

しおん研-ゼロつくゼミ

第5回

今日やること

- ▶ 誤差逆伝播法の理論
- ▶ 単純なレイヤの実装
- ▶ 活性化関数レイヤの実装
- ▶ Affineレイヤ・Softmaxレイヤの実装

学習のフレームワーク（復習）

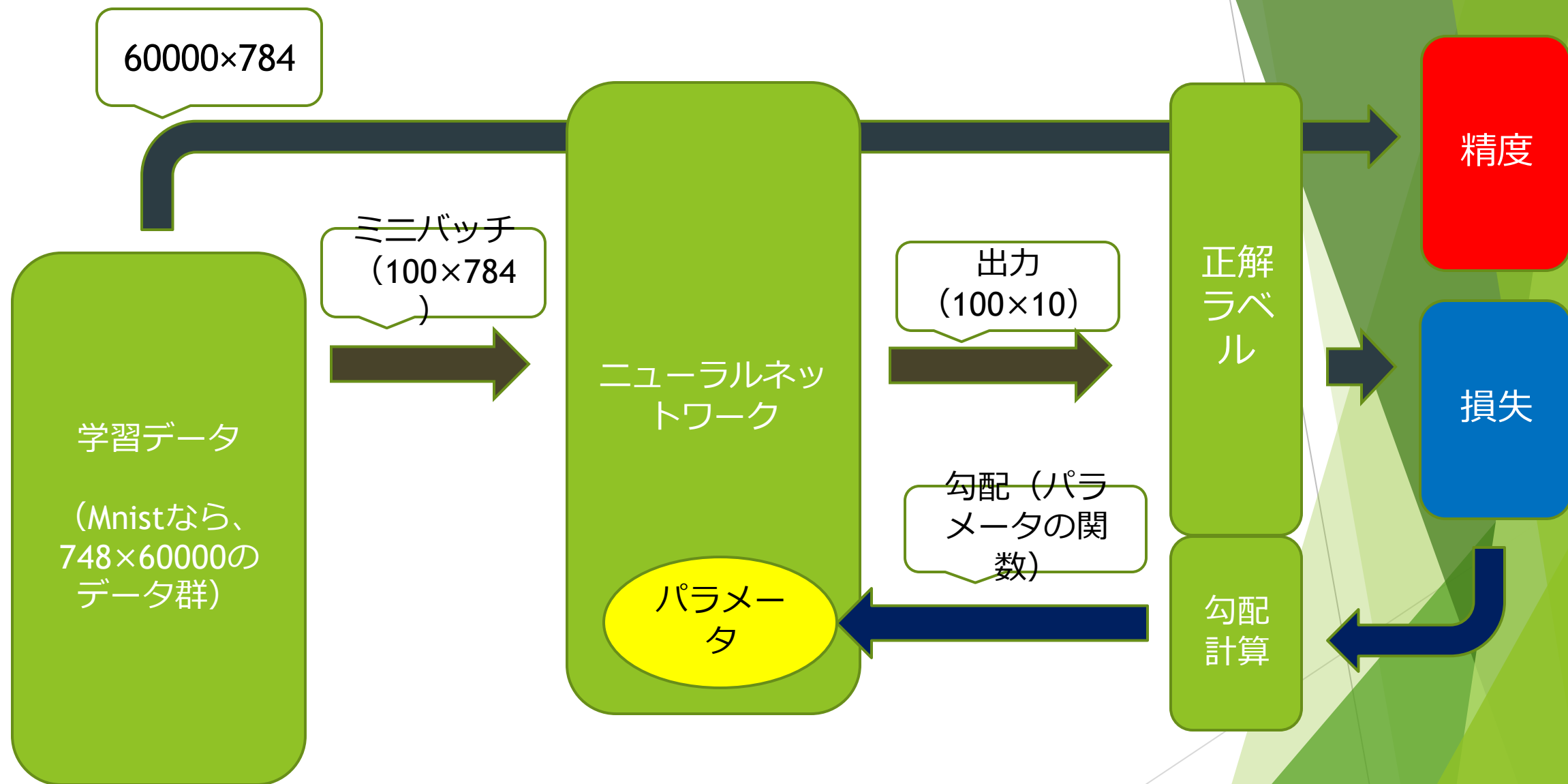
- ▶ 目的は、推論に最適な（＝もっとも適切な出力を返す）パラメータを見つけること！
- ▶ そのためにはどうすればいい？
 - ある推論に対して、それがどれくらい正しいか評価する関数を定義し、その評価が最大になるようにパラメータを定めれば良いのでは？

