

Семантический анализ фотографий с помощью глубоких нейронных сетей

02.03.02 - Фундаментальная информатика и информационные технологии

Выполнил:

студент 4 курса Д. И. Ивахненко

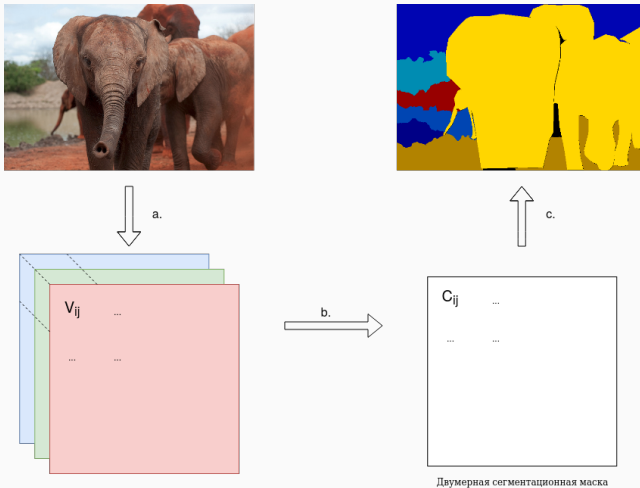
Научный руководитель:

к. ф.-м. н., ст.преп. М. В. Юрушкин

2020 г.

Институт ММиКН им. И.И. Воровича, Южный Федеральный Университет

Общая схема задачи сегментации



a. - Цифровое представление изображения

b. - Сегментация, отображение из RGB вектора V , стоящего на позиции ij , в идентификатор класса C

c. - Визуализация полученной маски

Пример задачи сегментации



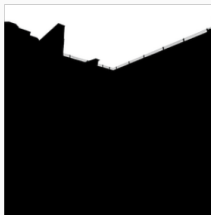
1. Результат работы сети



2. Исходное изображение

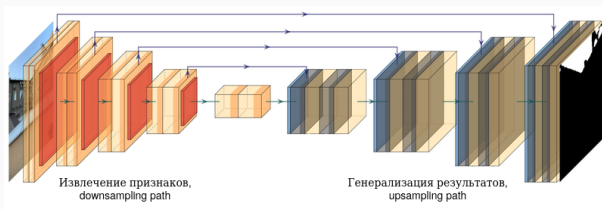


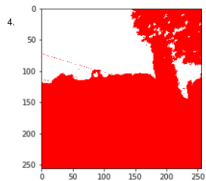
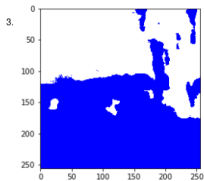
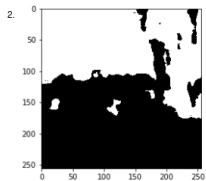
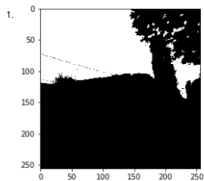
3. Коррекция результата



4. Размеченная маска

Общий вид FCN





Визуализация площадей пересечения и объединения.

1. Образец

2. Сгенерированная маска.

Сеть в предобученном состоянии для наглядной разницы между пересечением и объединением.

3. Пересечение

4. Объединение

Значение метрики для примера: 79.2986

Примеры изображений SkyFinder



Нормализация по пакету



График значения функции потерь к количеству пройденных образцов

Графики:

1. Оранжевый - без нормализации по пакету
2. Серый - с нормализацией по пакету

$$1. \mu_B = \frac{1}{m} \sum_{i=1}^m x_i$$

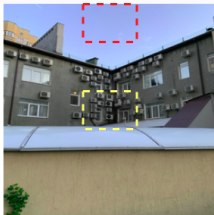
$$2. \sigma_B^2 = \frac{1}{m} \sum_{i=1}^m (x_i - \mu_B)^2$$

$$3. \hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$$

$$4. \text{BN}_{\gamma, \beta}(x_i) = \gamma \hat{x}_i + \beta$$

Красным отмечен регион, из которого случайным образом брался пример 1.

Желтым - пример 2.



```
Pre-softmax state; Size: torch.Size([2, 256, 256]),
1. tensor([-9.2658939, 10.0882759], dtype=torch.float64),
2. tensor([ 4.1472898, -4.4603243], dtype=torch.float64)
```

```
Post-softmax state; Size: torch.Size([2, 256, 256]),
1. tensor([ 0.0000000,  1.0000000], dtype=torch.float64),
2. tensor([ 0.9998173,  0.0001827], dtype=torch.float64)
```

