1.
필터의 경우 기존 데이터의 모서리 부분의 필터의 중앙에서 모서리까지 pixel 만큼의 둘레가 나라간다 만약 (a*a) 만큼의 데이터가 있었다면 위아래로 사라질 pixel 만큼의 둘레를 생성해 주면 기존의 데이터를 보존 가능하다.
(3,3)
생성된 padding pixel 양 : ((a+2)*(a+2)) - (a*a)

(5,5)
생성된 padding pixel 양 : ((a+4)*(a+4)) - (a*a)

7,7
생성된 padding pixel 양 : ((a+6)*(a+6)) - (a*a)

2.
필터가 적용될 때 최소한 1 pixel 이상이 겹치도록" 하는 Stride 의 최대값은 각 필터의 크기에서 1을 뺀 값이된다.
(3,3)
최대값 : (2 ,2)
(5,5)
최대값 : (4,4)
(7,7)
최대값 : (6,6)

3.

```
data = [[1, 0, 0, 1, 1, 0, 0, 1],
        [1, 0, 0, 1, 1, 0, 0, 1],
        [1, 0, 0, 1, 1, 0, 0, 1],
        [1, 0, 0, 1, 1, 0, 0, 1],
        [1, 0, 0, 1, 1, 0, 0, 1],
        [1, 0, 0, 1, 1, 0, 0, 1],
        [1, 0, 0, 1, 1, 0, 0, 1],
        [1, 0, 0, 1, 1, 0, 0, 1]]
data = asarray(data)
data = data.reshape(1, 8, 8, 1)

# 모델 생성
model = Sequential()
model.add(Conv2D(1, (3,3), input_shape=(8, 8, 1)))

# 모델 형상 (요약)
model.summary()

# 수직선 검출 필터를 위한 weight 설정
detector = [[[[0]],[[1]],[[0]]],
            [[[0]],[[1]],[[0]]],
            [[[0]],[[1]],[[0]]]]
weights = [asarray(detector), asarray([0.0])]
model.set_weights(weights)

# 입력 이미지에 필터 적용
yhat = model.predict(data)

for r in range(yhat.shape[1]):
    print([yhat[0,r,c,0] for c in range(yhat.shape[2])])
```

```
Model: "sequential_24"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_24 (Conv2D)           (None, 6, 6, 1)           10
=================================================================
Total params: 10
Trainable params: 10
Non-trainable params: 0
_________________________________________________________________
WARNING:tensorflow:7 out of the last 7 calls to <function Model.make_
[0.0, 0.0, 3.0, 3.0, 0.0, 0.0]
[0.0, 0.0, 3.0, 3.0, 0.0, 0.0]
[0.0, 0.0, 3.0, 3.0, 0.0, 0.0]
[0.0, 0.0, 3.0, 3.0, 0.0, 0.0]
[0.0, 0.0, 3.0, 3.0, 0.0, 0.0]
[0.0, 0.0, 3.0, 3.0, 0.0, 0.0]
```

```
[12]  data = [[1, 0, 0, 1, 1, 0, 0, 1],
              [1, 0, 0, 1, 1, 0, 0, 1],
              [1, 0, 0, 1, 1, 0, 0, 1],
              [1, 0, 0, 1, 1, 0, 0, 1],
              [1, 0, 0, 1, 1, 0, 0, 1],
              [1, 0, 0, 1, 1, 0, 0, 1],
              [1, 0, 0, 1, 1, 0, 0, 1],
              [1, 0, 0, 1, 1, 0, 0, 1]]

data = asarray(data)
data = data.reshape(1, 8, 8, 1)

# 모델 생성
model = Sequential()
model.add(Conv2D(1, (3,3), padding="same" , input_shape=(8, 8, 1)))

# 모델 형상 (요약)
model.summary()

# 수직선 검출 필터를 위한 weight 설정
detector = [[[[0]],[[1]],[[0]]],
            [[[0]],[[1]],[[0]]],
            [[[0]],[[1]],[[0]]]]
weights = [asarray(detector), asarray([0.0])]
model.set_weights(weights)

# 입력 이미지에 필터 적용
yhat = model.predict(data)

for r in range(yhat.shape[1]):
    print([yhat[0,r,c,0] for c in range(yhat.shape[2])])
```

```
Model: "sequential_23"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_23 (Conv2D)           (None, 8, 8, 1)           10
=================================================================
Total params: 10
Trainable params: 10
Non-trainable params: 0
_________________________________________________________________
WARNING:tensorflow:6 out of the last 6 calls to <function Model.make_predic
[2.0, 0.0, 0.0, 2.0, 2.0, 0.0, 0.0, 2.0]
[3.0, 0.0, 0.0, 3.0, 3.0, 0.0, 0.0, 3.0]
[3.0, 0.0, 0.0, 3.0, 3.0, 0.0, 0.0, 3.0]
[3.0, 0.0, 0.0, 3.0, 3.0, 0.0, 0.0, 3.0]
[3.0, 0.0, 0.0, 3.0, 3.0, 0.0, 0.0, 3.0]
[3.0, 0.0, 0.0, 3.0, 3.0, 0.0, 0.0, 3.0]
[3.0, 0.0, 0.0, 3.0, 3.0, 0.0, 0.0, 3.0]
[2.0, 0.0, 0.0, 2.0, 2.0, 0.0, 0.0, 2.0]
```

수직선 검출할 때, 일반 데이터에 padding이 적용하지 않는 경우 양 끝의 데이터가 소실되고 padding이 적용되는 경우 데이터가 소실되지 않고 정상적으로 수직선을 검출했다.