学習データ X → 出力データ $Y = f(X, W)$

正解データ t



評価関数 $L(W)$
= 損失関数



学習のフレームワーク

これまでの勾配計算：数値微分

- ▶ 一つのパラメータを更新するのに、ニューラルネットの推論課程を2回行う必要がある！
- ▶ つまり、一回の学習で、順方向計算を
 $2 \times (\text{パラメータの総数})$ 回行わなければならない！
- ▶ パラメータが増えると計算が終わらない・・・

数値微分に代わる勾配計算法：誤差逆伝播法

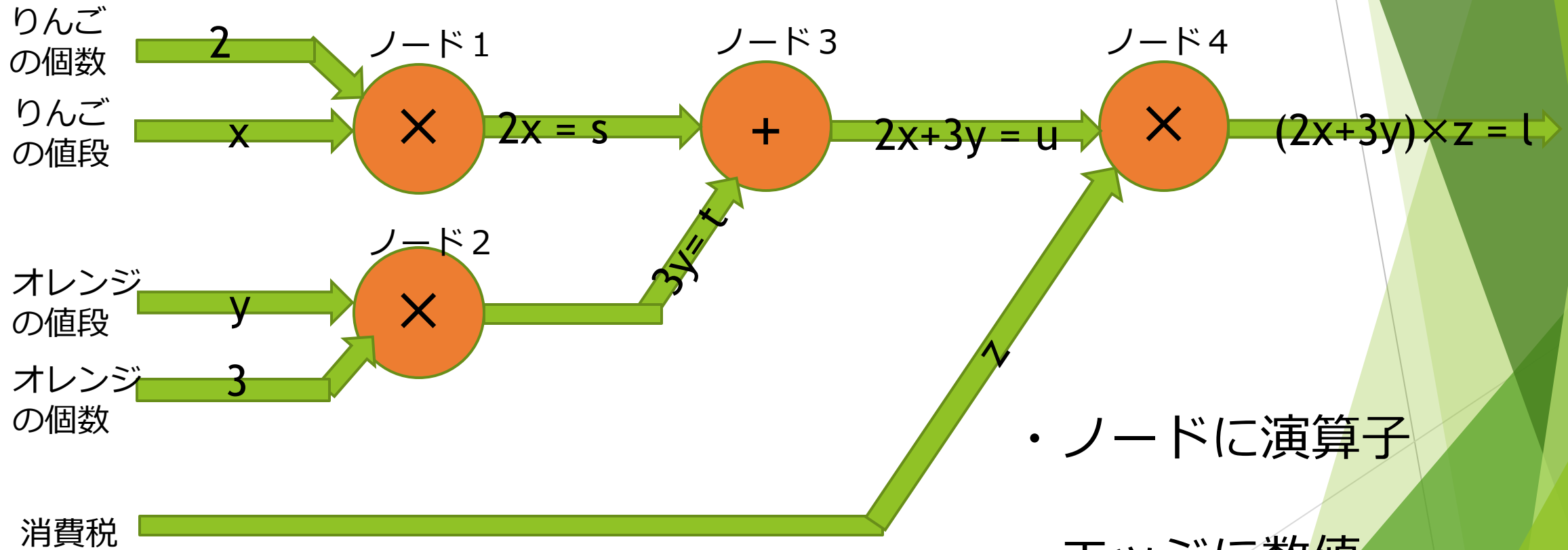
(例題)

► Polinaさんは毎日、ロシアのスーパーでりんごを2個とオレンジを3個買っています。ここで、

- 1.りんごが昨日と比べて10ルーブル値上がりしていた
- 2.オレンジが昨日と比べて5ルーブル値上がりしていた
- 3.暴動により、今日から消費税が20%から10%に下がった

