Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LClW23n

Paused

+ Code + Text All changes saved

RAM Disk

```
        return output
```

[1]

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)

# 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')
```

2s

```
Text(0.5, 1.0, 'Convolved Image')
```

0s completed at 8:56 PM

AISCIENCES

Udemy

Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

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

Paused

+ Code + Text All changes saved

RAM Disk

```
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
```

```
[5] 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)
```

0s completed at 8:56 PM

AISCIENCES

Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

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

Paused

+ Code + Text All changes saved

RAM Disk

Convolution using Pytorch

1s

```
import torch
import torch.nn as nn
from PIL import Image
import numpy as np
import matplotlib.pyplot as plt

# Load and preprocess the image
def load_image(image_path, image_size=(128, 128)):
    image = Image.open(image_path).convert('L')  # Convert to grayscale
    # image = image.resize(image_size)             # Resize the image
    image = np.array(image)                      # Convert to numpy array
    image = torch.tensor(image, dtype=torch.float32)  # Convert to torch tensor
    image = image.unsqueeze(0).unsqueeze(0)      # Add batch and channel dimensions
    return image

# Path to the image
image_path = r"/content/drive/MyDrive/Dataset/images.jpeg"

# Load the image
input_image = load_image(image_path)
print("Input Image Shape:", input_image.shape)
```

0s completed at 8:56 PM

Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LClW23n

Paused

+ Code + Text All changes saved

RAM Disk

```
import torch.nn as nn
from PIL import Image
import numpy as np
import matplotlib.pyplot as plt

# Load and preprocess the image
def load_image(image_path, image_size=(128, 128)):
    image = Image.open(image_path).convert('L')  # Convert to grayscale
    # image = image.resize(image_size)             # Resize the image
    image = np.array(image)                      # Convert to numpy array
    image = torch.tensor(image, dtype=torch.float32)  # Convert to torch tensor
    image = image.unsqueeze(0).unsqueeze(0)      # Add batch and channel dimensions
    return image

# Path to the image
image_path = r"/content/drive/MyDrive/Dataset/images.jpeg"

# Load the image
input_image = load_image(image_path)
print("Input Image Shape:", input_image.shape)


input_image = torch.tensor(input_image, dtype=torch.float32)

# Define a convolutional layer
conv_layer = nn.Conv2d(in_channels=1, out_channels=1, kernel_size=3, stride=1, padding=0)

# Set a specific filter for the convolutional layer
```

0s completed at 8:56 PM

Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LClW23n

Paused

+ Code + Text All changes saved

RAM Disk

```
import matplotlib.pyplot as plt

# Load and preprocess the image
def load_image(image_path, image_size=(128, 128)):
    image = Image.open(image_path).convert('L')  # Convert to grayscale
    # image = image.resize(image_size)           # Resize the image
    image = np.array(image)                      # Convert to numpy array
    image = torch.tensor(image, dtype=torch.float32)  # Convert to torch tensor
    image = image.unsqueeze(0).unsqueeze(0)      # Add batch and channel dimensions
    return image

# Path to the image
image_path = r"/content/drive/MyDrive/Dataset/images.jpeg"

# Load the image
input_image = load_image(image_path)
print("Input Image Shape:", input_image.shape)


input_image = torch.tensor(input_image, dtype=torch.float32)

# Define a convolutional layer
conv_layer = nn.Conv2d(in_channels=1, out_channels=1, kernel_size=3, stride=1, padding=0)

# Set a specific filter for the convolutional layer
conv_layer.weight = nn.Parameter(torch.tensor([[[[1, 0, -1],
                                                 [1, 0, -1],
```

0s completed at 8:56 PM

AISCIENCES

Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LClW23n

Paused

+ Code + Text All changes saved

RAM Disk

```
import matplotlib.pyplot as plt

# Load and preprocess the image
def load_image(image_path, image_size=(128, 128)):
    image = Image.open(image_path).convert('L')  # Convert to grayscale
    # image = image.resize(image_size)           # Resize the image
    image = np.array(image)                      # Convert to numpy array
    image = torch.tensor(image, dtype=torch.float32)  # Convert to torch tensor
    image = image.unsqueeze(0).unsqueeze(0)      # Add batch and channel dimensions
    return image

# Path to the image
image_path = r"/content/drive/MyDrive/Dataset/images.jpeg"

# Load the image
input_image = load_image(image_path)
print("Input Image Shape:", input_image.shape)


input_image = torch.tensor(input_image, dtype=torch.float32)

# Define a convolutional layer
conv_layer = nn.Conv2d(in_channels=1, out_channels=1, kernel_size=3, stride=1, padding=0)

# Set a specific filter for the convolutional layer
conv_layer.weight = nn.Parameter(torch.tensor([[[[1, 0, -1],
                                                 [1, 0, -1],
```

0s completed at 8:56 PM

AISCIENCES

Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LClW23n

Paused

+ Code + Text

RAM Disk

```
    image = Image.open(image_path).convert('L')  # Convert to grayscale
    # image = image.resize(image_size)             # Resize the image
    image = np.array(image)                      # Convert to numpy array
    image = torch.tensor(image, dtype=torch.float32)  # Convert to torch tensor
    image = image.unsqueeze(0).unsqueeze(0)      # Add batch and channel dimensions
    return image

