Machine Learning for Computer Vision

Exercise 2

Kodai Matsuoka, Yuyan Li

May 5, 2017

1 Iterated Conditional Models

The missing code is:

```
# unary terms
energy += - beta * math.log(unaries[x0,x1,l])

# pairwise terms
energy += 4 - [labels[x0-1,x1], labels[x0+1,x1],
            labels[x0,x1-1], labels[x0,x1+1]].count(l)
```

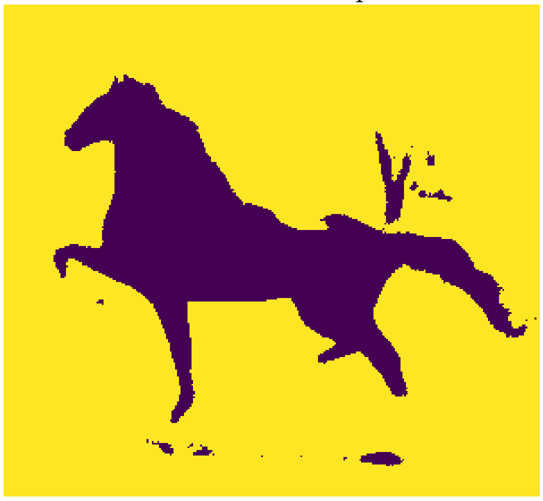
The regularizer *beta* changes the coarseness of the labeling.
The code to use probability pictures as unaries is:

```
# import predictions from exercise1
# prediction images are in folder predictions/
pred_paths = glob.glob("predictions/*")
pred = [skimage.img_as_float(skimage.io.imread(f)) for f in pred_paths]

# Getting rid of the zeros
for x in numpy.nditer(pred[0], op_flags=['readwrite']):
    if x == 0:
        x[...] = 1e-100
    if x == 1:
        x[...] = 1. - 1e-16

fg = p
bg = 1.-p
unaries = numpy.dstack((fg, bg))
```

In the whole program (*icm.py*) there is also an addition at the end to produce pictures of the labels. A few examples are shown here:



2 Higher order factors

The domain of x_z is $\{0, 1, 2, 3, 4, 5, 6, 7\}$. Each variable value represents one energy state. The pairwise factors are given in the following table:

x_z	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
x_0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
ϕ_{0z}	a	b	c	d	∞	∞	∞	∞	∞	∞	∞	∞	e	f	g	h

x_z	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
x_1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
ϕ_{1z}	0	0	∞	∞	0	0	∞	∞	∞	∞	0	0	∞	∞	0	0

x_z	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
x_1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
ϕ_{1z}	0	∞	0	∞	0	∞	0	∞	∞	0	∞	0	∞	0	∞	0

By using infinity in the pairwise factors, for any value for x_z there is only one value that each x_i can have which correspond with the energy given by ϕ_{012}.