Getting started

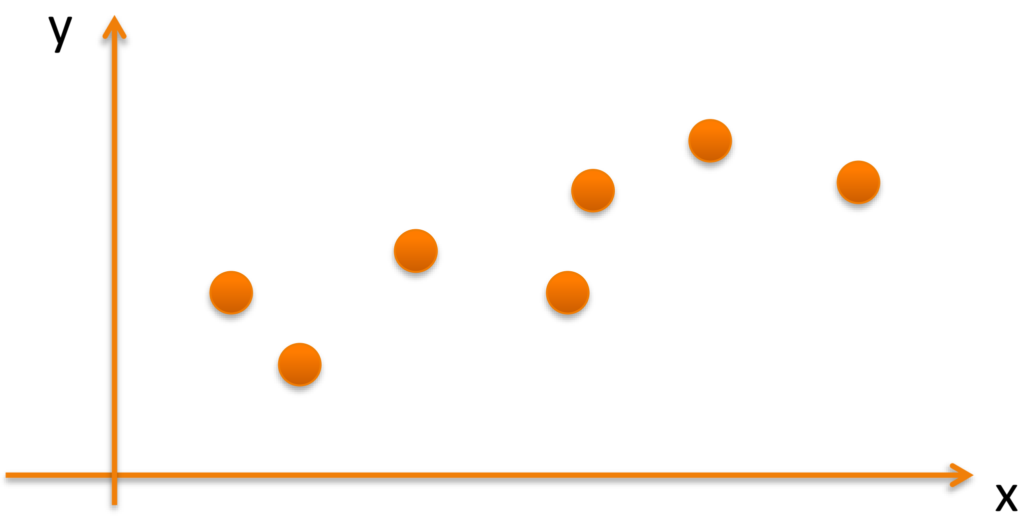
Thorough understanding of AI-CNN （Convolutional Neural Network）

- Thorough understanding of the mechanism of AI and CNN
- Mouse handwritten character recognition with Colaboratory

Table of contents

1-1. Machine learning Example 1) Simple linear regression

Simple linear regression is a linear function (y = ax + b) that can express the data of two variables x and y



There is data that can be represented by a straight line, and a relational expression is sought.

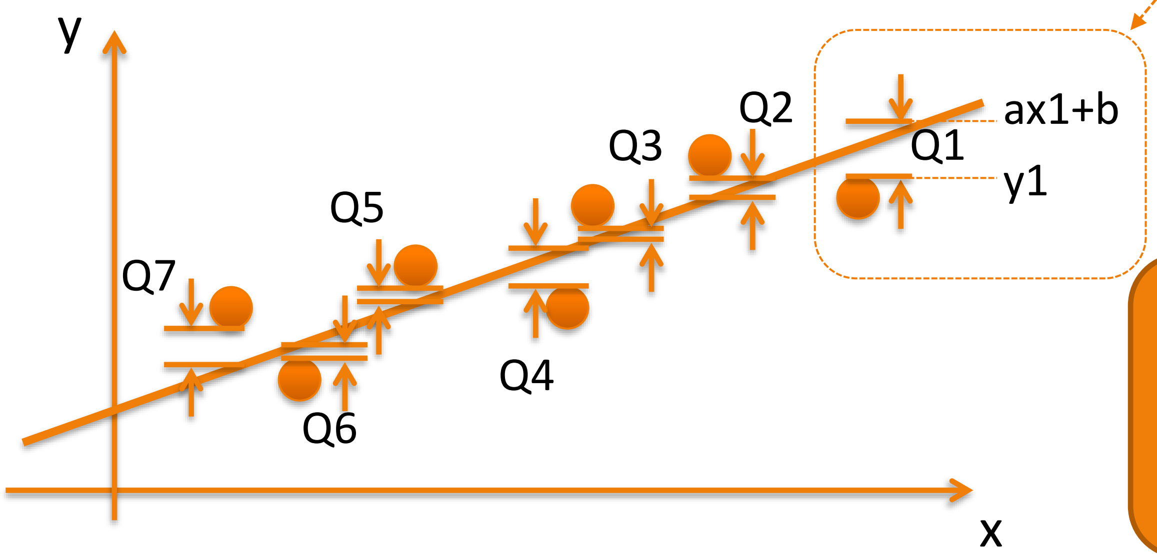
error is「Q1 = y1 – (ax1 + b)」

Since there is an error on the top and bottom, sum the squares

Sum of errors： $Qt = Q1 + Q2 + Q3 + \ldots$

$= \{ y1 - (ax1 + b)\}^2 + \{ y2 - (ax2 + b)\}^2 + \{ y3 - (ax3 + b)\}^2 + \ldots$

Relational expression that minimizes the sum of errors



<u>The parameters a and b are determined so that the total error (objective function) Qt is minimized.</u>

⇨ least squares method

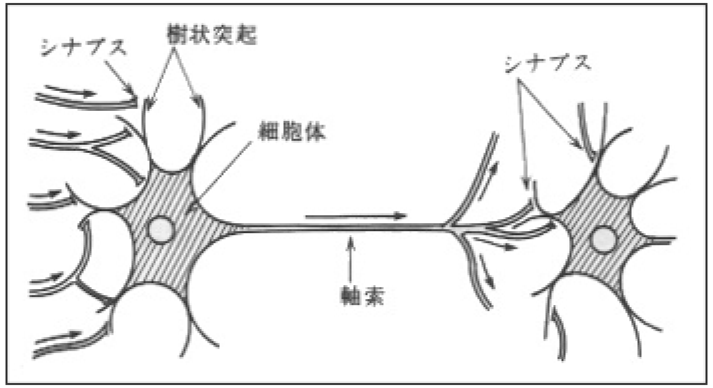
① Decide on a model (linear function this time)

② The sum of errors (loss function) due to learning data is minimal Determine the parameter that will be (or maximum)

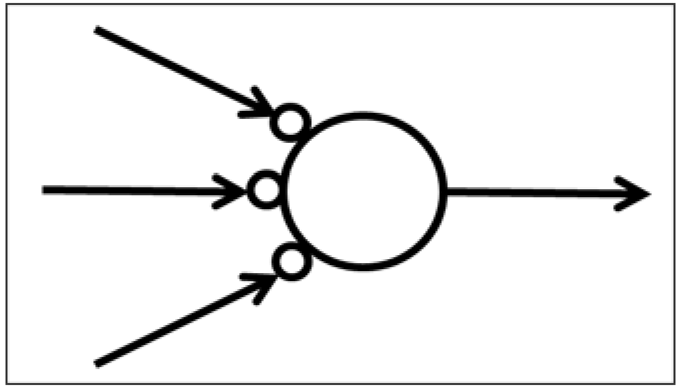
1-2. Machine learning　　Example 2) Perceptron (artificial neuron)

Artificial neurons (perceptrons) modeled on human brain neurons

Human neurons (nerve cells)



