

MAX POOLING

Input

$$\begin{bmatrix} 6 & 3 & 2 \\ 5 & 2 & 1 \\ 4 & 1 & 0 \end{bmatrix}$$

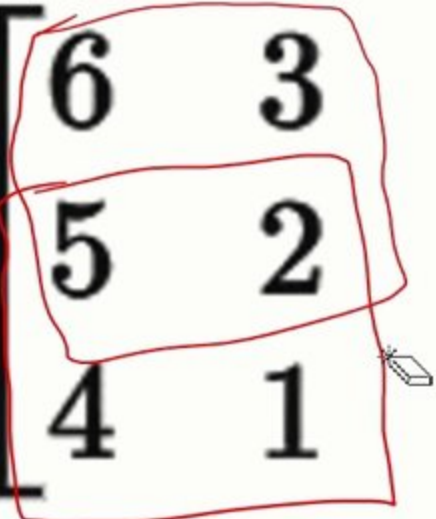
Pooling matrix
2x2

Output

$$\begin{bmatrix} 6 & 2 \\ 4 & 0 \end{bmatrix}$$

MAX POOLING

Input

$$\begin{bmatrix} 6 & 3 & 2 \\ 5 & 2 & 1 \\ 4 & 1 & 0 \end{bmatrix}$$


Pooling matrix
2x2

Output

$$\begin{bmatrix} 6 & 2 \\ 4 & 0 \end{bmatrix}$$

MAX POOLING

Input

$$\begin{bmatrix} 6 & 3 & 2 \\ 5 & 2 & 1 \\ 4 & 1 & 0 \end{bmatrix}$$

Pooling matrix
2x2

Output

$$\begin{bmatrix} 6 & 2 \\ 4 & 0 \end{bmatrix}$$

MAX POOLING

Input

$$\begin{bmatrix} 6 & 3 \\ 5 & 2 \\ 4 & 1 \\ & \end{bmatrix} \begin{matrix} 2 \\ 1 \\ 0 \end{matrix}$$

Pooling matrix
2x2

Output

$$\begin{bmatrix} 6 & 2 \\ 4 & 0 \end{bmatrix}$$

Pytorch.ipynb - Colab

Animations of Convolution and

Image Kernels explained visually

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=PB1e_6aXW3Cf

+ Code

+ Text

All changes saved

RAM

Disk

Paused

27s

[4]

from google.colab import drive

Mount Google Drive

drive.mount('/content/drive')

Mounted at /content/drive

0s

import numpy as np

import matplotlib.pyplot as plt

from PIL import Image

Function to perform convolution

def convolve2d(image, kernel):

kernel_height, kernel_width = kernel.shape

image_height, image_width = image.shape

output_height = image_height - kernel_height + 1

output_width = image_width - kernel_width + 1

output = np.zeros((output_height, output_width))

for i in range(output_height):

for j in range(output_width):

patch = image[i:i+kernel_height, j:j+kernel_width]

output[i, j] = np.sum(patch * kernel)

return output

1s

completed at 9:01 PM

AI SCIENCES

Odemy

Pytorch.ipynb - Colab

Animations of Convolution and

Image Kernels explained visually

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=PB1e_6aXW3Cf

+ Code

+ Text

All changes saved

✓

27s

[4]

drive.mount('/content/drive')

Mounted at /content/drive

import numpy as np

import matplotlib.pyplot as plt

from PIL import Image

Function to perform convolution

def convolve2d(image, kernel):

kernel_height, kernel_width = kernel.shape

image_height, image_width = image.shape

output_height = image_height - kernel_height + 1

output_width = image_width - kernel_width + 1

output = np.zeros((output_height, output_width))

for i in range(output_height):

for j in range(output_width):

patch = image[i:i+kernel_height, j:j+kernel_width]

output[i, j] = np.sum(patch * kernel)

return output

✓

2s

[5]

image_path = r"/content/drive/MyDrive/Dataset/images.jpeg"

