Multi-person Tracking

Trần Quang Minh

Hà nội, 22/04/2024

MỤC LỤC

PHẦN 1: TỔNG QUAN VỀ SOURCE YOLOV83-HUMAN

Kiến trúc hiện tại:

- ***Backbone, Neck*** : Yolov8n modify: Các khối CV thay bởi DW+PW và các kích hoạt phức tạp thay thành ReLU
- ***Head (Detector)***: Yolov3 – 3 map predict và sử dụng anchor-box. Đây là 1 hướng xử lý tạm thời vì yolov8 phần head sẽ khác hoàn toàn (anchor-box free, ...).
- ***Tracker*** : Tính toán match box chỉ dựa trên IoU và cũng là 1 hướng mang tính tạm thời, 2 thuật toán phổ thông trong các Traker hiện đại là *Hungarian Algorithm matching boxes* và *Kalman Filter* chưa được thực hiện vì các lí do về độ phức tạp khi chuyển sang code C.
- ***Tập dữ liệu***: COCO2017 (80k), Viettel-VHAC (64k)
- ***Input size***: 320 x 320
- ***Quantization***: PTQ, thực hiện sau train truyền thống.

Các đánh giá hiện tại:

- ***Num params***: 1000022
- ***Valid loss***: 8.5956
- ***Số chu kỳ chạy core Tmoby 8x8*** : ~ 17 triệu chu kì.
- ***FPS test trên ZCU102***: ~ 7 FPS

Các hướng cải thiện:

- ***Thiết kế lại Detertor***: Thiết kế lại đúng chuẩn Head của Yolov8
- ***Viết lại Tracker***: Viết lại theo chuẩn của một số Tracker hiện đại: DeepSort, ByteTrack, Deep OC Sort, Smile Track, ...
- ***Tối ưu training*** : Architecture, thuật toán train, ...
- ***Quantization***: QAT
- ***Viết metrix detect khác***: Viết lại metrix mAP50-95 theo chuẩn để đánh giá model.
- ***Viết metrix cho MOT***: MOTA, HOTA, ...

PHẦN 2: RUN CODES, RUN SCRIPTS

Server : minhtq22@172.21.5.76
Password : Vic@2023

2.1. Source yolov83-human

Đường dẫn: `/data/minhtq22/content-detection/yolov83-human/`

Step 1: **`cd /data/minhtq22/content-detection/yolov83-human/`**
Step 2: **`conda activate minhtq22_38`**
Step 3: Train model và lưu file weights.

Chú ý file label (trong các thư mục dataset) là file txt chứa nhiều dòng, mỗi dòng là nhãn cho mỗi ảnh train, trong đó thông tin trên mỗi dòng có dạng:

```
image_file_path box1 box2 ... boxN
```

trong đó, mỗi box-n sẽ có format:

```
x_min,y_min,x_max,y_max,class_i
```

Ví dụ:

```
path/to/img1.jpg 50,100,150,200,0 30,50,200,120,3
path/to/img2.jpg 120,300,250,600,2
...
```

Source này đã tạo sẵn **`dataset_320`** chứa thư mục ảnh **`images`** và chứa file nhãn **`label.txt`** đã qua xử lý của COCO2017 và tập **`dataset2_320`** tương tự cho dữ liệu của Viettel. 2 tập images lưu ảnh đã qua xử lý với size là 320 x 320.

Chạy 1 trong 2 scripts sau để train:

```
python train.py -tm light8n -s 320 -bs 32 -e <epochs> \
                -sp saved_models/<save_weight_name.h5> \
                -pt saved_models/<pretrained_weight_name.h5>
```

Hoặc:

```
python train2.py -tm light8n -s 320 -bs 32 -e <epochs> \
                 -sp saved_models/<save_weight_name.h5>  \
                 -pt saved_models/<pretrained_weight_name.h5>
```

Step 4: Tạo file model từ file weights

```
python w_to_model.py -tm light8n -s 320 -wn <weight_name.h5> \
                     -mn <model_name.h5>
```

Step 5: Inference trên float-point

```
python inference.py     -tm light8n             //mode khi train
                        -s 320                  //size khi train
                        -m <predict-mode>       //image / video
                        -i <image-path>         //path ảnh (for -m image)
                        -v <video-path>         //path vid (for -m video)
                        -w <model-path>         //path của file model
                        -cl config/classes.txt  //path của file classes
```

Ex:

```
python inference.py -tm light8n -s 320 -m image -i test/inp/test1.jpg -w saved_models/light8n_mix_2nd.h5 -cl config/classes.txt
(save in test/inp/out)

python inference.py -tm light8n -s 320 -m video -v test/inp/01b_0_104124.mp4 -w saved_models/light8n_mix_2nd.h5 -cl config/classes.txt
```

Step 6: Thực hiện các task trong phần quantize.