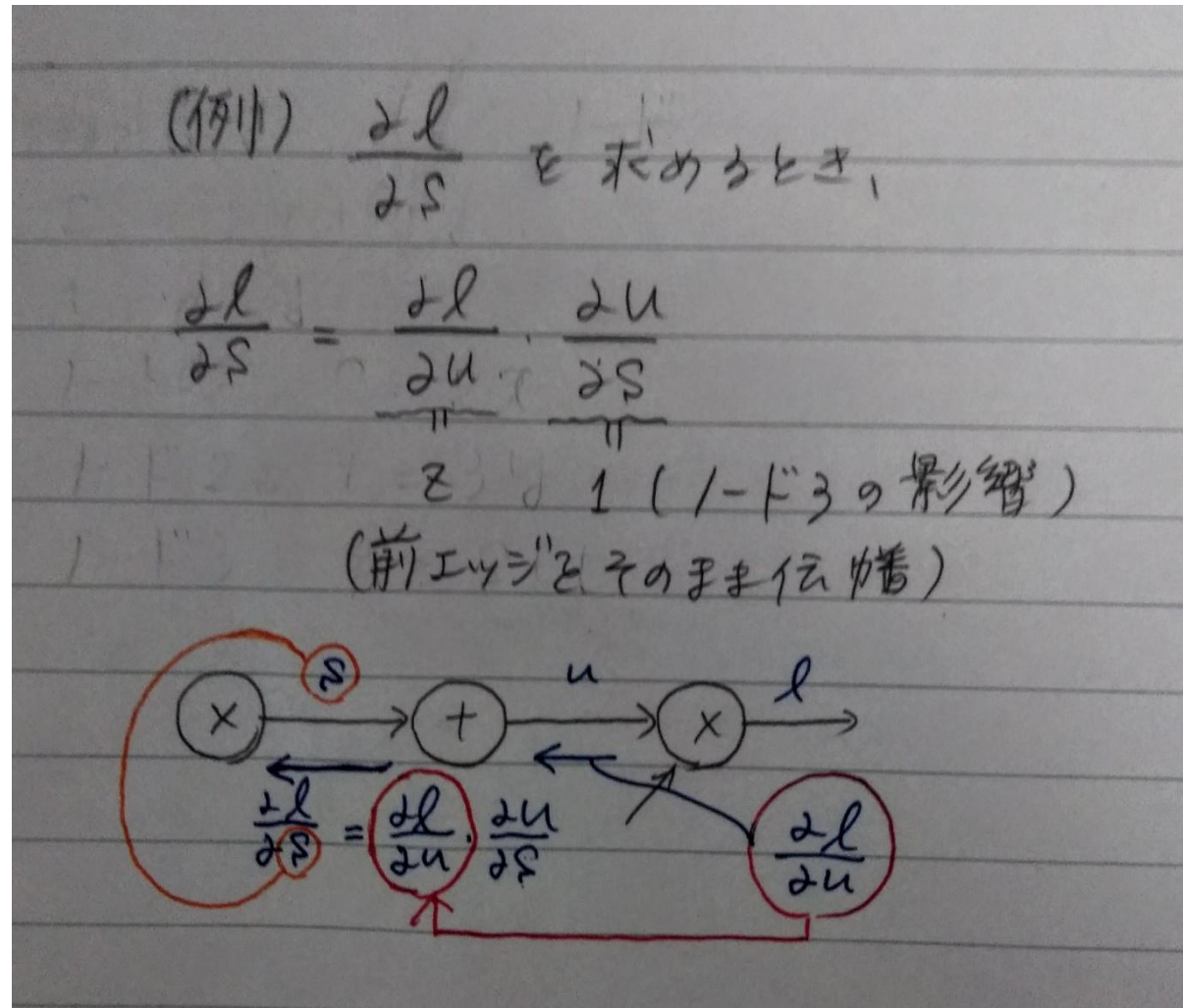
それぞれの時、Polinaさんは昨日より何ルーブル多く支払う必要があるのでしょうか。

