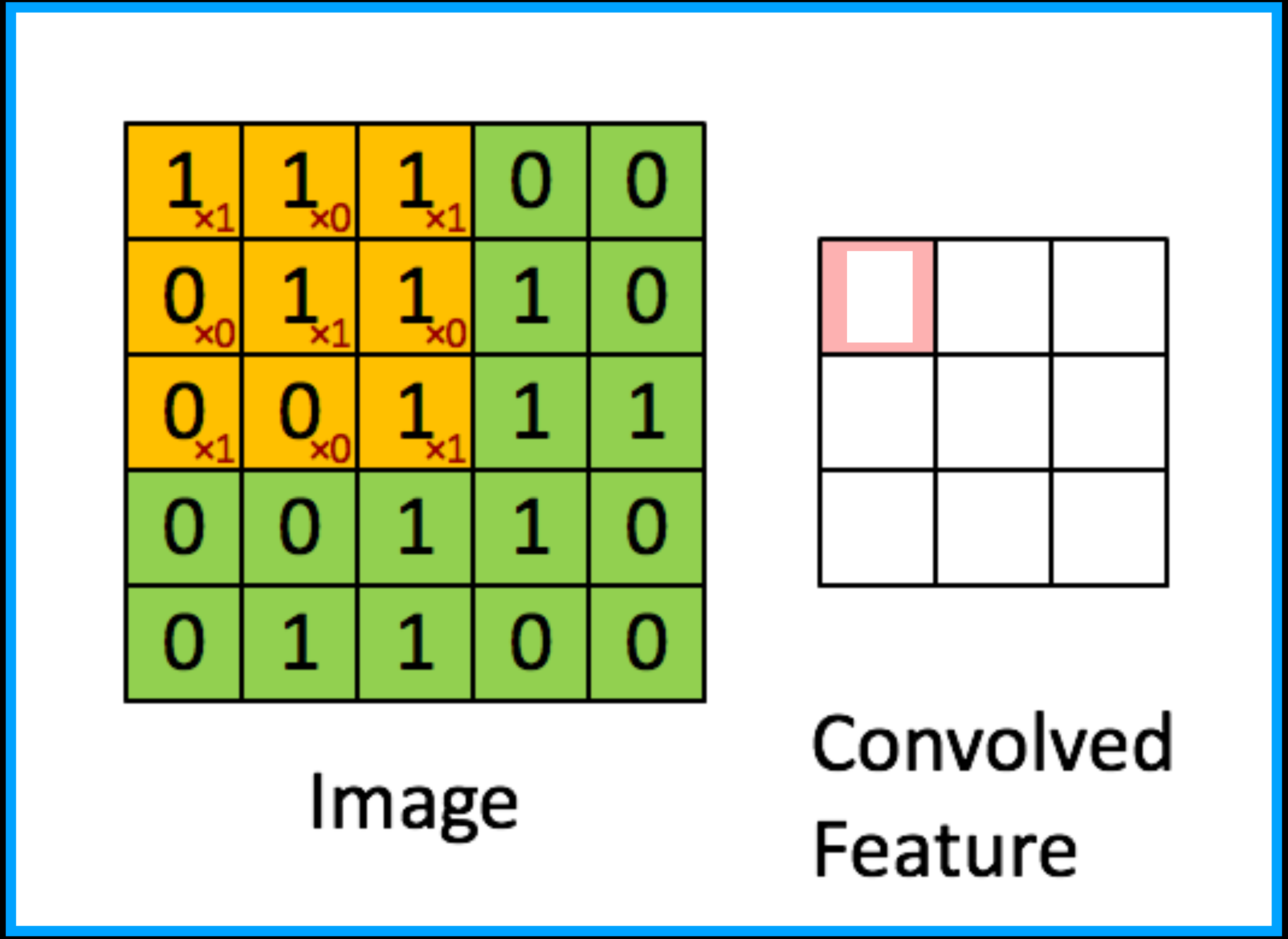
Convolution neural networks

Divyam Madaan

Objectives

- Building blocks of convolution neural networks
- Training and evaluation pipeline

Building blocks of a CNN



1 ×1	1 ×0	1 ×1	0	0
0 ×0	1 ×1	1 ×0	1	0
0 ×1	0 ×0	1 ×1	1	1
0	0	1	1	0
0	1	1	0	0

Convolution

Why Convolution?

