

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное
учреждение высшего образования
"Южно-Уральский государственный университет
(национальный исследовательский университет)"
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

**ОТЧЕТ
по производственной практике**

бакалавра направления 02.03.02 "Фундаментальная информатика и информацион-
ные технологии"

Выполнил:
студент группы КЭ - 401
И.Т. Харрасов

Проверил:
Ст. преподаватель кафедры СП
Н.С. Силкина
Дата: _____
Оценка: _____

Челябинск-2017

Оглавление

Постановка задачи.....	3
Обзор литературы и существующих решений.....	4
Проектирование и реализация.....	5
Тестирование.....	11
Заключение.....	14
Литература.....	15
Приложение 1.....	16

Постановка задачи

На производственной практике были поставлены следующие задачи:

1. Изучить основы библиотеки компьютерного зрения OpenCV
2. Выявить наиболее оптимальные фильтры для предобработки изображений для распознавания объектов
3. Подготовить (разметить) данные для обучения каскада Хаара.
4. Обучить каскад Хаара и с помощью него найти объекты на изображении.

Во время производственной практики я работал с библиотекой компьютерного зрения OpenCV версии 3.3.0 в задачах 1, 2, 4; и программой разметки изображений labelImg версии 1.4.3 в задаче 3. Используемый язык программирования — Python версии 3.6

В первой задаче надо было разобраться с основными инструментами, операциями над изображениями (геометрические преобразования фигур, работа с фильтрами, поиск контуров и т. д.).

Во второй задаче с помощью инструментов библиотеки OpenCV нужно было найти оптимальные параметры настроек фильтров для устранения различных шумов и выделения контуров.

В третьей задаче нужно было разметить объекты распознавания в изображениях, для дальнейшего обучения учебной выборки и распознавания в тестовых выборках.

В четвертой задаче нужно было обучить каскад Хаара и найти искомые объекты на изображении. А также исследовать качество обучения каскада Хаара, в зависимости от различных параметров.

Обзор литературы и существующих решений

Главными источниками информации являлись: Официальная документация библиотеки компьютерного зрения OpenCV, документация языка программирования Python.

Для изучения теоретических основ обучения каскада Хаара использовалась статья Пола Виолы и Майкла Джонса «Быстрое обнаружение объектов с использованием расширенного каскада простых функций.» (Paul Viola and Michael J. Jones. Rapid object detection using a boosted cascade of simple features.)

Для решения задачи также использовалась статья компании Azoft «Разработка системы распознавания текста на кассовых чеках»

Проектирование и реализация

OpenCV — это библиотека, содержащие алгоритмы компьютерного зрения, обработки изображений и численных алгоритмов общего назначения с открытым кодом. Данная библиотека разделена на модули для решения различных задач: например, для различных преобразований изображения (модуль opencv_imgproc), обнаружения различных объектов (модуль opencv_objdetect), для работы с машинным обучением (модуль opencv_ml) и т. д.

Для решения поставленных задач использовался язык программирования Python. Для примера был взят один из первых программ по преобразованию изображения с помощью OpenCV (см. Рис. 1, Таблица 1)

```
from matplotlib import pyplot as plt
import cv2
import numpy as np
%matplotlib inline

img = cv2.imread('2.jpg', 1)
kernel = np.ones((3,3), np.uint8)
erosion = cv2.erode(img, kernel, iterations = 1)
dilation = cv2.dilate(img, kernel, iterations = 1)
opening = cv2.morphologyEx(img, cv2.MORPH_OPEN, kernel)
closing = cv2.morphologyEx(img, cv2.MORPH_CLOSE, kernel)
gradient = cv2.morphologyEx(img, cv2.MORPH_GRADIENT, kernel)
tophat = cv2.morphologyEx(img, cv2.MORPH_TOPHAT, kernel)
blackhat = cv2.morphologyEx(img, cv2.MORPH_BLACKHAT, kernel)

plt.imshow(erosion)
plt.show()
plt.imshow(dilation)
plt.show()
plt.imshow(opening)
plt.show()
plt.imshow(closing)
plt.show()
plt.imshow(gradient)
plt.show()
plt.imshow(tophat)
plt.show()
plt.imshow(blackhat)
plt.show()
```

Таблица 1: Пример рабочего кода



В дальнейшем, для нахождения оптимальных фильтров были использованы различные комбинации математических морфологий, размываний и фильтров. Эти комбинации в дальнейшем должны быть использованы, для обнаружения контуров и дальнейшего распознавания специализированными инструментами. Результаты использования комбинаций будут проведены в разделе «Тестирование».

Третья задача была связана с подготовкой данных для обучения каскада Хаара. Этот метод был основан на 4 концепциях:

1. Простые прямоугольные функции, называемые функциями Хаара. - используются для визуального обнаружения объекта
2. Интегральное Изображение для упрощения поиска. - здесь объединяются более мелкие куски изображения в более крупные. Интегральное значение для каждого пикселя есть сумма всех пикселей над ним и слева от него. Начиная с левого верхнего угла и совершая обход вправо и вниз, все

изображение может быть интегрировано с несколькими целочисленными операциями на пиксель (его значение).

