BÁO CÁO ĐỒ ÁN CUỐI KỲ

Lớp: CS2225.CH1501

Môn: NHẬN DẠNG THỊ GIÁC VÀ ỨNG DỤNG

GV: PGS.TS Lê Đình Duy
Trường ĐH Công Nghệ Thông Tin, ĐHQG-HCM

NHẬN DIỆN HÀNH ĐỘNG DẮT XE MÁY VÀ XE ĐẠP

LÊ MINH HOÀNG - CH1702008

Link Github:

https://github.com/lmhoang47/CS2225.CH1501

Tóm tắt

- Tên đề tài: Nhận diện hành động dắt xe máy và xe đạp trên video
- Tóm tắt về đồ án và kết quả đạt được
  - Ứng dụng: phát hiện hành động ăn trộm xe, hành động dắt xe lên vỉa hè
  - Sử dụng TensorFlow - phương pháp RCNN và so sánh với phương pháp SDD
  - Kết quả: độ chính xác trung bình trong khoảng 50%
    - Hành động dắt xe đạp có độ chính xác rất cao, kể cả các trường hợp góc của khung ảnh hướng về đối tượng ở trước mặt, sau lưng và bên hông
    - Hành động dắt xe máy do dữ liệu hạn chế nên độ chính xác thấp. Kết quả thu được chỉ chấp nhận được đối với các loại xe classic và góc khung ảnh ở đằng sau lưng và bên hông

Ảnh của các thành viên của nhóm

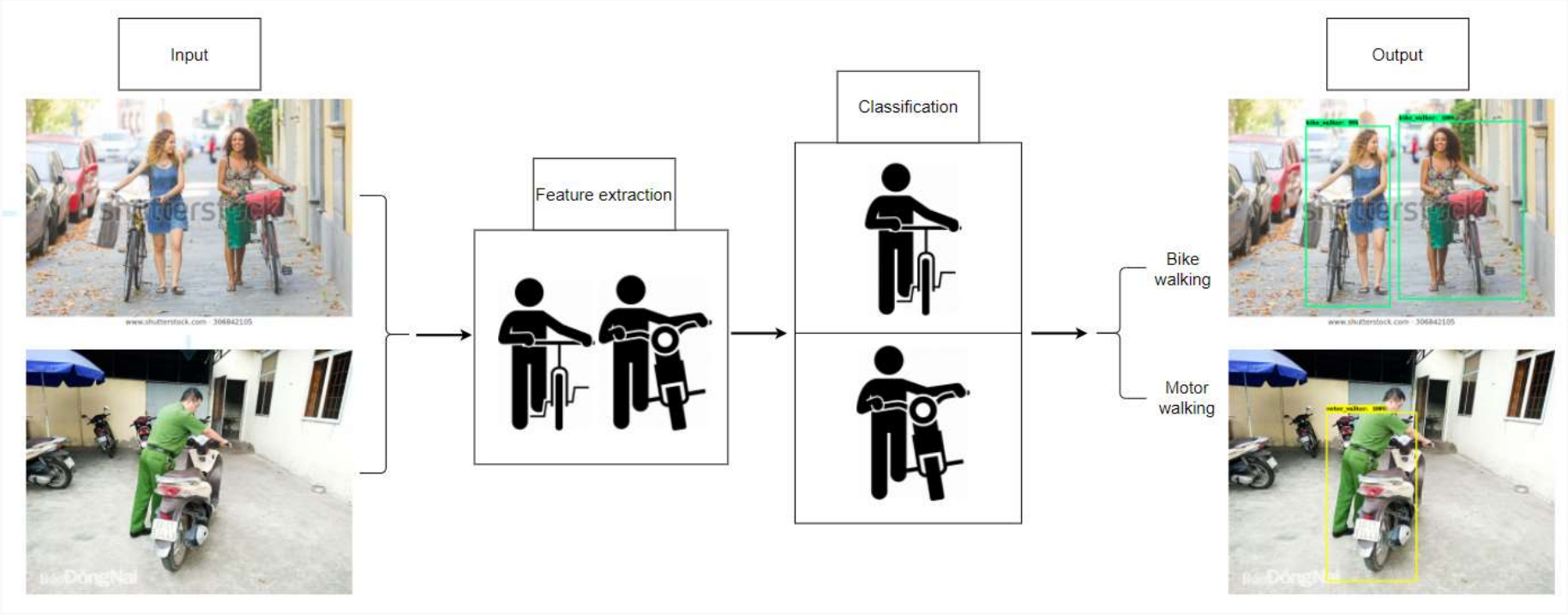
- Họ tên: Lê Minh Hoàng
- MSSV: CH7102008

Mô tả bài toán

- Input: Đoạn video, hoặc ảnh
- Output: Các khung hình có hành động dắt xe máy hoặc xe đạp sẽ có rounding box xung quanh đối tượng, cùng với độ chính xác nhận diện từ model
- Hướng tiếp cận: phương pháp giải bài toán chỉ giới hạn ở việc nhận diện hành động dắt xe, trong đó khi ảnh/video được upload lên thì các khung hình sẽ được trích xuất liên tục. Đồng thời thuật toán trong mô hình huấn luyện sẽ quét qua tất cả các hình này và nhận diện xem có hành động dắt xe hay không
- Giới hạn:
  - Số lượng đối tượng trong 1 khung ảnh < 10, ít chồng lấn
  - Tập dữ liệu huấn luyện của xe đạp phong phú, nhưng cho xe máy rất hạn chế ở góc của đối tượng và loại xe máy

Mô tả bài toán

- Minh hoạ



Loại bài toán ML

- Object Detection
- Sử dụng:
  - API:
    - Protocol bufers để cấu hình tham số của training model
    - Tensor Flow, version 1.15.0 để nhận dạng
    - COCO: load, parse, visualize khung hình nhận dạng và ghi chú
  - Pre-trained model #1: faster_rcnn_inception_v2_coco_2018_01_28
    - Pipeline: faster_rcnn_inception_v2
    - Number of training steps: 200,000
    - Augmentation: random_horizontal_flip
  - Pre-trained model #2: ssd_mobilenet_v2_coco_2018_03_29
    - Pipeline: ssd_mobilenet_v2_coco
    - Number of training steps: 30,000
    - Augmentation: sigmoid

  ➔ So sánh phương pháp RCNN với SSD

Dữ liệu

- Tổng số mẫu:
  - Training: 1,232 images
  - Testing: 310 images
  - Evaluation: 15 videos
- Cách thu thập
  - Nguồn dữ liệu: google image, depositephotos, vnexpress
  - Phương pháp gán nhãn: thủ công, sử dụng Labelimg
  - Số loại nhãn: 2
    - bike_walker
    - motor_walker

Xử lý dữ liệu

- Tạo file pbtx chứa thông tin các lớp cần nhận diện của các bức ảnh
- Chuyển đổi danh sách label dưới dạng xml thành 2 file csv chứa training và testing data
- Loại trừ các ảnh nếu thiếu bounding box
- Chuyển đổi thành TF record dưới dạng binary giúp cho Tensor flow model chạy nhanh hơn
- Tải về 2 base models tương ứng với 2 phương pháp đã chọn

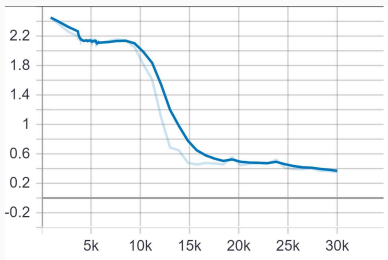
Cấu hình training pipeline

- Số class = 2
- Điều chỉnh kích thước ảnh: 300 x 300 px
- faster_rcnn_inception_v2
  - Batch size = 1
  - Số đối tượng nhận dạng tối đa trong 1 bức ảnh: 100 mỗi lớp
  - Image augmentation: SOFTMAX
  - Lưu lại checkpoint khi training để có thể tiếp tục huấn luyện bất kì thời điểm nào
- ssd_mobilenet_v2_coco
  - Batch size = 16
  - Số đối tượng nhận dạng tối đa trong 1 bức ảnh: 16 mỗi lớp
  - Image augmentation: SIGMOID
  - Lưu lại checkpoint khi training để có thể tiếp tục huấn luyện bất kì thời điểm nào

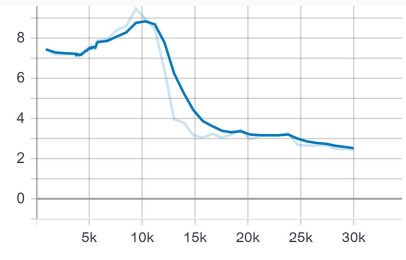
Kết quả huấn luyện - SSD

- Sử dụng phương pháp MAP
- Kết quả tại bước chạy 30,000 - SSD

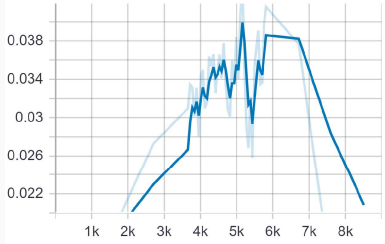
Average Precision (AP) @[IoU=0.50:0.95 | area= all | maxDets=100] = 0.577
Average Precision (AP) @[IoU=0.50 | area= all | maxDets=100] = 0.877
Average Precision (AP) @[IoU=0.75 | area= all | maxDets=100] = 0.689
Average Precision (AP) @[IoU=0.50:0.95 | area= small | maxDets=100] = -1.000
Average Precision (AP) @[IoU=0.50:0.95 | area=medium | maxDets=100] = 0.021
Average Precision (AP) @[IoU=0.50:0.95 | area= large | maxDets=100] = 0.595
Average Recall (AR) @[IoU=0.50:0.95 | area= all | maxDets= 1] = 0.607
Average Recall (AR) @[IoU=0.50:0.95 | area= all | maxDets= 10] = 0.688
Average Recall (AR) @[IoU=0.50:0.95 | area= all | maxDets=100] = 0.690
Average Recall (AR) @[IoU=0.50:0.95 | area= small | maxDets=100] = -1.000
Average Recall (AR) @[IoU=0.50:0.95 | area=medium | maxDets=100] = 0.083
Average Recall (AR) @[IoU=0.50:0.95 | area= large | maxDets=100] = 0.707



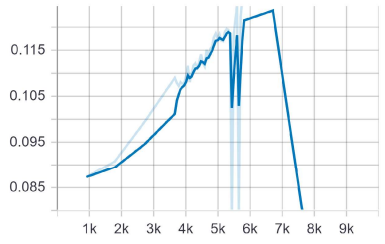
Loss_BoxClassifierLoss_localization_loss



Loss_BoxClassifierLoss_classification_loss



DetectionBoxes_Precision_mAP



DetectionBoxes_Recall_AR

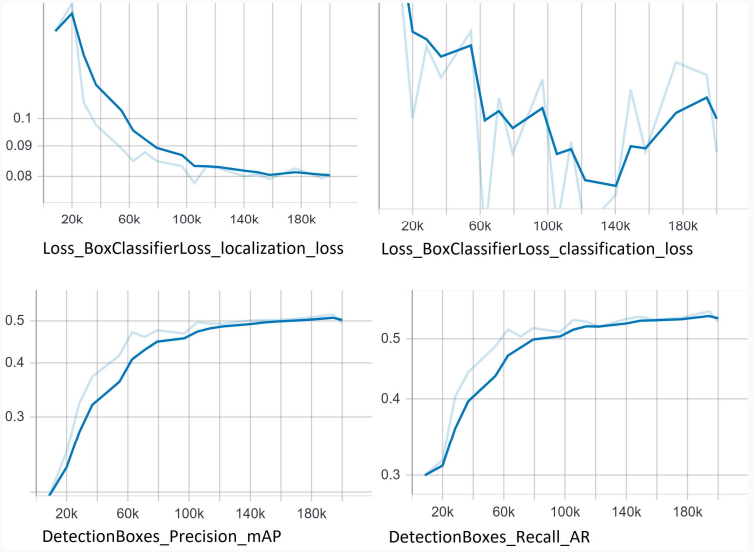
➔Nhận xét

- Độ chính xác khoảng 57%
- Dữ liệu training không có ảnh chụp đối tượng ở khoảng cách xa
- Với các đối tượng chiếm không gian lớn trong bức ảnh, độ chính xác của model lên tới 59.5%

Kết quả huấn luyện - RCNN

- Sử dụng phương pháp MAP
- Kết quả tại bước chạy 200,000 – RCNN

Average Precision (AP) @[IoU=0.50:0.95 | area= all | maxDets=100] = 0.494
Average Precision (AP) @[IoU=0.50 | area= all | maxDets=100] = 0.832
Average Precision (AP) @[IoU=0.75 | area= all | maxDets=100] = 0.516
Average Precision (AP) @[IoU=0.50:0.95 | area= small | maxDets=100] = -1.000
Average Precision (AP) @[IoU=0.50:0.95 | area=medium | maxDets=100] = 0.035
Average Precision (AP) @[IoU=0.50:0.95 | area= large | maxDets=100] = 0.509
Average Recall (AR) @[IoU=0.50:0.95 | area= all | maxDets= 1] = 0.533
Average Recall (AR) @[IoU=0.50:0.95 | area= all | maxDets= 10] = 0.603
Average Recall (AR) @[IoU=0.50:0.95 | area= all | maxDets=100] = 0.621
Average Recall (AR) @[IoU=0.50:0.95 | area= small | maxDets=100] = -1.000
Average Recall (AR) @[IoU=0.50:0.95 | area=medium | maxDets=100] = 0.094
Average Recall (AR) @[IoU=0.50:0.95 | area= large | maxDets=100] = 0.636



➔Nhận xét

- Độ chính xác khoảng 50%
- Dữ liệu training không có ảnh chụp đối tượng ở khoảng cách xa
- Với các đối tượng chiếm không gian lớn trong bức ảnh, độ chính xác của model duy trì ở mức 50%

Đánh giá 2 phương pháp từ thực nghiệm

- ssd_mobilenet_v2:
  - Điểm mạnh: thời gian huấn luyện ngắn
  - Điểm yếu:
    - Độ chính xác thấp
    - Nếu 2 lớp đối tượng có feature tương đối giống nhau, thì lớp nào có độ chính xác cao hơn sẽ chiếm ưu thế khi nội suy ra kết quả cuối cùng. Trong trường hợp này, lớp bike_walker thậm chí được suy đoán trong video có xe máy
- faster_rcnn_inception_v2:
  - Điểm mạnh: độ chính xác cải thiện hơn hẳn SSD trong mô hình mà các lớp đối tượng có feature tương đối giống nhau
  - Điểm yếu: nếu các đối tượng có rounding box chồng lấn, phương pháp RCNN không thể xác định chính xác vị trí của đối tượng mà tạo ra nhiều hơn 1 rounding box cho 1 đối tượng được nhận diện

Đánh giá kết quả của RCNN

- Lớp bike_walker
  - Độ chính xác tổng quan cao
  - Phân biệt khá tốt hành động dắt xe và hành động đạp xe đạp
- Lớp motor_walker
  - Độ chính thấp chủ yếu vì các lý do sau:
    - Các loại xe máy trong dữ liệu huấn luyện không phong phú
    - Trong khung hình chụp trực diện hành động dắt xe, với các loại xe motor classic, kết quả suy đoán dễ bị nhầm lẫn với hành động dắt xe đạp
    - Với các video có độ phức tạp cao như khung ảnh gồm hơn 20 đối tượng dắt xe đi sát nhau, hoặc trong trường hợp video trắng đen thì có nhiều đối tượng bị bỏ qua không nhận diện
  - Các khung hình chụp bên hông và sau lưng cho kết quả có độ chính xác cao

Định hướng phát triển

- Bổ sung vào bộ dữ liệu training:
  - Thêm ảnh ở góc chụp gần và xa
  - Thêm ảnh có label của các đối tượng chồng lấn lên nhau
  - Ảnh chụp gồm nhiều loại xe máy khác nhau
  - Bổ sung thêm ảnh chụp gồm cả 2 lớp đối tượng

  ➔ nhằm tránh overfitting và tăng độ chính xác nhận diện
- Cải tiến thuật toán Augmentation với Keras Preprocessing Layers để tăng độ phong phú cho dữ liệu training