Библиотека для автоматического дифференцирования и вычислений на GPU PyTorch

Яворская Мария

МГУ имени М. В. Ломоносова, факультет ВМК, кафедра ММП

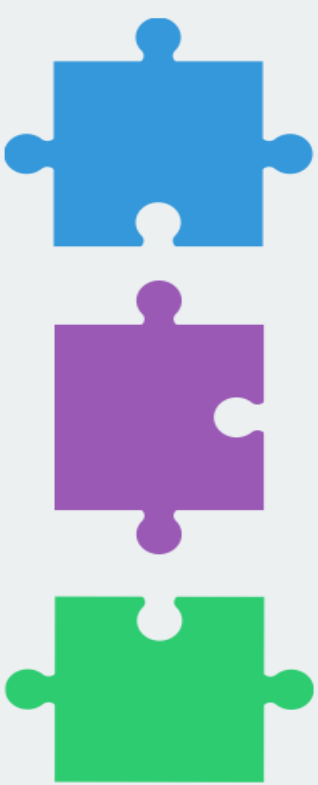
5 декабря 2017 г.

Проблематика

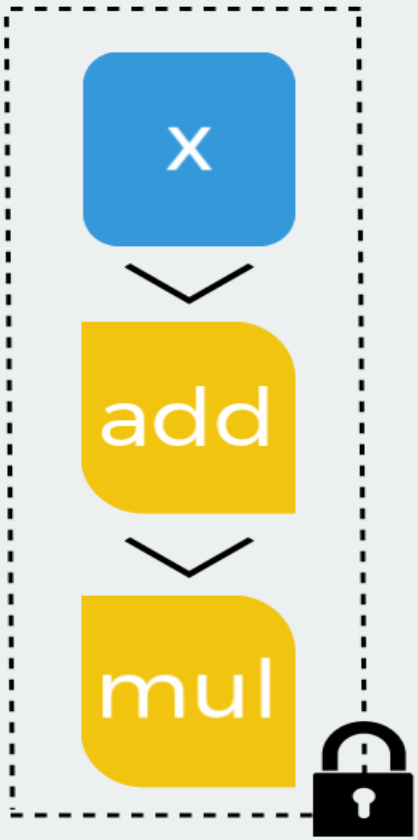
Фреймворк глубокого обучения должен уметь делать:

- Определять граф вычислений;
- Дифференцировать граф вычислений;
- Вычислять его.

