Classify trees

2015213506 Youyou Jiang

school of software

Table 1. Experiment environment

CPU	Intel core i5 4670
Memory	12G
disc	500G
GPU	GTX 780
OS	Ubuntu 14.04
Python	2.7.11
Anaconda2	4.0.0
gnuplot	5.0
jupyter	4.1.0

1

1.1 Pretreatment

After all the picture are resized to the size of $256 * 256$

1.Convert every picture to a vector of $1 * 10000$ and a label to form a matrix of $n * 10000$ named trainX and a matrix of $1 * n$ named trainY(when reading a picture,We resize it to $100 * 100$ and transform it into a Grayscale one)
2.Use pca to extract the first 1000 features
3.When applied KNN to the extracted feature,We have obtained a error rate of 0.427

1.2 Note:

Generate train.txt and val.txt:
put the files $train$,val in the path of src
put the in the same path with the folder train and execute the following commands:
$python \quad ProduceTrainvalTxt.py \quad -i \quad ../train \quad -o \quad train.txt$
$val.txt$ is produced same as $train.txt$
Resize to $256 * 256$
run:$sudo \quad python \quad -i \quad ../train$
Classify
run $python \quad Problem1.py$

2 Deep convnet

Table 2. Where the file should be put in

file	path
$Problem2.ipynb$	$caffe\_root/examples/$
$imagenet\_mean.binaryproto$	$caffe\_root/data/ilsvrc12/$
$tree\_mean.npy$	$caffe\_root/examples/imagenet/$
$*.prototxt$ (automatically generated)	$caffe\_root/examples/imagenet/$
$caffenet\_train\_iter\_20000.*$	$caffe\_root/examples/imagenet/$
$mygnuplotforcaffe.gnuplot$ and $parse_log.sh$	$caffe\_root/tools/extra/$

Table 3. list to be changed

cell 7,9	caffemodel path
cell 12,18	picture path
cell 18	testsnap.txt(Details see sample iteration steps)

2.1 Note:

In the caffe_root open *jupyter*,visit **http://localhost:8888/tree** and open *examples/Problem2.ipynb*,and choose Run All.

1.construction and display of the net
2.conv1(including filter and feature)'s visualization,
3.single sample's output changes over iteration,
4.histogram statistics on different layers.

Having trained over 20000 iterations,The test accuracy increased to 0.968
The following picture is about loss/accuracy vs iteration and obtained by means of the following steps:

redirect the console to a txt using *tee*
run parse_log.sh,
run mygnuplotcaffe.gunplot(using gnuplot)
For details,please refer to parse_log.sh and mygnuplotforcaffe.gnuplot

2.2 net setup and display

The net setup was implemented in the second cell of *Problem2.ipynb*,These codes produce 4 files of $*.prototxt$(solver,train,val,deploy)
The cell 5 implemented visualizing the net.

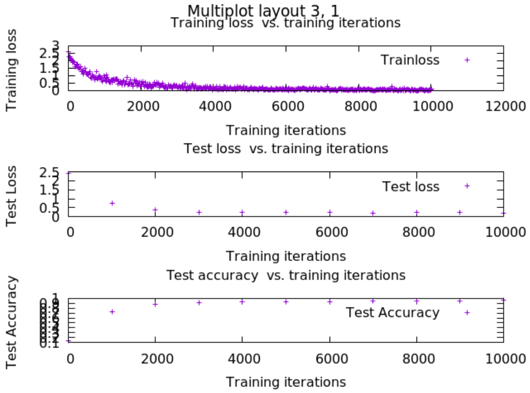


Fig. 1. loss,accuracy vs iteration

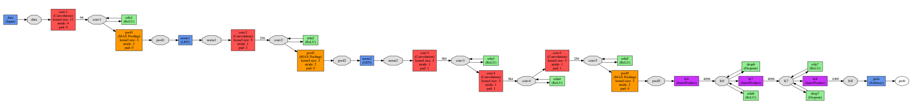


Fig. 2. net display

2.3 filter visualize

Note:cells from 7 to 16 must be run before extracting features and displaying filters.
The following picture are parameters in conv1 kernel and features filtered by conv1.

