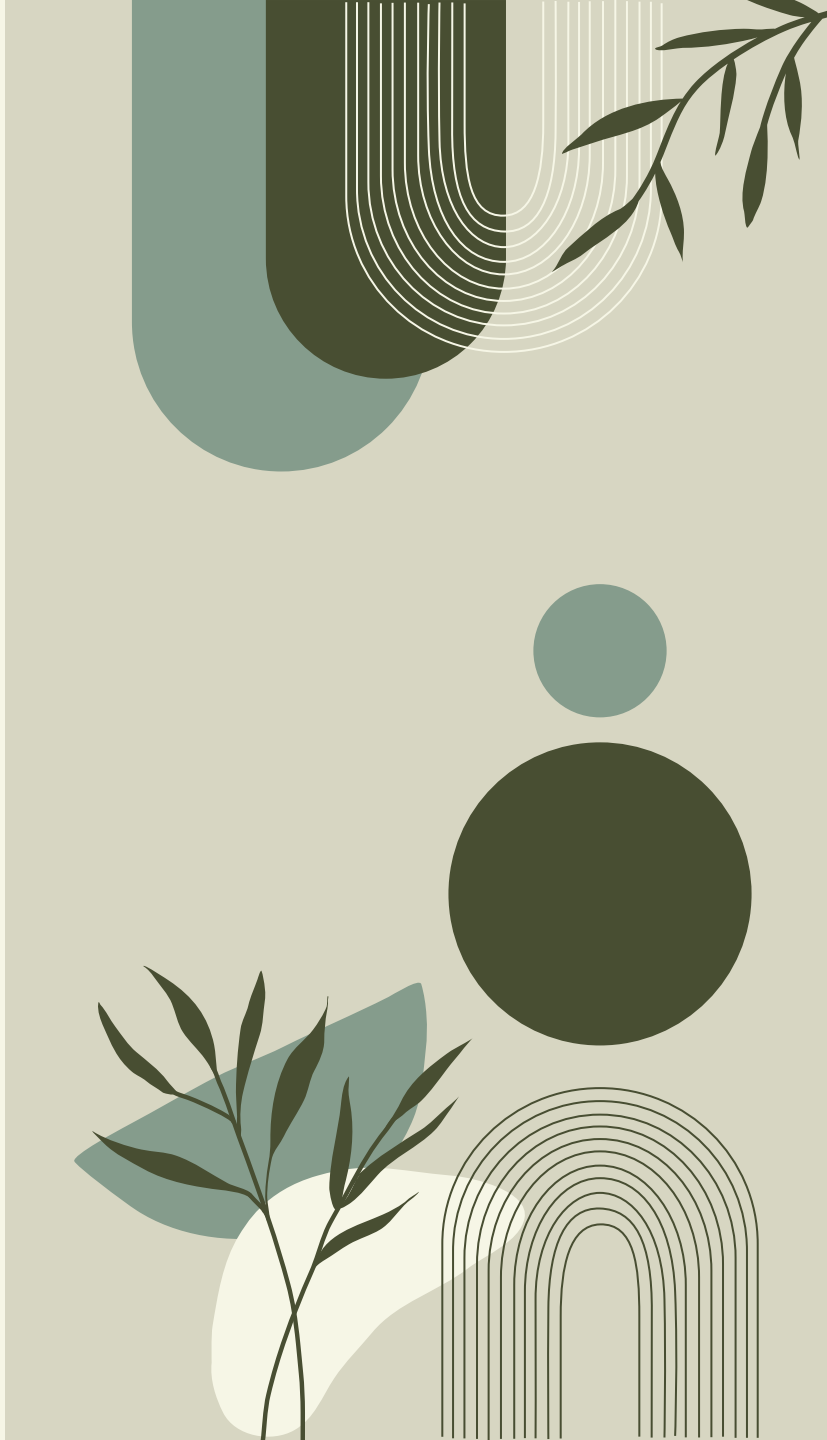
Kalibrácia dopravnej kamery pomocou detekcie úbežníkov vozidiel

Lea Michalíková



01

Porovnanie modelov

Pôvodný objektový detektor vs mnou použitý detektor

centernet/resnet

tensorflow/centernet-resnet

CenterNet Object detection model with the ResNet-v2-50 backbone, trained on COCO 2017 dataset with trainning images scaled to 512×512.

Inputs

A three-channel image of variable size - the model does **NOT** support batching. The input tensor is a `tf.uint8` tensor with shape `[1, height, width, 3]` with values in `[0, 255]`.

