Московский государственный технический университет им. Н.Э.

Баумана

Кафедра «Системы обработки информации и управления»

Домашнее Задание

по дисциплине

«Методы машинного обучения»

Выполнил:

студент группы ИУ5-22М

Лю Ченхао

Москва — 2024 г.

Задание

Домашнее задание по дисциплине направлено на анализ современных методов машинного обучения и их применение для решения практических задач. Домашнее задание включает три основных этапа:

- выбор задачи;
- теоретический этап;
- практический этап.

Этап выбора задачи предполагает анализ ресурса paperswithcode. Данный ресурс включает описание нескольких тысяч современных задач в области машинного обучения. Каждое описание задачи содержит ссылки на наиболее современные и актуальные научные статьи, предназначенные для решения задачи (список статей регулярно обновляется авторами ресурса). Каждое описание статьи содержит ссылку на репозиторий с открытым исходным кодом, реализующим представленные в статье эксперименты. На этапе выбора задачи обучающийся выбирает одну из задач машинного обучения, описание которой содержит ссылки на статьи и репозитории с исходным кодом. Теоретический этап включает проработку как минимум двух статей, относящихся к выбранной задаче. Результаты проработки обучающийся излагает в

теоретической части отчета по домашнему заданию, которая может включать:

- описание общих подходов к решению задачи;

конкретные топологии нейронных сетей, нейросетевых ансамблей или других моделей машинного обучения, предназначенных для решения задачи;

- математическое описание, алгоритмы функционирования, особенности обучения используемых для решения задачи нейронных сетей, нейросетевых ансамблей или других моделей машинного обучения;

- описание наборов данных, используемых для обучения моделей;

- оценка качества решения задачи, описание метрик качества и их значений;

- предложения обучающегося по улучшению качества решения задачи. Практический этап включает повторение экспериментов авторов статей на основе представленных авторами репозиториев с исходным кодом и возможное улучшение обучающимися полученных результатов. Результаты проработки обучающийся излагает в практической части отчета по домашнему заданию, которая может включать:

- исходные коды программ, представленные авторами статей, результаты документирования программ обучающимися с

использованием диаграмм UML, путем визуализации топологий нейронных сетей и другими способами;

- результаты выполнения программ, вычисление значений для описанных в статьях метрик качества, выводы обучающегося о воспроизводимости экспериментов авторов статей и соответствии практических экспериментов теоретическим материалам статей;

- предложения обучающегося по возможным улучшениям решения задачи, результаты практических экспериментов (исходные коды, документация) по возможному улучшению решения задачи.

Выбранная задача: «Семантическая сегментация»

1. Выбор задачи

В нейронных сетях семантическая сегментация - это задача, которая помечает каждый пиксель на изображении как принадлежащий к различной семантической категории. По сравнению с простыми задачами классификации, семантическая сегментация требует классификации каждого пикселя и, следовательно, требует более совершенной структуры нейронной сети и более сложного процесса обучения. Вот общие методы сегментации семантики и их ключевые аспекты:

1. **Применение сверточных нейронных сетей (CNN)**: Для сегментации семантики часто используют сверточные нейронные сети, потому что CNN обладают способностью эффективно изучать локальную информацию на изображении. Часто используемые архитектуры CNN включают U-Net, SegNet и DeepLab.

2. **Структура энкодер-декодер**: Многие модели сегментации семантики используют структуру энкодер-декодер, где энкодер используется для извлечения признаков из изображения, а декодер используется для отображения этих признаков обратно на исходное разрешение изображения и генерации результата сегментации семантики.
3. **Функции потерь**: Обычно используемые функции потерь включают функцию потерь перекрестной энтропии и функцию потерь Dice. Функция потерь перекрестной энтропии используется для измерения различий между выходом модели и истинными метками, в то время как функция потерь Dice более подходит для работы с несбалансированными классами.
4. **Аугментация данных**: Техники аугментации данных важны для сегментации семантики, поскольку они могут эффективно расширить обучающий набор данных и повысить обобщающую способность модели. Обычные операции аугментации данных включают случайное вращение, масштабирование, отражение и изменение цвета.
5. **Разреженная свертка (Dilated Convolution)**: Разреженная свертка - это операция свертки, используемая для увеличения зоны восприятия, и часто применяется в сегментации семантики для захвата более широкого контекста и повышения производительности модели.
6. **Многомасштабная обработка**: Для обработки объектов разных масштабов многие модели сегментации семантики используют многомасштабные стратегии, такие как введение пирамиды многомасштабных признаков или использование изображений с разными масштабами в качестве входных данных.
7. **Использование предварительно обученных моделей**: Предварительно обученные модели (например, модели, обученные на наборе данных ImageNet) обычно могут обеспечить лучшее начальное представление признаков, поэтому они широко применяются во многих задачах сегментации семантики.