3. Метод машинного обучения AdaBoost. AdaBoost - адаптивное усиление. Идея- если есть набор эталонных объектов, то можно составить один более совершенный и мощный классификатор. При этом в процессе обучения акцент делается на эталоны, которые распознаются «хуже».
4. Каскадный классификатор для эффективного совмещения множественных функций. Термин «каскады» вытекает из структуры организации «образцов». Набор классификаторов состоит из множества «слабых» классификаторов с целью создания одного «сильного» - метод машинного обучения под названием AdaBoost.

Для подготовки данных была использована программа labelImg(см. Рис. 2).

Данная программа имеет способность размечать прямоугольные области в изображениях и сохранять их в формате XML. В XML-файле сохраняются абсциссы и ординаты нижней левой и верхней правой точек, а также путь файла, название класса изображения и т. д.

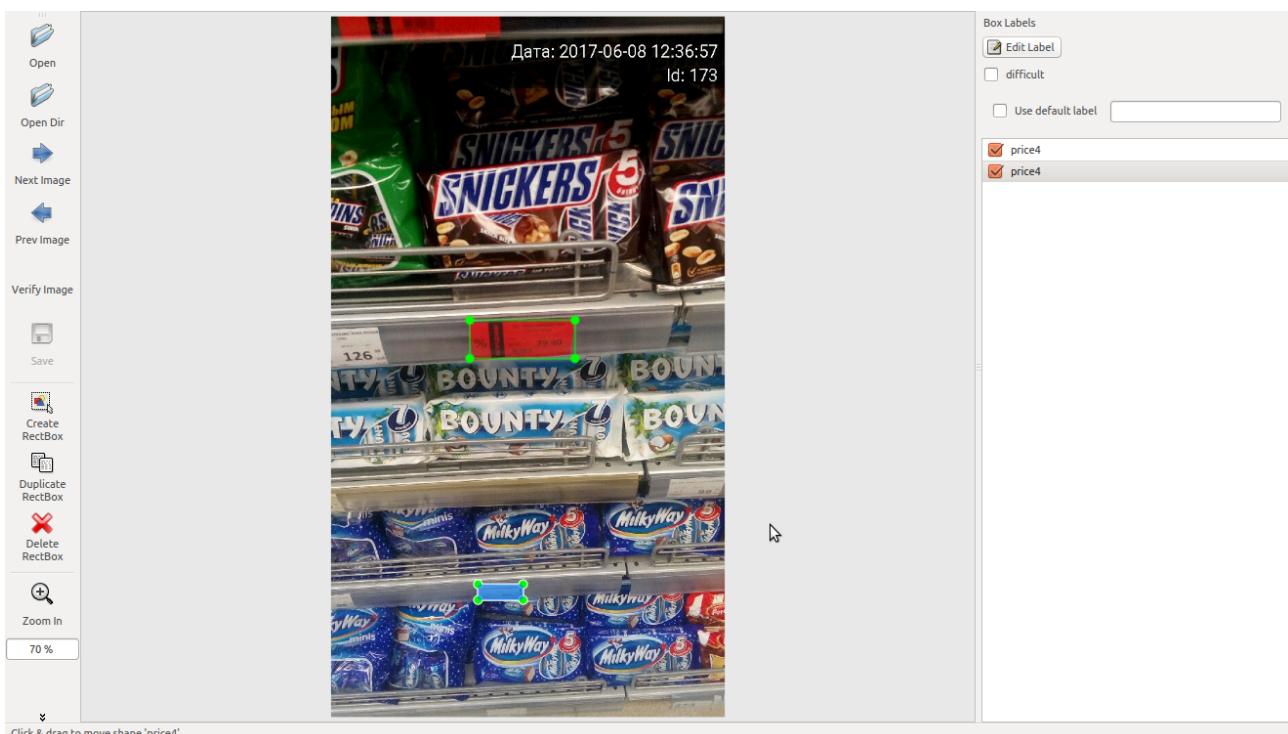


Рис. 2 Интерфейс программы labelImg

Данная программа предназначалась для разметки «хороших» данных — то есть для данных, которые должны содержать объект поиска. В данном случае объектом поиска являлся ценник в сети торговых центров. В ходе разметки около 5000 изображений были классифицированы в 5 классов. Подробнее классы описаны в разделе «Приложение»