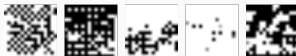
Формула:

$$\text{Softmax}(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

Свойства после применения

- $v_i \in [0, 1] \forall i \in [0, C]$
- $\sum_{i=0}^C v_i = 1$

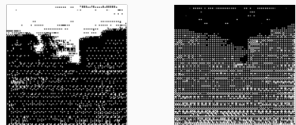
Выходы декодер-части



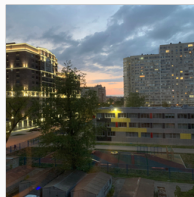
1. Карты признаков после первого декодер-блока.
Размеры выходного блока: 16x16x256



2. Карты признаков после третьего декодер-блока.
Размеры выходного блока: 64x64x64

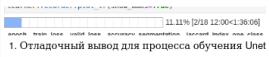


3. Карты признаков после четвертого декодер-блока.
Размеры выходного блока: 128x128x64



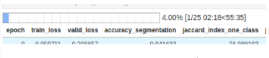
4. Исходное изображение

Выходы декодер-части



epoch	train loss	valid loss	accuracy	segmentation	jaccard index	one class
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

1. Отладочный вывод для процесса обучения U-Net



epoch	train loss	valid loss	accuracy	segmentation	jaccard index	one class
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

2. Отладочный вывод для процесса обучения LinkNet

В 1., 2. в квадратных скобках справа от progress bar процесса обучения указывается прогнозируемое время из расчета время на одну эпоху на количество эпох.
В 3. указаны конфигурации системы, на которой проводились эксперименты по обучению.

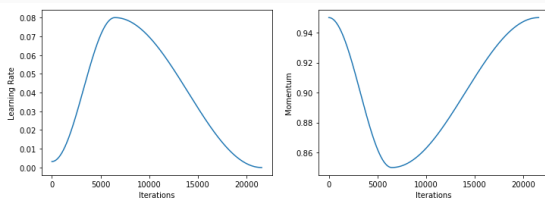
```
=== Software ===
python      : 3.7.7
fastai      : 1.0.61
fastprogress: 0.2.3
torch       : 1.5.0
nvidia driver: 440.33
torch cuda  : 10.1 / is available
torch cudnn  : 7603 / is enabled

=== Hardware ===
nvidia gpus : 1
torch devices: 1
- gpu0      : 6075MB | GeForce GTX 1060 6GB

=== Environment ===
platform    : Linux-5.3.0-51-generic-x86_64-with-debian-buster-sid
distro      : Ubuntu 19.10 eoan
conda env   : DiplomaEnv

3. Конфигурация системы
```

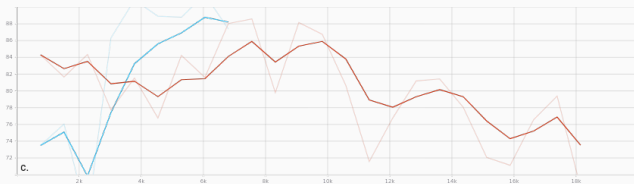
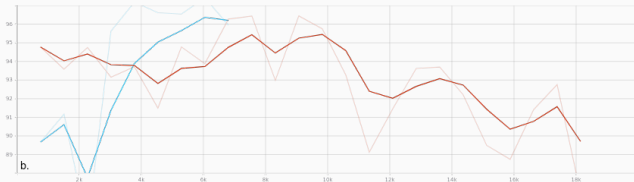
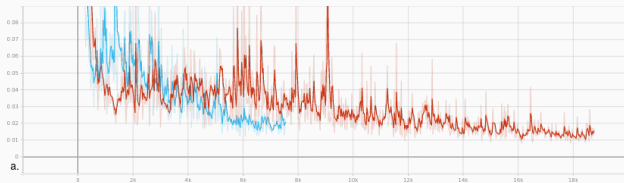
Оптимизация функции потерь



$$\omega_i = \omega_j - \left(\alpha \cdot \frac{\delta f(y, \tilde{y})}{\delta \omega_j} \right) + \gamma \cdot \Delta \omega_k$$

$$f(y, \tilde{y}) = -\frac{1}{n} \sum_{i=0}^n (\tilde{y}_i \cdot \log(y_i) + (1 - \tilde{y}_i) \cdot \log(1 - y_i))$$

Сравнение процессов обучения



Коррекция результатов сети



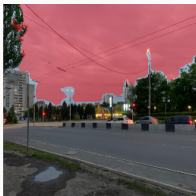
Применение FindContours позволяет устранить замкнутые внутренние ложноположительные регионы, причиной которых могло послужить наличие отражающих поверхностей на изображении (отмечено красным пунктиром).

При этом ложноположительные открытые регионы такой подход исправить не может (оранжевый пунктир).

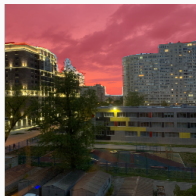
Сравнение результатов сети



1.



2.



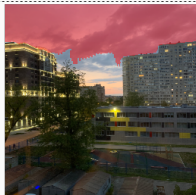
3.



4.



5.



6.