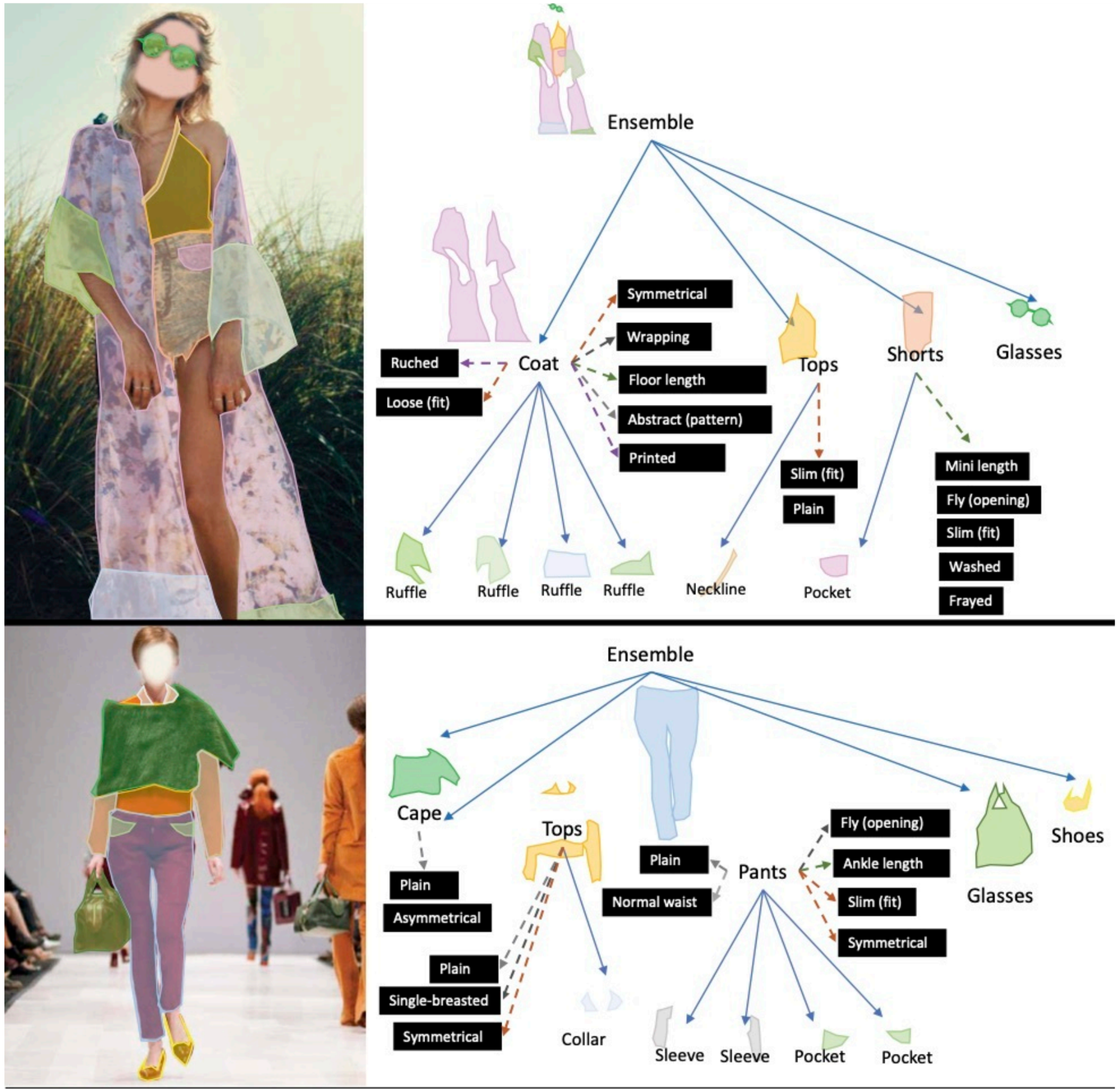
Курсовая работа по DL

iMaterialist (Fashion) 2020 at FGVC7
Fine-grained segmentation task for fashion and apparel
https://www.kaggle.com/c/imaterialist-fashion-2020-fgvc7/overview

Сергеев Кирилл CSC Novosibirsk

Описание конкурса

Дизайнеры знают, что они создают, но что и как люди носят на самом деле? Какие комбинации продуктов используют люди?



