PyTorch Learning Roadmap (Beginner to Advanced)

Stage 1: Prerequisites - Python & ML Basics

What You Need to Know

Before diving into PyTorch, make sure you have:

- **Python Basics**: Lists, dictionaries, loops, functions, classes

- **NumPy**: Array operations, broadcasting

- **Math Concepts**: Linear algebra (matrix multiplication, vectors), calculus (derivatives)

- **ML Concepts**: What is a model, training, validation, testing, loss function, gradient descent

Resources

- Python:

  - [W3Schools Python Tutorial](https://www.w3schools.com/python/)

  - [Real Python](https://realpython.com/)

- Math:

  - [Khan Academy - Linear Algebra](https://www.khanacademy.org/math/linear-algebra)

  - [3Blue1Brown - Essence of Linear Algebra (YouTube)](https://www.youtube.com/watch?v=kjBOesZCoqc)

- ML Basics:

  - [Google ML Crash Course](https://developers.google.com/machine-learning/crash-course)

Stage 2: PyTorch Fundamentals

Core Concepts

- **What is PyTorch?**: Deep learning framework like TensorFlow but more pythonic and dynamic.
- **Tensors**:
  - Creation: `torch.tensor`, `torch.zeros`, `torch.ones`
  - Operations: Add, multiply, reshape, slice
- **Autograd**:
  - Automatic differentiation using `requires_grad=True`
- **Building a Simple Neural Network**
- **Loss Functions**: MSE, CrossEntropy
- **Optimizers**: SGD, Adam

Resources

- [PyTorch 60-Minute Blitz](https://pytorch.org/tutorials/beginner/deep_learning_60min_blitz.html)
- [DeepLizard PyTorch YouTube Series](https://www.youtube.com/playlist?list=PLZyvi_9gamL-EE3zQJbU5N6gqum0U2b5x)
- [Official Docs: Beginner Tutorials](https://pytorch.org/tutorials/beginner/index.html)

Practice

- Implement linear regression
- Use `autograd` to compute gradients manually

Stage 3: Building Neural Networks with PyTorch

Key Topics

- **Linear Regression and Logistic Regression** using PyTorch

- **Neural Networks**: Building from scratch using `nn.Module`

- **Training Loops**:

  - Epochs, mini-batches, forward + backward pass

- **Evaluation**:

  - Validation loop, metrics (accuracy, precision)

- **Saving and Loading Models**:

  - `torch.save()`, `torch.load()`, `model.eval()`

Resources

- [PyTorch Official - Training Classifiers](https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html)

- [Aladdin Persson YouTube](https://www.youtube.com/c/AladdinPersson)

Practice

- MNIST digit classification

- Use GPU for training if available

Stage 4: Intermediate Deep Learning

What to Learn

- **CNNs (Convolutional Neural Networks)**:

  - Layers: Conv2d, MaxPool2d, ReLU, Flatten, Fully Connected

  - Applications: Image classification, object detection

- **RNNs & LSTMs**:
  - Sequence data: Time series, text
- **Custom Datasets & Transforms**:
  - `torch.utils.data.Dataset`, `DataLoader`, `torchvision.transforms`
- **Transfer Learning**:
  - Load pretrained models like ResNet, fine-tune

Resources
- [PyTorch Transfer Learning](https://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html)
- [FastAI Deep Learning Course](https://course.fast.ai/)

Practice
- CIFAR-10 image classifier with CNN
- Stock price prediction with LSTM
- Fine-tune ResNet for face detection

Stage 5: Advanced Deep Learning in PyTorch

Master These Topics
- **Transformers**:
  - Self-attention, BERT, GPT
- **Attention Mechanism**:
  - Used in sequence models, image captioning
- **GANs (Generative Adversarial Networks)**:
  - Generator vs. Discriminator

- **Object Detection Models**:

  - SSD, YOLOv5 with `torchvision` or external libraries

- **Model Deployment**:

  - TorchScript, ONNX, export models

Resources

- [Hugging Face Transformers Course](https://huggingface.co/learn)

- [Deep Learning with PyTorch Book](https://www.manning.com/books/deep-learning-with-pytorch)

- [Papers With Code](https://paperswithcode.com/)

Projects

- Build your own chatbot using transformers

- Create a GAN that generates fake handwritten digits

- Train a ViT (Vision Transformer) on image data

This roadmap sets you up with a solid progression to go from zero to advanced-level deep learning using PyTorch. Each stage includes theory, code, and project ideas. Once you're done with this path, youll be ready to work on real-world applications and even publish your own models.