DrDroid

PyTorch Cheatsheet

Initialization and Setup

import torch
Import PyTorch library

torch.manual_seed(seed)
Set random seed for reproducibility

torch.cuda.is_available()
Check if CUDA (GPU) is available

device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
Set device for tensor operations

Tensor Operations

torch.tensor(data)
Create a tensor from data

torch.zeros(shape)
Create a tensor filled with zeros

torch.ones(shape)
Create a tensor filled with ones

torch.randn(shape)
Create a tensor with random values from normal distribution

tensor.to(device)
Move tensor to specified device (CPU/GPU)

tensor.shape
Get tensor dimensions

tensor.dtype
Get tensor data type

tensor.requires_grad_(True)
Enable gradient computation for a tensor

Model Building & Training

torch.nn.Module
Base class for all neural network modules

model.train()
Set model to training mode

model.eval()
Set model to evaluation mode

optimizer.zero_grad()
Reset gradients to zero

loss.backward()
Compute gradients through backpropagation

optimizer.step()
Update model parameters

torch.no_grad()
Context manager to disable gradient calculation

Debugging & Profiling

tensor.detach()
Create a new tensor detached from computation graph

tensor.item()
Get single value from a scalar tensor

torch.autograd.set_detect_anomaly(True)
Enable anomaly detection

torch.autograd.profiler.profile()
Context manager for profiling

torch.autograd.gradcheck()
Check gradients for numerical correctness

Model Saving & Loading

torch.save(model.state_dict(), 'model.pth')
Save model state dictionary

model.load_state_dict(torch.load('model.pth'))
Load saved model state

torch.jit.save(scripted_model, 'model.pt')
Save TorchScript model

torch.jit.load('model.pt')
Load TorchScript model

Distributed Training

torch.distributed.init_process_group()
Initialize distributed training

torch.nn.parallel.DistributedDataParallel()
Wrap model for distributed training

torch.utils.data.distributed.DistributedSampler()
Sampler for distributed training

Memory Management

torch.cuda.memory_allocated()
Get allocated GPU memory

torch.cuda.memory_reserved()
Get reserved GPU memory

torch.cuda.empty_cache()
Free GPU cache memory

tensor.cpu()
Move tensor to CPU