Тема: Обучение нейронной сети PointNet++ для семантической сегментации облака точек.

Цель работы:

1. Освоить принципы работы архитектуры PointNet++.
2. Приобрести практические навыки подготовки данных 3D-объектов для семантической сегментации.
3. Проанализировать качество предсказаний модели.

Теоретическое введение

Семантическая сегментация облака точек — задача присвоения каждой точке 3D-облака метки класса (например, "стена", "пол", "стол", "кресло"). Это ключевая проблема в робототехнике, автономном вождении и дополненной реальности.

PointNet++ — развитие архитектуры PointNet, которая напрямую обрабатывает неупорядоченные наборы точек. Ключевые усовершенствования:

1. **Иерархическая структура:** Сеть организует точки в вложенные группы, создавая локальные области с разной степенью детализации (sampling & grouping).
2. **Извлечение локальных признаков:** Внутри каждой локальной области применяется мини-сеть, подобная PointNet, для извлечения признаков, учитывающих локальную геометрию.
3. **Интерполяция признаков (Upsampling):** Для задачи сегментации признаки после самых глубоких слоев интерполируются обратно на исходные точки, комбинируя локальную и глобальную информацию.

Архитектура позволяет модели эффективно учитывать не только глобальную структуру объекта, но и локальные особенности, что критически важно для точной сегментации.

Оборудование и программное обеспечение

1. **Аппаратное обеспечение:** Компьютер с GPU (желательно NVIDIA с поддержкой CUDA).
2. **Программное обеспечение:**
   - ОС: Linux или Windows.
   - Язык программирования: Python 3.8+.
   - Фреймворк: PyTorch.

- Библиотеки: `torch`, `torch.nn`, `numpy`, `open3d` или `trimesh` для визуализации, `scikit-learn` для метрик, `tqdm`.

3. **Датасет:**
   - Stanford 3D Indoor Spaces (S3DIS) . Датасет содержит сканы зданий с разметкой 13 классов (потолок, пол, стена, балка, колонна, окно, дверь, стол, стул, диван, книжная полка, доска, clutter). Ссылка на скачивание предоставляется преподавателем или на официальном ресурсе разработчика.
   - Дополнительный датасет предоставляемый преподавателем.

Порядок выполнения работы

Часть 1: Подготовка среды и данных

1. Установите необходимые библиотеки
2. Загрузите и подготовьте датасет S3DIS (Area_1) или датасет предоставленный преподавателем. Данные должны быть преобразованы в массив точек `(N, 6)`, где `N` — количество точек, а 6 каналов — это `(x, y, z, r, g, b)` и, опционально, нормали (минимальный случай (x,y,z)).
3. Реализуйте свой класс Dataset (самостоятельно разделите обучающую, валидационную и тесовые выборки).

Часть 2: Архитектура модели и обучение

1. Реализуйте или используйте готовую реализацию PointNet++ для сегментации
2. Определите функцию потерь. Для задачи семантической сегментации с несколькими классами используйте **Cross-Entropy Loss**. Учтите возможный дисбаланс классов в данных (можно использовать веса классов).
3. Настройте процесс обучения:
   - Оптимизатор: `Adam` с начальным learning rate = 0.001.
   - Планировщик (scheduler): `StepLR` для уменьшения learning rate каждые 20 эпох.
   - Количество эпох: 50-100 (можно увеличить).
   - Размер батча: Максимально возможный для вашего GPU (начните с 8-16) или CPU.
4. **Задание:** Запустите обучение модели. В процессе обучения записывайте значения функции потерь на тренировочной и валидационной выборках для построения графиков (по возможности и наличию навыков работы использовать TensorBoard).

Часть 3: Валидация и тестирование

1. Реализуйте функцию для расчета метрик качества сегментации на валидационном наборе:
    - **Overall Accuracy (OA):** Доля правильно классифицированных точек от общего числа.
    - **Mean Intersection over Union (mIoU):** Для каждого класса вычисляется IoU = TP / (TP + FP + FN), затем берется среднее значение по всем классам.
2. **Задание:** После завершения обучения:
    - Постройте графики обучения (train/val loss).
    - Рассчитайте OA и mIoU на тестовом наборе данных .
    - Визуализируйте результаты сегментации для нескольких тестовых блоков. Используйте `open3d` для отображения исходного облака (цвет RGB) и предсказанных классов (цветовая палитра по классам) или любую другую библиотеку или модуль для визуализации.

Часть 4: Анализ результатов

1. Проанализируйте график обучения. Присутствуют ли признаки переобучения (сильное расхождение train и val loss)? Как изменялась точность с ростом эпох?
2. Проанализируйте таблицу IoU по классам.
    - Какие классы сегментируются лучше всего (например, "пол", "потолок")? Почему?
    - Какие классы имеют низкий IoU (например, "доска", "диван")? В чем возможные причины (недостаток данных, схожесть геометрии с другими объектами, малый размер)?
3. **Задание:** Предложите и попробуйте реализовать способ улучшения качества модели.

Контрольные вопросы

1. В чем основное ограничение оригинального PointNet, которое преодолевает PointNet++?
2. Каковы основные проблемы при работе с реальными данными 3D-сканирования (на примере S3DIS), влияющие на качество сегментации?
3. Почему метрика mIoU является более информативной по сравнению с Overall Accuracy для задачи семантической сегментации?

Требования к отчету

Отчет должен содержать:

1. Титульный лист.

2. Цель работы.
3. Краткое теоретическое введение (схема архитектуры PointNet++).
4. Описание экспериментальной установки (версии ПО, параметры GPU).
5. **Результаты выполнения:**
    - Графики хода обучения (loss).
    - Таблицы с рассчитанными метриками OA и mIoU на тестовой выборке.
    - Скриншоты визуализации исходных облаков и результатов сегментации (минимум 3 примера).
    - Анализ качества по классам (какие сегментируются хорошо/плохо и почему).
6. Ответы на контрольные вопросы.
7. Выводы по работе, где указать: достигнута ли цель, с какими трудностями столкнулись, какие есть пути улучшения модели.

Справочные материалы

1. https://arxiv.org/pdf/1706.02413
2. Репозиторий с официальной реализацией: charlesq34/pointnet2
3. Пример: PyTorch реализация PointNet++ для сегментации: yanx27/Pointnet_Pointnet2_pytorch