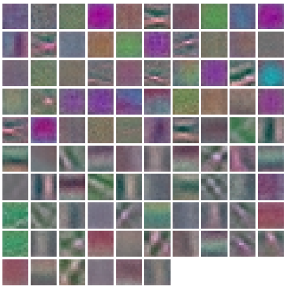


Fig. 3. filter parameters in conv1 visualization

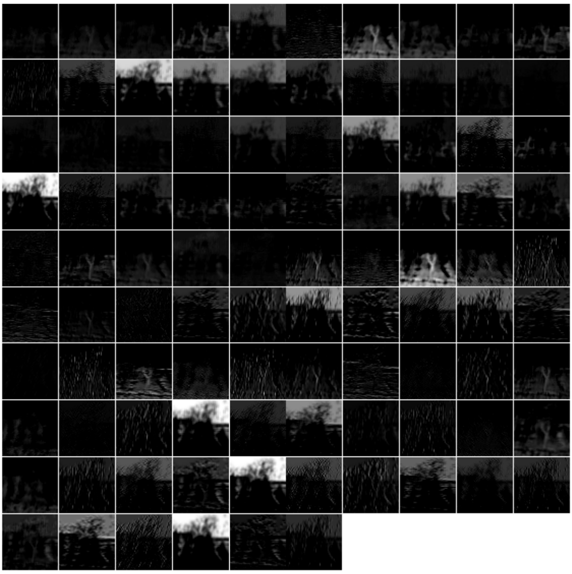


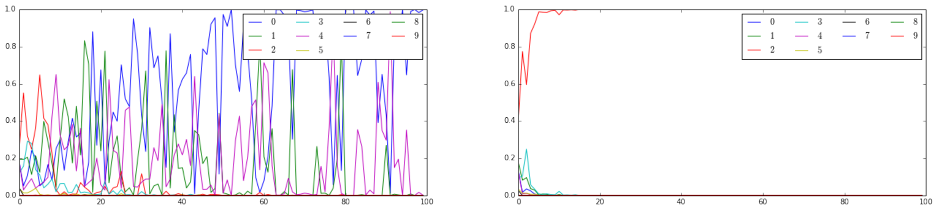
Fig. 4. Conv1 features display

2.4 Sample Iteration

The implementation of output iteration were saved in the last 2 cell of $caffe\_root/exapm$ as follows:

1.modify the snapshot frequency from 10000 to 100;
2.When Training CNN, We use linux 'tee' command to redirect the console content to a file.(append $2 > \&1|tee \quad testsnap.txt$ to the common command $sudo \quad ./build/tools/caffe \quad train...$)
3.Get all the $caffemodel$ path in the $output.txt$ to form a string list(see details,please refer to function $snapshotlist$ in $Problem.ipynb$)
4.For 2 different sample,apply different $.caffemodel$ to get features of $prob$ layer and get a matrix of $2 * caffemodelnum * NumberClasses$,Here the $NumberClasses$ has been set to 10 in previous net building.$caffemodelNum$ usually equals iterations divided 100.
5.For 2 different samples, for different $Numberclass$,plot matrix[i,:,j] vs 1:caffemodelNum

In graph below,left one is corresponding to a willow,legend 0 stands for willow, In the iteration from 0 to 100,There exists a few probability that y of legend 0 less than others, which this picture maybe was predicted as other tree and contributed to the training cost. In contrary, the right one is corresponding to a cedar, In the iteration,y of legend 2(cedar) is always far greater than others and not disclassified.



Willow Cedar

Fig. 5. prob output vs iteration

2.5 Histogram statistics on different layers

The src codes is saved in the cell 18,19 of *Problem2.ipynb*,And the function *GethistogramRange* worked for calculating ranges embracing 98% of layer parameters.
Fig.6 show the parameter distribution of all layers.