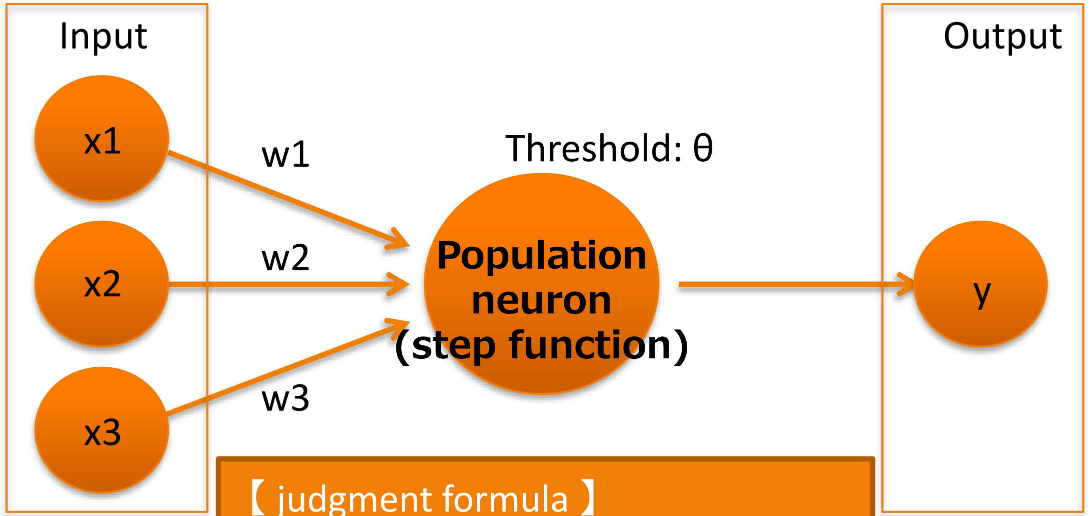
Neuron (nerve cell) modeling



https://aitokuconsult.hatenablog.com/entry/neuralnetwork

input signal
x1〜x3

Weight of input
w1〜w3



【 judgment formula 】
(x1 * w1) + (x1 * w1) + (x1 * w1) > θ

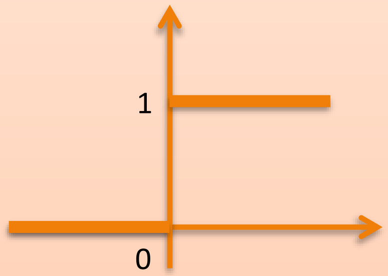
Ex) When each input is 1, 2, 3, all weights are 1, and the threshold is 5

$$(1 * 1) + (2 * 1) + (3 * 1) > 5$$
$$6 > 5$$

【Step function】
correct, so the output is "1"

There are many functions in the output

(1) Activation function: Step function

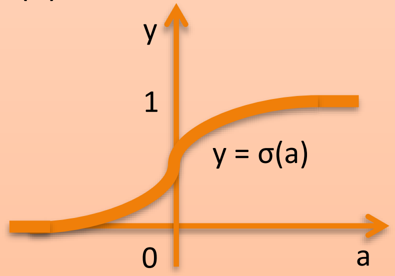


The output is a binary value of 0 or 1

(x1 * w1) + (x1 * w1) + (x1 * w1)
> θ

1 if true, 0 if false

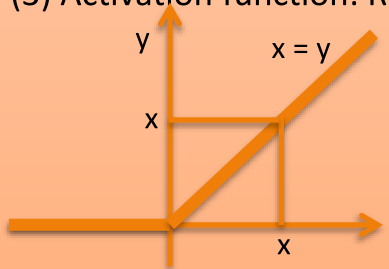
(2) Activation function: Sigmoid function



Output can be any value between 0 and 1

The output is expressed as y = σ(a).
σ is a sigmoid function and
σ(x) = 1 / (1 + Exp-x)a is called the "linear sum of the inputs“
(x1 * w1) + (x2 * w2) + (x3 * w3) – θ
Often used in the output layer.

(3) Activation function: Ramp function

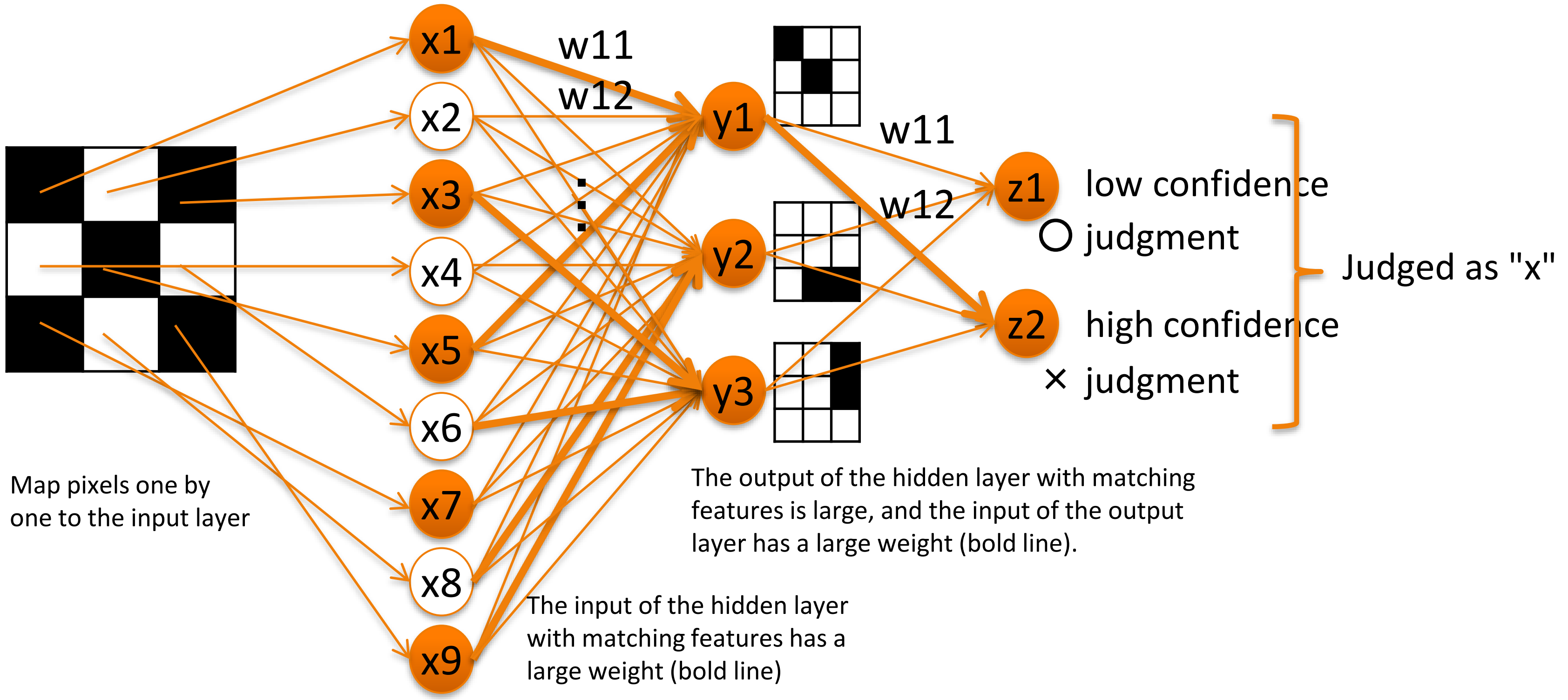


output is any value from 0

Simple and easy to calculate.
Commonly used in the middle class.

