Weight Difference Propagation for Stochastic Gradient Descent Learning

Shonan Institute of Technology
Fujisawa, Japan

Shahrzad Mahboubi and Hiroshi Ninomiya

Introduction

- AI has been used in various fields in recent years
  - The core technology in AI is Neural Network(NN)

- In NN, training is a very important factor
- Training in NN：An unconstrained optimization problem to minimize the error function $E(\mathbf{w})$ with regard to the weight vector $\mathbf{w}$

$$\min_{\mathbf{w}\in R^n} E(\mathbf{w}). \quad (1)$$

$\mathbf{w}$：Weight vector, $E(\mathbf{w})$：Error function

- NN training typically uses an algorithm based on error back propagation (BP), which is a gradient method （Gradient Descent (GD)）
- The error function $E(\mathbf{w})$ in BP :

$$E(\mathbf{w}) = \frac{1}{|T_r|}\sum_{p\in T_r} E_p(\mathbf{w}). \quad (2)$$

$|T_r|$：#. Data samples, p：# . Sample's pattern

Introduction

- 2 types of training exist in BP: **Batch** and **Mini-batch** training

① Batch training：

- Advantages：Training with high accuracy

Improved method

- Disadvantages：Training of large-data is difficult

② **Stochastic training**（Mini-batch training）：

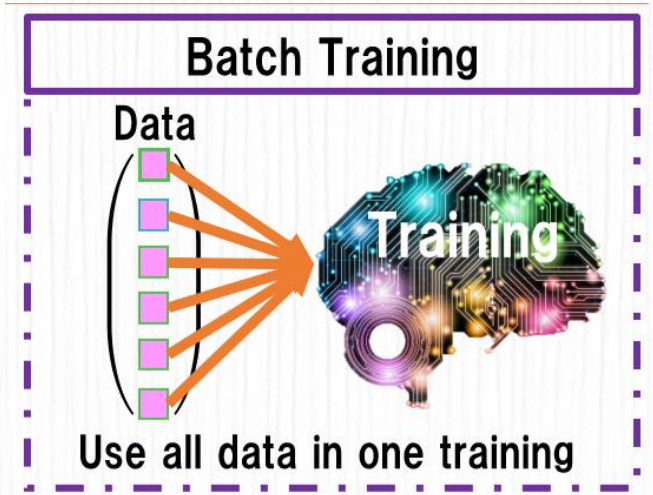
- Advantages：Large-data can be trained

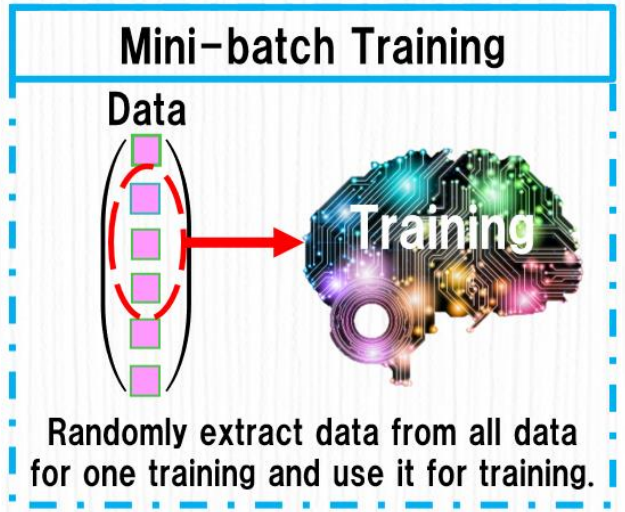
- Stochastic training with BP：

  Stochastic Gradient Descent（**SGD**）

- The error function $E_b(\mathbf{w})$ in SGD:

$$E_b(\mathbf{w}) = \frac{1}{b}\sum_{p\in X} E_p(\mathbf{w}). \qquad (3)$$





$b = |X|$：Mini-batch size, (in batch training, $X = T_r, b = |T_r|$),
$E_b(\mathbf{w})$：The error for each mini-batch size, $E_p(\mathbf{w})$：The error for all data sample

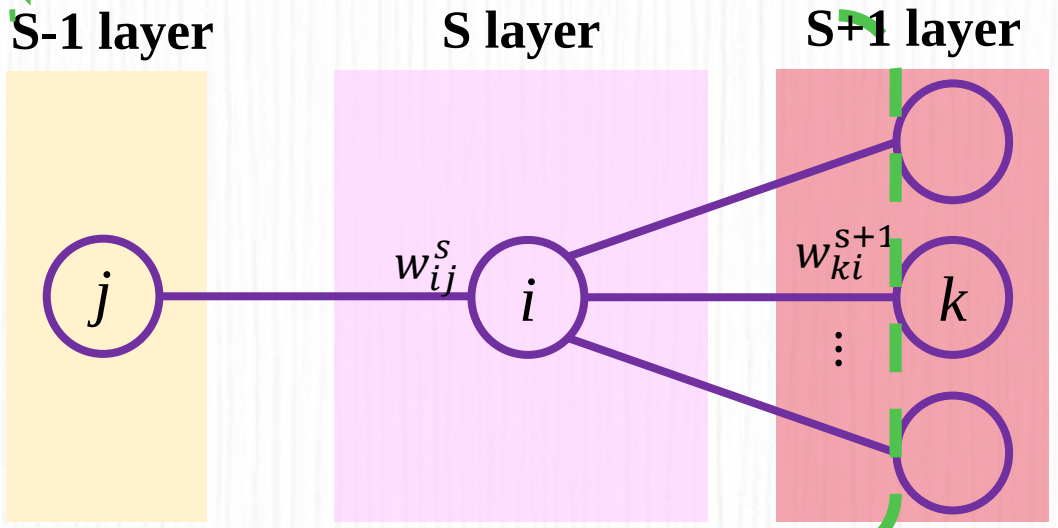
Introduction

➢ In SGD Training：

- For updating the weight, The inner product of **weight** and **backward element of output for all p include in b** are require

b：mini－batch size,　p：#. Sample pattern

For all p include in b



The inner product of the weights and the updated (difference) weights

•Disadvantages：Hardware and computation costs increase when implemented on micro-PC

Improved method

Focus point：

Back ward component for b ≅ **Update amount of weight (difference)**

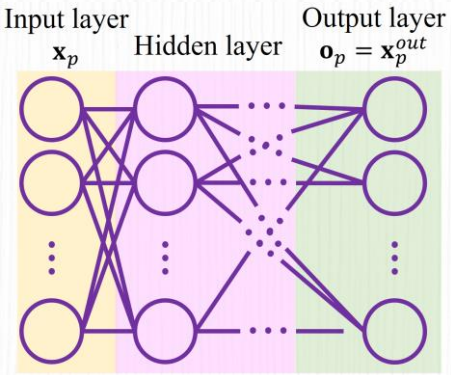
Propose method：Stochastic Weight Difference Propagation (SWDP)

It can be calculated by the inner product of the **weights** and **the difference between the weights**

Stochastic Gradient Descent (SGD)

- NN input/output relationship:

$$\mathbf{o}_p = \mathbf{x}_p^{\text{out}} = f_{NN}(\mathbf{w}, \mathbf{x}_p) \quad (4)$$



$\mathbf{o}_p = \mathbf{x}_p^{out}$: Output, $\mathbf{x}_p$: p-th input

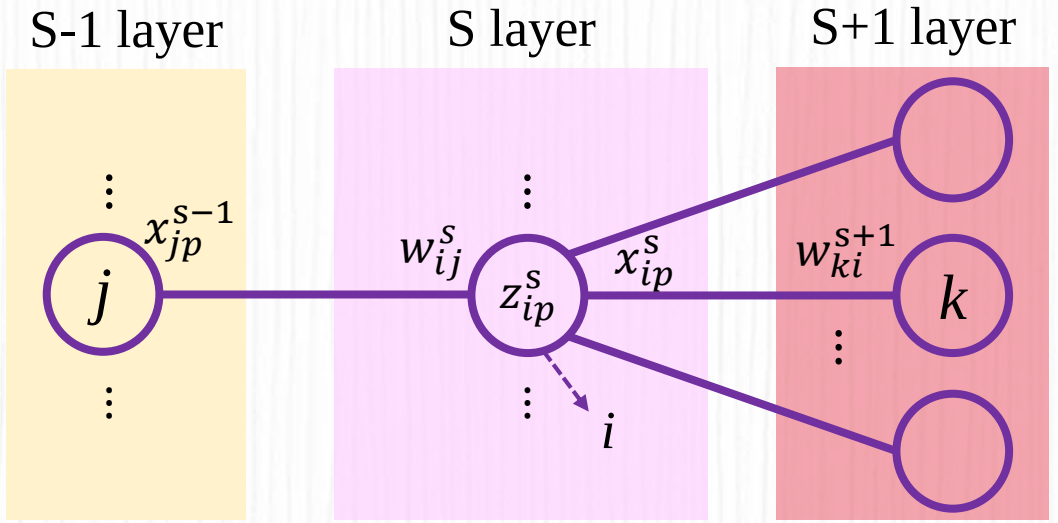
b : mini − batch size, p : #. Sample pattern

