Face Detection

BlazeFace

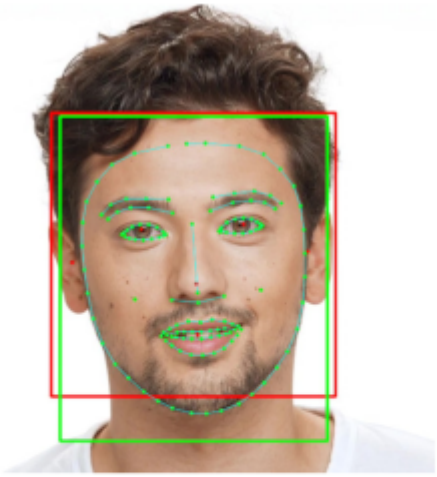

Figure 3. Pipeline example (best viewed in color). Red: BlazeFace output. Green: Task-specific model output.

- Google uses it as a face detector in MediaPipe Studio
- Paper
- Characteristics:
  - input image size: 128x128
  - model size: 224 KB
  - outputs (17 values):
    - bounding box: `ymin`, `xmin`, `ymax`, `xmax` (all normalized to [0, 1])
    - facial landmarks: `right_eye_x`, `right_eye_y`, `left_eye_x`, `left_eye_y`, `nose_x`, `nose_y`, `mouth_x`, `mouth_y`, `right_ear_x`, `right_ear_y`, `left_ear_x`, `left_ear_y`
  - number of anchor boxes: 896
  - Based on **SSD architecture** - predefined anchor boxes, but less than in SSD (Due to limited variance in human computing smaller feature maps is redundant)

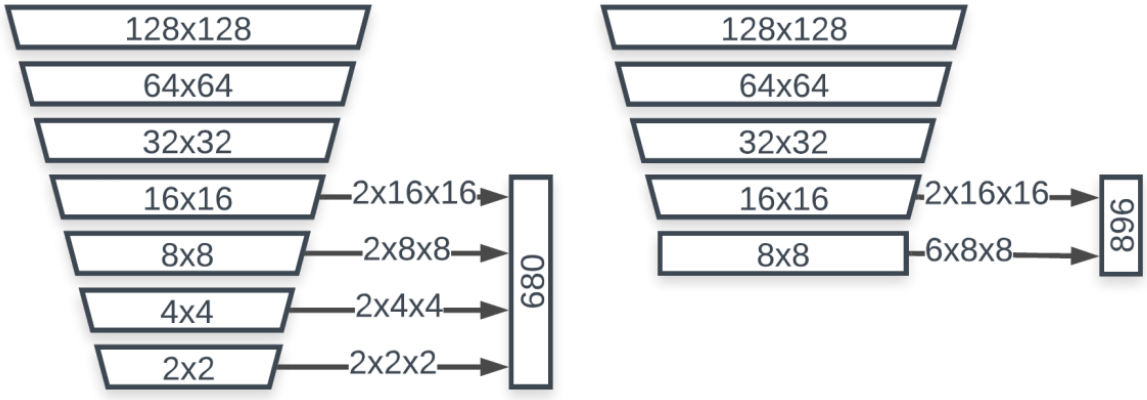

Figure 2. Anchor computation: SSD (left) vs. BlazeFace

- **depthwise convolutions** with kernels 5x5 - decreasing the total amount of bottlenecks required to reach a particular receptive field size, thus reducing the number of parameters and computations

- Inference details:

  1. Image size should be 128x128 with values ranging from -1 to 1
  2. Passing X through the network will result in confidence scores for each class and offsets for the anchor box. All that for each anchor box.
  3. Finding boxes for predefined anchor boxes
  4. Weighted Non-maximum suppression - "*achieves stabler, smoother tie resolution between overlapping predictions. (...) It incurs virtually no additional cost to the original NMS algorithm.* "

YOLO

Additional Resources

- R CNNs, SSDs, and YOLO
- SSD
- BlazeFace pytorch implementation
- SSD Anchor calculator
- YOLOv5-face-landmarks-cv2 implementation
- YOLOv5 grid and anchors