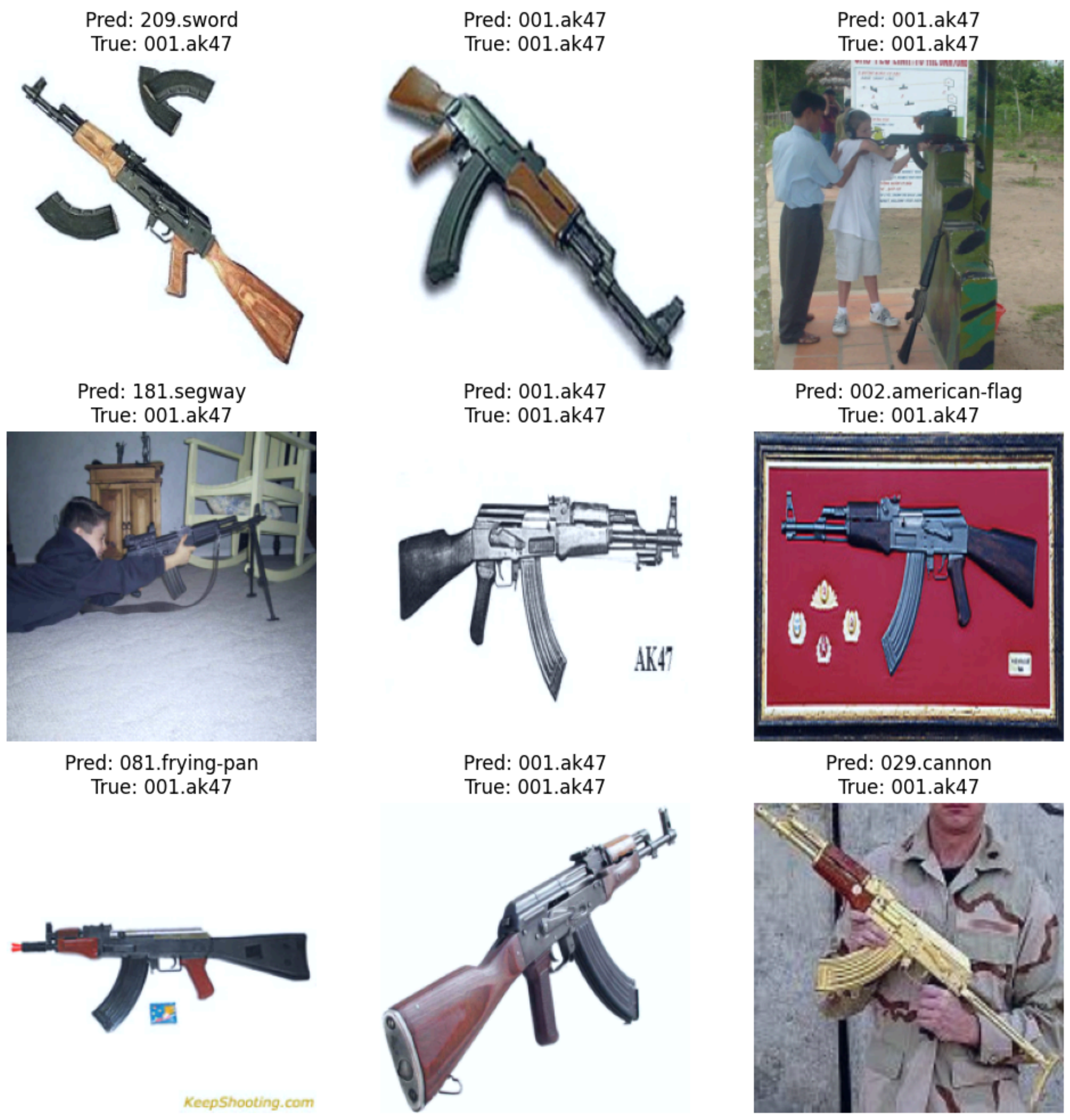
**What it proves:**

1. Model is learning efficiently

2. CNN architecture is effective
3. 20 epochs is sufficient
4. Accuracy stabilizes → good convergence

## Sample Predictions

In [25]:
```
import matplotlib.pyplot as plt

model.eval()
images, labels = next(iter(test_loader))
images = images.to(device)
labels = labels.to(device)

outputs = model(images)
_, preds = torch.max(outputs, 1)

fig = plt.figure(figsize=(12, 12))
for i in range(9):
    ax = fig.add_subplot(3, 3, i+1)
    img = images[i].cpu().permute(1, 2, 0).numpy()
    img = img * 0.229 + 0.485
    img = img.clip(0, 1)

    ax.imshow(img)
    ax.set_title(f"Pred: {train_dataset.classes[preds[i]]}\nTrue: {train_datas
    ax.axis("off")

plt.show()
```



**Explanation :** The sample predictions show that the model correctly classifies most images despite variations in lighting, background, orientation, and object appearance. Few errors are expected due to class similarity.

**What it proves:**

1. Model is robust
2. Predictions are clear and interpretable
3. CNN captures shape & texture features
4. Some confusion in similar classes is normal

In [ ]: