人工智能之深度学习

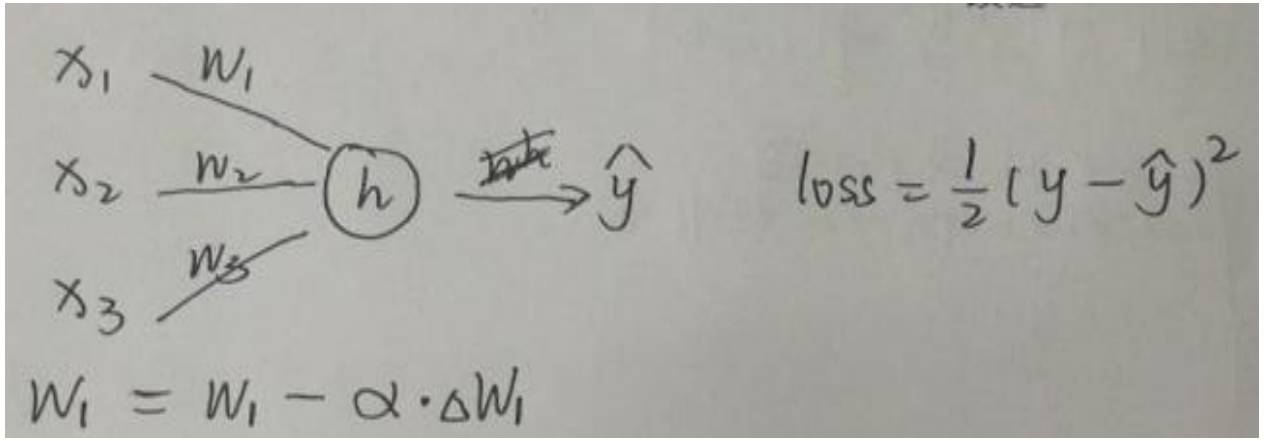
第一章 深度学习入门-BP-讲义

主讲人：Vincent Ying

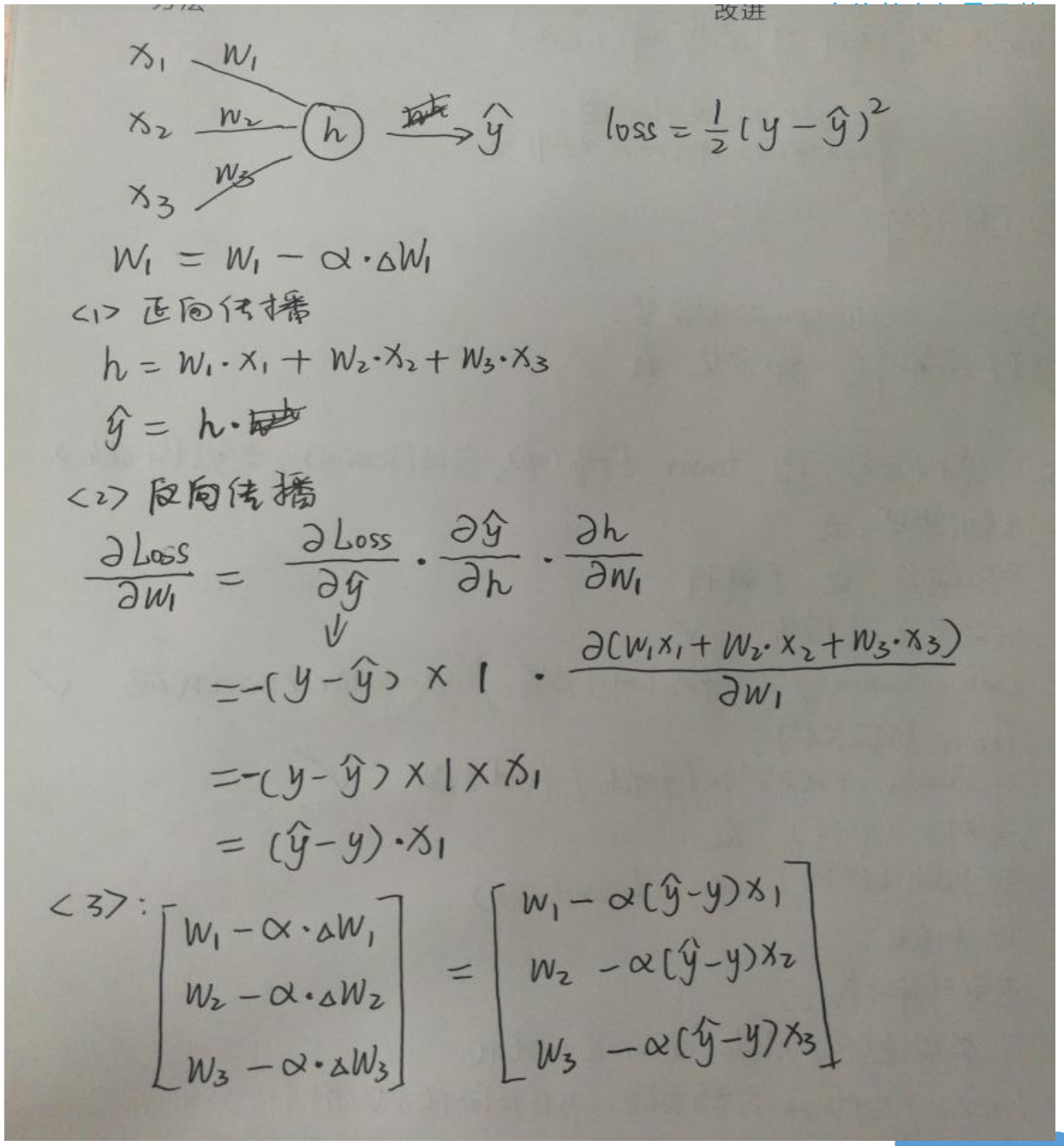
上海育创网络科技有限公司

北风网 IBEIFENG.COM

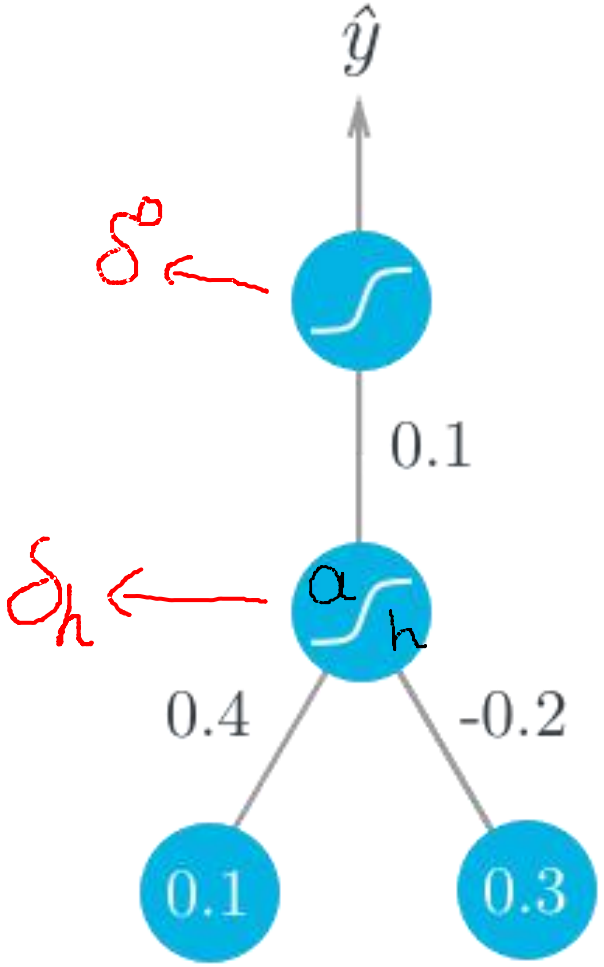
课程目录



手动画图演示这个梯度下降的数学推导



反向传播练习



激活函数均使用sigmoid； sigmoid的导数是?

sigmoid(−0.02)=0.495.

sigmoid(0.0495)=0.512

Y值为：1

学习率η为：0.5

问题：求输入层--隐藏层 和 隐藏层--输出层 权重更新步长!