2. Теоретический этап

Часть I

Тема《Семантическая сегментация в машинном обучении》

1. **Контекст и значимость семантической сегментации**:

- Семантическая сегментация - это важная задача в области обработки изображений, которая заключается в присвоении каждому пикселю изображения соответствующего семантического класса, что позволяет более детально понимать и анализировать изображение.
- В отличие от простых задач классификации изображений и обнаружения объектов, семантическая сегментация требует, чтобы модель не только распознавала классы объектов на изображении, но и точно определяла границы каждого объекта на уровне пикселей, что является критически важным для многих приложений, таких как автономное вождение, медицинский анализ изображений и земельное наблюдение.

2. **Методы семантической сегментации**:

- В статье представлены три основных метода семантической сегментации: сегментация по областям, сегментация на основе полносверточных сетей и сегментация с использованием слабого надзора.
- Методы сегментации по областям обычно основаны на алгоритмах роста области или графовых алгоритмах разреза, которые направлены на разделение изображения на различные

области в зависимости от схожести пикселей и присвоения этим областям определенных семантических классов.

- Методы на основе полносверточных сетей (FCN) изменяют традиционные сверточные нейронные сети (CNN) таким образом, чтобы они могли принимать изображения любого размера и выдавать плотные результаты классификации с соответствующим разрешением. FCN часто является одним из передовых методов семантической сегментации.
- Методы с использованием слабого надзора используют дополнительную вспомогательную информацию (например, метки на уровне изображения или ограничивающие рамки), чтобы улучшить производительность модели, и они обычно могут быть эффективно обучены при высоких затратах на разметку данных.

3. **Подробное описание полносверточной сети (FCN)**:

- Статья подробно описывает структуру и шаги реализации FCN, особенно сосредотачиваясь на архитектуре FCN-8. FCN-8 использует слои деконволюции и пропускные соединения для достижения точной семантической сегментации.
- Благодаря пропускным соединениям FCN может объединять информацию о признаках с разных уровней для достижения

лучшей сегментации, сохраняя при этом информацию о высоком разрешении.

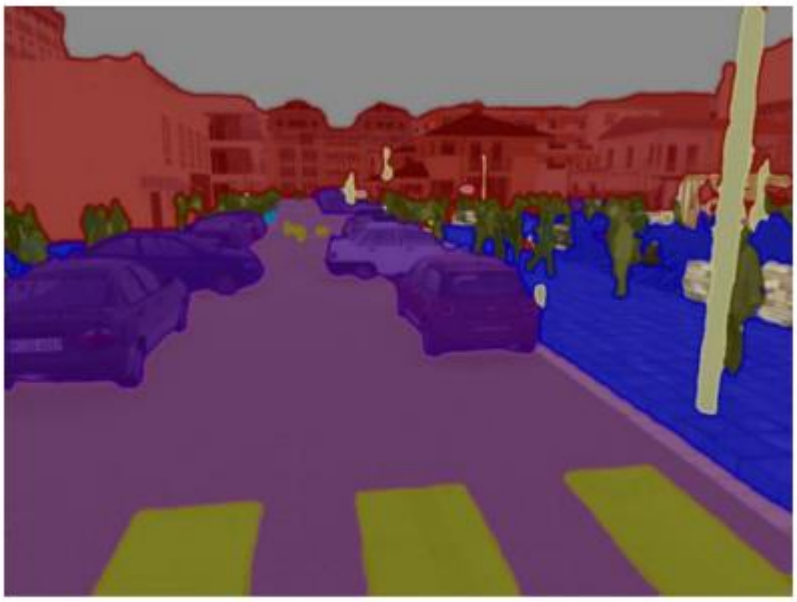


Рис. 1. Результат задачи семантической сегментации.

