Relative-Rotations

Unknown Author

April 12, 2015

Things I will try the PASCAL/Imagenet Setting: 1. Take bounding boxes that are just rescaled to constant 256 x 256 2. Try cropping the boxes carefully. 3. On Picking run the experiment of trying with different layers - how well the network generalizes. Create a pose embedding. Do visual reasoning in this domain. ie. two buses subtracted from each other + car shifts the car in the right way.Initially I was making an error that mirror flipping was on while predicting the relative rotations. Even with this I got some reasonable weights while training from scratch. They are plotted below.

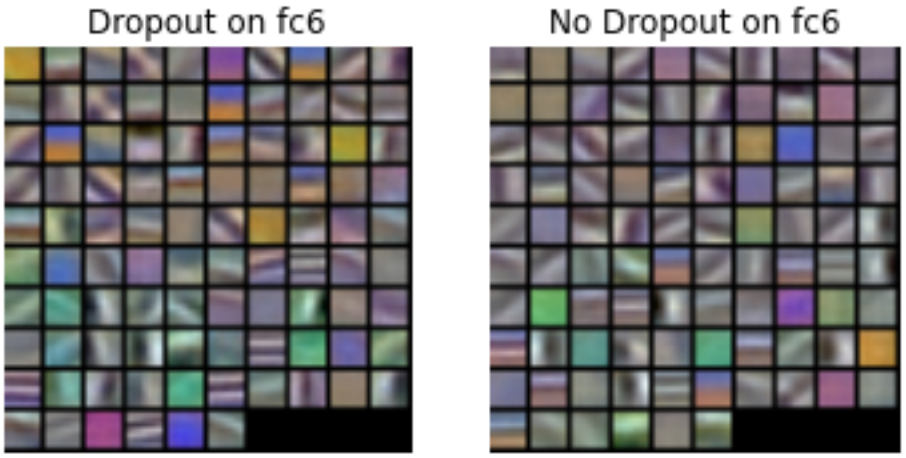
In [2]:
```
%matplotlib inline
%load_ext autoreload
%autoreload 2
codeDir     = '/work4/pulkitag-code/pkgs/caffe-v2-2/modelFiles/keypoints/code/'
expDir      = '/work4/pulkitag-code/pkgs/caffe-v2-2/modelFiles/keypoints/exp/'
snapshotDir = '/data1/pulkitag/snapshots/keypoints/'
import caffe
import os
import my_pycaffe as mp
import my_pycaffe_io as mpio
import matplotlib.pyplot as plt
import h5py as h5
import other_utils as ou
import collections as co
#import plotly.plotly as ply
#plotly.tools.set_credentials_file(username='pulkit.audacious', api_key='9cqniif4ai',
#Experiment and snapshot paths.
#Load the module
currDir = os.getcwd()
os.chdir(codeDir)
import process3d as p3d
import process_caltech as pc
os.chdir(currDir)
expStr = 'rotObjs_128_kmedoids30_20'
defFile = os.path.join(expDir, expStr, 'caffenet_siamese.prototxt')
```

In [13]:
```
    #This network was trained by concatenating fc-6 features. Prior to concatenation t
    numIterations  = 60000
    numIterations2 = 90000
    imSz           = 128
    expStr     = 'rotObjs_128_kmedoids30_20'
    modelName = 'keypoints_siamese_scratch_iter_%d.caffemodel' % numIterations
    defFile    = os.path.join(expDir, expStr, 'keynet_siamese_deploy.prototxt')
    modelFile = os.path.join(snapshotDir, 'exprotObjs_lblkmedoids30_20_imSz%d'% imSz,
    net        = mp.MyNet(defFile, modelFile)
    #Network without dropouts on fc-6 features.
    modelName2 = 'keypoints_siamese_scratch_nodrop_fc6_iter_%d.caffemodel' % numIterat
    modelFile2 =  os.path.join(snapshotDir,"exprotObjs_lblkmedoids30_20_imSz%d" % imSz
    net2       = mp.MyNet(defFile, modelFile2)
    fig = plt.figure()
    ax1 = plt.subplot(1,2,1)
    ax2 = plt.subplot(1,2,2)
    net.vis_weights('conv1', ax=ax1, titleName='Dropout on fc6')
```

```
    net2.vis_weights('conv1', ax=ax2, titleName='No Dropout on fc6')
    ax1.axis('off')
    ax2.axis('off')
```

```
(-0.5, 79.5, 79.5, -0.5)
```

Out [13]:



In []:

