딥러닝 첫걸음

4. 신경망과 분류 (MultiClass)

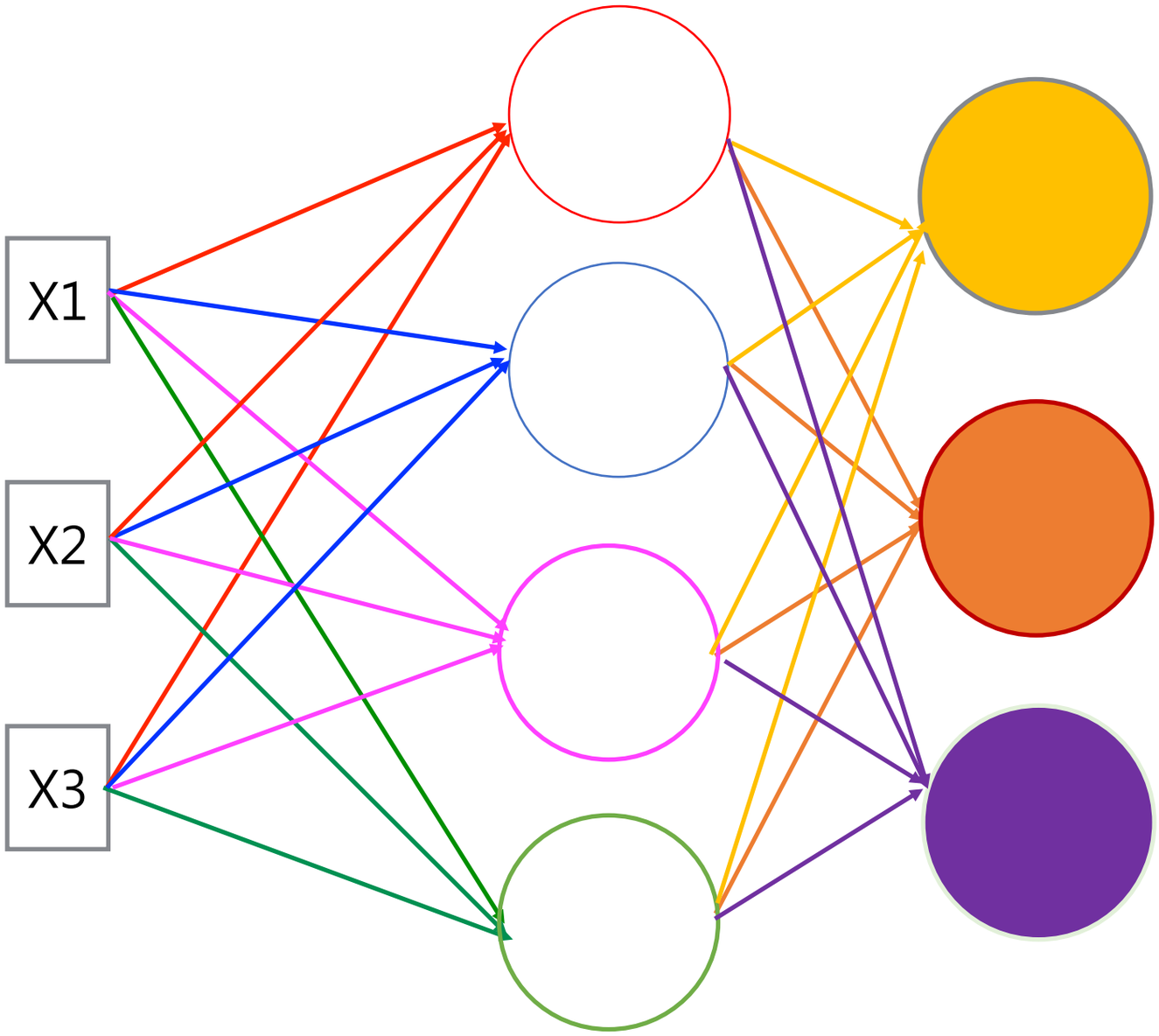
다범주 분류 신경망

- Categorization(분류): 예측 대상 = 범주

- 이진분류: 예측 대상 범주가 2가지인 경우
  - 출력층 node 1개 다층신경망 분석 (3장의 내용)

- 다범주 분류: 예측 대상 범주가 3가지 이상인 경우
  - 출력층 node 2개 이상 다층신경망 분석
  - 비용함수 : Softmax 함수 사용

다범주 분류신경망



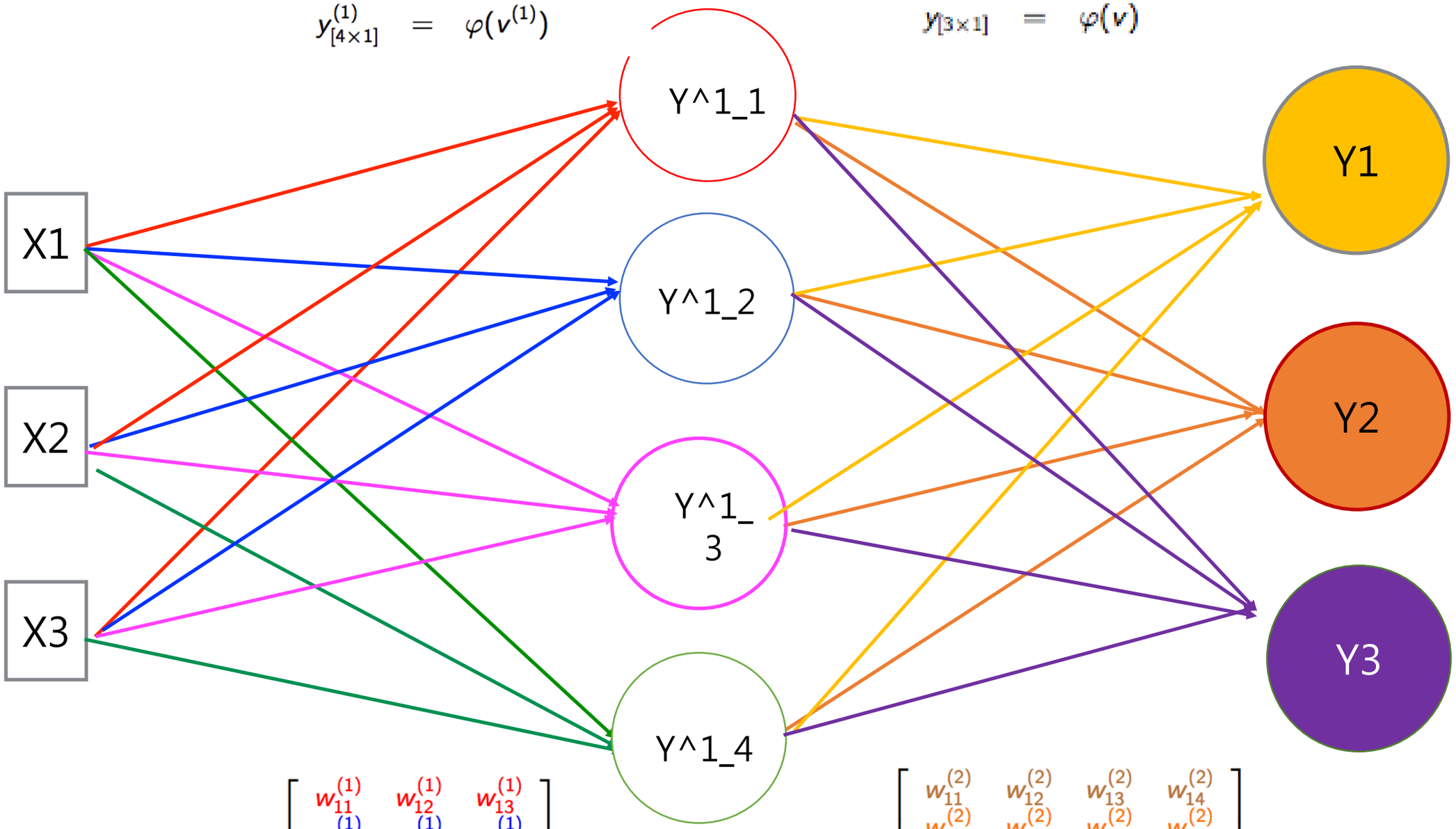
D =

범주 1.	범주2	범주 3
1	0	0
0	1	0
0	0	1

다범주 분류 신경망: 입출력

$$v^{(1)}_{[4\times1]} = W^{(1)}_{[4\times3]} X_{[3\times1]}$$
$$y^{(1)}_{[4\times1]} = \varphi(v^{(1)})$$

$$v_{[3\times1]} = W^{(L)}_{[3\times4]} y^{(L-1)}_{[4\times1]}$$
$$y_{[3\times1]} = \varphi(v)$$