(解答例) 計算グラフ

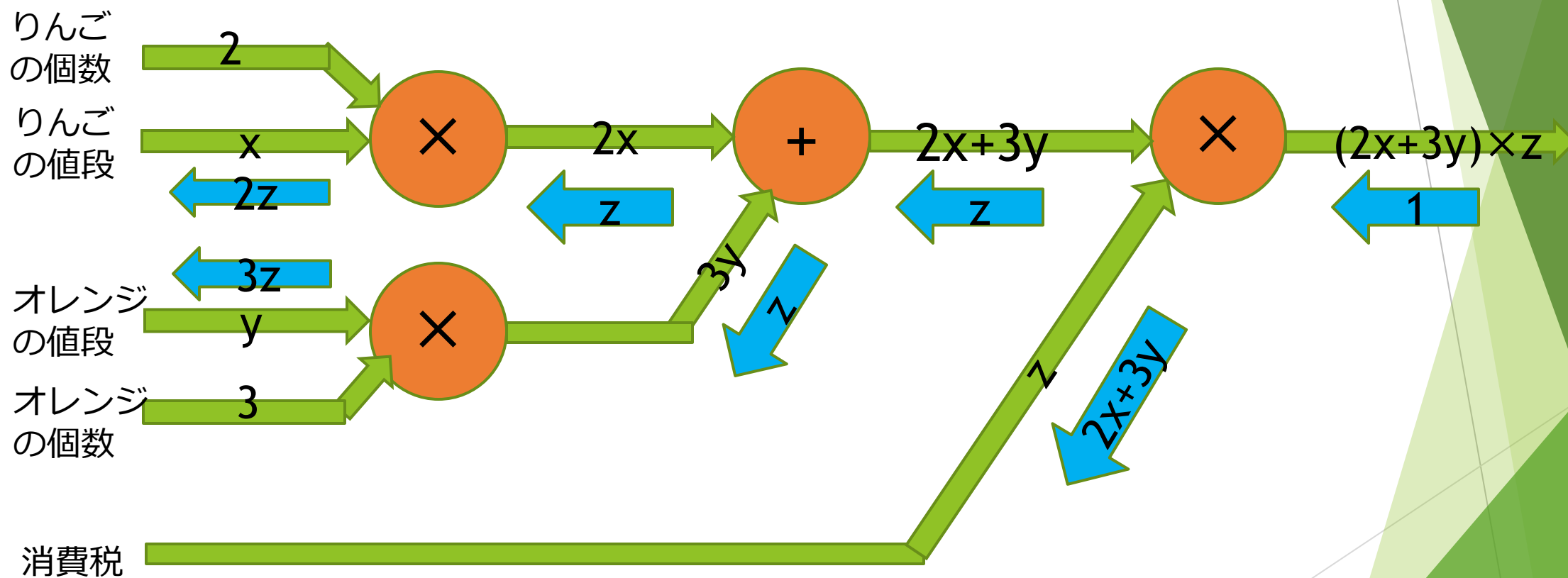


- ・ ノードに演算子
- ・ エッジに数値

誤差逆伝播法：勾配を逆方向に伝播する！ (575)



勾配を逆方向に伝播する！



活性化関数レイヤの逆伝播 (Sigmoid)

▶ Sigmoidレイヤ

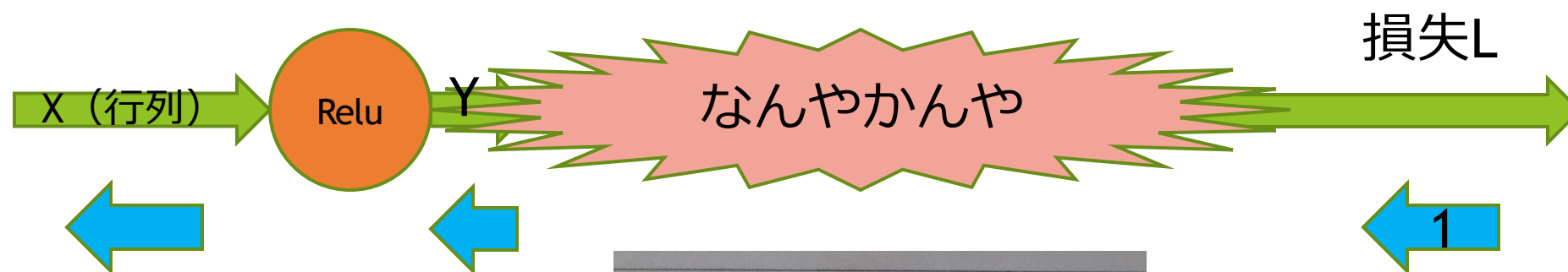


活性化関数レイヤの逆伝播 (Sigmoid)

