Applied Deep Learning

PyTorch Tutorial

葉浩同

Created in 2021

Deep Learning Frameworks

1. Format data into tensors and construct datasets.
2. Create models with predefined layers.
3. Calculate gradient and update network parameters.
4. Run with hardware accelerators or distributed systems.

2019

MIULAB

Deep Learning Framework

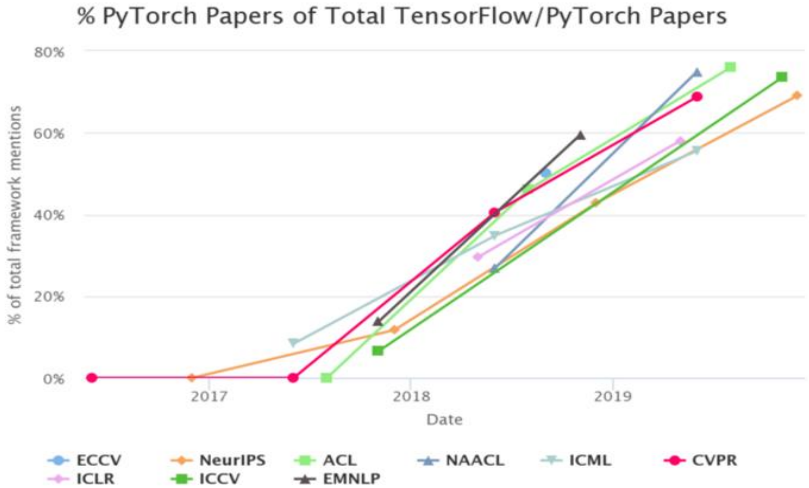
Today

TensorFlow vs PyTorch

- Job Listings

Framework	Indeed	Monster	Simply Hired	LinkedIn	Mean
TensorFlow	66.3%	66.8%	65.5%	67.7%	**66.6%**
PyTorch	33.7%	33.2%	34.5%	32.3%	**33.8%**

- Research



Source: https://towardsdatascience.com/is-pytorch-catching-tensorflow-ca88f9128304

Tensor

```
tensor([[[ 72,  83,  97,  ..., 212, 206, 209],
         [ 72,  80,  92,  ..., 211, 187, 184],
         [ 78,  81,  88,  ..., 193, 168, 175],
         ...,
         [132, 110, 115,  ...,  46,  46,  45],
         [135, 130, 136,  ...,  45,  43,  41],
         [127, 127, 126,  ...,  42,  39,  36]],

        [[ 67,  78,  92,  ..., 220, 214, 217],
         [ 67,  75,  87,  ..., 219, 195, 192],
         [ 73,  76,  83,  ..., 200, 175, 182],
         ...,
         [120,  98, 102,  ...,  41,  41,  40],
         [123, 118, 123,  ...,  40,  38,  36],
         [115, 115, 113,  ...,  37,  34,  31]],

        [[ 73,  84,  98,  ..., 233, 227, 230],
         [ 73,  81,  93,  ..., 232, 208, 205],
         [ 79,  82,  89,  ..., 216, 191, 198],
         ...,
         [ 98,  76,  83,  ...,  47,  47,  46],
         [101,  96, 104,  ...,  46,  44,  42],
         [ 93,  93,  94,  ...,  43,  40,  37]]], dtype=torch.uint8)
```

- Similar to NumPy ndarray, but can be used on GPUs. Tensors also store gradients.
- Document Tutorial

Dataset

- Dataset
  - Store all your data samples.
- Data loader
  - Determine how to load a batch from the dataset.
  - Some settings include: batch size, shuffle, num_workers.
- Document Tutorial

```
from torch.utils.data import Dataset, DataLoader

class NumbersDataset(Dataset):
    def __init__(self, n):
        self.samples = list(range(n))

    def __len__(self):
        return len(self.samples)

    def __getitem__(self, idx):
        return self.samples[idx]


dataset = NumbersDataset(100)
dataloader = DataLoader(dataset, batch_size=32)
for d in dataloader:
    print(d)
```

