RICE UNIVERSITY

Week-2: Pytorch for Computer Vision

Motivation: Why Image Processing and Analysis?

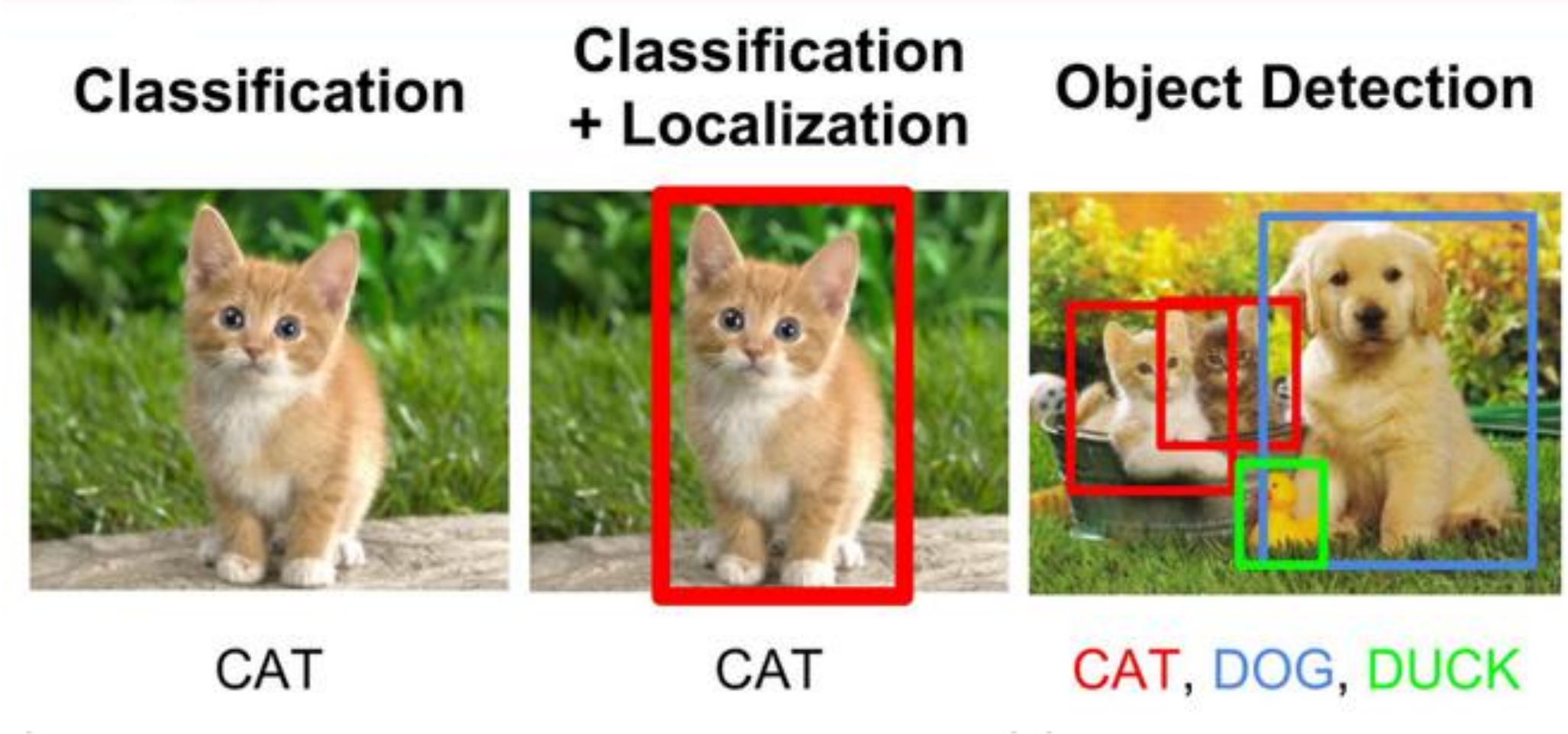


Image Datasets -- **MNIST**

- Handwritten digit dataset introduced by LeCun et al., 1998
- 70,000 grayscale images (28×28), digits 0–9
- 60K training / 10K test split
- Simple, lightweight benchmark for image classification

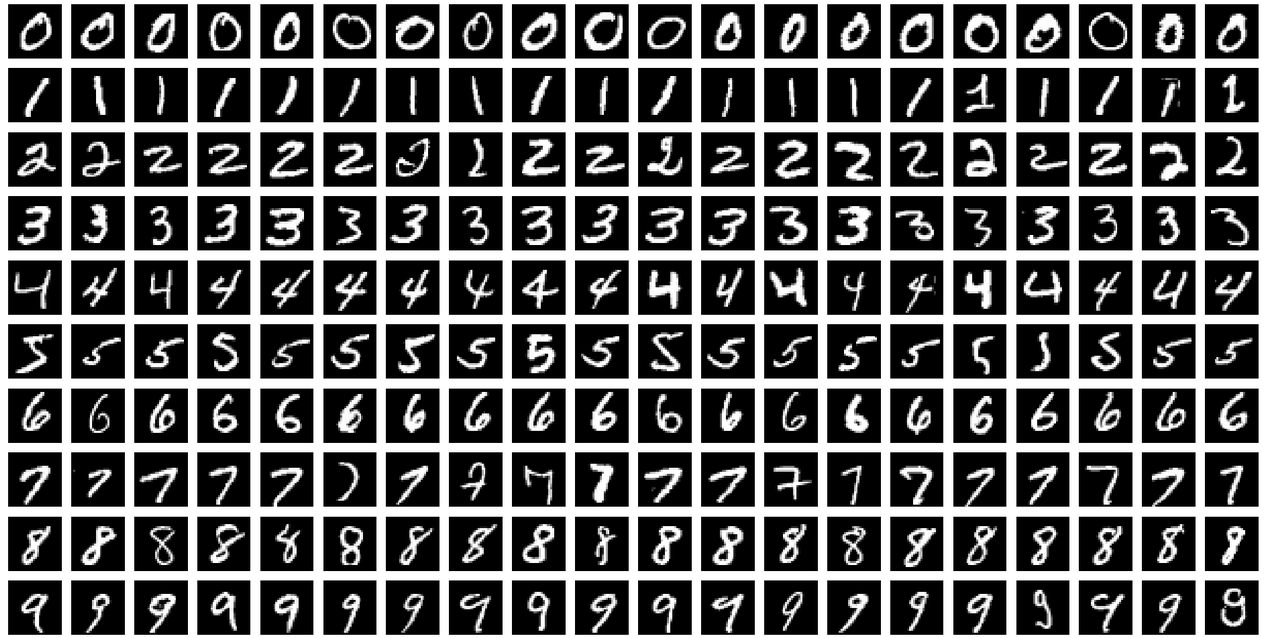
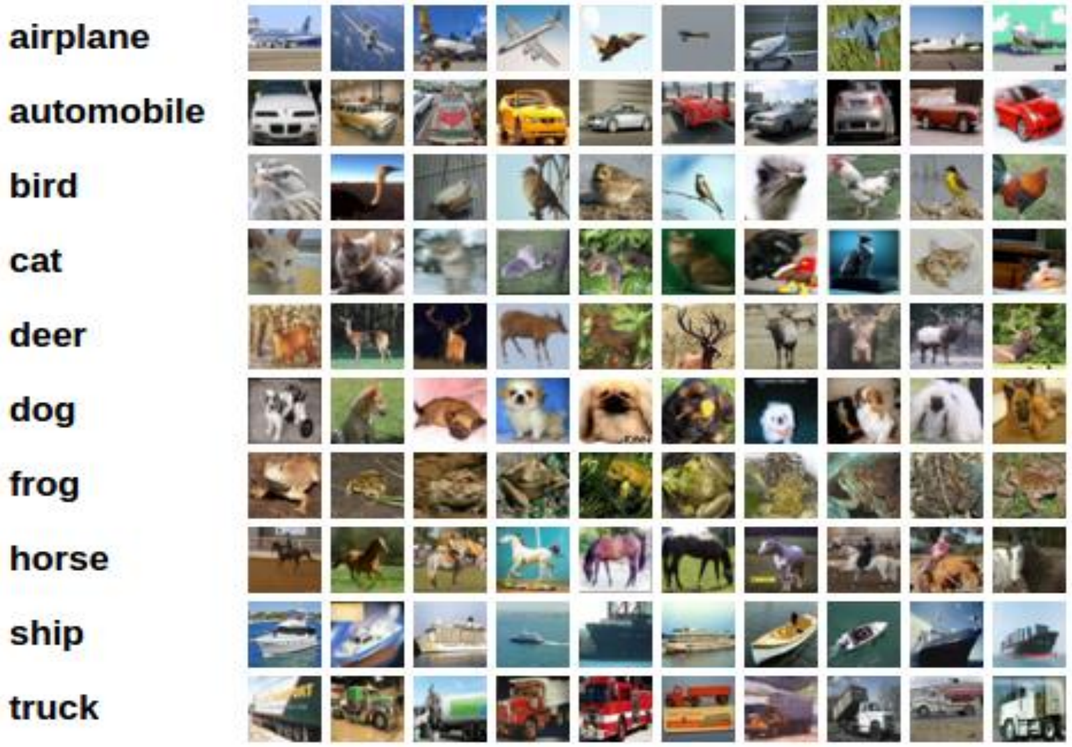