$$y_{ij} = \frac{1}{1 + e^{-x_{ij}}}$$
$$\frac{\partial y_{ij}}{\partial x_{ij}} = \frac{e^{-x_{ij}}}{(1 + e^{-x_{ij}})^2}$$
$$= y_{ij}(1 - y_{ij})$$
$$\therefore \frac{\partial L}{\partial x_{ij}} = \frac{\partial L}{\partial y_{ij}} \cdot y_{ij}(1 - y_{ij})$$

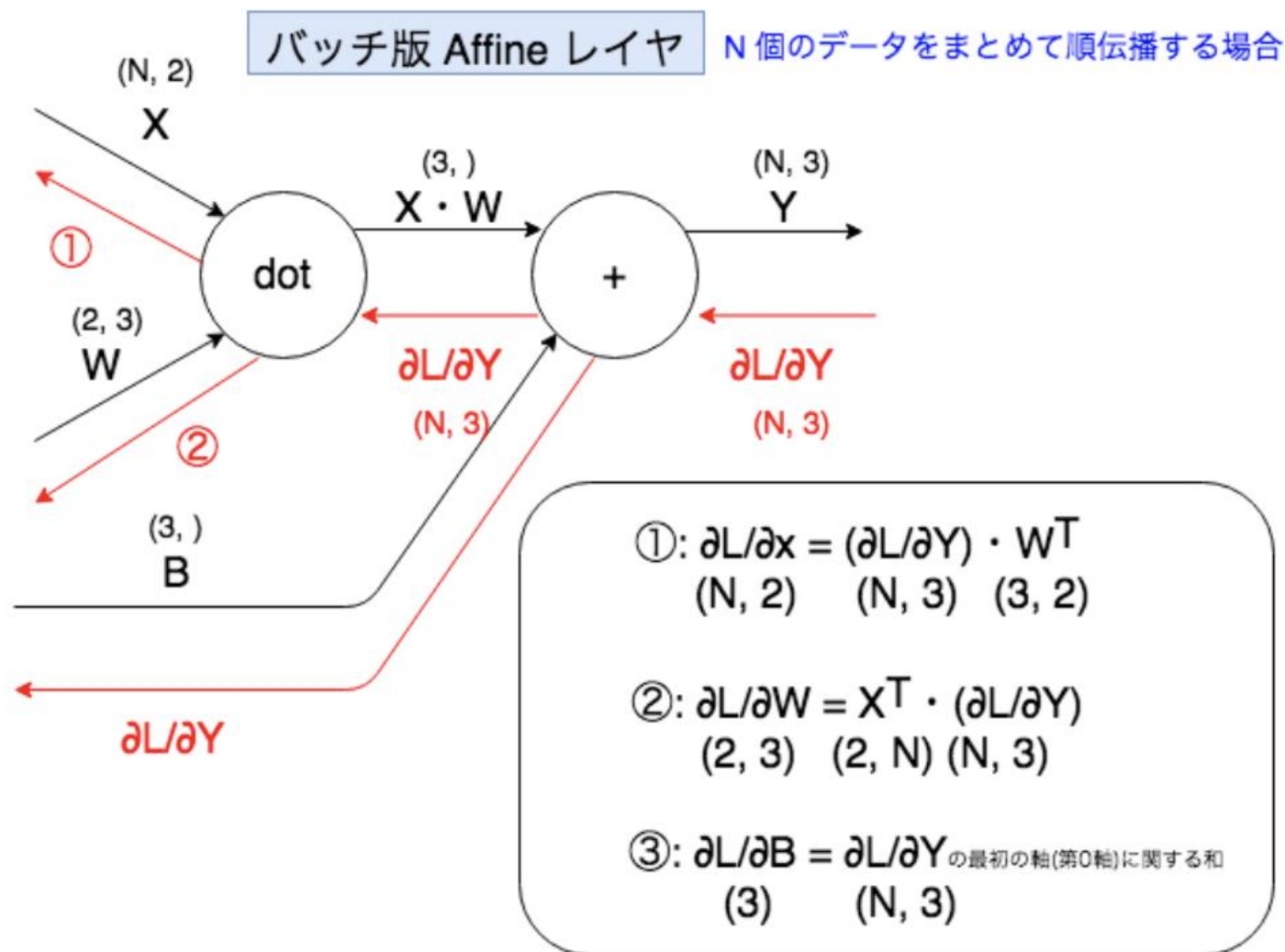
活性化関数レイヤの逆伝播 (Relu)