The src codes implementing histogram on different layers in 17th cell

3 Deep network extract feature

The src codes implementing feature extraction is saved in the path *examples/Problem3.i* of *caffe_root*,
Using previous trained results *caffenet.solverstate*, I have extracted features of *pool*5 and *fc*6 saved in the form of *.mat*.
Classifiers used here are composed of knn,adaboost(decision tree),bpnn.Their src codes are in the folder of *Problem*2 The below table has recoded the accuracy applied different classifiers to *pool*5 features and *fc*6 features

Table 4. Classifers accuracy on different extracted features

Accuracy \ methods / features	KNN	adaboost	neural network
pool5	0.88	0.85	0.8
fc6	0.95	0.9	0.94

Table 5. Classifers elapsed time accuracy on different extracted features

Elapsed time \ methods / features	KNN	adaboost	neural network
pool5	325s	1137s	1100s
fc6	146s	331s	465s

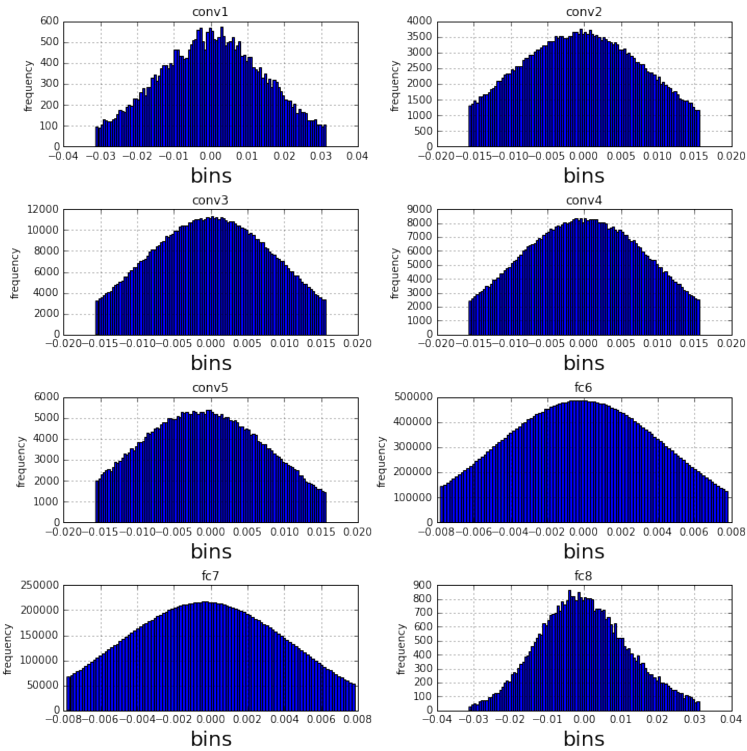


Fig. 6. Histogram statistics on different layers

The below pictures are error rate changes over iterations using methods of $adaboost(decisiontree(Fig.7))$ and $bpnn(Fig.8)$

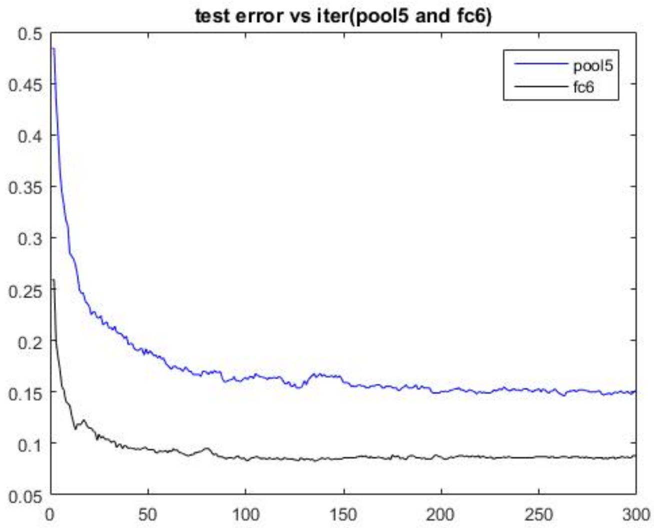


Fig. 7. adaboost decision tree (pool5 and fc6)

4 Deep network tricks

Use $Problem.ipynb$ to change the net, and use command to train the net.