1-4. Machine learning　Example 3) Neural network

In order to understand the neural network, the intermediate layer (hidden layer) learns with a single layer model (Neural network that judges ○ “circle” and × “cross” of 3*3 images)



Image

Input layer

Input information to all hidden layers as it is

hidden layer

The hidden layer has a detection pattern and performs feature extraction

Output layer

Make ○ × decision

Each neuron weights the input and outputs the result calculated by the threshold

【 Middle layer formula】

a1 = (x1 * w11) + (x2 * w12) ・・・ + (x9 * w19) – θ1
a2 = (x1 * w21) + (x2 * w22) ・・・ + (x9 * w29) – θ2
a3 = (x1 * w31) + (x2 * w32) ・・・ + (x9 * w39) – θ3
y1 = σ(a1), y2 = σ(a2), y3 = σ(a3)
σ is a sigmoid function

【 Output layer formula 】

z1 = (y1 * w11) + (y2 * w12) + (y3 * w13) – θ1
z2 = (y1 * w11) + (y2 * w22) + (y3 * w23) – θ2

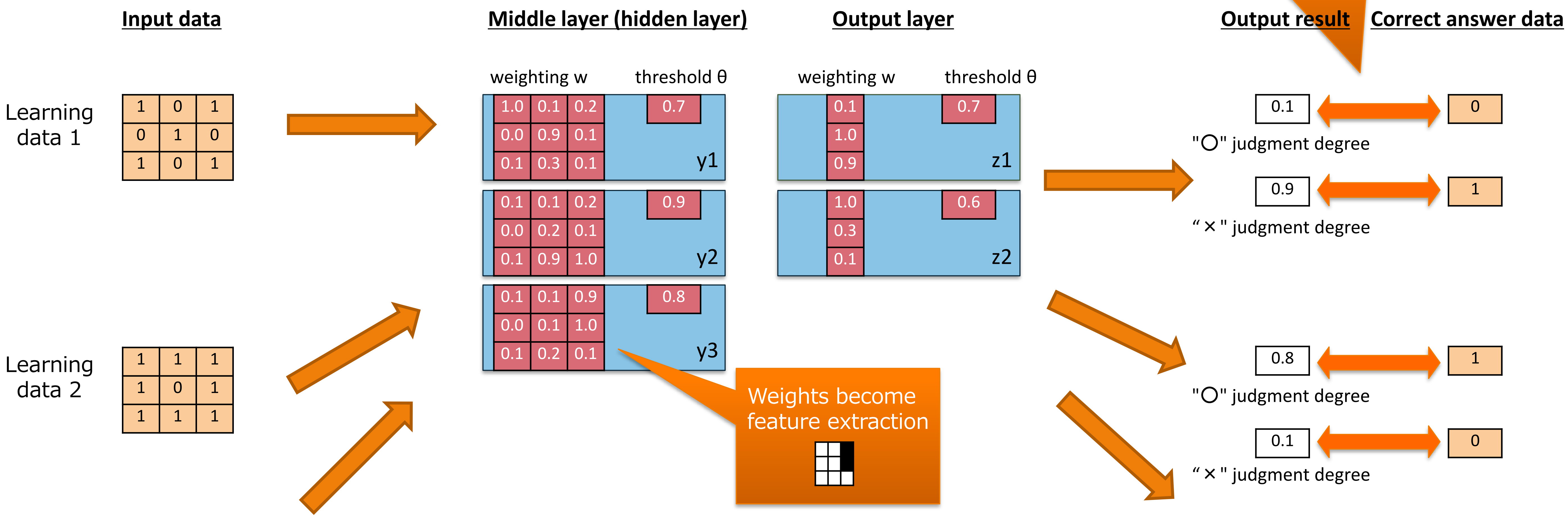
1-4. Machine learning　Example 3) Neural network

1.0 : given data

1.0 : parameters to be optimized

Parameters are optimized so that the output is equal to the correct data from the given data [Input data and correct data are given as a set]

Parameters (weighting) so as to be close to all correct dataand threshold)



Learning with large amounts of data

Hidden layer and output layer are common for all training data
(optimize parameters for all training data)

2. Google Colabratory and MNIST data

GoogleColabratory

A Python development environment provided by Google for AI research and learning.
No preparation such as installation is required, and it can be used immediately with a web browser.
You can use it for free, but there is a limit of 12 hours at a time.

MNIST data

MNIST stands for "Mixed National Institute of Standards and Technology database".
It is provided free of charge for training AI on datasets publicly available on the internet.

The image dataset consists of 60,000 images of handwritten digits and 10,000 test images.

3-1. AI program

```
import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

(x_train, y_train), (x_test, y_test) = keras.datasets.mnist.load_data()

x_train = x_train.astype("float32") / 255
x_test = x_test.astype("float32") / 255

x_train = np.expand_dims(x_train, -1)
x_test = np.expand_dims(x_test, -1)

y_train = keras.utils.to_categorical(y_train, 10)
y_test = keras.utils.to_categorical(y_test, 10)

model = keras.Sequential(
    [
        keras.Input(shape=(28, 28, 1)),
        layers.Conv2D(32, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Flatten(),
        layers.Dropout(0.5),
        layers.Dense(10, activation="softmax"),
    ]
)

model.summary()

model.compile(loss="categorical_crossentropy", optimizer="adam", metrics=["accuracy"])

model.fit(x_train, y_train, batch_size=128, epochs=15, validation_split=0.1)

score = model.evaluate(x_test, y_test, verbose=0)
print("Test loss:", score[0])
print("Test accuracy:", score[1])
```

① Define the library to use

② Prepare data

- Load training and validation data (MNIST) to be used (x: image data, y: correct label, train: 60,000 images, test: 10,000 images)
- Convert 0 to 255 grayscale to 0 to 1 values
- Add one dimension (to match the data format handled by AI)
- Convert integer values of label data to binary class matrix (Example: "Integer 2" is expressed as "0,0,1,0,0,0,0,0,0,0")