Add your answers here

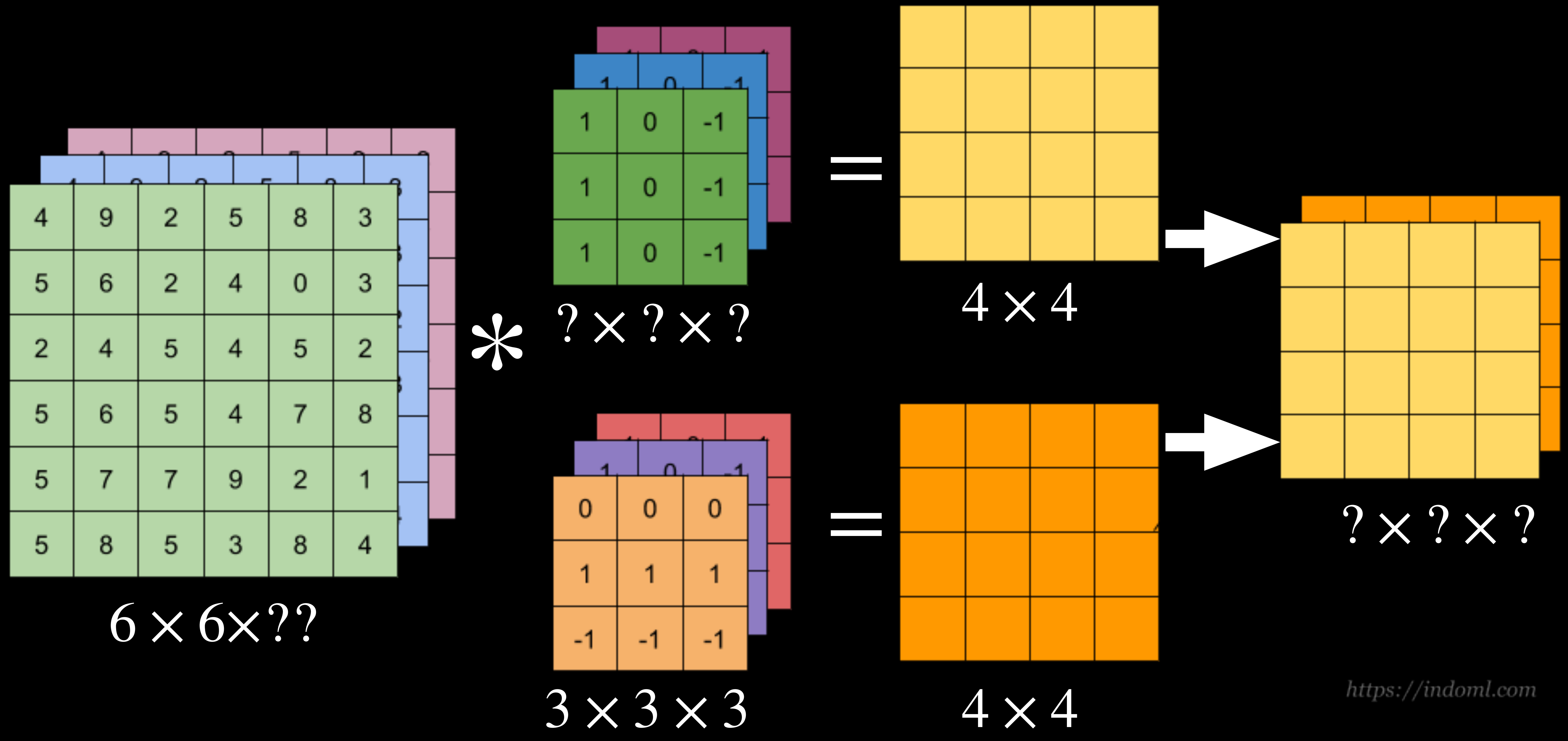
Definition

self.conv = nn.Conv2d(in_channels, out_channels, kernel_size, stride, padding)

Call

x = self.conv(input)