1、sigmoid 函数的导数 f‘(W * a) =f(W * a)(1−f(W * a))

2、输出节点的误差项（error term）可表示为

δ0 = (y - Y^) f'(W * a) =

3、隐藏节点的误差项

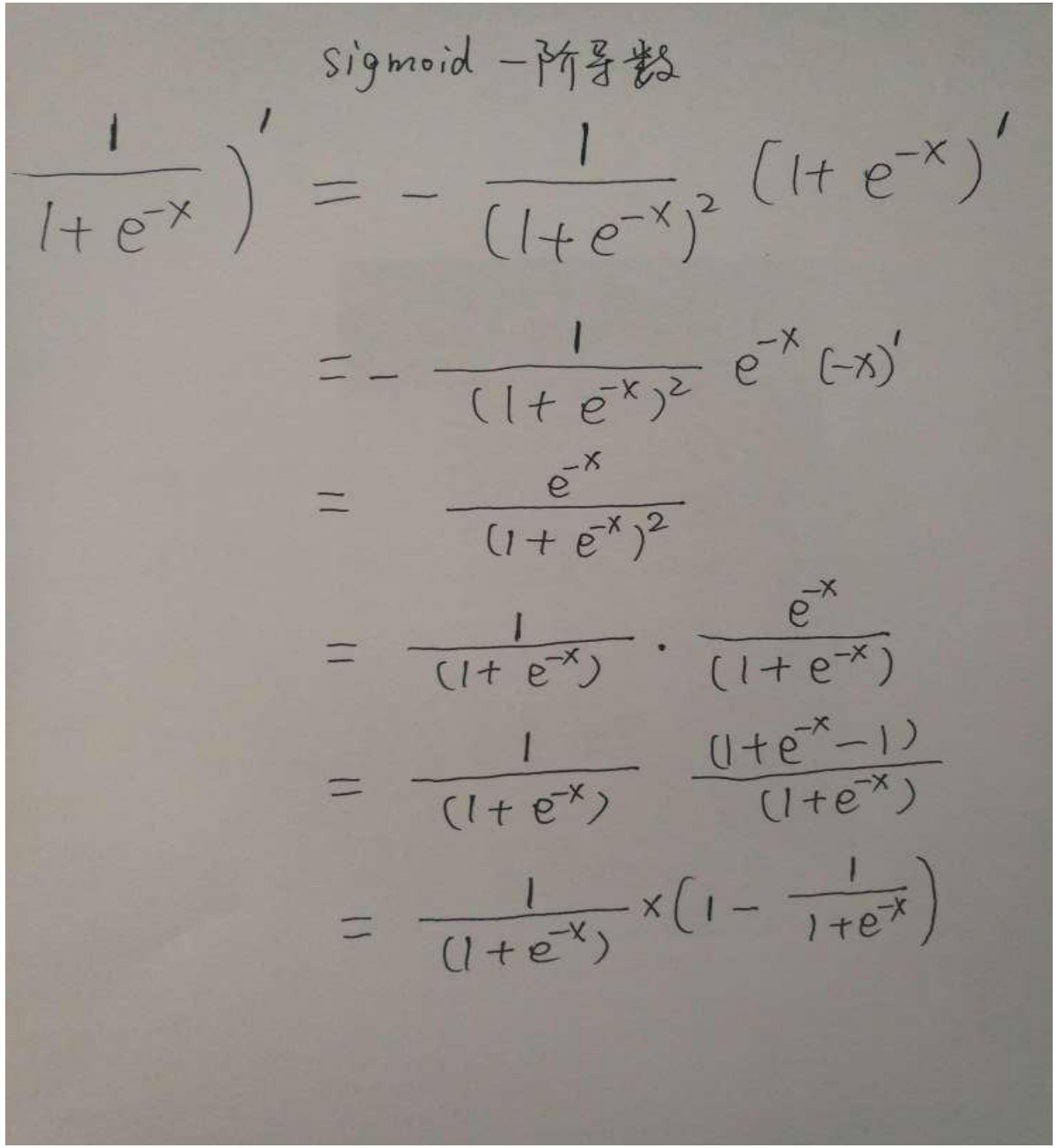
δh=W * δ0 * f ’(h)=

4、隐藏层-输出层权重更新步长=学习率*输出节点误差*隐藏节点激活值

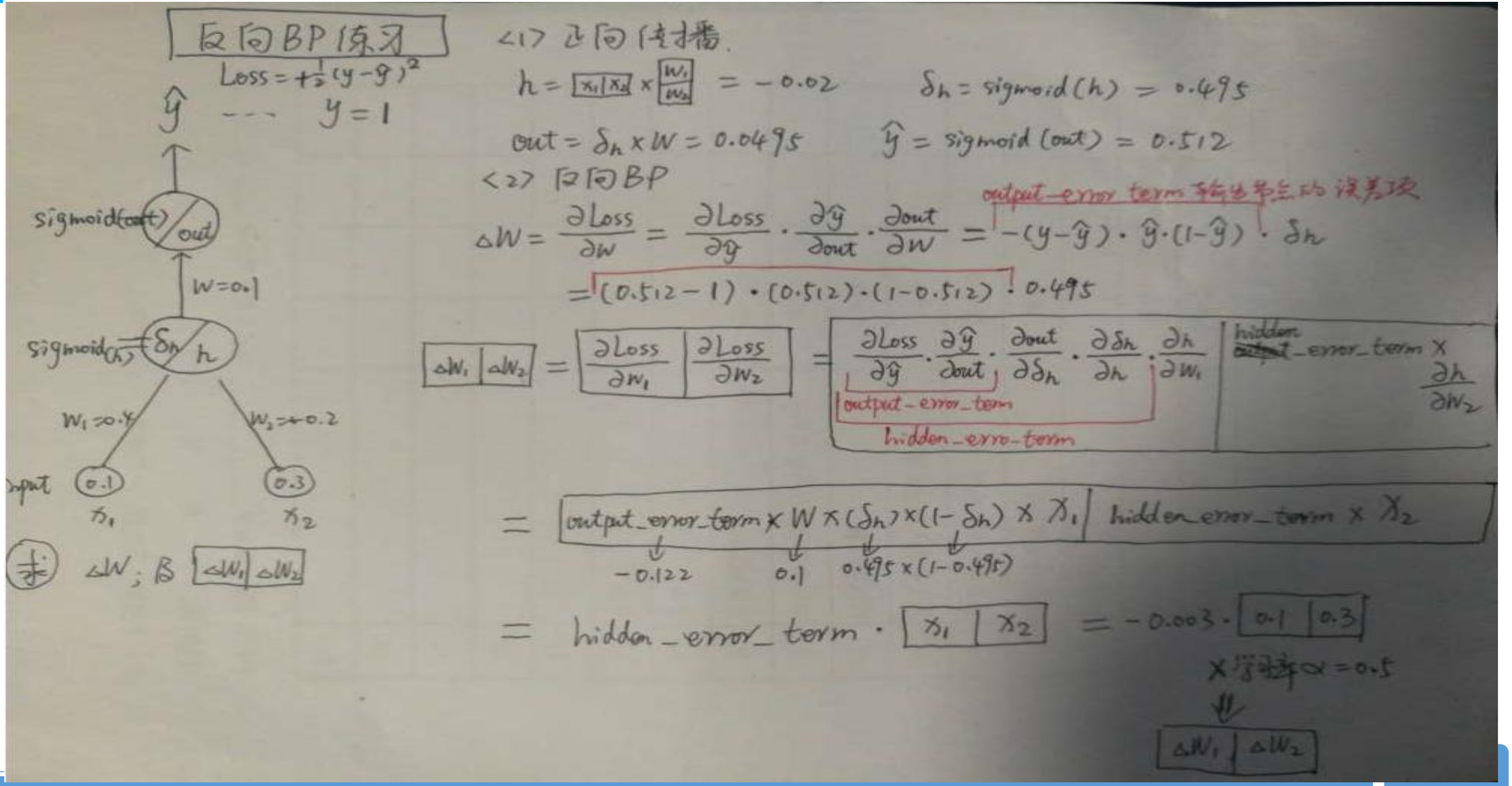
ΔW=η*δ0*a=

5、输入-隐藏层权重 w_i= 学习率*隐藏节点误差*输入值

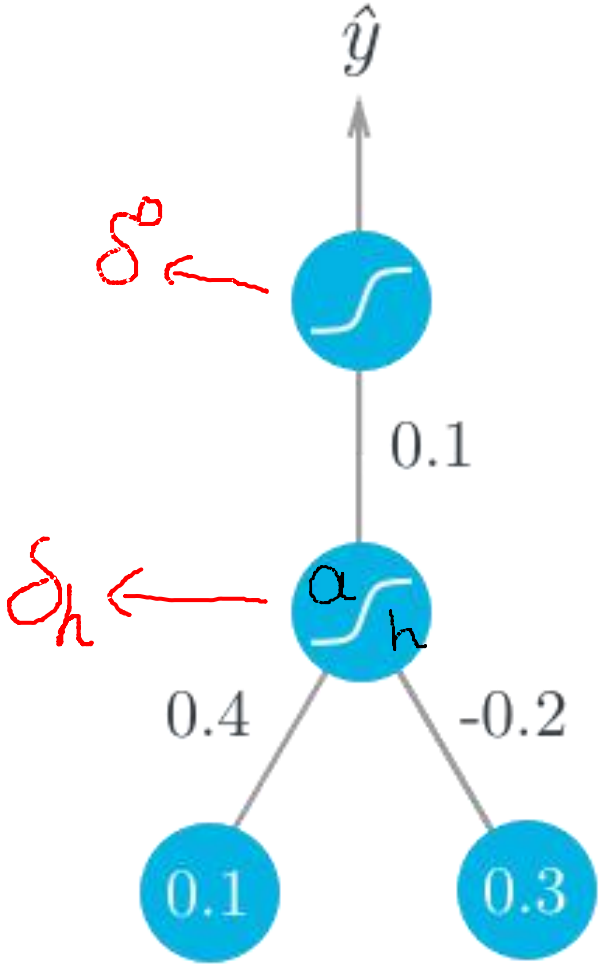
ΔWi=η*δh*Xi=


Sigmoid 一阶导数
$\left(\frac{1}{1+e^{-x}}\right)' = -\frac{1}{(1+e^{-x})^2}(1+e^{-x})'$
$= -\frac{1}{(1+e^{-x})^2} e^{-x}(-x)'$
$= \frac{e^{-x}}{(1+e^{-x})^2}$
$= \frac{1}{(1+e^{-x})} \cdot \frac{e^{-x}}{(1+e^{-x})}$
$= \frac{1}{(1+e^{-x})} \frac{(1+e^{-x}-1)}{(1+e^{-x})}$
$= \frac{1}{(1+e^{-x})} \times \left(1 - \frac{1}{1+e^{-x}}\right)$

反向传播练习答案


反向BP练习
Loss = $\frac{1}{2}(y-\hat{y})^2$
$\hat{y}$ ---- $y=1$
sigmoid(out) / out
W=0.1
sigmoid(h) / δ_h / h
$W_1=0.4$
$W_2=0.2$
0.1
0.3
x_1
x_2
<1> 正向传播
$h = [x_1 | x_2] \times \begin{bmatrix} W_1 \\ W_2 \end{bmatrix} = -0.02$
$\delta_h = sigmoid(h) = 0.495$
$out = \delta_h \times W = 0.0495$
$\hat{y} = sigmoid(out) = 0.512$
<2> 反向BP
output_error_term 传递到上面的误差项
$\Delta W = \frac{\partial Loss}{\partial W} = \frac{\partial Loss}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial out} \cdot \frac{\partial out}{\partial W} = -(y-\hat{y}) \cdot \hat{y} \cdot (1-\hat{y}) \cdot \delta_h$
$= (0.512-1) \cdot (0.512) \cdot (1-0.512) \cdot 0.495$