Описание конкурса

- В этом конкурсе предлагается разработать алгоритмы, которые помогут сделать важный шаг на пути к автоматическому распознаванию продуктов - точно сегментировать и расставить атрибутивные метки для изображений людей.

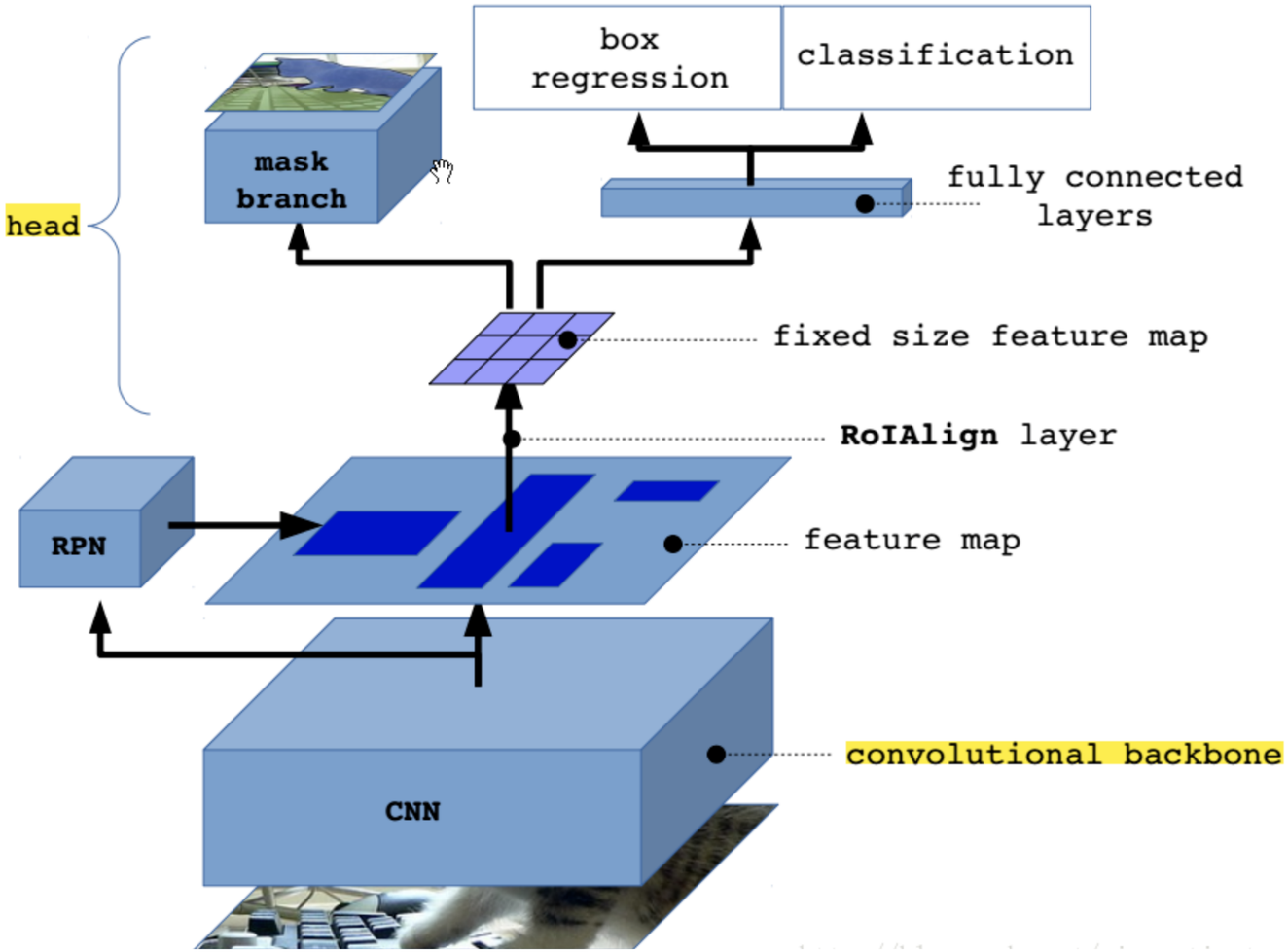
- Визуальный анализ одежды - тема, которой в последние годы уделяется все больше внимания. Способность распознавать изделия одежды и связанные с ними атрибуты по изображениям может улучшить опыт покупок для потребителей и повысить эффективность работы профессионалов в области моды.