# Path to the image
image_path = r"/content/drive/MyDrive/Dataset/images.jpeg"

# Load the image
input_image = load_image(image_path)
print("Input Image Shape:", input_image.shape)


# input_image = torch.tensor(input_image, dtype=torch.float32)

# Define a convolutional layer
conv_layer = nn.Conv2d(in_channels=1, out_channels=1, kernel_size=3, stride=1, padding=0)

# Set a specific filter for the convolutional layer
conv_layer.weight = nn.Parameter(torch.tensor([[[[1, 0, -1],
                                                 [1, 0, -1],
                                                 [1, 0, -1]]]], dtype=torch.float32))

# Perform convolution
conv_output = conv_layer(input_image)
```

0s completed at 8:56 PM

AISCIENCES

Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LClW23n

Paused

+ Code + Text

RAM Disk

```
    image = Image.open(image_path).convert('L')  # Convert to grayscale
    # image = image.resize(image_size)             # Resize the image
    image = np.array(image)                      # Convert to numpy array
    image = torch.tensor(image, dtype=torch.float32)  # Convert to torch tensor
    image = image.unsqueeze(0).unsqueeze(0)      # Add batch and channel dimensions
    return image

# Path to the image
image_path = r"/content/drive/MyDrive/Dataset/images.jpeg"

# Load the image
input_image = load_image(image_path)
print("Input Image Shape:", input_image.shape)

# Define a convolutional layer
conv_layer = nn.Conv2d(in_channels=1, out_channels=1, kernel_size=3, stride=1, padding=0)

# Set a specific filter for the convolutional layer
conv_layer.weight = nn.Parameter(torch.tensor([[[[1, 0, -1],
                                                 [1, 0, -1],
                                                 [1, 0, -1]]]], dtype=torch.float32))

# Perform convolution
conv_output = conv_layer(input_image)


# Display the image
plt.imshow(input_image[0, 0].numpy(), cmap='gray')
```

0s completed at 8:56 PM

Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LClW23n

Paused

+ Code + Text

RAM Disk

```
    image = Image.open(image_path).convert('L')  # Convert to grayscale
    # image = image.resize(image_size)             # Resize the image
    image = np.array(image)                      # Convert to numpy array
    image = torch.tensor(image, dtype=torch.float32)  # Convert to torch tensor
    image = image.unsqueeze(0).unsqueeze(0)      # Add batch and channel dimensions
    return image

# Path to the image
image_path = r"/content/drive/MyDrive/Dataset/images.jpeg"

# Load the image
input_image = load_image(image_path)
print("Input Image Shape:", input_image.shape)

# Define a convolutional layer
conv_layer = nn.Conv2d(in_channels=1, out_channels=1, kernel_size=3, stride=1, padding=0)

# Set a specific filter for the convolutional layer
conv_layer.weight = nn.Parameter(torch.tensor([[[[1, 0, -1],
                                                 [1, 0, -1],
                                                 [1, 0, -1]]]], dtype=torch.float32))

# Perform convolution
conv_output = conv_layer(input_image)


# Display the image
plt.imshow(input_image[0, 0].numpy(), cmap='gray')
```

0s completed at 8:56 PM

Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LClW23n

Paused

+ Code + Text All changes saved

RAM Disk

[2] 0s
```
image = np.random.rand(5, 5)
image
```

```
array([[0.09865053, 0.52521243, 0.36051894, 0.21917541, 0.96039324],
       [0.36312466, 0.20557749, 0.09425741, 0.21215107, 0.04927506],
       [0.7737655 , 0.10031832, 0.44264185, 0.97337431, 0.33021284],
       [0.11717513, 0.63664965, 0.31029961, 0.4635026 , 0.76889252],
       [0.45942061, 0.89752827, 0.7664678 , 0.72507998, 0.83271031]])
```

Convolution using Pytorch

```
import torch
import torch.nn as nn
from PIL import Image
import numpy as np
import matplotlib.pyplot as plt

# Load and preprocess the image
def load_image(image_path, image_size=(128, 128)):
    image = Image.open(image_path).convert('L')  # Convert to grayscale
    # image = image.resize(image_size)             # Resize the image
    image = np.array(image)                      # Convert to numpy array
    image = torch.tensor(image, dtype=torch.float32)  # Convert to torch tensor
    image = image.unsqueeze(0).unsqueeze(0)      # Add batch and channel dimensions
    return image
```

0s completed at 8:56 PM

AISCIENCES

Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LClW23n

Paused

+ Code + Text All changes saved

RAM Disk

[2] 0s
```
image = np.random.rand(5, 5)
image
```

```
array([[0.09865053, 0.52521243, 0.36051894, 0.21917541, 0.96039324],
       [0.36312466, 0.20557749, 0.09425741, 0.21215107, 0.04927506],
       [0.7737655 , 0.10031832, 0.44264185, 0.97337431, 0.33021284],
       [0.11717513, 0.63664965, 0.31029961, 0.4635026 , 0.76889252],
       [0.45942061, 0.89752827, 0.7664678 , 0.72507998, 0.83271031]])
```

Convolution using Pytorch

```
import torch
import torch.nn as nn
from PIL import Image
import numpy as np
import matplotlib.pyplot as plt

