Задачи CV

Шабалин Евгений

Что такое CV

Computer Vision - область компьютерных наук, включающая в себя теорию и технологию создания машин, которые могут анализировать изображения или видео (много изображений подряд).

Задачи CV

С помощью компьютерного зрения люди находят какие-либо паттерны, закономерности и особенности на изображениях.

Выделяют несколько различных разделов.

Computer Vision tasks

Semantic Segmentation

GRASS, CAT, TREE, SKY

No objects, just pixels

Classification + Localization

CAT

Single Object

Object Detection

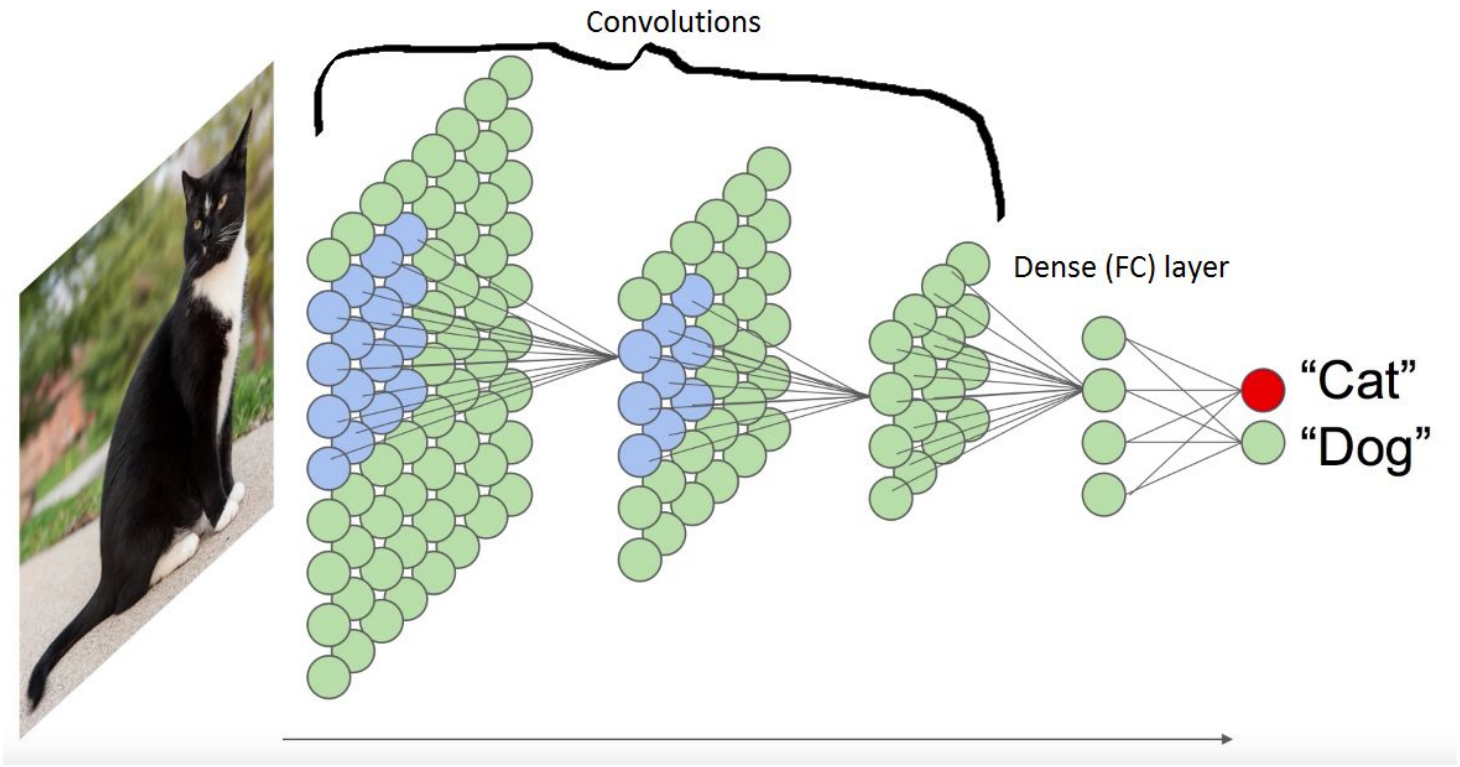
DOG, DOG, CAT

Instance Segmentation

DOG, DOG, CAT

Multiple Object

Классификация



Сегментация

Сегментация — процесс деления изображения на несколько областей (сегментов). Это позволяет упростить изображение, чтобы его было легче анализировать. Можно понимать сегментацию как попиксельную классификацию