Немного о датасете

- Во-первых датасет содержит 46 объектов одежды (27 основных предметов одежды и 19 деталей одежды) и 294 связанных с ними атрибута.
- Во-вторых, в общей сложности мы имеем 50 тысяч изображений одежды (с масками сегментации и мелкими атрибутами) из повседневной жизни, с мероприятий знаменитостей и интернет-магазинов
- Изображения были размечены как экспертами в данной области, так и толпой для более тонкой сегментации.

Подход к решению задачи

Mask-RCNN



Mask-RCNN

- Mask R-CNN развивает архитектуру Faster R-CNN путём добавления ещё одной ветки, которая предсказывает положение маски, покрывающей найденный объект, и, таким образом решает уже задачу instance segmentation
- Маска представляет собой просто прямоугольную матрицу, в которой 1 на некоторой позиции означает принадлежность соответствующего пикселя объекту заданного класса, 0 — что пиксель объекту не принадлежит

Проблемы

Оптимизация обучения №1

LR-warmup-scheduler

```python
def warmup_lr_scheduler(optimizer, warmup_iters, warmup_factor):

    def f(x):
        if x >= warmup_iters:
            return 1
        alpha = float(x) / warmup_iters
        return warmup_factor * (1 - alpha) + alpha

    return torch.optim.lr_scheduler.LambdaLR(optimizer, f)
```

```python
    lr_scheduler = None
    if epoch == 0:
        warmup_factor = 1. / 1000
        warmup_iters = min(1000, len(data_loader) - 1)

        lr_scheduler = utils.warmup_lr_scheduler(optimizer, warmup_iters, warmup_factor)
```

Результат

До:

```
Epoch: [0]  [    0/22812]  eta: 11:23:07  lr: 0.001000  loss: 5.2345 (5.2345)  loss_classifier: 3.7028 (3.7028)  loss_box_re
g: 0.2995 (0.2995)  loss_mask: 1.0204 (1.0204)  loss_objectness: 0.1595 (0.1595)  loss_rpn_box_reg: 0.0523 (0.0523)  time: 1.
7967  data: 0.3920  max mem: 2396
Epoch: [0]  [  100/22812]  eta: 5:53:43  lr: 0.001000  loss: 1.2035 (2.5113)  loss_classifier: 0.2723 (0.7964)  loss_box_reg:
0.1548 (0.2712)  loss_mask: 0.6532 (0.7407)  loss_objectness: 0.1003 (0.6185)  loss_rpn_box_reg: 0.0205 (0.0845)  time: 1.008
5  data: 0.6388  max mem: 2993
Epoch: [0]  [  200/22812]  eta: 5:59:20  lr: 0.001000  loss: 1.3586 (1.9277)  loss_classifier: 0.3603 (0.5817)  loss_box_reg:
0.2314 (0.2464)  loss_mask: 0.6138 (0.6866)  loss_objectness: 0.0904 (0.3572)  loss_rpn_box_reg: 0.0175 (0.0558)  time: 1.088
9  data: 0.7166  max mem: 3047
Epoch: [0]  [  300/22812]  eta: 5:46:38  lr: 0.001000  loss: 1.2408 (1.7162)  loss_classifier: 0.3151 (0.5034)  loss_box_reg:
0.1961 (0.2376)  loss_mask: 0.6199 (0.6640)  loss_objectness: 0.0818 (0.2661)  loss_rpn_box_reg: 0.0175 (0.0450)  time: 1.007
9  data: 0.6387  max mem: 3117
Epoch: [0]  [  400/22812]  eta: 5:41:53  lr: 0.001000  loss: 1.1743 (1.6143)  loss_classifier: 0.3121 (0.4673)  loss_box_reg:
0.2018 (0.2334)  loss_mask: 0.6110 (0.6522)  loss_objectness: 0.0543 (0.2214)  loss_rpn_box_reg: 0.0171 (0.0401)  time: 1.023
1  data: 0.6506  max mem: 3117
Epoch: [0]  [  500/22812]  eta: 5:42:40  lr: 0.001000  loss: 1.2229 (1.5518)  loss_classifier: 0.3045 (0.4440)  loss_box_reg:
0.1955 (0.2293)  loss_mask: 0.6214 (0.6446)  loss_objectness: 0.0951 (0.1959)  loss_rpn_box_reg: 0.0252 (0.0378)  time: 1.017
5  data: 0.6496  max mem: 3196
```