In [17]:
```
#Define the experiment
prms = p3d.get_exp_prms(imSz=128, lblType='uniform', numSamplesTrain=40000, numSamples
                        azBin=30, elBin=10)

#Get the statistics of the training data.
#Training data is PASCAL 2012 + Entire Imagenet.
annCountTrain = p3d.get_class_statistics_exp(prms, 'train')
#Get the statistics of the val data
annCountVal   = p3d.get_class_statistics_exp(prms, 'val')
```

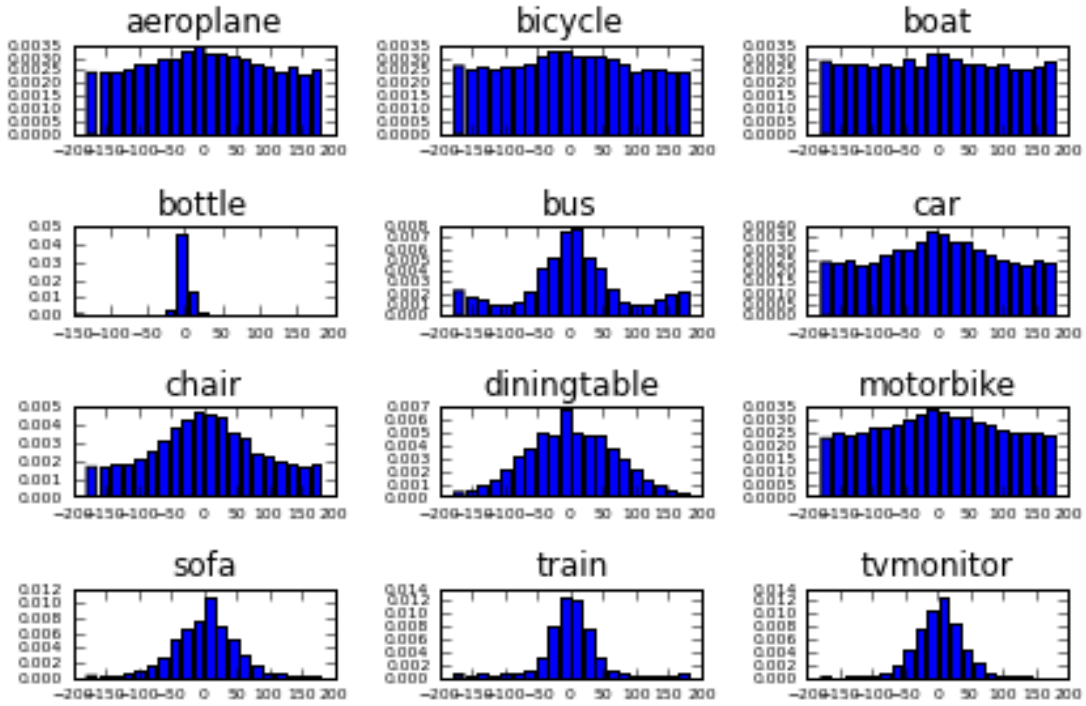
In [18]:
```
annCount = {'%s' % key: [val] for key,val in annCountTrain.iteritems()}
annCount = {'%s' % key: annCount[key] + [val] for key,val in annCountVal.iteritems()}
formatStr = "{:<12} {:<20} {:<20}"
print formatStr.format('Class','Train-BBox', 'Val-BBox')
print '-' * 100
for key, val in sorted(annCount.items()):
    bbTr, bbVl = val
    print ("%s" % formatStr).format(key, bbTr, bbVl)
```

```
Class        Train-BBox           Val-BBox
----------------------------------------------------------------------------
------------------------------
aeroplane    2314                 361
bicycle      1656                 340
boat         2489                 265
bottle       1696                 272
bus          1356                 283
car          6194                 594
chair        2059                 1032
diningtable  2611                 288
motorbike    1570                 339
sofa         1749                 294
train        1601                 298
tvmonitor    1570                 336
```

In [54]:
```
# Get statistics of pairwise labels
lblType    = 'diff'
numSamples = 20000
imSz       = 128
cropType   = 'contPad'
prms       = p3d.get_exp_prms(imSz=imSz, lblType=lblType, numSamplesTrain=numSamples,
pairLabels = p3d.get_pair_label_stats(prms, setName='train')
```