- The input-output relationship of the i-th neuron in layer S:

$$z_{i,p}^{s} = \sum_j w_{ij}^{s} \cdot x_{j,p}^{s-1}, \quad (5)$$

$$x_{i,p}^{s} = f\left(z_{i,p}^{s}\right). \quad (6)$$

Feed Forward：For all p include in b



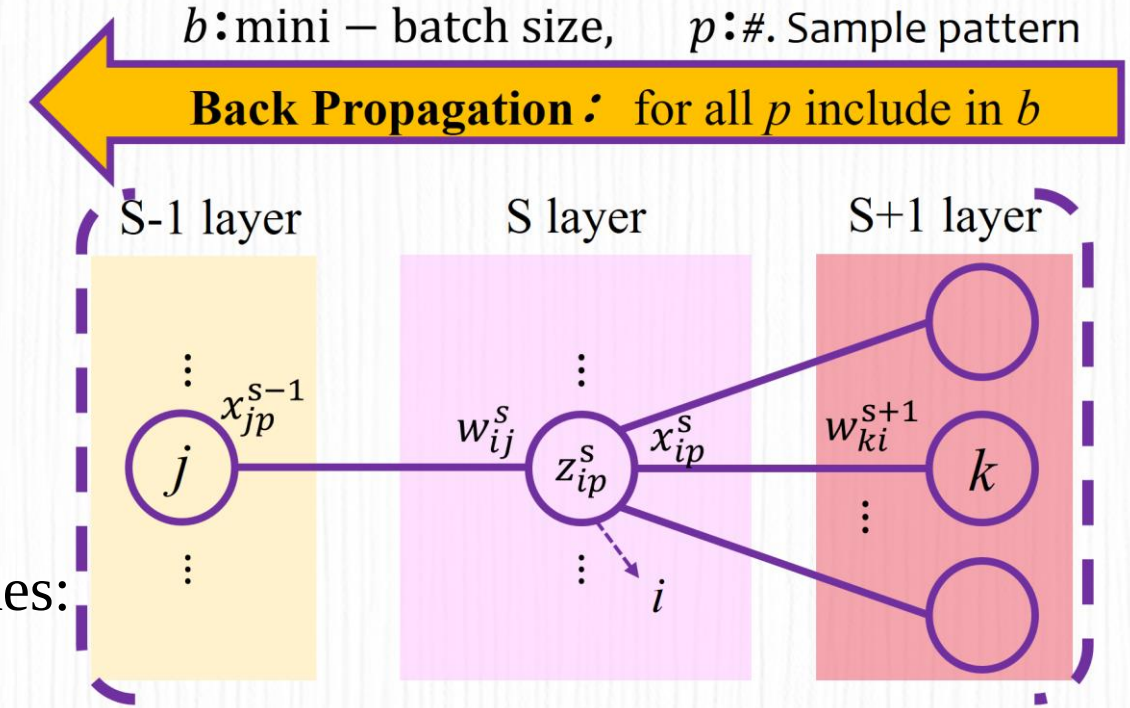
$f\left(z_{i,p}^{s}\right)$: Activation function

Stochastic Gradient Descent (SGD)

➢Update formula for weight w_{ij}^s in SGD:

$$w_{ij}^s(t+1) = w_{ij}^s(t) - \eta \underbrace{\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^s}}_{\text{gradient}}, \quad (7)$$

t：#. Iteration counts, η：Learning rate



➢ By using the chain rule, the gradient $\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^s}$ becomes:

$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^s} = \frac{1}{b}\sum_{p\in X}\frac{\partial E_p(\mathbf{w})}{\partial w_{ij}^s} = \frac{1}{b}\sum_{p\in X}\frac{\partial E_p(\mathbf{w})}{\partial x_{i,p}^s}\cdot\frac{\partial x_{i,p}^s}{\partial z_{i,p}^s}\cdot\frac{\partial z_{i,p}^s}{\partial w_{ij}^s}$$

If S layer is output layer? Or not?

$$= \frac{1}{b}\sum_{p\in X}\frac{\partial E_p(\mathbf{w})}{\partial x_{i,p}^s} f'\left(z_{i,p}^s\right) x_{j,p}^{s-1}. \quad (8)$$

Differentiation of the activation function

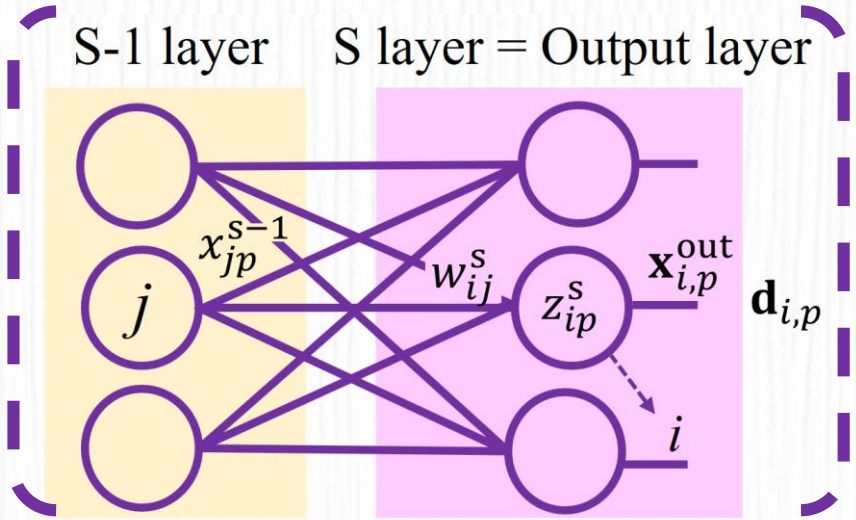
Stochastic Gradient Descent (SGD)

If S layer is output layer (S = out):

b: mini − batch size, p: #. Sample pattern

Back Propagation: for all p include to b

➢ $\frac{\partial E_p(\mathbf{w})}{\partial x_{i,p}^{s}} = \frac{\partial E_p(\mathbf{w})}{\partial x_{i,p}^{\text{out}}}$ is the output layer's differentiation



If error function = Mean Square Error (MSE):

MSE: $$E_b(\mathbf{w}) = \frac{1}{b}\sum_{p\in X}\frac{1}{2}\left\|d_{i,p} - x_{i,p}^{\text{out}}\right\|^2. \qquad (9)$$

$d_{i,p}$: $p-$ th teach signal for $i-$ th neuron

➢ Gradient $\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}}$ is :

$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}} = \frac{1}{b}\sum_{p\in X}\frac{\partial E_p(\mathbf{w})}{\partial x_{i,p}^{s}}\cdot f'\left(z_{i,p}^{s}\right)\cdot x_{j,p}^{s-1} = \frac{1}{b}\sum_{p\in X} -(d_{i,p} - x_{i,p}^{\text{out}})\cdot f'\left(z_{i,p}^{s}\right)\cdot x_{j,p}^{s-1}. \qquad (10)$$

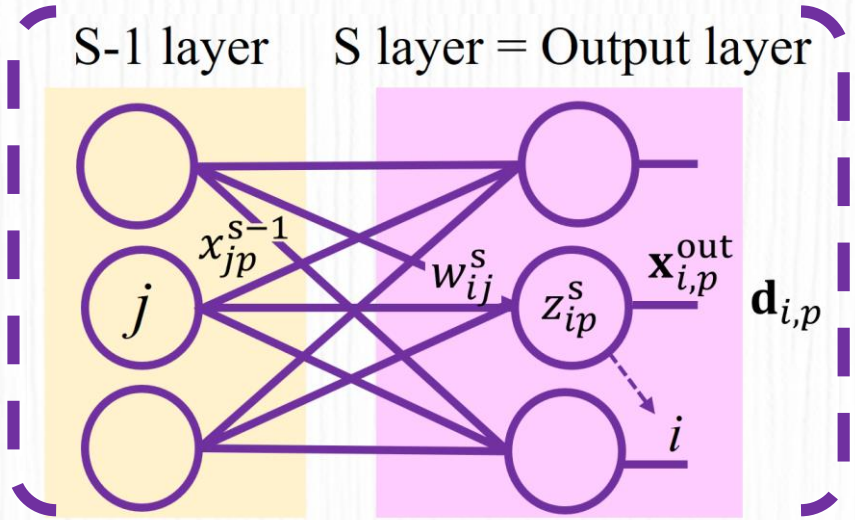
Stochastic Gradient Descent (SGD)

If S layer is output layer (S = out):

b: mini − batch size, p: #. Sample pattern

Back Propagation: for all p include to b

➢ $\frac{\partial E_p(\mathbf{w})}{\partial x_{i,p}^{s}} = \frac{\partial E_p(\mathbf{w})}{\partial x_{i,p}^{out}}$ is the output layer's differentiation



If error function = Cross Entropy (CE), and activation function = softmax function:

CE: $$E_b(\mathbf{w}) = -\frac{1}{b}\sum_{p\in X}\sum_{l=1}^{L} d_{l,p}\log(x_{l,p}^{out}), \quad (11)$$

L: #. classification class

Softmax: $$x_{l,p}^{out} = f(z_{l,p}^{s}) = \frac{\exp(z_{l,p}^{s})}{\sum_{l=1}^{L}\exp(z_{l,p}^{s})}. \quad (12)$$

➢ If $l = i$, gradient $\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}}$ is:

$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}} = \frac{1}{b}\sum_{p\in X}\frac{\partial E_p(\mathbf{w})}{\partial x_{i,p}^{s}}\cdot f'(z_{i,p}^{s})\cdot x_{j,p}^{s-1} = -\frac{1}{b}\sum_{p\in X}(d_{i,p} - x_{i,p}^{out})\cdot x_{j,p}^{s-1}. \quad (13)$$

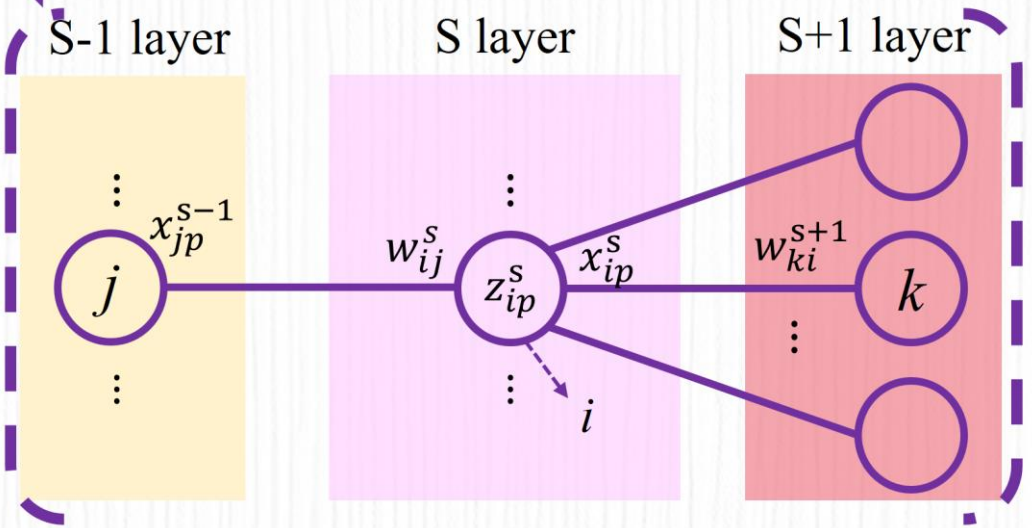
Stochastic Gradient Descent (SGD)

