PyTorch Core Modules

Module

- Model Definition & Computation
- Traning & Optimization
- Data Handling & Parallel Loading
- Hardware Acceleration
- Model Optimization & Deployment
- Distributed & Scalable Training

torch
It provides core modules like Tensor, Autograd, nn, Optim, and Data to build, train, and optimize deep learning models efficiently.

torch.onnx
Exports PyTorch models to ONNX format for cross-framework and production deployment.

torch.nn
It provides building blocks like layers, loss functions, and modules to define and structure neural networks.

torch.autogard
Automatically computes gradients for tensors to enable backpropagation.

torch.optim
Provides optimization algorithms (like SGD, Adam) to update model parameters during training.

PyTorch

torch.jit
Converts PyTorch models into optimized, serializable representations for faster execution.

torch.utils.data
Handles datasets and data loading with batching, shuffling, and parallelism.

torch.quantization
Reduces model size and improves inference speed using lower-precision arithmetic.

torch.distributed
Enables distributed and multi-node training for large-scale models.

troch.cuda
Provides GPU acceleration and CUDA support for faster tensor computations.

torch.backend
Controls backend libraries (like cuDNN) to optimize performance and determinism.

troch.multiprocessing
Supports parallel data loading and multi-process training workflows.