Building blocks of a CNN

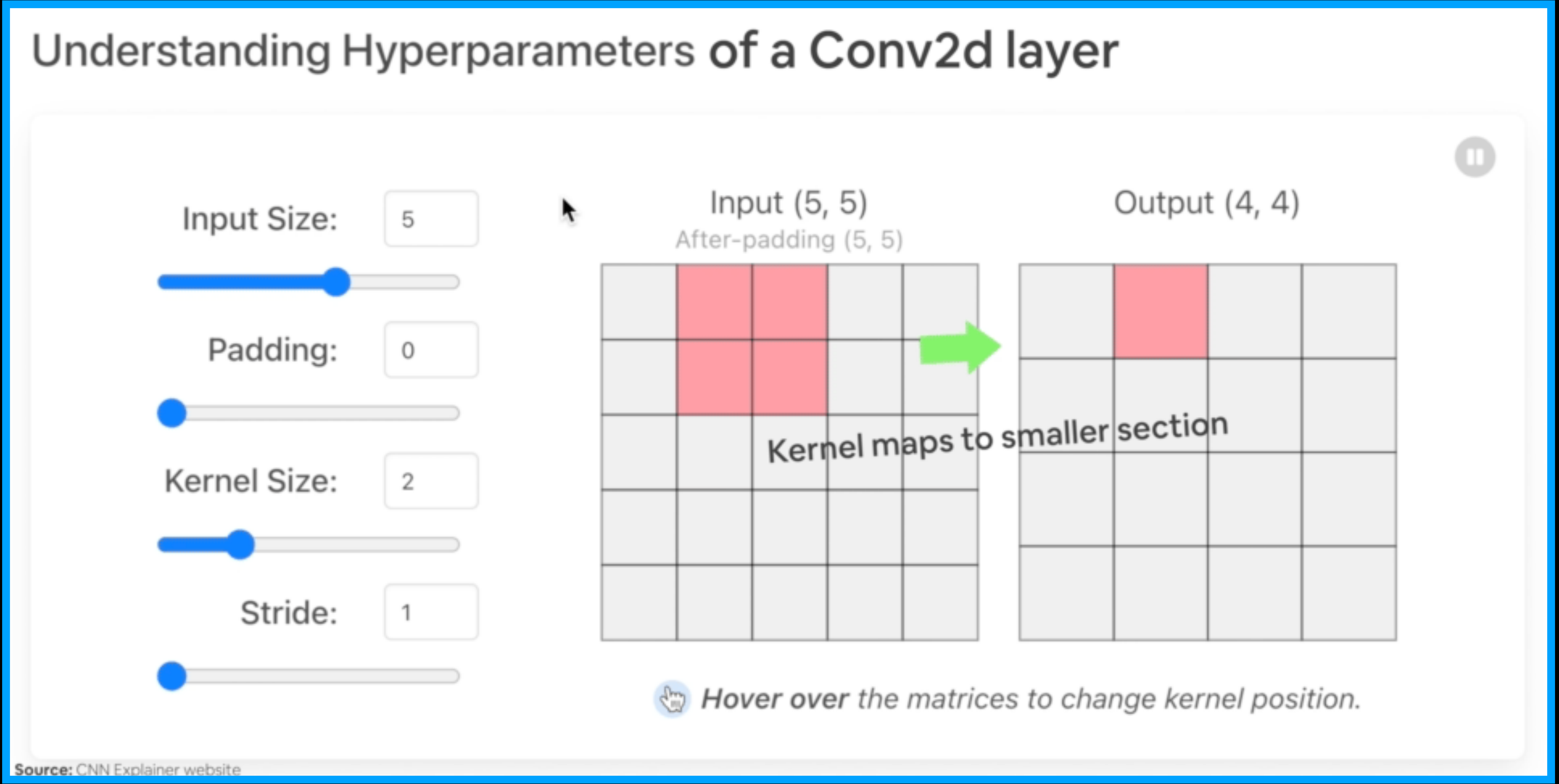


4	9	2	5	8	3
5	6	2	4	0	3
2	4	5	4	5	2
5	6	5	4	7	8
5	7	7	9	2	1
5	8	5	3	8	4

1	0	-1
1	0	-1
1	0	-1

0	0	0
1	1	1
-1	-1	-1

Building blocks of a CNN



Understanding Hyperparameters of a Conv2d layer

Input Size: 5

Padding: 0

Kernel Size: 2

Stride: 1

Hover over the matrices to change kernel position.