If S layer is other than output layer：

$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^s} = \frac{1}{b}\sum_{p\in X}\frac{\partial E_p(\mathbf{w})}{\partial x_{i,p}^s}\cdot f'\left(z_{i,p}^s\right).x_{j,p}^{s-1} \quad (8)$$

b：mini − batch size,　p：#. Sample pattern

Back Propagation： for all p include in b



➢ $\frac{\partial E_p(\mathbf{w})}{\partial x_{i,p}^s}$ is :

$$\frac{\partial E_p(\mathbf{w})}{\partial x_{i,p}^s} = \sum_k \frac{\partial E_p(\mathbf{w})}{\partial x_{k,p}^{s+1}}\cdot\frac{\partial x_{k,p}^{s+1}}{\partial z_{k,p}^{s+1}}\cdot\frac{\partial z_{k,p}^{s+1}}{\partial x_{i,p}^s}$$

$$= \sum_k \frac{\partial E_p(\mathbf{w})}{\partial x_{k,p}^{s+1}}\cdot f'(z_{k,p}^{s+1})\cdot w_{ki}^{s+1}. \quad (14)$$

Require for all p include into b

➢ The gradient $\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^s}$ is :

The inner product of the weight and back propagation component

Back Propagation component

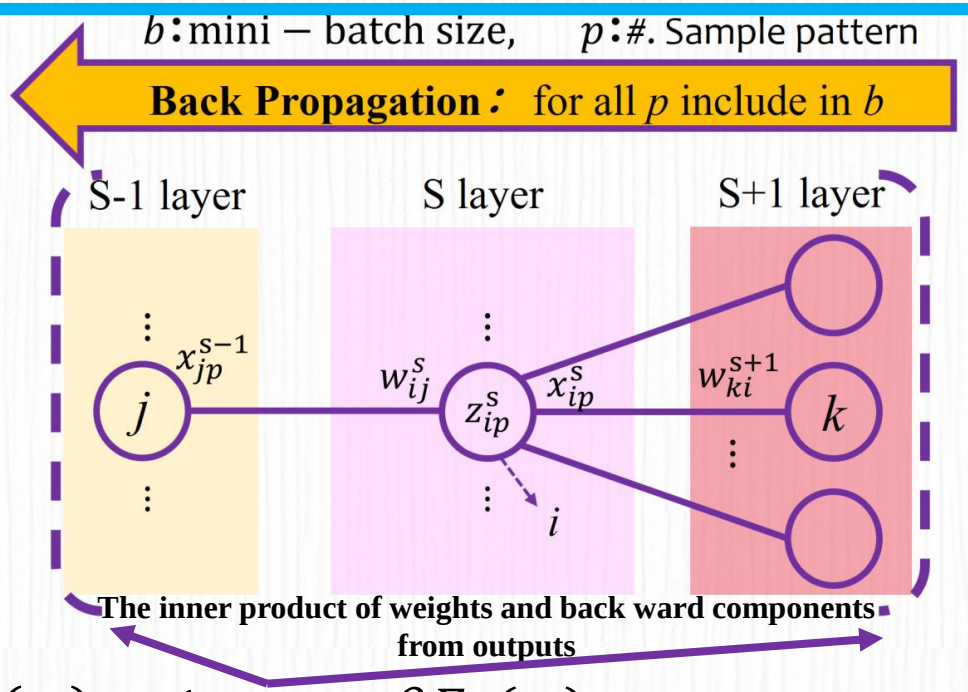
$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^s} = \frac{1}{b}\sum_{p\in X}\sum_k \left(\frac{\partial E_p(\mathbf{w})}{\partial x_{k,p}^{s+1}}\cdot f'(z_{k,p}^{s+1})\right)\cdot w_{ki}^{s+1}\cdot f'\left(z_{i,p}^s\right)\cdot x_{j,p}^{s-1}. \quad (15)$$

Propose method : Stochastic Weight Difference Propagation(SWDP)

Back propagation component for b ⊂ Weights update amount (difference)

In SGD training：

Require the inner product of weights and **back propagation components of output for all p include in b**



$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}} = \frac{1}{b}\sum_{p\in X}\sum_{k}\frac{\partial E_p(\mathbf{w})}{\partial x_{k,p}^{s+1}}\cdot f'(z_{k,p}^{s+1})\cdot w_{ki}^{s+1}\cdot f'\left(z_{i,p}^{s}\right)\cdot x_{j,p}^{s-1}. \quad (15)$$

Propose method : Stochastic Weight Difference Propagation(SWDP)

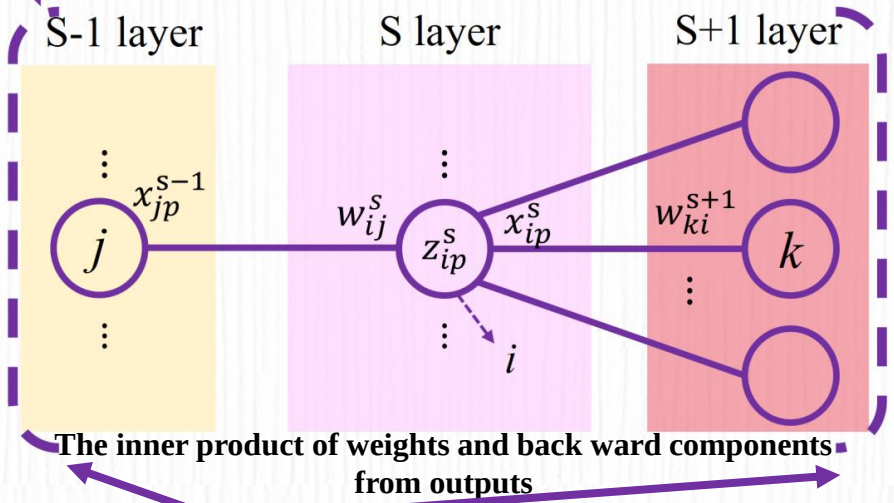
Back propagation component for b ≅ Weights update amount (difference)

In SGD training：

Require the inner product of weights and **back propagation components of output for all p include in b**

b：mini − batch size, p：#. Sample pattern

Back Propagation： for all p include in b



The inner product of weights and back ward components from outputs

$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}} = \frac{1}{b}\sum_{p\in X}\sum_{k}\frac{\partial E_p(\mathbf{w})}{\partial x_{k,p}^{s+1}}\cdot f'(z_{k,p}^{s+1})\cdot w_{ki}^{s+1}\cdot f'\left(z_{i,p}^{s}\right)\cdot x_{j,p}^{s-1}. \quad (15)$$

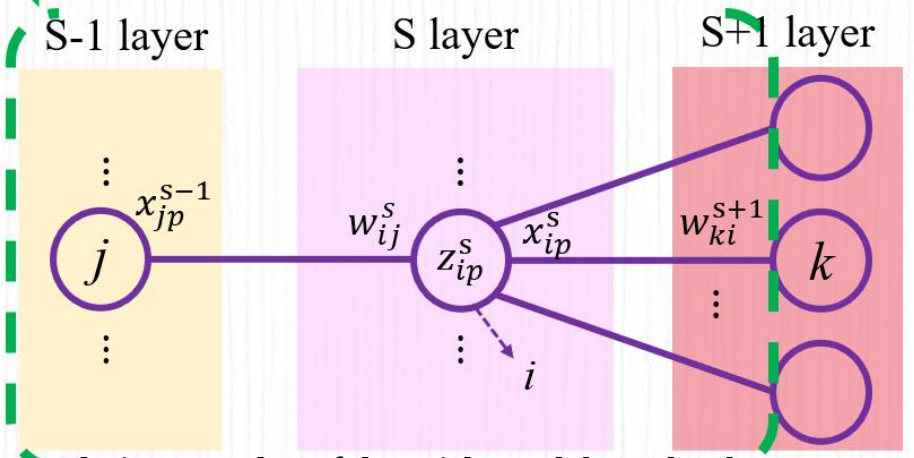
Propose method：SWDP

In SWDP training：

It can be calculated by the inner product of the weights and **the difference between the weights**

b：mini − batch size, p：#. Sample pattern

Back Propagation： for all p include in b



The inner product of the weights and the updated (difference) weights

Propose method : Stochastic Weight Difference Propagation(SWDP)

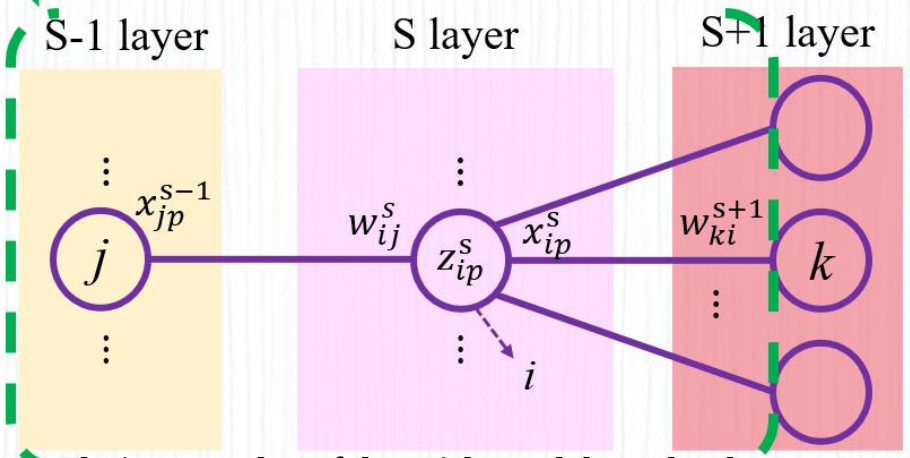
Propose method：SWDP

In SWDP training：

It can be calculated by the inner product of the weights and **the difference between the weights**

b：mini − batch size,　p：#. Sample pattern

Back Propagation：for all p include in b



The inner product of the weights and the updated (difference) weights

➢The update formula of weight w_{ij}^{s} in SGD:

$$w_{ij}^{s}(t+1) = w_{ij}^{s}(t) - \eta \frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}}, \qquad (7)$$

t：#. Iteration counts,　η：Learning rate

➢ The gradient $\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}}$ is :

