Austin Donovan

To determine which template performed better, I created the matlab function on the following page which is supplied with an `iou_thr` parameter which it uses to run the `detection_accuracy` function against all faces with detection thresholds ranging from 0 to 1, by steps of 0.1. Running this with the parameter 0.5, yielded the output on the page 3. Based on this output, I would choose to ship the package using the cropped template, as it performs better on average.

Across all face detection thresholds, the templates have similar amounts of false negatives, thus the templates are about equally good at actually finding faces. Because they both have similar levels of "correctness" the tie-breaker is the speed in "normal" cases. I believe normal situations for face detection are when the system is reasonable sure, or at most slightly unsure, about if a face is present or not. The face detection thresholds representing these cases are those between 0.40 and 0.60.

The cropped template has significantly fewer false positives for normal detection thresholds, which means that a user would be processing a lot less incorrect data under the assumption that it is correct. Within this range the uncropped template generates noticeably more true positives than the cropped template (without a noticeable difference in false negatives), indicating that the system finds several faces around an actual face. This would also lead to wasting user's processing power on potentially processing faces that are really the same face.

However, the cropped template does yield far more false positives when the face detection threshold is low (<= 0.30), which is troublesome. In my opinion, however, the situations where a customer would be wanting to find things with such a low probability of actually being a face are special cases and not the norm. Thus, the library should simply instead stress that it is a general-purpose library and should not be expected to perform well under such circumstances.

```
function [] = run_all_tests(iou_thresh)
      template_normal = imread('average_face.png');
      template_cropped = imread('average_face_cropped.png');
      scales = make_scales_array(1, 5, 1.1);

      fprintf('iou_thresh = %4.2f\n', iou_thresh);
      fprintf('Thresh Template Type        tp    fp    fn\n');
      for detection_thr = 0 : 0.1 : 1
      [norm_tp, norm_fp, norm_fn] = detection_accuracy(...
            'ground_truth.txt', template_normal, scales, …
            detection_thr, iou_thresh);

      [crop_tp, crop_fp, crop_fn] = detection_accuracy(...
            'ground_truth.txt', template_cropped, scales, …
            detection_thr, iou_thresh);

      fprintf('%4.2f    Normal      %5d %5d %5d\n', ...
            detection_thr, norm_tp, norm_fp, norm_fn);
      fprintf('%4.2f    Cropped     %5d %5d %5d\n', ...
            detection_thr, crop_tp, crop_fp, crop_fn);
      fprintf('\n');
      end
end
```

iou_thresh = 0.50

Thresh	Template Type	tp	fp	fn
0.00	Normal	351	6474	5
0.00	Cropped	569	11752	4
0.10	Normal	264	3063	7
0.10	Cropped	437	6317	4
0.20	Normal	228	2513	7
0.20	Cropped	358	4657	5
0.30	Normal	170	1839	9
0.30	Cropped	202	2489	7
0.40	Normal	103	928	11
0.40	Cropped	89	630	12
0.50	Normal	55	315	23
0.50	Cropped	33	51	33
0.60	Normal	24	54	42
0.60	Cropped	15	4	50
0.70	Normal	5	2	60
0.70	Cropped	4	0	61
0.80	Normal	0	1	65
0.80	Cropped	0	0	65
0.90	Normal	0	0	65
0.90	Cropped	0	0	65
1.00	Normal	0	0	65
1.00	Cropped	0	0	65