Outputs

The output dictionary contains:

- `num_detections`: a `tf.int` tensor with only one value, the number of detections `[N]`.
- `detection_boxes`: a `tf.float32` tensor of shape `[N, 4]` containing bounding box coordinates in the following order: `[ymin, xmin, ymax, xmax]`.
- `detection_classes`: a `tf.int` tensor of shape `[N]` containing detection class index from the label file.
- `detection_scores`: a `tf.float32` tensor of shape `[N]` containing detection scores.

faster_rcnn/inception_resnet_v2

google/faster-rcnn-inception-resnet-v2

Object detection model trained on Open Images V4 with ImageNet pre-trained Inception Resnet V2 as image feature extractor.

Inputs

A three-channel image of variable size - the model does **NOT** support batching. The input tensor is a `tf.float32` tensor with shape `[1, height, width, 3]` with values in `[0.0, 1.0]`.

Outputs

The output dictionary contains:

- `detection_boxes`: a `tf.float32` tensor of shape `[N, 4]` containing bounding box coordinates in the following order: `[ymin, xmin, ymax, xmax]`.
- `detection_class_entities`: a `tf.string` tensor of shape `[N]` containing detection class names as Freebase MIDs.
- `detection_class_names`: a `tf.string` tensor of shape `[N]` containing human-readable detection class names.
- `detection_class_labels`: a `tf.int64` tensor of shape `[N]` with class indices.
- `detection_scores`: a `tf.float32` tensor of shape `[N]` containing detection scores.

