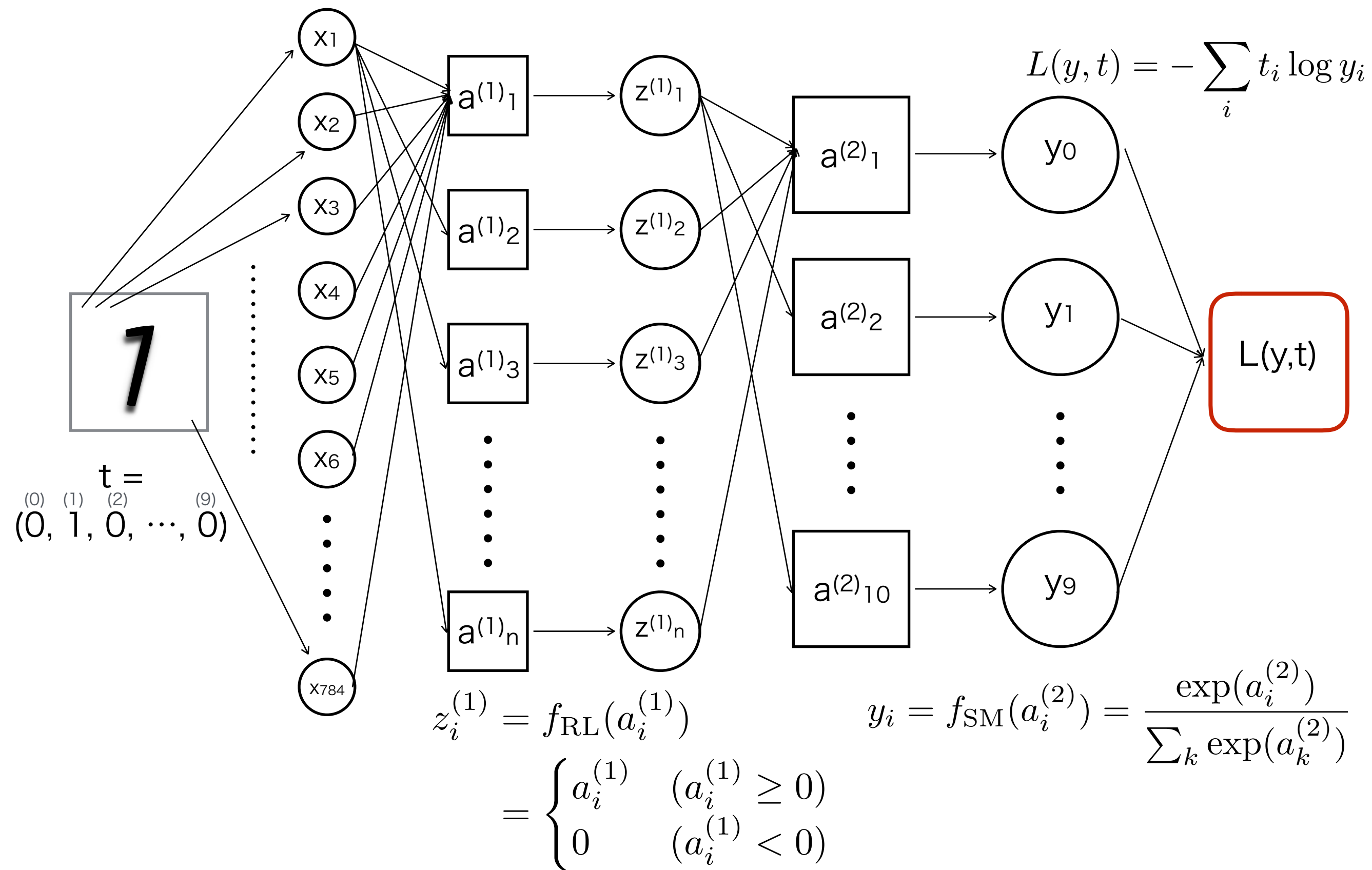


ディープな学習会

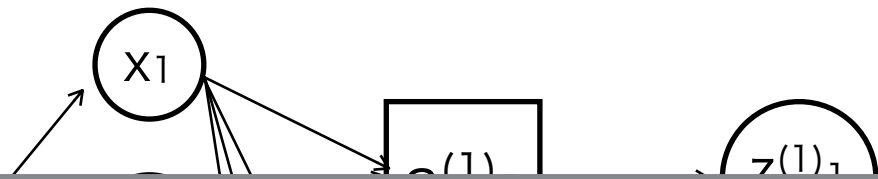
第5章 誤差逆伝播法

～後半～

$$\mathbf{a}^{(1)} = \mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)} \quad \mathbf{a}^{(2)} = \mathbf{W}^{(2)}\mathbf{z}^{(1)} + \mathbf{b}^{(2)}$$



$$\mathbf{a}^{(1)} = \mathbf{W}^{(1)} \mathbf{x} + \mathbf{b}^{(1)} \quad \mathbf{a}^{(2)} = \mathbf{W}^{(2)} \mathbf{z}^{(1)} + \mathbf{b}^{(2)}$$



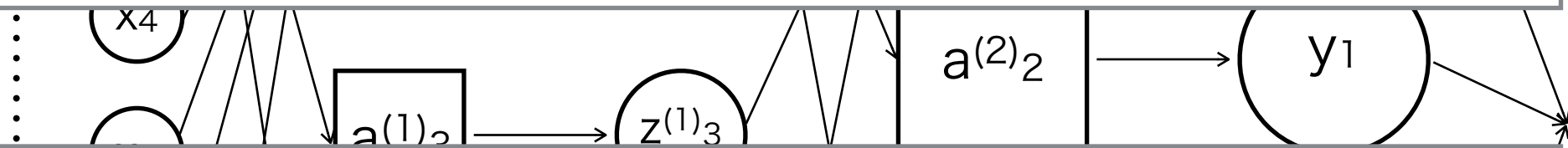
$$L(y, t) = - \sum_i t_i \log y_i$$

これからやること：パラメータの更新

(例)

$$W_{ij}^{(1)} \leftarrow W_{ij}^{(1)} + \eta \frac{\partial L(\mathbf{y}; \mathbf{t})}{\partial W_{ij}^{(1)}}$$

1



求めたいもの：

$$\frac{\partial L(y; t)}{\partial W_{ij}^{(1)}}, \quad \frac{\partial L(y; t)}{\partial b_i^{(1)}}, \quad \frac{\partial L(y; t)}{\partial W_{ij}^{(2)}}, \quad \frac{\partial L(y; t)}{\partial b_i^{(2)}}$$

$$\dots 784 \times n + n + n \times 10 + 10 = 795n + 10$$

隠れ層数 $n = 50$ として，39760個のパラメータ

→ 数値微分すると，79520回 predict を呼び出す！



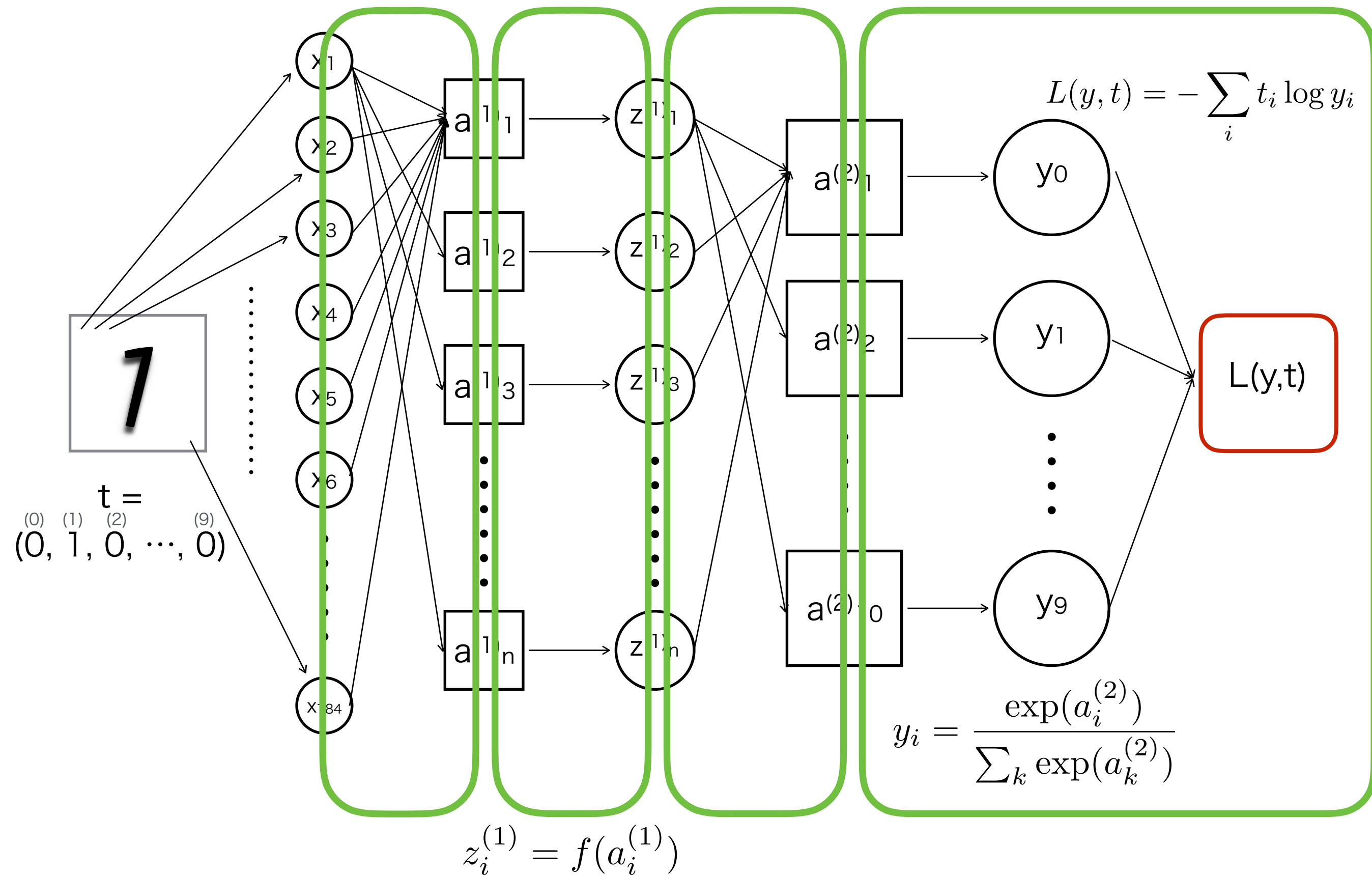
求められるなら予め解析的に求めない？？？

$$z_i = J(u_i)$$

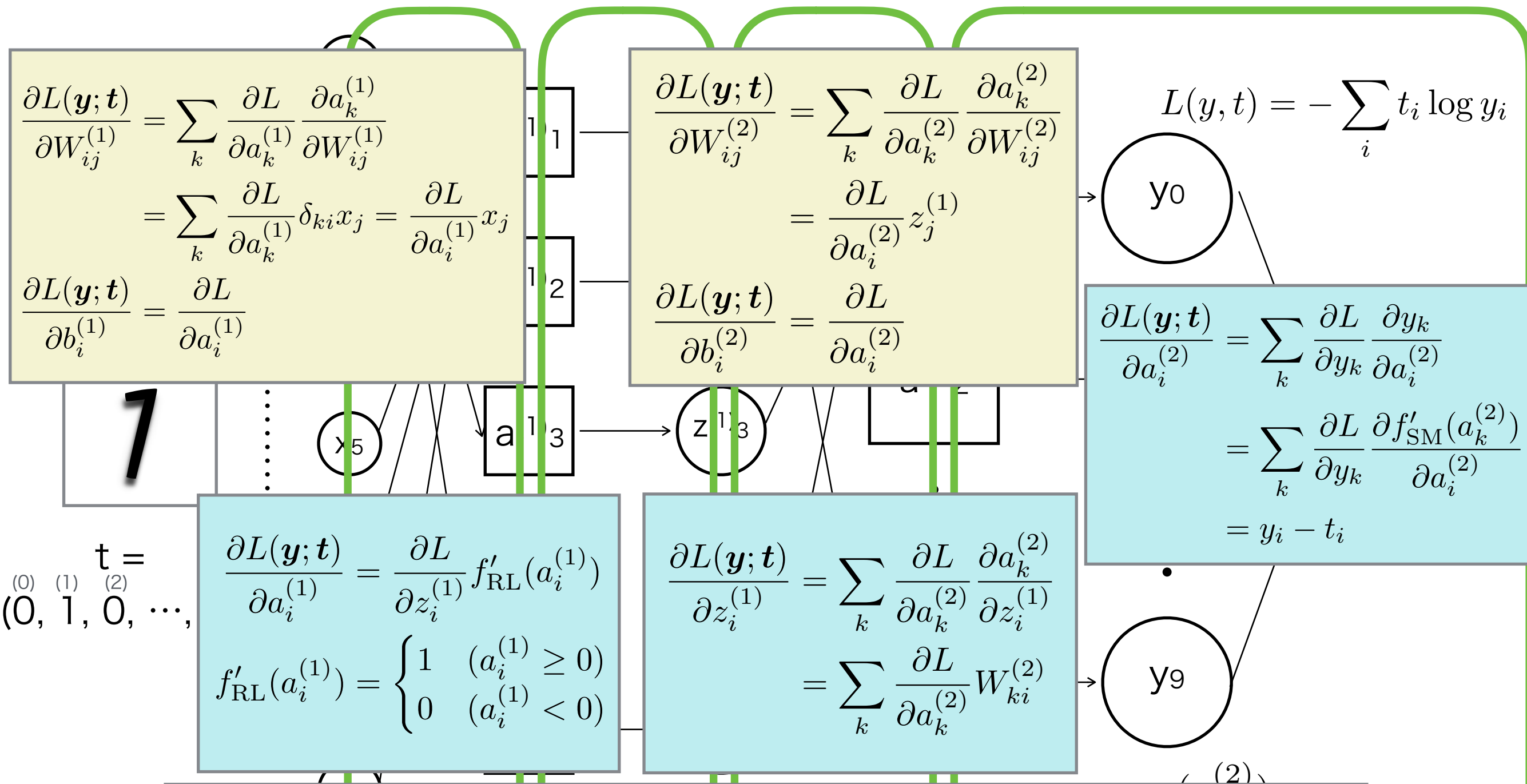
$L(y, t)$

\mathbf{t}
 $(0, 1, 0)$

$$\mathbf{a}^{(1)} = \mathbf{W}^{(1)} \mathbf{x} + \mathbf{b}^{(1)} \quad \mathbf{a}^{(2)} = \mathbf{W}^{(2)} \mathbf{z}^{(1)} + \mathbf{b}^{(2)}$$

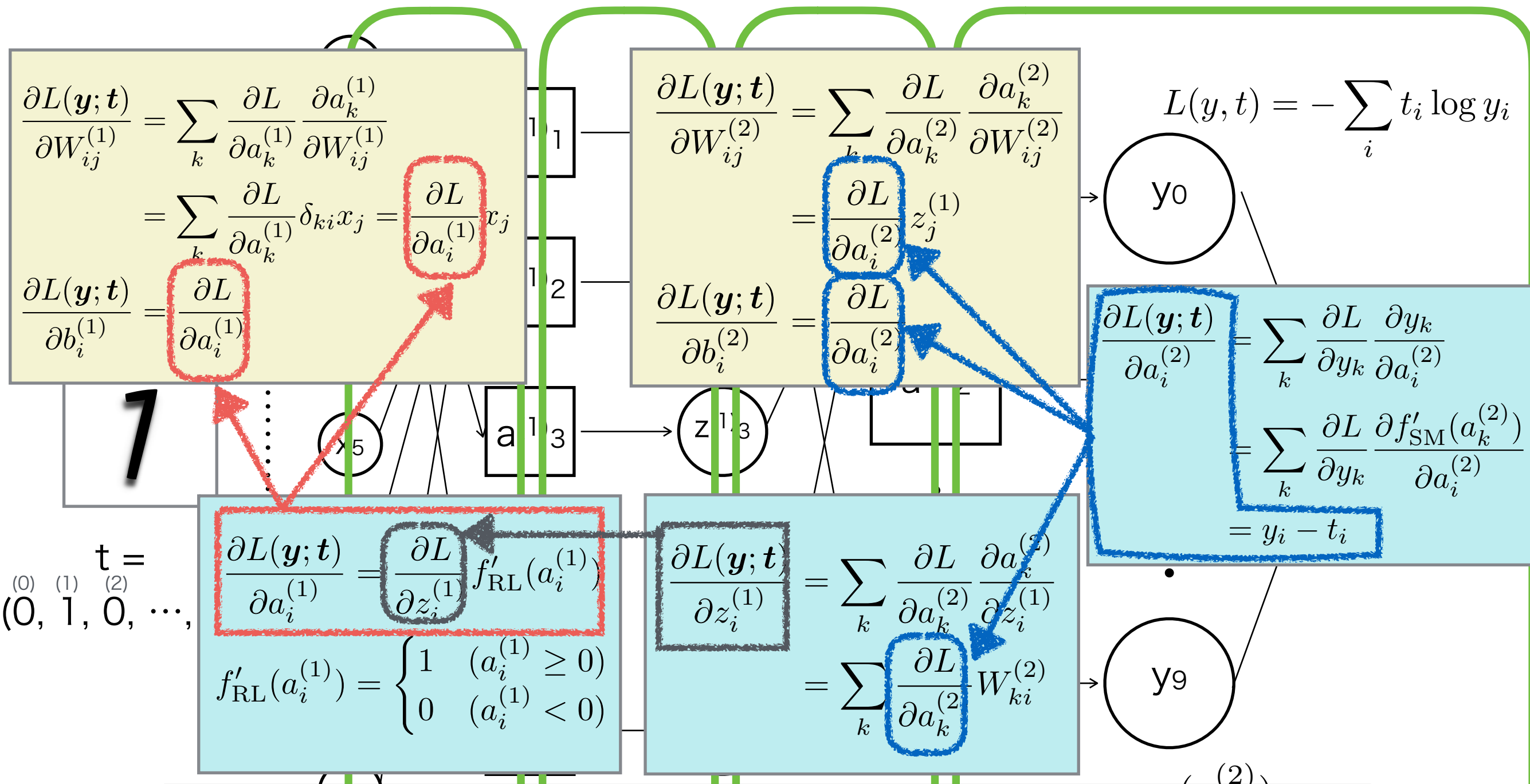


$$\mathbf{a}^{(1)} = \mathbf{W}^{(1)} \mathbf{x} + \mathbf{b}^{(1)} \quad \mathbf{a}^{(2)} = \mathbf{W}^{(2)} \mathbf{z}^{(1)} + \mathbf{b}^{(2)}$$



- ・まず順方向に $L(\mathbf{y}; \mathbf{t})$ を計算 (途中の $\mathbf{z}^{(1)}$, \mathbf{x} も保存しておく)
- ・逆方向に, 各レイヤーごと微分値・必要なパラメータを計算

$$\mathbf{a}^{(1)} = \mathbf{W}^{(1)} \mathbf{x} + \mathbf{b}^{(1)} \quad \mathbf{a}^{(2)} = \mathbf{W}^{(2)} \mathbf{z}^{(1)} + \mathbf{b}^{(2)}$$