Source: CNN Explainer website

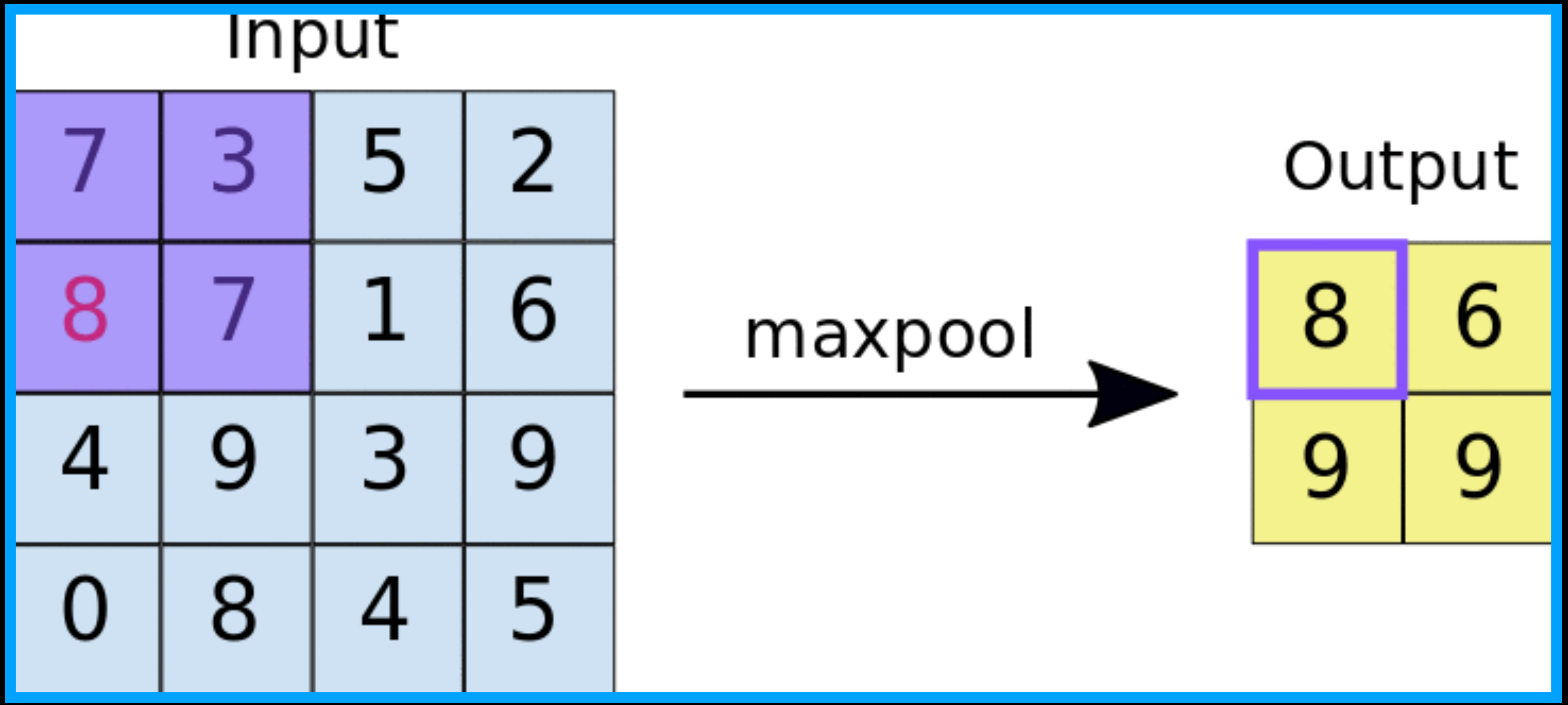
Definition

```
self.conv = nn.Conv2d(in_channels, out_channels, kernel_size, stride, padding)
```

Call

```
x = self.conv(input)
```

Building blocks of a CNN



7	3	5	2
8	7	1	6
4	9	3	9
0	8	4	5

8	6
9	9

Invariance

MaxPooling

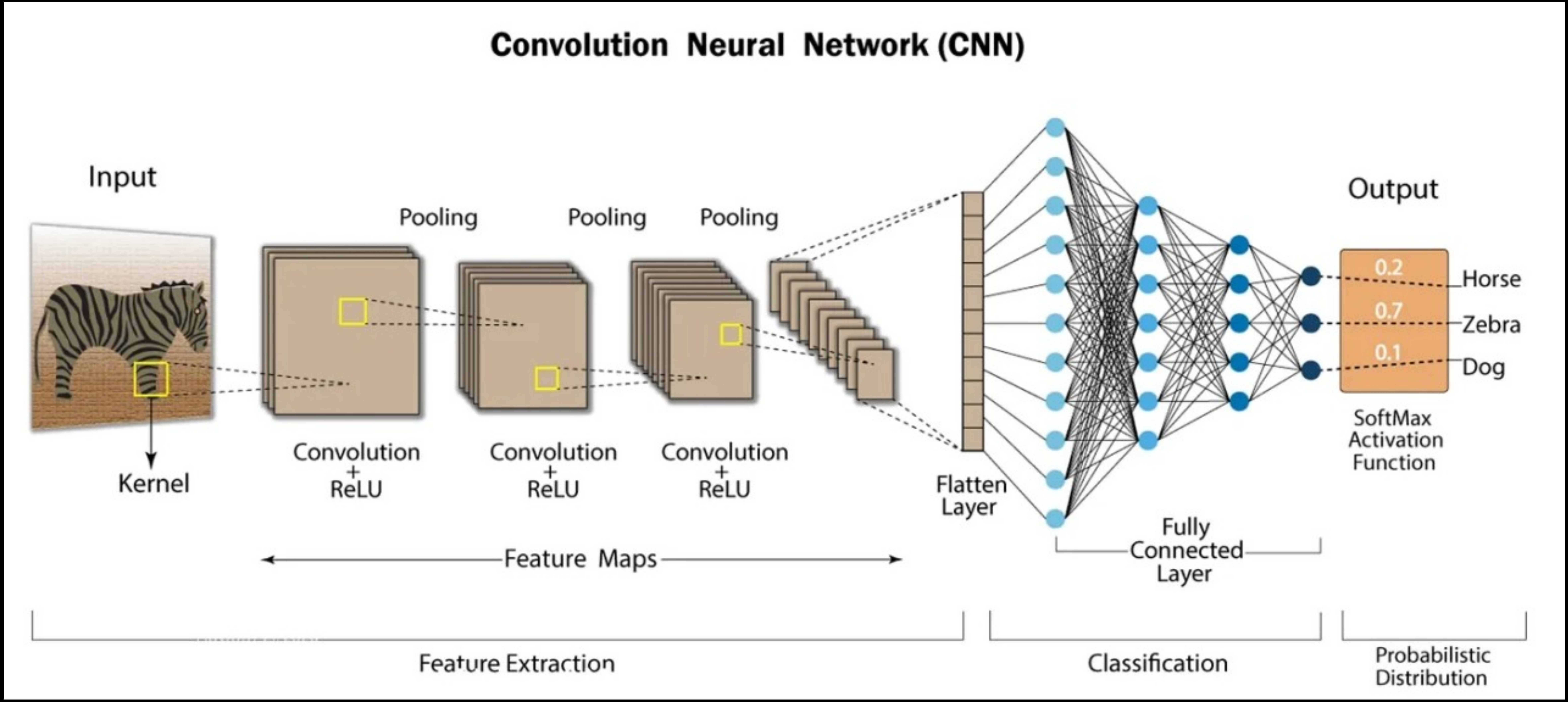
Definition

```
self.pool = nn.MaxPool2d(kernel_size, stride, padding)
```