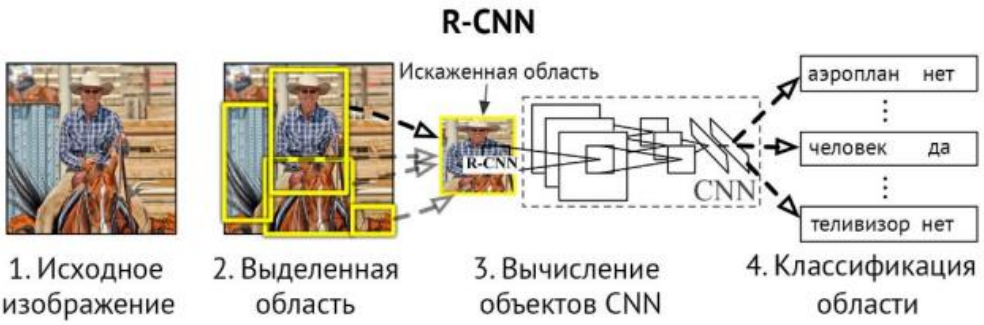


Рис. 2. Архитектура R-CNN

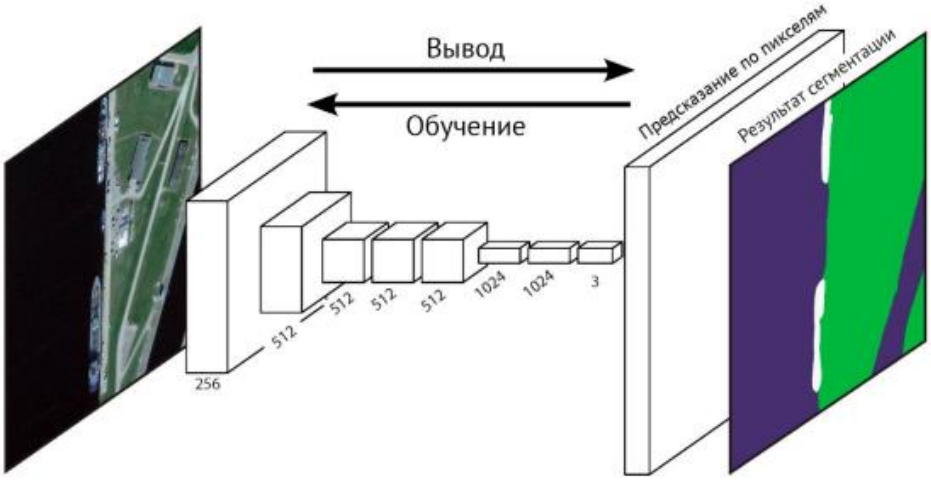


Рис. 3. Архитектура FCN

Рис. 4. Обучение Boxsup

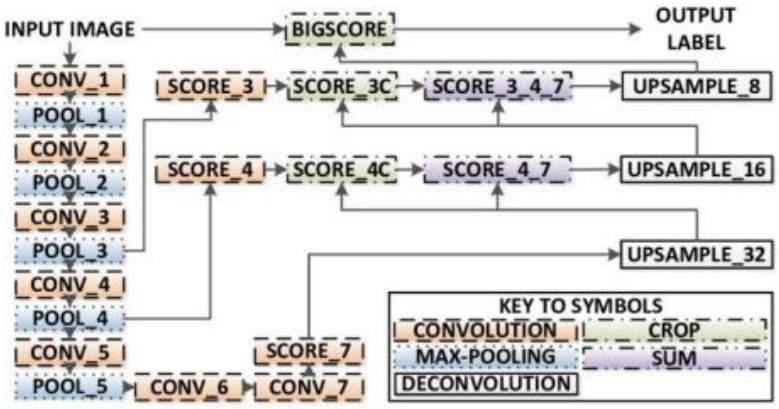


Рис. 5. Архитектура FCN-8

Рис. 6. Результат распознавания нейронной сети FCN-8

Часть II

Тема《АНАЛИЗ СОВРЕМЕННЫХ МЕТОДОВ СЕГМЕНТАЦИИ ИЗОБРАЖЕНИЙ》

1. **Сегментация изображения:**

- *Определение:* Сегментация изображений - это процесс разделения изображения на несколько частей или сегментов для удобства анализа. Цель состоит в том, чтобы выделить объекты или области интереса на изображении и отделить их от фона или других объектов.
- *Методы:* Существует несколько методов сегментации изображений, включая традиционные методы, основанные на цифровой обработке изображений, а также современные методы, использующие глубокое обучение.