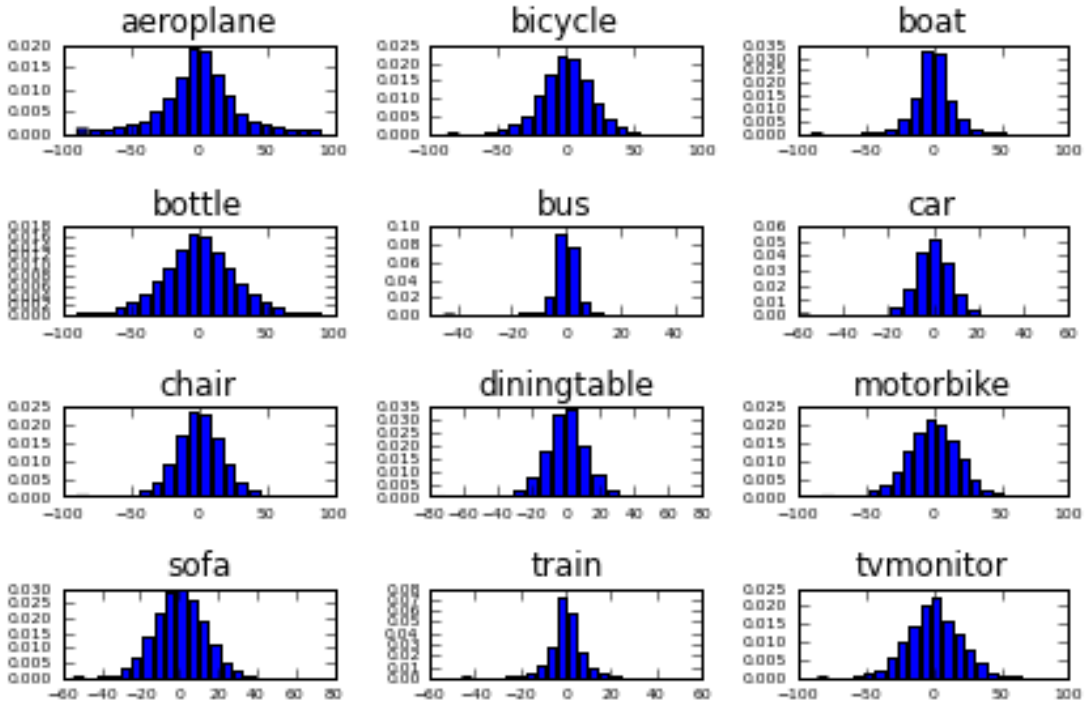
In [58]:
```
count = 1
print "Distribution of azimuth"
for (key, val) in sorted(pairLabels.items()):
    ax      = plt.subplot(4,3,count)
    azimuth = val[:,0]
    n, bins, patches = plt.hist(azimuth, 20, normed=1, histtype='bar', rwidth=0.8)
    plt.title(key)
    ax.tick_params(axis='x', labelsize=6)
    ax.tick_params(axis='y', labelsize=6)
    count += 1
plt.tight_layout()
```

```
Distribution of azimuth
```



In [59]:
```
#pairLabels = p3d.get_pair_label_stats(prms, setName='val')
count = 1
fig = plt.figure()
print "Distribtuion of Elevations"
for (key, val) in sorted(pairLabels.items()):
    ax      = plt.subplot(4,3,count)
    elevation = val[:,1]
    n, bins, patches = plt.hist(elevation, 20, normed=1, histtype='bar', rwidth=0.8)
    plt.title(key)
    ax.tick_params(axis='x', labelsize=6)
    ax.tick_params(axis='y', labelsize=6)
    count += 1
plt.tight_layout()
```