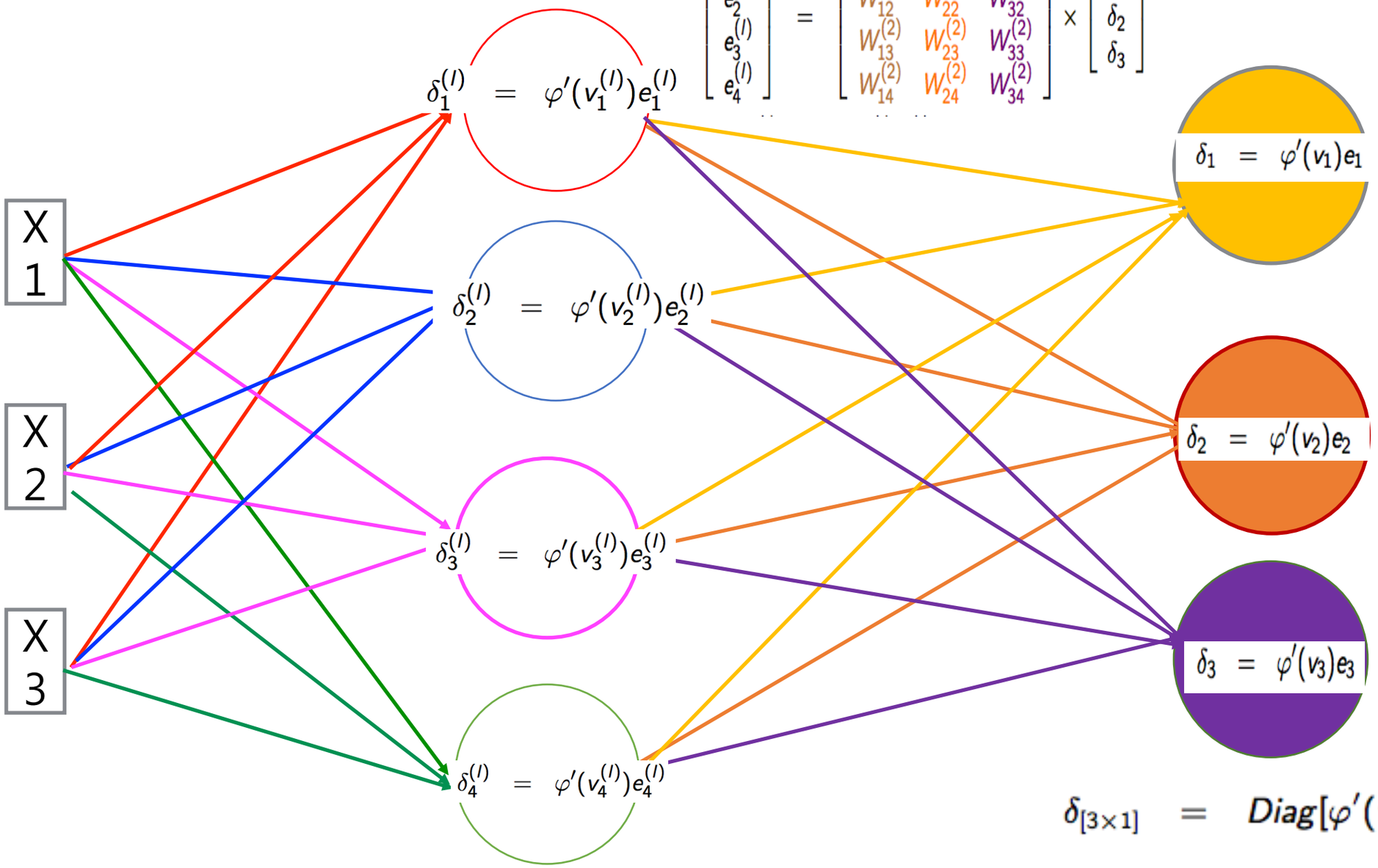
$$= \begin{bmatrix} w^{(1)}_{11} & w^{(1)}_{12} & w^{(1)}_{13} \\ w^{(1)}_{21} & w^{(1)}_{22} & w^{(1)}_{23} \\ w^{(1)}_{31} & w^{(1)}_{32} & w^{(1)}_{33} \\ w^{(1)}_{41} & w^{(1)}_{42} & w^{(1)}_{43} \end{bmatrix}$$

$$\begin{bmatrix} w^{(2)}_{11} & w^{(2)}_{12} & w^{(2)}_{13} & w^{(2)}_{14} \\ w^{(2)}_{21} & w^{(2)}_{22} & w^{(2)}_{23} & w^{(2)}_{24} \\ w^{(2)}_{31} & w^{(2)}_{32} & w^{(2)}_{33} & w^{(2)}_{34} \end{bmatrix}$$

다범주 분류 신경망: 역전파

$$e^{(l)}_{[4\times1]} = [W^{(l+1)}]^T_{[4\times3]}\delta^{(l+1)}_{[3\times1]}$$

$$\begin{bmatrix} e_1^{(l)} \\ e_2^{(l)} \\ e_3^{(l)} \\ e_4^{(l)} \end{bmatrix} = \begin{bmatrix} w_{11}^{(2)} & w_{21}^{(2)} & w_{31}^{(2)} \\ w_{12}^{(2)} & w_{22}^{(2)} & w_{32}^{(2)} \\ w_{13}^{(2)} & w_{23}^{(2)} & w_{33}^{(2)} \\ w_{14}^{(2)} & w_{24}^{(2)} & w_{34}^{(2)} \end{bmatrix} \times \begin{bmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \end{bmatrix}$$



$$e_{[3\times1]} = y_{[3\times1]} - d_{[3\times1]}$$

$$e_1 = y_1 - d_1$$

$$e_2 = y_2 - d_2$$

$$e_3 = y_3 - d_3$$

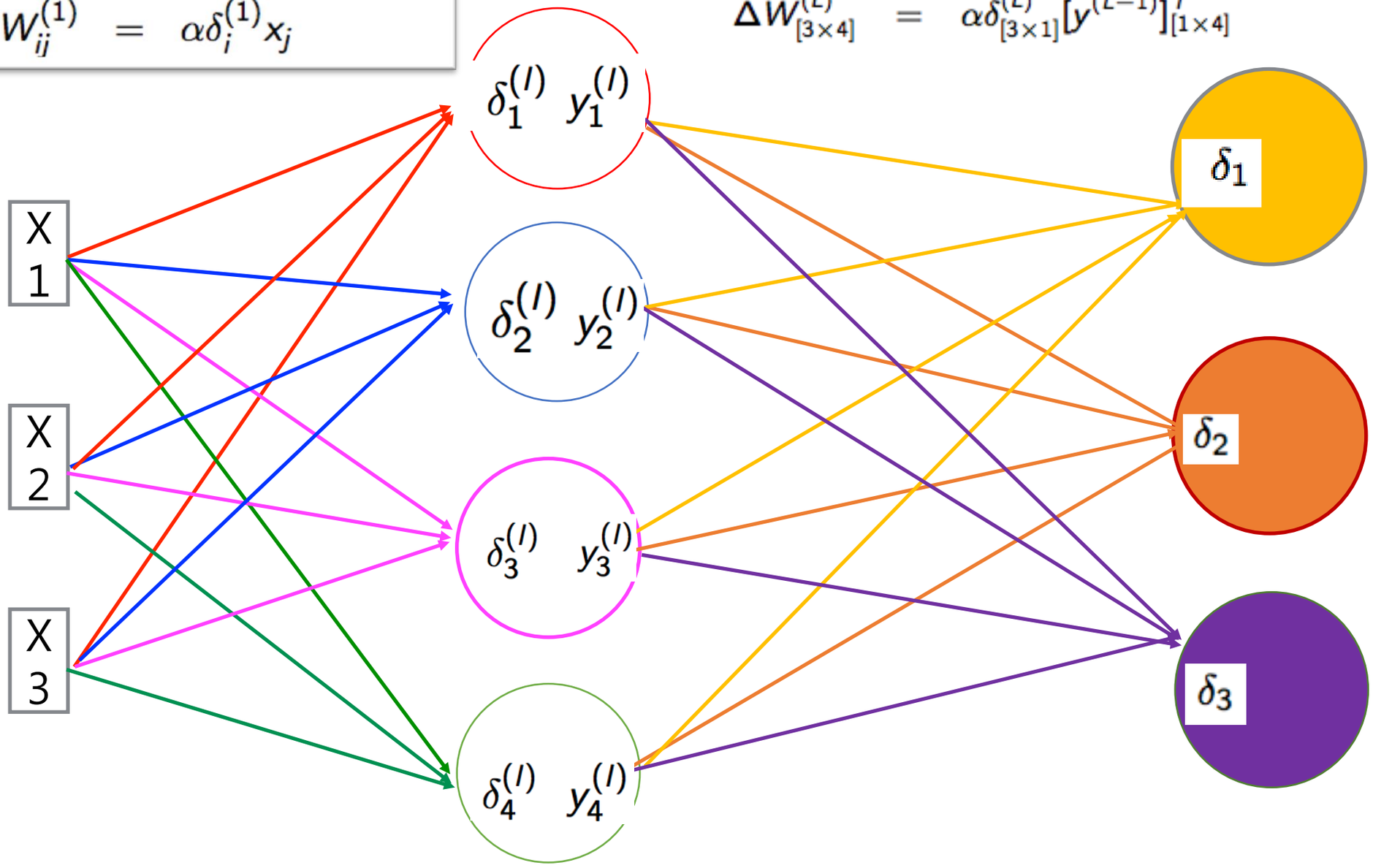
$$\delta_{[3\times1]} = Diag[\varphi'(v_{[3\times1]})]e_{[3\times1]}$$

