Семантическая сегментация

Екатерина Шалимова, Влад Шахуро

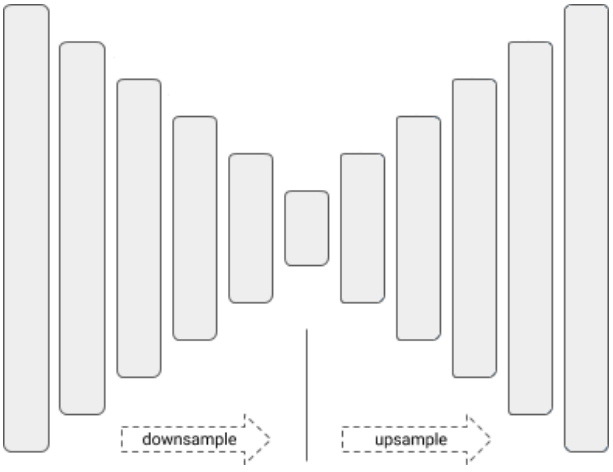
Graphics & Media Lab
Vision Group

В данном задании необходимо реализовать бинарную сегментацию изображений птиц. В задании описываются две модели: базовая, энкодер-декодер, и более продвинутая, U-net, которая является улучшением базовой модели.

1. Энкодер-декодер

Многие современные архитектуры для сегментации изображений имеют в своей основе энкодер-декодер. Рассмотрим его устройство.

Сначала свёрточные слои извлекают из изображения признаки, постепенно понижая разрешение карт признаков, так же, как это делается в сетях для классификации. Это позволяет использовать в качестве инициализации этих слоёв веса сетей, предобученных на задачах классификации, например, Imagenet, что ускоряет обучение. Далее декодер постепенно восстанавливает исходное разрешение (например, с помощью upsampling или транспонированных сверток).

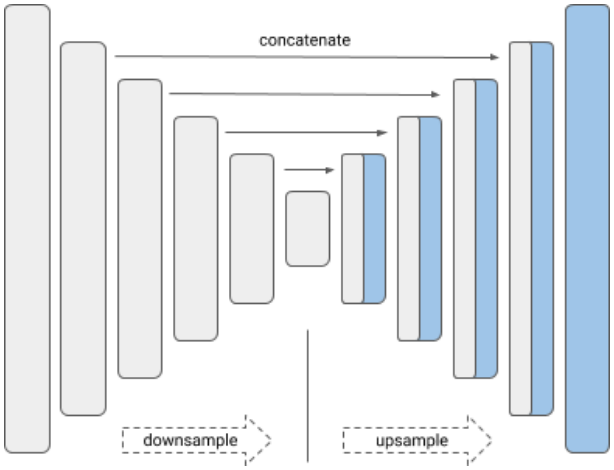


Реализуйте простой энкодер-декодер для сегментации изображений. Сделайте небольшие выборки для обучения и валидации (например, из 200 классов птиц можно выбрать один) и попробуйте

обучить на них нейросеть. 5-10 минут обучения на CPU должно быть достаточно. Если вам кажется, что выбранная вами архитектура обучается слишком медленно, уменьшите количество параметров (например, за счет числа слоёв или карт признаков в слоях). Качество результата также заметно зависит от темпа обучения. На этом этапе вам не требуется высокое качество, главное, чтобы сеть выдавала что-то более осмысленное, чем случайный шум или нули.

2. Архитектура U-Net

U-Net[1] — это достаточно простая, но при этом эффективная архитектура для сегментации изображений. От энкодера-декодера она отличается наличием skip-связей: выходы слоев энкодера конкатенируются с выходами слоёв декодера соответствующих разрешений перед подачей на вход следующему слою декодера.