Distribtuion of Elevations

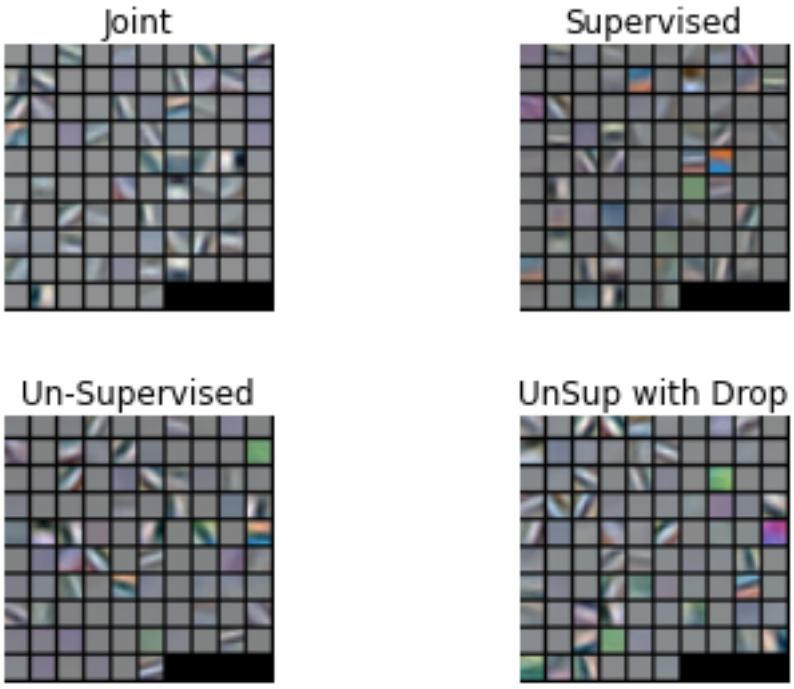


In [24]:

```
#New training where instead of taking the Medoid of the cluster, I trained on the bins
# These examples contain information from all boxes whose minimum size > 50px in the o

prms   = p3d.get_exp_prms(cropType='contPad', numSamplesTrain=40000, lblType='uniform')
numIter = 50000
fig = plt.figure()
ax1 = plt.subplot(2,2,1)
ax2 = plt.subplot(2,2,2)
ax3 = plt.subplot(2,2,3)
ax4 = plt.subplot(2,2,4)

## Get Supervised + rotation loss example
#Unforutnately this was not done programattically :/
snapDir   = '/data1/pulkitag/pascal3d/snapshots/'
modelFile = os.path.join(snapDir, ("imSz128_lbl-uni-az30el10_crp-contPad16_ns4e+04_mb5
                                   "iter_%d.caffemodel") % numIter)
defFile =    ("/work4/pulkitag-code/pkgs/caffe-v2-2/modelFiles/pascal3d/exp/"
              "imSz128_lbl-uni-az30el10_crp-contPad16_ns4e+04_mb50/caffenet_siamese_fc6
netJoint   = mp.MyNet(defFile, modelFile)
print modelFile
netJoint.vis_weights('conv1', ax=ax1, titleName='Joint')
ax1.axis('off')

#Get the Supervised example.
cPrms = p3d.get_caffe_prms(isScratch=True, isClassLbl=True, concatLayer='fc6', noRot=T
exp = p3d.get_experiment_object(prms, cPrms)
exp.init_from_self()
modelFile = exp.get_snapshot_name(numIter=numIter)
defFile   = exp.files_['netdef']
netSup    = mp.MyNet(defFile, modelFile)
print modelFile
netSup.vis_weights('conv1', ax=ax2, titleName='Supervised')
ax2.axis('off')

#Get  Unsupervised example.
cPrms = p3d.get_caffe_prms(isScratch=True, isClassLbl=False, concatLayer='fc6', noRot=
```

```
exp = p3d.get_experiment_object(prms, cPrms)
exp.init_from_self()
modelFile = exp.get_snapshot_name(numIter=numIter)
defFile   = exp.files_['netdef']
netUnsup    = mp.MyNet(defFile, modelFile)
print modelFile
netUnsup.vis_weights('conv1', ax=ax3, titleName='Un-Supervised')
ax3.axis('off')

#Get  Unsupervised example.
cPrms = p3d.get_caffe_prms(isScratch=True, isClassLbl=False, concatLayer='fc6', noRot=
exp = p3d.get_experiment_object(prms, cPrms)
exp.init_from_self()
modelFile = exp.get_snapshot_name(numIter=numIter)
defFile   = exp.files_['netdef']
netUnsup    = mp.MyNet(defFile, modelFile)
print modelFile
netUnsup.vis_weights('conv1', ax=ax4, titleName='UnSup with Drop')
ax4.axis('off')
plt.tight_layout()
```

```
/data1/pulkitag/pascal3d/snapshots/imSz128_lbl-uni-az30el10_crp-
contPad16_ns4e+04_mb50/pascal3d_scratch_concat-
fc6_iter_50000.caffemodel
Ignoring line: # Autotmatically generated solver prototxt
/data1/pulkitag/pascal3d/snapshots/pascal3d_imSz128_lbl-uni-
az30el10_crp-contPad16_ns4e+04_mb50/caffenet_scratch_sup_noRot_fc6_ite
r_50000.caffemodel
Ignoring line: # Autotmatically generated solver prototxt
/data1/pulkitag/pascal3d/snapshots/pascal3d_imSz128_lbl-uni-
az30el10_crp-contPad16_ns4e+04_mb50/caffenet_scratch_unsup_fc6_iter_50
000.caffemodel
Ignoring line: # Autotmatically generated solver prototxt
/data1/pulkitag/pascal3d/snapshots/pascal3d_imSz128_lbl-uni-
az30el10_crp-contPad16_ns4e+04_mb50/caffenet_scratch_unsup_fc6_drop_it
er_50000.caffemodel
```