После:

```
Epoch: [0]  [    0/22812]  eta: 9:40:34  lr: 0.000002  loss: 5.9877 (5.9877)  loss_classifier: 4.1939 (4.1939)  loss_box_reg:
0.3680 (0.3680)  loss_mask: 1.3819 (1.3819)  loss_objectness: 0.0359 (0.0359)  loss_rpn_box_reg: 0.0080 (0.0080)  time: 1.527
0  data: 0.1207  max mem: 2449
Epoch: [0]  [  100/22812]  eta: 7:02:26  lr: 0.000102  loss: 1.6350 (2.3292)  loss_classifier: 0.5755 (1.1322)  loss_box_reg:
0.4820 (0.3956)  loss_mask: 0.5038 (0.7173)  loss_objectness: 0.0245 (0.0644)  loss_rpn_box_reg: 0.0148 (0.0197)  time: 1.188
4  data: 0.7880  max mem: 3382
Epoch: [0]  [  200/22812]  eta: 6:33:23  lr: 0.000202  loss: 1.1851 (1.8447)  loss_classifier: 0.4186 (0.8053)  loss_box_reg:
0.3069 (0.3834)  loss_mask: 0.4074 (0.5816)  loss_objectness: 0.0285 (0.0548)  loss_rpn_box_reg: 0.0146 (0.0195)  time: 0.972
2  data: 0.5848  max mem: 3495
Epoch: [0]  [  300/22812]  eta: 6:47:58  lr: 0.000302  loss: 1.2289 (1.6175)  loss_classifier: 0.4192 (0.6699)  loss_box_reg:
0.2890 (0.3549)  loss_mask: 0.4095 (0.5197)  loss_objectness: 0.0487 (0.0525)  loss_rpn_box_reg: 0.0215 (0.0205)  time: 1.393
9  data: 1.0042  max mem: 3495
Epoch: [0]  [  400/22812]  eta: 6:32:16  lr: 0.000402  loss: 1.2621 (1.5154)  loss_classifier: 0.4472 (0.6047)  loss_box_reg:
0.3408 (0.3429)  loss_mask: 0.3661 (0.4958)  loss_objectness: 0.0393 (0.0519)  loss_rpn_box_reg: 0.0164 (0.0201)  time: 0.704
1  data: 0.3167  max mem: 3495
Epoch: [0]  [  500/22812]  eta: 6:19:19  lr: 0.000501  loss: 1.0678 (1.4427)  loss_classifier: 0.3525 (0.5605)  loss_box_reg:
0.2293 (0.3301)  loss_mask: 0.4056 (0.4795)  loss_objectness: 0.0427 (0.0521)  loss_rpn_box_reg: 0.0236 (0.0205)  time: 1.088
9  data: 0.7053  max mem: 3495
```

Оптимизация обучения №2

Mixed precision for faster training on NVIDIA GPUs

Большинство фреймворков глубокого обучения, включая PyTorch, по умолчанию обучают с 32-битной арифметикой с плавающей точкой (FP32). Однако это не является необходимым для достижения полной точности для многих моделей глубокого обучения. В 2017 году исследователи NVIDIA разработали методику обучения со смешанной точностью, которая сочетает формат одинарной точности (FP32) с форматом половинной точности (например, FP16) при обучении сети, и достигает такой же точности, как и обучение с FP32, используя те же гиперпараметры, с дополнительными преимуществами в производительности на графических процессорах NVIDIA:

- Более короткое время обучения
- Меньшие требования к памяти

Результат

До:

```
Epoch: [0]  [    0/22812]  eta: 16:40:59  lr: 0.000002  loss: 5.7308 (5.7308)  loss_classifier: 3.5354 (3.5354)  loss_box_
g: 0.2763 (0.2763)  loss_mask: 1.6508 (1.6508)  loss_objectness: 0.2582 (0.2582)  loss_rpn_box_reg: 0.0102 (0.0102)  time:
6328  data: 1.0621  max mem: 2106
Epoch: [0]  [  100/22812]  eta: 6:21:10  lr: 0.000102  loss: 1.2323 (2.1554)  loss_classifier: 0.3773 (0.9570)  loss_box_r
0.3320 (0.3981)  loss_mask: 0.4912 (0.7102)  loss_objectness: 0.0166 (0.0694)  loss_rpn_box_reg: 0.0114 (0.0207)  time: 0.
4  data: 0.3858  max mem: 3475
Epoch: [0]  [  200/22812]  eta: 6:20:52  lr: 0.000202  loss: 0.9883 (1.7220)  loss_classifier: 0.2931 (0.7058)  loss_box_r
0.2431 (0.3730)  loss_mask: 0.3516 (0.5668)  loss_objectness: 0.0434 (0.0573)  loss_rpn_box_reg: 0.0129 (0.0191)  time: 0.
0  data: 0.3376  max mem: 3475
Epoch: [0]  [  300/22812]  eta: 6:24:12  lr: 0.000302  loss: 1.0992 (1.5351)  loss_classifier: 0.3667 (0.5997)  loss_box_r
0.3028 (0.3520)  loss_mask: 0.4030 (0.5091)  loss_objectness: 0.0375 (0.0536)  loss_rpn_box_reg: 0.0242 (0.0206)  time: 0.
8  data: 0.3707  max mem: 3475
Epoch: [0]  [  400/22812]  eta: 6:23:34  lr: 0.000402  loss: 1.0789 (1.4373)  loss_classifier: 0.3166 (0.5462)  loss_box_r
0.2476 (0.3344)  loss_mask: 0.4343 (0.4831)  loss_objectness: 0.0483 (0.0529)  loss_rpn_box_reg: 0.0139 (0.0207)  time: 1.
2  data: 0.6157  max mem: 3475
```

После:

```
Epoch: [0]  [    0/22812]  eta: 1 day, 2:57:40  lr: 0.000002  loss: 6.0450 (6.0450)  loss_classifier: 3.7122 (3.7122)  loss_b
ox_reg: 0.5111 (0.5111)  loss_mask: 1.7738 (1.7738)  loss_objectness: 0.0304 (0.0304)  loss_rpn_box_reg: 0.0176 (0.0176)  tim
e: 4.2548  data: 2.8593  max mem: 2599
Epoch: [0]  [  100/22812]  eta: 5:22:35  lr: 0.000102  loss: 1.5068 (2.1054)  loss_classifier: 0.5067 (0.9624)  loss_box_reg:
0.4276 (0.3821)  loss_mask: 0.4665 (0.6876)  loss_objectness: 0.0206 (0.0537)  loss_rpn_box_reg: 0.0169 (0.0197)  time: 0.765
9  data: 0.3742  max mem: 3434
Epoch: [0]  [  200/22812]  eta: 5:37:10  lr: 0.000202  loss: 1.1686 (1.7170)  loss_classifier: 0.3825 (0.7169)  loss_box_reg:
0.3098 (0.3732)  loss_mask: 0.3884 (0.5606)  loss_objectness: 0.0302 (0.0473)  loss_rpn_box_reg: 0.0174 (0.0190)  time: 0.940
5  data: 0.5505  max mem: 3434
Epoch: [0]  [  300/22812]  eta: 5:35:04  lr: 0.000302  loss: 1.1321 (1.5586)  loss_classifier: 0.3919 (0.6206)  loss_box_reg:
0.2655 (0.3553)  loss_mask: 0.3997 (0.5130)  loss_objectness: 0.0397 (0.0496)  loss_rpn_box_reg: 0.0175 (0.0201)  time: 0.734
0  data: 0.3489  max mem: 3503
Epoch: [0]  [  400/22812]  eta: 5:31:07  lr: 0.000402  loss: 1.0921 (1.4464)  loss_classifier: 0.3562 (0.5583)  loss_box_reg:
0.2332 (0.3371)  loss_mask: 0.3619 (0.4818)  loss_objectness: 0.0467 (0.0494)  loss_rpn_box_reg: 0.0131 (0.0199)  time: 0.712
6  data: 0.3368  max mem: 3503
```

Некоторые особенности и параметры

- Сеть обучалась 4 эпохи более одних суток
- Размер батча — 2
- Размер входного изображения — (256, 256, 3)
- Число скрытых слоев в Mask-RCNN — 512

Результаты сегментации