► Reluレイヤ



$$\frac{\partial y_{ij}}{\partial x_{ij}} = \begin{cases} 1 & (x_{ij} > 0) \\ 0 & (x_{ij} \leq 0) \end{cases}$$

Affineレイヤの逆伝播



Affineレイヤ の逆伝播 (証明)

$$\boxed{3} \quad y_{ij} = a_{ij} + b_j$$

$$dL = \sum_{i,j} \frac{\partial L}{\partial y_{ij}} dy_{ij} \quad (\because \text{全微分の公式})$$

$$\frac{\partial L}{\partial b_k} = \sum_{i,j} \frac{\partial L}{\partial y_{ij}} \cdot \frac{\partial y_{ij}}{\partial b_k}$$

$$= \sum_i \frac{\partial L}{\partial y_{ij}} \cdot \frac{\partial y_{ij}}{\partial b_k}$$

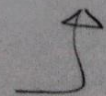
$$= \sum_i \frac{\partial L}{\partial y_{ij}}$$

↓
dout の i 行 = どの和

$$\frac{\partial L}{\partial a_{kl}} = \sum_{i,j} \frac{\partial L}{\partial y_{ij}} \cdot \frac{\partial y_{ij}}{\partial a_{kl}}$$

$$= \frac{\partial L}{\partial y_{kl}}$$

勾配をそのまま伝播



Affineレイ ヤの逆伝播 (証明)

[1] [2]

$$a_{ij} = \sum_k x_{ik} w_{kj}$$

$$dL_j = \sum_{i,j} \frac{\partial L}{\partial y_{ij}} \cdot dy_{ij}$$

$$= \sum_{i,j} \frac{\partial L}{\partial y_{ij}} dy_{ij}$$

$$\frac{\partial L}{\partial x_{kl}} = \sum_{i,j} \frac{\partial L}{\partial y_{ij}} \cdot \frac{\partial y_{ij}}{\partial x_{kl}}$$

$$= \sum_{i,j} \frac{\partial L}{\partial y_{ij}} \cdot \frac{\partial y_{ij}}{\partial a_{ij}} \cdot \frac{\partial a_{ij}}{\partial x_{kl}}$$

$$= \sum_j \frac{\partial L}{\partial a_{kj}} \cdot w_{lj}$$

$$= \sum_j \frac{\partial L}{\partial y_{kj}} \cdot w_{lj}$$

$$\therefore \frac{\partial L}{\partial X} = \frac{\partial L}{\partial Y} \cdot W^T \dots [1]$$