Image Datasets -- **Cifar-10**



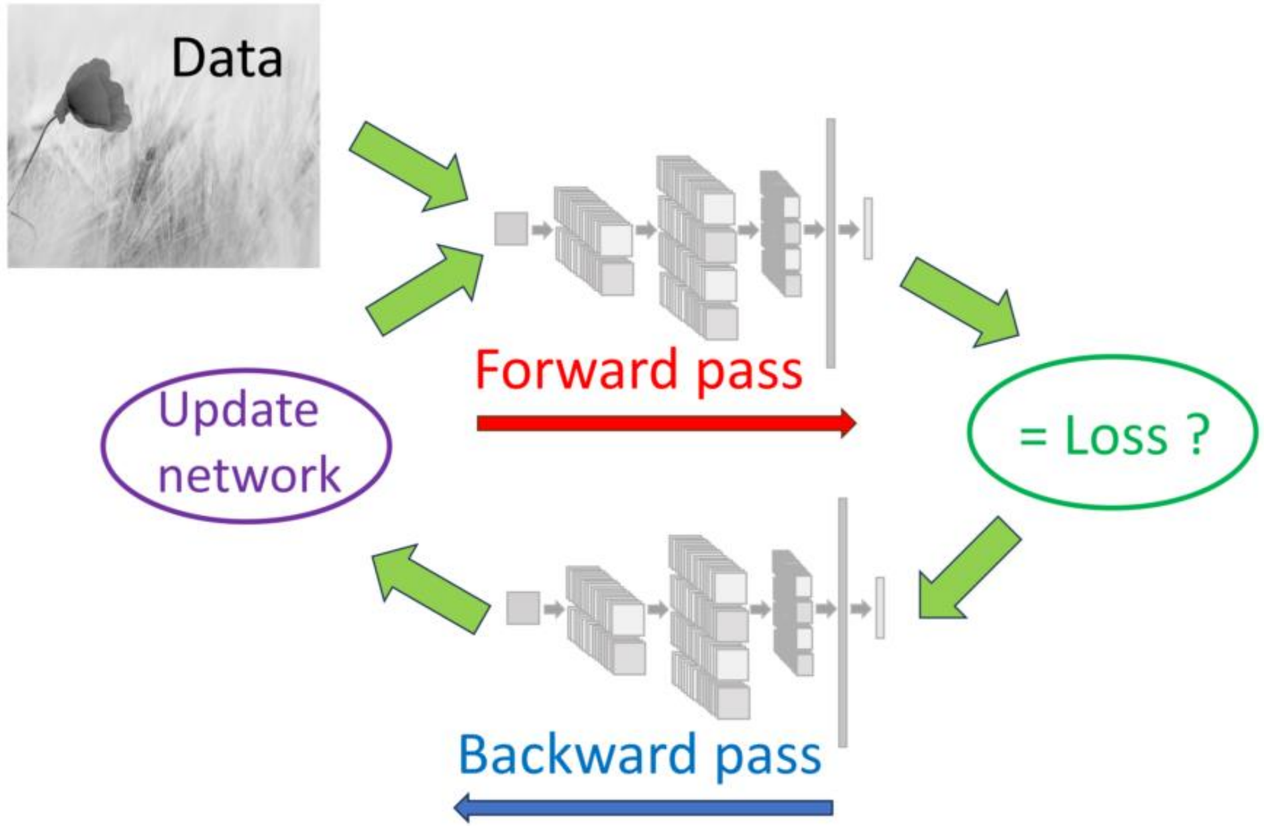
- Classic dataset introduced by Krizhevsky et al., 2009
- 60,000 color images (32×32), 10 object classes
- 50K training / 10K test split
- Compact, runs fast on CPU/GPU hardware

