COURSE: DSA4050

COURSE TITLE: Deep Learning for Computer Vision

INSTRUCTOR: DR. EDWARD OMBUI

Lesson Notes: Semantic Segmentation and Instance Segmentation

---

Week 6 Overview

This week, we explore **segmentation tasks in computer vision**, where the goal is to classify every pixel in an image. Unlike traditional image classification, which assigns a single label to an image, segmentation tasks aim to **identify objects at the pixel level**.

We will focus on two key segmentation techniques:

1. **Semantic Segmentation:** Classifies all pixels belonging to the same category as one.
    - Example: Identifying **all cars** in an image but not distinguishing between individual cars.
2. **Instance Segmentation:** Not only classifies objects but also **distinguishes between individual instances** of the same class.
    - Example: Separating **one car from another** in the same image.

You will also study **popular segmentation architectures** such as:

- **Fully Convolutional Networks (FCNs):** Transform traditional CNNs into pixel-level classifiers.
- **U-Net:** An encoder-decoder network widely used for **biomedical image segmentation**.

By the end of this week, you will be able to **differentiate between segmentation types, understand segmentation models, and implement segmentation tasks using FCNs and U-Net**.

1. Understanding Semantic and Instance Segmentation

What is Semantic Segmentation?

Semantic segmentation assigns each pixel to a **category** (e.g., car, road, pedestrian). All objects of the same category are given the same label, without distinguishing between different instances.

Example Use Cases:
✅ **Autonomous Driving:** Identifying road, lanes, vehicles, and pedestrians.
✅ **Medical Imaging:** Segmenting organs, tumors, or abnormalities in MRI scans.
✅ **Aerial Image Analysis:** Identifying land, water, and buildings in satellite images.

Example Output:

- Input: Image with multiple cars.
- Output: A mask where all car pixels are marked in **one color**.

What is Instance Segmentation?

Instance segmentation goes beyond semantic segmentation by **differentiating each object instance** within the same class.

Example Use Cases:
✅ **Self-Driving Cars:** Detecting and tracking individual vehicles and pedestrians.
✅ **Retail and Inventory Management:** Counting products and monitoring stock.
✅ **Healthcare:** Identifying different cell types in microscopy images.

Example Output:

- Input: Image with multiple people.
- Output: A mask where each person has a **unique color** to distinguish between individuals.

📖 **Reference:** He et al. (2017). *Mask R-CNN: Instance Segmentation Framework.*

2. Segmentation Architectures: FCNs and U-Net

2.1 Fully Convolutional Networks (FCNs)

📌 **Key Idea:** Transform a traditional CNN into a fully convolutional model where the output is a **pixel-wise** prediction.

✅ **Strengths:**

- Efficient for dense predictions.
- Can adapt classification models for segmentation.

❌ Limitations:

- Struggles with preserving fine-grained details.
- May produce blurry segmentation boundaries.

🔧 Implementation of FCN using TensorFlow

```
import tensorflow as tf
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Conv2D, Conv2DTranspose, Input

# Define FCN model
input_layer = Input(shape=(256, 256, 3))
conv1 = Conv2D(64, (3, 3), activation='relu', padding='same')(input_layer)
conv2 = Conv2D(128, (3, 3), activation='relu', padding='same')(conv1)
conv3 = Conv2DTranspose(64, (3, 3), strides=(2, 2), activation='relu',
padding='same')(conv2)
output_layer = Conv2D(1, (1, 1), activation='sigmoid')(conv3)

model = Model(inputs=input_layer, outputs=output_layer)
model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])
model.summary()
```

📖 **Reference:** Long et al. (2015). *Fully Convolutional Networks for Semantic Segmentation.*

2.2 U-Net: Encoder-Decoder Architecture

📌 **Key Idea:** Uses an **encoder-decoder** structure with skip connections to capture both **global context** and **fine details**.

✅ Strengths:

- Works well for **biomedical image segmentation**.
- Maintains high accuracy with limited data.

❌ Limitations:

- Computationally intensive for large images.
- Requires data augmentation for better generalization.

🔧 Implementation of U-Net using TensorFlow

```
from tensorflow.keras.layers import MaxPooling2D, concatenate

def unet_model(input_shape=(256, 256, 3)):
    inputs = Input(shape=input_shape)

    # Encoder
    conv1 = Conv2D(64, (3, 3), activation='relu', padding='same')(inputs)
    pool1 = MaxPooling2D((2, 2))(conv1)

    conv2 = Conv2D(128, (3, 3), activation='relu', padding='same')(pool1)
    pool2 = MaxPooling2D((2, 2))(conv2)

    # Decoder
    up1 = Conv2DTranspose(64, (3, 3), strides=(2, 2), activation='relu',
padding='same')(pool2)
    merge1 = concatenate([conv1, up1], axis=3)
    output_layer = Conv2D(1, (1, 1), activation='sigmoid')(merge1)

    model = Model(inputs=inputs, outputs=output_layer)
    model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])
    return model

model = unet_model()
model.summary()
```

📖 **Reference:** Ronneberger et al. (2015). *U-Net: Convolutional Networks for Biomedical Image Segmentation.*

3. Training and Evaluating a Segmentation Model

3.1 Dataset Preparation

1. **Choose a Dataset:** Common datasets include **COCO, PASCAL VOC, Cityscapes**, or a **custom medical dataset**.
2. **Preprocess Images:** Resize images, normalize pixel values, and apply data augmentation.

3.2 Training the Model

```
history = model.fit(train_images, train_masks, epochs=10,
validation_data=(val_images, val_masks))
```

3.3 Evaluation Metrics

Metric	Description
IoU (Intersection over Union)	Measures overlap between predicted and ground truth masks.
Dice Coefficient	Measures similarity between two sets (higher is better).
Pixel Accuracy	Percentage of correctly classified pixels.

4. Learning Outcomes

By the end of this session, you should be able to:
✅ **Differentiate** between **semantic segmentation and instance segmentation**.
✅ **Describe and implement Fully Convolutional Networks (FCNs)** and **U-Net**.
✅ **Train and test a segmentation model** using a dataset.

5. Laboratory Assignment

Task:

Implement Fully Convolutional Networks (FCNs) and U-Net for segmentation tasks.

Steps:

1. Select a dataset (e.g., medical, street scene segmentation).
2. Implement both **FCN and U-Net architectures**.
3. Train the models using **IoU and Dice Coefficient** as metrics.
4. Compare model performance.

📌 Deliverables:

- Trained segmentation models.
- Evaluation report comparing **FCN vs. U-Net** performance.
- Sample segmentation results.

📖 **Reference:** Ronneberger et al. (2015). *U-Net for Biomedical Image Segmentation.*

6. Conclusion

✅ **Semantic segmentation** classifies **all pixels** in an image.
✅ **Instance segmentation** differentiates **individual objects** within the same category.
✅ **FCNs and U-Net** are widely used **segmentation architectures**.
✅ **Segmentation models are evaluated using IoU, Dice Coefficient, and Pixel Accuracy**.

📌 **Next Steps:**

- Experiment with **custom datasets**.
- Use **transfer learning** to improve segmentation accuracy.
- Apply segmentation to **real-world applications** like medical imaging and self-driving cars.