Call

```
x = self.pool(F.relu(self.conv(input))
```

Convolution neural network



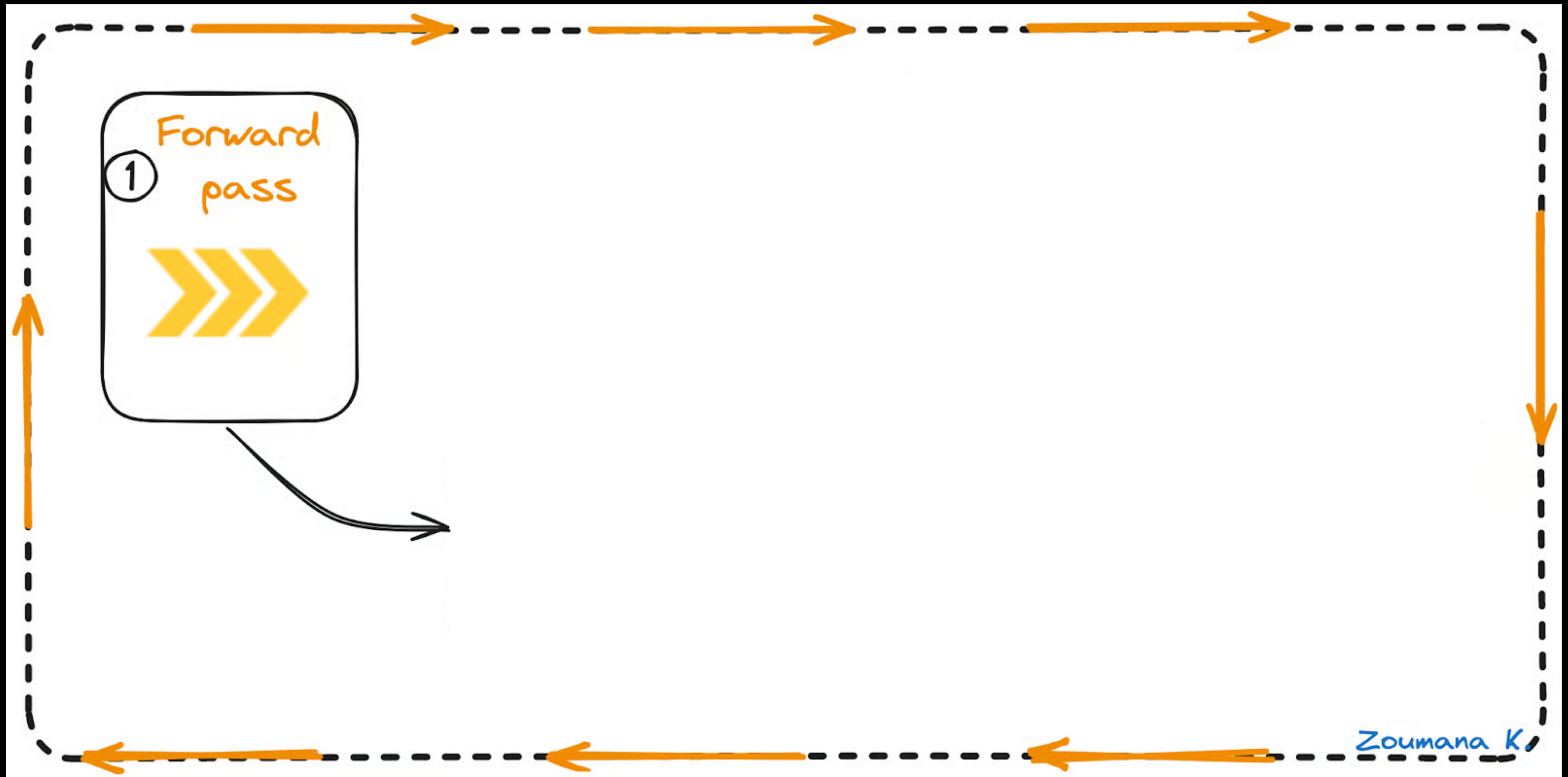
Objectives

- Building blocks of convolution neural networks
- Training and evaluation pipeline

Building a classifier