$$\delta^{(l)}_{[4\times1]} = Diag[\varphi'(v^{(l)})_{[4\times1]}]_{[4\times4]}e^{(l)}_{[4\times1]}$$

다범주 분류 신경망: 가중치 조정

$$\Delta W^{(1)}_{[4\times3]} = \alpha\delta^{(1)}_{[4\times1]}[X]^T_{[1\times3]}$$
$$\Delta W^{(1)}_{ij} = \alpha\delta^{(1)}_{i}x_j$$

$$\Delta W^{(l)}_{i,j} = \alpha\delta^{(l)}_{i}y^{(l-1)}_{j}$$
$$\Delta W^{(L)}_{[3\times4]} = \alpha\delta^{(L)}_{[3\times1]}[y^{(L-1)}]^T_{[1\times4]}$$



다범주신경망 학습

STEP 1 가중치 초기화

$$W^{(1)}_{[4\times3]} = \begin{bmatrix} w^{(1)}_{11} & w^{(1)}_{12} & w^{(1)}_{13} \\ w^{(1)}_{21} & w^{(1)}_{22} & w^{(1)}_{23} \\ w^{(1)}_{31} & w^{(1)}_{32} & w^{(1)}_{33} \\ w^{(1)}_{41} & w^{(1)}_{42} & w^{(1)}_{43} \end{bmatrix}$$

$$W^{(2)}_{[3\times4]} = \begin{bmatrix} w^{(2)}_{11} & w^{(2)}_{12} & w^{(2)}_{13} & w^{(2)}_{14} \\ w^{(2)}_{21} & w^{(2)}_{22} & w^{(2)}_{23} & w^{(2)}_{24} \\ w^{(2)}_{31} & w^{(2)}_{32} & w^{(2)}_{33} & w^{(2)}_{34} \end{bmatrix}$$

다범주신경망 학습 (2)

STEP 2 입력데이터 → 결과

1. 입력층

$$x_{[3\times1]} = [x_1, x_2, x_3]^T$$

2. 은닉층

$$\begin{aligned} v^{(1)}_{[4\times1]} &= W^{(1)}_{[4\times3]} x_{[3\times1]} \\ y^{(1)}_{[4\times1]} &= \varphi(v^{(1)}) \\ v^{(l+1)}_{[3\times1]} &= W^{(l+1)}_{[3\times4]} y^{(l)}_{[4\times1]} \\ y^{(l+1)}_{[3\times1]} &= \varphi(v^{(l+1)}) \end{aligned}$$

3. 출력층

$$\begin{aligned} v_{[3\times1]} &= W^{(L)}_{[3\times4]} y^{(L-1)}_{[4\times1]} \\ y_{[3\times1]} &= \varphi(v_{[3\times1]}) \end{aligned}$$

다범주신경망 학습 (3)

STEP3 오차, δ

1. 출력층

$$
\begin{aligned}
e_{[3\times1]} &= y_{[3\times1]} - d_{[3\times1]} \\
\delta_{[3\times1]} &= Diag[\varphi'(v_{[3\times1]})]e_{[3\times1]}
\end{aligned}
$$

2. 은닉층

$$
\begin{aligned}
e^{(l)}_{[4\times1]} &= [W^{(l+1)}]^T_{[4\times3]}\delta^{(l+1)}_{[3\times1]} \\
\delta^{(l)}_{[4\times1]} &= Diag[\varphi'(v^{(l)})_{[4\times1]}]_{[4\times4]}e^{(l)}_{[4\times1]}
\end{aligned}
$$

STEP4 가중치조정

$$
\begin{aligned}
\Delta W^{(l)}_{ij} &= \alpha\delta^{(l)}_i y^{(l-1)}_j \\
\Delta W^{(L)}_{[3\times4]} &= \alpha\delta^{(L)}_{[3\times1]}[y^{(L-1)}]^T_{[1\times4]} \\
\Delta W^{(l)}_{[4\times3]} &= \alpha\delta^{(l)}_{[4\times1]}[X]^T_{[1\times3]} \\
W'^{(l)} &= W^{(l)} + \Delta W^{(l)}
\end{aligned}
$$

Softmax function

$$
\begin{aligned}
\varphi &: R^n \to R^n \\
\varphi(v_i) &= \frac{exp(v_i)}{\sum_{k=1}^{n} exp(v_k)}
\end{aligned}
$$

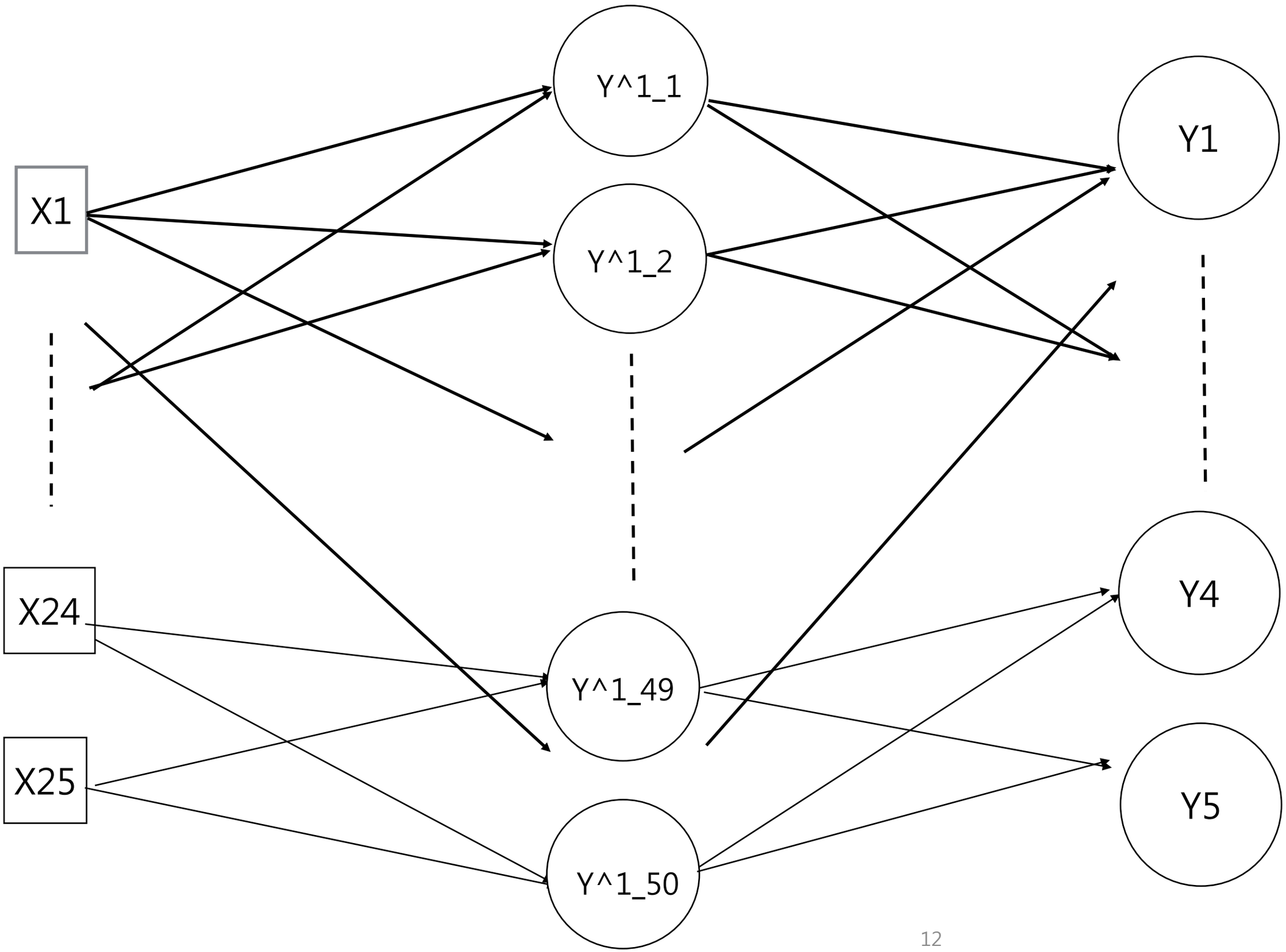
$$
\begin{aligned}
\frac{\partial \varphi(v_i)}{\partial v_i} &= \frac{exp(v_i)\sum_{k=1}^{n} exp(v_k) - exp(v_i)exp(v_i)}{(\sum_{k=1}^{n} exp(v_k))^2} \\
&= \frac{exp(v_i)}{\sum_{k=1}^{n} exp(v_k)} \cdot \frac{\sum_{k=1}^{n} exp(v_k) - exp(v_i)}{\sum_{k=1}^{n} exp(v_k)} \\
&= \frac{exp(v_i)}{\sum_{k=1}^{n} exp(v_k)} \cdot \left[1 - \frac{exp(v_i)}{\sum_{k=1}^{n} exp(v_k)}\right] \\
&= \varphi(v_i)(1 - \varphi(v_i))
\end{aligned}
$$