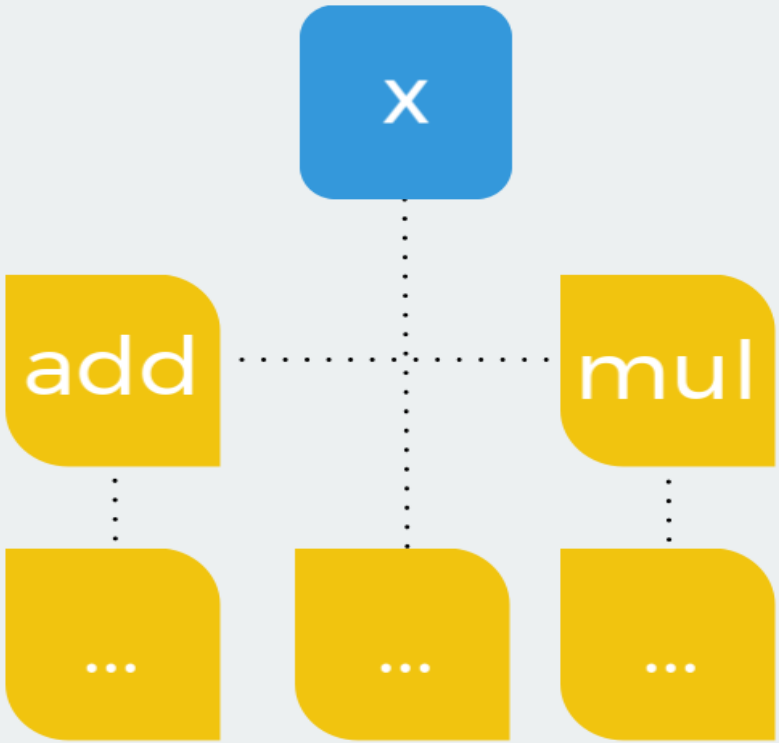
Категории фреймворков



Caffe
Caffe2
CNTK
Kaldi
DL4J
Keras



Theano
TensorFlow
MXNet



Torch
PyTorch

Реализация через NumPy

```python
import numpy as np

def MyNetworkForward(weights, bias, x):
    h1 = weights @ x + bias
    a1 = np.tanh(h1)

    return a1

y = MyNetworkForward(weights, bias, x)
loss = np.mean((y - y_hat) ** 2)
```

Реализация через PyTorch

```
1  import torch
2
3  def MyNetworkForward(weights, bias, x):
4      h1 = weights @ x + bias
5      a1 = torch.tanh(h1)
6
7      return a1
8
9  weights = weights.cuda()
10 bias = bias.cuda()
11 x = x.cuda()
12
13 y = MyNetworkForward(weights, bias, x)
14 loss = torch.mean((y - y_hat) ** 2)
15
16 loss.backward()
```

Создание тензора

```
>>> torch.FloatTensor()
[torch.FloatTensor with no dimension]
```

```
>>> a = torch.IntTensor([[1, 2], [3, 4]])
>>> a

 1  2
 3  4
[torch.IntTensor of size 2x2]
```

```
>>> torch.FloatTensor(2, 3)

 2.9621e-25  4.5687e-41  2.9621e-25
 4.5687e-41  4.4842e-44  7.1466e-44
[torch.FloatTensor of size 2x3]
```

Индексирование

```
>>> a = torch.IntTensor([[1, 2], [4, 5]])
>>> a[0, 1]

 2
```

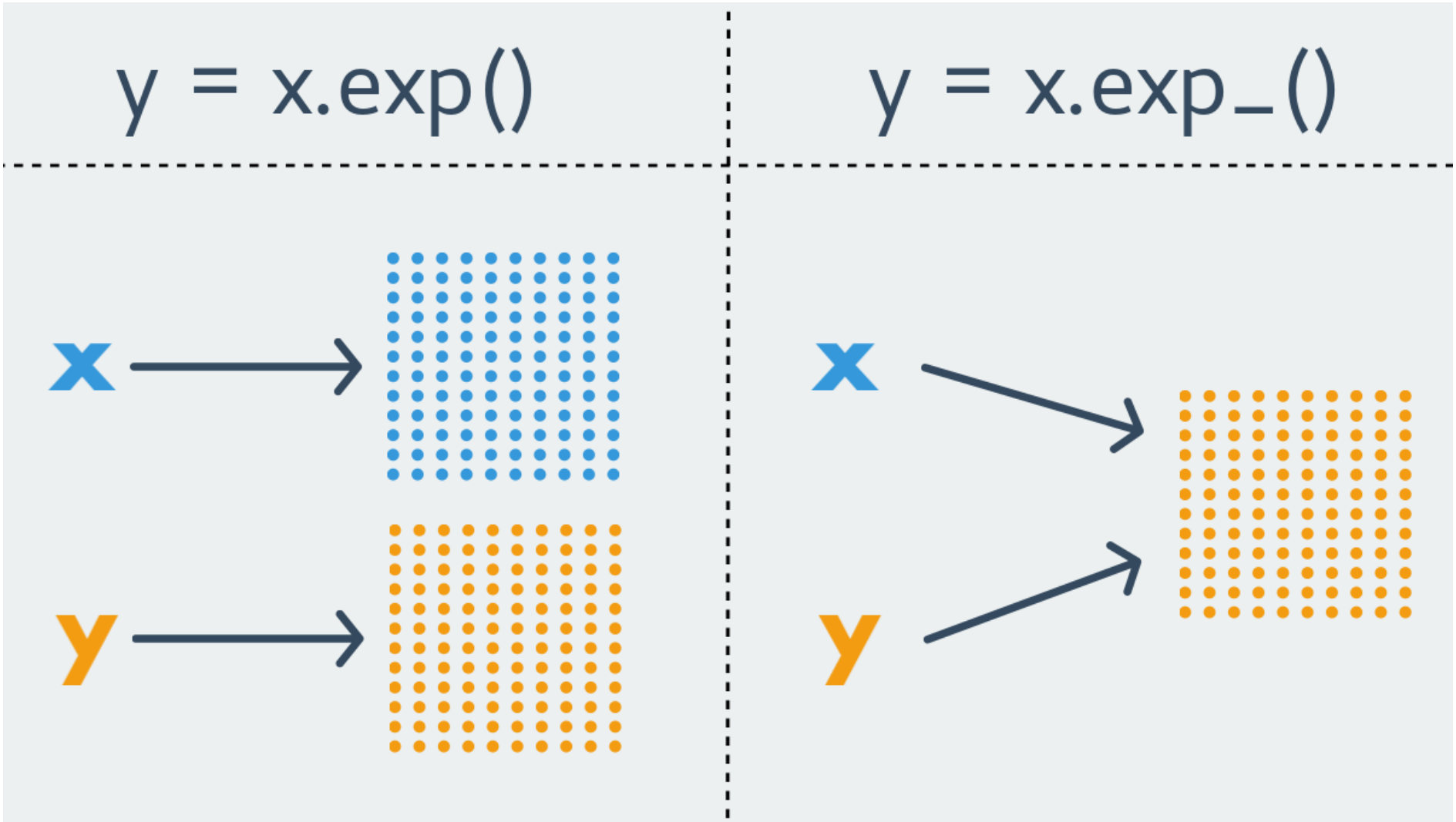
```
>>> a[torch.LongTensor([0])]

 1  2
[torch.IntTensor of size 1x2]
```

```
>>> a[torch.ByteTensor([[1,0], [0, 1]])]

 1
 5
[torch.IntTensor of size 2]
```

Операции над тензорами



Операции над тензорами

Основные операции над тензорами:

- Функции инициализации;
- Математические операции;
- Из NumPy и обратно;
- Broadcasting.

Broadcasting

```
# broadcastable
>>> x = torch.FloatTensor(5, 7, 3)
>>> y = torch.FloatTensor(5, 7, 3)

# broadcastable
>>> x = torch.FloatTensor(5, 3, 4, 1)
>>> y = torch.FloatTensor(   3, 1, 1)

# not broadcastable
>>> x = torch.FloatTensor(5, 2, 4, 1)
>>> y = torch.FloatTensor(   3, 1, 1)

# not broadcastable
>>> x = torch.FloatTensor()
>>> y = torch.FloatTensor(2, 2)
```

CUDA

```
>>> import torch
>>> a = torch.FloatTensor(10000, 10000).uniform_()
>>> b = torch.FloatTensor(10000, 10000).uniform_()
>>> c = a.cuda().mul_(b.cuda()).cpu()
```

Автоматическое дифференцирование

Нужно продифференцировать:

$$f(x_1, x_2) = x_1 x_2 + x_1^2.$$

Переобозначим:

$$\begin{aligned} w_1 &= x_1, \\ w_2 &= x_2, \\ w_3 &= w_1 w_2, \\ w_4 &= w_1^2, \\ w_5 &= w_3 + w_4. \end{aligned}$$

Заключение

$$\text{PyTorch} = \text{NumPy} + \text{CUDA} + \text{AD}.$$

Полезные ссылки:

- Официальный сайт;
- PyTorch — ваш новый фреймворк глубокого обучения;
- PyTorch in 5 Minutes.