1s

completed at 9:01 PM

ASCIENCES

Odemy

Pytorch.ipynb - Colab

Animations of Convolution and

Image Kernels explained visually

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=PB1e_6aXW3Cf

RAM

Disk

Paused

+ Code

+ Text

All changes saved

▶

```
output = np.zeros((output_height, output_width))

for i in range(output_height):
    for j in range(output_width):
        patch = image[i:i+kernel_height, j:j+kernel_width]
        output[i, j] = np.sum(patch * kernel)

return output

# Function to perform max pooling
def max_pooling(image, pool_size):
    pool_height, pool_width = pool_size
    image_height, image_width = image.shape

    output_height = image_height // pool_height
    output_width = image_width // pool_width

    output = np.zeros((output_height, output_width))

    for i in range(output_height):
        for j in range(output_width):
            patch = image[i*pool_height:(i+1)*pool_height, j*pool_width:(j+1)*pool_width]
            output[i, j] = np.max(patch)

    return output
```

1s

completed at 9:01 PM

ASCIENCES

Odemy

Pytorch.ipynb - Colab

Animations of Convolution and

Image Kernels explained visually

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=PB1e_6aXW3Cf

Paused

+ Code + Text All changes saved

Mounted at /content/drive

import numpy as np
import matplotlib.pyplot as plt
from PIL import Image

Function to perform convolution
def convolve2d(image, kernel):
 kernel_height, kernel_width = kernel.shape
 image_height, image_width = image.shape

 output_height = image_height - kernel_height + 1
 output_width = image_width - kernel_width + 1

 output = np.zeros((output_height, output_width))

 for i in range(output_height):
 for j in range(output_width):
 patch = image[i:i+kernel_height, j:j+kernel_width]
 output[i, j] = np.sum(patch * kernel)

 return output

Function to perform max pooling
def max_pooling(image, pool_size):
 pool_height, pool_width = pool_size
 image_height, image_width = image.shape

1s completed at 9:01 PM

AI SCIENCES

Odemy

同：

```
return output
```

✓ 1s completed at 9:01 PM



```
return output
```

```
def max_pooling(image, pool_size):
```

```
output_height = image_height // pool_height
output_width = image_width // pool_width
```

```
for i in range(output_height):
```

```
patch = image[i*pool_height:(i+1)*pool_height, j*pool_width:(j+1)*pool_width]
output[i, j] = np.max(patch)
```

```
return output
```

✓ 1s completed at 9:01 PM

✓ RAM



✓
2s

Pytorch.ipynb - Colab

Animations of Convolution and

Image Kernels explained visually

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=PB1e_6aXW3Cf

Paused

RAM

Disk

↑

↓

↻

💬

⚙️

📄

🗑️

⋮

+ Code + Text Saving...

0s

```
for i in range(output_height):
    for j in range(output_width):
        patch = image[i:i+kernel_height, j:j+kernel_width]
        output[i, j] = np.sum(patch * kernel)

    return output

# Function to perform max pooling
def max_pooling(image, pool_size):
    pool_height, pool_width = pool_size
    image_height, image_width = image.shape

    output_height = image_height // pool_height
    output_width = image_width // pool_width

    output = np.zeros((output_height, output_width))


    for i in range(output_height):
        for j in range(output_width):
            patch = image[i*pool_height:(i+1)*pool_height, j*pool_width:(j+1)*pool_width]
            output[i, j] = np.max(patch)


    return output
```

2s

[5] image_path = r"/content/drive/MyDrive/Dataset/images.jpeg"

completed at 9:18 PM





Pytorch.ipynb - Colab

Animations of Convolution and

Image Kernels explained visually

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=IC8-fpW_W2--

Paused

+ Code + Text

0s

image_path = r"/content/drive/MyDrive/Dataset/images.jpeg"

image = Image.open(image_path).convert('L') # Convert to grayscale

image = np.array(image)

image = np.random.rand(5, 5)

Define a kernel (e.g., edge detection)

kernel = np.array([

[1, 0, -1],

[1, 0, -1],

[1, 0, -1]

])

Apply convolution

convolved_image = convolve2d(image, kernel)

max_pooling(co)

Plot the results

fig, axes = plt.subplots(1, 2, figsize=(15, 5))

axes[0].imshow(image, cmap='gray')

axes[0].set_title('Original Image')

axes[1].imshow(convolved_image, cmap='gray')

axes[1].set_title('Convolved Image')

Text(0.5, 1.0, 'Convolved Image')

RAM

Disk

↑

↓

↻

💬

⚙️

📄

🗑️

⋮

0s

completed at 9:18 PM

ΔSCIENCES

Odemy

Pytorch.ipynb - Colab

Animations of Convolution and

Image Kernels explained visually

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=IC8-fpW_W2--

Paused

+ Code + Text

0s

image_path = r"/content/drive/MyDrive/Dataset/images.jpeg"

image = Image.open(image_path).convert('L') # Convert to grayscale

image = np.array(image)

image = np.random.rand(5, 5)

Define a kernel (e.g., edge detection)

kernel = np.array([

[1, 0, -1],

[1, 0, -1],

[1, 0, -1]

])

Apply convolution

convolved_image = convolve2d(image, kernel)

max_pooling(convolved_image, (2, 2))

Plot the results

fig, axes = plt.subplots(1, 2, figsize=(15, 5))

axes[0].imshow(image, cmap='gray')

axes[0].set_title('Original Image')

axes[1].imshow(convolved_image, cmap='gray')

axes[1].set_title('Convolved Image')

Text(0.5, 1.0, 'Convolved Image')

RAM

Disk

↑

↓

↺

💬

⚙️

📄

🗑️

⋮

0s

completed at 9:18 PM

ΔSCIENCES

Odemy



```
# Plot the results
fig, axes = plt.subplots(1, 2, figsize=(15, 5))
axes[0].imshow(image, cmap='gray')
axes[0].set_title('Original Image')
axes[1].imshow(convolved_image, cmap='gray')
axes[1].set_title('Convolved Image')
```

[↑]



Pytorch.ipynb - Colab

Animations of Convolution and

Image Kernels explained visually

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=IC8-fpW_W2--

+ Code

+ Text

All changes saved

▶

Define a kernel (e.g., edge detection)

kernel = np.array([

[1, 0, -1],

[1, 0, -1],

[1, 0, -1]

])

Apply convolution

convolved_image = convolve2d(image, kernel)

max_pooling_img = max_pooling(convolved_image, (2, 2))

Plot the results

fig, axes = plt.subplots(1, 2, figsize=(15, 5))

axes[0].imshow(image, cmap='gray')

axes[0].set_title('Original Image')

axes[1].imshow(convolved_image, cmap='gray')

axes[1].set_title('Convolved Image')

axes[1].imshow(convolved_image, cmap='gray')

axes[1].set_title('Convolved Image')

Text(0.5, 1.0, 'Convolved Image')

Original Image

Convolved Image

0s

completed at 9:18 PM

AI SCIENCES

Odemy

Pytorch.ipynb - Colab

Animations of Convolution and

Image Kernels explained visually

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=IC8-fpW_W2--

+ Code

+ Text

All changes saved

▶

Define a kernel (e.g., edge detection)

kernel = np.array([

[1, 0, -1],

[1, 0, -1],

[1, 0, -1]

])

Apply convolution

convolved_image = convolve2d(image, kernel)

max_pooling_img = max_pooling(convolved_image, (2, 2))

Plot the results

fig, axes = plt.subplots(1, 2, figsize=(15, 5))

axes[0].imshow(image, cmap='gray')

axes[0].set_title('Original Image')

axes[1].imshow(convolved_image, cmap='gray')

axes[1].set_title('Convolved Image')

axes[1].imshow(convolved_image, cmap='gray')

axes[1].set_title('Max Pooling Image')

Text(0.5, 1.0, 'Convolved Image')

Original Image

Convolved Image

0s

completed at 9:18 PM

ASCIENCES

Odemy

Pytorch.ipynb - Colab

Animations of Convolution and

Image Kernels explained visually

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=IC8-fpW_W2--

+ Code

+ Text

All changes saved

▶

Define a kernel (e.g., edge detection)

kernel = np.array([

[1, 0, -1],

[1, 0, -1],

[1, 0, -1]

])

Apply convolution

convolved_image = convolve2d(image, kernel)

max_pooling_img = max_pooling(convolved_image, (2, 2))

Plot the results

fig, axes = plt.subplots(1, 2, figsize=(15, 5))

axes[0].imshow(image, cmap='gray')

axes[0].set_title('Original Image')

axes[1].imshow(convolved_image, cmap='gray')

axes[1].set_title('Convolved Image')

axes[1].imshow(convolved_image, cmap='gray')

axes[1].set_title('Max Pooled Image')

Text(0.5, 1.0, 'Convolved Image')

Original Image

Convolved Image

0s

completed at 9:18 PM

AI SCIENCES

Odemy

Pytorch.ipynb - Colab

Animations of Convolution and

Image Kernels explained visually

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=IC8-fpW_W2--

+

Code

+

Text

All changes saved

RAM

Disk

Paused

↑

↓

↻

💬

⚙️

📄

🗑️

⋮

▶

```
# Define a kernel (e.g., edge detection)
kernel = np.array([
    [1, 0, -1],
    [1, 0, -1],
    [1, 0, -1]
])

# Apply convolution
convolved_image = convolve2d(image, kernel)

max_pooling_img = max_pooling(convolved_image, (2, 2))

# Plot the results
fig, axes = plt.subplots(1, 2, figsize=(15, 5))
axes[0].imshow(image, cmap='gray')
axes[0].set_title('Original Image')
axes[1].imshow(convolved_image, cmap='gray')
axes[1].set_title('Convolved Image')
axes[1].imshow(max_pooling_img, cmap='gray')
axes[1].set_title('Max Pooled Image')
```

↻

Text(0.5, 1.0, 'Convolved Image')

Original Image

Convolved Image

0

25

0

25

0s

completed at 9:18 PM

AI SCIENCES

Odemy

```
# Define a kernel (e.g., edge detection)
kernel = np.array([
    [1, 0, -1],
    [1, 0, -1],
    [1, 0, -1]
])

# Apply convolution
convolved_image = convolve2d(image, kernel)

max_pooling_img = max_pooling(convolved_image, (2, 2))

# Plot the results
fig, axes = plt.subplots(1, 3, figsize=(15, 5))
axes[0].imshow(image, cmap='gray')
axes[0].set_title('Original Image')
axes[1].imshow(convolved_image, cmap='gray')
axes[1].set_title('Convolved Image')
axes[1].imshow(max_pooling_img, cmap='gray')
axes[1].set_title('Max Pooled Image')
```



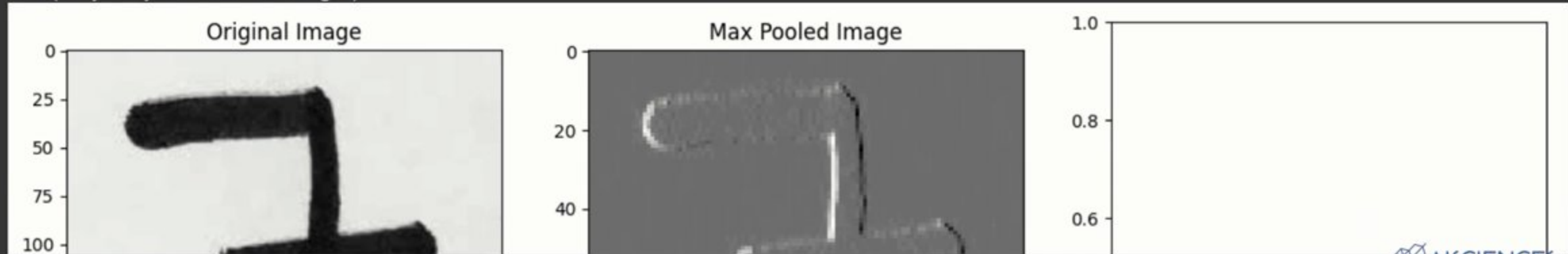
```
[1, 0, -1]
])

# Apply convolution
convolved_image = convolve2d(image, kernel)

max_pooling_img = max_pooling(convolved_image, (2, 2))

# Plot the results
fig, axes = plt.subplots(1, 3, figsize=(15, 5))
axes[0].imshow(image, cmap='gray')
axes[0].set_title('Original Image')
axes[1].imshow(convolved_image, cmap='gray')
axes[1].set_title('Convolved Image')
axes[2].imshow(max_pooling_img, cmap='gray')
axes[2].set_title('Max Pooled Image')
```

Text(0.5, 1.0, 'Max Pooled Image')



1s completed at 9:21 PM

Pytorch.ipynb - Colab

Animations of Convolution and

Image Kernels explained visually

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=IC8-fpW_W2--

☆

Paused

+ Code + Text

```
[1, 0, -1]
])

# Apply convolution
convolved_image = convolve2d(image, kernel)

max_pooling_img = max_pooling(convolved_image, (2, 2))

# Plot the results
fig, axes = plt.subplots(1, 3, figsize=(15, 5))
axes[0].imshow(image, cmap='gray')
axes[0].set_title('Original Image')
axes[1].imshow(convolved_image, cmap='gray')
axes[1].set_title('Convolved Image')
axes[2].imshow(max_pooling_img, cmap='gray')
axes[2].set_title('Max Pooled Image')
```

...

✓ 1s completed at 9:21 PM

ΔSCIENCES

Odemy

Pytorch.ipynb - Colab

Animations of Convolution and

Image Kernels explained visual

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=IC8-fpW_W2--

RAM

Disk

Paused

+ Code + Text

1s

```
fig, axes = plt.subplots(1, 3, figsize=(15, 5))
axes[0].imshow(image, cmap='gray')
axes[0].set_title('Original Image')
axes[1].imshow(convolved_image, cmap='gray')
axes[1].set_title('Convolved Image')
axes[2].imshow(max_pooling_img, cmap='gray')
axes[2].set_title('Max Pooled Image')

Text(0.5, 1.0, 'Max Pooled Image')
```

Original Image

Convolved Image

Max Pooled Image

1s

completed at 9:22 PM

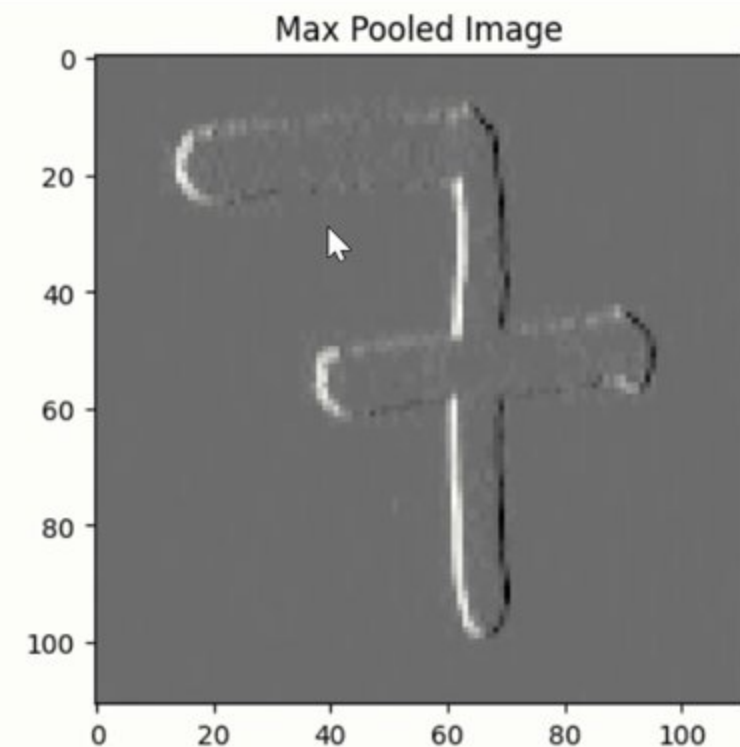
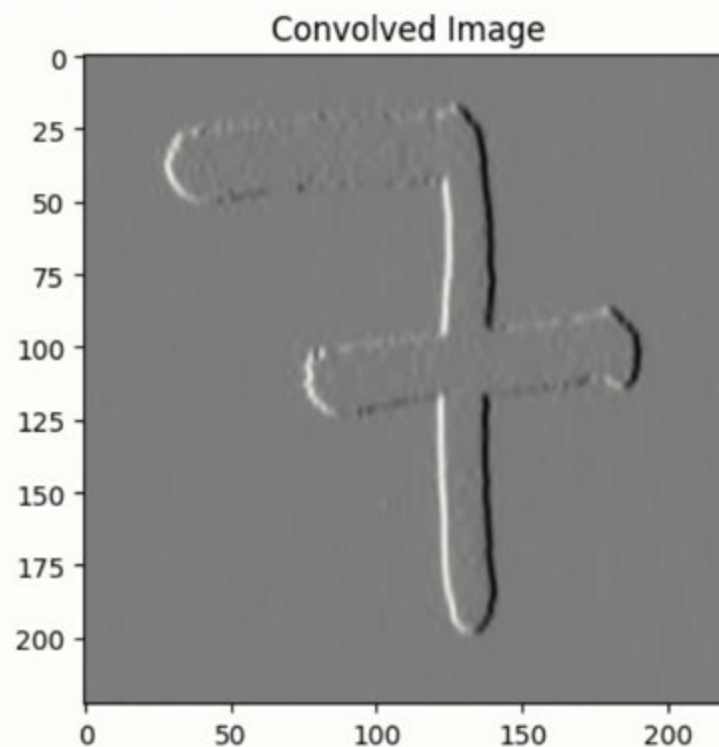
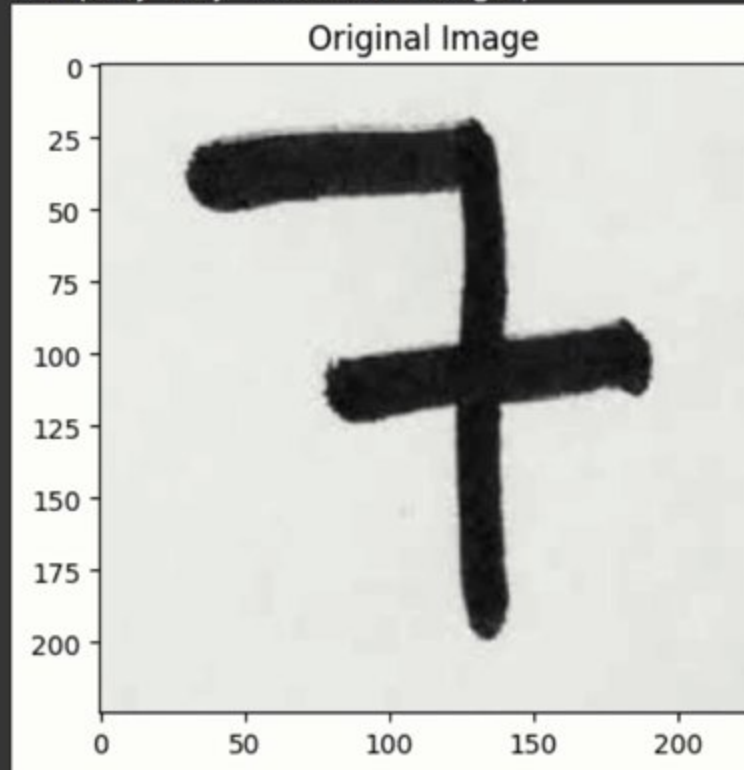
AI SCIENCES

Odemy

+ Code + Text All changes saved

```
axes[1].imshow(convolved_image, cmap='gray')
axes[1].set_title('Convolved Image')
axes[2].imshow(max_pooling_img, cmap='gray')
axes[2].set_title('Max Pooled Image')
```

```
➡ Text(0.5, 1.0, 'Max Pooled Image')
```



CODES & DATA ARE AVAILABLE AT
WWW.AISCIENCES.ACADEMY/COURSE-PYTORCH

✓ 1s completed at 9:22 PM