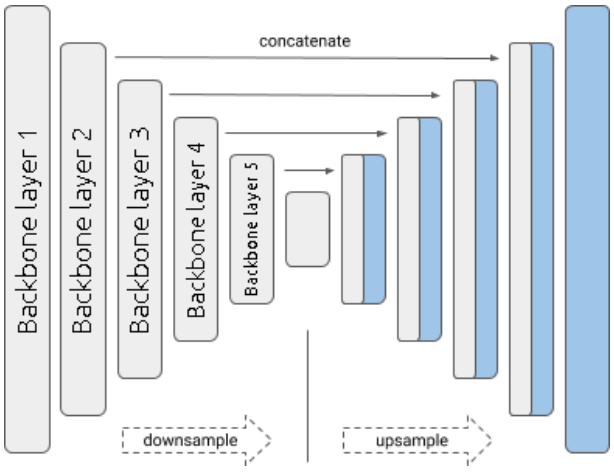
Реализуйте архитектуру U-Net, аналогичную по количеству слоев и карт признаков вашему энкодеру-декодеру из предыдущего пункта (то есть они должны отличаться только наличием skip connections). Обучите полученную модель на той же выборке, что и в предыдущем пункте. Сравните результаты работы энкодера-декодера и U-Net по метрике IoU и визуально.

3. Улучшение сегментации

В этом пункте задания вам необходимо добиться более высокого результата на полном наборе данных. Улучшения базовой архитектуры, которые вам могут понадобиться:

1. Большее число параметров сети.
2. Использование batch normalization или instance normalization[2].
3. Использование предобученных весов: замените энкодер (слои, понижающие разрешение) на первые несколько слоёв сети, предобученной на классификации изображений Imagenet, например, MobileNet или ResNet.

[1] Ronneberger et al. U-net: Convolutional networks for biomedical image segmentation.
[2] Ulyanov et al. Instance normalization: The missing ingredient for fast stylization.



Вам выдается публичная выборка с папками `train` и `test`, которые нужно использовать для обучения и валидации. Разрешается использовать только эти данные для обучения, а также предобученные на ImageNet веса из `tensorflow.keras.applications`. Другие внешние данные и библиотеки с готовыми моделями для сегментации использовать нельзя.

В проверяющую систему необходимо загрузить файл с обученной моделью `segmentation_model.hdf5` и файл `segmentation.py` с реализованными функциями создания и обучения модели и предсказания карты сегментации для изображения. Функция `predict(model, img_path)` предсказывает карту вероятностей класса фон-объект для изображения. Карта вероятностей — это двумерная матрица размера $H \times W$, где H и W — высота и ширина сегментируемого изображения. В каждой ячейке такой матрицы записано вещественное число от 0 до 1 — вероятность принадлежности соответствующего пикселя изображения объекту. Функция обучения нейросети `train_model(train_data_path)` обучает модель. Несмотря на то, что вызов функции закомментирован в тестовом скрипте, она должна быть в решении. Запуск функции на публичной обучающей выборке должен позволять воспроизвести сданную в `hdf5`-файле модель с небольшой погрешностью, связанной со случайностью в процессе обучения. Решения без функции обучения могут быть не засчитаны. Обратите внимание, что изображения как в тренировочной, так и в тестовой выборке могут быть черно-белыми, а не только цветными. Эту ситуацию нужно корректно обрабатывать (например, делать из одноканального изображения трехканальное путем дублирования канала).

Скрипт для тестирования оценивает качество сегментации путем подсчета метрики IoU. Точность IoU на скрытой выборке конвертируется в итоговый балл:

IoU $\geqslant 0.85$ — 10 баллов,

IoU $\geqslant 0.80$ — 9 баллов,

IoU $\geqslant 0.75$ — 8 баллов,

IoU $\geqslant 0.70$ — 7 баллов,

IoU $\geqslant 0.60$ — 6 баллов,

IoU $\geqslant 0.50$ — 5 баллов,

IoU $\geqslant 0.40$ — 4 балла,

IoU $\geqslant 0.30$ — 3 балла,

IoU $\geqslant 0.20$ — 2 балла.