工作记录

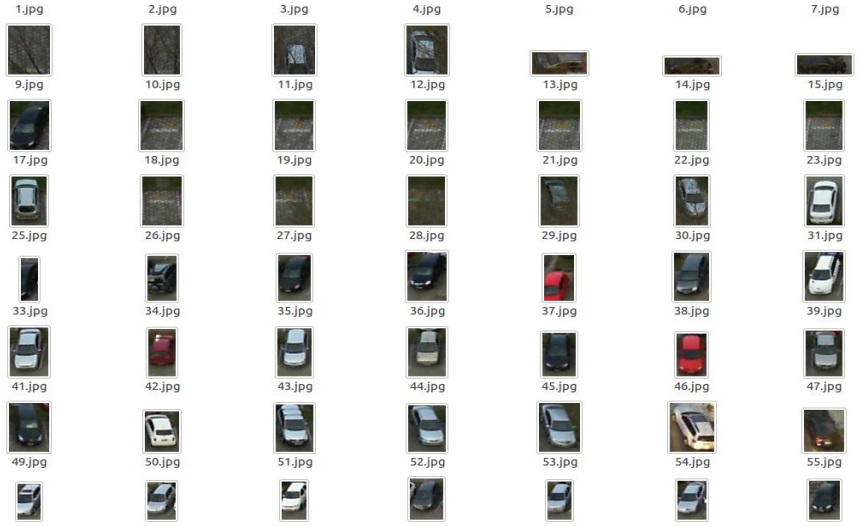
1.常用分类器分类车位(空缺和非空缺)
2.分析实验结果并优化

1.常用分类器分类车位:

a) **数据:训练测试数据集**

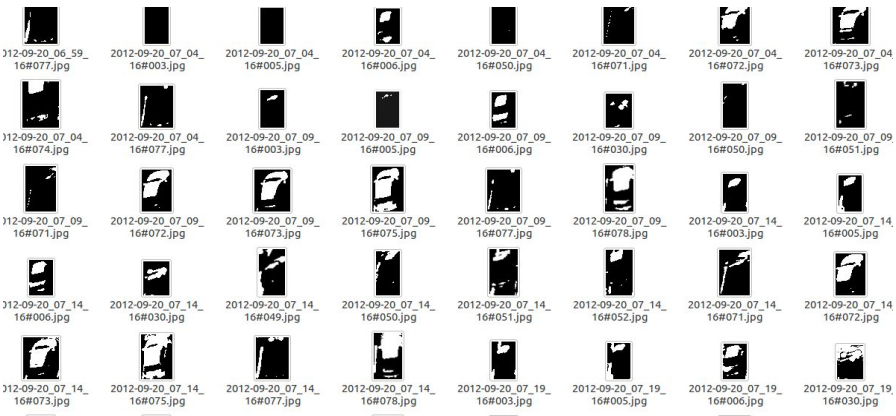
40W 张车位图片:30W 张训练,10W 张测试

图片贴图:



b) **数据处理：图像二值化处理 归一化大小 lmdb 存储**

二值化处理后图像贴图:



归一化大小和 lmdb 存储:

```
EXAMPLE=examples/imagenet/myimages
TOOLS=build/tools

TRAIN_DATA_ROOT=/home/gnss/devdata/Data/PKLot/PKLotSegmented/PUC/
VAL_DATA_ROOT=/home/gnss/devdata/Data/PKLot/PKLotSegmented/PUC/

# Set RESIZE=true to resize the images to 256x256. Leave as false if images have
# already been resized using another tool.
RESIZE=true
if $RESIZE; then
  RESIZE_HEIGHT=60
  RESIZE_WIDTH=40
  echo "resize img"
else
  RESIZE_HEIGHT=0
  RESIZE_WIDTH=0
fi

if [ ! -d "$TRAIN_DATA_ROOT" ]; then
  echo "Error: TRAIN_DATA_ROOT is not a path to a directory: $TRAIN_DATA_ROOT"
  echo "Set the TRAIN_DATA_ROOT variable in create_imagenet.sh to the path" \
       "where the ImageNet training data is stored."
  exit 1
fi

if [ ! -d "$VAL_DATA_ROOT" ]; then
  echo "Error: VAL_DATA_ROOT is not a path to a directory: $VAL_DATA_ROOT"
  echo "Set the VAL_DATA_ROOT variable in create_imagenet.sh to the path" \
       "where the ImageNet validation data is stored."
  exit 1
fi
```