Сегментация

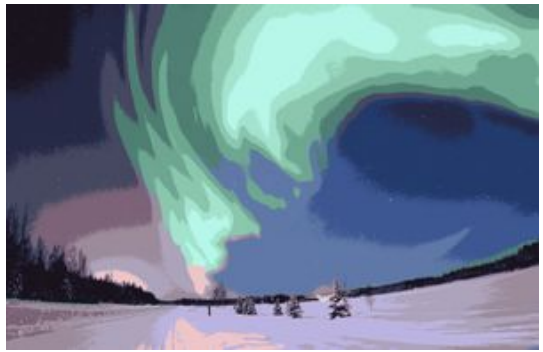
Без нейронных сетей:

- Кластеризация
- Гистограммы
- Выделение границ
- Метод разрастания
- Деление графа

Сегментация

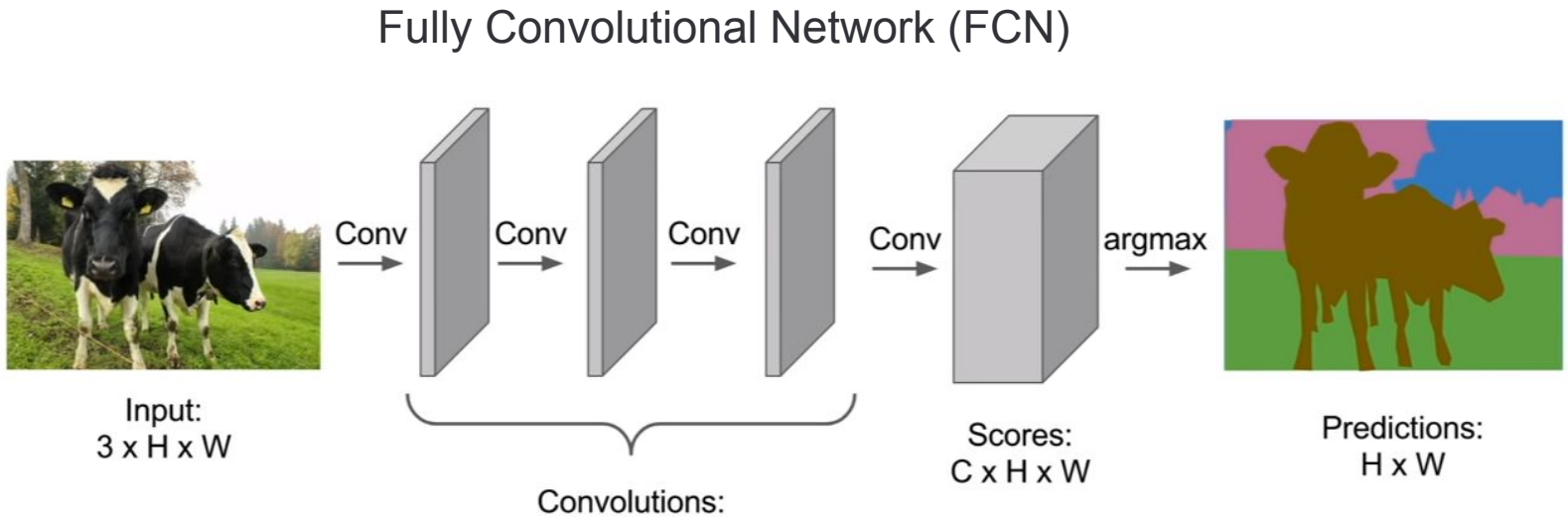
Без нейронных сетей:

- Кластеризация
- Гистограммы
- Выделение границ
- Метод разрастания
- Деление графа