1계 도함수는 Sigmoid 함수와 동일

Softmax.m

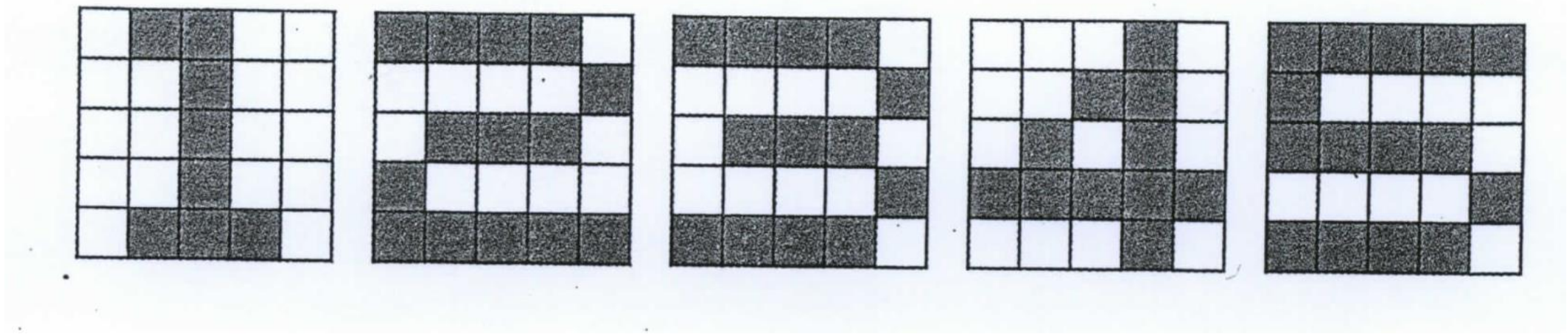
```
function y = Softmax(x)

ex=exp(x);
y=(ex)/sum(ex);
end
```

예제: 글자맞추기
입력 25 ->은닉 50 ->출력5

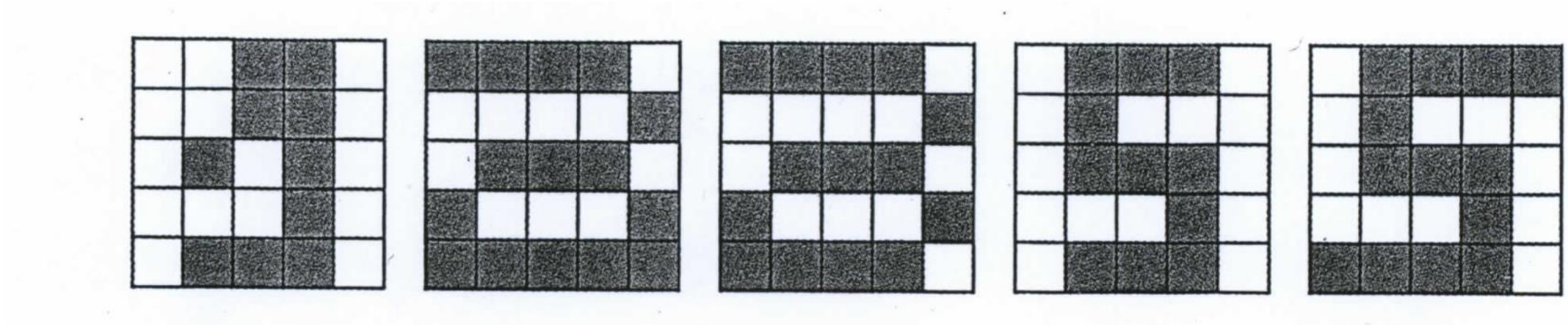


학습데이터, 검증데이터

학습데이터



검증데이터



training function: Multiclass.m

function [W1, W2] = Multiclass(W1,W2,X,D)

1. Learning Rate

alpha=0.9;

2. Loop Setup.: SGD

N=5;

for k=1:N

3. Input-Output

x=reshape(X(:,:,k),25,1); %25*1

d=D(k,:)'; %5*1

v1=W1*x; %(50*1=50*25 x 25*1

y1=Sigmoid(v1);

v=W2*y1; %(5*1=5*50 x 50*1

y=Softmax(v);

4. Back Propagation

e=d-y;

delta=e; %(5*1)

e1=W2'*delta; %(50*5)*(5*1)

delta1=y1.*(1-y1).*e1; %(50*1)

5. Update

dW1=alpha*delta1*x'; %(50*1)*(1*25)

W1=W1+dW1;

dW2=alpha*delta*y1'; %(5*1)*(1*50)

W2=W2+dW2;

End % end of SGD loop

end % end of function

TestMulticlass .m

```
clear all
rng(3);
1. Data loading
X=zeros(5,5,5);

X(:,:,1)=[0 1 1 0 0;
          0 0 1 0 0;
          0 0 1 0 0;
          0 0 1 0 0;
          0 1 1 1 0
          ];

X(:,:,2)=[1 1 1 1 0;
          0 0 0 0 1;
          0 1 1 1 0;
          1 0 0 0 0;
          1 1 1 1 1
          ];
X(:,:,3)=[1 1 1 1 0;
          0 0 0 0 1;
          0 1 1 1 0;
          0 0 0 0 1;
          1 1 1 1 0
          ];

X(:,:,4)=[0 0 0 1 0;
          0 0 1 1 0;
          0 1 0 1 0;
          1 1 1 1 1;
          0 0 0 1 0
          ];
X(:,:,5)=[1 1 1 1 1;
          1 0 0 0 0;
          1 1 1 1 0;
          0 0 0 0 1;
          1 1 1 1 0
          ];
D=[1 0 0 0 0;
   0 1 0 0 0;
   0 0 1 0 0;
   0 0 0 1 0;
   0 0 0 0 1
   ];

2. 가중치 초기화
W1=2*rand(50,25)-1;
W2=2*rand(5,50)-1;
W10=W1;
W20=W2;

3. 기계 학습
for epoch = 1:1000
    [W1,
W2]=Multiclass(W1,W2,X,D);
end

4. 추정
N=5;
for k=1:N
    x=reshape(X(:,:,k),25,1);
    v1=W1*x;
    y1=Sigmoid(v1);
    v=W2*y1;
    y=Softmax(v)
end
```