This is weird that the filters with k-Medoid clustering look so much better than filters obtained with the Euler angle uniform binning. The critical differences between the two approaches is as following:

1. First only considers small rotations (< 30 degrees) and bins all the other rotation into twenty categories.
2. First uses 0.005 weight std to initialize the FC layers, the latter uses 0.01 for initialization. This was done as with 0.005 init, the second case I was getting all zeros as output in the top layers.
3. First uses pascal only, whereas second used imagenet also.
4. There is a difference in classes that are bein used. First uses a set of 15 manually chosen classes that are mostly rigid, 12 uses the 12 Rigid objects in pascal3d. The first includes the classes of person, sheep, bird, cow and horse that are ignored by the second. The second has bottle and diningtable which is not there in the first.

Possible things to do:

1. Compare accuracy on an auxiliary dataset - say Caltech101? Any suggestions?
2. The issue with using less data - should I use the same architecture as the one used by the unsupervised method - which has a lot more training data available? Is that fine?

In []:
```
#The visualizations for the picking data.
#I figured I havent trained a joint model for azimuth and polar angle which works - so
```

In [36]:
```
#Performance on caltech
ou = reload(ou)
maxLayer    = [1,2,3,4,5,6]
prms = pc.get_prms()
accScratch, accClassScratch = pc.read_accuracy(prms, isPreTrain=False, preTrainStr=Non
                                               isFineLast=True, initLr=0.01)

accPre60K, accClassPre60K = pc.read_accuracy(prms, isPreTrain=True, preTrainStr='rotOb
                                               isFineLast=True, initLr=0.001)

accPre120K, accClassPre120K = pc.read_accuracy(prms, isPreTrain=True, preTrainStr='rot
                                               isFineLast=True, initLr=0.001)

alexPrms = pc.get_prms(imSz=256)
```

```
accAlex, accClassAlex = pc.read_accuracy(alexPrms, isPreTrain=True, preTrainStr='alex'

tableArgs = co.OrderedDict([('Layers',range(1,7)), ('Scratch', accClassScratch),
            ('Pre60K', accClassPre60K), ('Pre120K', accClassPre120K), ('Alex', accClas
print "Mean Accuracy of classes - Finetune only the last layer."
print '-' * 50
ou.make_table(**tableArgs)
```

```
Mean Accuracy of classes - Finetune only the last layer.
--------------------------------------------------
Layers          Scratch         Pre60K          Alex
Pre120K

--------------------------------------------------------------------------
-----

1               0.395           0.447           0.529           0.429

2               0.452           0.552           0.699           0.554

3               0.443           0.538           0.745           0.545

4               0.438           0.498           0.789           0.560

5               0.252           0.507           0.860           0.531

6               0.173           0.397           0.882           0.480
```

In [17]:
```
maxLayer    = [1,2,3,4]
prms = pc.get_prms()
accScratch, accClassScratch = pc.read_accuracy(prms, isPreTrain=False, preTrainStr=Non
                                               isFineLast=False, initLr=0.0001, maxLayer=

accPre60K, accClassPre60K = pc.read_accuracy(prms, isPreTrain=True, preTrainStr='rotOb
                                               isFineLast=False, initLr=0.01, maxLayer=ma

tableArgs = co.OrderedDict([('Layers',maxLayer), ('Scratch', accClassScratch),
            ('Pre60K', accClassPre60K)])
print "Mean Accuracy of classes - Finetune ALL the layers."
print '-' * 50
ou.make_table(**tableArgs)
```

```
-------------------------------------------------------------------
-------------
    IOError                                   Traceback (most recent
call last)



        <ipython-input-17-129ccbe9890b> in <module>()
          2 prms = pc.get_prms()
          3 accScratch, accClassScratch = pc.read_accuracy(prms,
isPreTrain=False, preTrainStr=None,
    ----> 4
isFineLast=False, initLr=0.0001, maxLayer=maxLayer)
          5
          6 accPre60K, accClassPre60K = pc.read_accuracy(prms,
```

```
isPreTrain=True, preTrainStr='rotObjs_kmedoids30_20_iter60K',



        /work4/pulkitag-code/pkgs/caffe-v2-2/modelFiles/keypoints/docu
mentation/process_caltech.pyc in read_accuracy(prms, isPreTrain,
preTrainStr, isFineLast, maxLayer, initLr, initStd)



        /usr/lib/python2.7/dist-packages/h5py/_hl/files.pyc in
__init__(self, name, mode, driver, libver, userblock_size, **kwds)
        205
        206             fapl = make_fapl(driver, libver, **kwds)
    --> 207             fid = make_fid(name, mode, userblock_size,
fapl)
        208
        209         Group.__init__(self, fid)



        /usr/lib/python2.7/dist-packages/h5py/_hl/files.pyc in
make_fid(name, mode, userblock_size, fapl, fcpl)
         77
         78     if mode == 'r':
    ---> 79         fid = h5f.open(name, h5f.ACC_RDONLY, fapl=fapl)
         80     elif mode == 'r+':
         81         fid = h5f.open(name, h5f.ACC_RDWR, fapl=fapl)



        /usr/lib/python2.7/dist-packages/h5py/h5f.so in h5py.h5f.open
(h5py/h5f.c:1806)()



        IOError: unable to open file (File accessibilty: Unable to
open file)
```