Сегментация

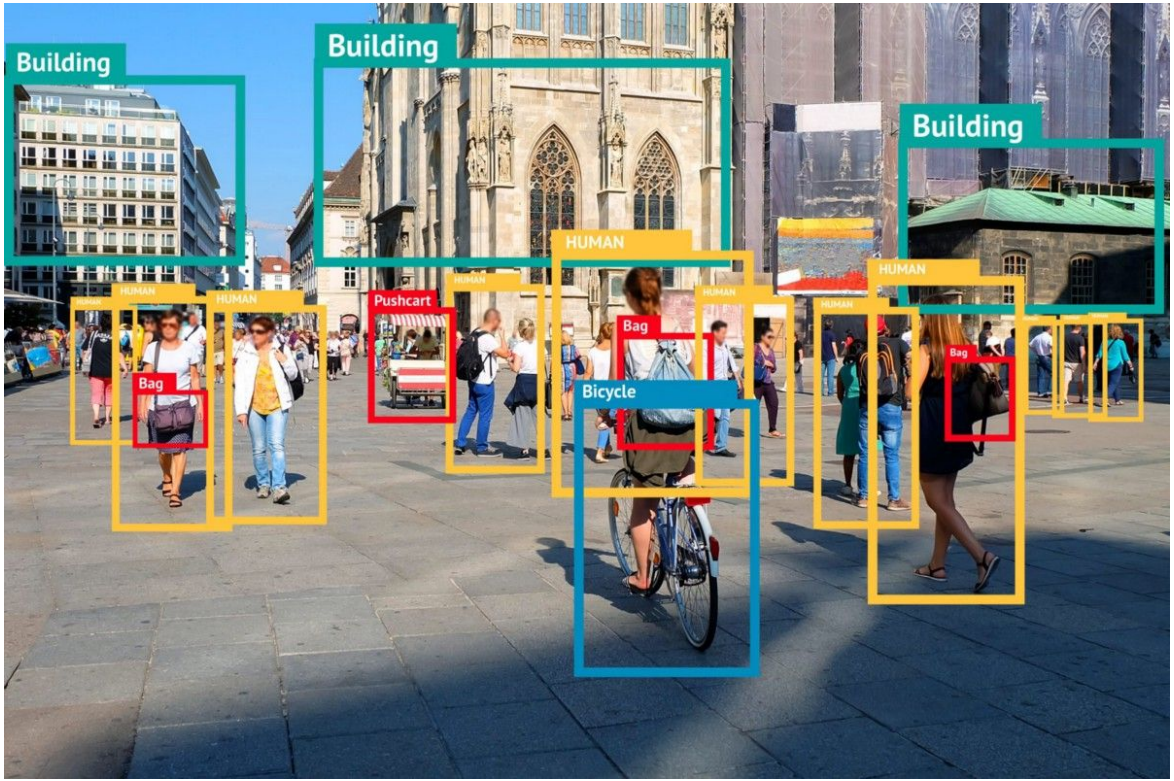
С использованием нейронок:



Детектирование

Задача детектирования заключается в поиске заданных объектов на изображении и их выделении.

Детектирование



Детектирование на видео

Проблема: картинок очень много и обрабатывать отдельно каждую слишком долго.

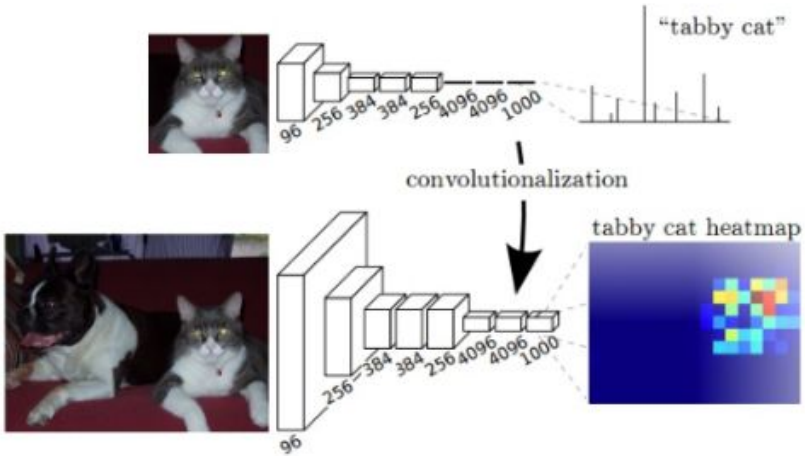
Что делать?

Детектирование на видео

Проблема: картинок очень много и обрабатывать отдельно каждую слишком долго.

Есть 2 варианта: one-shot и two-shot. Первый неточный, второй медленный

One shot detectors



Для каждой ячейки в последнем conv слое предказываем координаты бокса и класс объекта с центром в ячейке.

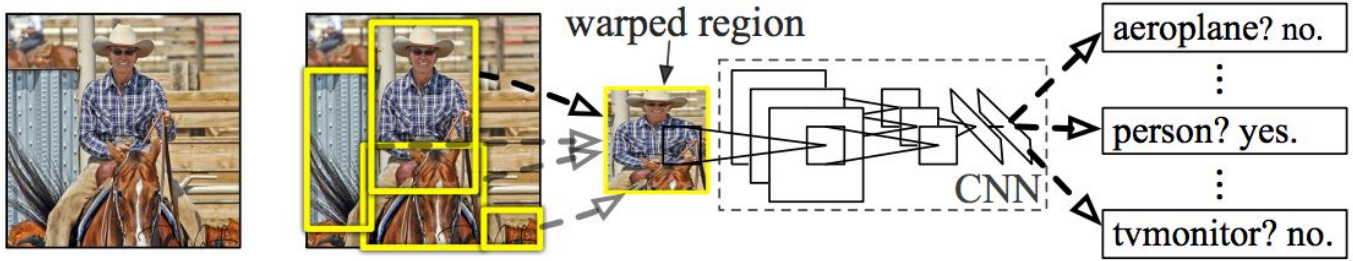
R-CNN

Подход относительно прост: возьмем существующее решение (AlexNet, например) и изменим входные данные. С помощью selective search сгенерируем много различных регионов, в которых может находиться объект и будем смотреть отдельно на них.