Step 1: Load your dataset

Step 2: Train your model

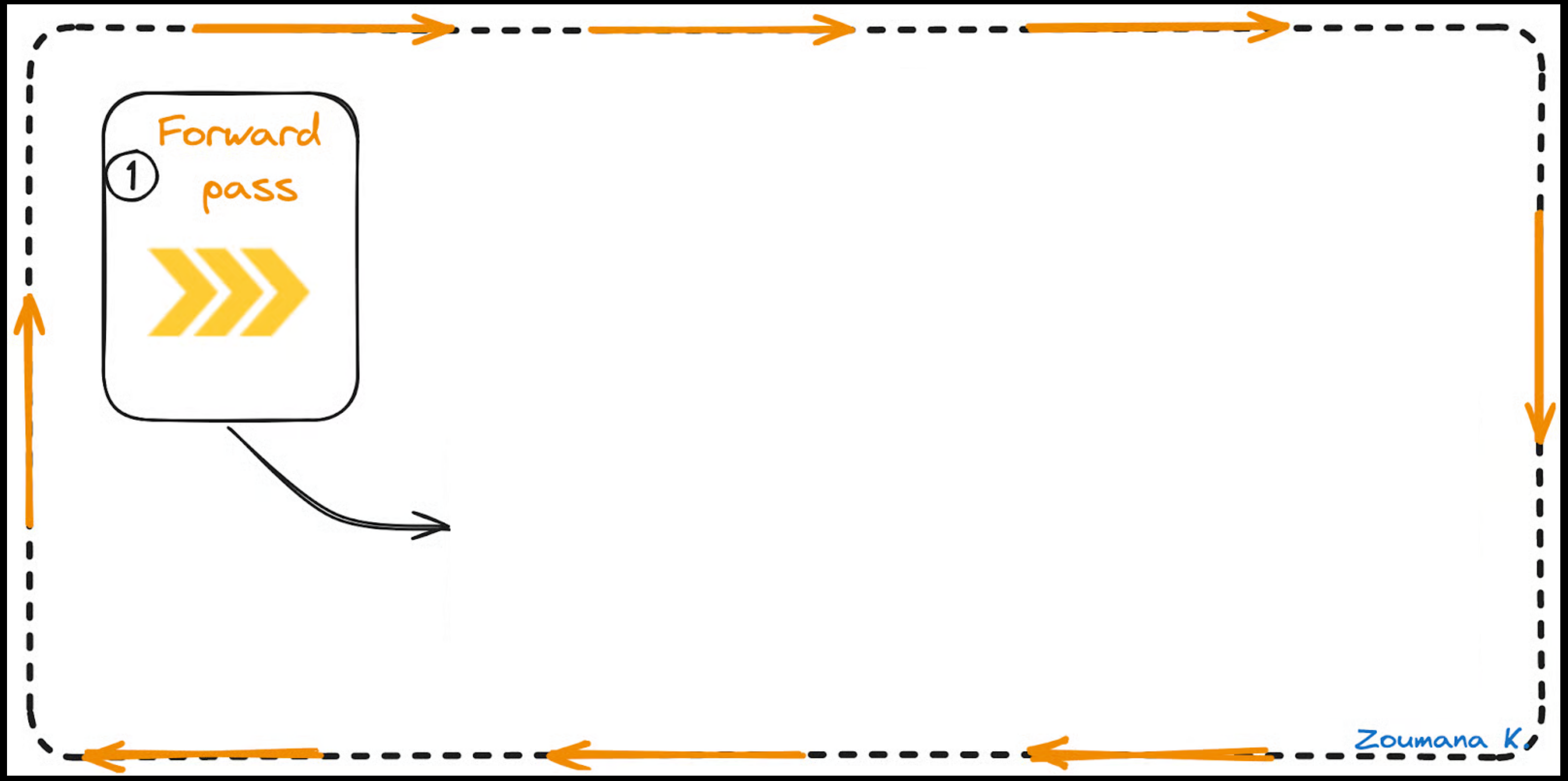


```
for epoch in range(epochs):
    for i, data in enumerate(train_loader):
        inputs, labels = data
        optimizer.zero_grad()
        outputs = model(inputs)
```