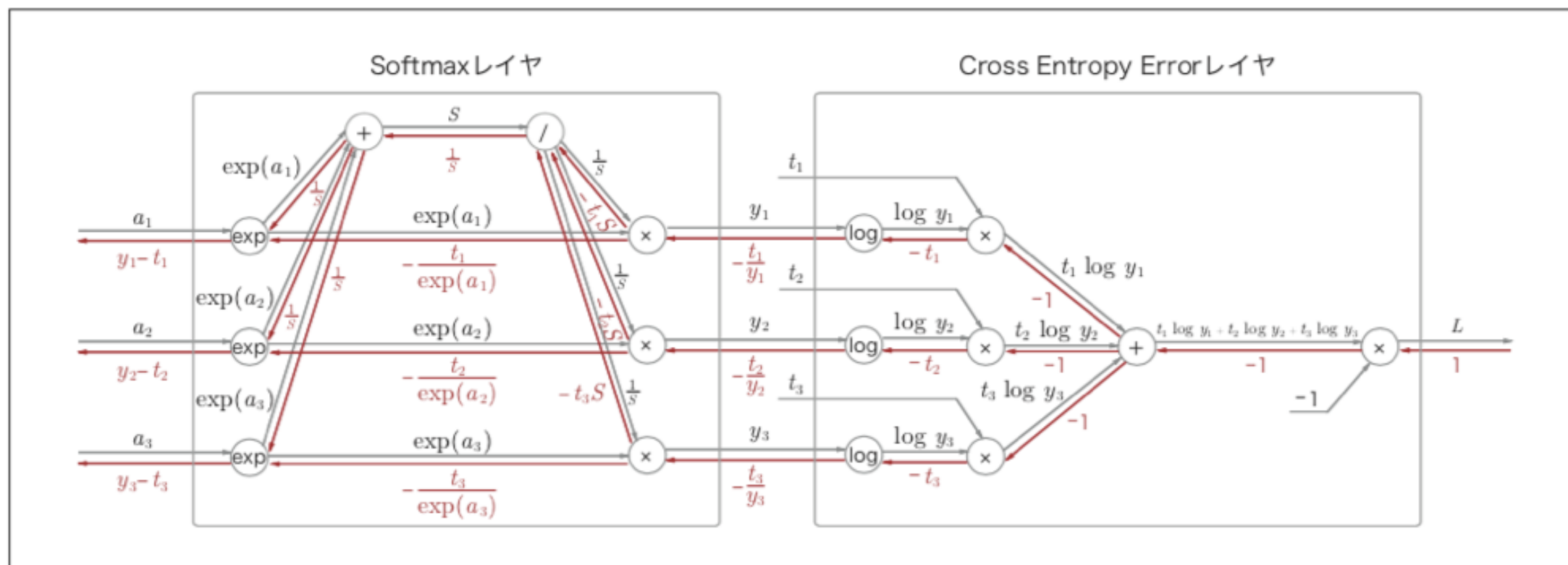
$$\frac{\partial L}{\partial w_{kl}} = \sum_{i,j} \frac{\partial L}{\partial y_{ij}} \cdot \frac{\partial y_{ij}}{\partial a_{ij}} \cdot \frac{\partial a_{ij}}{\partial w_{kl}}$$

$$= \sum_i \frac{\partial L}{\partial a_{il}} \cdot x_{ik}$$

$$\therefore \frac{\partial L}{\partial W} = X^T \cdot \frac{\partial L}{\partial Y} \dots [2]$$

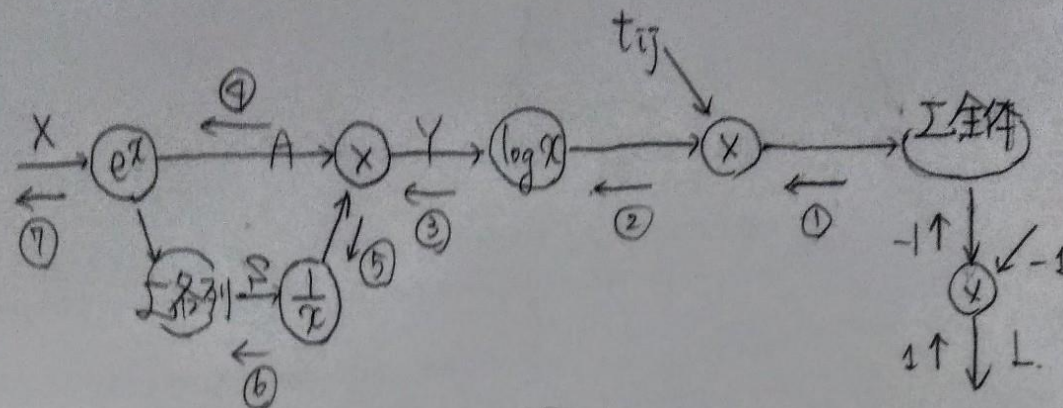
SoftmaxWithLossレイヤの逆伝播

Softmax-with-Loss レイヤ(ソフトマックス関数と交差エントロピー誤差)の計算グラフ



SoftmaxWithLoss レイヤの逆伝播 (証明)

