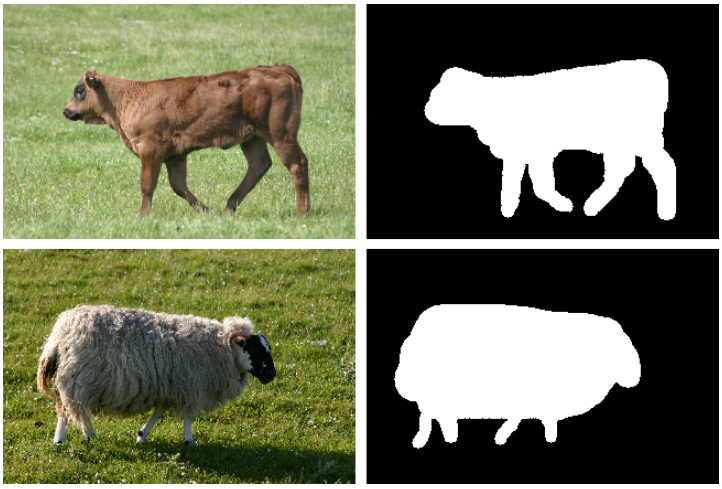
Сегментация изображений

Влад Шахуро, Владимир Гузов

Graphics & Media Lab
Vision Group

Обзор задания

В данном задании предлагается реализовать алгоритм бинарной сегментации изображений на основе графических моделей MRF.



Описание задания

Марковские случайные поля

В контексте задачи сегментации изображений, марковское случайное поле (MRF) — это графическая модель, энергия которой записывается в виде:

$$E(Y) = \sum_{i \in \mathcal{V}} \phi_i(y_i) + \sum_{(i,j) \in \mathcal{E}} \phi_{ij}(y_i, y_j), y_i \in P,$$

где $\mathcal{V}$ — множество индексов переменных, $\mathcal{E}$ — связанные парными потенциалами пиксели, $\phi_i : P \to \mathbb{R}$ — унарные потенциалы, $\phi_{ij} : P \times P \to \mathbb{R}$ — парные потенциалы.

Для нашей задачи бинарной сегментации рассмотрим модель со следующими ограничениями:

1. переменные y_i дискретны и принимают два значения: 0 и 1. То есть, каждая переменная y_i соответствует пикселю на изображении и указывает к какому классу принадлежит объект (0 — фон, 1 — объект)

2. в $\mathcal{E}$ находятся пары соседних друг с другом пикселей (соседним считается ближайший пиксель, находящийся строго слева, справа, над или под текущим).

Постановка задачи

В данном задании необходимо реализовать подсчет унарных и парных потенциалов и, используя готовую библиотеку разреза графов, выполнить сегментацию, определив принадлежность каждого пикселя к фону или объекту.

Унарные потенциалы

Подсчёт унарных потенциалов необходимо реализовать с помощью нейросети, принимающей на вход небольшую область изображения размером от 7×7 до 13×13 и возвращающей вектор длины 2 вероятностей принадлежности центрального пикселя этой области к фону и объекту. Вероятности впоследствии преобразуются в унарные потенциалы по формуле

$$\phi_i(y_i) = -\log(\mathsf{P}(y_i|I_i)),$$

где $\mathsf{P}(y_i|I_i))$ — вышеупомянутая вероятность пикселя I_i принадлежать фону или объекту.

Таким образом, для вычисления унарных потенциалов для каждого пикселя из изображения вырезается квадрат с этим пикселем в центре. На краях изображение предлагается продлить любым известным методом. Нейросеть рекомендуется делать на основе комбинации из нескольких сверточных слоев и, при необходимости, pooling-слоя с добавлением полносвязных слоёв и softmax в конце.

Парные потенциалы

Для подсчета парных потенциалов предлагается использовать обобщенную модель Поттса, которая поощряет прохождение границ разреза там, где есть скачки цвета:

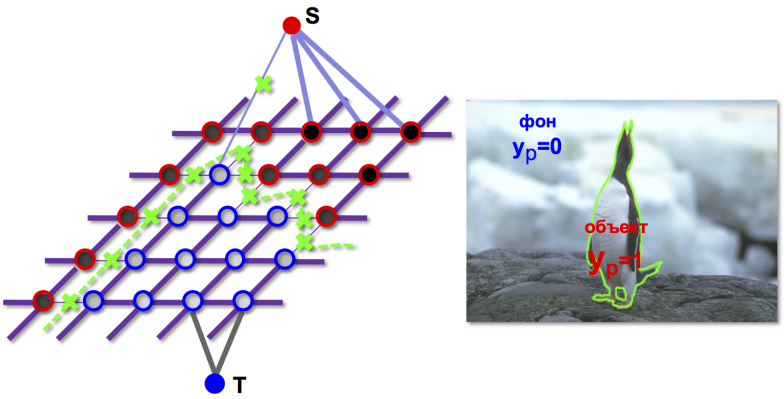
$$\phi_{ij}(y_i, y_j) = (1 - \delta(y_i, y_j))\psi_{ij},$$

$$\psi_{ij} = A + B\exp(-\frac{\|I_i - I_j\|^2}{2\sigma^2}),$$

где $\delta(y_i, y_j)$=1, если $y_i = y_j$ и 0 иначе, I_i — пиксель изображения, а A, B и σ необходимо определить.

Разрез графов

Чтобы свести задачу сегментации к задаче на разрез графов, необходимо построить взвешенный граф, вершинам которого соответствуют переменные y_i, причем вершины связаны друг с другом, если соответствующие пиксели на изображении являются соседями, а весами связей являются соответствующие веса парных потенциалов ψ_{ij}. К данному графу добавляются 2 вершины **S** и **T**, каждая из которых связана со всеми ранее определенными вершинами. Весами этих связей являются унарные потенциалы, для связей с **T** это потенциалы $\phi_i(y_i = 0)$, а для **S** это $\phi_i(y_i = 1)$.



В качестве инструмента, реализующего алгоритм разреза графов, используйте библиотеку PyMaxflow.

Некоторые полезные функции из библиотеки:

`g = maxflow.Graph[float]()` — создание графа с весами типа float

`nodes = g.add_grid_nodes(img.shape)` — создание сетки вершин

`g.add_grid_edges(nodes, phi_ij)` — задание весов связей между соседями (можно вызывать несколько раз с разным параметром `structure` для задания нужной структуры связей, примеры можно найти в документации)

`g.add_grid_tedges(nodes, phi_i1, phi_i0)` — задание весов связей вершин **S** и **T** с остальными вершинами графа

`g.maxflow()` — нахождение максимального потока $\Rightarrow$ минимального разреза

`g.get_grid_segments(nodes)` — получение результата

Интерфейс программы, данные и скрипт для тестирования

Необходимо реализовать две функции: `train_unary_model`, обучающую нейросеть, предсказывающую унарные потенциалы и `segmentation`, выполняющую сегментацию изображений.

Функция `train_unary_model` возвращает готовую модель нейросети, а `segmentation` — список из N двумерных массивов numpy, в которых содержатся метки пикселей (1 — объект, 0 — фон) соответствующих изображений, где N — количество поданных картинок.

Скрипт для тестирования `segmentation_test` принимает на вход директории с тренировочной и тестовой выборками, обучает нейросеть на тренировочных данных и подсчитывает средний по выборке IoU (Intersection over Union) между разметкой тестовых данных и предсказанием алгоритма. Ключ `-v` сохраняет получившиеся сегментации в указанную папку.

Полезные ресурсы

Библиотека PyMaxflow
Документация к PyMaxflow
Библиотека Keras
Курс Д.П.Ветрова «Графические модели»