# Load and preprocess the image
def load_image(image_path, image_size=(128, 128)):
    image = Image.open(image_path).convert('L')  # Convert to grayscale
    # image = image.resize(image_size)             # Resize the image
    image = np.array(image)                      # Convert to numpy array
    image = torch.tensor(image, dtype=torch.float32)  # Convert to torch tensor
    image = image.unsqueeze(0).unsqueeze(0)      # Add batch and channel dimensions
    return image
```

0s completed at 8:56 PM

AISCIENCES

```
    image = torch.tensor(image, dtype=torch.float32)  # Convert to torch tensor
    image = image.unsqueeze(0).unsqueeze(0)      # Add batch and channel dimensions
    return image

# Path to the image
image_path = r"/content/drive/MyDrive/Dataset/images.jpeg"

# Load the image
input_image = load_image(image_path)
print("Input Image Shape:", input_image.shape)

# Define a convolutional layer
conv_layer = nn.Conv2d(in_channels=1, out_channels=1, kernel_size=3, stride=1, padding=0)

# Set a specific filter for the convolutional layer
conv_layer.weight = nn.Parameter(torch.tensor([[[[1, 0, -1],
                                                 [1, 0, -1],
                                                 [1, 0, -1]]]], dtype=torch.float32))

# Perform convolution
conv_output = conv_layer(input_image)


# Display the image
plt.imshow(input_image[0, 0].numpy(), cmap='gray')
plt.title("Input Image")
plt.show()
print("Input Image Shape:", input_image.shape)
```

```
    image = torch.tensor(image, dtype=torch.float32)  # Convert to torch tensor
    image = image.unsqueeze(0).unsqueeze(0)      # Add batch and channel dimensions
    return image

# Path to the image
image_path = r"/content/drive/MyDrive/Dataset/images.jpeg"

# Load the image
input_image = load_image(image_path)
print("Input Image Shape:", input_image.shape)

# Define a convolutional layer
conv_layer = nn.Conv2d(in_channels=1, out_channels=1, kernel_size=3, stride=1, padding=0)

# Set a specific filter for the convolutional layer
conv_layer.weight = nn.Parameter(torch.tensor([[[[1, 0, -1],
                                                 [1, 0, -1],
                                                 [1, 0, -1]]]], dtype=torch.float32))

# Perform convolution
conv_output = conv_layer(input_image)


# Display the image
plt.imshow(input_image[0, 0].numpy(), cmap='gray')
```

Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LClW23n

Paused

+ Code + Text All changes saved

RAM Disk

```
    image = torch.tensor(image, dtype=torch.float32)  # Convert to torch tensor
    image = image.unsqueeze(0).unsqueeze(0)      # Add batch and channel dimensions
    return image

# Path to the image
image_path = r"/content/drive/MyDrive/Dataset/images.jpeg"

# Load the image
input_image = load_image(image_path)
print("Input Image Shape:", input_image.shape)

# Define a convolutional layer
conv_layer = nn.Conv2d(in_channels=1, out_channels=1, kernel_size=3, stride=1, padding=0)

# Set a specific filter for the convolutional layer
conv_layer.weight = nn.Parameter(torch.tensor([[[[1, 0, -1],
                                                 [1, 0, -1],
                                                 [1, 0, -1]]]], dtype=torch.float32))

# Perform convolution
conv_output = conv_layer(input_image)


# Display the image
plt.imshow(input_image[0, 0].numpy(), cmap='gray')
```

0s completed at 8:56 PM

Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LClW23n

Paused

+ Code + Text All changes saved

RAM Disk

```
image_path = '/content/drive/MyDrive/Dataset/images.jpeg'

# Load the image
input_image = load_image(image_path)
print("Input Image Shape:", input_image.shape)

# Define a convolutional layer
conv_layer = nn.Conv2d(in_channels=1, out_channels=1, kernel_size=3, stride=1, padding=0)

# Set a specific filter for the convolutional layer
conv_layer.weight = nn.Parameter(torch.tensor([[[[1, 0, -1],
                                                 [1, 0, -1],
                                                 [1, 0, -1]]]], dtype=torch.float32))

# Perform convolution
conv_output = conv_layer(input_image)


# Display the image
plt.imshow(input_image[0, 0].numpy(), cmap='gray')
plt.title("Input Image")
plt.show()
print("Input Image Shape:", input_image.shape)

# Display the image
plt.imshow(conv_output[0, 0].detach().numpy(), cmap='gray')
```

0s completed at 8:56 PM

AISCIENCES

Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LClW23n

Paused

+ Code + Text All changes saved

RAM Disk

```
# Define a convolutional layer
conv_layer = nn.Conv2d(in_channels=1, out_channels=1, kernel_size=3, stride=1, padding=0)

# Set a specific filter for the convolutional layer
conv_layer.weight = nn.Parameter(torch.tensor([[[[1, 0, -1],
                                                 [1, 0, -1],
                                                 [1, 0, -1]]]], dtype=torch.float32))

# Perform convolution
conv_output = conv_layer(input_image)


# Display the image
plt.imshow(input_image[0, 0].numpy(), cmap='gray')
plt.title("Input Image")
plt.show()
print("Input Image Shape:", input_image.shape)

