Algorithm 1 haar_featlist($window_y = 24$, $window_x = 24$, double *rectangle_patterns [$10 \times no_{rectangles}$], $no_{rectangles}$)

```
index_features = 0
index_rectangle = 0
{no_rectangles = the TOTAL number of rectangles regardless of the pattern}
for r = 0, r< no_rectangles do
  temp ← (id of current pattern) {as they wrote it: rect_param[0 + index_rectangle] and is
  initially 0}
  if id_current_feature != temp then
    id_current_feature ← temp {id_current_feature is initially 0}
    W ← (width of current pattern) {as they wrote it: rect_param[1 + index_rectangle]}
    H ← (height of current pattern) {as they wrote it: rect_param[2 + index_rectangle]}
    {24×24 is the size of the sub-window – so I guess we don't have to slice anything}
    {loop over the image trying to fit the current pattern}
    for w = W, w< 24+1, w = w+W do
      for h = H, h< 24+1, h = h+H do
        for y = 0, y+h < 24+1, y ++ do
          for x = 0, x+w < 24+1, x = ++ do
            Features[0 + index_features] ← id_current_feature
            {store the top-left coordinates of the pattern in the sub-window}
            Features[1 + index_features] ← x
            Features[2 + index_features] ← y
            {store the width and height of the pattern in the sub-window}
            Features[3 + index_features] ← w
            Features[4 + index_features] ← h
            Features[5 + index_features] ← index_rectangle
            {it is stored as an array instead of matrix – because they don't know the future
            size of it, maybe? – and so the size of one feature is 6}
            index_features ← index_features + 6
          end for
        end for
      end for
    end for
  end if
  {the rectangles are stored as an array instead of matrix – I can't see the reason here – and
  so the size of one rectangle is 10 ..look at the top of haar.c for more info}

  index_rectangle ← index_rectangle + 10
end for
```