$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}} = \frac{1}{b}\sum_{p\in X} \frac{\partial E_p(\mathbf{w})}{\partial x_{i,p}^{s}} \cdot f'\left(z_{i,p}^{s}\right) \cdot x_{j,p}^{s-1}. \qquad (8)$$

If S layer is output layer（S = out）:

If S layer is other than output layer：

Propose method : Stochastic Weight Difference Propagation(SWDP)

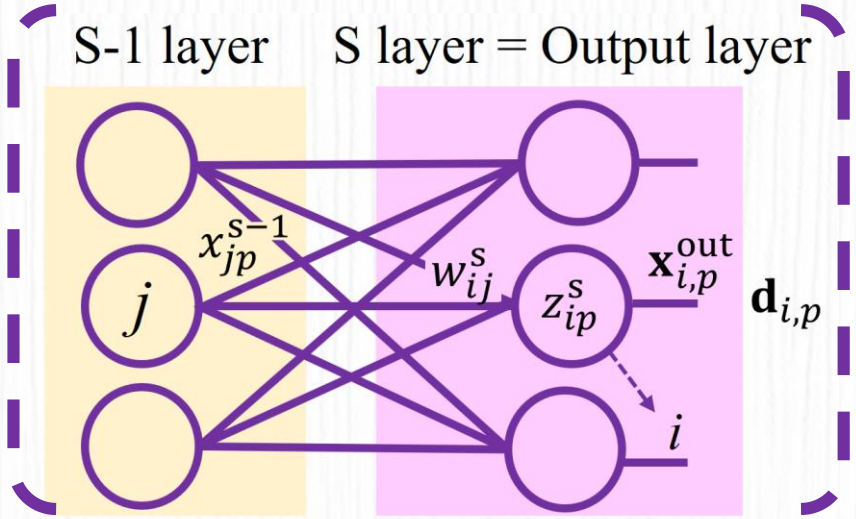
If S layer is output layer（S = out）: **SWDP and SGD have the same calculation for gradient**

b：mini − batch size, p：#. sample pattern

- $\frac{\partial E_p(\mathbf{w})}{\partial x_{i,p}^{s}} = \frac{\partial E_p(\mathbf{w})}{\partial x_{i,p}^{\text{out}}}$ is the output layer's differentiation

Back Propagation：for all p include to b

- The gradient $\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}}$ is :



If error function＝Mean Square Error (MSE)：

$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}} = \frac{1}{b}\sum_{p\in X} -(d_{i,p} - x_{i,p}^{\text{out}}) \cdot f'(z_{i,p}^{s}) \cdot x_{j,p}^{s-1}. \quad (10)$$

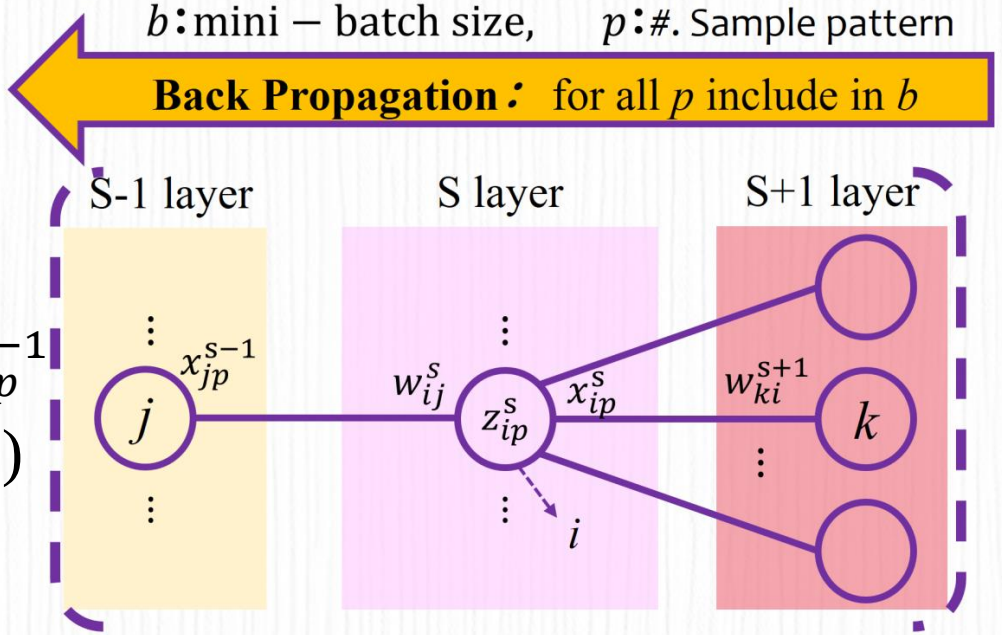
$d_{i,p}$：Teach signal

If error function＝Cross Entropy，and activation function＝softmax：

$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}} = -\frac{1}{b}\sum_{p\in X} (d_{i,p} - x_{i,p}^{\text{out}}) \cdot x_{j,p}^{s-1}. \quad (13)$$

Propose method : Stochastic Weight Difference Propagation(SWDP)

If S layer is other than output layer：



- The gradient $\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}}$ is :

$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}} = \frac{1}{b}\sum_{p\in X}\sum_{k}\frac{\partial E_p(\mathbf{w})}{\partial x_{k,p}^{s+1}}\cdot f'(z_{k,p}^{s+1})\cdot w_{ki}^{s+1}\cdot f'\left(z_{i,p}^{s}\right)\cdot x_{j,p}^{s-1} \quad (15)$$

Exchange

$$= \frac{1}{b}\sum_{k}\sum_{p\in X}\frac{\partial E_p(\mathbf{w})}{\partial x_{k,p}^{s+1}}\cdot f'(z_{k,p}^{s+1})\cdot w_{ki}^{s+1}\cdot f'\left(z_{i,p}^{s}\right)\cdot x_{j,p}^{s-1} \quad (16)$$

- By assuming $x_{i,p}^{s} \neq 0$, (16) can be transformed as in (17) :

$$= \sum_{k}\frac{1}{b}\sum_{p\in X}\frac{\partial E_p(\mathbf{w})}{\partial x_{k,p}^{s+1}}\cdot f'(z_{k,p}^{s+1}).x_{i,p}^{s}\cdot w_{ki}^{s+1}\cdot\frac{f'\left(z_{i,p}^{s}\right)}{x_{i,p}^{s}}\cdot x_{j,p}^{s-1} \quad (17)$$

Propose method : Stochastic Weight Difference Propagation(SWDP)

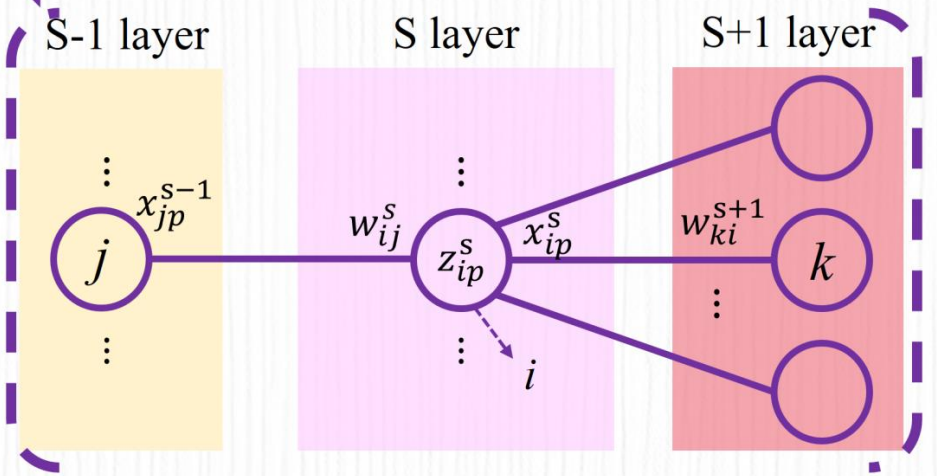
If S layer is other than output layer：

$$= \sum_k \frac{1}{b} \sum_{p \in X} \frac{\partial E_p(\mathbf{w})}{\partial x_{k,p}^{s+1}} \cdot f'\left(z_{k,p}^{s+1}\right) . x_{i,p}^{s} \cdot w_{ki}^{s+1} \cdot \frac{f'\left(z_{i,p}^{s}\right)}{x_{i,p}^{s}} \cdot x_{j,p}^{s-1} \quad (17)$$

$$= \sum_k \left(\frac{1}{b} \sum_{p \in X} \frac{\partial E_p(\mathbf{w})}{\partial x_{k,p}^{s+1}} \cdot f'\left(z_{k,p}^{s+1}\right) . x_{i,p}^{s} \cdot w_{ki}^{s+1} \cdot \frac{f'\left(z_{i,p}^{s}\right)}{x_{i,p}^{s}} \cdot x_{j,p}^{s-1} \right) \quad (17)$$

b：mini − batch size, p：#. Sample pattern

Back Propagation： for all p include in b



- Approximating (17) from the product of mean to the mean of product :

$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}} \cong \sum_k \left(\frac{1}{b} \sum_{p \in X} \frac{\partial E_p(\mathbf{w})}{\partial x_{k,p}^{s+1}} \cdot f'\left(z_{k,p}^{s+1}\right) \cdot x_{i,p}^{s} \cdot w_{ki}^{s+1} \right) \cdot \left(\frac{1}{b} \sum_{p \in X} \frac{f'\left(z_{i,p}^{s}\right)}{x_{i,p}^{s}} \cdot x_{j,p}^{s-1} \right) . \quad (18)$$

Propose method : Stochastic Weight Difference Propagation(SWDP)