Image Datasets -- **ImageNet**

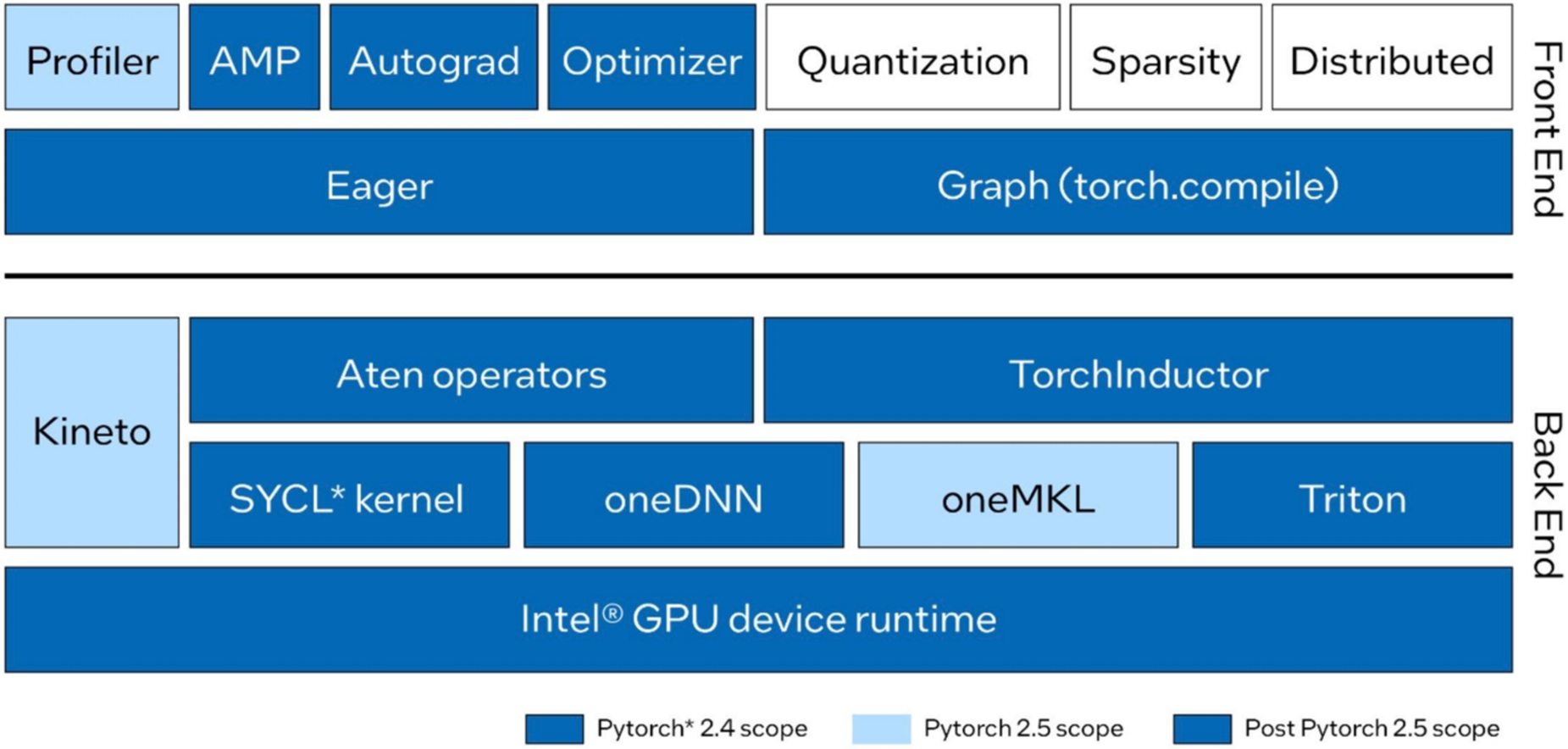
- Large-scale visual dataset introduced by Deng et al., 2009
- Over 14 million labeled images, 21K+ object categories.
- Standard subset: 1.28M training / 50K validation images (1000 classes).
- High-resolution images (varied sizes), challenging benchmark for deep CNNs.

Training Loop Essentials

- Move data to device (CPU/GPU)
- Forward pass → compute predictions
- Compute loss → backward() → update weights
- Repeat for multiple epochs



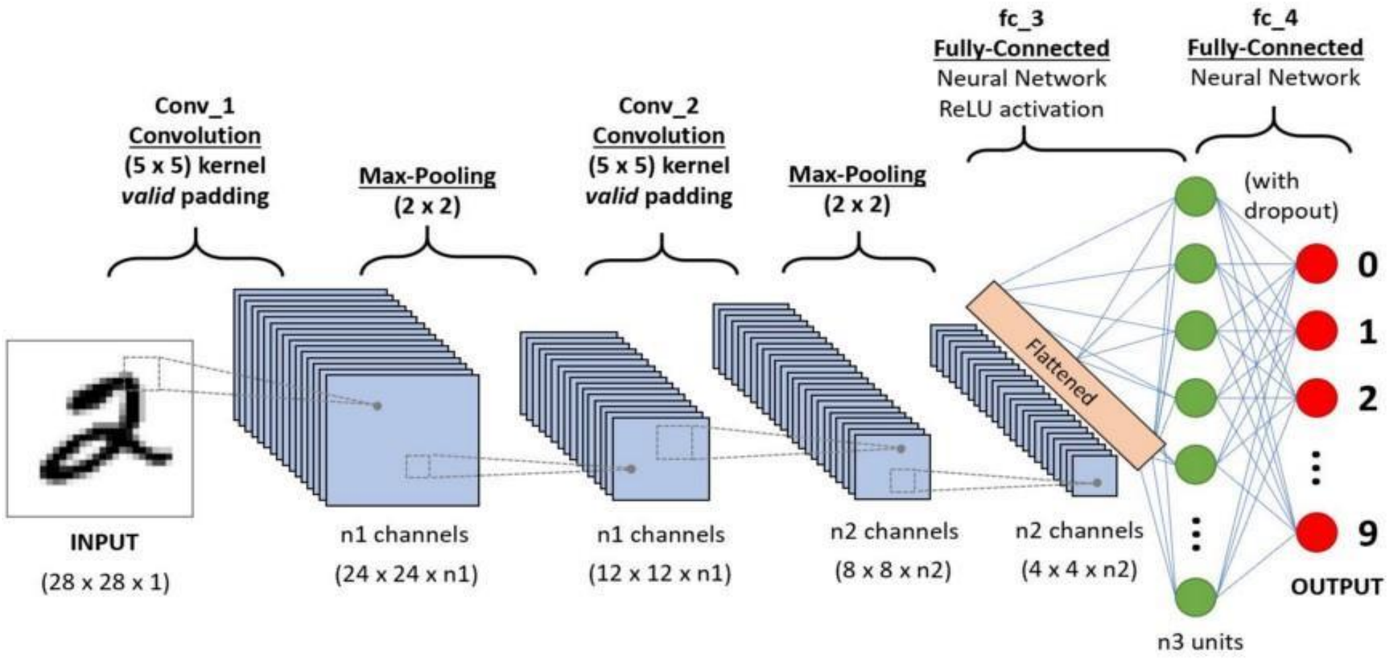
PyTorch and Deep Learning Ecosystem



oneDNN stands for oneAPI Deep Neural Network Library and oneMKL stands for oneAPI Math Kernel Library.