Các task trong phần này được phát triển trong thư mục **quantize_infer_model**, viết trong file **quantize_infer_model/save_quantize_light8n.py**. Để chạy task "ptq" cần chạy "fuse" trước, để chạy "gen_ref" cần chạy "fuse"-"ptq" trước để có file json đầu vào.

```
python quantize_infer_model/save_quantize_light8n.py –m <mode>
```

Trong đó <mode> lần lượt nhận các giá trị:

- "fuse" : fuse model, thực hiện đầu tiên. Path fife weight sửa trong hàm fuse_task() hoặc có thể viết lại thành tham số truyền
- "ptq" : PTQ model, thực hiện thứ 2. Path file json sửa ở dòng json_path, path file weight sửa trong ptq_task() hoặc có thể viết lại thành tham số truyền
- "detect" : detect 1 ảnh ở chế độ integer-only. Path ảnh sửa trong detect_task() hoặc có thể viết lại thành tham số truyền
- "track" : track 1 video ở chế độ integer-only. Path video sửa trong track_task() hoặc có thể viết lại thành tham số truyền
- "gen_ref" : Phục vụ cho việc test code C trong core Tmoby – sinh các file reference output bằng python để đối sánh với output trong code C
- "gen_inp" : Phục vụ cho code C trong core Tmoby – sinh file input dạng số để làm input trong code C.

Note: Các file model tốt nhất đang dùng cho training:

- ligh8n base weights : *saved_models/light8n_mix_2nd_w.h5*
- light8n fuse weights : *saved_models/ light8n_fuse_1st_w.h5*
- ligh8n base model : *saved_models/light8n_mix_2nd.h5*
- light8n fuse model : *saved_models/ light8n_fuse_1st.h5*
- light8n json : *quantize_infer_model/saved/light8n-1st-fix.json*

2.2. Các source khác tham khảo

2.2.1. MOT-human-tf2: Multi-person Tracking based Yolov8-detector

(Source này chạy thẳng file .py chứ chưa viết scripts)

- Bản này performance không tốt bằng *yolov83-human*

- Backbone-neck theo yolov8n light. Phần head thay đổi ở xử lý hậu kì theo cách đơn giản hơn nhiều so với source gốc của Ultralytics, dẫn đến tiền xử lý cũng khác để phù hợp với hậu xử lý. Có thể cải thiện bản này bằng cách viết xử lý hậu kỳ khớp với source gốc Ultralytics.

Path: `/data/minhtq22/content-tracking/mot-human-tf2/`

2.2.2. SiamRPNpp-tf2: Single Object Tracking với kiến trúc 2 input – 2 output

(Source này chạy thẳng file .py chứ chưa viết scripts)

- Phần quantize phức tạp hơn yolov8 do đầu vào 2 input
- Phần postprocess được modify lại đơn giản hơn so với source gốc, model được modify light weights, dữ liệu train nhỏ (30gb) hơn nhiều so với dữ liệu train mà paper cung cấp (4T) nên performance kém model gốc khá nhiều. Tuy nhiên model có performance vẫn tốt trên một số loại đối tượng cụ thể như động vật, phương tiện, ...Có thể cải thiện performance bằng cách cải thiện từ những vấn đề trên.

```
/data/minhtq22/content-tracking/siam-rpnpp-tf2/
/data/minhtq22/content-tracking/siam-rpnpp-mobilev2-tf2/
/data/minhtq22/content-tracking/siam-trans-mobilev2-tf2/
```

2.2.3. Yolov3-ceatec

(Source này đính kèm riêng file .pdf hướng dẫn scripts bên trong)

- Implement Yolov3 với các option tùy chọn mạng yolov3/yolov3-tiny/yolov3-custom cho bài toán detect chi tiết trên vật thể

```
Path     : /data/minhtq22/content-detection/yolov3-ceatec/
Path doc : /data/minhtq22/content-detection/yolov3-ceatec/TRAIN YOLOV3 WITH
           TENSORFLOW.pdf
```