2. **Типы задач сегментации:**

- *Семантическая сегментация:* В этой задаче каждому пикселю изображения присваивается метка, которая указывает на его семантическую принадлежность к определенному классу объектов (например, "машина", "человек", "дерево").
- *Сегментация экземпляра:* Здесь целью является выделение каждого отдельного объекта на изображении и присвоение каждому объекту уникальной метки.
- *Паноптическая сегментация:* Этот тип сегментации сочетает в себе как семантическую сегментацию, так и сегментацию экземпляра, обеспечивая информацию как о классе объекта,

так и об уникальной идентификации каждого объекта на изображении.

3. **Сферы применения сегментации изображений:**

- *Медицинская область:* Сегментация изображений широко используется в медицинской области для анализа медицинских изображений, таких как снимки МРТ, КТ и рентгеновские снимки. Это помогает в диагностике различных заболеваний и патологий.
- *Автомобильная промышленность:* Сегментация изображений играет ключевую роль в разработке систем восприятия для автономных транспортных средств, помогая автомобилям распознавать дорожные объекты, пешеходов и другие автомобили.
- *Видеонаблюдение и безопасность:* Сегментация изображений применяется для анализа видеопотока с камер наблюдения, позволяя системам безопасности обнаруживать и отслеживать объекты и инциденты.
- *Картирование и разведка:* В области картирования и разведки сегментация изображений помогает анализировать изображения, полученные с помощью дронов или спутников,

для обнаружения объектов или изменений в окружающей среде.

4. **Традиционные методы сегментации:**

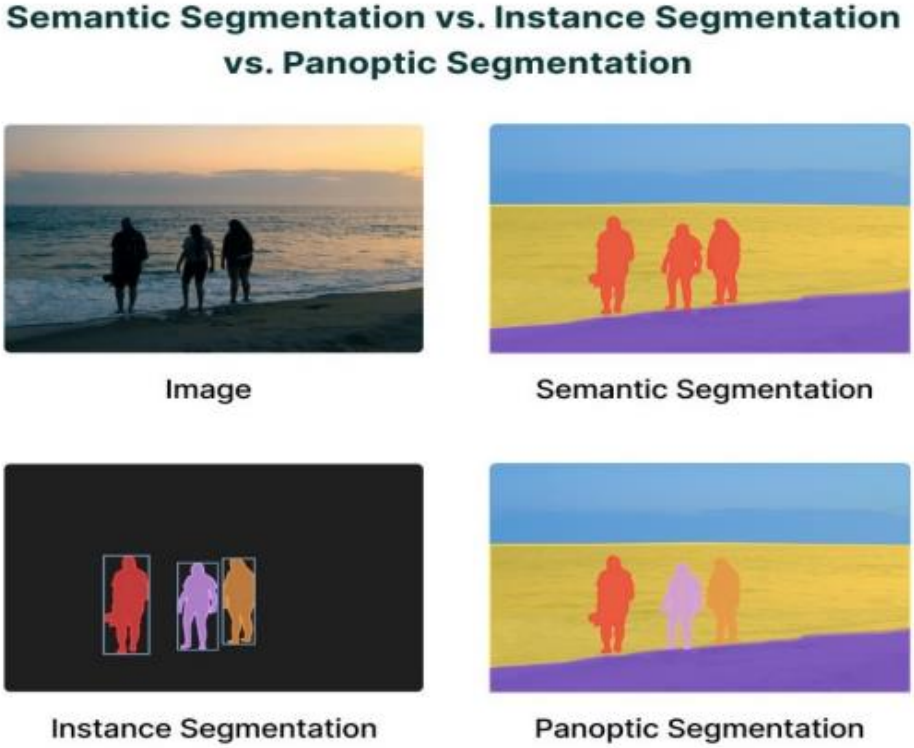
- *Методы на основе порога:* Пиксели изображения разделяются на два класса на основе определенного порогового значения яркости или цвета.
- *Сегментация по регионам:* Изображение разделяется на регионы на основе связности пикселей, которые имеют схожие свойства (например, яркость, цвет).
- *Сегментация краев:* Этот метод направлен на выделение границ объектов на изображении путем обнаружения резких переходов в интенсивности пикселей.
- *Сегментация на основе кластеризации:* Пиксели изображения группируются в кластеры на основе их сходства, используя методы кластерного анализа, такие как k-средних или иерархическая кластеризация.

