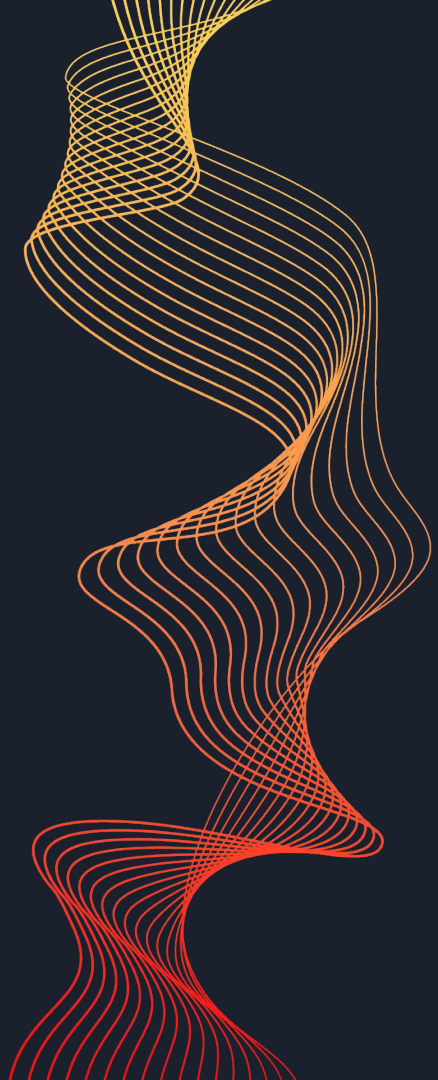
Deep Learning in PyTorch

This presentation provides an overview of deep learning in PyTorch and its applications.

As well as integration of pytorch with matplotlib to visualize deep learning models

A presentation by

Aaryadev Ghosalkar



What is PyTorch

PyTorch, developed by Facebook's AI Research lab (FAIR) in 2016, is an open-source framework for deep learning. Its dynamic data flow, real-time model adjustments, and seamless integration with Python syntax and ecosystem which makes it a top choice for deep learning. It offers an extensive toolkit for building, training, and deploying deep learning models.

https://pytorch.org/features/

PyTorch Basics

01 Tensors

Tensors are fundamental building blocks in PyTorch, similar to arrays in numerical computing libraries. They can represent scalars, vectors, matrices, and higher-dimensional arrays

02 Models

PyTorch provides a special class called "module" that allows us to create machine learning models. This class makes it easy to define the structure and behavior of our models

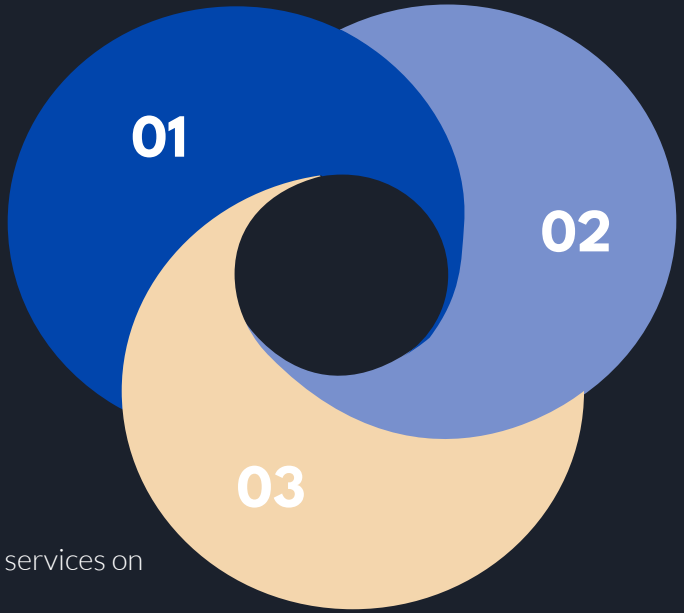
03 Saving the model

Once we've trained our model, we can save it for later use. PyTorch offers simple methods to save the entire model's configuration and learned patterns so that we can use it to make predictions without retraining.

Where is PyTorch being used?

At Tesla

Pytorch is being used for autopilot systems



At Apple

Apple Neural Engine uses pytorch transformers

At Microsoft

The language modeling services on Azure

Pytorch at tesla

Tesla relies on PyTorch to develop its autonomous driving systems, employing computer vision through convolutional neural networks (CNNs) from eight vehicle cameras. The company uses shared Hydra Nets for multi-task settings, enabling predictions for road features and environment tags. PyTorch suitability for multitasking training and model-parallel methods makes it ideal for this complex setup. Tesla trains around 48 networks, performing 1,000 predictions, taking 70,000 GPU hours for autopilot development. PyTorch role in handling this intricate training showcases its importance in achieving Tesla's self-driving objectives.

https://www.youtube.com/watch?v=oBklltKXtDE

PyTorch at Microsoft

Since 2018, Microsoft has integrated PyTorch for their language modeling service, addressing issues with other frameworks' slow transitions and inconsistencies. Utilizing an in-house toolkit, Microsoft crafted custom tasks and architectures on PyTorch. They introduced PyTorch Enterprise on Azure, a support service for PyTorch users. PyTorch is pivotal for enhancing Microsoft 365, Bing, and Xbox through innovative models. As a prominent PyTorch contributor, Microsoft's contributions include the PyTorch Profiler.

https://azure.microsoft.com/en-us/resources/developers/pytorch/

PyTorch at Apple

Apple leverages PyTorch in diverse ways, including GPU-accelerated training on Mac hardware for faster model development. Collaborating with Apple's Metal engineering team, PyTorch enables local machine learning workflows with swift model training. Additionally, Apple employs PyTorch to deploy Transformers on the Apple Neural Engine (ANE), an energy-efficient engine for ML inference on Apple silicon. Apple's open-source reference PyTorch implementation of the Transformer architecture allows seamless deployment of advanced models on Apple devices. This optimized implementation aids developers in conserving app memory, maintaining app responsiveness, and preserving device battery life.

https://machinelearning.apple.com/research/neural-engine-transformers

Integrating PyTorch with the existing ecosystem

PyTorch works well with NumPy and Pandas, which data scientists and researchers are already acquainted with these tools. This blending makes handling data, getting it ready, and digging into analysis while dealing with deep learning models a breeze.

PyTorch offers a seamless bridge between NumPy arrays and its own tensors, meaning you can make the most of what you already know without any stress. Converting PyTorch tensors to NumPy arrays is also a walk in the park.

Even more, tensors play well within data frames, and the process of converting back and forth between data frames and tensors is refreshingly straightforward.

Another cool thing which you can explore is integrating PyTorch with the Apache Spark and Hadoop or BigQuery tools, to run Deep learning models on really large distributed datasets

Thank you for your time and attention 🙂