③ Define AI model

④ Display AI model

⑤ Parameter optimization (AI learning) setting

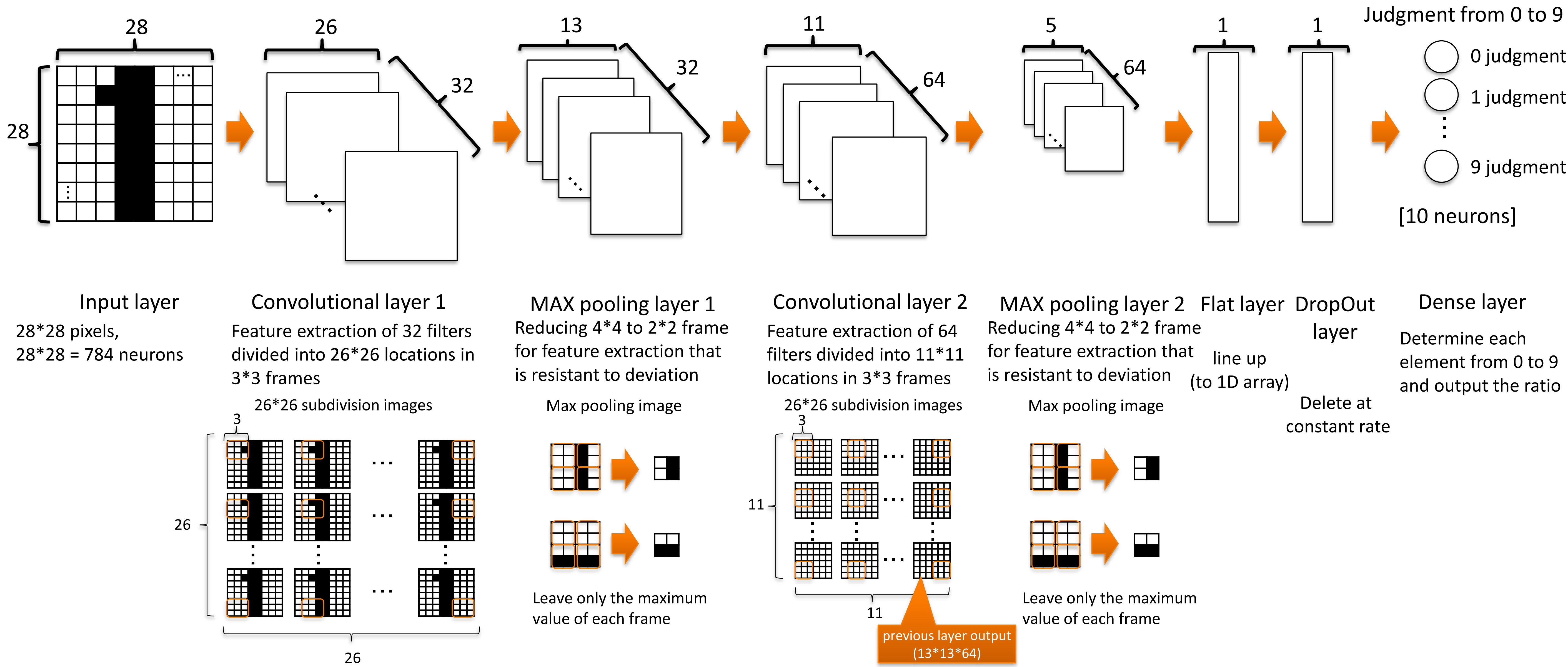
⑥ Execution of parameter optimization (AI learning)

⑦ Evaluate AI performance after learning using test data

3-2. Convolutional neural network

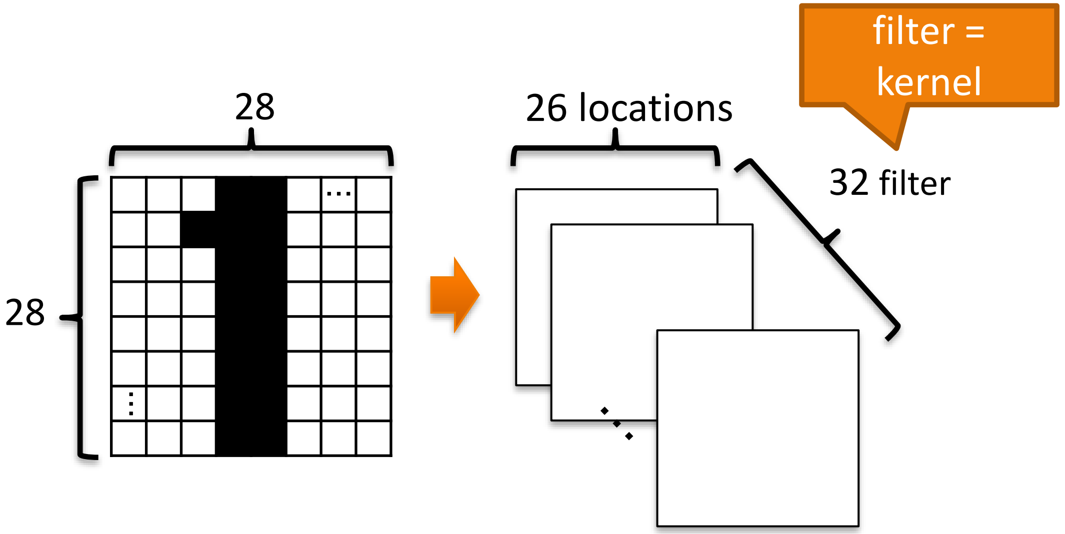
Deep learning : Neural network with two or more intermediate layers

Convolutional neural network : A method of learning by subdividing the intermediate layer



3-3. convolutional layer

```
layers.Conv2D(32, kernel_size=(3, 3), activation="relu"),
```



Input layer

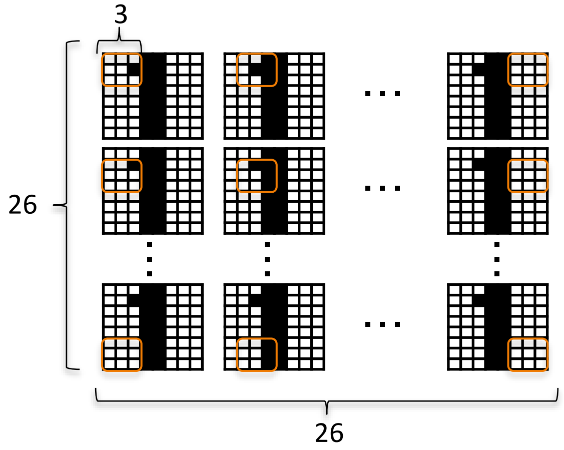
28*28 pixels,
28*28 = 784 neurons

Convolutional layer 1

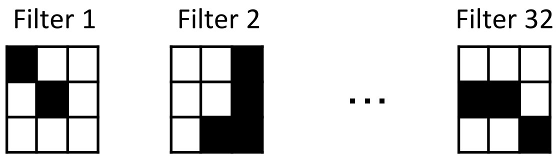
Feature extraction of 32 filters divided into 26*26 locations in 3*3 frames

Segment the entire 28*28 image with the same one filter (3*3) and extract features

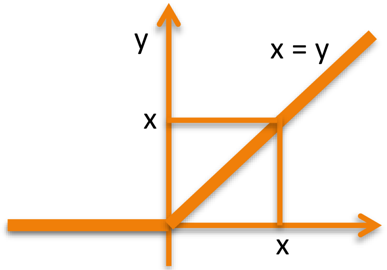
(Features are extracted by dividing into 26*26 sections)



Each of the 32 filters features separately



[Neuron output]
Activation function: Ramp function
(ReLU: Rectified Linear Unit)



output is any value from 0
The output magnitude is determined according to the input

By extracting features separately from the filter, the number of parameters to be optimized can be greatly reduced.