# Display the image
plt.imshow(conv_output[0, 0].detach().numpy(), cmap='gray')
plt.title("Conv Image")
plt.show()
print("Conv Image Shape:", conv_output.shape)
```

0s completed at 8:56 PM

AISCIENCES

Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LCIW23n

Paused

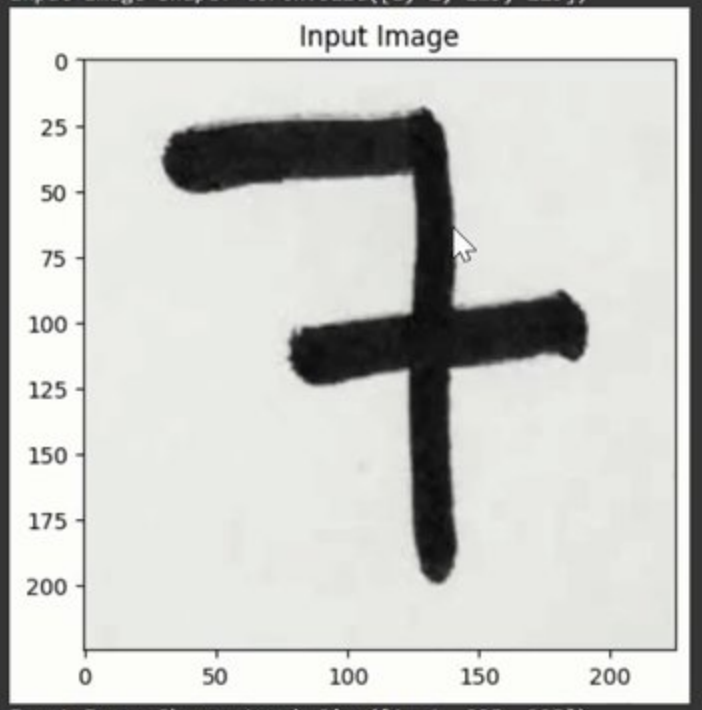
+ Code + Text Saving...

RAM Disk

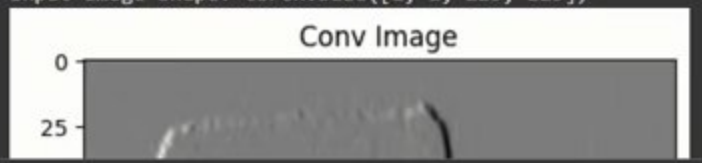
```
# Display the image
plt.imshow(conv_output[0, 0].detach().numpy(), cmap='gray')
plt.title("Conv Image")
plt.show()
print("Conv Image Shape:", conv_output.shape)
```

2s

```
Input Image Shape: torch.Size([1, 1, 225, 225])
```




```
Input Image Shape: torch.Size([1, 1, 225, 225])
```



1s completed at 9:01 PM

AISCIENCES

Udemy

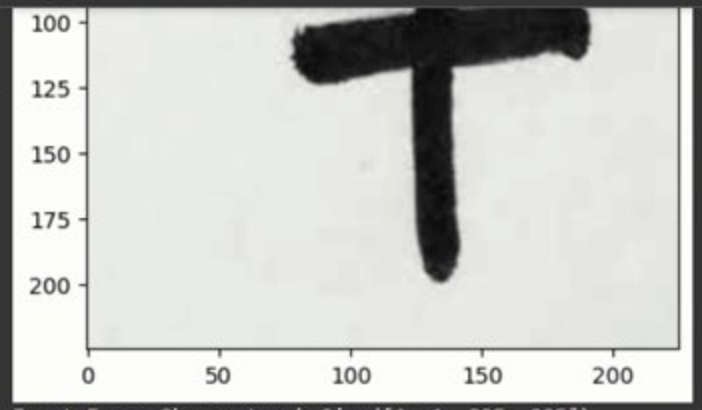
Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LCIW23n

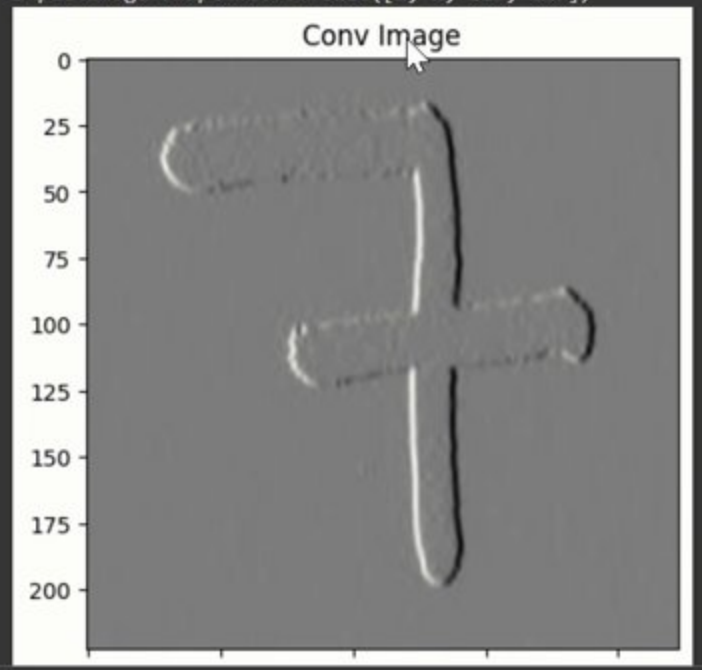
Paused

+ Code + Text All changes saved

RAM Disk




