جامعة الأمير مقرن بن عبد العزيز
University of Prince Mugrin

Department of Artificial Intelligence

College of Computer and Cyber Sciences

Introduction to Deep Learning

Segmentation with a Pre-Trained Model in PyTorch

1. Learning Objectives

By the end of this lab, students will:

- Understand the concept of **semantic segmentation**.
- Learn how to fine-tune a model for segmentation tasks.
- Train and evaluate a segmentation model on the **Cityscapes dataset**.

2. Explanation of Key Concepts

- **Semantic Segmentation:**

  Is a deep learning algorithm that involves classifying each pixel in an image into a predefined label or category. Unlike object detection, which identifies and localizes objects with bounding boxes, semantic segmentation provides a pixel-wise classification, effectively segmenting the image into meaningful regions.

- **Cityscapes Dataset:**

  The Cityscapes Dataset is a large-scale dataset designed for semantic segmentation, instance segmentation, and scene understanding in urban environments. It contains high-resolution images captured from street scenes in multiple European cities.

  Key Features of Cityscapes:

  Size: Around 25 thousand images.
  Image Format: Images are 2048×1024 pixels, in RGB format.
  Labels: Has 30 classes grouped into 8 categories (only 19 are commonly used for training).

- **Data Augmentation:**

  It involves applying random transformations to training images to increase dataset diversity and improve model generalization.

3. Activities

➢ Exercise 1: Fine-Tuning a Pre-Trained Model for Cityscapes Segmentation.

In this exercise, we will fine-tune a pre-trained model for semantic segmentation on the Cityscapes dataset using PyTorch. You will Load and preprocess the dataset, modify the model architecture to match the dataset labels, apply data augmentation techniques to enhance training, and train the model then evaluate its performance on validation images.

```
# Code provided in the notebook
```

4. Tasks

➢ Task 1:

In the provided segmentation notebook, some errors have been intentionally introduced in the data loading or model inference phase. Your task is to:

1. Identify and fix the error.
2. Explain why the error occurs and how your fix resolves the issue.

Submit your jupyter notebook & Add a screenshot of the result:

```
# preprocessing and augmentation
def get_augmentations(train=True):
    if train:
        return Compose([
            Resize(96, 256),  # Resize images to (256, 512)
            HorizontalFlip(p=0.5), # Flip images horizontally with 50% probability
            RandomBrightnessContrast(p=0.2), # Randomly change brightness/contrast with 20% probabi
            ToTensorV2() # Convert image/mask to PyTorch tensors
        ])
    else: # No data augmentation for validation (Only preprocessing)
        return Compose([
            Resize(96, 256),
            ToTensorV2()
        ])
✓ 0.0s
```

```python
# Training loop
num_epochs = 10
num_classes = 19

for epoch in range(num_epochs):
    train_loss = 0.0

    for images, masks in train_loader:

        # Ensurs the pixel values in masks are in the range [0, num_classes - 1].
        masks = torch.clamp(masks, 0, num_classes-1)
        images, masks = images.float().to(device), masks.long().to(device)

        optimizer.zero_grad()
        outputs = model(images)
        loss = loss_func(outputs, masks)
        loss.backward()
        optimizer.step()

        train_loss += loss.item()

    print(f"Epoch [{epoch+1}/{num_epochs}], Loss: {train_loss/len(train_loader):.4f}"

print('Training complete!')
```

✓ 2m 8.3s

```python
# Validation loop
num_epochs = 10

for epoch in range(num_epochs):
    model.eval()
    correct_pixels = 0
    total_pixels = 0

    with torch.no_grad():
        for images, masks in val_loader:
            masks = torch.clamp(masks, 0, num_classes-1)
            images, masks = images.float().to(device), masks.long().to(device)

            outputs = model(images)  # Model prediction (logits)
            predictions = torch.argmax(outputs, dim=1)  # Get class with highest probabi

            correct_pixels += (predictions == masks).sum().item()
            total_pixels += masks.numel()  # Total number of pixels

accuracy = correct_pixels / total_pixels * 100  # Convert to percentage
print(f"Final Accuracy: {accuracy:.2f}%")
```

✓ 16.6s

```
...   Epoch [1/10], Loss: 0.1840
      Epoch [2/10], Loss: 0.1227
      Epoch [3/10], Loss: 0.1196
      Epoch [4/10], Loss: 0.1169
      Epoch [5/10], Loss: 0.1162
      Epoch [6/10], Loss: 0.1121
      Epoch [7/10], Loss: 0.1101
      Epoch [8/10], Loss: 0.1083
      Epoch [9/10], Loss: 0.1049
      Epoch [10/10], Loss: 0.1023
      Training complete!
```

Explanation:

The 1st error I needed to fix, was due to the file structure. Mine was different from what was provided in the code, so I adjusted the code to match the structure of my own dataset.

The 2nd error I found was due to tensor size error. The input provided was (256, 512) while the expected input was (96, 256). Fixing this solved the runtime error.

After that, I noticed in the validation loop, we were actually using the train_loader which was incorrect, so I changed it to val_loader.

The last thing I changed was ‘labels’ in the training loop. In all parts of the code we were using the variable ‘masks’, but in one instance within the training loop we used ‘labels’ instead. Although, I don’t think this would cause any problems, I adjusted it to match the rest of the code.

5. References

Cityscape Segmentation || UNet || PyTorch

Semantic Segmentation with PyTorch: U-NET from scratch | by Alessandro Mondin | Medium

Cityscapes — Torchvision main documentation

How to use Glob() function to find files recursively in Python? - GeeksforGeeks

Writing Custom Datasets, DataLoaders and Transforms — PyTorch Tutorials 2.6.0+cu124 documentation

mcordts/cityscapesScripts: README and scripts for the Cityscapes Dataset

vision/torchvision/datasets/cityscapes.py at main · pytorch/vision

cityscapes dataset | Kaggle

Semantic_Seg_FCNN

Cityscapes-data-exploration | Kaggle

Semantic Segmentation - MATLAB & Simulink

Cityscapes Dataset | Papers With Code

Cityscapes Dataset – Semantic Understanding of Urban Street Scenes

What is data augmentation? | IBM