Efficient feature extraction

3-4. MAX pooling layer

```
layers.MaxPooling2D(pool_size=(2, 2)),
```

MaxPooling2D processing (pool_size = 2)

Extract the maximum value in the 2*2 frame

1	5	9	5
0	3	8	3
2	1	9	6
0	1	0	2

5	

1	5	9	5
0	3	8	3
2	1	9	6
0	1	0	2

	9

When implemented in all sections

Get the maximum value of each partition

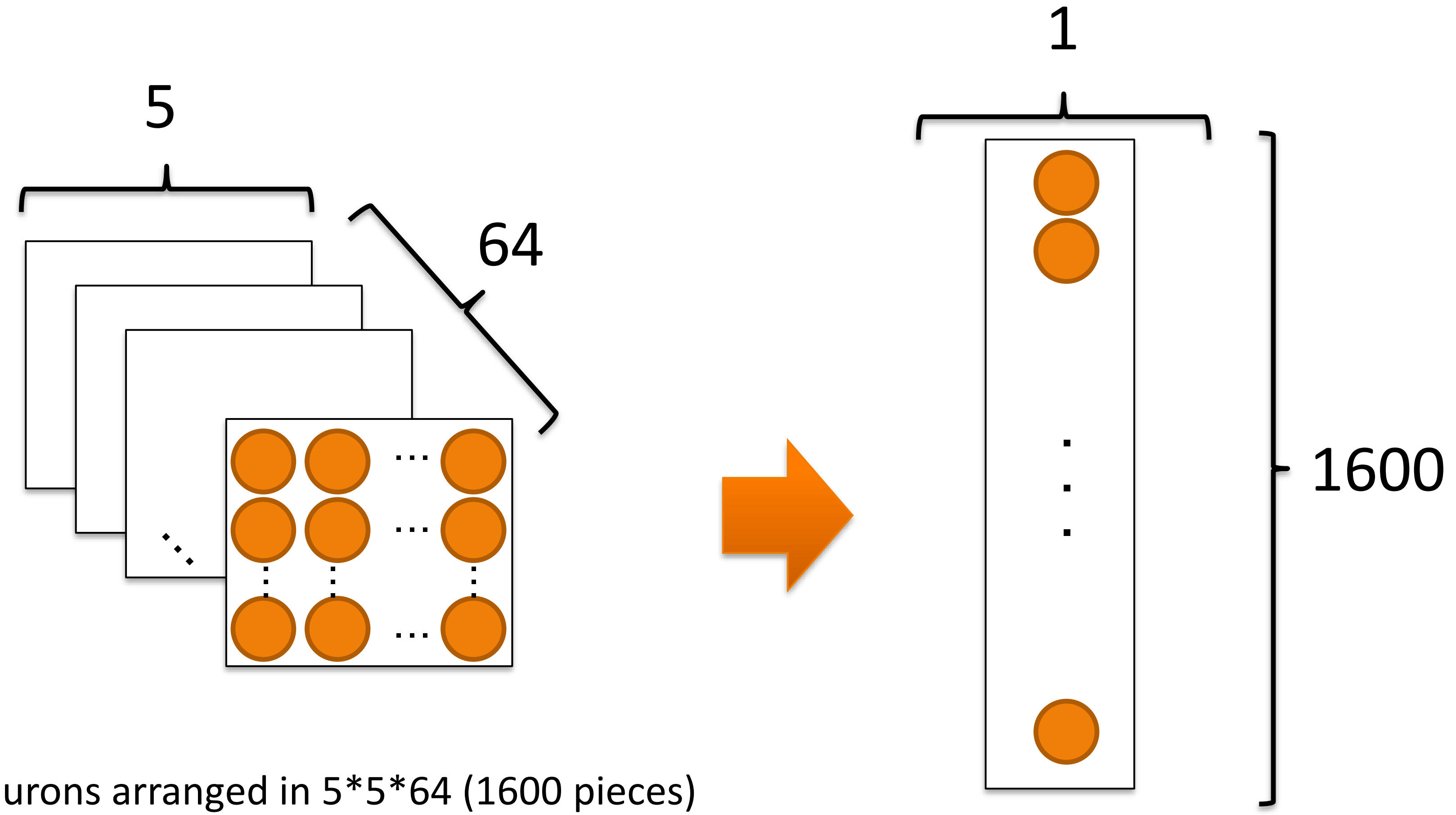
1	5	9	5
0	3	8	3
2	1	9	6
0	1	0	2

5	9
2	9

Capable of roughly capturing features and extracting features that are resistant to deviations, etc.