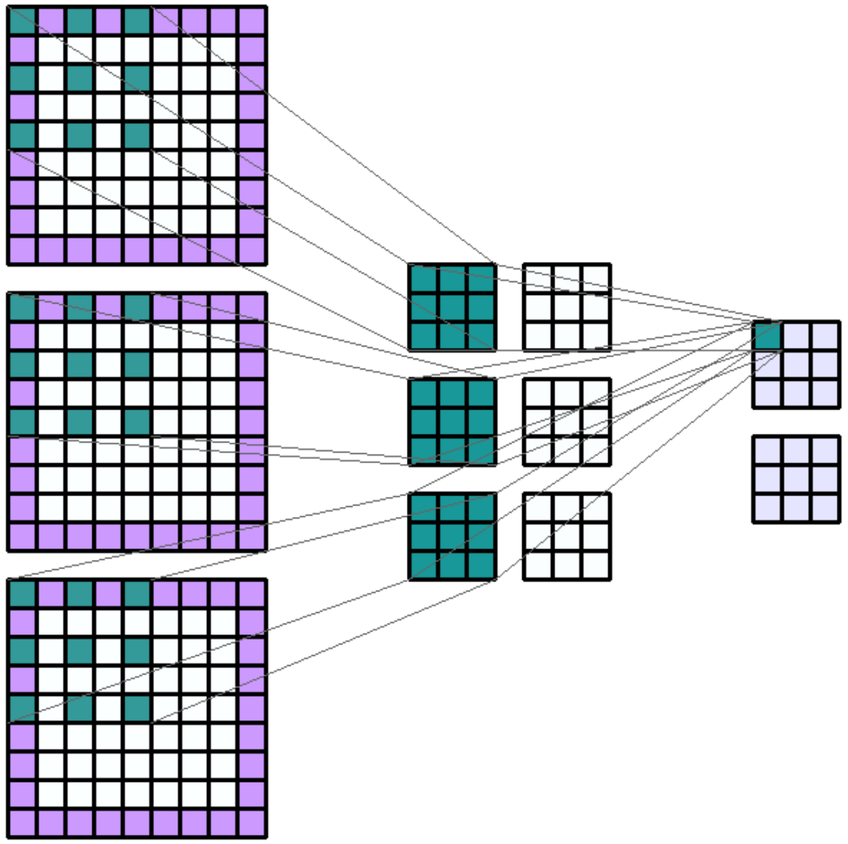
Common Operators in CIFAR-10 Model

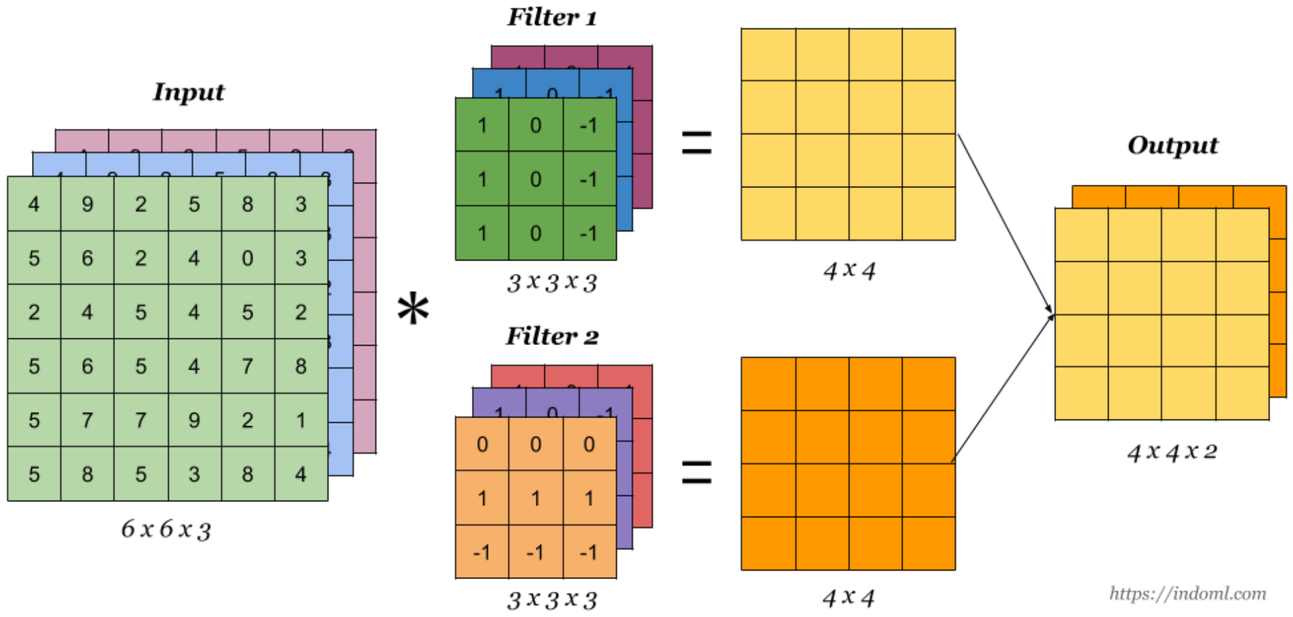
- Conv2d: extracts spatial features
- ReLU: adds nonlinearity
- MaxPool2d: downsampling
- Linear: fully connected layer
- CrossEntropyLoss: combines Softmax + log loss



