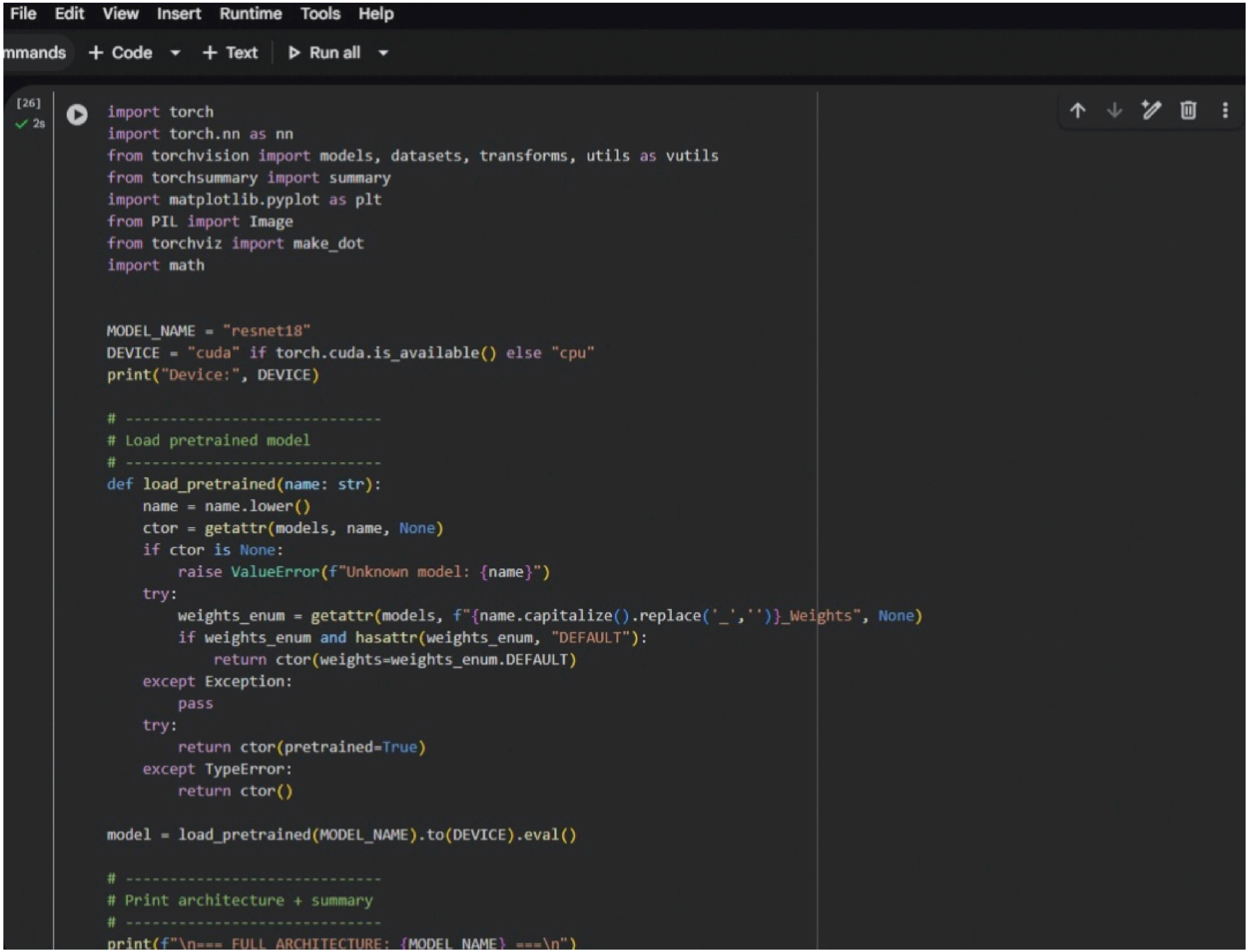

File Edit View Insert Runtime Tools Help
mmands + Code + Text Run all
[26]
2s

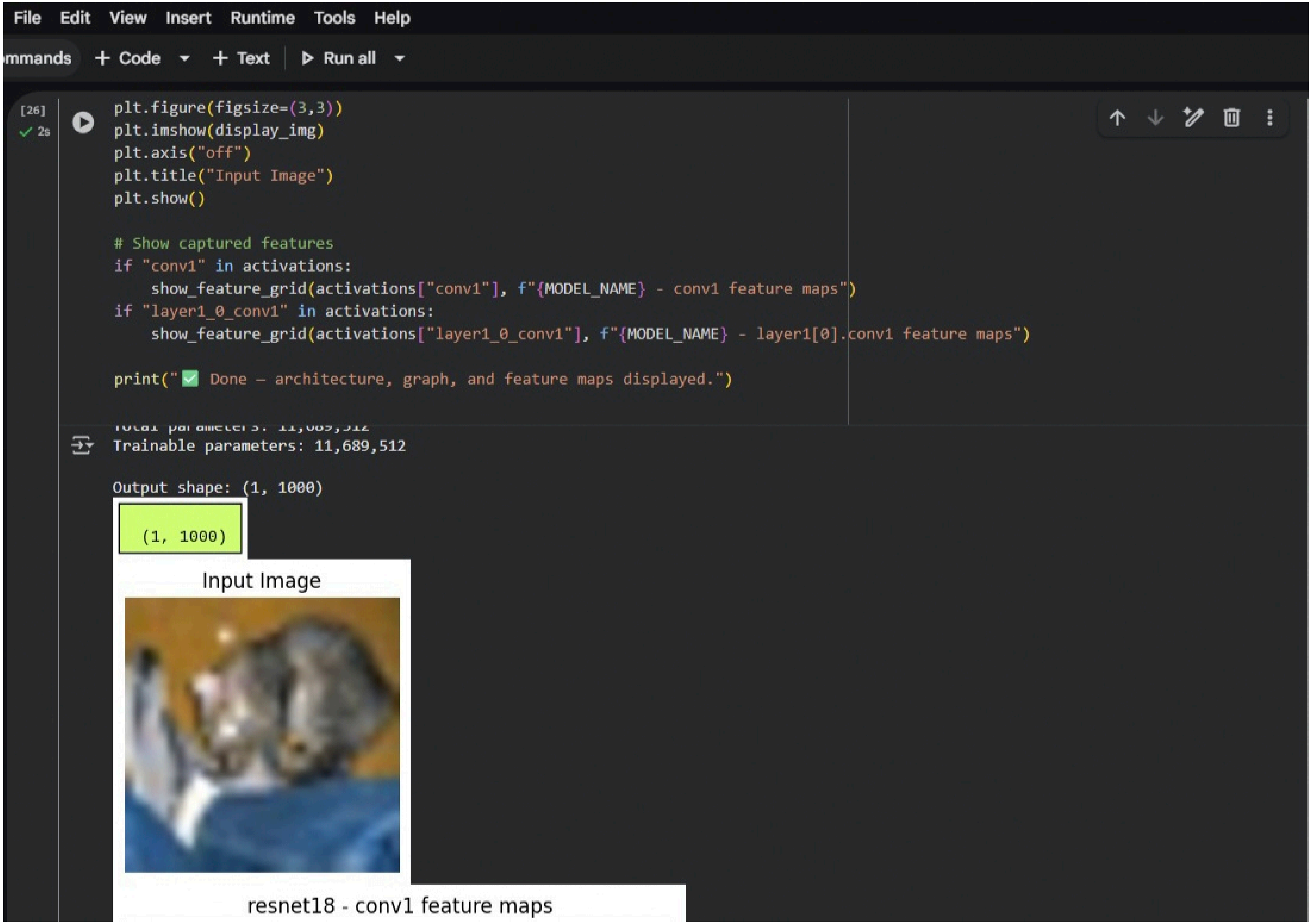
```python
import torch
import torch.nn as nn
from torchvision import models, datasets, transforms, utils as vutils
from torchsummary import summary
import matplotlib.pyplot as plt
from PIL import Image
from torchviz import make_dot
import math


MODEL_NAME = "resnet18"
DEVICE = "cuda" if torch.cuda.is_available() else "cpu"
print("Device:", DEVICE)

# ----------------------------
# Load pretrained model
# ----------------------------
def load_pretrained(name: str):
    name = name.lower()
    ctor = getattr(models, name, None)
    if ctor is None:
        raise ValueError(f"Unknown model: {name}")
    try:
        weights_enum = getattr(models, f"{name.capitalize().replace('_','')}_Weights", None)
        if weights_enum and hasattr(weights_enum, "DEFAULT"):
            return ctor(weights=weights_enum.DEFAULT)
    except Exception:
        pass
    try:
        return ctor(pretrained=True)
    except TypeError:
        return ctor()

model = load_pretrained(MODEL_NAME).to(DEVICE).eval()

# ----------------------------
# Print architecture + summary
# ----------------------------
print(f"\n=== FULL ARCHITECTURE: {MODEL_NAME} ===\n")
```



```
plt.figure(figsize=(3,3))
plt.imshow(display_img)
plt.axis("off")
plt.title("Input Image")
plt.show()

# Show captured features
if "conv1" in activations:
    show_feature_grid(activations["conv1"], f"{MODEL_NAME} - conv1 feature maps")
if "layer1_0_conv1" in activations:
    show_feature_grid(activations["layer1_0_conv1"], f"{MODEL_NAME} - layer1[0].conv1 feature maps")

print("✅ Done — architecture, graph, and feature maps displayed.")
```

```
Total parameters: 11,689,512
Trainable parameters: 11,689,512

Output shape: (1, 1000)
```