Softmax With Loss.



$$\textcircled{1} \begin{pmatrix} -1 & \dots & -1 \\ \vdots & & \vdots \\ -1 & \dots & -1 \end{pmatrix} \begin{matrix} m \\ n \end{matrix}, \begin{pmatrix} y = \sum_k x_k \cdot a_{kj} \text{ とき,} \\ \frac{\partial y}{\partial x_k} = 1 \end{pmatrix}$$

$$\textcircled{2} \begin{pmatrix} -t_{1j} & \dots & -t_{1n} \\ \vdots & & \vdots \\ -t_{mj} & \dots & -t_{mn} \end{pmatrix} \quad \textcircled{3} \left(\frac{\partial L}{\partial Y} \right)_{ij} = - \frac{t_{ij}}{Y_{ij}} = - \frac{t_{ij} \cdot S_i}{A_{ij}}$$

$$\textcircled{5} \left(\frac{\partial L}{\partial \left(\frac{1}{S} \right)} \right)_i = \sum_{j=1}^n (-t_{ij} S_i) = -S_i$$

$$\textcircled{6} \left(\frac{\partial L}{\partial S} \right)_i = \frac{1}{S_i}$$

$$\textcircled{7} \left(\frac{\partial L}{\partial X} \right)_{ij} = \frac{A_{ij}}{S_i} - \frac{t_{ij}}{A_{ij}} \cdot A_{ij} = Y_{ij} - t_{ij} \quad \uparrow$$

レイヤの実装

▶ レイヤ = 関数だと思えばOK

▶ レイヤ実装の基本（ここでは）

```
class hoge:
```

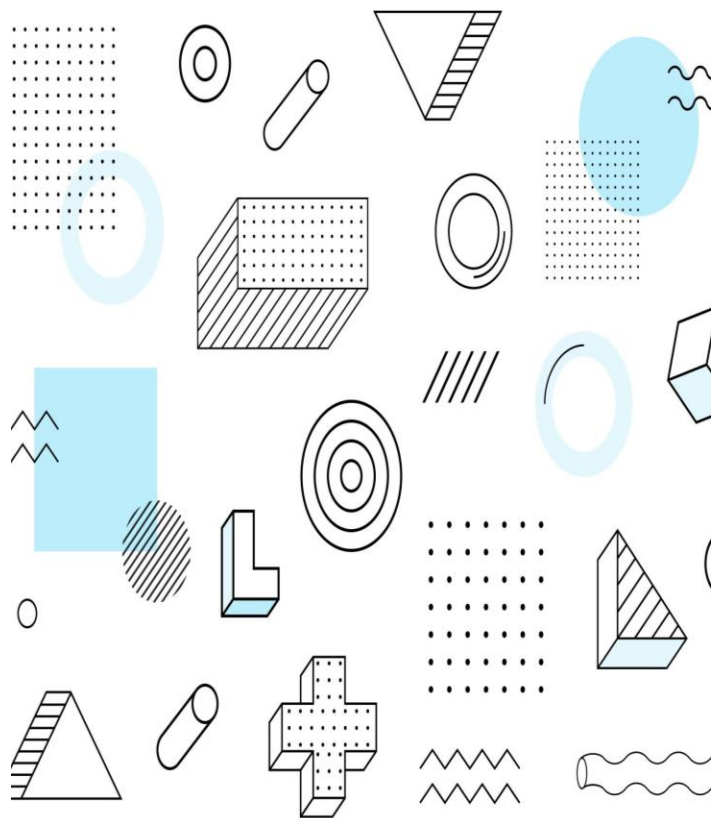
```
    def __init__(ハイパーパラメータ):
```

```
        #ハイパーパラメータに加え、順伝播で算出した値を逆伝播でも使う時にインスタンス変数  
        として格納
```

```
    def forward(self,x):
```

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

▶ このレイヤを逆にたどることで、勾配が計算できるようにする！



Google Colab で実装

- ・ 加算、乗算レイヤ
- ・ 活性化関数レイヤ
- ・ Affine、Softmaxレイヤの実装

画像の出典：いずれも <https://www.dragonarrow.work/articles/176>