5. **Методы на основе глубокого обучения:**

- *Архитектуры нейронных сетей:* Современные методы сегментации изображений, основанные на глубоком обучении,

включают в себя различные архитектуры нейронных сетей, такие как U-Net, SegNet и DeepLab.

- *Сверточные нейронные сети (CNN):* CNN широко используются для сегментации изображений из-за их способности эффективно изучать признаки изображений на разных уровнях абстракции.
- *Применение предобученных моделей:* Использование предварительно обученных моделей, обученных на больших наборах данных, позволяет достичь высокой производительности на новых задачах сегментации с минимальным объемом обучающих данных.


Semantic Segmentation vs. Instance Segmentation
vs. Panoptic Segmentation
Image
Semantic Segmentation
Instance Segmentation
Panoptic Segmentation

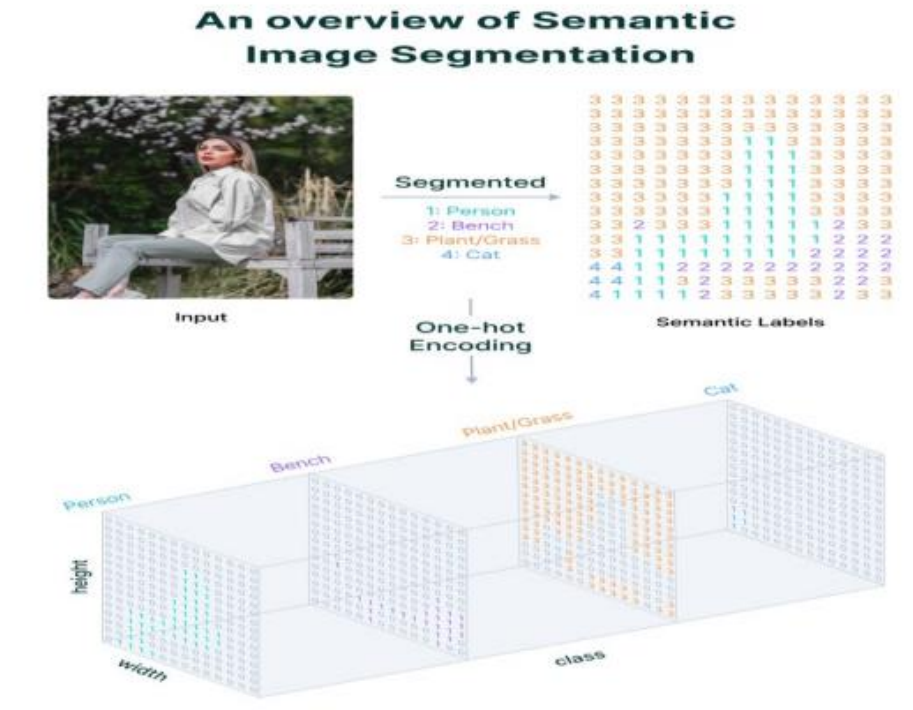

An overview of Semantic
Image Segmentation
Input
Segmented
1: Person
2: Bench
3: Plant/Grass
4: Cat
Semantic Labels
One-hot
Encoding
Person
Bench
Plant/Grass
Cat
height
width
class