4.1 Some tricks

1.learning rate:
1.1 learning rate initial value is very important,if not suitable, Cost may go to NaN,
1.2 when cost changes are approximately flatten, We can decreased learning rate to get a new decreasing of cost. (when convergence on a learning rate, stop and resuming on a lower learning rage)
2.Extracted features far away from the output lead to low accuracy for other classifiers.
3. when $NumberClass$ is different,the less the $NumberClass$,the more centralized the $Conv$ or fc weight distribution.(depending histograms when $NumberClass = 10$ compared to $NumberClass = 1000$),$Fig.9$ show the parameter distribution of $NumberClass = 1000$
4.When there exists not the layer of Norm, The final accuracy was not affected.
5.When $Relu$ layer was deleted from the Net, The cost will not convergence.

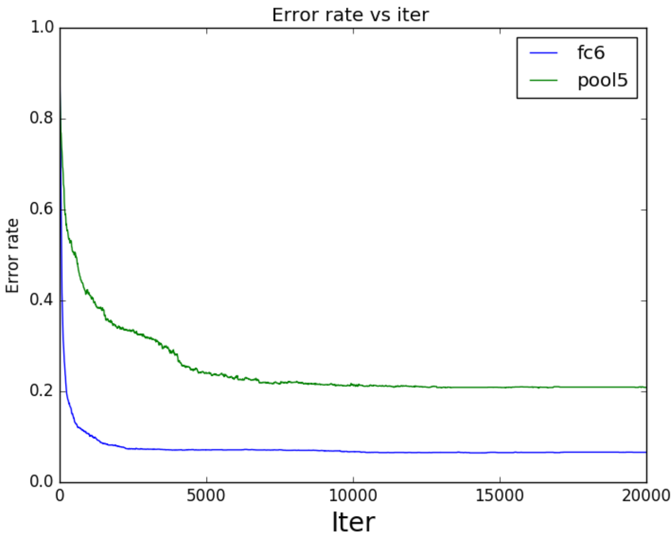


Fig. 8. neural network(pool5 and fc6)

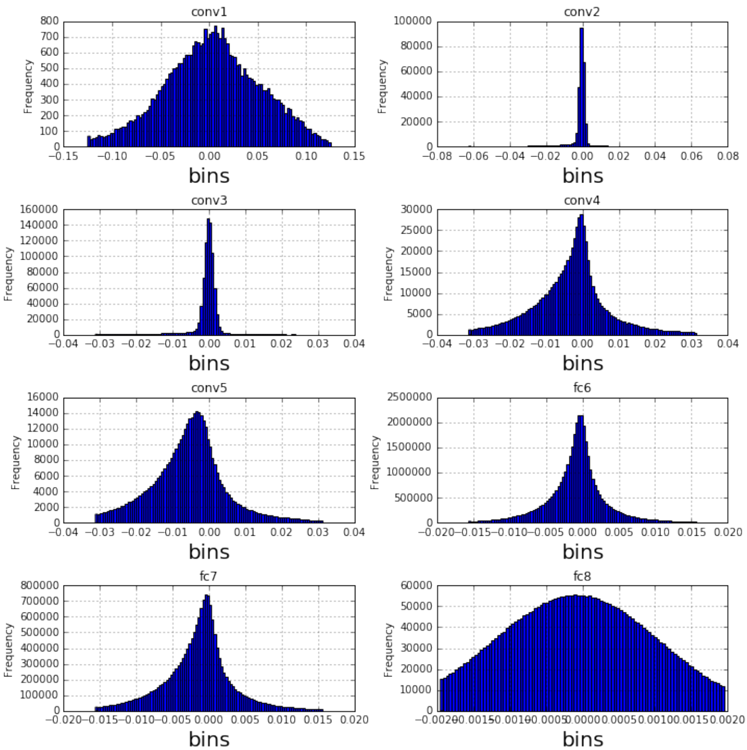


Fig. 9. Histogram statistics on different layers