Common Operators in CIFAR-10 Model

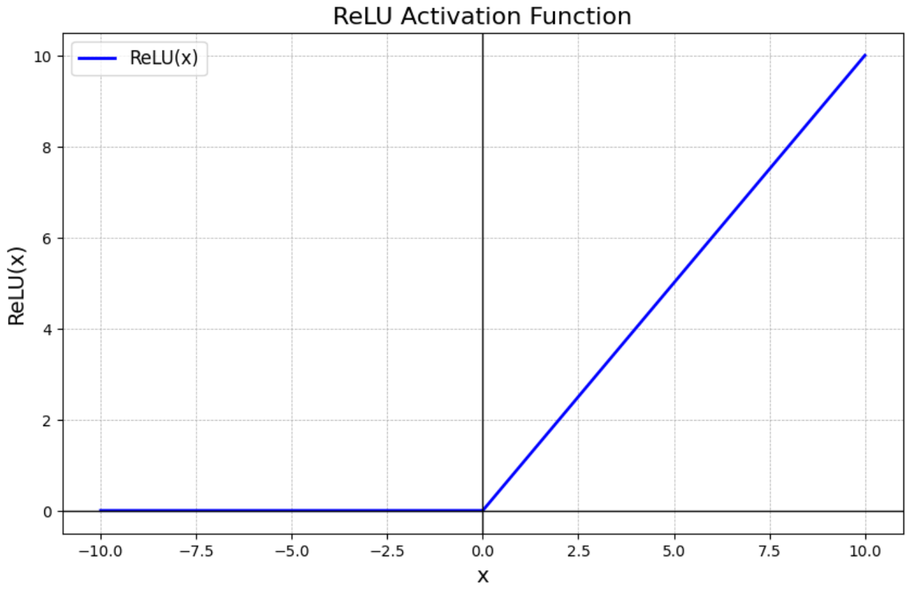
- **Conv2d: extracts spatial features**

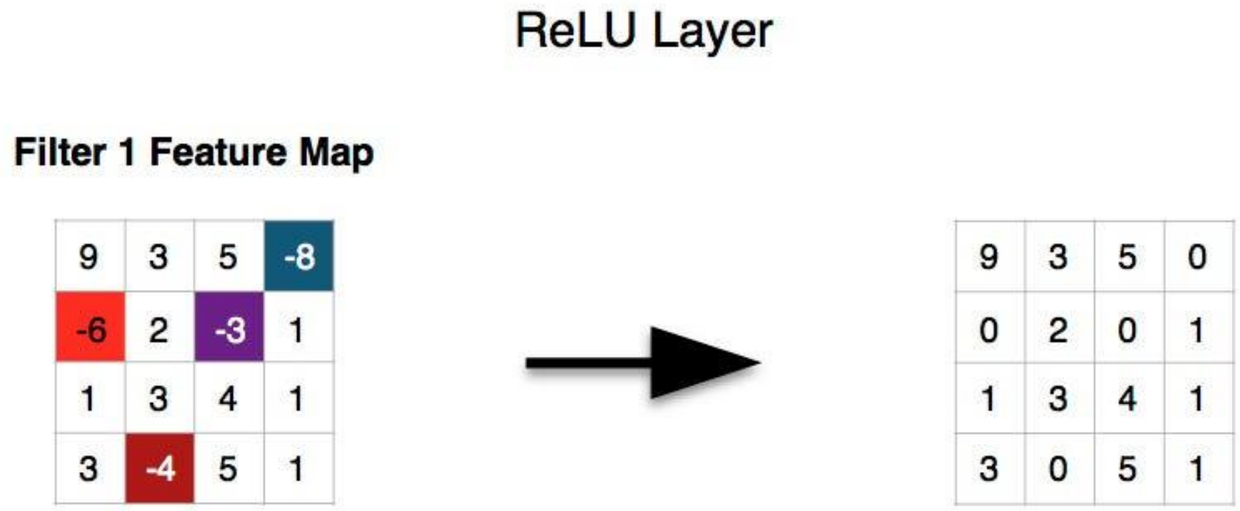




Common Operators in CIFAR-10 Model

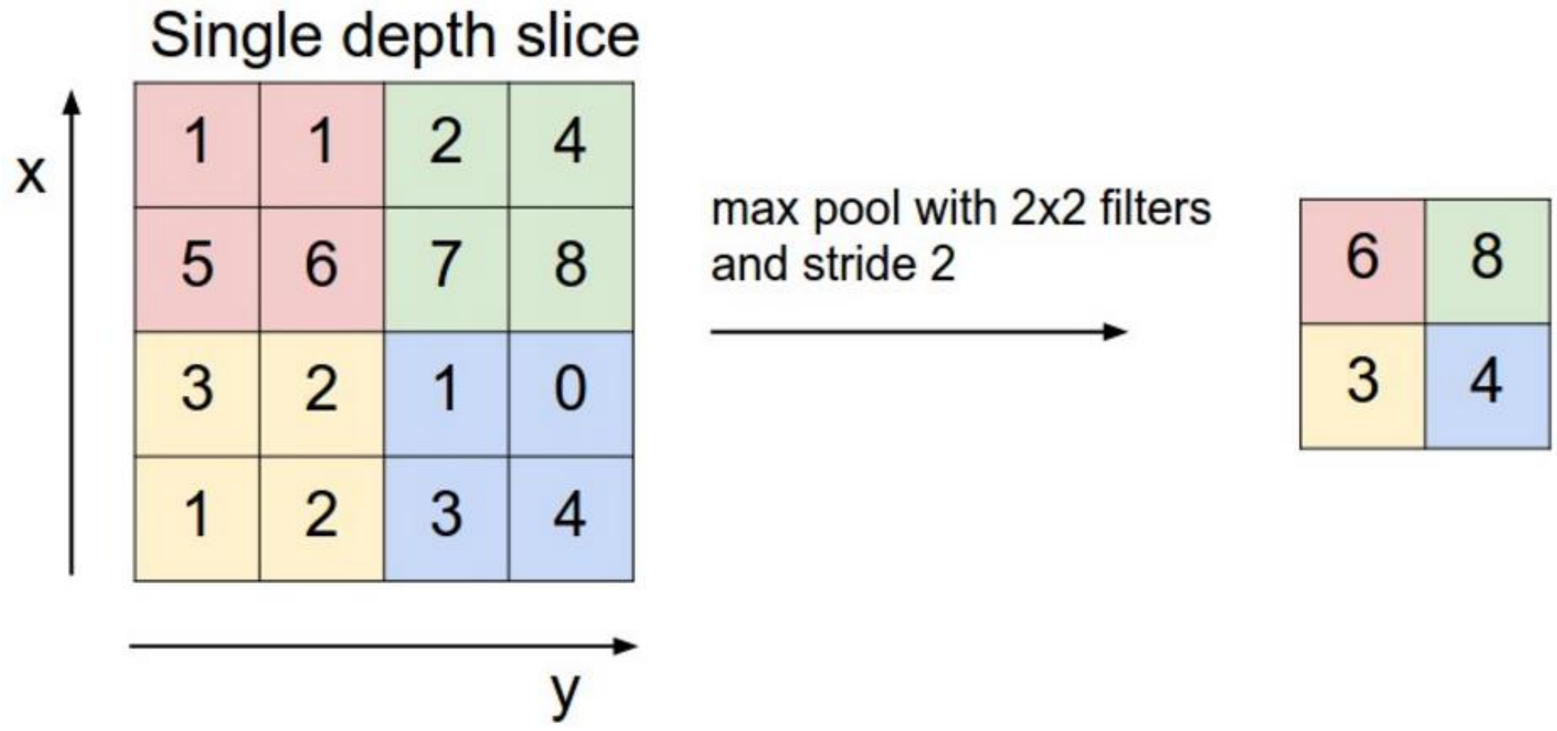
- Conv2d: extracts spatial features
- **ReLU: adds nonlinearity**