$[\Delta W_1 | \Delta W_2] = \left[\frac{\partial Loss}{\partial W_1} \middle| \frac{\partial Loss}{\partial W_2}\right] = \frac{\partial Loss}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial out} \cdot \frac{\partial out}{\partial \delta_h} \cdot \frac{\partial \delta_h}{\partial h} \cdot \frac{\partial h}{\partial W_1}$ | hidden_error_term $\times \frac{\partial h}{\partial W_2}$
output_error_term
hidden_error_term
$= [\text{output\_error\_term} \times W \times (\delta_h) \times (1-\delta_h) \times x_1 \mid \text{hidden\_error\_term} \times x_2]$
-0.122 0.1 $0.495 \times (1-0.495)$
$= \text{hidden\_error\_term} \cdot [x_1 | x_2] = -0.003 \cdot [0.1 | 0.3]$
学习率 $\alpha = 0.5$
$[\Delta W_1 | \Delta W_2]$

反向传播-答案



Step 1　正向传播

$h=\sum Wi*Xi=0.1\times0.4-0.2\times0.3=-0.02$

隐藏层节点的输出$a = f(h) = sigmoid(-0.02)=0.495.$

该神经网络的output为 $Y^{\wedge} =f(W\cdot a)=sigmoid(0.1\times0.495)=0.512$

Step 2　反向传播

1、sigmoid 函数的导数 $f'(W * a) =f(W * a)(1-f(W * a))$

2、输出节点的误差项（error term）可表示为

$\delta0 = (y - Y^{\wedge}) f'(W * a) = (1-0.512)\times0.512\times(1-0.512)=0.122$

3、隐藏节点的误差项

$\delta h=W * \delta0 * f'(h)=0.1\times0.122\times0.495\times(1-0.495)=0.003$

4、隐藏层-输出层权重更新步长=学习率*输出节点误差*隐藏节点激活值

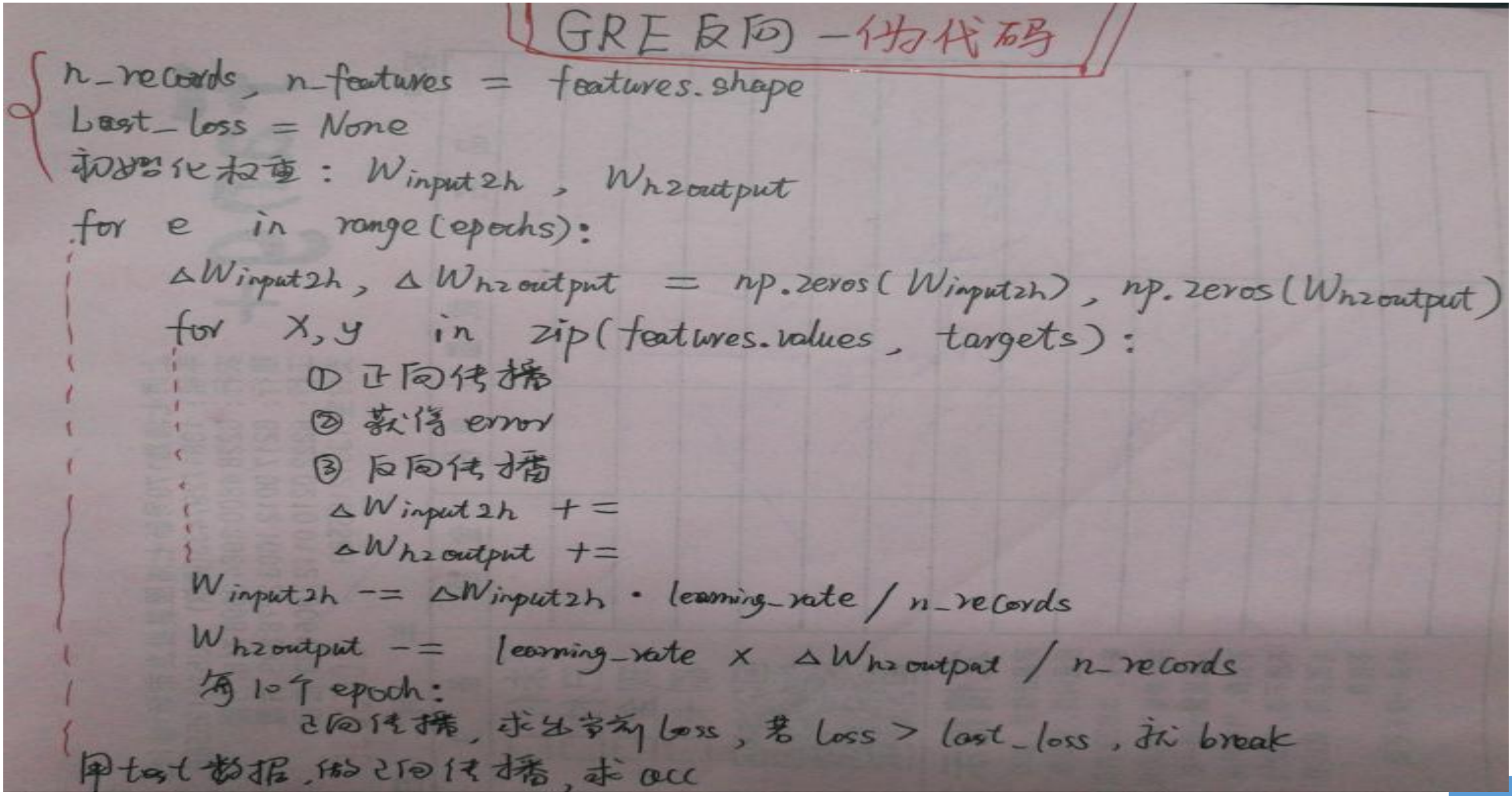
$\Delta W=\eta*\delta0*a=0.5\times0.122\times0.495=0.0302$

