

Learning From Data

Guidance to Programming Assignment 2

Feng Zhao zhaof17@mails.tsinghua.edu.cn

10/23/2020

Overview

- ▶ Rank 2 tensor
- ▶ Autodiff and multilayer neural network
 - ▶ Terminology and API
 - ▶ Implementation detail
 - ▶ Examples
- ▶ Practical guide

Rank 2 tensor

- ▶ another name for matrix
- ▶ vector and scalar: special rank 2 tensor
- ▶ shape method

```
1 | from lfdnn import tensor, operator
2 | a = tensor([3, 4], 't')
3 | print(a.shape)
```

- ▶ concatenation by operator
- ▶ delayed evaluation

```
4 | import numpy as np
5 | b = operator.relu(a)
6 | feed = {'t': np.random.normal(size=[3, 4])}
7 | print(b.eval(feed))
```

Autodiff and multilayer neural network

- ▶ Autodiff: Automatic differentiation
- ▶ small h : $\frac{f(x+h)-f(x)}{h}$
- ▶ use chain rule

```
8 | print(b.differentiate(a, feed))
9 | print(a.back(b, feed))
```

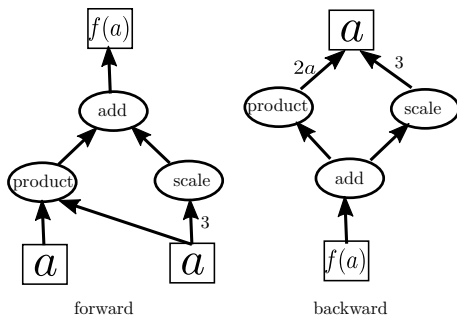
- ▶ forward pass: `b.eval`
- ▶ backward pass: `b.back`

Logistic model

```
10 | w = tensor([4, 1], 'w')
11 | b = tensor([1, 1], 'b')
12 | h = operator.add(operator.matmul(a, w), b)
13 | y = operator.sigmoid(h)
14 | feed.update({'w': np.ones([4, 1]),
15 |             'b': np.array([[2]])})
16 | y.eval(feed)
```

Implementation details

$$f(a) = a^2 + 3a$$



```
1 from lfdnn import tensor
2 class add(tensor):
3     def _eval(self, feed):
4         pass
5     def _derivative(self, feed, input, target):
6         pass
```

Examples

Implement matrix multiplication `matmul(A,B)`

▶ `A = self.input_list[0]`

▶ `B = self.input_list[1]`

```
def _eval(self, feed):  
    return A @ B
```

Suppose $A \neq B$, using chain rule: $\frac{\partial f(A \cdot B)}{\partial A} = \nabla_{AB} f \cdot B^T$

```
def _derivative(self, feed, A, target):  
    b = self.back(target, feed)  
    return b @ B.T
```

Practical guide

- ▶ cache the computed value to speed up: `primg(feed)`
- ▶ recommended steps to finish PA2
 1. `python test.py TestAutoDifferential`
 2. `python test.py TestMLP`
 3. `python test.py TestLogisticModel`
 4. `python test.py TestRidgeModel (bonus)`
- ▶ enjoy learning from data neural network (`lfdnn`)