3-5. Flat layer

```
layers.Flatten(),
```



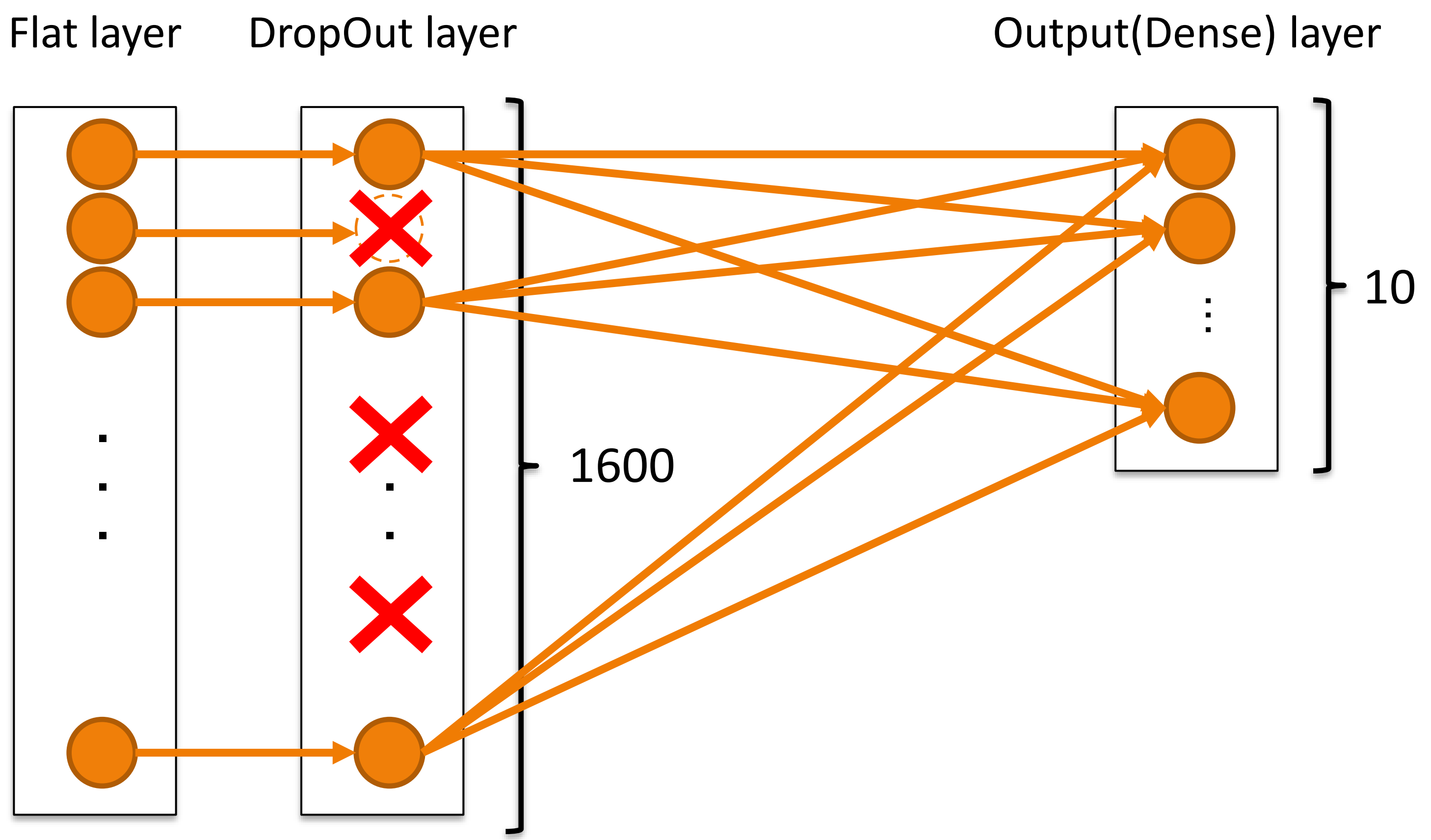
Neurons arranged in 5*5*64 (1600 pieces)

rearrange in one row
(Rearrange to 1D array)

The role of preparing data
for the next layer

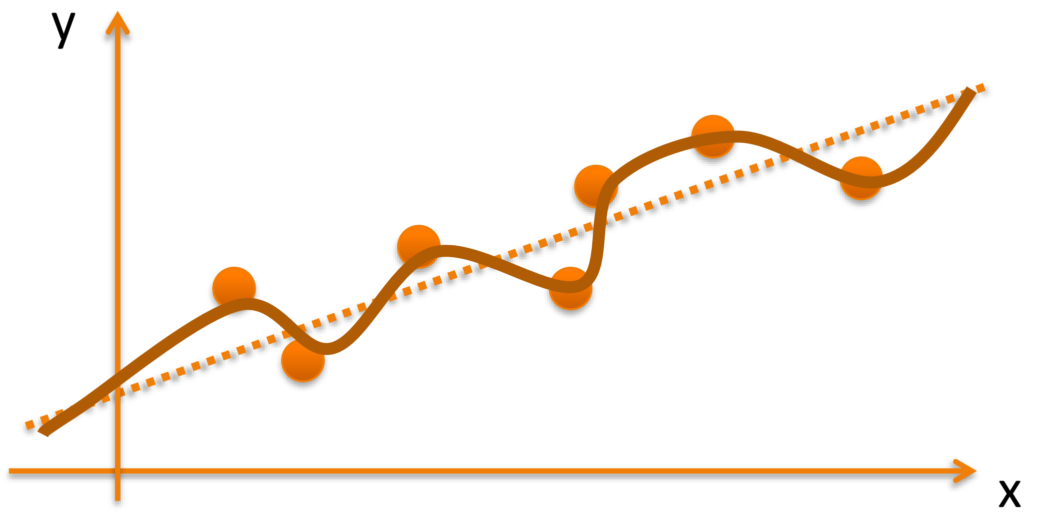
3-6. DropOut layer

```
layers.Dropout(0.5),
```



Effective in preventing overfitting

The desired relational expression is dotted line,
But Overfitting is a relational expression that
is too biased in the given data.



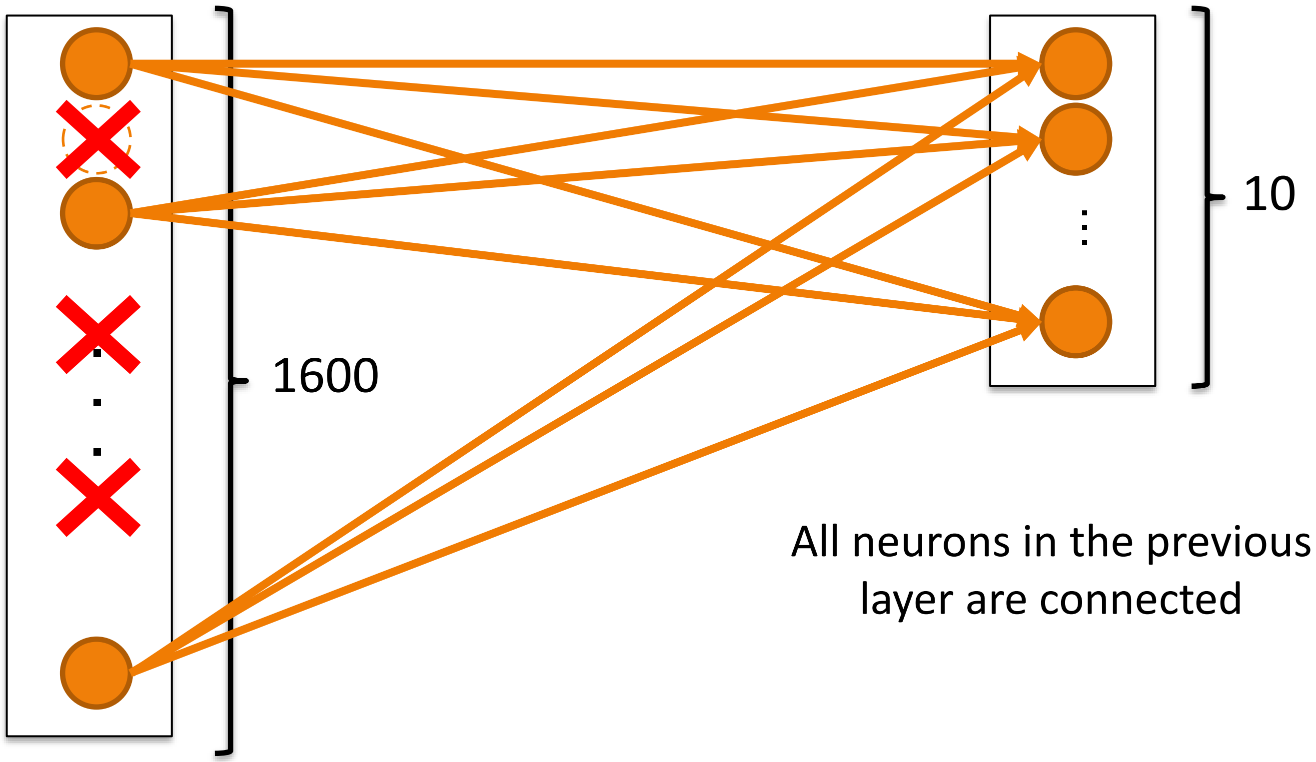
Randomly disables a certain percentage of neuron outputs.
This time, the exclusion rate is 0.5, so it will be deleted at a rate of 50%.