```
Input Image Shape: torch.Size([1, 1, 225, 225])
```



1s completed at 9:01 PM

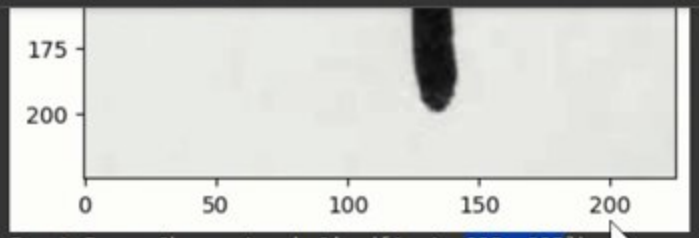
Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LCIW23n

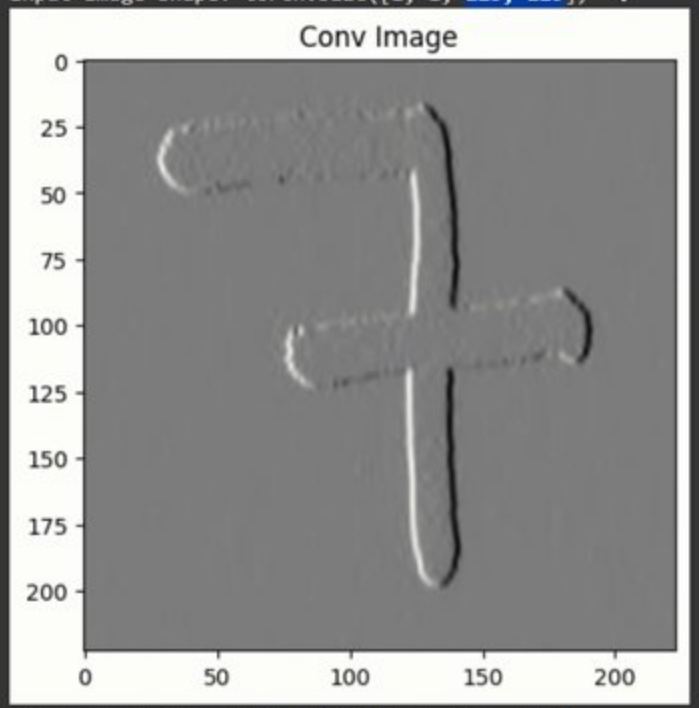
Paused

+ Code + Text All changes saved

RAM Disk




```
Input Image Shape: torch.Size([1, 1, 225, 225])
```




```
Conv Image Shape: torch.Size([1, 1, 223, 223])
```

[ ]

1s completed at 9:01 PM

Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LClW23n

Paused

+ Code + Text All changes saved

RAM Disk

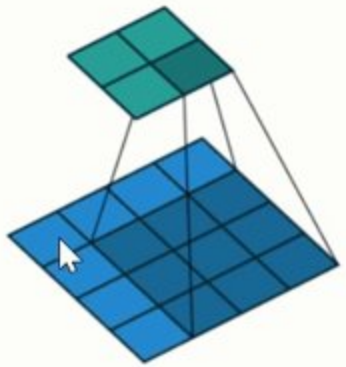
```
    image = Image.open(image_path).convert('L')  # Convert to grayscale
    # image = image.resize(image_size)             # Resize the image
    image = np.array(image)                      # Convert to numpy array
    image = torch.tensor(image, dtype=torch.float32)  # Convert to torch tensor
    image = image.unsqueeze(0).unsqueeze(0)      # Add batch and channel dimensions
    return image

# Path to the image
image_path = r"/content/drive/MyDrive/Dataset/images.jpeg"

# Load the image
input_image = load_image(image_path)
print("Input Image Shape:", input_image.shape)

# Define a convolutional layer
conv_layer = nn.Conv2d(in_channels=1, out_channels=1, kernel_size=3, stride=1, padding=0)

# Set a specific filter for the convolutional layer
conv_layer.weight = nn.Parameter(torch.tensor([[[[1, 0, -1],
                                                 [1, 0, -1],
                                                 [1, 0, -1]]]], dtype=torch.float32))