Рисунок 3

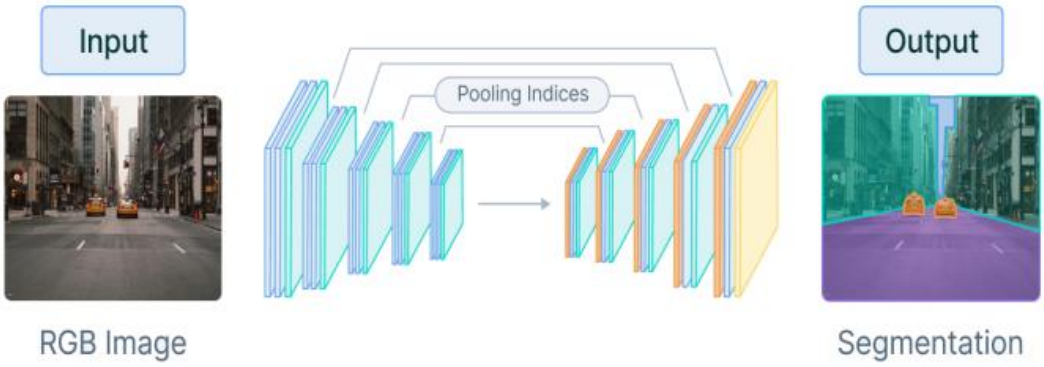


Рисунок 4

3. Практическая часть

[13]:
```python
import torch
import torch.nn as nn
import torch.optim as optim
from torch.utils.data import DataLoader
from torchvision.transforms import transforms
from torchvision.datasets import ImageFolder
from torchvision.utils import save_image
from PIL import Image
import os
import matplotlib.pyplot as plt
```

...

[14]:
```python
# 定义自定义数据集类
class CustomDataset(torch.utils.data.Dataset):
    def __init__(self, root_images, root_masks, transform=None):
        self.root_images = root_images
        self.root_masks = root_masks
        self.transform = transform
        self.images = os.listdir(root_images)
        self.masks = os.listdir(root_masks)

    def __len__(self):
        return len(self.images)

    def __getitem__(self, idx):
        img_name = os.path.join(self.root_images, self.images[idx])
        mask_name = os.path.join(self.root_masks, self.masks[idx])
        image = Image.open(img_name).convert("RGB")
        mask = Image.open(mask_name).convert("L")

        if self.transform:
            image = self.transform(image)
            mask = self.transform(mask)

        return image, mask
```

[15]:
```python
# 定义U-Net模型
class UNet(nn.Module):
    def __init__(self):
        super(UNet, self).__init__()
        # 定义U-Net的编码器部分和解码器部分
        # 编码器部分
        self.encoder = nn.Sequential(
            nn.Conv2d(3, 64, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.Conv2d(64, 64, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(kernel_size=2, stride=2)
        )
        # 解码器部分
        self.decoder = nn.Sequential(
            nn.Conv2d(64, 128, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.Conv2d(128, 128, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.ConvTranspose2d(128, 64, kernel_size=2, stride=2)
        )
        # 最后的输出层
        self.final_layer = nn.Conv2d(64, 1, kernel_size=1)

    def forward(self, x):
        # 编码器部分
        x1 = self.encoder(x)
        # 解码器部分
        x2 = self.decoder(x1)
        # 最后的输出层
        x_out = self.final_layer(x2)
        return x_out
```

[16]:
```python
# 定义数据预处理的转换
transform = transforms.Compose([
    transforms.Resize((256, 256)),
    transforms.ToTensor()
])
```

[17]:
```python
# 定义数据预处理的转换
transform = transforms.Compose([
    transforms.Resize((256, 256)),
    transforms.ToTensor()
])
```

[18]:
```python
# 加载数据集
root_images = "C:/Users/L C H/Desktop/MMO/作业/data/images"
root_masks = "C:/Users/L C H/Desktop/MMO/作业/data/masks"
dataset = CustomDataset(root_images, root_masks, transform=transform)
dataloader = DataLoader(dataset, batch_size=4, shuffle=True)
```

[19]:
```python
# 实例化U-Net模型
model = UNet()
```

[20]:
```python
# 定义损失函数和优化器
criterion = nn.BCEWithLogitsLoss()  # 二元交叉熵损失函数
optimizer = optim.Adam(model.parameters(), lr=0.001)
```