If S layer is other than output layer：

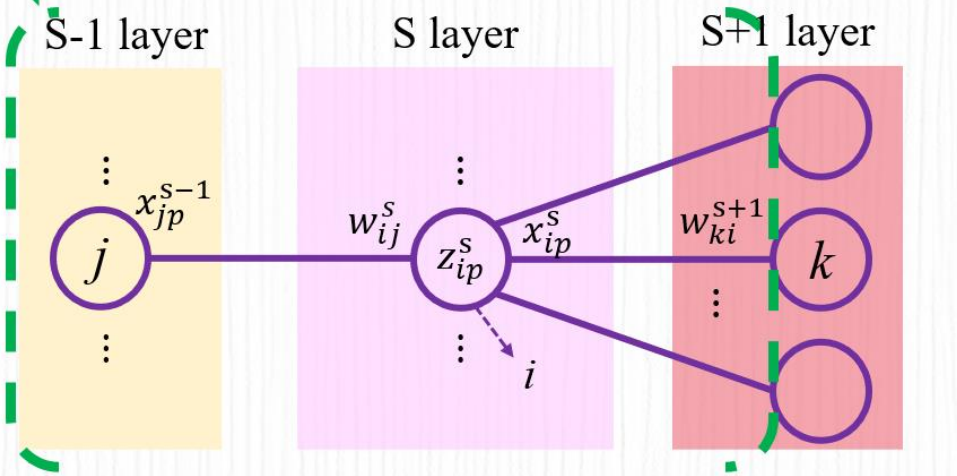
$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}} \cong \sum_k \left(\frac{1}{b} \sum_{p \in X} \frac{\partial E_p(\mathbf{w})}{\partial w_{ki}^{s+1}} \cdot w_{ki}^{s+1} \right) \cdot \left(\frac{1}{b} \sum_{p \in X} \frac{f'\left(z_{i,p}^{s}\right)}{x_{i,p}^{s}} \cdot x_{j,p}^{s-1} \right). \quad (18)$$

$$\frac{\partial E_p(\mathbf{w})}{\partial x_{k,p}^{s+1}} \cdot f'\left(z_{k,p}^{s+1}\right) \cdot x_{i,p}^{s} = \frac{\partial E_p(\mathbf{w})}{\partial w_{ki}^{s+1}} \quad (19)$$

$$\frac{1}{b} \sum_{p \in X} \frac{\partial E_p(\mathbf{w})}{\partial w_{ki}^{s+1}} = \frac{\partial E_b(\mathbf{w})}{\partial w_{ki}^{s+1}} \quad (20)$$

b：mini − batch size,　p：#. Sample pattern

Back Propagation：for all p include in b



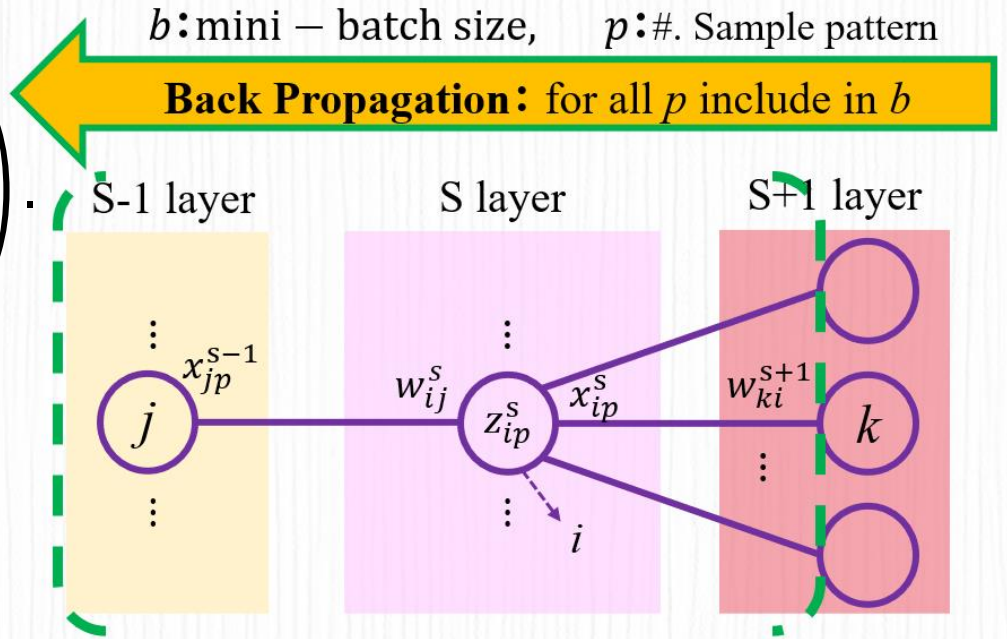
➢ Substitute $\frac{\partial E_b(\mathbf{w})}{\partial w_{ki}^{s+1}}$ in (20) into (18) :

$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}} \cong \sum_k \left(\frac{\partial E_b(\mathbf{w})}{\partial w_{ki}^{s+1}} \cdot w_{ki}^{s+1} \right) \cdot \left(\frac{1}{b} \sum_{p \in X} \frac{f'\left(z_{i,p}^{s}\right)}{x_{i,p}^{s}} \cdot x_{j,p}^{s-1} \right). \quad (21)$$

Propose method : Stochastic Weight Difference Propagation(SWDP)

If S layer is other than output layer：

$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}} \cong \sum_k \left(\frac{\partial E_b(\mathbf{w})}{\partial w_{ki}^{s+1}} \cdot w_{ki}^{s+1} \right) \cdot \left(\frac{1}{b} \sum_{p \in X} \frac{f'\left(z_{i,p}^{s}\right)}{x_{i,p}^{s}} \cdot x_{j,p}^{s-1} \right) \tag{21}$$



- The update formula of weight w_{ki}^{s+1} in SGD:

$$w_{ki}^{s+1}(t+1) = w_{ki}^{s+1}(t) - \eta \frac{\partial E_b(\mathbf{w})}{\partial w_{ki}^{s+1}}, \tag{22}$$

- The update amount (difference) of weight w_{ki}^{s+1} is :

$$\Delta w_{ki}^{s+1} = w_{ki}^{s+1}(t+1) - w_{ki}^{s}(t) = -\eta \frac{\partial E_b(\mathbf{w})}{\partial w_{ki}^{s+1}}, \tag{23}$$

$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ki}^{s+1}} = -\frac{1}{\eta} \Delta w_{ki}^{s+1} \tag{24}$$

Propose method : Stochastic Weight Difference Propagation(SWDP)

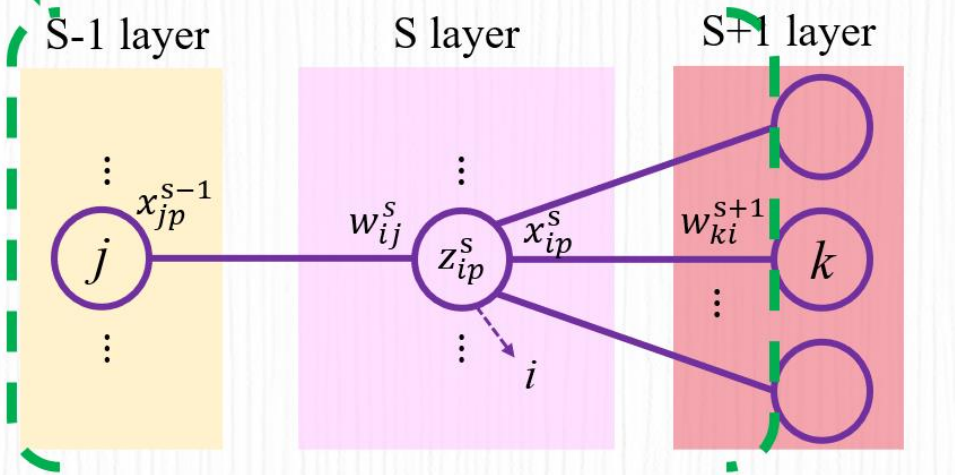
If S layer is other than output layer：

$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}} \cong \sum_k \left(\frac{\partial E_b(\mathbf{w})}{\partial w_{ki}^{s+1}} \cdot w_{ki}^{s+1} \right) \cdot \left(\frac{1}{b} \sum_{p \in X} \frac{f'\left(z_{i,p}^{s}\right)}{x_{i,p}^{s}} \cdot x_{j,p}^{s-1} \right). \quad (21)$$

$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ki}^{s+1}} = -\frac{1}{\eta} \Delta w_{ki}^{s+1} \quad (24)$$

b：mini − batch size,　p：#. Sample pattern

Back Propagation：for all p include in b



➢ Substitute $-\frac{1}{\eta}\Delta w_{ki}^{s+1}$ in (24) into (21) :

$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^{s}} \cong \sum_k \left(-\frac{1}{\eta} \Delta w_{ki}^{s+1} \cdot w_{ki}^{s+1} \right) \cdot \left(\frac{1}{b} \sum_{p \in X} \frac{f'\left(z_{i,p}^{s}\right)}{x_{i,p}^{s}} \cdot x_{j,p}^{s-1} \right). \quad (25)$$

Propose method : Stochastic Weight Difference Propagation(SWDP)

If S layer is other than output layer：

- The gradient $\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^s}$ is :

$$\frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^s} \cong \sum_k \left(-\frac{1}{\eta} \Delta w_{ki}^{s+1} \cdot w_{ki}^{s+1} \right) \cdot \left(\frac{1}{b} \sum_{p \in X} \frac{f'\left(z_{i,p}^s\right)}{x_{i,p}^s} \cdot x_{j,p}^{s-1} \right). \quad (25)$$

- The update formula of weight w_{ij}^s in SGD :

$$w_{ij}^s(t+1) = w_{ij}^s(t) - \eta \frac{\partial E_b(\mathbf{w})}{\partial w_{ij}^s}, \quad (7)$$