R-CNN = Selective Search

+ Classification

R-CNN: *Regions with CNN features*



1. Input image

2. Extract region proposals (~2k)

3. Compute CNN features

4. Classify regions

R-CNN

Хорошо? Не очень

Получились хорошие результаты, но с очень большими затратами по времени (47 секунд на одну картинку для VGG16).

Fast/Faster R-CNN

Fast R-CNN: добавим region of interest (RoI) pooling вместо обычного

Faster R-CNN: вместо определения регионов по изначальному изображению будем это делать после CNN слоев

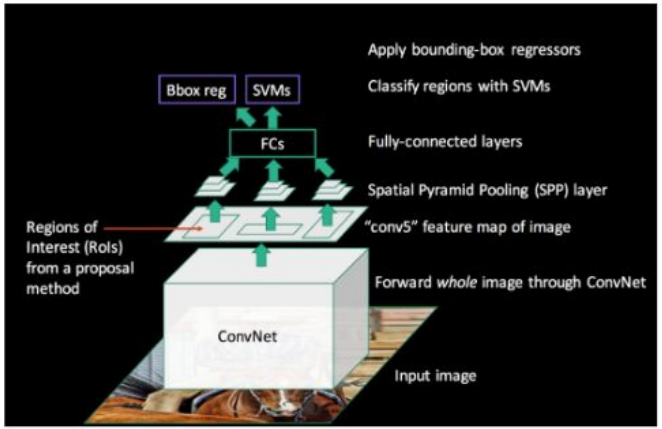
Fast/Faster R-CNN

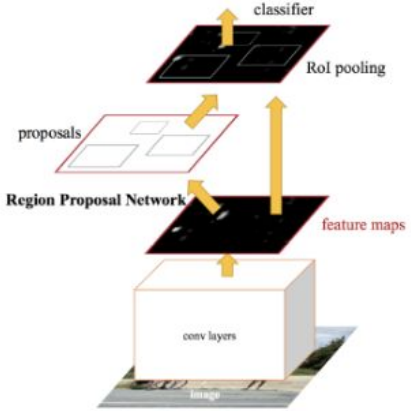
Fast => Faster

Fast R-CNN => Faster R-CNN

Вычисляем proposals самой сетью.

2 секунды => 0.2 секунды (10 раз быстрее)