3-7. Dense layer (fully connected layer)

```
layers.Dense(10, activation="softmax"),
```

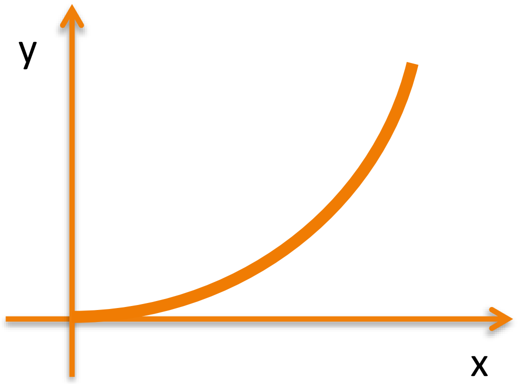
DropOut layer

Output(Dense) layer



All neurons in the previous layer are connected

[Neuron output]
activation function: softmax function



$$y1 = \frac{exp(x1)}{exp(x1) + exp(x2) \ldots exp(xn)}$$

$$y2 = \frac{exp(x2)}{exp(x1) + exp(x2) \ldots exp(xn)}$$

$$\ldots$$

$$yn = \frac{exp(xn)}{exp(x1) + exp(x2) \ldots exp(xn)}$$

A function whose output varies greatly as the input increases
Since it is a relational expression that indicates the proportion of each from the whole, the sum of each element of the output value (probability of the classification label) is 1 (100%).

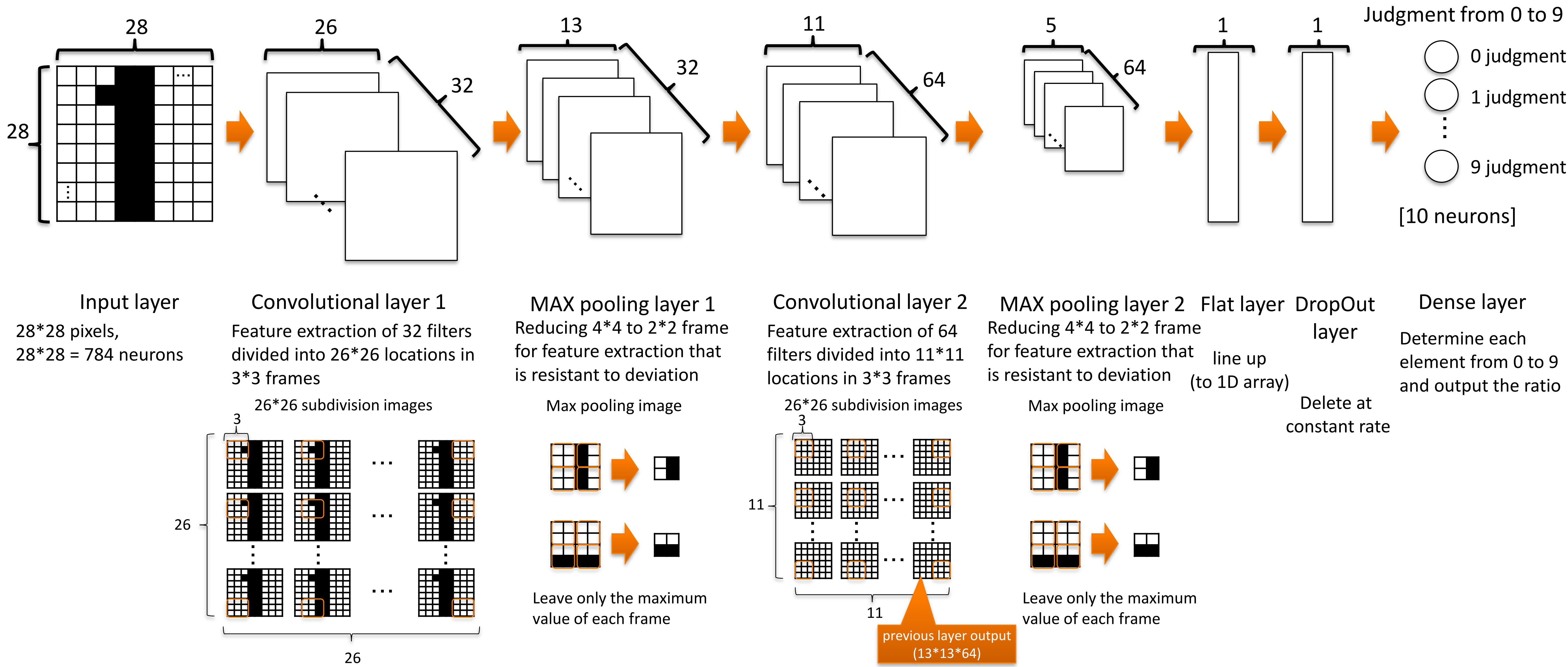
The softmax function with two elements (two classification labels) is the same as the sigmoid function.

Numerical judgment from 0 to 9 is performed

3-2. Convolutional neural network

Deep learning : Neural network with two or more intermediate layers

Convolutional neural network : A method of learning by subdividing the intermediate layer



3-8. AI learning (optimization of parameters [weights and thresholds])

```
model.compile(loss="categorical_crossentropy", optimizer="adam", metrics=["accuracy"])
```

`loss="categorical_crossentropy"`

Specifying a loss (objective) function.
(The loss function is the metric we try to minimize while training the model)
Commonly used loss functions in multiclass classification tasks

`optimizer="adam"`

adam (Adaptive Moment Estimation) is an optimization algorithm that is widely used as an improved version of gradient descent.
adam has features that automatically improve the adaptability of learning, such as adjusting the learning rate and using momentum.

`metrics=["accuracy"]`

The evaluation metric that is set. Evaluation metrics are used to evaluate model performance during and after training.
Accuracy rates for training and validation data are calculated and displayed during training.