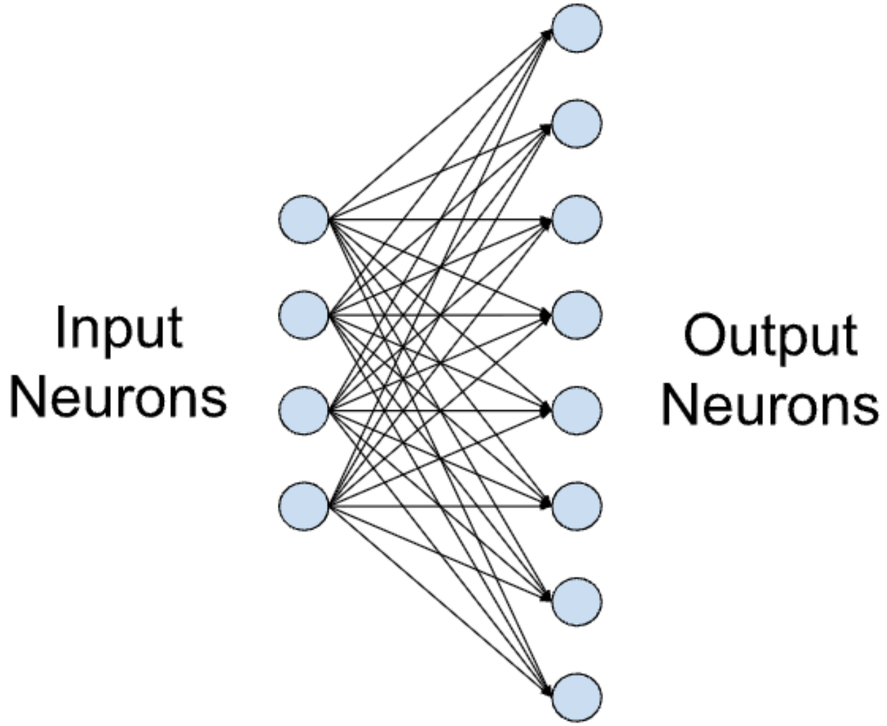
Common Operators in CIFAR-10 Model

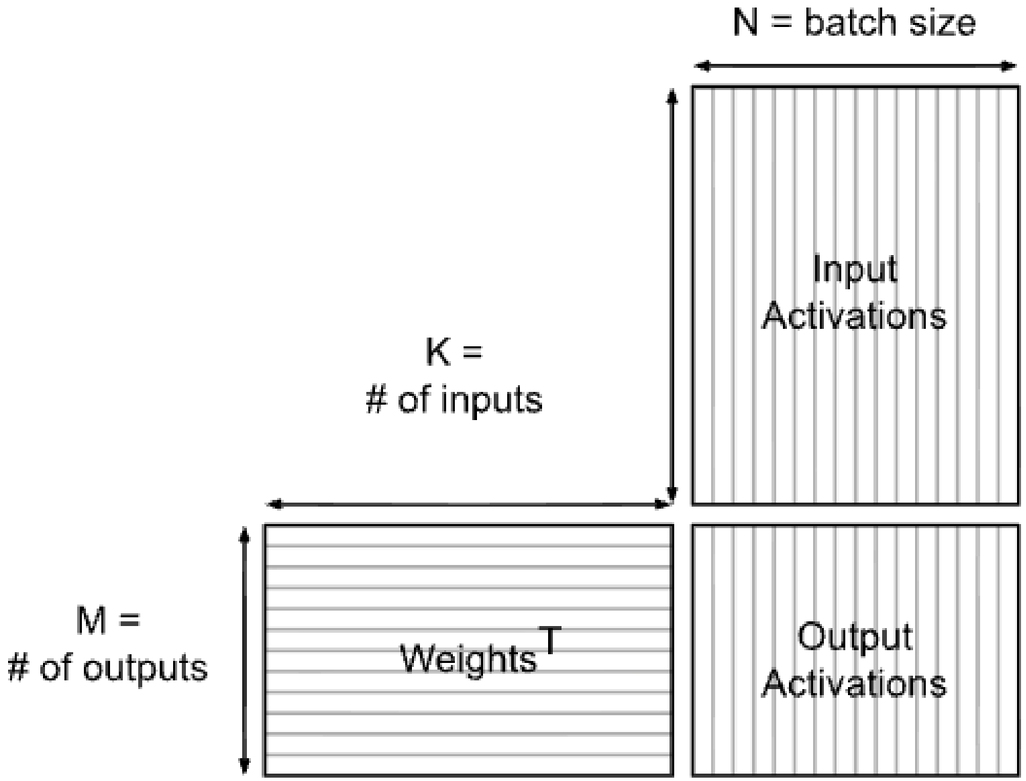
- Conv2d: extracts spatial features
- ReLU: adds nonlinearity
- **MaxPool2d: downsampling**



Common Operators in CIFAR-10 Model

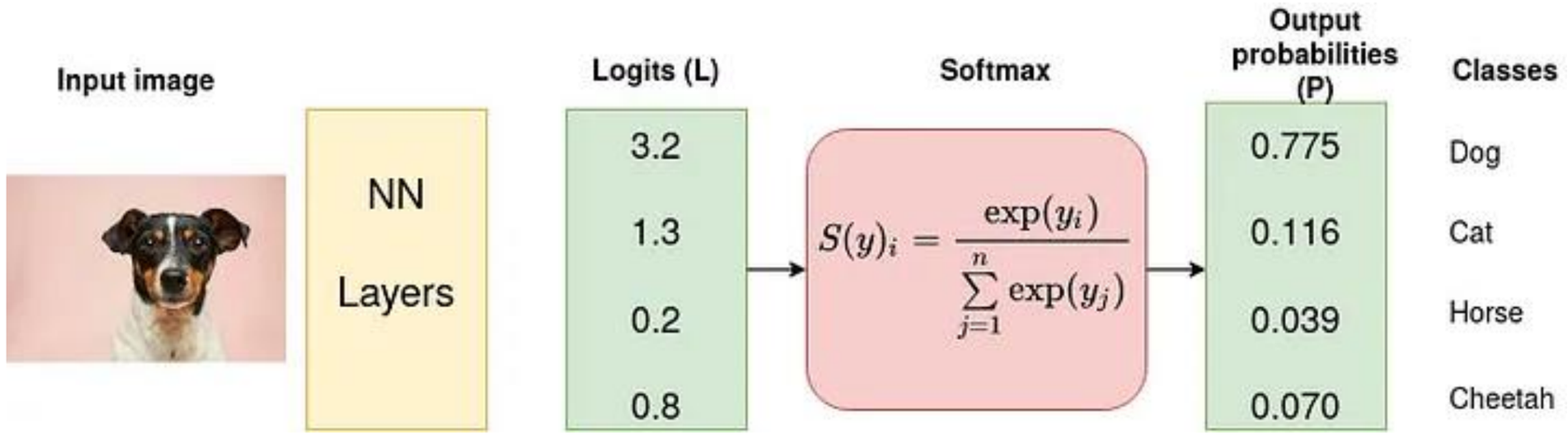
- Conv2d: extracts spatial features
- ReLU: adds nonlinearity
- MaxPool2d: down sampling
- **Linear: fully connected layer**