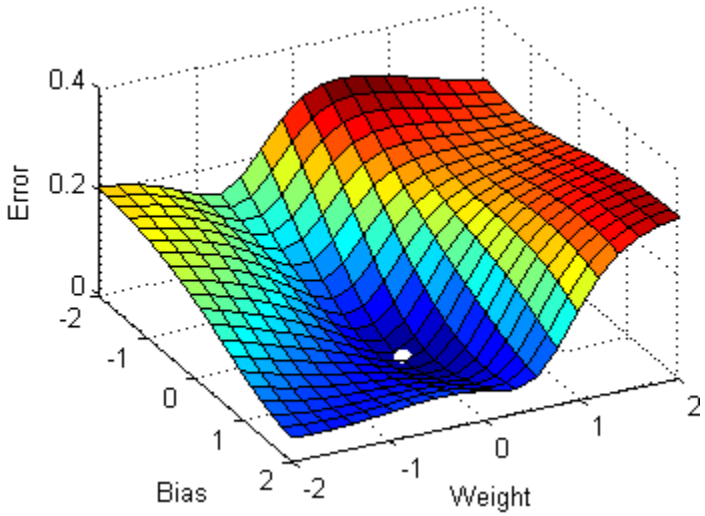
Model

- `nn.Module` is the base class for all neural network modules.
- Define a model by creating a subclass of `nn.Module`.
  - `__init__` declares all the layers in the model.
  - `forward` describes the computation graph.
- `torch.nn.XXX` provides a variety of layers.
- Document Tutorial

```python
import torch
import torch.nn as nn

class NeuralNetwork(nn.Module):
    def __init__(self, input_dim, output_dim):
        super(NeuralNetwork, self).__init__()

        self.linear = nn.Linear(input_dim, output_dim)

    def forward(self, x):
        logits = self.linear(x)

        return logits


x = torch.randn(1, 100)
network = NeuralNetwork(100, 1)
prediction = network(x)
```

Gradient Descent

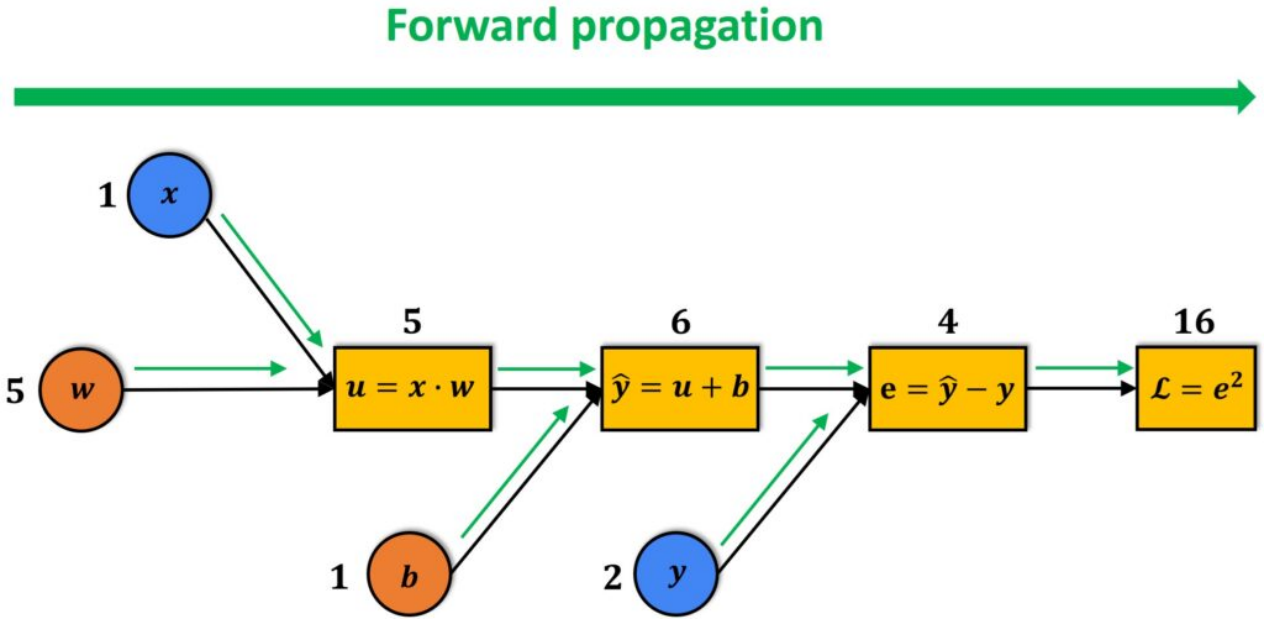
- Find a local minima of the error (loss) function.
- Example of a loss function
  - L1(y, prediction) = |y - prediction|
- Use the gradient of the loss function to find the direction to descend.



