Multimedia Project SoSe 23

Assignment 4

Multimedia Computing Lab
Prof. Dr. Rainer Lienhart
Katja Ludwig
Julian Lorenz

Submission: May 17, 2023

Please take a look at assignment 0 on how to submit your work.

Exercise 4.1 Anchor Grid 10 Points

In this exercise you will write a function that generates a grid of anchors for a given shape of an output feature map. Write a function with the following prototype:

```
def get_anchor_grid(
        num_rows: int,
        num_cols: int,
        scale_factor: float,
        anchor_widths: Sequence[float],
        aspect_ratios: Sequence[float],
)
```

`anchor_widths` should contain all possible box widths in pixels and `aspect_ratios` should contain all possible box aspect ratios. The aspect ratios are given as height per width, so a box that is 3 units high and 4 units wide has an aspect ratio of 0.75.

The `scale_factor` represents the ratio between image size and feature map size and is equal to image width / `num_cols`. For example, if the image has a width of 128px and the feature map has 16 columns, you will have a `scale_factor = 8.0`.

Order the coordinates in the last dimension such that `AnnotationRect.fromarray()` works with the anchor grid.

Exercise 4.2 Label Grid 10 Points

In this exercise you will implement the label grid. Reuse your AnnotationRect code from assignment 3.

(a) Write a function

```
def iou(rect1, rect2)
```

that computes the Intersection over Union of two AnnotationRect objects.

(b) Write a function

```
def get_label_grid(anchor_grid, gts, min_iou)
```

that takes your anchor grid from excercise 4.1 (the 5D array) and a list of all groundtruth boxes of an image. min_iou is the minimal IoU that is required to classify an overlap as relevant. Return the corresponding label grid.

(c) Calculate the label grid for image #00542033 from the dataset. Use an anchor grid with a scale factor of 8.0 and derive num_rows and num_cols from the image size. Find suitable values for scales and aspect_ratios.

Draw all boxes from the anchor grid that have an overlap greater than 0.7 and make sure that all humans in the image have one or multiple boxes around them. If not, tweak your parameters for the anchor grid a bit more. Save the resulting image with the boxes to your hard drive and include it in your submission.

Exercise 4.3 Finalizing the Data Pipeline 8 Points

To finish your PyTorch data pipeline, your MMP_Dataset will need to return the label grid.

(a) Somewhere outside of the dataset, instantiate an anchor grid with suitable settings. Use this anchor grid and an image size of 224 × 224 pixels for the rest of this exercise.

(b) Now it's time to modify def __getitem__(self, idx). It must now return a 3-tuple of image tensor, label grid, and image id. The image id is the filename of the selected image (without the file extension) converted to an integer. We need it for the next assignment.

(c) The MMP_Dataset requires an anchor grid. Therefore, modify get_dataloader to accept an anchor grid: def get_dataloader(path_to_data, image_size, batch_size, num_workers, anchor_grid) and make sure that the function works as expected.

(d) Create a data loader for the training set with the def get_dataloader function with batch size 8. Use the data loader to generate at least 12 batches and draw all positive boxes for the 6th image of each batch together with the image. It is common and intended to get multiple boxes around a single object. Submit all resulting images.

(e) Modify your data loader such that it also works with test data. __getitem__ must still return a 3-tuple but the returned value for the label does not matter.