imSz128_ntr30_run0 pre-random_ft-all_mxl-1_inLr1e-04

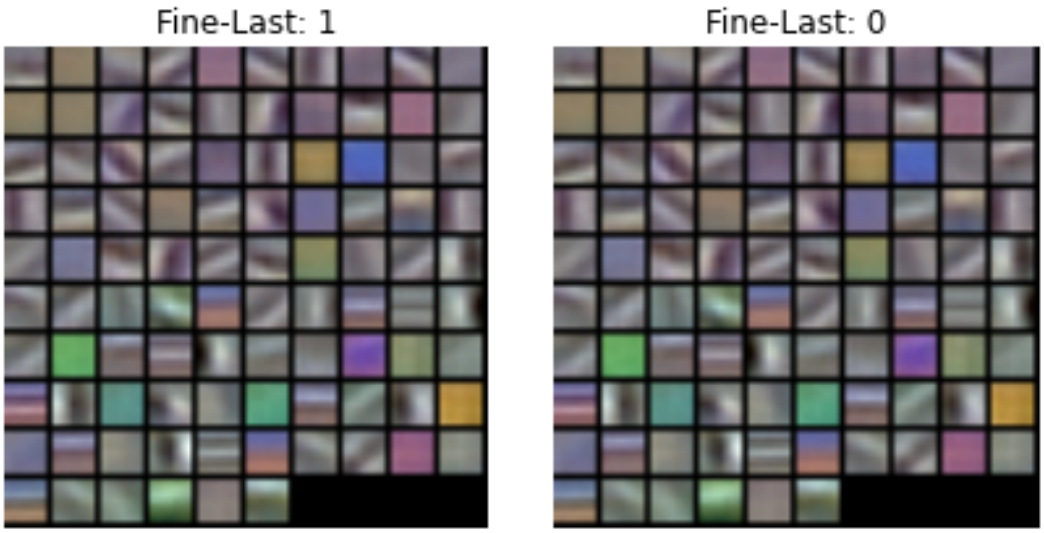
In [39]:
```
accClassScratch
```

Out [39]:
```
array([ 0.49164349,  0.5151068 ], dtype=float32)
```

In [8]:
```
#Visualize weights after finetuning.
prms  = pc.get_prms()
preTrainStr = 'rotObjs_kmedoids30_20_nodrop_iter120K'
isFineLast  = [True, False]
initLr = [0.001, 0.000001]
initStd = [0.01, 0.001]
```

```
for (i,ff) in enumerate(isFineLast):
    cPrms = pc.get_caffe_prms(isPreTrain=True, maxLayer=1,
                              preTrainStr=preTrainStr, isFineLast=ff,
                               initLr=initLr[i], initStd=initStd[i], testNum=None)
    ax = plt.subplot(1,len(isFineLast),i+1)
    exp = pc.get_experiment_object(prms, cPrms)
    exp.init_from_self()
    modelFile = exp.get_snapshot_name(numIter=5001)
    defFile   = exp.files_['netdef']
    netUnsup     = mp.MyNet(defFile, modelFile)
    print modelFile
    netUnsup.vis_weights('conv1',ax=ax, titleName='Fine-Last: %d' % ff)
    ax.axis('off')
plt.tight_layout()
```

```
Ignoring line: # Autotmatically generated solver prototxt
/data1/pulkitag/caltech101/snapshots/imSz128_ntr30_run0/caffenet_pre-
rotObjs_kmedoids30_20_nodrop_iter120K_ft-last_mxl-1_inLr1e-
03_iter_5001.caffemodel
Ignoring line: # Autotmatically generated solver prototxt
/data1/pulkitag/caltech101/snapshots/imSz128_ntr30_run0/caffenet_pre-
rotObjs_kmedoids30_20_nodrop_iter120K_ft-all_mxl-1_inLr1e-06_inSd1e-
03_iter_5001.caffemodel
```