- まず順方向に $L(\mathbf{y}; \mathbf{t})$ を計算 (途中の $\mathbf{z}^{(1)}$, \mathbf{x} も保存しておく)
- 逆方向に, 各レイヤーごと微分値・必要なパラメータを計算

SoftMax & CrossEntropy Layer

$$\mathbf{a}^{(1)} = \mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)} \quad \mathbf{a}^{(2)} = \mathbf{W}^{(2)}\mathbf{z}^{(1)} + \mathbf{b}^{(2)}$$

out

$$\begin{aligned} \frac{\partial L(\mathbf{y}; \mathbf{t})}{\partial a_i^{(2)}} &= \sum_k \frac{\partial L}{\partial y_k} \frac{\partial y_k}{\partial a_i^{(2)}} \\ &= \sum_k \frac{\partial L}{\partial y_k} \frac{\partial f'_{\text{SM}}(a_k^{(2)})}{\partial a_i^{(2)}} \\ &= y_i - t_i \end{aligned}$$

$$L(\mathbf{y}, \mathbf{t}) = - \sum_i t_i \log y_i$$

y_0

y_1

\vdots

y_9

$L(\mathbf{y}, \mathbf{t})$

$$y_i = \frac{\exp(a_i^{(2)})}{\sum_k \exp(a_k^{(2)})}$$

```
class SoftmaxWithLoss:
```

```
    def __init__(self):
```

```
        self.loss = None
```

```
        self.y = None
```

```
        self.t = None
```

```
    def forward(self, y, t):
```

```
        self.t = t
```

```
        self.y = y
```

```
        self.loss =
```

```
        cross_entropy_error(self.y, self.t)
```

```
        return self.loss
```

```
    def backward(self, dout=1):
```

```
        batch_size = self.t.shape[0]
```

```
        dx = (self.y - self.t) / batch_size
```

```
        return dx
```

Affine Layer (2)

$$\mathbf{a}^{(1)} = \mathbf{W}^{(1)} \mathbf{x} + \mathbf{b}^{(1)} \quad \mathbf{a}^{(2)} = \mathbf{W}^{(2)} \mathbf{z}^{(1)} + \mathbf{b}^{(2)}$$

class Affine:

def __init__(self, W, b):

self.W = W

self.b = b

self.x = None

self.dW = None

self.db = None

def forward(self, x):

self.x = x

out = np.dot(x, self.W) + self.b

return out

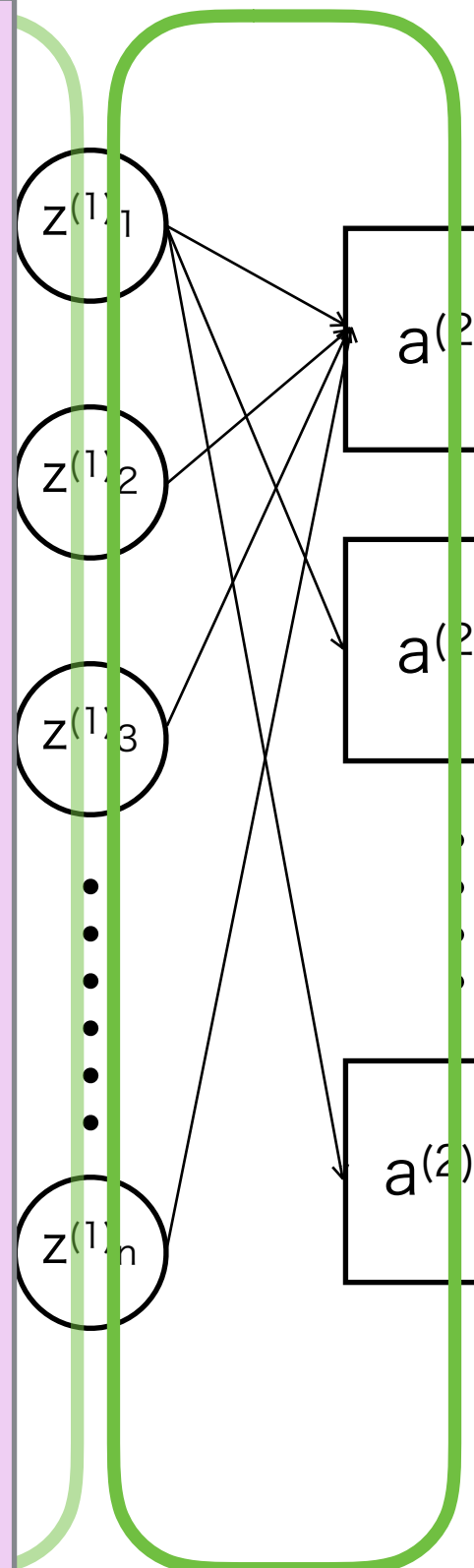
def backward(self, dout):