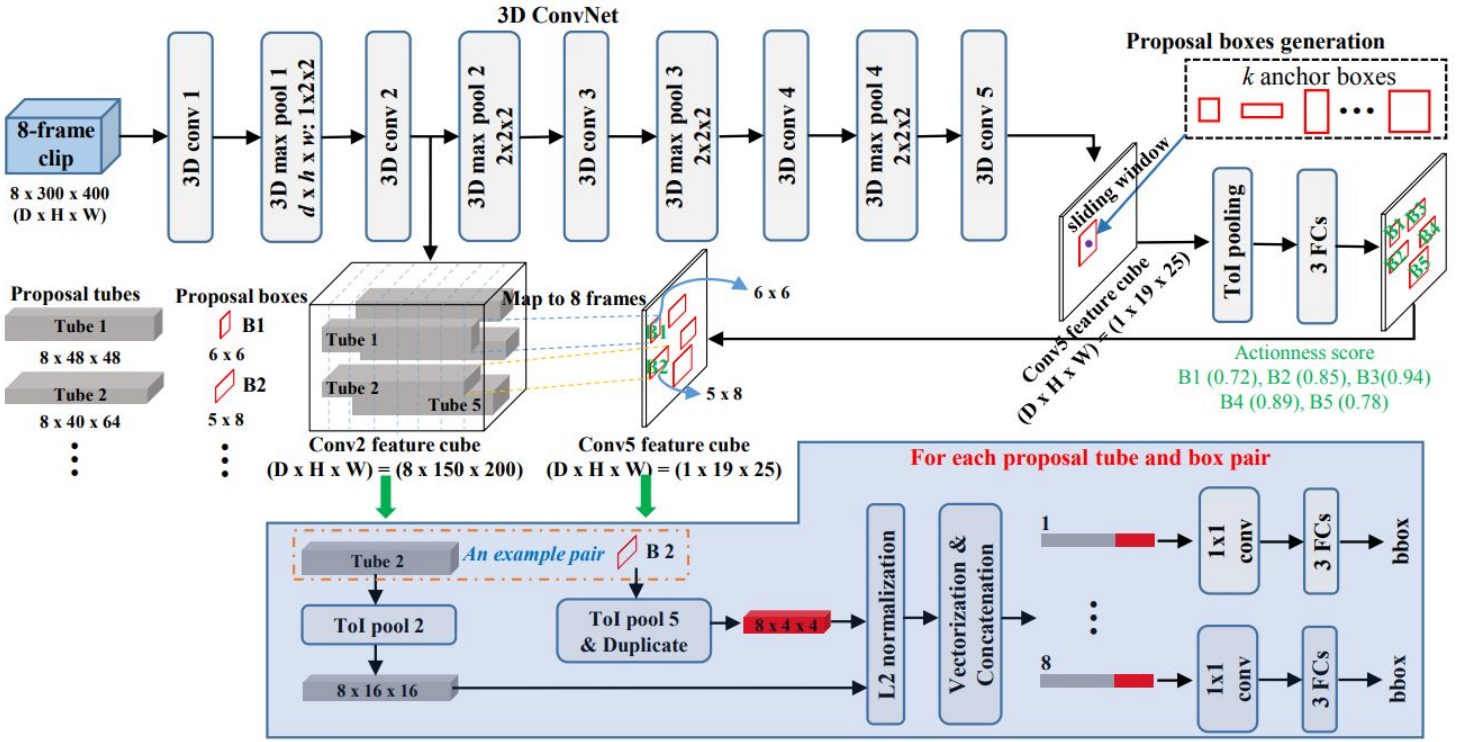


T-CNN

Все предыдущие подходы были разработаны для статичных изображений и никак не использовали факт, что в близких по времени картинках особенности будут расположены в похожих местах.

Взяли все подходы из Faster R-CNN и адаптируем их для клипов по 8 кадров (Tubes).

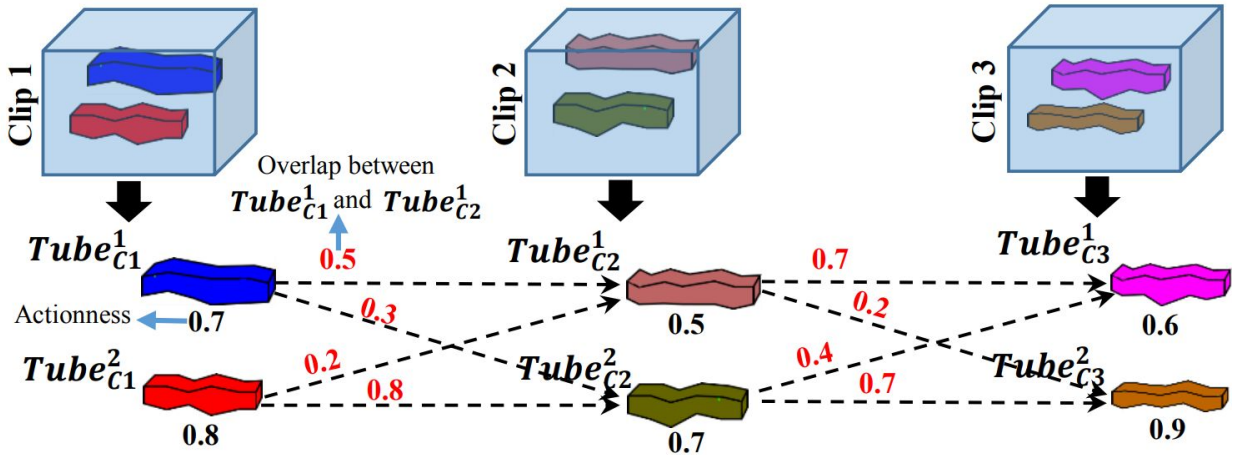
T-CNN


3D ConvNet
8-frame clip
8 x 300 x 400
(D x H x W)
3D conv 1
3D max pool 1
d x h x w: 1x2x2
3D conv 2
3D max pool 2
2x2x2
3D conv 3
3D max pool 3
2x2x2
3D conv 4
3D max pool 4
2x2x2
3D conv 5
Proposal boxes generation
k anchor boxes
sliding window
ToI pooling
3 FCs
Conv5 feature cube
(D x H x W) = (1 x 19 x 25)
Actionness score
B1 (0.72), B2 (0.85), B3(0.94)
B4 (0.89), B5 (0.78)
Proposal tubes
Tube 1
8 x 48 x 48
Tube 2
8 x 40 x 64
Proposal boxes
B1
6 x 6
B2
5 x 8
Map to 8 frames
Tube 1
Tube 2
Tube 5
6 x 6
5 x 8
Conv2 feature cube
(D x H x W) = (8 x 150 x 200)
Conv5 feature cube
(D x H x W) = (1 x 19 x 25)
For each proposal tube and box pair
Tube 2
An example pair
B 2
ToI pool 2
8 x 16 x 16
ToI pool 5
& Duplicate
8 x 4 x 4
L2 normalization
Vectorization &
Concatenation
1
8
1x1
conv
3 FCs
bbox