RealMulticlass .m : 입력데이터

```
clear all
rng(3);
X=zeros(5,5,5);

1. 기계학습
TestMultiClass;

2. Data loading
X=zeros(5,5,5);

X(:,:,1)=[0 0 1 1 0;
          0 0 1 1 0;
          0 1 0 1 0;
          0 0 0 1 0;
          0 1 1 1 0
          ];

X(:,:,2)=[1 1 1 1 0;
          0 0 0 0 1;
          0 1 1 1 0;
          1 0 0 0 1;
          1 1 1 1 1
          ];

X(:,:,3)=[1 1 1 1 0;
          0 0 0 0 1;
          0 1 1 1 0;
          1 0 0 0 1;
          1 1 1 1 0
          ];

X(:,:,4)=[0 1 1 1 0;
          0 1 0 0 0;
          0 1 1 1 0;
          0 0 0 1 0;
          0 1 1 1 0
          ];

X(:,:,5)=[0 1 1 1 1;
          0 1 0 0 0;
          0 1 1 1 0;
          0 0 0 1 0;
          1 1 1 1 0
          ];

3. 추정
N=5;
for k=1:N
    x=reshape(X(:,:,k),25,1);
    v1=W1*x;
    y1=Sigmoid(v1);
    v=W2*y1;
    y=Softmax(v)
end
```

실습 . Matlab => R

- Softmax.m
- Multiclass.m
- TestMulticlass.m
- RealMulticlass.m

- .m 에서 과업 특성 파악 => 과업 list 작성 => R code

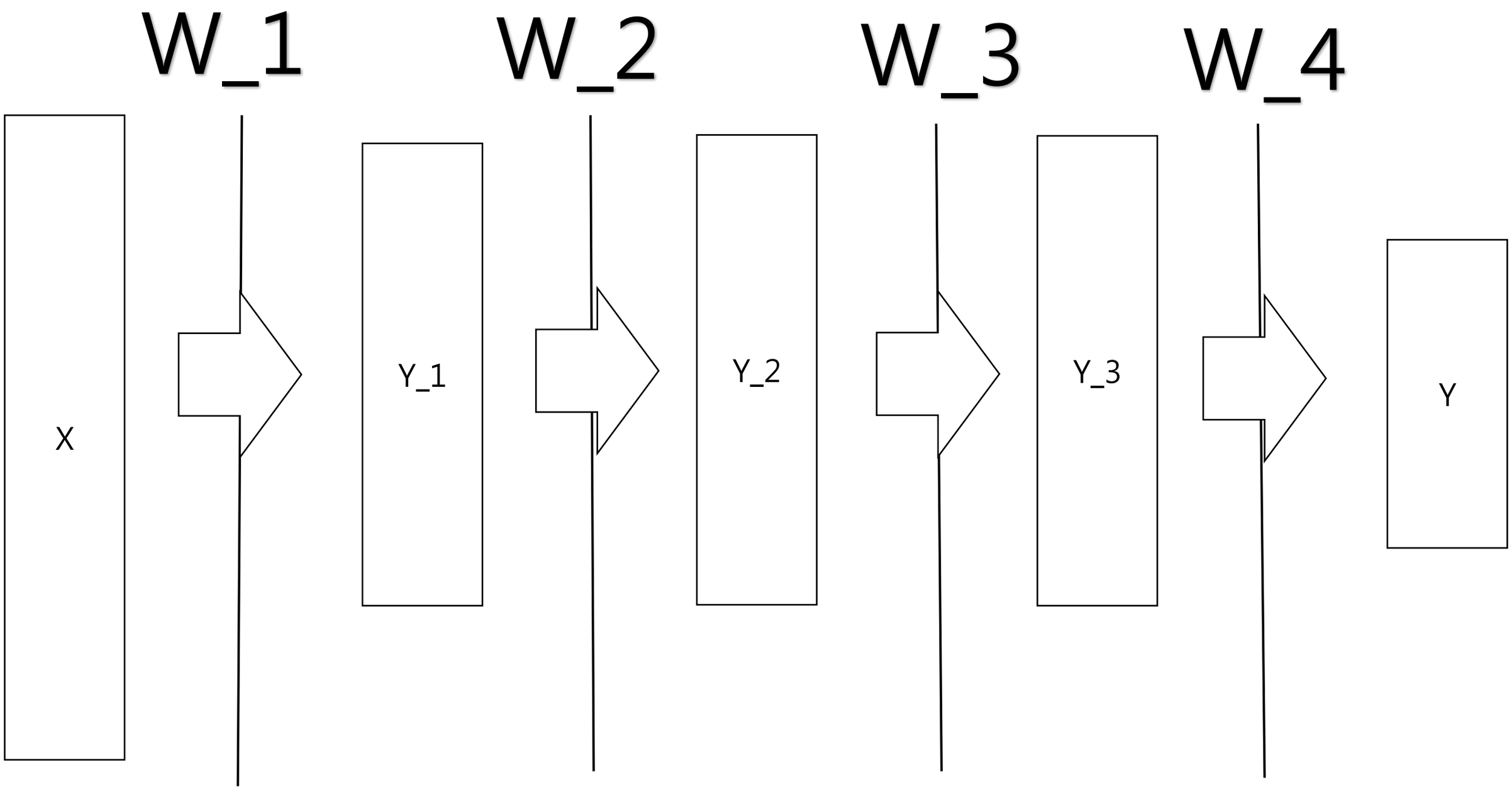
딥러닝 첫걸음

5. 딥러닝(심층신경망)

심층신경망: 은닉층 2개 이상 다층신경망

- 심층신경망의 문제점: 그래디언트 소실, 과적합, 계산부담
  - 그래디언트 소실 : 입력층에 가까운 은닉층에 출력층의 오차 정보 소실
    - ReLU 함수를 활성화 함수로 활용
  - 과적합
    - Dropout으로 해소: 무작위로 일부 node를 제외
  - 계산부담: 하드웨어의 발달 + 병렬처리로 해소
- 예제: 글자맞추기
  - 입력층 (25) → 제1은닉층(20) → 제2은닉층(20) → 제3은닉층(20) → 출력층(5)

