PyTorch Tutorial: Part I

Yunfei Teng

yt1208@nyu.edu

Department of Electrical and Computer Engineering
New York University Tandon School of Engineering

Feb 22, 2021

Outline

What is a neural network?

- "...a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs.
  In "Neural Network Primer: Part I" by Maureen Caudill, AI Expert, Feb. 1989.
- How to decompose a large neural network into smaller parts? One possible way to consider it: Neuron $\rightarrow$ Layer $\rightarrow$ Network.
- **All the nodes of neural networks should be differentiable in order to use Backpropagation**. Most popular optimization algorithms for deep learning models are gradient-based methods.
- **It can be considered as a tool which is able to approximate any function ("universal approximators")**. We will use an example of regression problem $f(w;x) = w_0 + w_1 x + w_2 x^2 \cdots w_n x^n$ to illustrate this statement later.

Why is it hard to train a neural network?

- It's non-convex? Most likely yes.
  New optimization methods and architectures are invented to alleviate this problem.
    - Entropy-SGD: `https://arxiv.org/pdf/1611.01838.pdf`
    - ResNet: `https://arxiv.org/pdf/1512.03385.pdf`
- It's a black box? Maybe.
  New visualization tools helped us to understand it.
    - Visualizing and Understanding Convolutional Networks: `https://cs.nyu.edu/~fergus/papers/zeilerECCV2014.pdf`
    - Visual BackProp: `https://arxiv.org/pdf/1611.05418.pdf`
- The reasons are complex: Learning rate, optimizer, initialization, data, pipeline, model, architecture, etc. All these factors count for results.

How to build a neural network in PyTorch?

In PyTorch the smallest unit is layer and neurons are implicitly included in layers.
Each layer has its own parameters. These parameters belong to autograd. Variable class with attributes of *data* and *grad*. A layer's weights and gradients are saved in *data* and *grad* respectively.

Remember Neural Networks should be differentiable? It would be wonderful if we have something doing this job automatically. PyTorch can do it!
PyTorch autograd tutorial: `http://pytorch.org/tutorials/beginner/blitz/autograd_tutorial.html`

In the next slide we will discuss three important classes in PyTorch: torch.utils.data, torch.nn and torch.optim.

How to build a neural network in PyTorch?

torch.nn class:

- Basic layers:
  - Linear layers: Apply a linear transformation over an input.
  - Convolution layers: Apply a convolution over an input.
  - Activation layers: Introduce non-linearity.
- Other important layers:
  - Pooling layers: Reduce the spatial size of the representation.
  - Normalization layers: Reduce the Covariate Shift.
  - Dropout layers: Prevent overfitting.
- Loss functions: Define cost for difference between output and target.

Related document: `http://pytorch.org/docs/master/nn.html`

torch.utils.data: Load and process data.

Related document: `http://pytorch.org/docs/master/data.html`

torch.optim: Using different optimization methods decides how neural networks make use of these gradients.

Related document: `http://pytorch.org/docs/master/optim.html`

Introduction to PyTorch and its advantage

Official Introduction:

- It's a **Python** first package.
- It's not only designed for neural networks, but also works as a **GPU-ready Tensor library**.
- Compared with Tensorflow and other packages, PyTorch is more friendly to researchers since the codes are more **concise**.
- The graph is **built on the fly**, so you can change the architecture at any time!
- You can **switch between CPU and GPU** easily.

How could you get more from learning PyTorch?

- Read documents and practice by solving problems
- Follow PyTorch GitHub and Forum.
- Try to contribute to the community.

Let's do it!

You can download the codes of this tutorial from my Github: `https://github.com/yunfei-teng/PyTorch-Tutorial-Spring-2021`.

1. Regression for polynomial functions
   - Understand why neural networks are "universal approximators".
2. MNIST digit classification
   - Learn to design a small neural network.

You can find support from PyTorch websites:

- To learn and understand PyTorch quickly `http://pytorch.org/tutorials/beginner/deep_learning_60min_blitz.html`
- For more examples, go to official Github: `https://github.com/pytorch/examples`
- For more specific explanations, go to official documents: `http://pytorch.org/docs/master/index.html`