dx = np.dot(dout, self.W.T)

self.dW = np.dot(self.x.T, dout)

self.db = np.sum(dout, axis=0)

return dx



calc.

$$\frac{\partial L(\mathbf{y}; \mathbf{t})}{\partial W_{ij}^{(2)}} = \sum_k \frac{\partial L}{\partial a_k^{(2)}} \frac{\partial a_k^{(2)}}{\partial W_{ij}^{(2)}}$$

$$= \frac{\partial L}{\partial a_i^{(1)}} z_j^{(1)}$$

$$\frac{\partial L(\mathbf{y}; \mathbf{t})}{\partial b_i^{(2)}} = \frac{\partial L}{\partial a_i^{(2)}}$$

out

$$\frac{\partial L(\mathbf{y}; \mathbf{t})}{\partial z_i^{(1)}} = \sum_k \frac{\partial L}{\partial a_k^{(2)}} \frac{\partial a_k^{(2)}}{\partial z_i^{(1)}}$$

$$= \sum_k \frac{\partial L}{\partial a_k^{(2)}} W_{ki}^{(2)}$$

$$z_i^{(1)} = f(a_i^{(1)})$$

ReLu Layer

$$\mathbf{a}^{(1)} = \mathbf{W}^{(1)} \mathbf{x} + \mathbf{b}^{(1)} \quad \mathbf{a}^{(2)} = \mathbf{W}^{(2)} \mathbf{z}^{(1)} + \mathbf{b}^{(2)}$$

out

$$\frac{\partial L(\mathbf{y}; \mathbf{t})}{\partial a_i^{(1)}} = \frac{\partial L}{\partial z_i^{(1)}} f'_{\text{RL}}(a_i^{(1)})$$

$$f'_{\text{RL}}(a_i^{(1)}) = \begin{cases} 1 & (a_i^{(1)} \geq 0) \\ 0 & (a_i^{(1)} < 0) \end{cases}$$

1

$\mathbf{t} =$

$(0, 1, 0, \dots, 0)$

x_5

x_6

\vdots

x_{784}

$a^{(1)}_3$

\vdots

$a^{(1)}_n$

$z^{(1)}_1$

$z^{(1)}_2$

$z^{(1)}_3$

\vdots

$z^{(1)}_n$

class Relu:

def __init__(self):

self.mask = None

def forward(self, x):

self.mask = (x < 0)

out = x.copy()

out[self.mask] = 0

return out

def backward(self, dout):

dout[self.mask] = 0

dx = dout

return dx

$\log y_i$

\mathbf{t}

$$\sum_k \exp(a_k^{(2)})$$

$$z_i^{(1)} = f(a_i^{(1)})$$

Affine Layer (1)

$$\mathbf{a}^{(1)} = \mathbf{W}^{(1)} \mathbf{x} + \mathbf{b}^{(1)} \quad \mathbf{a}^{(2)} = \mathbf{W}^{(2)} \mathbf{z}^{(1)} + \mathbf{b}^{(2)}$$