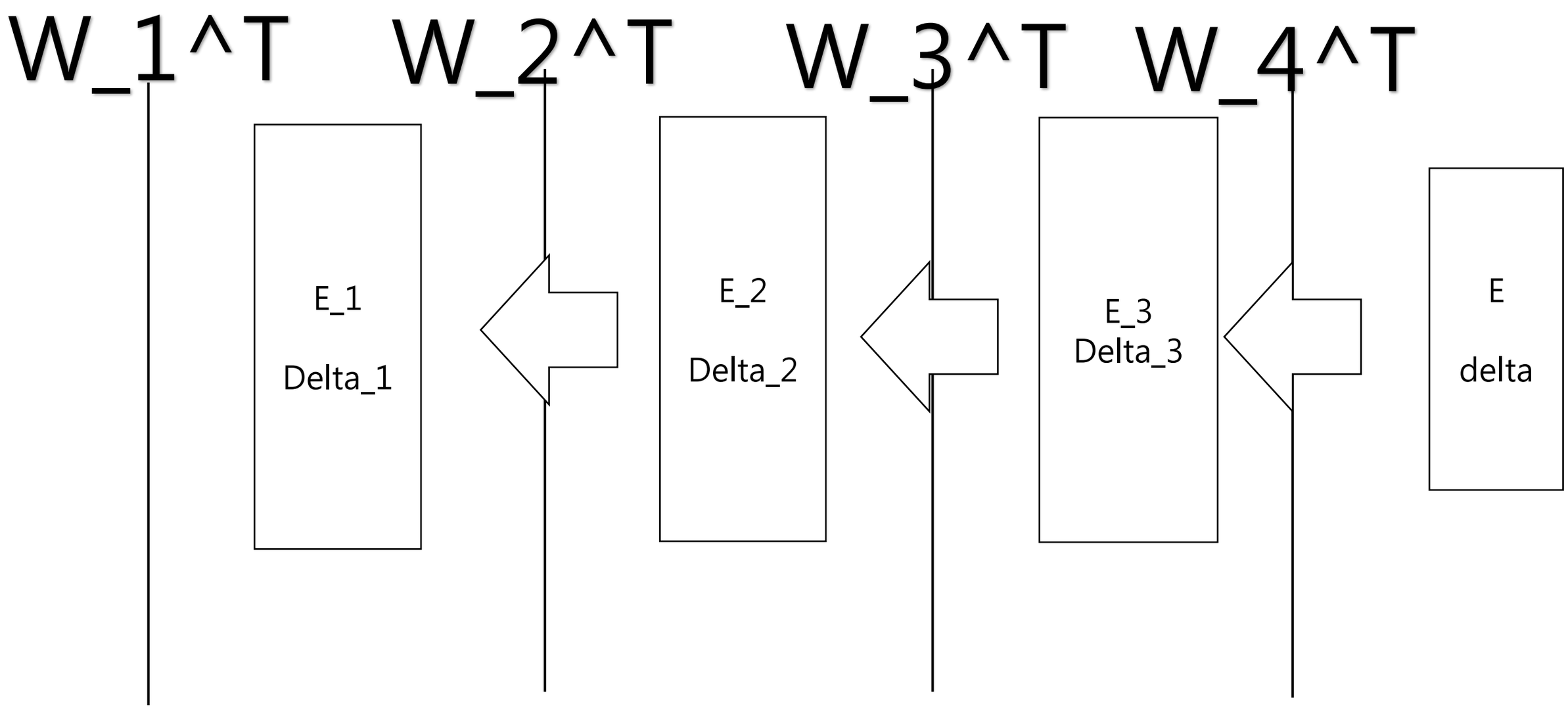
예제: 글자맞추기 입출력



$$
\begin{aligned}
v^{(1)} &= W^{(1)}X \\
y^{(1)} &= \varphi(v^{(1)}) \\
v^{(l+1)} &= W^{(l+1)}y^{(l)} \\
y^{(l+1)} &= \varphi(v^{(l+1)})
\end{aligned}
$$

$$
\begin{aligned}
v &= W^{(L)}y^{(L-1)} \\
y &= \varphi(v)
\end{aligned}
$$

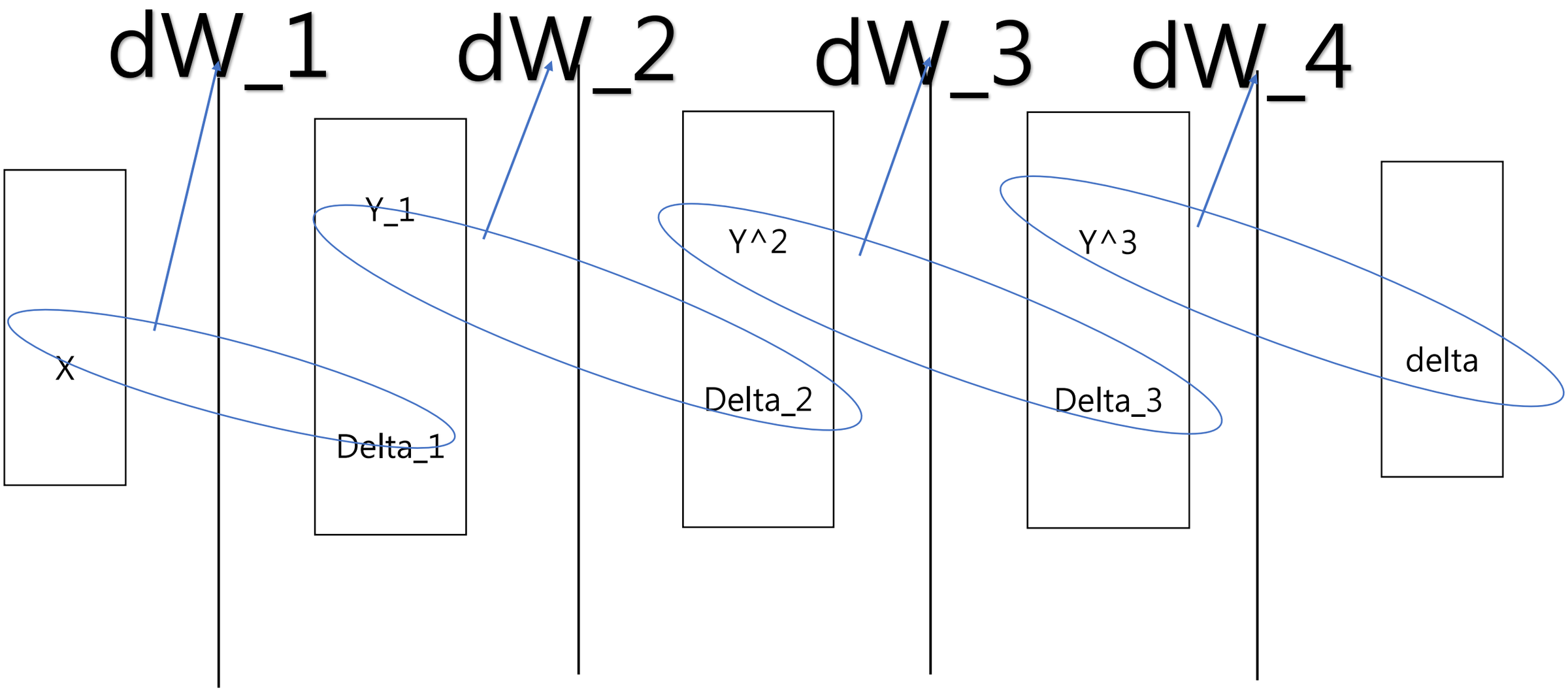
예제: 글자맞추기-BackProp



$$e^{(l)} = [W^{(l+1)}]^T \delta^{(l+1)} \qquad e = y - d$$

$$\delta^{(l)} = Diag[\varphi'(v^{(l)})]e^{(l)} \qquad \delta = Diag[\varphi'(v)]e$$

예제: 글자맞추기-가중치조정



$$\Delta w_{i,j}^{(l)} = \alpha \delta_i^{(l)} y_j^{(l-1)}$$

$$\Delta W^{(l)} = \alpha \delta^{(l)} y^{(l-1)^T}$$

$$W'^{(l)} = W^{(l)} + \Delta W^{(l)}$$

심층신경망 학습

STEP 1 가중치 초기화

$$W^{(1)}_{[20\times25]}$$

$$W^{(2)}_{[20\times20]}$$

$$W^{(3)}_{[20\times20]}$$

$$W^{(4)}_{[5\times20]}$$

심층신경망 학습 (2)

STEP 2 입력데이터 → 결과

1. 입력층

$$X_{[25\times 1]}$$

2. 은닉층

$$\begin{aligned} v^{(1)}_{[20\times 1]} &= W^{(1)}_{[20\times 25]} X_{[25\times 1]} \\ y^{(1)}_{[20\times 1]} &= \varphi(v^{(1)}) \\ v^{(2)}_{[20\times 1]} &= W^{(2)}_{[20\times 20]} y^{(1)}_{[20\times 1]} \\ y^{(2)}_{[20\times 1]} &= \varphi(v^{(2)}) \\ v^{(3)}_{[20\times 1]} &= W^{(3)}_{[20\times 20]} y^{(2)}_{[20\times 1]} \\ y^{(3)}_{[20\times 1]} &= \varphi(v^{(3)}) \end{aligned}$$

3. 출력층

$$\begin{aligned} v_{[5\times 1]} &= W^{(4)}_{[5\times 20]} y^{(3)}_{[20\times 1]} \\ y_{[5\times 1]} &= \varphi(v_{[5\times 1]}) \end{aligned}$$

심층신경망 학습 (3)

STEP3 오차, δ

1. 출력층

$$
\begin{aligned}
e_{[5\times1]} &= y_{[5\times1]} - d_{[5\times1]} \\
\delta_{[5\times1]} &= Diag[\varphi'(v_{[5\times1]})]e_{[5\times1]}
\end{aligned}
$$

2. 은닉층

$$
\begin{aligned}
e^{(3)}_{[20\times1]} &= [W^{(4)}]^T_{[20\times5]}\delta_{[5\times1]} \\
\delta^{(3)}_{[20\times1]} &= Diag[\varphi'(v^{(3)})_{[20\times1]}]e^{(3)}_{[20\times1]} \\
e^{(2)}_{[20\times1]} &= [W^{(3)}]^T_{[20\times20]}\delta^{(3)}_{[20\times1]} \\
\delta^{(2)}_{[20\times1]} &= Diag[\varphi'(v^{(2)})_{[20\times1]}]e^{(2)}_{[20\times1]} \\
e^{(1)}_{[25\times1]} &= [W^{(2)}]^T_{[25\times20]}\delta^{(2)}_{[20\times1]} \\
\delta^{(1)}_{[25\times1]} &= Diag[\varphi'(v^{(1)})_{[25\times1]}]e^{(1)}_{[25\times1]}
\end{aligned}
$$