5、输入-隐藏层权重 w_i= 学习率*隐藏节点误差*输入值

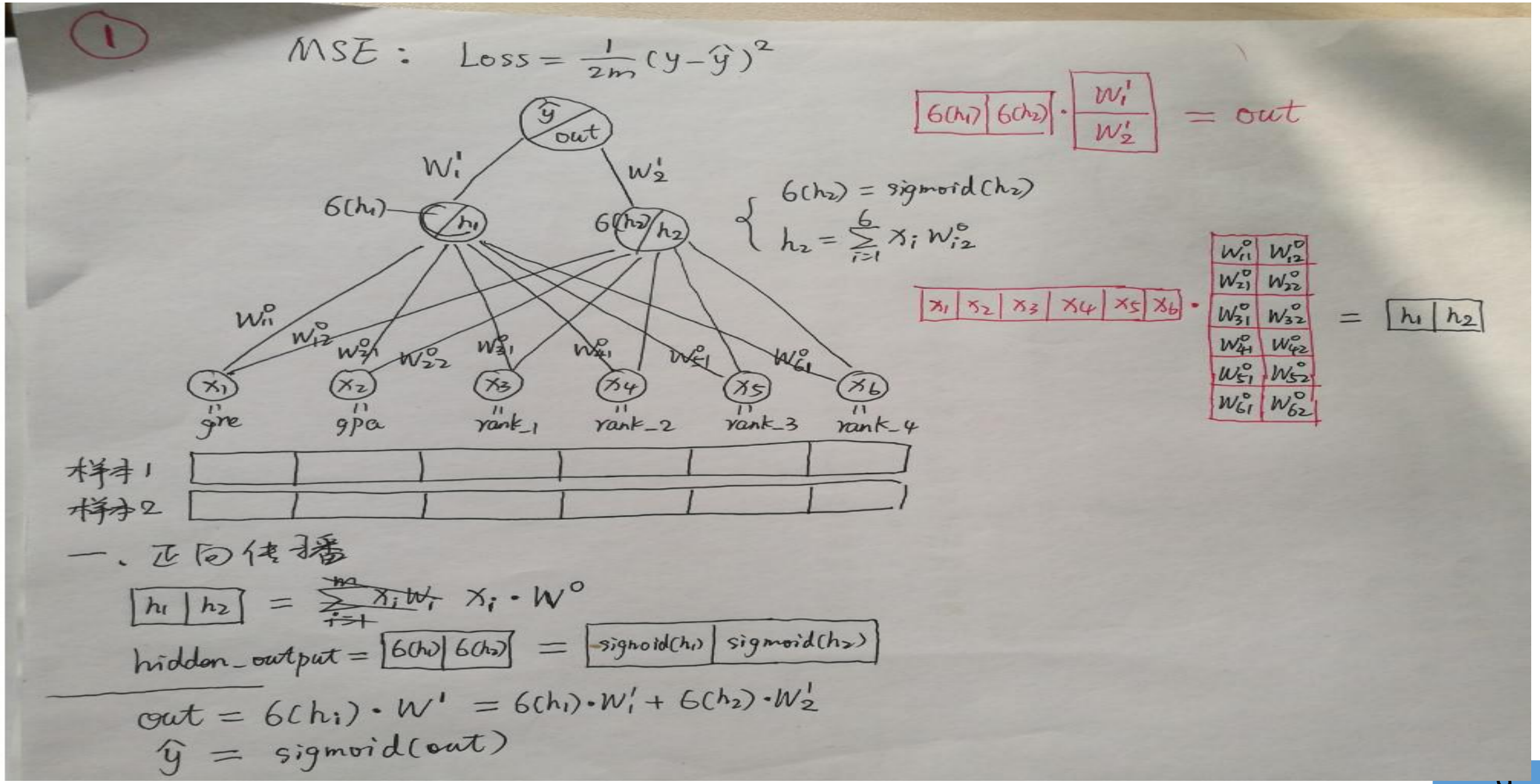
$\Delta Wi=\eta*\delta h*Xi=(0.5\times0.003\times0.1,0.5\times0.003\times0.3)=(0.00015,0.00045)$

02Code反向传播练习

GRE反向-伪代码


GRE反向-伪代码
n_records, n_features = features.shape
last_loss = None
初始化权重：W_input2h , W_h2output
for e in range(epochs):
ΔW_input2h, ΔW_h2output = np.zeros(W_input2h), np.zeros(W_h2output)
for X, y in zip(features.values, targets):
① 正向传播
② 获得 error
③ 反向传播
ΔW_input2h +=
ΔW_h2output +=
W_input2h -= ΔW_input2h * learning_rate / n_records
W_h2output -= learning_rate * ΔW_h2output / n_records
每10个epoch:
正向传播，求出当前loss，若 loss > last_loss，就 break
用test数据，做正向传播，求acc

GRE反向传播-推导1


①
MSE: $Loss = \frac{1}{2m}(y-\hat{y})^2$
$\hat{y}$ / out
w_1'
w_2'
$g(h_1)$
h_1
$g(h_2)$ h_2
$g(h_2) = sigmoid(h_2)$
$h_2 = \sum_{i=1}^{6} x_i w_{i2}^0$
$g(h_1)\ g(h_2) \cdot \begin{matrix} w_1' \\ w_2' \end{matrix} = out$
$x_1\ x_2\ x_3\ x_4\ x_5\ x_6 \cdot \begin{matrix} w_{11}^0 & w_{12}^0 \\ w_{21}^0 & w_{22}^0 \\ w_{31}^0 & w_{32}^0 \\ w_{41}^0 & w_{42}^0 \\ w_{51}^0 & w_{52}^0 \\ w_{61}^0 & w_{62}^0 \end{matrix} = h_1\ h_2$
x_1 gre
x_2 gpa
x_3 rank_1
x_4 rank_2
x_5 rank_3
x_6 rank_4
样本1
样本2
一、正向传播
$h_1\ h_2 = x_i \cdot W^0$
hidden_output = $g(h_1)\ g(h_2)$ = $sigmoid(h_1)\ sigmoid(h_2)$
$out = g(h_i) \cdot W' = g(h_1) \cdot w_1' + g(h_2) \cdot w_2'$
$\hat{y} = sigmoid(out)$

GRE反向传播--推导

二、反向传播

① input-to-hidden

$$W_{11}^{\circ} = \frac{\partial Loss}{\partial W_{11}^{\circ}} = \frac{\partial Loss}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial out} \cdot \frac{\partial out}{\partial G(h_1)} \cdot \frac{\partial G(h_1)}{\partial h_1} \cdot \frac{\partial h_1}{\partial W_{11}^{\circ}}$$

$$= -(y-\hat{y}) \cdot sigmoid(out)[1-sigmoid(out)] \cdot W_1' \cdot sigmoid(h_1)[1-sigmoid(h_1)] \cdot x_1$$