<Reference> About how to solve equations

Transform the Formula

$$3x + 2y = 13 \quad \cdots (1)$$
$$5x + 3y = 21 \quad \cdots (2)$$

Transforming (1) gives

$$3x = 13 - 2y$$
$$x = (13 - 2y) \div 3$$

Substituting into (2) gives

$$(5 * (13 - 2y) \div 3) + 3y = 21$$
$$5 * (13 - 2y) + 9y = 63$$
$$65 - 10y + 9y = 63$$
$$y = 2$$

Substituting into (1) gives

$$3x + 2 * 2 = 13$$
$$x = 3$$

Can be obtained exactly, but only if the relational expression is obtained by transforming

Find by Substituting Values

$$3x + 2y = 13 \quad \cdots (1)$$
$$5x + 3y = 21 \quad \cdots (2)$$

If we substitute x=2 and y=2

$$3 * 2 + 2 * 2 = 10 < 13$$
$$5 * 2 + 3 * 2 = 16 < 21$$

Value is small, enter a larger value
Substituting x=3 and y=2, we get

$$3 * 3 + 2 * 2 = 13 = 13$$
$$5 * 3 + 3 * 2 = 21 = 21$$

Since the correct answers match, the correct answer is

$x=3$, $y=2$

The correct answer can be obtained for any complicated relational expression.

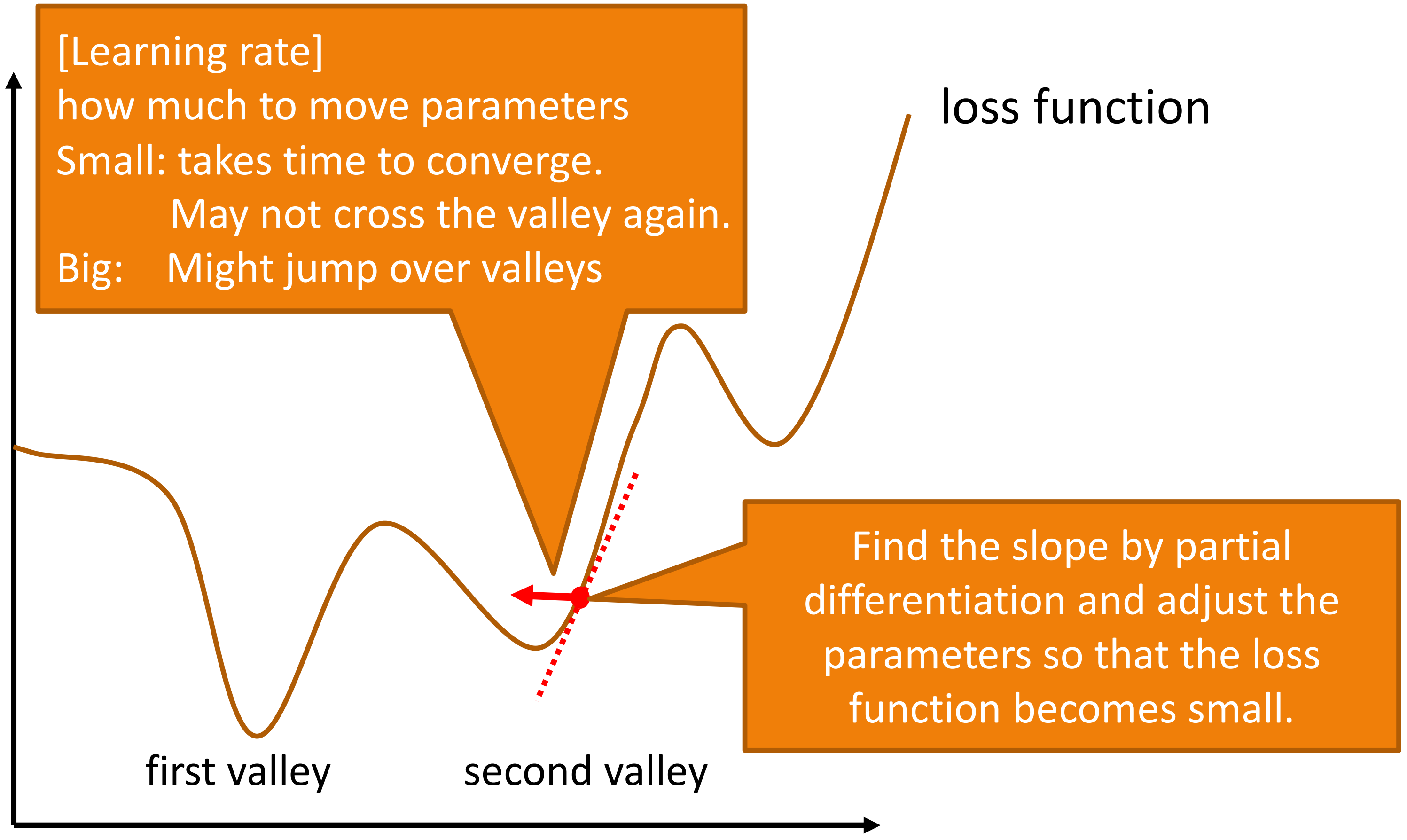
3-9. Gradient descent

Gradient Descent is a type of optimization algorithm used to find parameters that minimize a loss function.

The basic idea of gradient descent is to compute the gradient of the loss function (i.e. the derivative of the loss function with respect to each parameter) and reduce the value of the loss function by updating the parameters in the direction the gradient points.
Specifically, the procedure is as follows.

1. Set initial values for parameters.
2. Compute the gradient of the loss function. This is the partial derivative of the loss function with respect to each parameter.
3. Update the parameters using the gradient multiplied by the learning rate (usually a small positive value). This updates the parameters in the direction of decreasing values of the loss function.
4. Repeat steps 2 and 3 until convergence or until the specified number of epochs is reached.

Image of Gradient Descent



3-10. AI learning (optimization of parameters [weights and thresholds])

```
model.fit(x_train, y_train, batch_size=128, epochs=15, validation_split=0.1)
```

`validation_split=0.1`

10% is set for verification data after learning. Therefore, the training data is 90%. Since there are 60,000 MNIST data in total, 54,000 are used for learning.

`epochs=15`

An epoch is to learn the data all at once. This time, it will be one epoch if all 54000 image data are learned.
Since 15 is set, 54000 image data will be learned 15 times.

`batch_size=128`

Batch size is the number of data to learn at once. Batch means processing together.
Since 128 sheets are processed together, 54000 sheets require 422 batch processing (=54000/128). Therefore, one epoch is batch processing 422 times, and the processing status can be checked in batch units.

3-11. AI evaluation (after AI learning)

```
score = model.evaluate(x_test, y_test, verbose=0)
print("Test loss:", score[0])
print("Test accuracy:", score[1])
```

`model.evaluate(x_test, y_test, verbose=0)`

: Evaluate AI performance with 10,000 MNIST test data.(verbose is a learning status information display parameter. 0 is not displayed in particular. Set 1 to display a progress bar, etc.)

Loss : Loss function value on test data.
The parameters have been optimized (AI learns) so that the loss function becomes small.

Accuracy : Accuracy rate on test data. Percentage of correct judgments.