심층신경망 학습 (4)

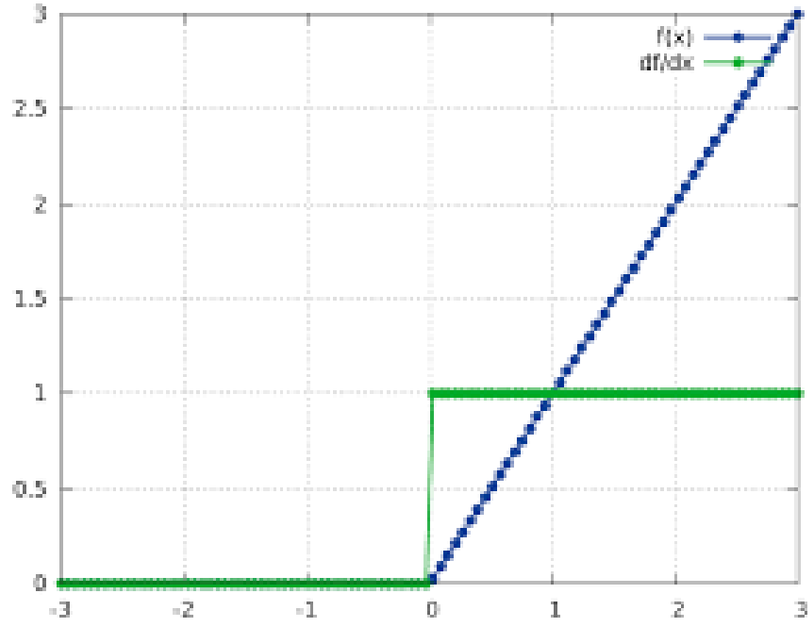
STEP4 가중치조정

$$
\begin{aligned}
\Delta W_{i,j}^{(l)} &= \alpha \delta_i^{(l)} y_j^{(l-1)} \\
\Delta W_{[5\times 20]}^{(4)} &= \alpha \delta_{[5\times 1]} [y^{(3)}]_{[1\times 20]}^T \\
\Delta W_{[20\times 20]}^{(3)} &= \alpha \delta_{[20\times 1]}^{(3)} [y^{(2)}]_{[1\times 20]}^T \\
\Delta W_{[20\times 20]}^{(2)} &= \alpha \delta_{[20\times 1]}^{(2)} [y^{(1)}]_{[1\times 20]}^T \\
\Delta W_{[20\times 25]}^{(1)} &= \alpha \delta_{[20\times 1]}^{(1)} [X]_{[1\times 25]}^T \\
W'^{(l)} &= W^{(l)} + \Delta W^{(l)}
\end{aligned}
$$

ReLU function

$$\varphi(v) = \max\{0, v\} = \begin{cases} v & v > 0 \\ 0 & v \leq 0 \end{cases}$$

$$\varphi'(v) = \begin{cases} 1 & v > 0 \\ 0 & v \leq 0 \end{cases}$$

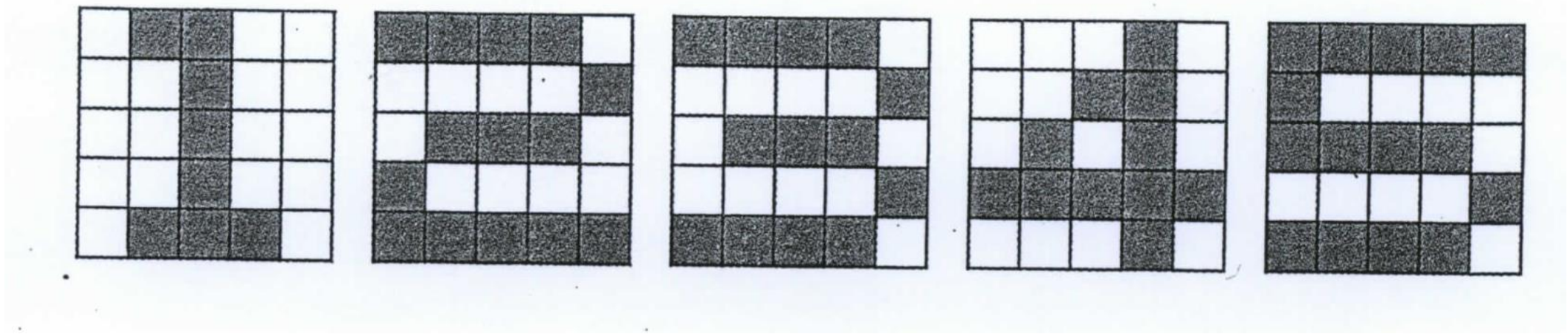


ReLU.m

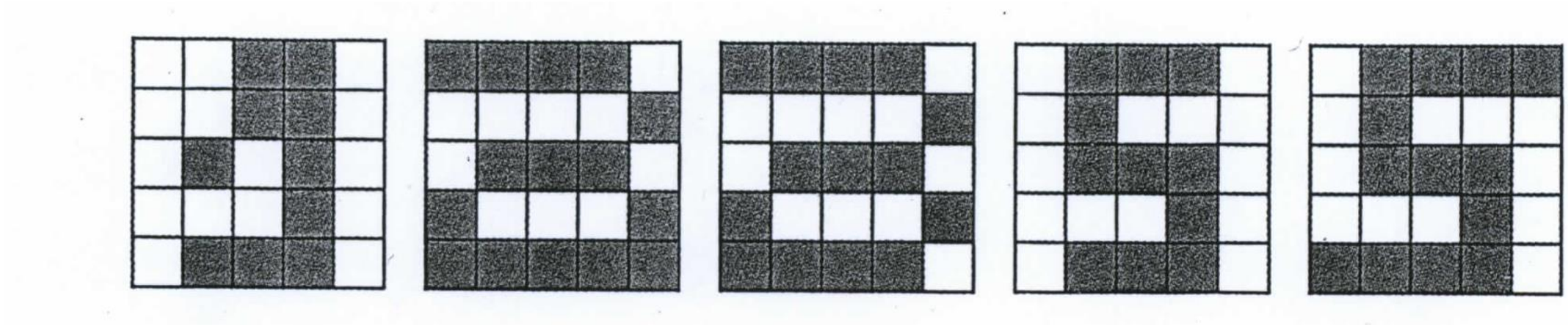
```
function y=ReLU(x)

y=max(0,x);

end
```

학습데이터, 검증데이터

학습데이터



검증데이터



training function: DeepReLu.m

function[W1,W2,W3,W4]=DeepReLU(W1,W2,W3,W4,X,D)

1. Learning Rate

alpha=0.01;

2. SGD

N=5;
for k=1:N
x=reshape(X(:,:,k),25,1);
v1=W1*x;
y1=ReLU(v1);
v2=W2*y1;
y2=ReLU(v2);
v3=W3*y2;
y3=ReLU(v3);
v=W4*y3;
y=softmax(v);

4. Back Propagation

d=D(k,:)';
e=d-y;
delta=e;
e3=W4'*delta;
delta3=(v3>0).*e3;
e2=W3'*delta3;
delta2=(v2>0).*e2;
e1=W2'*delta2;
delta1=(v1>0).*e1;

5. Update

dW4=alpha*delta*y3';
W4=W4+dW4;
dW3=alpha*delta3*y2';
W3=W3+dW3;
dW2=alpha*delta2*y1';
W2=W2+dW2;
dW1=alpha*delta1*x';
W1=W1+dW1;

End % end of SGD loop

end % end of function

TestDeepRelu .m

```
clear all

1. Data loading
X=zeros(5,5,5);

X(:,:,1)=[0 1 1 0 0;
          0 0 1 0 0;
          0 0 1 0 0;
          0 0 1 0 0;
          0 1 1 1 0
          ];

X(:,:,2)=[1 1 1 1 0;
          0 0 0 0 1;
          0 1 1 1 0;
          1 0 0 0 0;
          1 1 1 1 1
          ];
X(:,:,3)=[1 1 1 1 0;
          0 0 0 0 1;
          0 1 1 1 0;
          0 0 0 0 1;
          1 1 1 1 0
          ];
X(:,:,4)=[0 0 0 1 0;
          0 0 1 1 0;
          0 1 0 1 0;
          1 1 1 1 1;
          0 0 0 1 0
          ];
X(:,:,5)=[1 1 1 1 1;
          1 0 0 0 0;
          1 1 1 1 0;
          0 0 0 0 1;
          1 1 1 1 0
          ];
D=[1 0 0 0 0;
   0 1 0 0 0;
   0 0 1 0 0;
   0 0 0 1 0;
   0 0 0 0 1
   ];
```