T-CNN

$$S = \frac{1}{m}\sum_{i=1}^{m} Actionness_i + \frac{1}{m-1}\sum_{j=1}^{m-1} Overlap_{j,j+1} \quad (2)$$

T-CNN



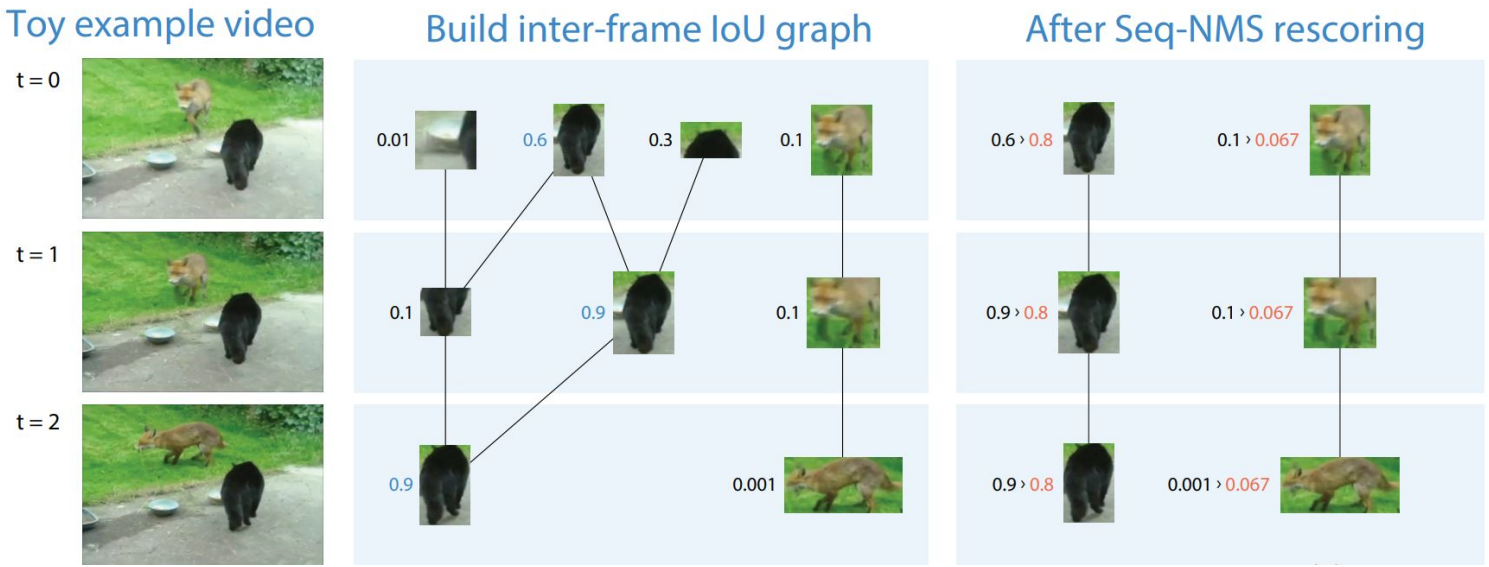
$$S1 = Tube^1_{C1} \rightarrow Tube^1_{C2} \rightarrow Tube^1_{C3} = 1.8/3 + 1.2/2 \quad S2 = Tube^1_{C1} \rightarrow Tube^1_{C2} \rightarrow Tube^2_{C3} = 2.1/3 + 0.7/2$$

$$S3 = Tube^1_{C1} \rightarrow Tube^2_{C2} \rightarrow Tube^1_{C3} = 2.0/3 + 0.7/2 \quad S4 = Tube^1_{C1} \rightarrow Tube^2_{C2} \rightarrow Tube^2_{C3} = 2.3/3 + 1.0/2$$

