```
class Solution {
    public int maximalRectangle(char[][] matrix) {
        if (matrix.length == 0 || matrix[0].length == 0) {
            return 0;
        }
        int col = matrix[0].length, max = 0;
        int[] h = new int[col];
        for (int i = matrix.length-1; i >= 0; i--) {
            for (int j = 0; j < col; j++) {
                if (matrix[i][j] == '0') {
                    h[j] = 0;
                } else {
                    h[j]++;
                }
            }
            max = Math.max(max, largestRectangleArea(h));
        }
        return max;
    }

    public int largestRectangleArea(int[] heights) {
        Stack<Integer> stack = new Stack<>();
        int max = 0;
        for (int i = 0; i <= heights.length; i++){
            if (stack.isEmpty() ||
                (i != heights.length && heights[i] >= heights[stack.peek()])) {
                stack.push(i);
            } else {
                int index = stack.pop();
                max = Math.max(max, heights[index] * (stack.isEmpty() ?
                                            i : i - stack.peek() - 1));
                i--;
            }
        }
        return max;
    }
}
```

Given a 2D binary matrix filled with 0's and 1's, find the largest rectangle containing only 1's and return its area.

```
Input:
[
  ["1","0","1","0","0"],
  ["1","0","1","1","1"],
  ["1","1","1","1","1"],
  ["1","0","0","1","0"]
]
Output: 6
```