error

output_error_term

hidden_error

hidden_error_term

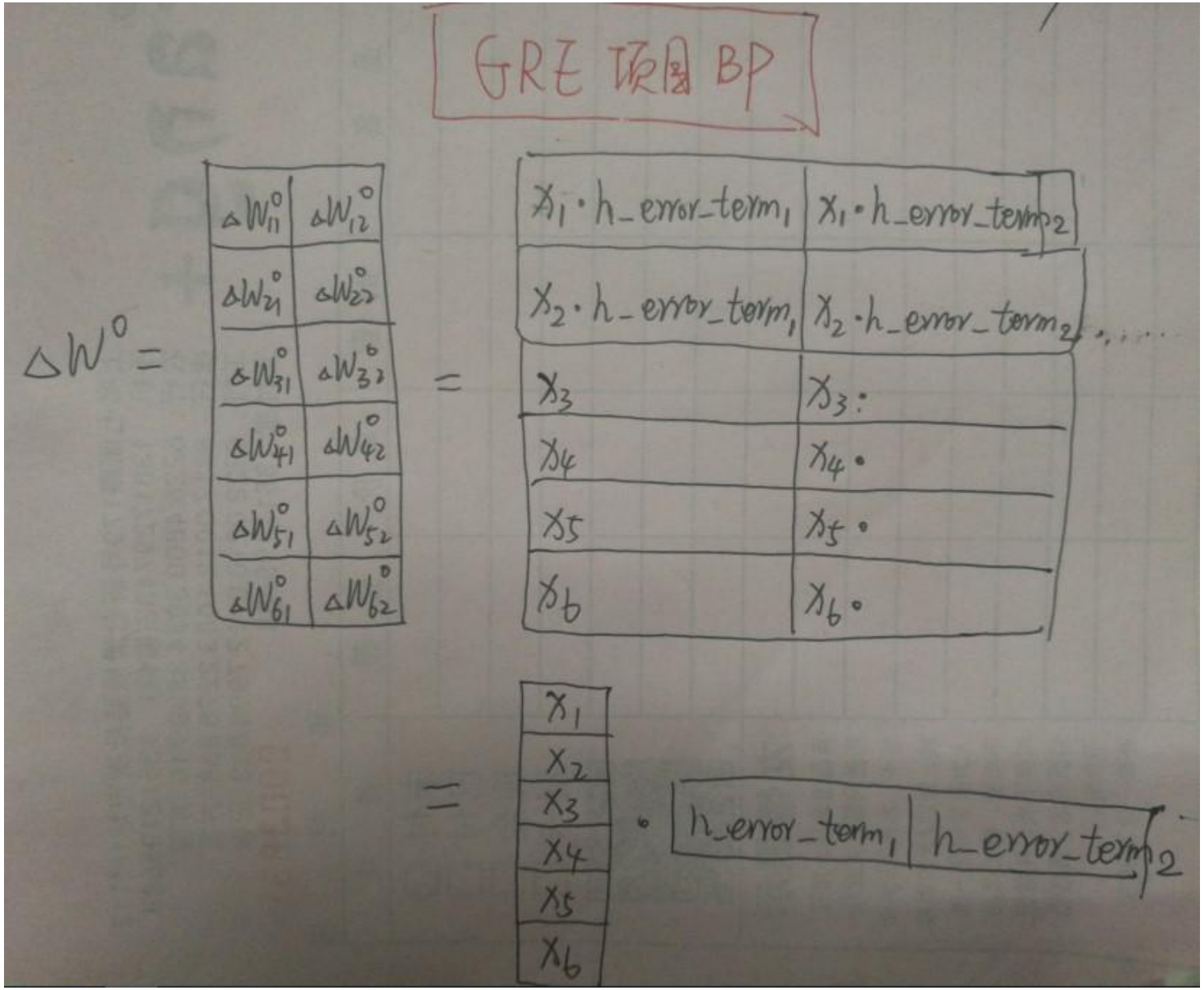
$$hidden\_error = output\_error\_term \cdot [W_1', W_2']$$

$$hidden\_error\_term = hidden\_error \cdot [sigmoid(h_1)[1-sigmoid(h_1)), sigmoid(h_2)(1-sigmoid(h_2))]$$

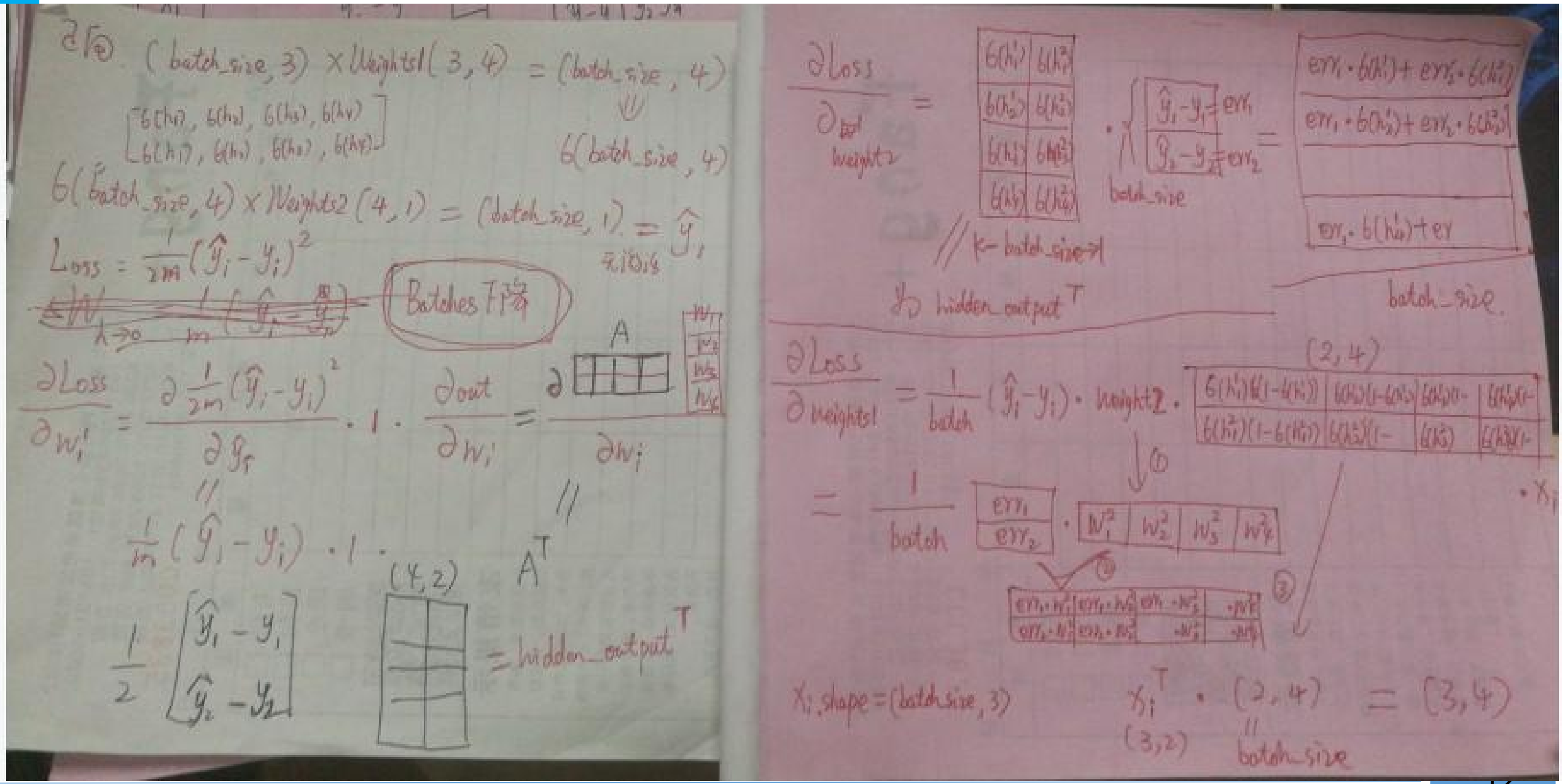
shape

$$\Delta W^{\circ} = \begin{bmatrix} \Delta W_{11}^{\circ} & \Delta W_{12}^{\circ} \\ \Delta W_{21}^{\circ} & \Delta W_{22}^{\circ} \\ \Delta W_{31}^{\circ} & \Delta W_{32}^{\circ} \\ \Delta W_{41}^{\circ} & \Delta W_{42}^{\circ} \\ \Delta W_{51}^{\circ} & \Delta W_{52}^{\circ} \\ \Delta W_{61}^{\circ} & \Delta W_{62}^{\circ} \end{bmatrix} = \begin{bmatrix} \quad \end{bmatrix} \cdot hidden\_error\_term \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix}$$