Source: https://medium.com/@hakobavjyan/stochastic-gradient-descent-sgd-10ce70fea389

Computation Graph

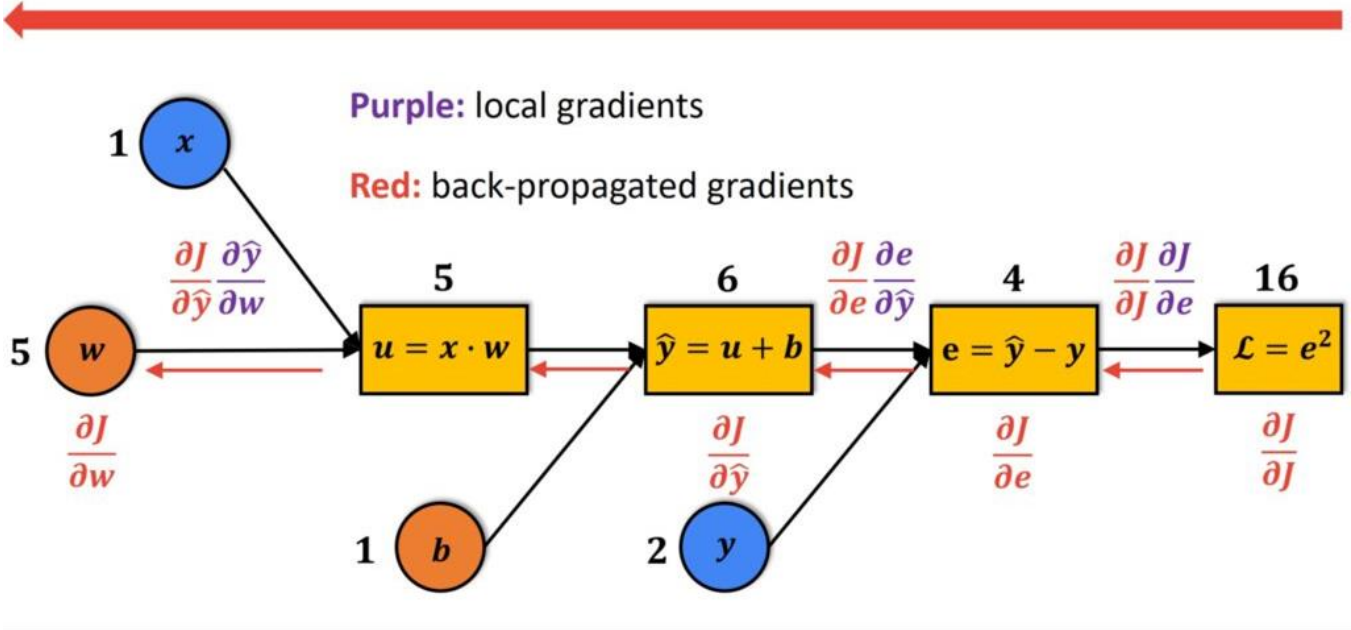
- `predictoin = network(x)`



Source: http://datahacker.rs/004-computational-graph-and-autograd-with-pytorch/

Back Propagation

- `loss.backward()`



Source: http://datahacker.rs/004-computational-graph-and-autograd-with-pytorch/

Optimizer

- Update model parameters with gradients.
- `torch.optim.XXX` already implements a variety of optimization algorithms.
- Document Tutorial

```
import torch.nn as nn
import torch.optim as optim

loss_fn = nn.CrossEntropyLoss()
learning_rate = 0.0001
optimizer = optim.SGD(network.parameters(), lr=learning_rate)

loss = loss_fn(prediction, y)
optimizer.zero_grad()  # clear gradients
loss.backward()        # calculate new gradients
optimizer.step()       # update network parameters
```

Other Resources

- A more detailed guide on all the topics shown in this slides: https://pytorch.org/tutorials/beginner/basics/intro.html
- https://pytorch.org/tutorials/
- https://pytorch.org/docs/
- https://www.tensorflow.org/tutorials/
- https://www.tensorflow.org/api_docs/python/tf/