处理后文件图:

c) 算法选择

i. 常用机器学习算法:SVM KNN LF RF NB 等

主要代码贴图:

```
print 'reading training and testing data...'
train_x, train_y, test_x, test_y = read_data(data_file)
num_train, num_feat = train_x.shape
num_test, num_feat = test_x.shape
is_binary_class = (len(np.unique(train_y)) == 2)
print '****************** Data Info *********************'
print '#training data: %d, #testing_data: %d, dimension: %d' % (num_train, num_test, num_fe

for classifier in test_classifiers:
    print '****************** %s ******************' % classifier
    start_time = time.time()
    model = classifiers[classifier](train_x, train_y)
    print 'training took %fs!' % (time.time() - start_time)
    predict = model.predict(test_x)
    if model_save_file != None:
        model_save[classifier] = model
    if is_binary_class:
        precision = metrics.precision_score(test_y, predict)
        recall = metrics.recall_score(test_y, predict)
        print 'precision: %.2f%%, recall: %.2f%%' % (100 * precision, 100 * recall)
    accuracy = metrics.accuracy_score(test_y, predict)
    print 'accuracy: %.2f%%' % (100 * accuracy)
```

最终结果贴图:

```
precision: 74.20%, recall: 29.63%
accuracy: 63.56%
****************** KNN ********************
training took 380.196391s!
precision: 84.22%, recall: 55.86%
accuracy: 75.34%
****************** LR ********************
training took 3.818952s!
precision: 73.49%, recall: 32.00%
accuracy: 64.07%
****************** RF ********************
training took 24.226505s!
precision: 93.32%, recall: 59.83%
accuracy: 79.92%
****************** DT ********************
training took 23.775869s!
precision: 94.19%, recall: 60.12%
accuracy: 80.31%
****************** SVM ********************
```

PS:写日志时 SVM 结果还未计算完毕

ii. 深度卷积网(使用 cifar10 训练后效果一般，0.75 左右 RGB 原始图)

2.实验结果分析:

从最终测试的 accuray 来看，最终的准确率并不高,以为算法使用错误，下载 mnsit 测试后，准确率平均 0.95 以上，原因可能在数据集上或者数据集的处理上

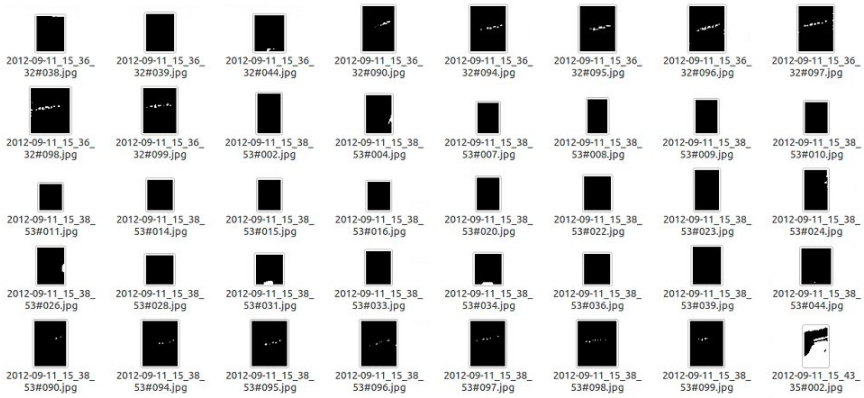
1.数据集分析:40W 张图像，数量足够，质量上人眼观察也足够清晰

2.数据集处理上:

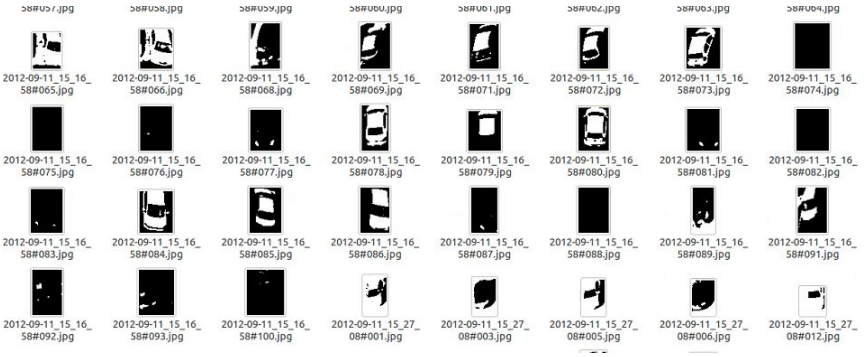
i. 图像二值化处理:观察处理后的图像发现空车位和非空车位在二值化处理后有不少全黑的

图像，也就是说，这部分不像对训练测试来说没有任何用途，还会影响最终的结果，二值化贴图如下：

空车位截图：



非空车位截图：



结论:可见二值化处理在此处并不适合

ii. **训练测试数据未进行缩放:最终输入算法的矩阵中的值在 0-255 之间，未缩放到 0-1 之间**

(PS:实验之前考虑了这个问题，实验时忘记处理，今后得注意，仔细认真)

优化方案:

对原始图像只进行缩放处理，不进行二值化处理

对输入算法算法的数据进行合理的缩放