Building a classifier

Step 1: Load your dataset

Step 2: Train your model

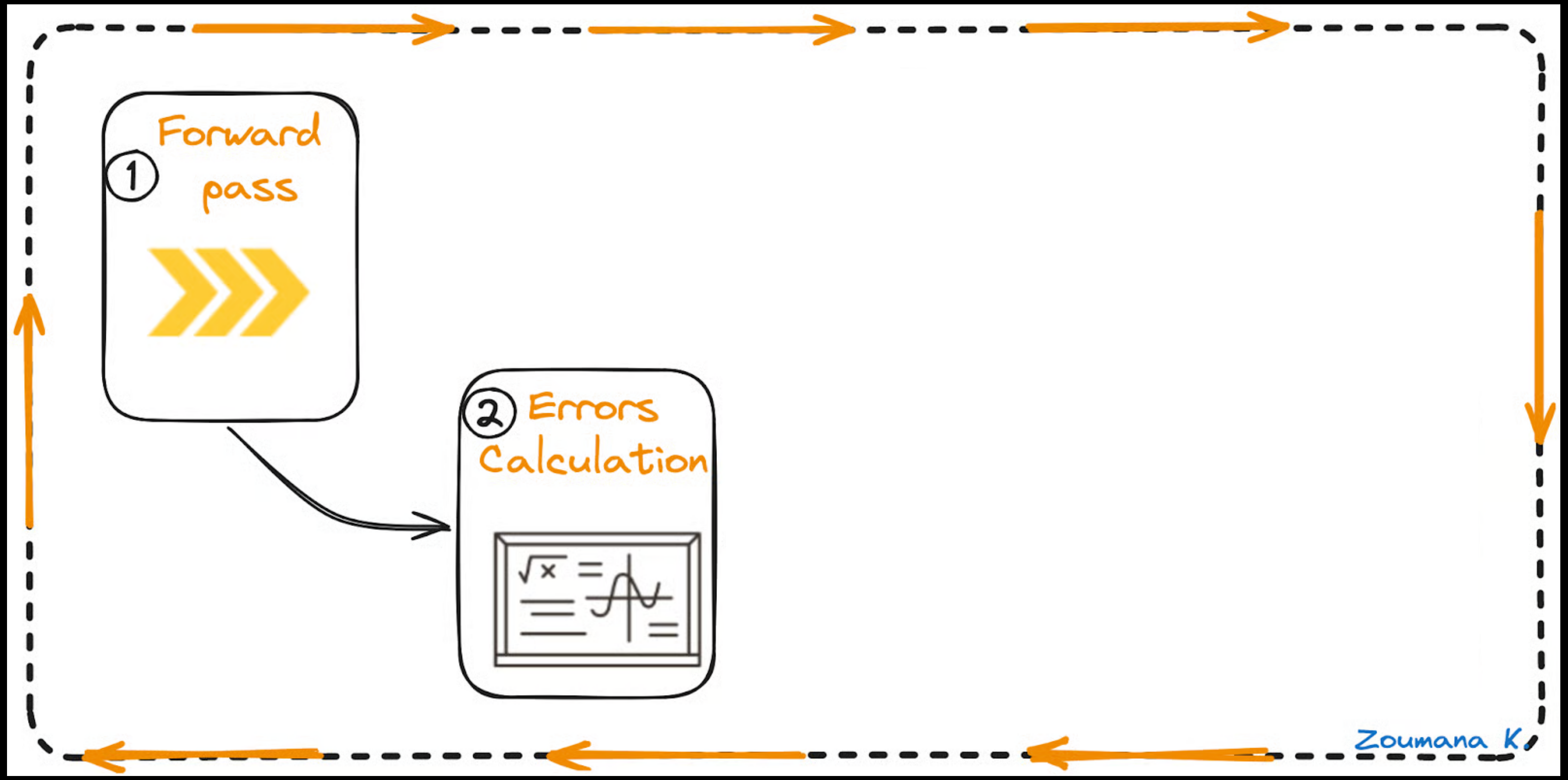


```
for epoch in range(epochs):
    for i, data in enumerate(train_loader):
        inputs, labels = data
        optimizer.zero_grad()
        outputs = model(inputs)
        loss = criterion(outputs, labels)
```

Building a classifier

Step 1: Load your dataset

Step 2: Train your model



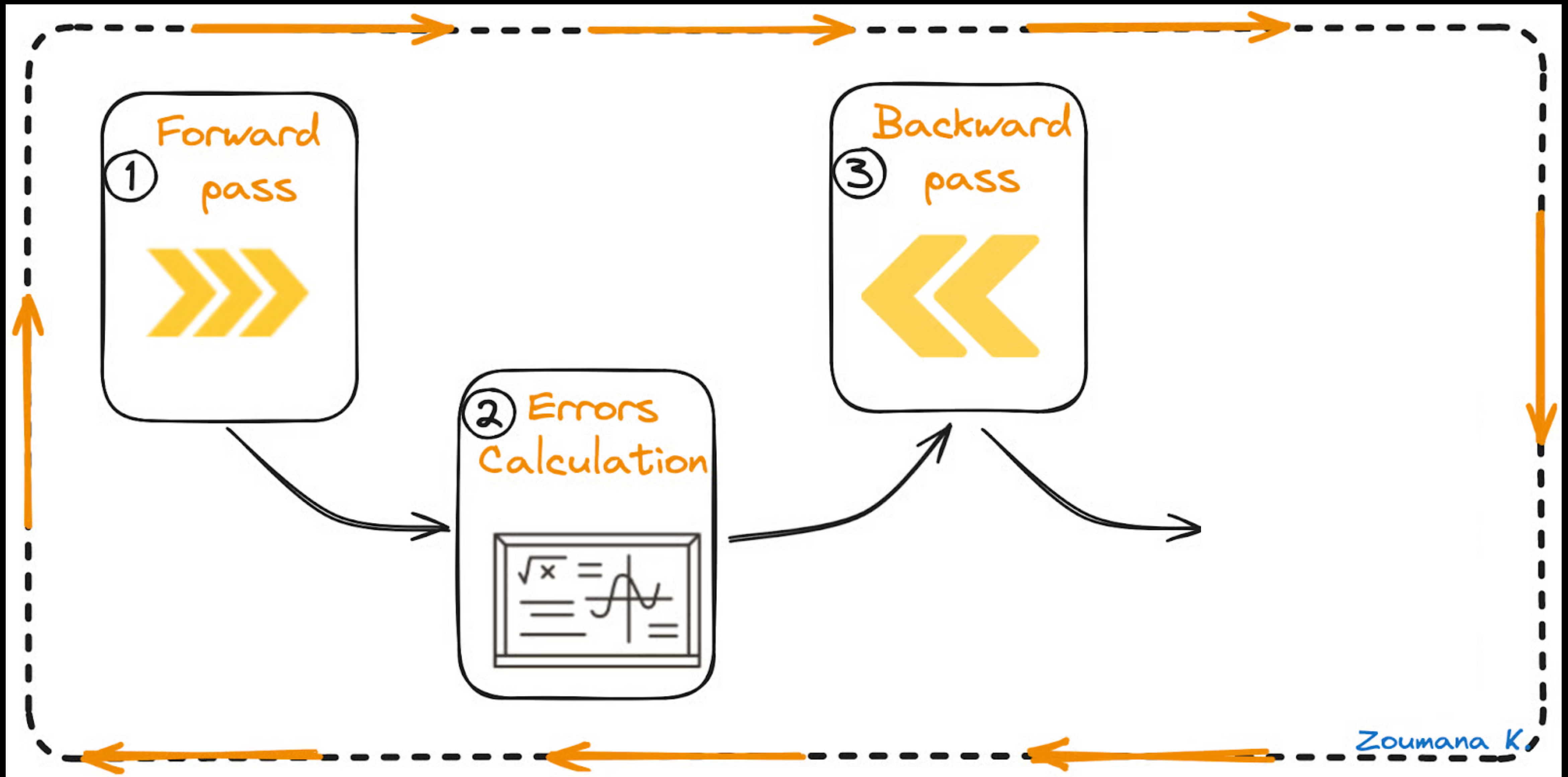
```
for epoch in range(epochs):
    for i, data in enumerate(train_loader):
        inputs, labels = data
        optimizer.zero_grad()
        outputs = model(inputs)
        loss = criterion(outputs, labels)
        loss.backward()
```

Building a classifier

Step 1: Load your dataset

Step 2: Train your model

Step 3: Evaluate your model



```
for epoch in range(epochs):
    for i, data in enumerate(train_loader):
        inputs, labels = data
        optimizer.zero_grad()
        outputs = model(inputs)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()
```