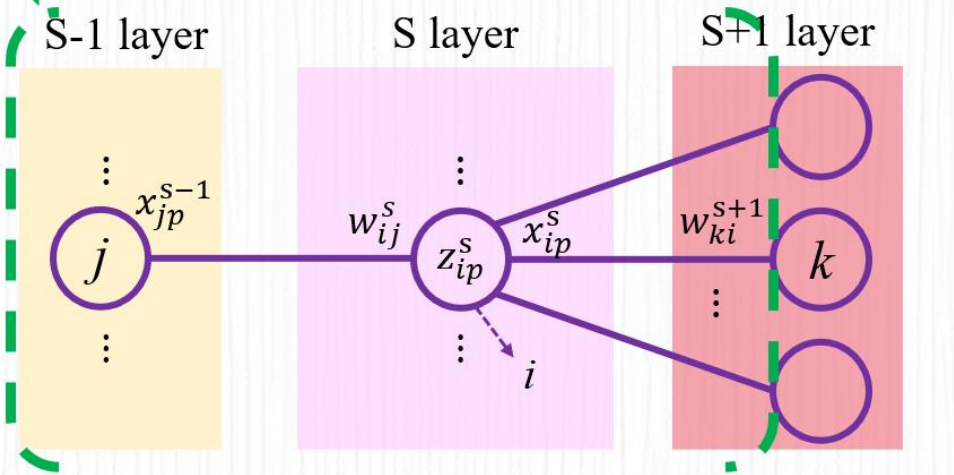
t：#. Iteration counts, η：Learning rate

- The update formula of weight w_{ij}^s in SWDP :

$$= w_{ij}^s(t) + \sum_k \left(\Delta w_{ki}^{s+1} \cdot w_{ki}^{s+1} \right) \cdot \left(\frac{1}{b} \sum_{p \in X} \frac{f'\left(z_{i,p}^s\right)}{x_{i,p}^s} \cdot x_{j,p}^{s-1} \right) \quad (26)$$

b：mini − batch size, p：#. Sample pattern

Back Propagation：for all p include in b



In **SWDP** training：
It can be calculated by the inner product of the weights and **the difference between the weights**

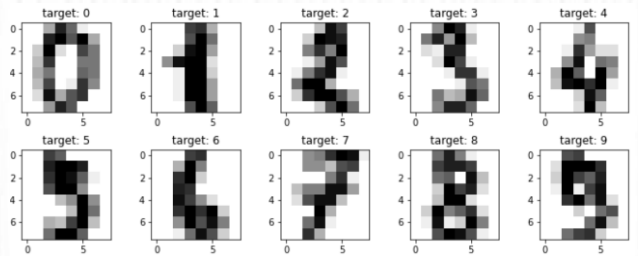
Experiments

2-bit parity problem

- **#. Training data**: $|T_r| = 4$
- **Training algorithms**:
  SGD & SWDP
- **Error and activation functions**:
  MSE & Sigmoid
- **mini-batch size**: b = 1, 2 & 3
- **Learning rate**: $\eta = 0.1$
- **Input**: 2 **Output**: 2
- **Network structure**:
  With 1 hidden layer 2-10-2
  With 2 hidden layer 2-10-10-2

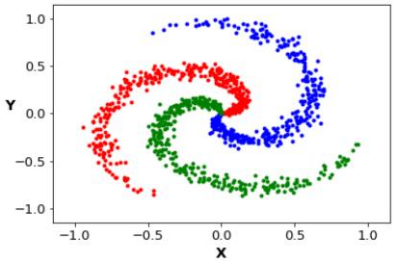
8 × 8 MNIST

- **#. Training data**: $|T_r| = 1{,}347$
- **#. Test data**: $|T_e| = 450$
- **Training algorithms**:
  SGD & SWDP
- **mini-batch size**:
  b = 1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20 & 32
- **Learning rate**: $\eta = 0.1$
- **Input**: 64 **Output**: 10
- **Network structure**:
  With 1 hidden layer 64-10-10



3 Spiral

- **#. Training data**: $|T_r| = 1{,}050$
- **Training algorithms**:
  SGD & SWDP
- **mini-batch size**:
  b = 1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20 & 32
- **Learning rate**: $\eta = 0.1$
- **Input**: 2 **Output**: 3
- **Network structure**:
  With 1 hidden layer 2-10-3



Experiments

2-bit parity problem

- **#. Training data：** $|T_r| = 4$
- **Training algorithms：**

  SGD & SWDP
- **Error and activation functions：**

  MSE & Sigmoid
- **mini-batch size：** b = 1, 2 & 3
- **Learning rate：** $\eta = 0.1$
- **Input：** 2 **Output：** 2
- **Network structure：**

  With 1 hidden layer 2-10-2
  With 2 hidden layer 2-10-10-2

8 × 8 MNIST

- **#. Training data：** $|T_r| = 1{,}347$
- **#. Test data：** $|T_e| = 450$
- **Training algorithms：**

  SGD & SWDP
- **mini-batch size：**

  b = 1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20 & 32
- **Learning rate：** $\eta = 0.1$
- **Input：** 64 **Output：** 10
- **Network structure：**

  With 1 hidden layer 64-10-10

3 Spiral

- **#. Training data：** $|T_r| = 1{,}050$
- **Training algorithms：**

  SGD & SWDP
- **mini-batch size：**

  b = 1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20 & 32
- **Learning rate：** $\eta = 0.1$
- **Input：** 2 **Output：** 3
- **Network structure：**

  With 1 hidden layer 2-10-3

Experiment results ①:2-bit parity problem (NN： 2-10-2)

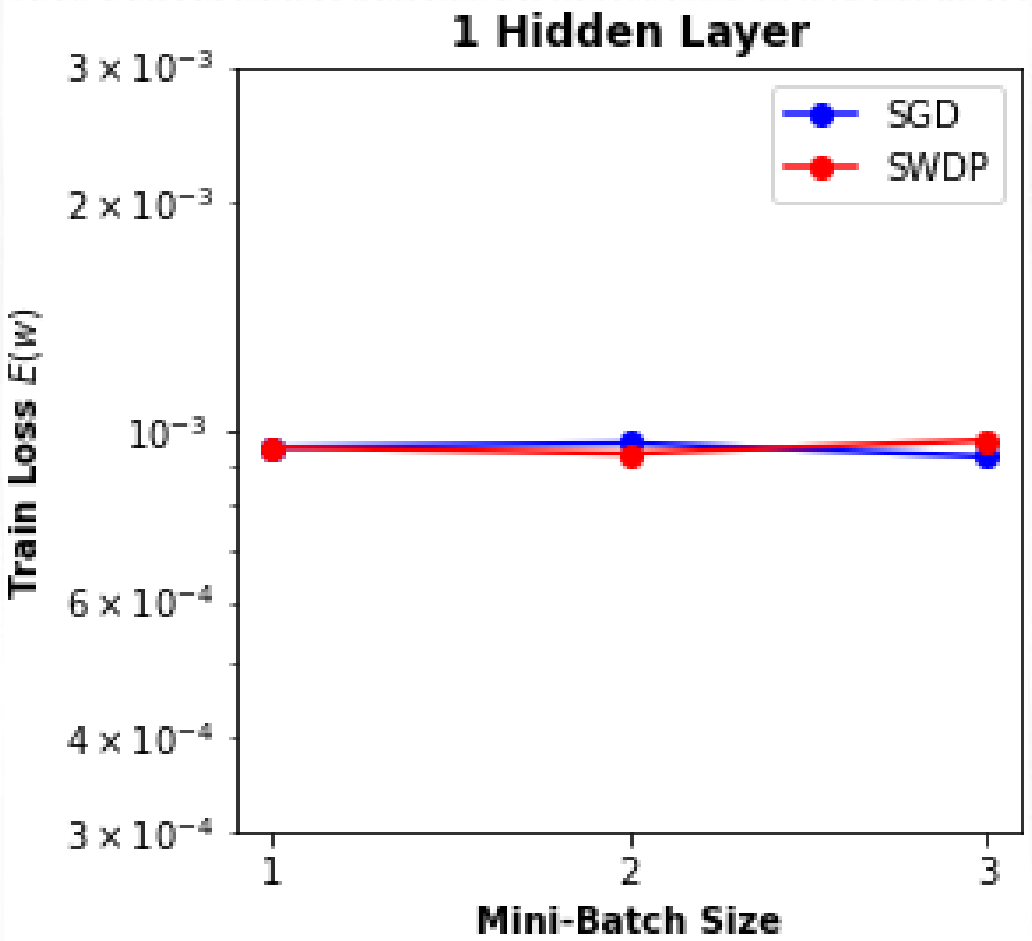


Fig 1. Change in training error for mini-batches

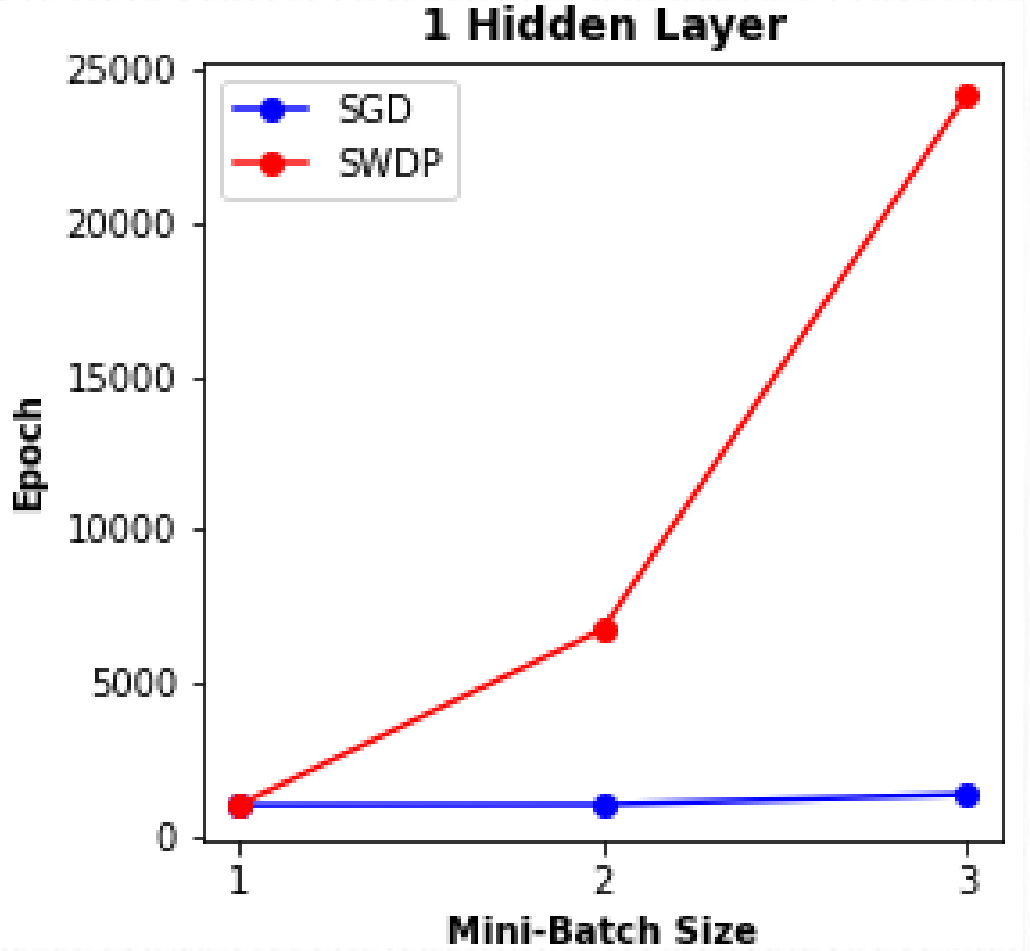


Fig 2. Change in the iteration count for mini-batches

Experiment results ①:2-bit parity problem(NN： 2-10-10-2)