2. 가중치 초기화

```
W1=2*rand(20,25)-1;
W2=2*rand(20,20)-1;
W3=2*rand(20,20)-1;
W4=2*rand(5,20)-1;
```

3. 기계 학습

```
for epoch = 1:1000
    [W1, W2,W3,W4]
=DeepReLU(W1,W2,W3,W4,X,D);
end
```

4. 추정

```
N=5;
for k=1:N
    x=reshape(X(:,:,k),25,1);
 v1=W1*x;
 y1=ReLU(v1);
 v2=W2*y1;
 y2=ReLU(v2);
 v3=W3*y2;
 y3=ReLU(v3);
 v=W4*y3;
 y=softmax(v)
    end
```

RealRelu.m : 입력데이터 실험

```
clear all
rng(3);
X=zeros(5,5,5);

1. 기계학습
TestDeepReLU;

2. Data loading
X=zeros(5,5,5);

X(:,:,1)=[0 0 1 1 0;
          0 0 1 1 0;
          0 1 0 1 0;
          0 0 0 1 0;
          0 1 1 1 0
          ];

X(:,:,2)=[1 1 1 1 0;
          0 0 0 0 1;
          0 1 1 1 0;
          1 0 0 0 1;
          1 1 1 1 1
          ];

X(:,:,3)=[1 1 1 1 0;
          0 0 0 0 1;
          0 1 1 1 0;
          1 0 0 0 1;
          1 1 1 1 0
          ];

X(:,:,4)=[0 1 1 1 0;
          0 1 0 0 0;
          0 1 1 1 0;
          0 0 0 1 0;
          0 1 1 1 0
          ];

X(:,:,5)=[0 1 1 1 1;
          0 1 0 0 0;
          0 1 1 1 0;
          0 0 0 1 0;
          1 1 1 1 0
          ];

3. 추정
N=5;
for k=1:N
    x=reshape(X(:,:,k),25,1);
    v1=W1*x;
    y1=ReLU(v1);

    v2=W2*y1;
    y2=ReLU(v2);

    v3=W3*y2;
    y3=ReLU(v3);

    v=W4*y3;
    y=softmax(v)
end
```

실습 . Matlab => R

- Relu.m
- DeepReLu.m
- TestdeepReLu.m

- .m 에서 과업 특성 파악 => 과업 list 작성 => R code

Dropout: 일부 node를 임의로 제외

- 과적합 문제: 학습데이터에서 지나치게 많은 정보가 들어간다
- Dropout: 학습데이터에서 무작위로 일부 정보를 제외한다.
- Dropout 과정
  1. Random sampling으로 제외할 node를 고른다 (Dropout.m)
  2. 1.에서 제외한 node를 빼고 입력-> 출력 과정을 거친다. (y)
  3. 2.에서 얻은 y를 이용하여 역전파 과정을 거친다. (delta, e)
  4. 2. 3 에서 얻은 y와 delta를 이용하여 가중치를 조정한다.
  5. 1.-4.를 반복한다.
     - (2-5: DeepDropout.m)
- (실습하지 않습니다.)

Dropout.m

```
function ym=Dropout(y1,ratio)
[m,n]=size(y1);
ym=zeros(m,n);

num=round(m*n*(1-ratio));
idx=randperm(m*n,num);
ym(idx)=1/(1-ratio);


end
```

training function: DeepDropout.m

```
function[W1,W2,W3,W4]=DeepDropout(W1,W2
,W3,W4,X,D)

1. Learning Rate
alpha=0.01;
2. SGD
N=5;
for k=1:N
v1=W1*x;
   y1=Sigmoid(v1);
   y1=y1.*Dropout(y1,0.2);

   v2=W2*y1;
   y2=Sigmoid(v2);
   y2=y2.*Dropout(y2,0.2);

   v3=W3*y2;
   y3=Sigmoid(v3);
   y3=y3.*Dropout(y3,0.2);

   v=W4*y3;
   y=softmax(v);
```

```
4. Back Propagation
   d=D(k,:)';
   e=d-y;
   delta=e;
   e3=W4'*delta;
   delta3=y3.*(1-y3).*e3;
   e2=W3'*delta3;
   delta2=y2.*(1-y2).*e2;
   e1=W2'*delta2;
   delta1=y1.*(1-y1).*e1;
5. Update
    dW4=alpha*delta*y3';
   W4=W4+dW4;
   dW3=alpha*delta3*y2';
   W3=W3+dW3;
   dW2=alpha*delta2*y1';
   W2=W2+dW2;
   dW1=alpha*delta1*x';
   W1=W1+dW1;
end % end of SGD loop
end  % end of function
```

TestDeepDropout .m

```
clear all

1. Data loading
X=zeros(5,5,5);

X(:,:,1)=[0 1 1 0 0;
          0 0 1 0 0;
          0 0 1 0 0;
          0 0 1 0 0;
          0 1 1 1 0
          ];

X(:,:,2)=[1 1 1 1 0;
          0 0 0 0 1;
          0 1 1 1 0;
          1 0 0 0 0;
          1 1 1 1 1
          ];
X(:,:,3)=[1 1 1 1 0;
          0 0 0 0 1;
          0 1 1 1 0;
          0 0 0 0 1;
          1 1 1 1 0
          ];
X(:,:,4)=[0 0 0 1 0;
          0 0 1 1 0;
          0 1 0 1 0;
          1 1 1 1 1;
          0 0 0 1 0
          ];
X(:,:,5)=[1 1 1 1 1;
          1 0 0 0 0;
          1 1 1 1 0;
          0 0 0 0 1;
          1 1 1 1 0
          ];
D=[1 0 0 0 0;
   0 1 0 0 0;
   0 0 1 0 0;
   0 0 0 1 0;
   0 0 0 0 1
   ];

2. 가중치 초기화
W1=2*rand(20,25)-1;
W2=2*rand(20,20)-1;
W3=2*rand(20,20)-1;
W4=2*rand(5,20)-1;

3. 기계 학습
for epoch = 1:1000
    [W1, W2,W3,W4]
=DeepDropout(W1,W2,W3,W4,X,D);
end

4. 추정
N=5;
for k=1:N
    x=reshape(X(:,:,k),25,1);
 v1=W1*x;
 y1=Sigmoid(v1);
 v2=W2*y1;
 y2=Sigmoid(v2);
 v3=W3*y2;
 y3=Sigmoid(v3);
 v=W4*y3;
 y=softmax(v)
    end
```

Realdropout.m :입력데이터 실험

```
clear all
rng(3);
X=zeros(5,5,5);

1. 기계학습
TestDeepDropout;

2. Data loading
X=zeros(5,5,5);

X(:,:,1)=[0 0 1 1 0;
          0 0 1 1 0;
          0 1 0 1 0;
          0 0 0 1 0;
          0 1 1 1 0
          ];

X(:,:,2)=[1 1 1 1 0;
          0 0 0 0 1;
          0 1 1 1 0;
          1 0 0 0 1;
          1 1 1 1 1
          ];

X(:,:,3)=[1 1 1 1 0;
          0 0 0 0 1;
          0 1 1 1 0;
          1 0 0 0 1;
          1 1 1 1 0
          ];

X(:,:,4)=[0 1 1 1 0;
          0 1 0 0 0;
          0 1 1 1 0;
          0 0 0 1 0;
          0 1 1 1 0
          ];

X(:,:,5)=[0 1 1 1 1;
          0 1 0 0 0;
          0 1 1 1 0;
          0 0 0 1 0;
          1 1 1 1 0
          ];

3. 추정
N=5;
for k=1:N
    x=reshape(X(:,:,k),25,1);

    v1=W1*x;
    y1=Sigmoid(v1);

    v2=W2*y1;
    y2=Sigmoid(v2);

    v3=W3*y2;
    y3=Sigmoid(v3);

    v=W4*y3;
    y=softmax(v)
end
```