BBM418 Introduction to Computer Vision Lab. # Single Object Tracking with Regression Networks

Meryem Ülkü Kara
21727355
Department of Computer Engineering
Hacettepe University
Ankara, Turkey
b21727355@cs.hacettepe.edu.tr

Overview

In this assignment, it is expected to implement a basic single object viewer by training the network on the given videos. We try to predict the ground truth of moving objects. I handled my homework in two ways.

1 Model1

In the first, I grouped the frames and ground truth tables in pairs, passed the steps of reshaping on them, and gave these four data to the model. In the model, I first cropped for the first frame and extracted the feature, then I cropped and extracted the feature for the second frame. I used the first of my ground truth tables for these crops. Then I combined these two properties and aimed to find a prediction with it, it should be predicting the second one of my prediction ground truth tables I found. But while the shape of my predictionum is a value like [1,2000], my ground truth table has only 4 values. There was a size mismatch here and no matter how hard I tried, I couldn't find any solution to it.

DATASET: dataset = torch.utils.data.TensorDataset

(frame1$_t ensor, frame2_t ensor, bounding_b ox1_t ensor, bounding_b ox2_t ensor$)

MODEL TRAIN:

for frame1, frame2, bounding$_b ox1, bounding_b ox2 inloader : optimizer.zero_g rad()$

resized$_t ransform = transforms.Resize((224, 224))$

CROP FRAME1 crop$_f rame1 = crop_f rame(frame1, bounding_b ox1) FEATUREEXTRACTION features_1 = model(crop_f rame1)$

CROP FRAME2 crop$_f rame2 = crop_f rame(frame2, bounding_b ox1) FEATUREEXTRACTION features_2 = model(crop_f rame2)$

COMBINING FEATURE VECTORS combined$_f eatures = torch.cat((features_1, features_2), dim = 1)$

print(combined$_f eatures.shape)$

Predictions predictions =model.fc(combined$_f eatures)$

lass calculation and back propagation loss = criterion(predictions, bounding$_b ox2)$

CROP METHOD : def crop$_frame(frame, bbox)$:

BOUNDING BOX VALUES x, y, width, height = bbox.squeeze().tolist()

FIND CENTER center$_x = x + (width/2) center_y = y + (height/2)$

ENLARGED BBOX VALUES enlarged$_width = width * 2 enlarged_height = height * 2$

CROPPED FRAME cropped$_frame = frame[:, :, int(center_y - (enlarged_height/2)) : int(center_y + (enlarged_height/2)), int(center_x - (enlarged_width/2)) : int(center_x + (enlarged_width/2))] return cropped_frame$

2 Model2

In the second time, I read the frames and ground truths and grouped them in pairs. Since there was a size mismatch in the model and I couldn't solve it in my first handling, I cropped my frames as desired before entering the model in this model and found the expected cropped frames. Then I got my cropped frame1, cropped frame2 , expected bounding box , expected cropped frame2. In this part, I tried to train the model in two ways, the first was to train the cropped frame1, cropped frame2 , and the expected cropped frame2 data. I aimed to extract and combine features for Frame1 and Frame2, then make predictions with this feature extraction and find the expected frame2, but here again I encountered a size mismatch. I tried many things like removing fc-layer but I couldn't get past this issue. On the other hand, I tried to find the expected bounding box by taking the cropped frame1, cropped frame2 data as input value, but I couldn't get over the size mismatch errors.

CROP METHOD :

def crop$_frame(frame, bbox)$:

GET BBOX VALUES x, y, width, height = bbox

frame$_height, frame_width, = frame.shape$

CONTROL BOUNDING BOX HEIGHT LARGER OR NOT TO FRAME HEIGHT if height > frame$_height$: $height = frame_height CONTROL FOR WEIGHT if width > frame_width$: $width = frame_width$

CALCULATE CENTER center$_x = x + (width/2) center_y = y + (height/2)$

CALCULATE ENLARGED BBOX enlarged$_width = width * 2 enlarged_height = height * 2$

FIND CROPPED FRAME cropped$_frame = frame[int(center_y - (enlarged_height/2)) : int(center_y + (enlarged_height/2)), int(center_x - (enlarged_width/2)) : int(center_x + (enlarged_width/2))]$

return cropped$_frame$

2.1 Model2-1

DATASET: dataset = CustomDataset(cropped$_images1, cropped_images2, cropped_images3)$

MODEL TRAIN

for frame1, frame2, frame3 in data$_loader$: $optimizer.zero_grad()$

EXTRACT FEATURES WITH CROPPED$_FRAME1 features1 = model(frame1) Featureextraction$

EXTRACT FEATURES WITH CROPPED$_FRAME2 features2 = model(frame2) Featureextraction$

COMBINING FEATURES VECTORS combined$_features = torch.cat((features1, features2), dim = 1)$

print(combined$_features.shape)print(frame3.shape)$

Predictions predictions = combined$_featuresprint(predictions)$

lass calculation and back propagation loss = criterion(combined$_features, frame3)loss.backward()$

2.2 Model2-2

DATASET: dataset = CustomDataset(cropped$_images1, cropped_images2, bbox)$

MODEL TRAIN feature$_extractor = torch.nn.Sequential(*list(model.children())[: -1])$

for frames, targets in data$_loader$:

frame1 = frames[0].to(device) frame2 = frames[1].to(device) targets = targets.to(device)

optimizer.zero$_grad()$

FEATURE EXTRACT features1 = feature$_extractor(frame1)features2 = feature_extractor(frame2)$

COMBINE FEATURE VECTORS combined$_features = torch.cat((features1.flatten(start_dim = 1), features2.flatten(start_dim = 1)), dim = 1)$

hidden = torch.relu(fc1(combined$_features))predictions = fc2(hidden)$

PREDICTED GROUND TRUTH BOXES ground$_truth_box = frame3$

loss = criterion(predictions, ground$_truth_box)loss.backward()optimizer.step()$

I tried many methods, but in all of them I got errors during the training phase and could not solve my mistakes.