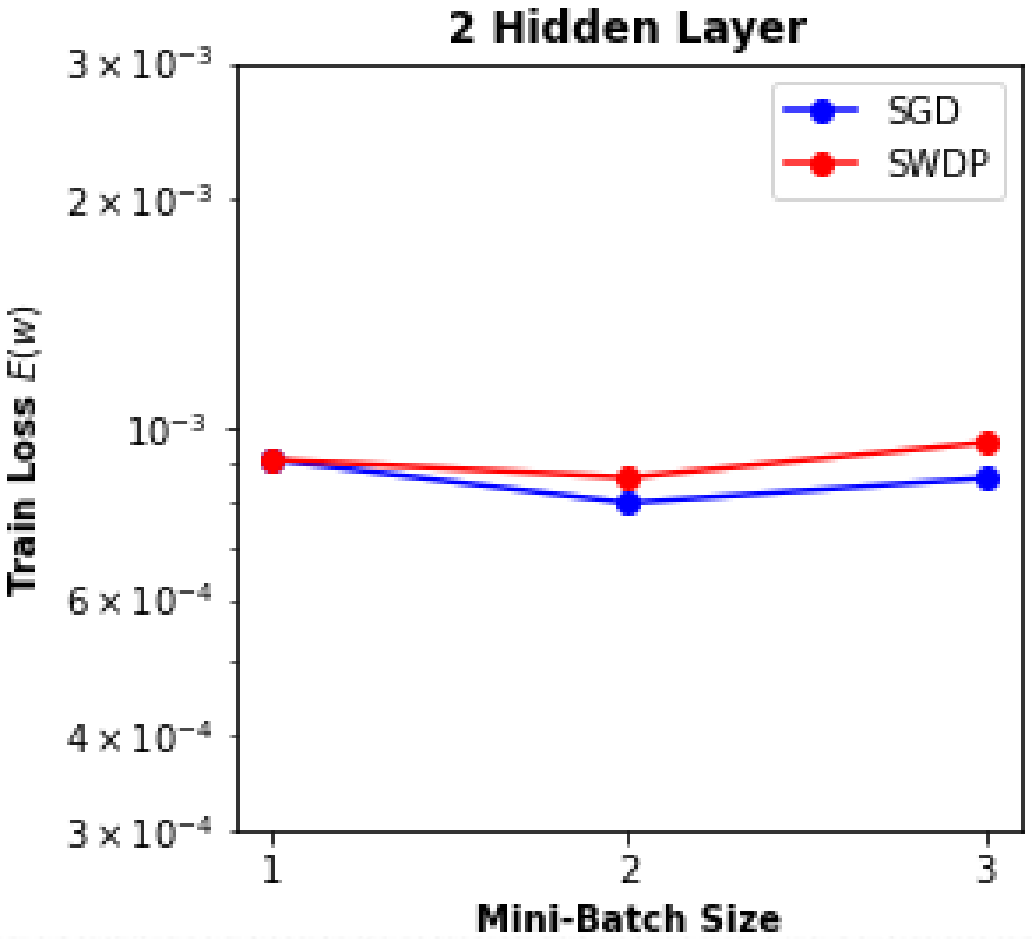


Fig 3. Change in training error for mini-batches

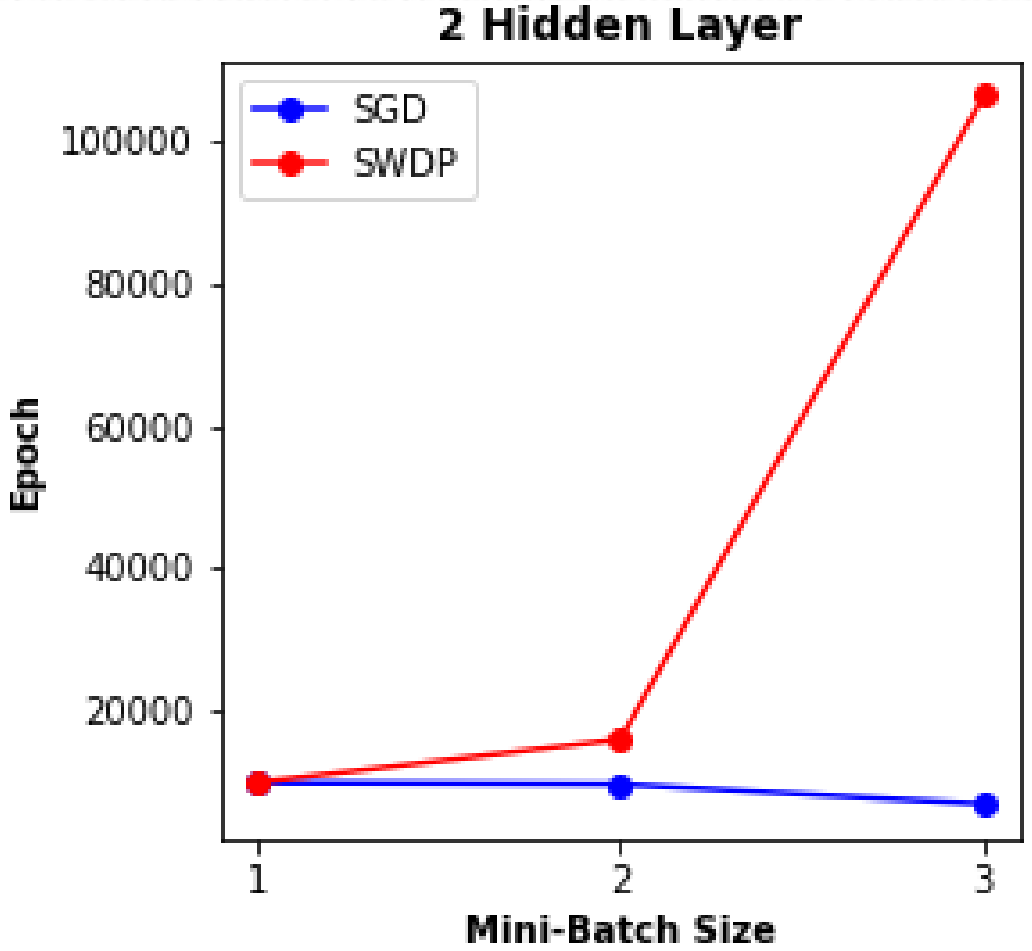


Fig 4. Change in the iteration count for mini-batches

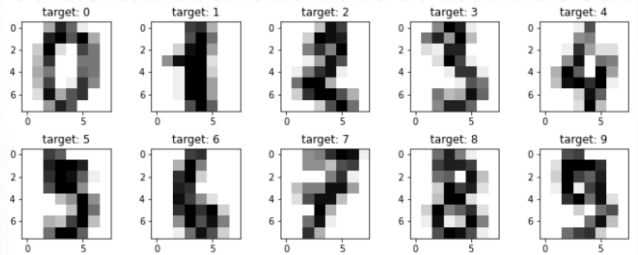
Experiments

2-bit parity problem

- #. Training data: $|T_r| = 4$
- Training algorithms:
  SGD & SWDP
- Error and activation functions:
  MSE & Sigmoid
- mini-batch size: b = 1, 2 & 3
- Learning rate: $\eta = 0.1$
- Input: 2 Output: 2
- Network structure:
  With 1 hidden layer 2-10-2
  With 2 hidden layer 2-10-10-2

8 × 8 MNIST

- **#. Training data: $|T_r| = 1{,}347$**
- **#. Test data: $|T_e| = 450$**
- **Training algorithms:**
  SGD & SWDP
- **mini-batch size:**
  b = 1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20 & 32
- **Learning rate: $\eta = 0.1$**
- **Input: 64 Output: 10**
- **Network structure:**
  With 1 hidden layer 64-10-10



3 Spiral

- #. Training data: $|T_r| = 1{,}050$
- Training algorithms:
  SGD & SWDP
- mini-batch size:
  b = 1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20 & 32
- Learning rate: $\eta = 0.1$
- Input: 2 Output: 3
- Network structure:
  With 1 hidden layer 2-10-3