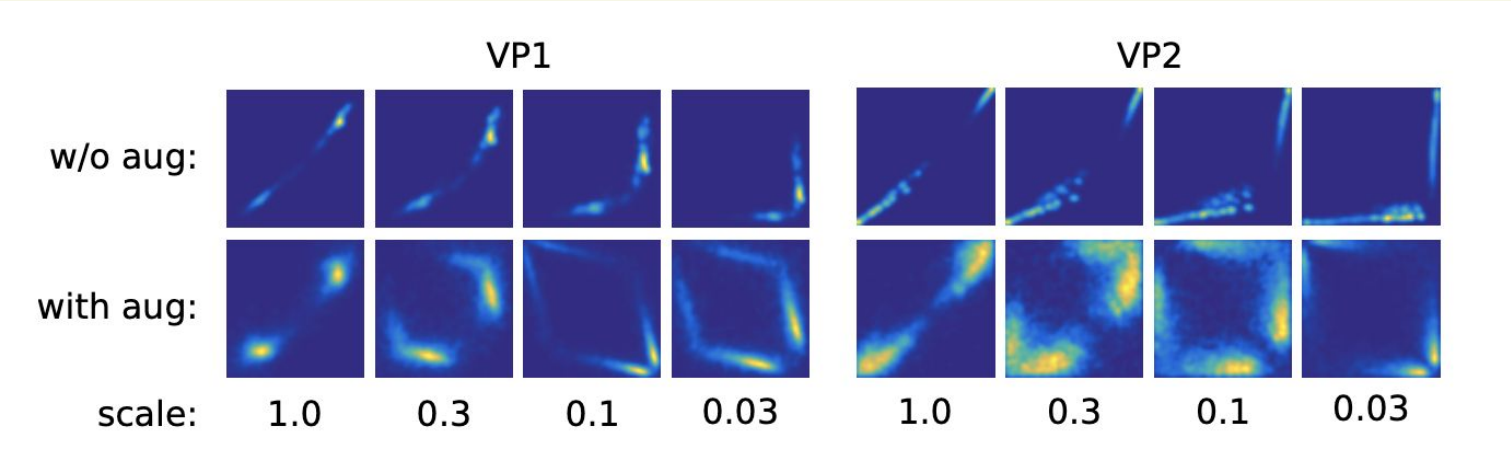
pôvodný model vs nový model

02

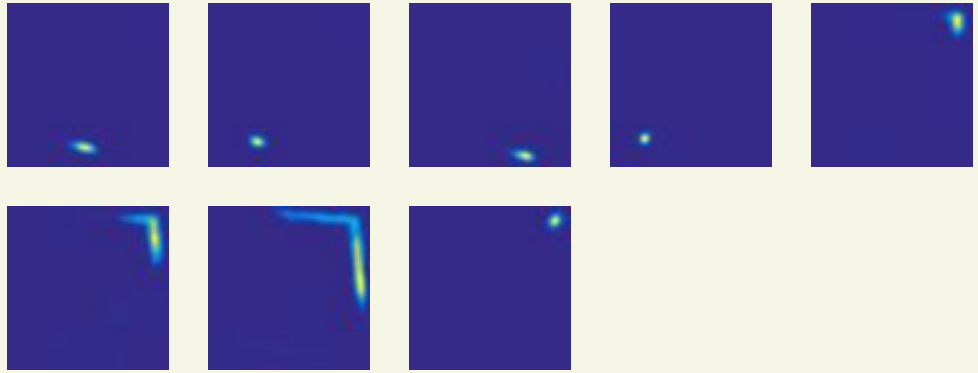
Ukážka výsledkov

Ako sa zdetekovali autá pomocou nového modelu

Existujúce výsledky



Moje (čiastočné) výsledky



Detekcie v súbore

[{"filename": "000/000000.jpg", "frame_cnt": 1, "boxes": [[0.3864296376705169 7, 0.3933395743370056, 0.515129029750824, 0.528646051883697 5], [0.453079491853714, 0.911588907241821 3, 0.592233061790466 3, 1.0], [0.126772150397300 72, 0.650961279869079 6, 0.188103079795837 4, 0.722844481468200 7], [0.353865683078765 87, 0.35624319314956665, 0.459930002689361 6, 0.488844990730285 64], [0.27283298969268 8, 0.8150820136070251, 0.341865062713623 05, 0.89779096841812 13], [0.233823537826538 09, 0.95105856657028 2, 0.31971463561058044, 1.0], [0.24307543039321 9, 0.678412079811096 2, 0.31923416256904 6, 0.771264195442199 7], [0.12283276021480 56, ...

Kód

Screenshoty najpodstatnejších častí kódu

```
with tf.Session() as sess:
    sess.run(tf.global_variables_initializer())
    sess.run(tf.tables_initializer())

    for filename in filenames:
        frame = cv2.imread(os.path.join(path, 'frames', session, filename))
        frame_cnt += 1
        if frame is None:
            continue

        input_tensor = frame[np.newaxis, :, :, ::-1]
        result = sess.run(detector_output, feed_dict={image_placeholder: input_tensor})

        frame = frame / 255.0

        boxes = result["detection_boxes"]
        labels = result["detection_class_labels"]
        scores = result["detection_scores"]

        # car_label = 3 #old
        car_label = 571
        l = np.logical_and(scores > conf, labels == car_label)
        boxes = boxes[l]
        scores = scores[l]
```

```python
for box in boxes:
    x_min = int(1920 * box[1]) # todo: remove magic numbers
    y_min = int(1080 * box[0])
    x_max = int(1920 * box[3] + 1)
    y_max = int(1080 * box[2] + 1)

    box_center = np.array([x_min + x_max, y_min + y_max]) / 2
    box_scale = np.array([x_max - x_min, y_max - y_min]) / 2

    car = frame[y_min:y_max, x_min:x_max, :]
    car = cv2.resize(car, (args.input_size, args.input_size), cv2.INTER_CUBIC)

    heatmap_pred = heatmap_model.predict(car[np.newaxis, ...] / 255.0)
    pred_vps, pred_vars = process_heatmaps(heatmap_pred[-1], scales)

    vp1_var = pred_vars[0, :, 0]
    vp2_var = pred_vars[0, :, 1]
    vp1_var_idx = np.argmin(vp1_var, axis=-1)
    vp2_var_idx = np.argmin(vp2_var, axis=-1)

    vp1_box = pred_vps[0, vp1_var_idx, :2]
    vp2_box = pred_vps[0, vp2_var_idx, 2:]

    vp1 = box_scale * vp1_box + box_center
    vp2 = box_scale * vp2_box + box_center

    principal_point = np.array([frame.shape[1] / 2 + 0.5, frame.shape[0] / 2 + 0.5])
    focal = get_focal(vp1, vp2, principal_point)
    m = (vp1[1] - vp2[1]) / (vp1[0] - vp2[0])
    b1 = vp1[1] - m * vp1[0]
    b2 = vp2[1] - m * vp2[0]
```

Ďakujeme

Priestor na otázky

slidesMania

Free themes and templates for
Google Slides or **PowerPoint**

NOT to be sold as is or modified!

Sharing is caring!

Read FAQ on slidesmania.com

Do not remove the slidesmania.com text on the sides.