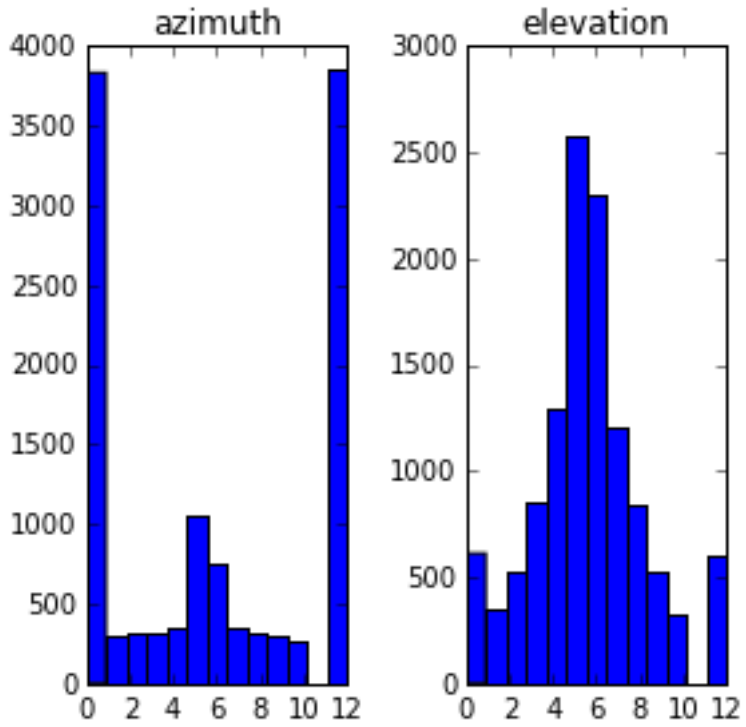
0.1 Current Thoughts

One thing which is still unclear is what is the best way of supervision to train the networks. Should it be regression, classification, within classification should it be predicting uniformaly across all bins, or limit the amount of transformation or something else? I have experiments with:

1. Regression - seems to be hard to train.
2. Classifying euler angles uniformly – seems to produce a lot of dead filters.
3. K-Medoid clustering only on objects within 30 degrees seems to produce the best filters at the moment.
4. Now, I am going to try to predict the euler angles seperately but within 30 degrees only.

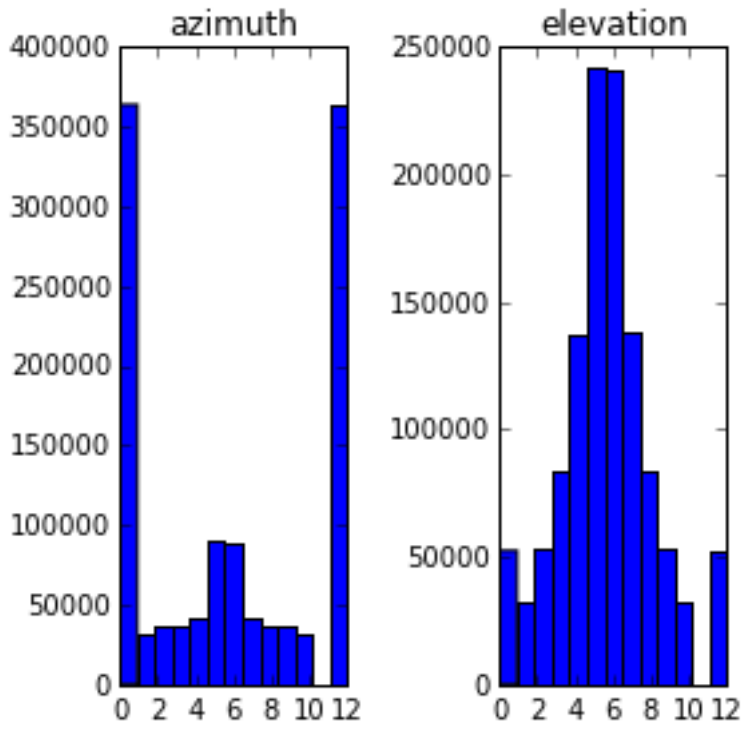
In [16]:
```
prms  = p3d.get_exp_prms(imSz=256, lblType='rotLim', numSamplesTrain=1e+05, azBin=6, e
names = ['azimuth', 'elevation']
#Distribution in the val set
gen    = mpio.GenericWindowReader(prms['paths']['windowFile']['val'])
lbls   = gen.get_all_labels()
fig = plt.figure()
for i in range(2):
    ax = plt.subplot(1,lbls.shape[1], i+1)
    plt.hist(lbls[:,i], len(prms['azBinRange']) + 2)
    plt.title(names[i])
plt.tight_layout()
count = ou.count_unique(lbls, 12)
print count/sum(count)
```

```
All lines already read
[ 0.15122222  0.04547222  0.05127778  0.06038889  0.07338889
0.12858333
  0.1125      0.07088889  0.05988889  0.05086111  0.04402778
0.02777778
  0.12372222]
```