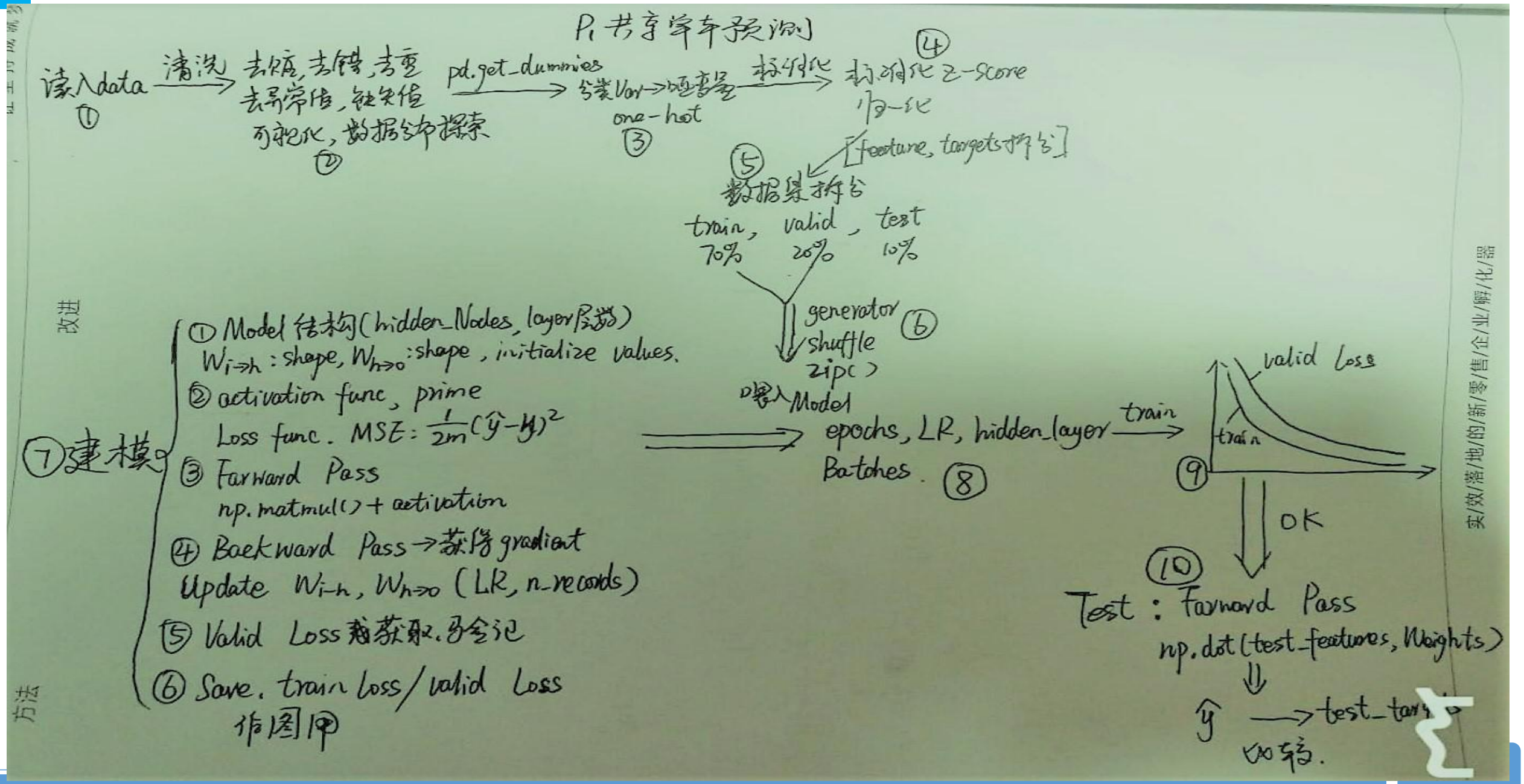
GRE反向传播--推导2


GRE 项目 BP
$\Delta W^0 =$
ΔW_{11}^0 ΔW_{12}^0
ΔW_{21}^0 ΔW_{22}^0
ΔW_{31}^0 ΔW_{32}^0
ΔW_{41}^0 ΔW_{42}^0
ΔW_{51}^0 ΔW_{52}^0
ΔW_{61}^0 ΔW_{62}^0
=
$x_1 \cdot h\_error\_term_1$ $x_1 \cdot h\_error\_term_2$
$x_2 \cdot h\_error\_term_1$ $x_2 \cdot h\_error\_term_2$
x_3 $x_3 \cdot$
x_4 $x_4 \cdot$
x_5 $x_5 \cdot$
x_6 $x_6 \cdot$
=
x_1
x_2
x_3
x_4
x_5
x_6
$\cdot$ $h\_error\_term_1$ $h\_error\_term_2$

GRE MBGD反向传播--推导



共享单车项目-项目流程


P1 共享单车预测
读入data ①
清洗
去除,去错,去重
去异常值,缺失值
可视化,数据分布探索 ②
pd.get_dummies
分类Var→哑变量
one-hot ③
标准化
④
标准化 z-score
归一化
[feature, targets拆分]
⑤
数据集拆分
train, valid, test
70% 20% 10%
generator
shuffle
zip()
⑥
喂入Model
⑦建模
① Model 结构(hidden_Nodes, layer层数)
W_i→h : shape, W_h→o : shape, initialize values.
② activation func, prime
Loss func. MSE: 1/2m (ŷ-y)²
③ Farward Pass
np.matmul() + activation
④ Baekward Pass → 梯度 gradient
Update W_i-h, W_h→o (LR, n_records)
⑤ Valid Loss 获取,多会记
⑥ Save. train loss/valid loss
作图用
改进
方法
epochs, LR, hidden_layer
Batches. ⑧
train
⑨
valid Loss
train
OK
⑩
Test : Farward Pass
np.dot(test_features, Weights)
ŷ → test_tar
反转.
实/效/落/地/的/新/零/售/企/业/孵/化/器

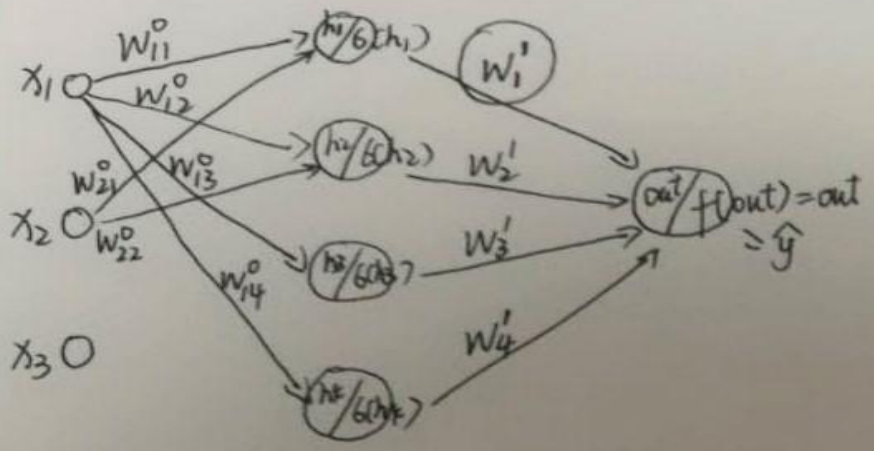
单车反向传播-伪代码

共享单车伪代码

```
n_record, n_feature = x_train.shape()
初始化W
W_input2h , W_h2output
for e in range(epochs):
    ΔW_input2h , ΔW_h2output = np.zeros(W_input2h) , np.zeros(W_h2output)
    for X, y in zip(x_train, targets)
        ① 正向传播
        ② 获得error
        ③ 反向传播
        ΔW_input2h += , ΔW_h2output +=
    W_input2h -= learning_rate x ΔW_input2h / n_record
    W_h2output -= learning_rate x ΔW_h2output / n_record
    if e // 10 ==0:
        计算指标，并打印出来。
使用test_data，进行正向传播，获得pred，用测试指标判断效果！
```

单车反向传播-推导1

反向传播



input hidden output target

正向Pass

<1>

$$[x_1\ x_2\ x_3]\begin{bmatrix} w_{11}^0 & w_{12}^0 & w_{13}^0 & w_{14}^0 \\ w_{21}^0 & w_{22}^0 & w_{23}^0 & w_{24}^0 \\ w_{31}^0 & w_{32}^0 & w_{33}^0 & w_{34}^0 \end{bmatrix} = [h_1, h_2, h_3, h_4]$$

$\Downarrow$ sigmoid

$$[\sigma(h_1), \sigma(h_2), \sigma(h_3), \sigma(h_4)]$$

hidden

<2>

$$[\sigma(h_1), \sigma(h_2), \sigma(h_3), \sigma(h_4)] \cdot \begin{bmatrix} w_1' \\ w_2' \\ w_3' \\ w_4' \end{bmatrix} = out \rightarrow f(out) = \hat{y}$$

output layer

求 $-Delta\_W_{h\to o}$ 梯度. Loss func $MSE = \frac{1}{2m}(\hat{y}-y)^2$

$$\frac{\partial MSE}{\partial w_1'} = \frac{\partial \frac{1}{2m}(\hat{y}-y)^2}{\partial \hat{y}} \cdot \frac{\partial f(out)}{out} \cdot \frac{\partial out}{\partial w_1'}$$

$f(x)=x'$

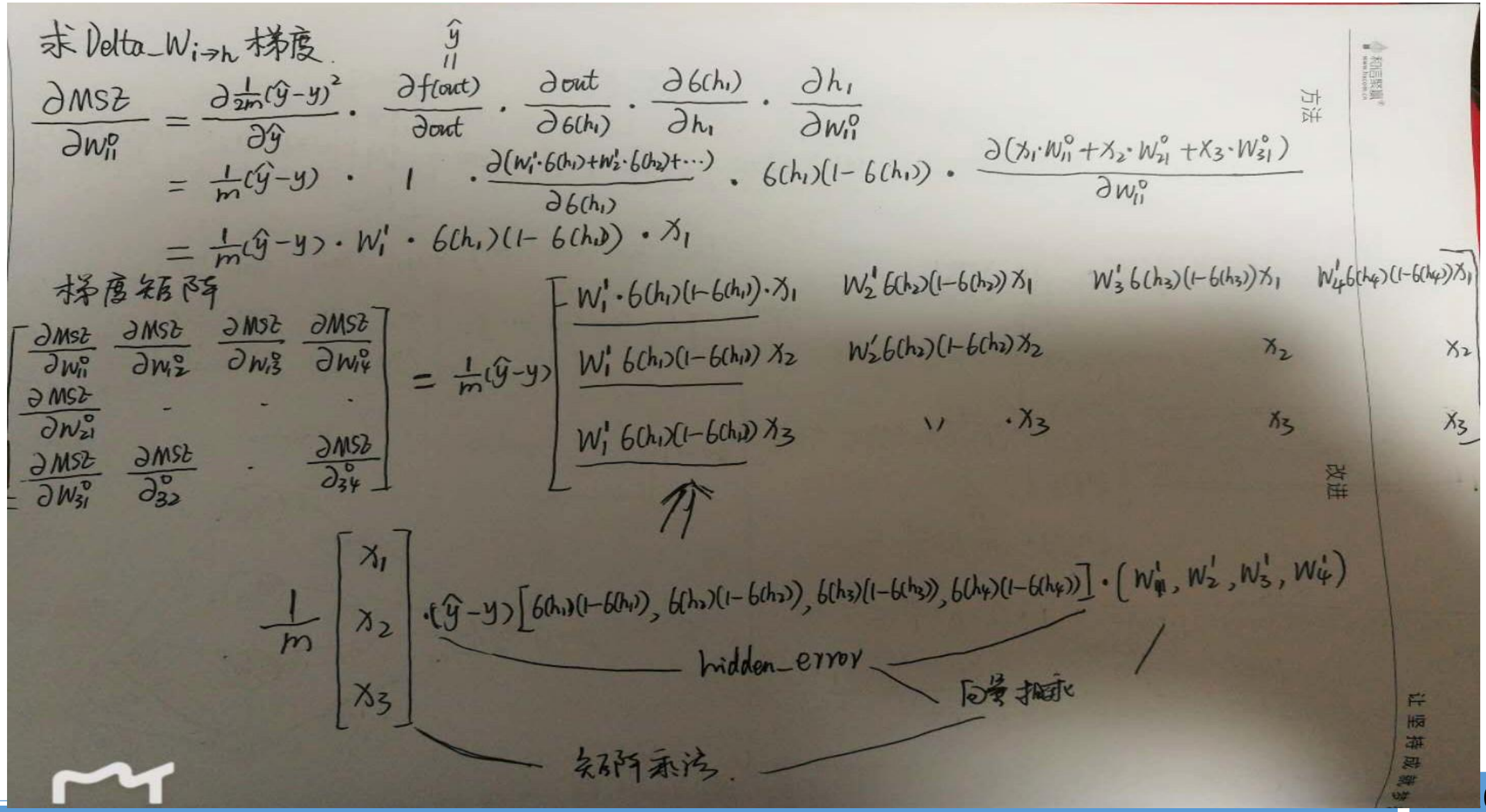
$$= \frac{1}{m}(\hat{y}-y) \cdot 1 \cdot \frac{\partial(w_1'\cdot\sigma(h_1)+w_2'\cdot\sigma(h_2)+w_3'\cdot\sigma(h_3)+w_4'\cdot\sigma(h_4))}{\partial w_1'}$$

$$= \frac{1}{m}(\hat{y}-y)\cdot\sigma(h_1)$$

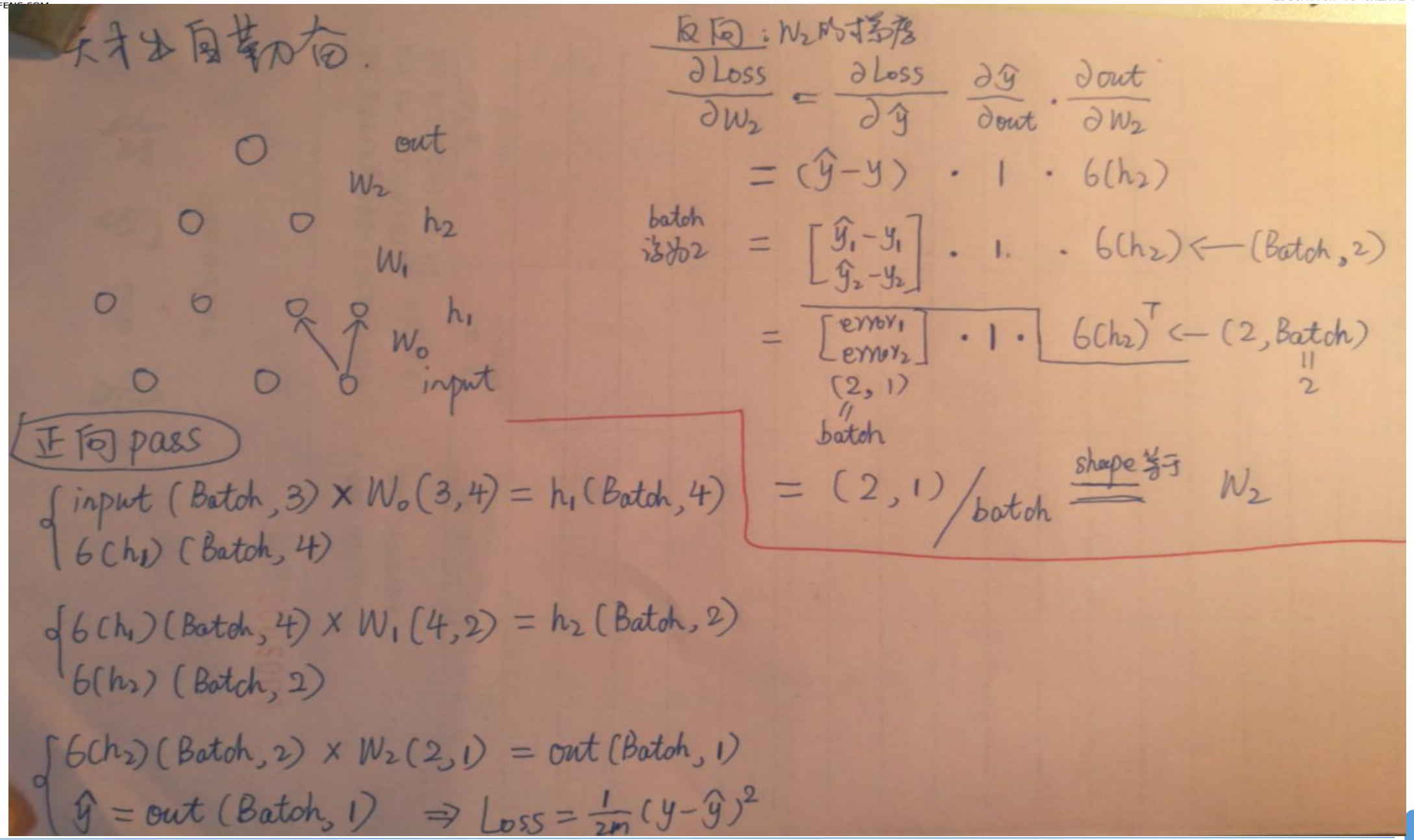
$\Downarrow\Downarrow$

$$Delta\_W_{h\to o} = \frac{(\hat{y}-y)}{m} \cdot \begin{bmatrix} \sigma(h_1) \\ \sigma(h_2) \\ \sigma(h_3) \\ \sigma(h_4) \end{bmatrix} = \boxed{\frac{1}{m}(\hat{y}-y)\cdot hidden\_outputs}$$