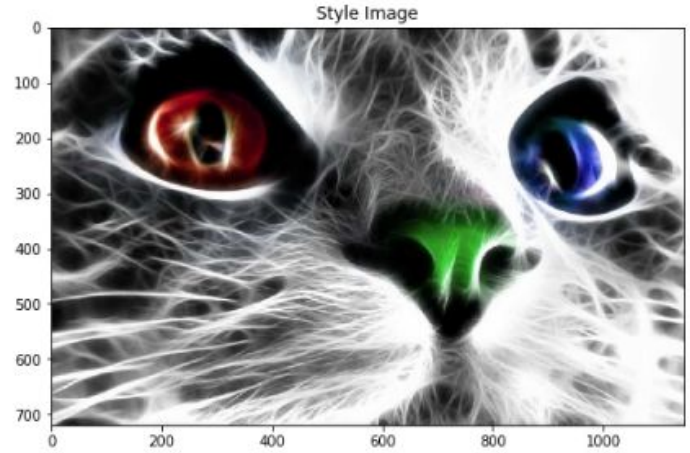
$$S5 = Tube^2_{C1} \rightarrow Tube^1_{C2} \rightarrow Tube^1_{C3} = 1.9/3 + 0.9/2 \quad S6 = Tube^2_{C1} \rightarrow Tube^1_{C2} \rightarrow Tube^2_{C3} = 2.2/3 + 0.4/2$$

$$S7 = Tube^2_{C1} \rightarrow Tube^2_{C2} \rightarrow Tube^1_{C3} = 2.1/3 + 1.2/2 \quad S8 = Tube^2_{C1} \rightarrow Tube^2_{C2} \rightarrow Tube^2_{C3} = 2.4/3 + 1.5/2$$

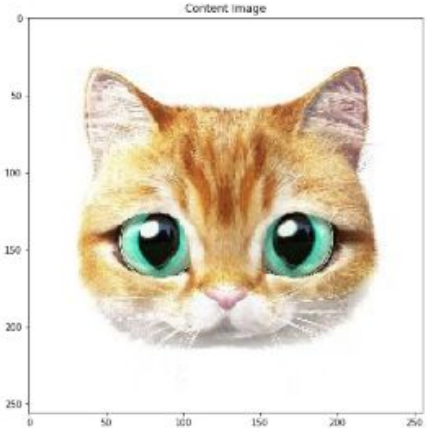
Seq-NMS


Toy example video
t = 0
t = 1
t = 2
Build inter-frame IoU graph
0.01
0.6
0.3
0.1
0.1
0.9
0.1
0.9
0.001
After Seq-NMS rescoring
0.6 › 0.8
0.1 › 0.067
0.9 › 0.8
0.1 › 0.067
0.9 › 0.8
0.001 › 0.067

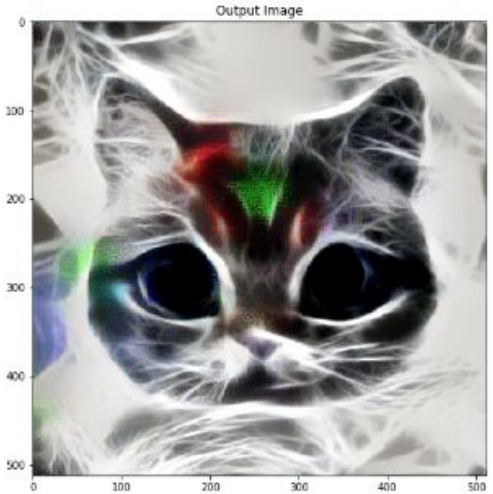
Перенос стиля



+



=



Перенос стиля

Как работает?

Перенос стиля

Начнем с контента

$$\mathcal{L}_{\text{content}}(\vec{p}, \vec{x}, l) = \frac{1}{2} \sum_{i,j} \left(F_{ij}^{l} - P_{ij}^{l} \right)^2 .$$