# Perform convolution
conv_output = conv_layer(input_image)


# Display the image
plt.imshow(input_image[0, 0].numpy(), cmap='gray')
```

2s

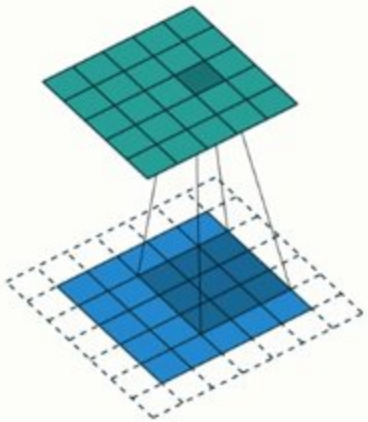
1s completed at 9:01 PM

Convolution

padding = 0, stride = 1

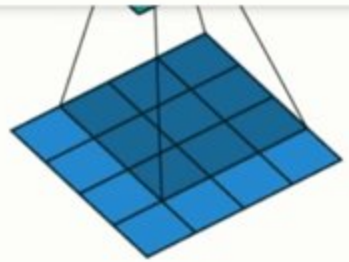


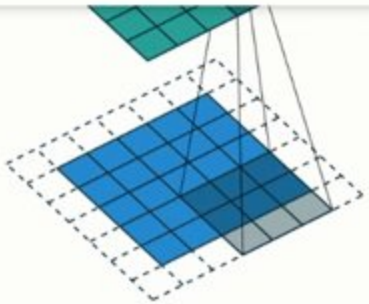
padding = 1, stride = 1



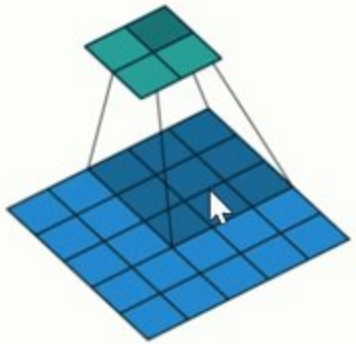
padding = 0, stride = 2

padding = 1, stride = 2

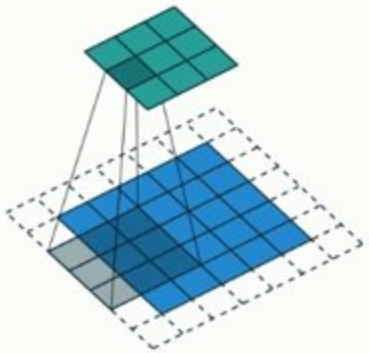


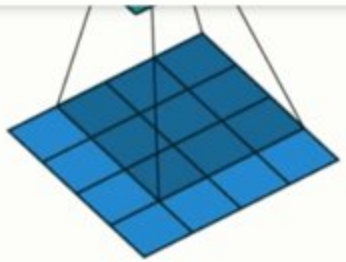


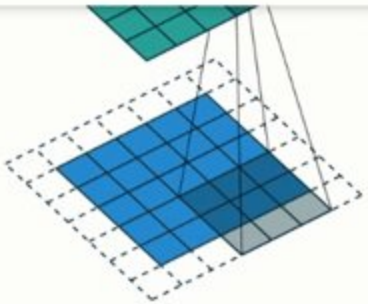
padding = 0, stride = 2



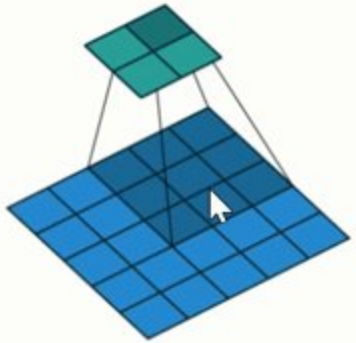
padding = 1, stride = 2



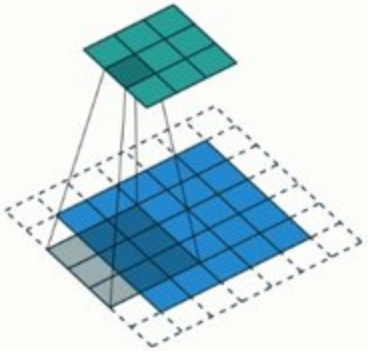


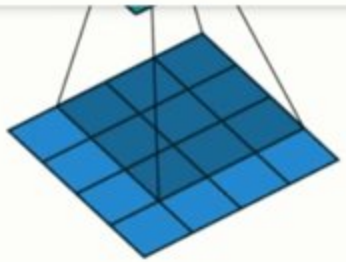


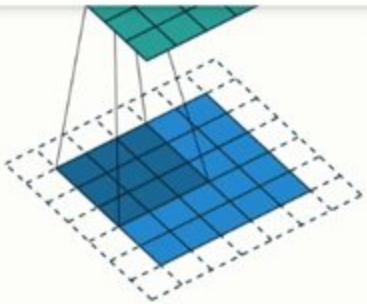
padding = 0, stride = 2



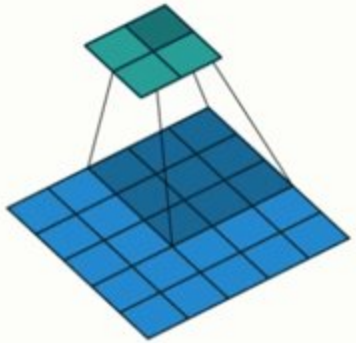
padding = 1, stride = 2



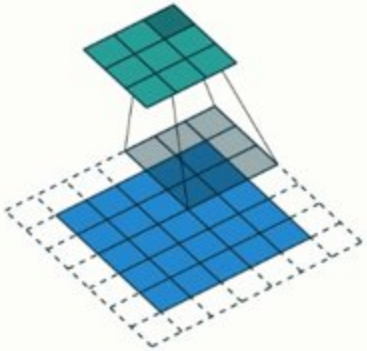


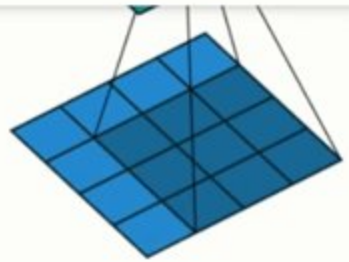


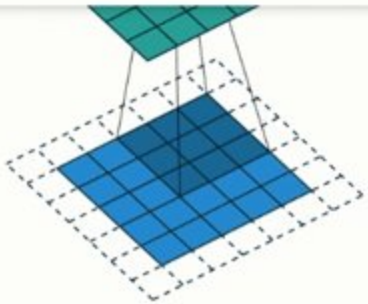
padding = 0, stride = 2



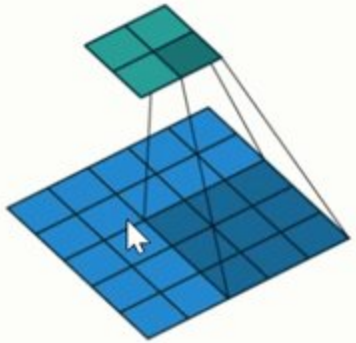
padding = 1, stride = 2



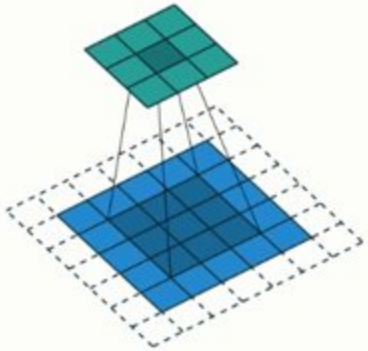




padding = 0, stride = 2

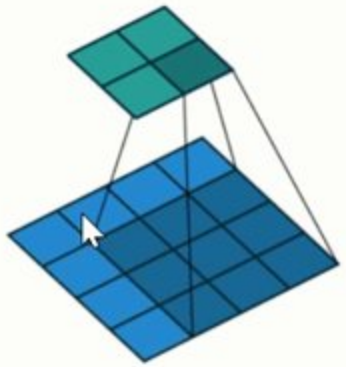


padding = 1, stride = 2

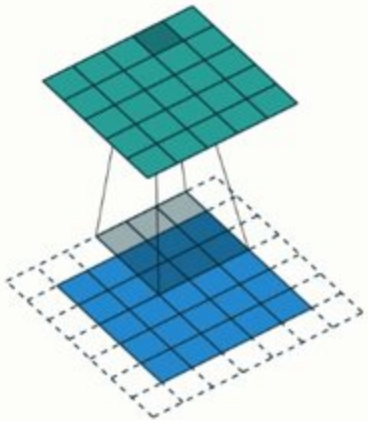


Convolution

padding = 0, stride = 1

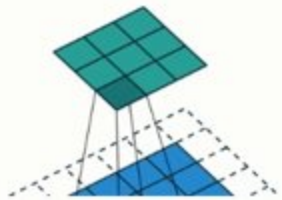


padding = 1, stride = 1



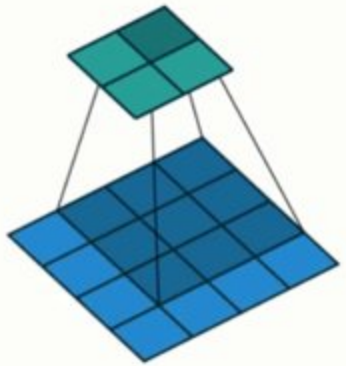
padding = 0, stride = 2

padding = 1, stride = 2

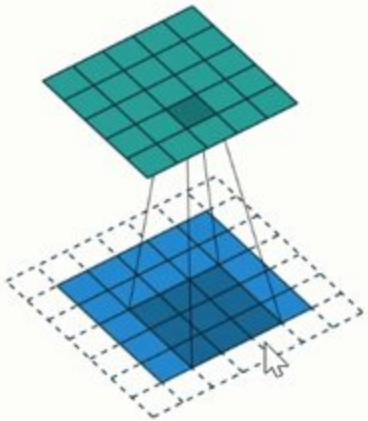


Convolution

padding = 0, stride = 1

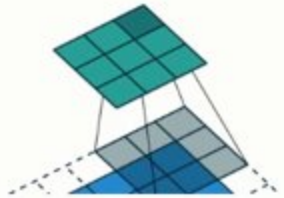


padding = 1, stride = 1



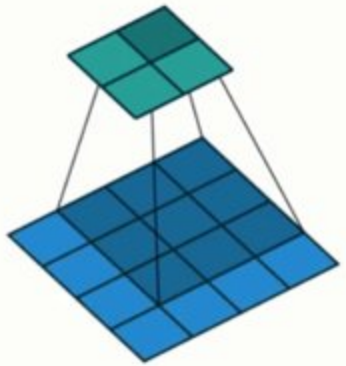
padding = 0, stride = 2

padding = 1, stride = 2

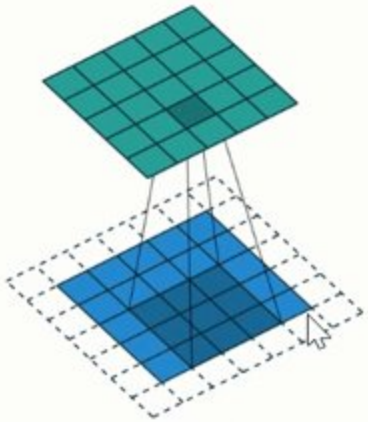


Convolution

padding = 0, stride = 1

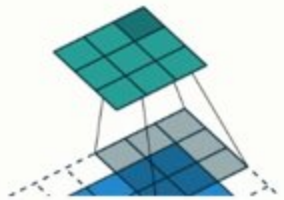


padding = 1, stride = 1



padding = 0, stride = 2

padding = 1, stride = 2



Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

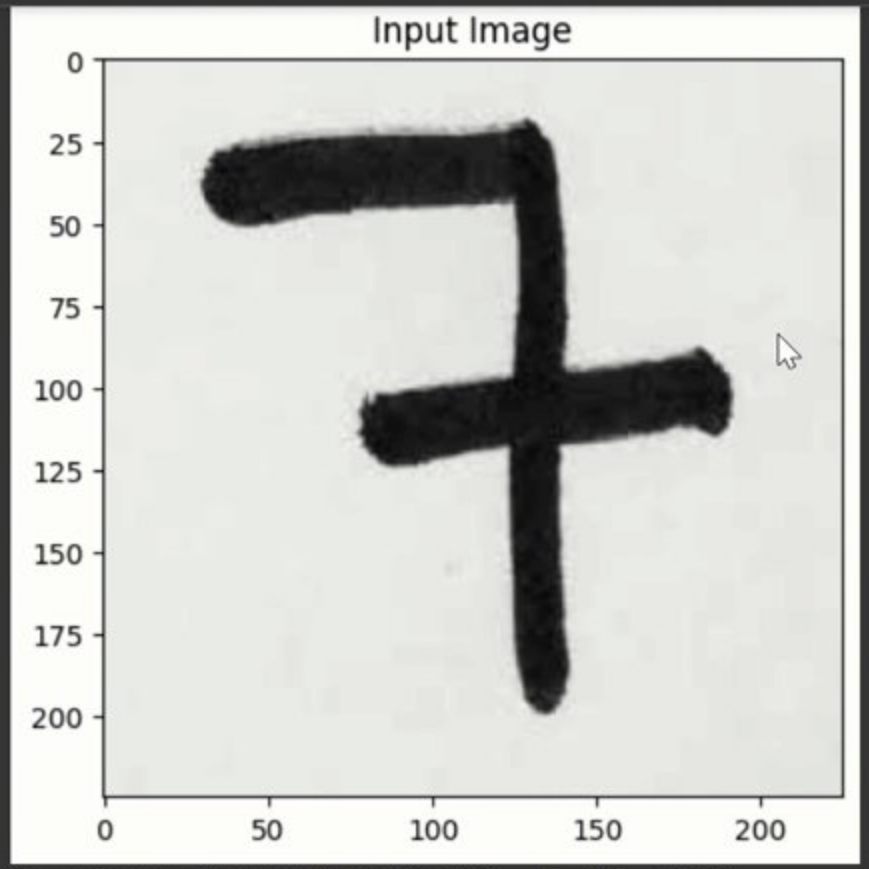
colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LCIW23n

Paused

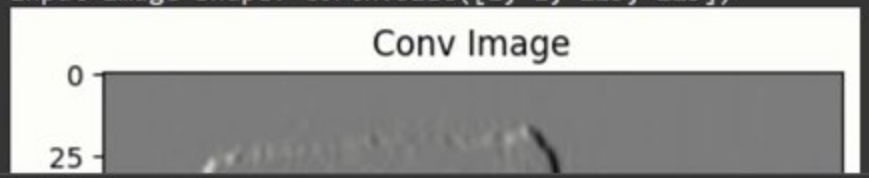
+ Code + Text All changes saved

RAM Disk

2s