[21]:
```
num_epochs = 10
for epoch in range(num_epochs):
    for images, masks in dataloader:
        optimizer.zero_grad()
        outputs = model(images)
        loss = criterion(outputs, masks)
        loss.backward()
        optimizer.step()
    print(f'Epoch [{epoch+1}/{num_epochs}], Loss: {loss.item():.4f}')
```

```
Epoch [1/10], Loss: 0.0181
Epoch [2/10], Loss: 0.0113
Epoch [3/10], Loss: 0.0212
Epoch [4/10], Loss: 0.0195
Epoch [5/10], Loss: 0.0140
Epoch [6/10], Loss: 0.0185
Epoch [7/10], Loss: 0.0142
Epoch [8/10], Loss: 0.0191
Epoch [9/10], Loss: 0.0156
Epoch [10/10], Loss: 0.0142
```

[22]:
```
# 保存模型
torch.save(model.state_dict(), 'unet_model.pth')
print('Model saved successfully.')
```

```
Model saved successfully.
```

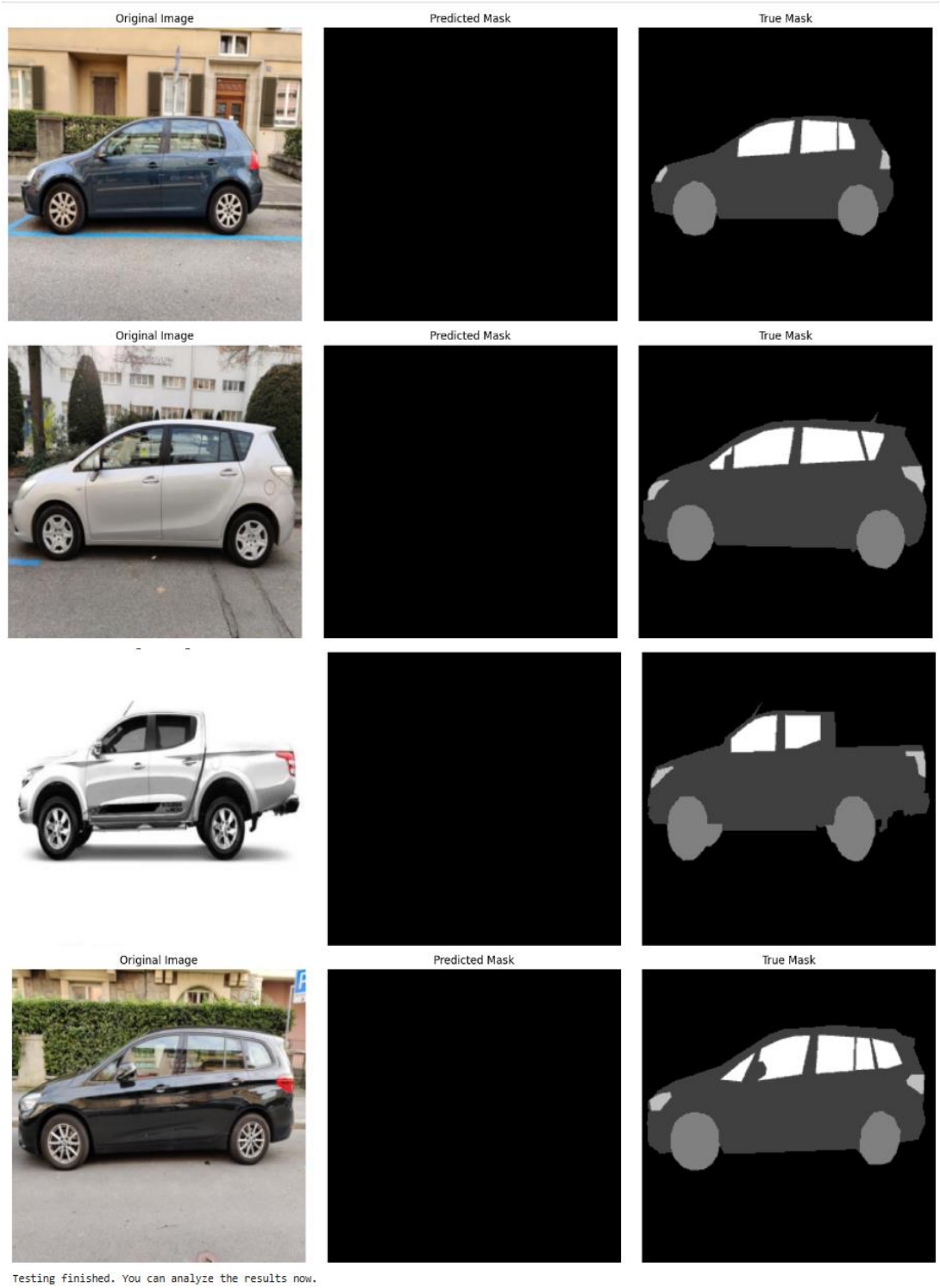
[23]:
```
# 可视化显示原始图像、模型预测的掩码图像和真实掩码图像
def visualize_result(original_images, predicted_masks, true_masks):
    num_images = original_images.size(0)
    fig, axes = plt.subplots(num_images, 3, figsize=(15, 5 * num_images))
    for i in range(num_images):
        axes[i, 0].imshow(original_images[i].permute(1, 2, 0))
        axes[i, 0].set_title('Original Image')
        axes[i, 0].axis('off')

        axes[i, 1].imshow(predicted_masks[i][0].cpu(), cmap='gray')
        axes[i, 1].set_title('Predicted Mask')
        axes[i, 1].axis('off')

        axes[i, 2].imshow(true_masks[i][0].cpu(), cmap='gray')
        axes[i, 2].set_title('True Mask')
        axes[i, 2].axis('off')
    plt.tight_layout()
    plt.show()
```

```
# 测试模型
model.eval()
with torch.no_grad():
    for images, masks in dataloader:
        outputs = model(images)
        # 将输出转换为概率图
        outputs = torch.sigmoid(outputs)
        # 将输出二值化为掩码图像
        predicted_masks = (outputs > 0.5).float()
        # 显示结果
        visualize_result(images, predicted_masks, masks)
        break  # 只测试一个批次的数据

print('Testing finished. You can analyze the results now.')
```