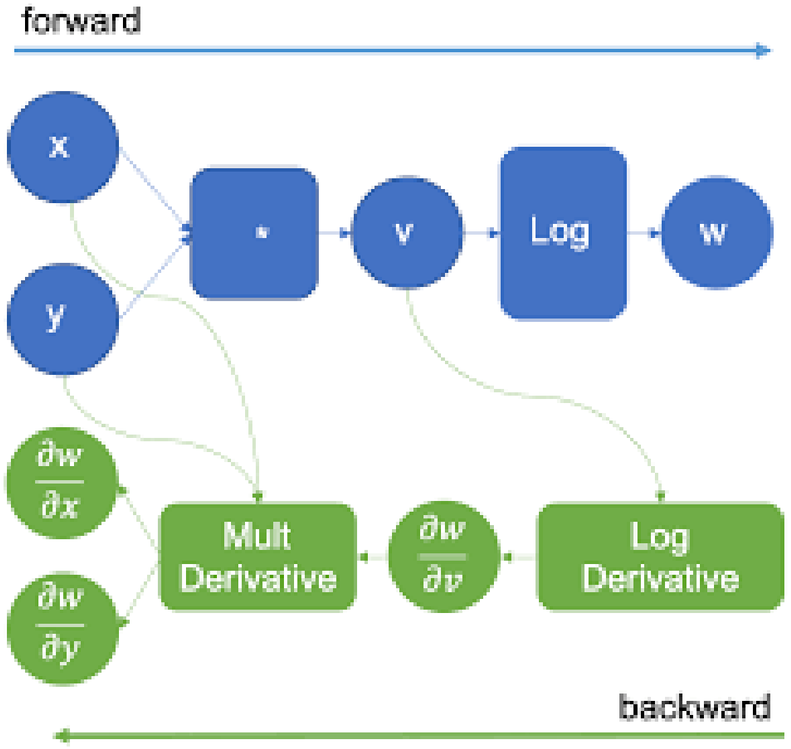
Common Operators in CIFAR-10 Model

- Conv2d: extracts spatial features
- ReLU: adds nonlinearity
- MaxPool2d: downsampling
- Linear: fully connected layer
- **CrossEntropyLoss: combines Softmax + log loss**



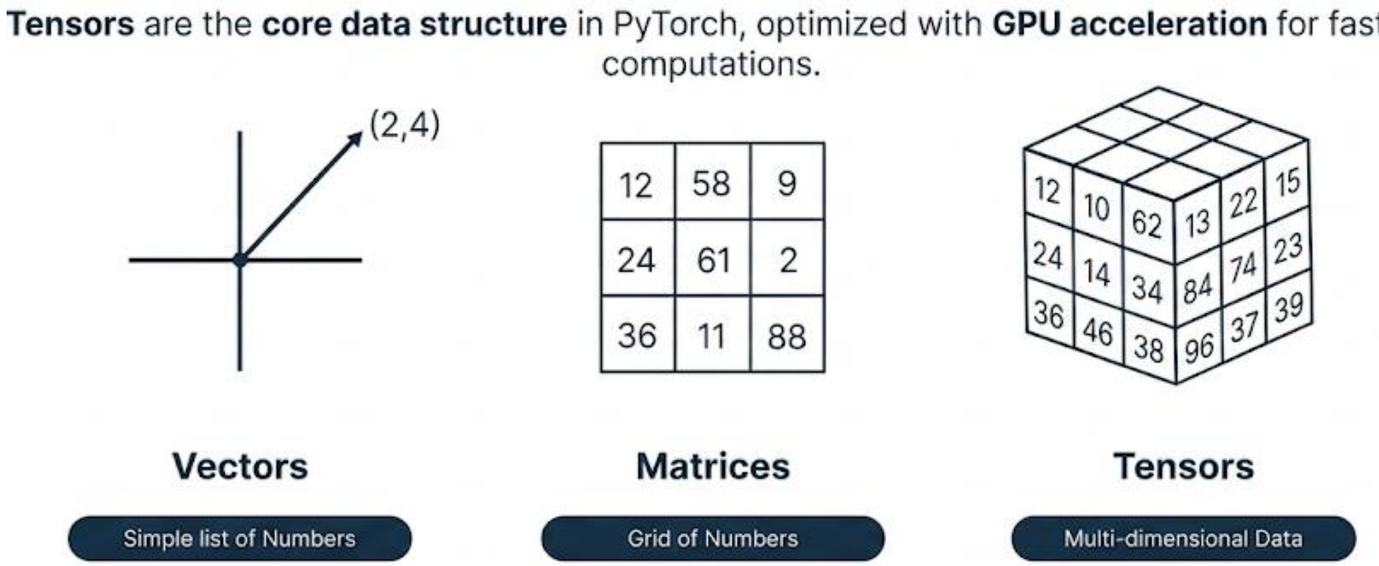
Autograd and Computational Graph

- PyTorch dynamically builds a computation graph as operations run
- Each operator records gradient computation rules
- loss.backward() traverses the graph automatically
- Gradients stored in tensor.grad