In [12]:
```
#Distribution in the val set
gen    = mpio.GenericWindowReader(prms['paths']['windowFile']['train'])
lbls   = gen.get_all_labels()
fig = plt.figure()
for i in range(2):
    ax = plt.subplot(1,lbls.shape[1], i+1)
    plt.hist(lbls[:,i], len(prms['azBinRange']) + 2)
    plt.title(names[i])
plt.tight_layout()
```

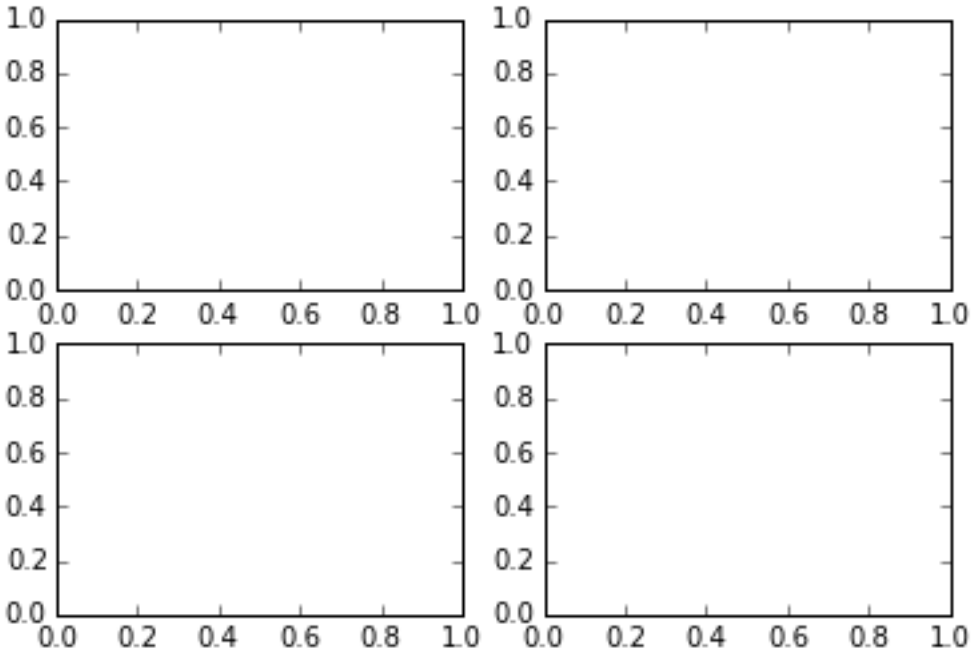
```
All lines already read
```



In [10]:
```
prms  = p3d.get_exp_prms(cropType='contPad', numSamplesTrain=100000, lblType='rotLim')
numIter = 50000
fig = plt.figure()
ax = plt.subplot(2,2,1)
ax2 = plt.subplot(2,2,2)
ax3 = plt.subplot(2,2,3)
ax4 = plt.subplot(2,2,4)
cPrms = p3d.get_caffe_prms()
exp = pc.get_experiment_object(prms, cPrms,0)
exp.init_from_self()
modelFile = exp.get_snapshot_name(numIter=10000)
defFile   = exp.files_['netdef']
netUnsup    = mp.MyNet(defFile, modelFile)
print modelFile
netUnsup.vis_weights('conv1',ax=ax, titleName='Fine-Last: %d' % ff)
ax.axis('off')
plt.tight_layout()
```

```
---------------------------------------------------------------------
-------------
    KeyError                                  Traceback (most recent
call last)


        <ipython-input-10-fd456c77a7c4> in <module>()
          7 ax4 = plt.subplot(2,2,4)
          8 cPrms = p3d.get_caffe_prms()
    ----> 9 exp = pc.get_experiment_object(prms, cPrms,0)
         10 exp.init_from_self()
         11 modelFile = exp.get_snapshot_name(numIter=10000)
```

```
        /work4/pulkitag-code/pkgs/caffe-v2-2/modelFiles/keypoints/docu
mentation/process_caltech.pyc in get_experiment_object(prms, cPrms,
deviceId)


        KeyError: 'expDir'
```



In []: