Homework 0

In this homework we generated a pipeline that changes images through a variety of transformations in order to inflate the size of the dataset. These transformations included resizing, cropping, jittering colors, and rotations. Examples of the full pipeline are shown in **figure 1.**

Figure 1: The top panels shows the results of a set of randomized transforms on a single image. The bottom panel shows different images with larger changes.

In general, the image processing algorithm operates by using a set of functions which take and return the same kind of output, so that they can be composed with one another. This allows the amount of change to be tuned and varied among batches, and greatly increases the size of available data. An interesting experiment would be to see how small the changes could be made while still helping the learning algorithm.

Unfortunately, I wasn't able to get the square-crop working. The unfinished code was left as a commented section of student_code.py. My approach was as follows:

1) Find the slope of the bounding lines of the rotated rectangle. This is given by, for the example of an rectangular image with w>h, sin(theta)/cos(theta) for the bottom side, since the vertical component of the bottom side is l * sin(theta), horizontal component l * cos(theta).
2) Knowing the slope of one line gives you the slope of all lines: if the bottom side has slope m, the left side as slope as slope -1/m
3) To make the problem easier, we can set the y-intercept of the bottom line to be 0. The y intercepts of the top and left lines are the same. The actual shape of the rectangle has rotational symmetry, so these arguments hold for l>w etc.
4) The length of the vertical side of the triangle is the difference between the top and left side, since any rectangle not touching the left side could be moved along the rotated rectangle's diagonal until it touched. This length is H = (y=mx+b) - (y=-1/x+b), since both x's are the same.
5) The length of the horizontal side is L = (x = y/m) - (x = m(y-c)). Substituting -x/m + c for y gives this length in terms of the x-coordinate of where the vertical side touches the left side of the rotated rectangle.
6) The area of the maximum rectangle is H*L. Taking the derivative w.r.t x and setting to 0 should give the x-coordinate that gives the maximum rectangle.
7) Given the length and height of the maximum rectangle, it can just be centered to give the desired output.

However, this algorithm always returned an x-coordinate on the midpoint of the left side, and for lower angles, generally < 20 degrees, the output width would start to explode.