Tensors: The Heart of PyTorch

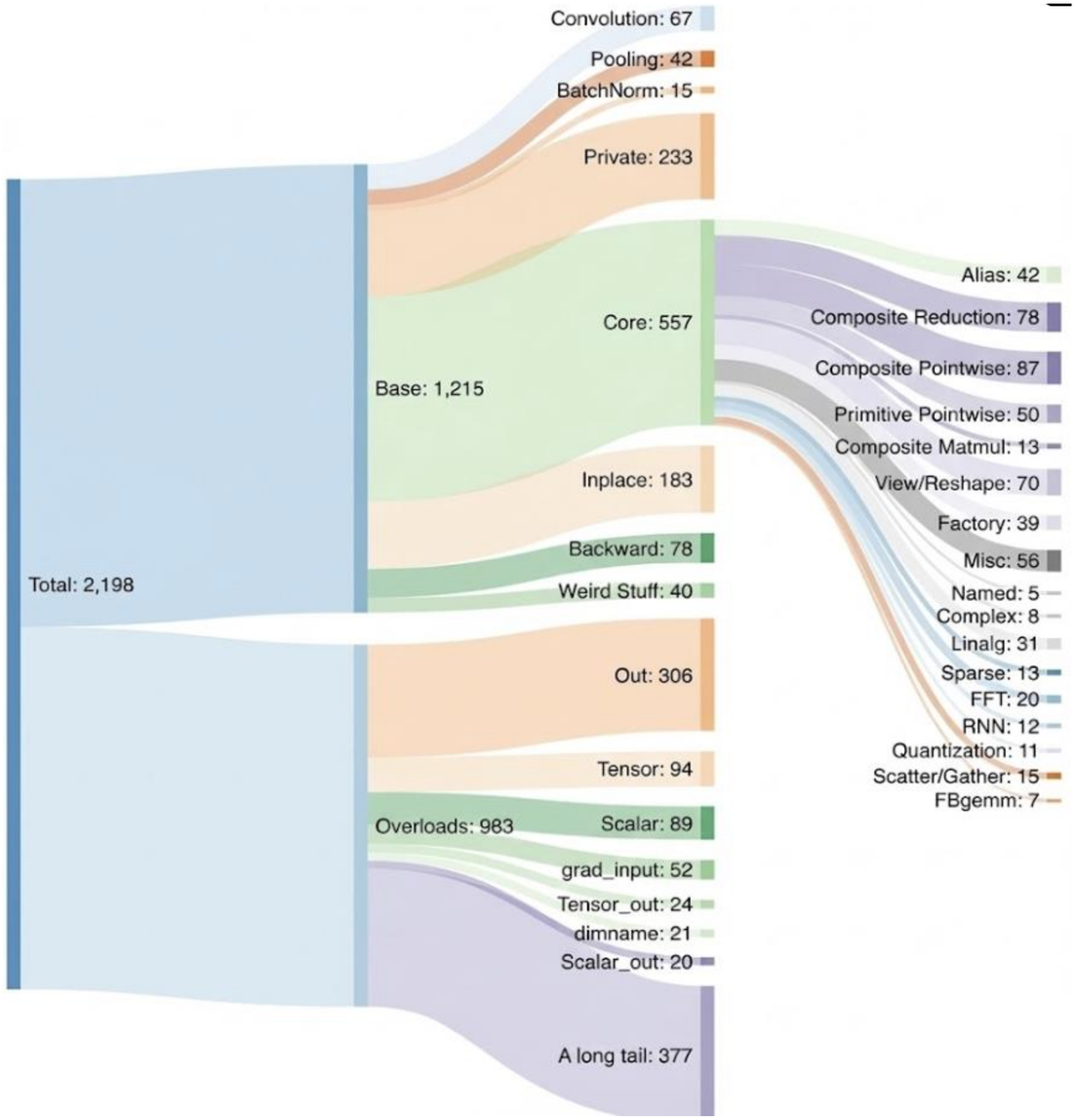
- Similar to NumPy arrays but GPU accelerated
- Support automatic differentiation for deep learning
- Example shapes: Image [3,32,32], Batch [64,3,32,32]
- Common ops: torch.add, torch.mul, torch.matmul, torch.mean, torch.cat



Tensors: The Heart of PyTorch

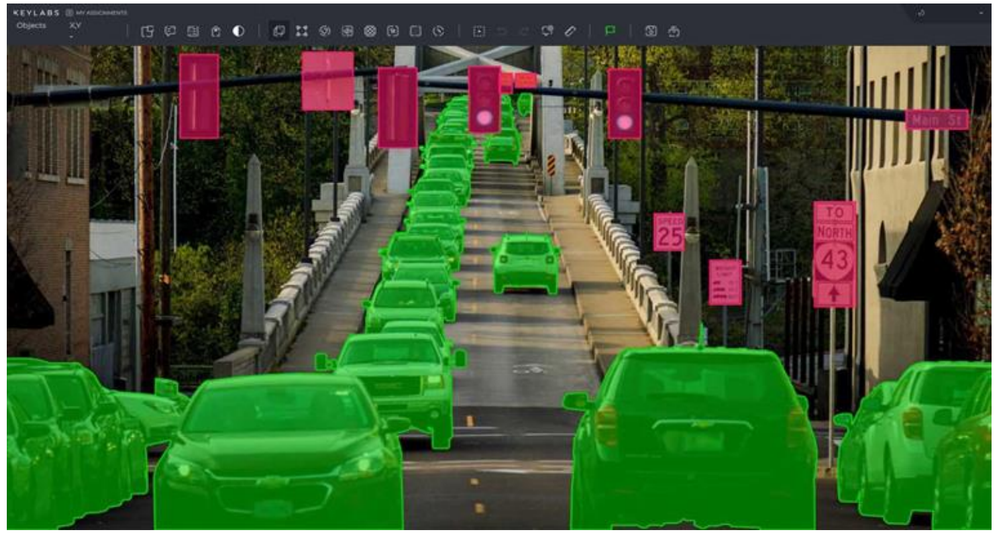
PyTorch Data Types Reference Table

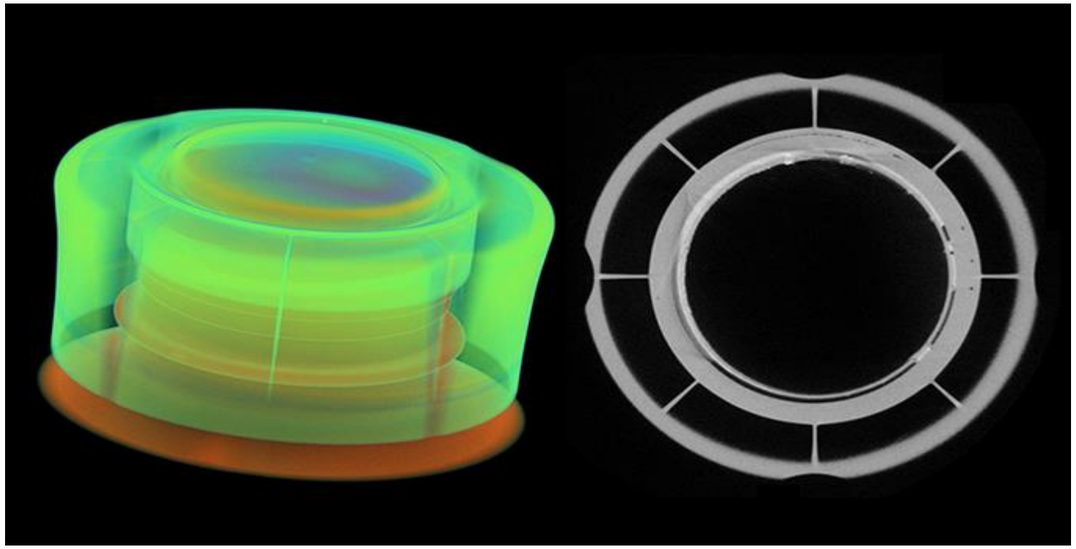
Data type	CPU tensor	GPU tensor
32-bit floating point	torch.FloatTensor	torch.cuda.FloatTensor
64-bit floating point	torch.DoubleTensor	torch.cuda.DoubleTensor
16-bit floating point	torch.HalfTensor	torch.cuda.HalfTensor
8-bit integer (unsigned)	torch.ByteTensor	torch.cuda.ByteTensor
8-bit integer (signed)	torch.CharTensor	torch.cuda.CharTensor
16-bit integer (signed)	torch.ShortTensor	torch.cuda.ShortTensor
32-bit integer (signed)	torch.IntTensor	torch.cuda.IntTensor
64-bit integer (signed)	torch.LongTensor	torch.cuda.LongTensor



Real-World Applications

- Object recognition for autonomous systems.
- Medical diagnostics and defect detection.
- Wildlife monitoring and smart city vision.
- Foundation for detection and segmentation tasks.





RICE UNIVERSITY

yuke.wang@rice.edu