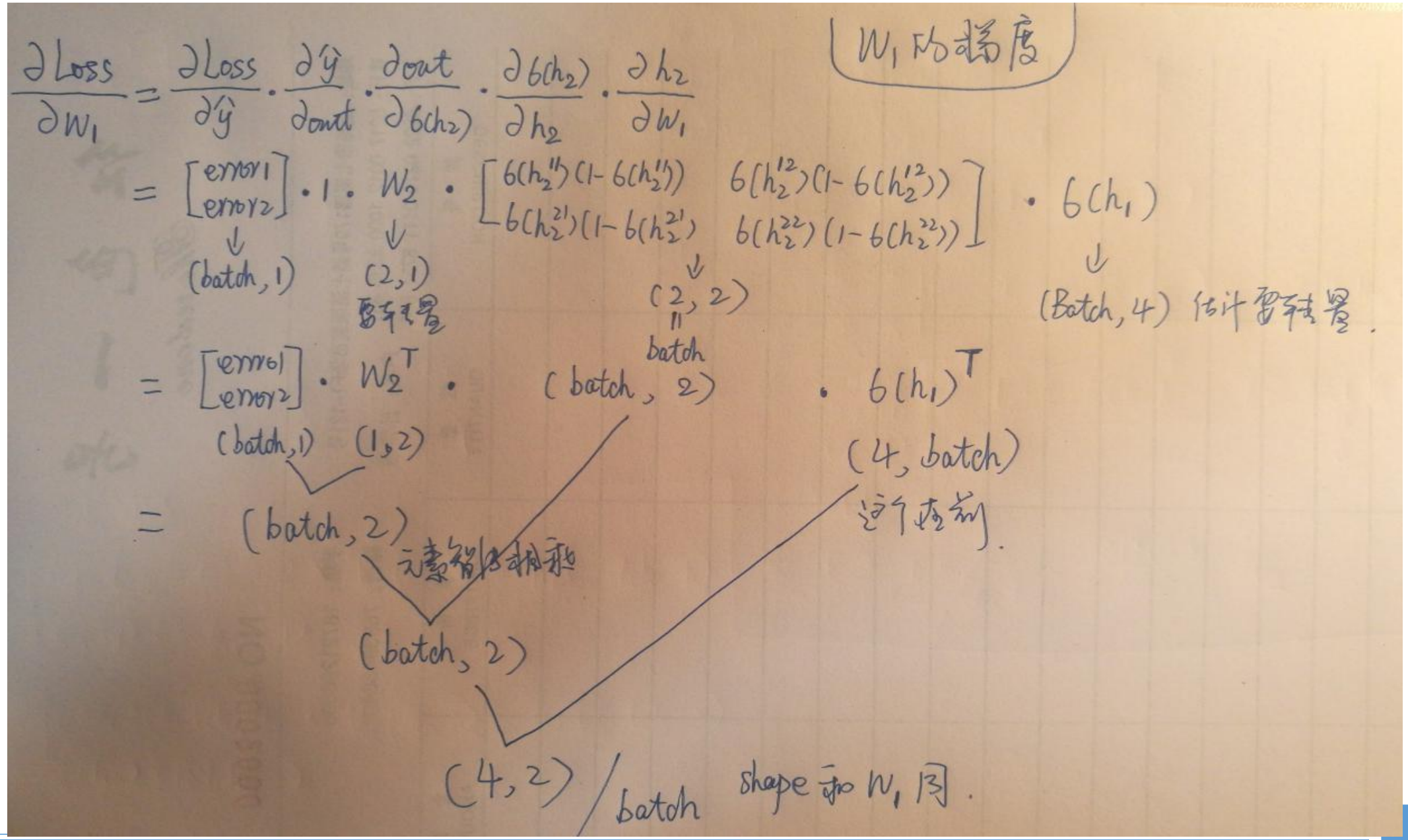
单车反向传播-推导2



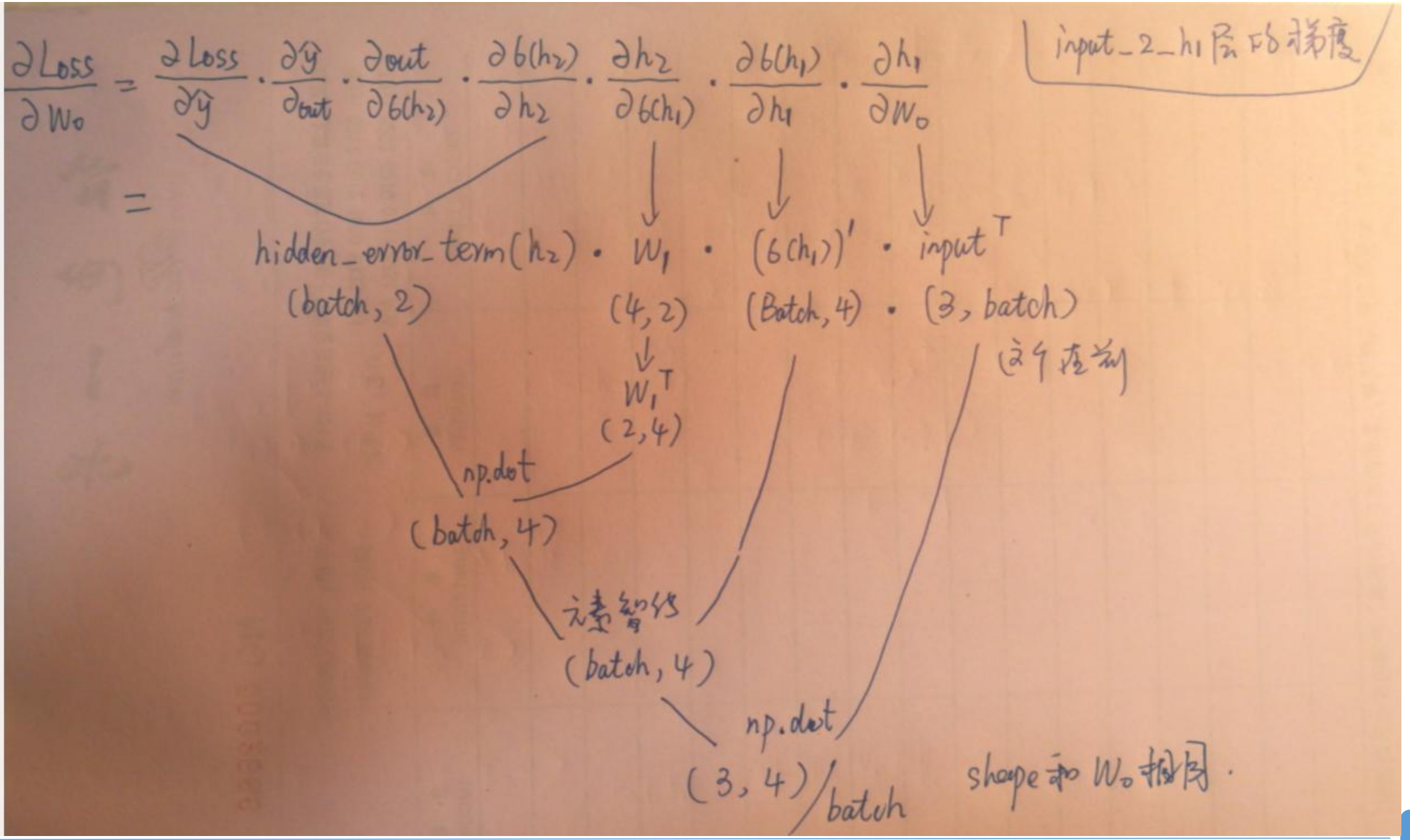
单车项目 多隐藏层 BP


out
W_2
h_2
W_1
h_1
W_0
input
正向pass
input (Batch, 3) × W_0(3,4) = h_1(Batch, 4)
$\sigma(h_1)$ (Batch, 4)
$\sigma(h_1)$ (Batch, 4) × W_1(4,2) = h_2(Batch, 2)
$\sigma(h_2)$ (Batch, 2)
$\sigma(h_2)$ (Batch, 2) × W_2(2,1) = out (Batch, 1)
$\hat{y}$ = out (Batch, 1) ⇒ Loss = $\frac{1}{2m}(y-\hat{y})^2$
反向：W_2的求导
$\frac{\partial Loss}{\partial W_2} = \frac{\partial Loss}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial out} \cdot \frac{\partial out}{\partial W_2}$
$= (\hat{y}-y) \cdot 1 \cdot \sigma(h_2)$
batch 为2
$= \begin{bmatrix} \hat{y}_1-y_1 \\ \hat{y}_2-y_2 \end{bmatrix} \cdot 1 \cdot \sigma(h_2) \leftarrow (Batch, 2)$
$= \begin{bmatrix} error_1 \\ error_2 \end{bmatrix} \cdot 1 \cdot \sigma(h_2)^T \leftarrow (2, Batch)$
(2, 1) = 2
// batch
$= (2,1)/batch$ shape等于 W_2

单车项目 多隐藏层 BP


W1的梯度
(batch, 1)
(2, 1)
需转置
(2, 2)
batch
(Batch, 4) 估计要转置.
(batch, 2)
元素相乘
(batch, 2)
(4, batch)
运行梯前.
(4, 2)
batch
shape和W1同.

单车项目 多隐藏层 BP


input_2_h1层的梯度
hidden_error_term(h2) · W1 · (6(h1))' · input^T
(batch, 2)
(4, 2)
(Batch, 4)
(3, batch)
W1^T
(2, 4)
np.dot
(batch, 4)
元素积
(batch, 4)
np.dot
(3, 4)
batch
shape和W0相同.