```
Input Image Shape: torch.Size([1, 1, 225, 225])
```



1s completed at 9:01 PM

Pytorch.ipynb - Colab | Animations of Convolution and | Image Kernels explained visuall

colab.research.google.com/drive/1imQs4frEn2sYP0VW7SjxgJwC5ZgNZdpv#scrollTo=sud58LClW23n

Paused

+ Code + Text All changes saved

RAM Disk

```python
import torch.nn as nn
from PIL import Image
import numpy as np
import matplotlib.pyplot as plt

# Load and preprocess the image
def load_image(image_path, image_size=(128, 128)):
    image = Image.open(image_path).convert('L')  # Convert to grayscale
    # image = image.resize(image_size)             # Resize the image
    image = np.array(image)                      # Convert to numpy array
    image = torch.tensor(image, dtype=torch.float32)  # Convert to torch tensor
    image = image.unsqueeze(0).unsqueeze(0)      # Add batch and channel dimensions
    return image

# Path to the image
image_path = r"/content/drive/MyDrive/Dataset/images.jpeg"

# Load the image
input_image = load_image(image_path)
print("Input Image Shape:", input_image.shape)

# Define a convolutional layer
conv_layer = nn.Conv2d(in_channels=1, out_channels=1, kernel_size=3, stride=1, padding=0)

# Set a specific filter for the convolutional layer
conv_layer.weight = nn.Parameter(torch.tensor([[[[1, 0, -1],
                                                 [1, 0, -1],
                                                 [1, 0, -1]]]], dtype=torch.float32))

# Perform convolution
conv_output = conv_layer(input_image)
```

2s

✓ 1s completed at 9:01 PM