Experiment results ②： 8 × 8 MNIST

Fig 5. Change in accuracy for mini batches

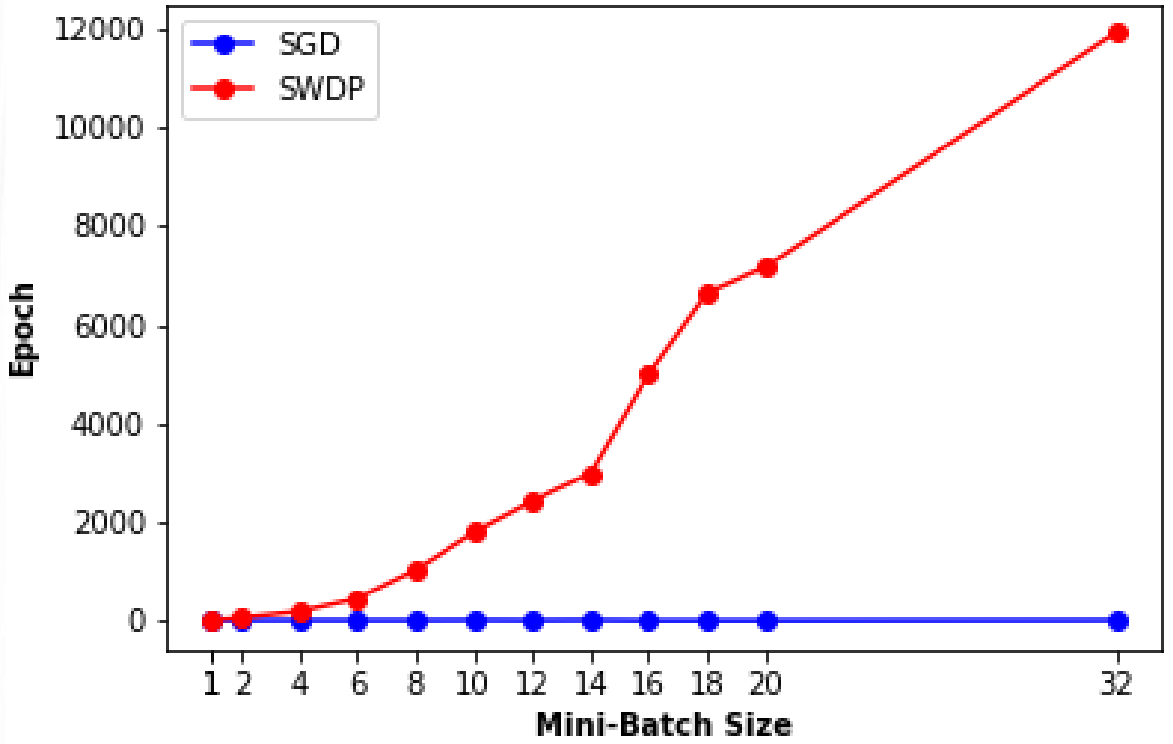


Fig 6. Change in the iteration count for mini-batches

Experiments

2-bit parity problem

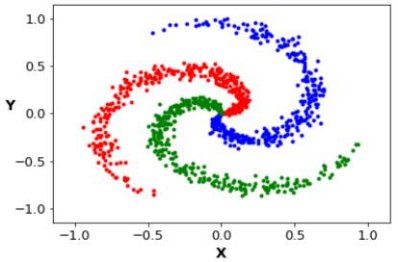
- #. Training data: $|T_r| = 4$
- Training algorithms:
  SGD & SWDP
- Error and activation functions:
  MSE & Sigmoid
- mini-batch size: b = 1, 2 & 3
- Learning rate: $\eta = 0.1$
- Input: 2 Output: 2
- Network structure:
  With 1 hidden layer 2-10-2
  With 2 hidden layer 2-10-10-2

8 × 8 MNIST

- #. Training data: $|T_r| = 1{,}347$
- #. Test data: $|T_e| = 450$
- Training algorithms:
  SGD & SWDP
- mini-batch size:
  b = 1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20 & 32
- Learning rate: $\eta = 0.1$
- Input: 64 Output: 10
- Network structure:
  With 1 hidden layer 64-10-10

3 Spiral

- **#. Training data: $|T_r| = 1{,}050$**
- **Training algorithms:**
  SGD & SWDP
- **mini-batch size:**
  b = 1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20 & 32
- **Learning rate: $\eta = 0.1$**
- **Input: 2 Output: 3**
- **Network structure:**
  With 1 hidden layer 2-10-3



Experiment results③： 3 Spiral

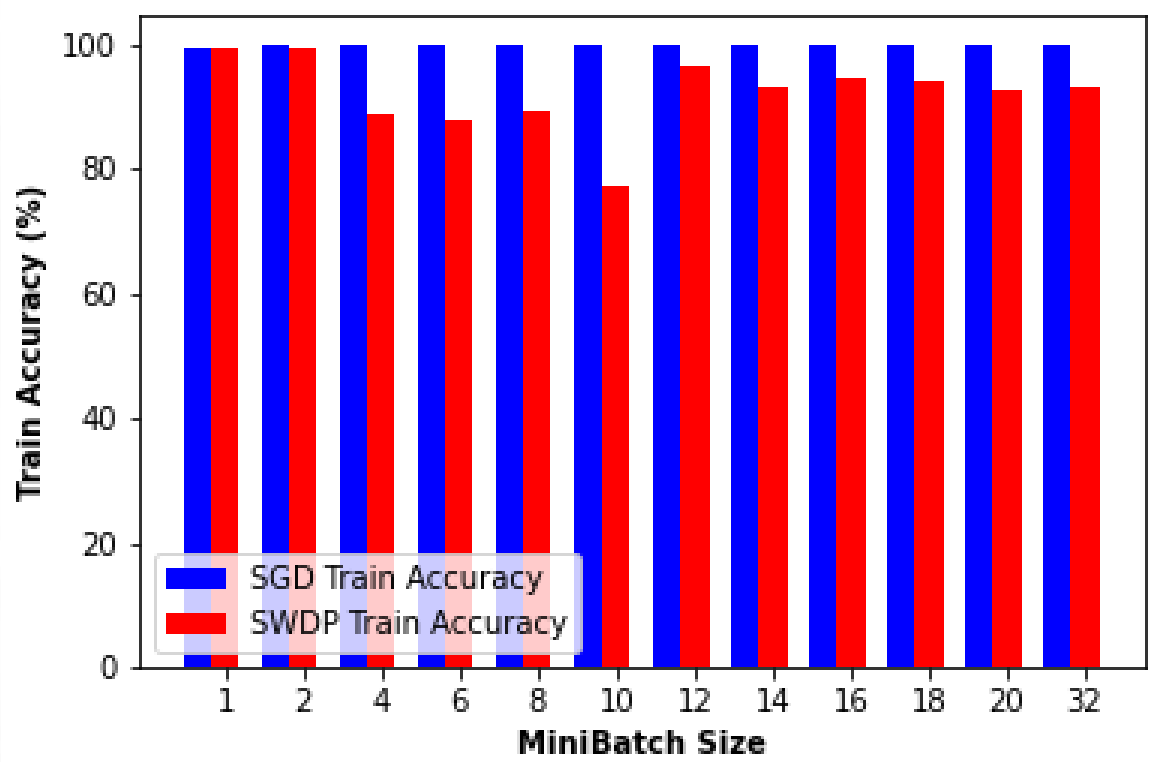


Fig 7. Change in training accuracy for mini-batch size

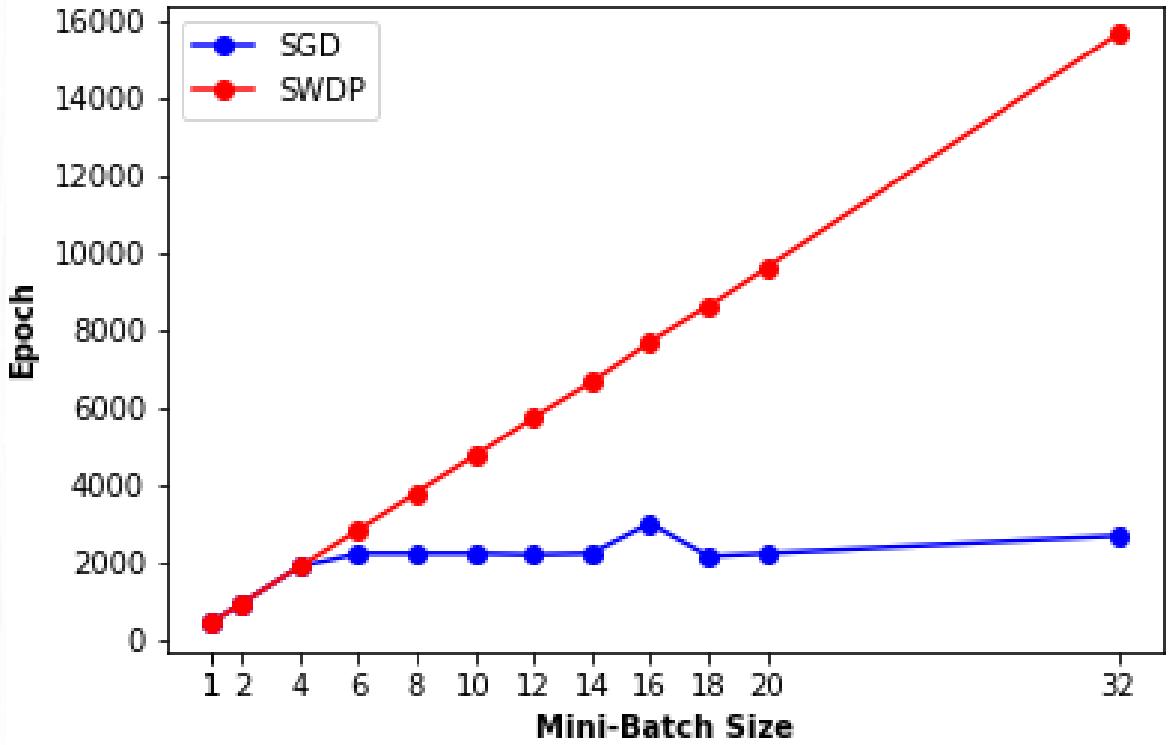


Fig 8. Change in the iteration count for mini-batches

Conclusion

To decrease the cost of Stochastic Gradient Descent (SGD)

↓ **Proposed method to improve**

Stochastic Weight Difference Propagation (SWDP)

Back propagation component for *b* ≅ Weights update amount (difference)

In SGD training：

Require the inner product of weight and **back propagation component** of weight **for all p include to b**

In SWDP training：

It can be calculated by the inner product of the weights and **the difference between the weights**

- Experiments shows that although the #. iterations increases, SWDP can train with almost the same accuracy as SGD.
- In future works:
  1. Improve SWDP to allow for robust training.
  2. Investigation of the effectiveness of the SWDP for huge and complex problems.
  3. Implement the SWDP in hardware such as FPGA and verify its effectiveness.

THANK YOU !