$\vec{p}$ - исходная картинка, $\vec{x}$ - то, что мы хотим получить

Перенос стиля

Теперь стиль

$$G^l_{ij} = \sum_k F^l_{ik} F^l_{jk}.$$

Перенос стиля

Теперь стиль

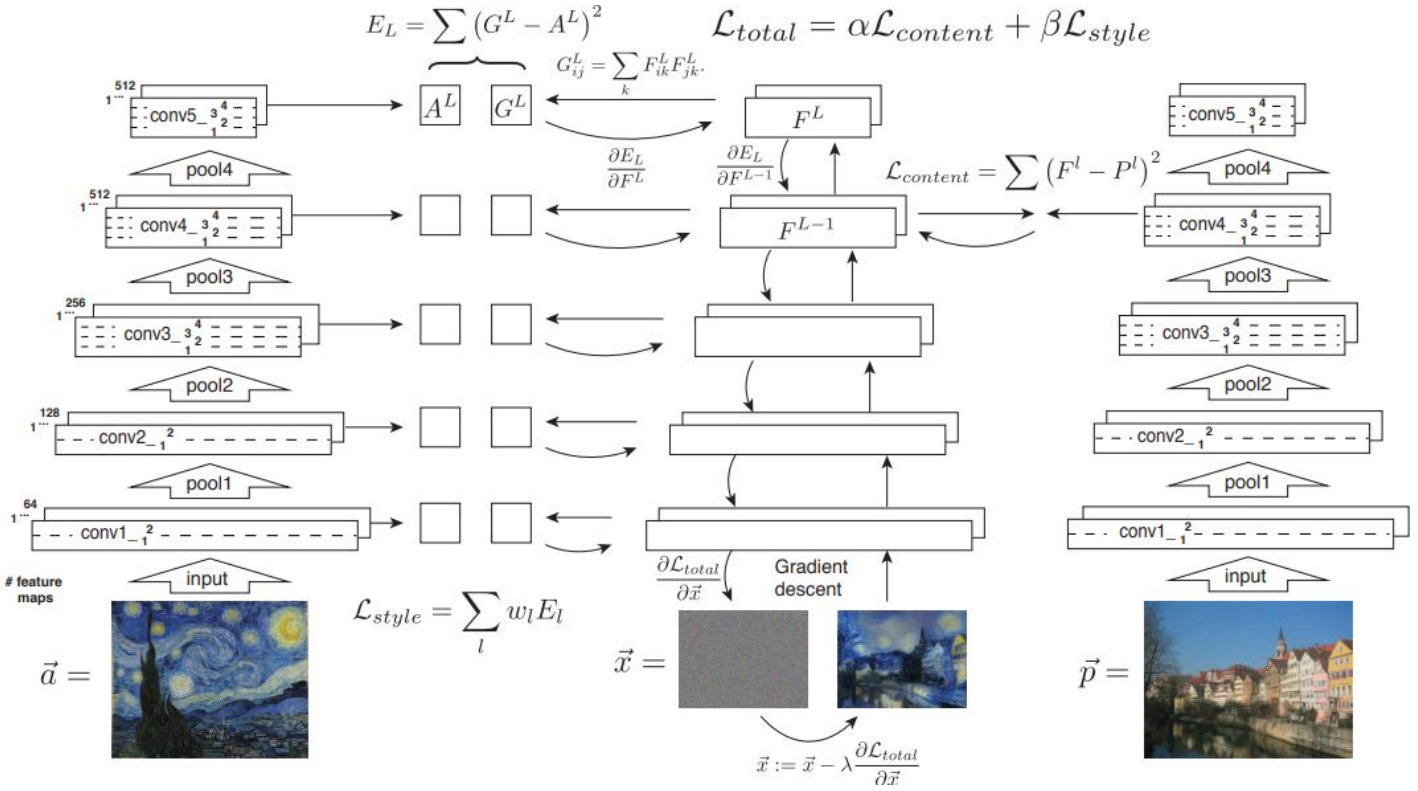
$$G^l_{ij} = \sum_k F^l_{ik} F^l_{jk}.$$

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} \left(G^l_{ij} - A^l_{ij}\right)^2$$

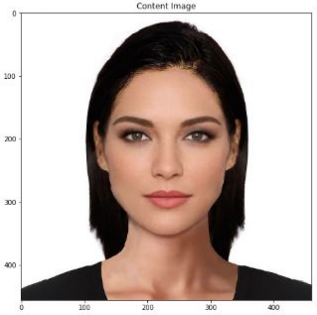
and the total style loss is

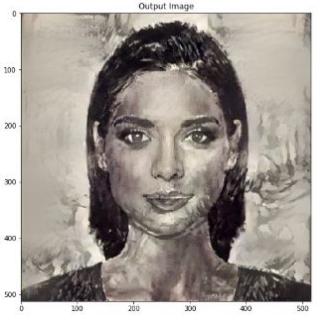
$$\mathcal{L}_{\text{style}}(\vec{a}, \vec{x}) = \sum_{l=0}^{L} w_l E_l,$$

Перенос стиля



Результаты





Что нас ждет

Вопросы

- Чем отличается сегментация от детектирования
- Что такое RoI и как работает
- Как создается представления контента и стиля в задаче переноса стиля и почему по-разному

Источники

- R-CNN https://arxiv.org/abs/1311.2524
- Fast R-CNN https://arxiv.org/abs/1504.08083
- Faster R-CNN https://arxiv.org/abs/1506.01497
- T-CNN https://arxiv.org/pdf/1703.10664.pdf
- Style transfer

  https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Gatys_Image_Style_Transfer_CVPR_2016_paper.pdf