4 Заключение

Мы применяем сверточные нейронные сети, такие как U-Net и DeepLab, для сегментации изображений, извлекая признаки с помощью структуры энкодер-декодер. Наши модели обучаются с

использованием различных функций потерь, таких как перекрестная энтропия и функция Dice, чтобы точно определить классы объектов на изображении. Мы также активно используем техники аугментации данных, чтобы расширить обучающий набор и повысить обобщающую способность модели. Разреженная свертка и многомасштабная обработка помогают нам захватывать широкий контекст и обрабатывать объекты разных размеров. Наш подход также включает использование предварительно обученных моделей, чтобы улучшить начальное представление признаков и повысить производительность модели.

5 Список использованных источников

[1] Демин И С, Рыбкин С В. Семантическая сегментация в машинном обучении[J]. Международный студенческий научный вестник, 2019 (1): 51-51.

[2] Лукашик Д В. Анализ современных методов сегментации изображений[J]. Экономика и качество систем связи, 2022, 2: 57.

[3] Касатиков Н Н. Методы нейронных сетей при распознавании образов антропогенных объектов[C]//Состояние и перспективы развития современной науки по направлению «Геоинформационные платформы военного назначения». 2021: 84-93.

[5] Vijay Badrinarayanan, Alex Kendall, SegNet: A Deep Convolutional EncoderDecoder Architecture for Image Segmentation, 2016. – С 14.

[6] Olaf Ronneberger, Philipp Fischer, U-Net: Convolutional Networks for BiomedicalImage Segmentation, 2015. – C. 8.

[7] Liang-Chieh Chen, George Papandreou, DeepLab: Semantic Image Segmentationwith Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs, 2017. –C. 14.

[8] XuhangLiana, YanweiPang, Cascaded hierarchical atrous spatial pyramid poolingmodule for semantic segmentation, 2020.

[9] A. Dadhich Practical COMPUTER VISION: EXTRACT INSIGHTFUL INFORMATIONFROM IMAGES USING TENSORFLOW, KERAS, AND OPENCV // Packt Publishing Ltd.2018. c. 57-67

[10] J. C. Russ THE IMAGE PROCESSING HANDBOOK // CRC Press. 2016. c. 43-57