«Плохие» данные не должны были содержать ценник совсем, или должны были охватить не более пятидесяти процентов от площади ценника при пересечении. Реализация кода приведена ниже:

```
import pandas
import cv2
import random
import time

df = pandas.read_csv('./good.csv')
df.path
img = cv2.imread(df.path[0], 1)
x, y, w, h = df.xMin[0], df.yMin[0], (df.xMax[0] - df.xMin[0]), (df.yMax[0] - df.yMin[0])
roi = img[y:y+h, x:x+w]
pos = cv2.rectangle(img, (x,y), (x+w,y+h), (0,0,0),1)

def haar_crops(n_begin, n_end):
    flags = 0
    for i in range(n_begin, n_end):
        for j in range(0,100):
            img = cv2.imread(df.path[i], 1)
            x, y, w, h = df.xMin[i], df.yMin[i], (df.xMax[i] - df.xMin[i]),
            (df.yMax[i] - df.yMin[i])
            x1 = random.randint(0, img.shape[1])
            y1 = random.randint(0, img.shape[0])
            w1 = random.randint(0, img.shape[1])
            h1 = random.randint(0, img.shape[0])

            #Проверка условия площади negative_sample
            S_image = img.shape[0] * img.shape[1]
            S_crop = w1 * h1
            is_S = S_crop/S_image > 0.1
            #Проверка на касание
            isTouchByX = x1 + w1 <= x
            isTouchByY = y1 + h1 <= y
            #Проверка на пересечение с positive_sample в глубину или ширину
            #1/2 от positive_sample
            isInterByX = x1 + w1 < int(0.5 * w + x)
            isInterByY = y1 + h1 < int(0.5 * h + y)
            #Проверка на выход за границы картинки
            isInterBorderByX = x1 + w1 < img.shape[1]
            isInterBorderByY = y1 + h1 < img.shape[0]
            #Проверка на нахождение negative_sample ниже чека
            isInterByX_A = int(0.5*w + x) < x1
            isInterByY_A = int(0.5*h + y) < y1

            if((isTouchByX | isTouchByY) & (is_S) & (isInterBorderByX &
            isInterBorderByY)):
                flags += 1
                roi_neg = img[y1:y1+h1, x1:x1+w1]
                cv2.imwrite('./BadCases/' + str(flags) + '_' + str(df.idIn-
Photo[i]) + '_' + df.filename[i], roi_neg)
                print(flags, end=' ')
                break
            elif ((isInterByX | isInterByY) & is_S & (isInterBorderByX &
            isInterBorderByY)):
                flags += 1
                roi_neg = img[y1:y1+h1, x1:x1+w1]
                cv2.imwrite('./BadCases/' + str(flags) + '_' + str(df.idIn-
Photo[i]) + '_' + df.filename[i], roi_neg)
```

```

        print(flags, end=' ')
        break
    elif ((isInterByX_A & isInterByY_A) & is_S & (isInterBorderByX
& isInterBorderByY)):
        flags += 1
        roi_neg = img[y1:y1+h1, x1:x1+w1]
        cv2.imwrite('./BadCases/' + str(flags) + '_' + str(df.idIn-
Photo[i]) + '_' + df.filename[i], roi_neg)

        print(flags, end=' ')
        break
    else:
        pass
        # print("N", end='')

haar_crops(200, 203)

```

Таблица 2: Создание "плохих" экземпляров

Затем «хорошие» и «плохие» данные должны быть записаны в файлы форматов .dat и .txt соответственно, причем, если в хороших файлах записывается не только полный путь, но и координаты пути, то «плохие» данные имеют только полный путь.

После всех подготовок можно приступать к обучению каскада Хаара. Для этого в терминале запускаются 2 команды: первая (см. Таблица 3) — создает файл формата *.vec, хранящий «хорошие» данные, вторая (см. Таблица 4) — запускает сам процесс обучения

```
opencv_createsamples -info /home/darius/haarcascade/trainpictures/good.dat -vec
/home/darius/haarcascade/samples.vec -w 40 -h 30
```

Таблица 3: Команда создания файла, хранящего "хорошие" данные

```
opencv_traincascade -data /home/darius/haarcascade -vec /home/darius/haarcas-
cade/samples.vec -bg /home/darius/haarcascade/bg.txt -numstages 25 -minhitrate
0.995 -maxFalseAlarmRate 0.5 -numPos 340 -numNeg 552 -w 40 -h 20 -mem 1024
-mode ALL
```

Таблица 4: Команда запуска обучения каскада Хаара

Тестирование

В данной таблице рассматривается использование различных фильтров и морфологических операций, используемых в библиотеке компьютерного зрения OpenCV для выявления четкости цифр в ценнике. Для теста была взята одна из фотографий (см. Рис. 3) и на ней были протестиированы, все фильтры и морфологические операции. По горизонтали расположены морфологические операции, по вертикали — фильтры или их композиции. Знак +/- обозначает, что для данного фильтра и операции потребуется дополнительные исследования.



Рис. 3 Тестовая картинка

	erode	dilate	morph_open	morph_close	morph_gradient	morph_top_hat	morph_blackhat
исходная	+	+	+	+	+	+	+
threshgauss1	-	-	-	-	-	-	-
threshgauss	-	-	-	-	-	-	-
tozero	-	+	+	+	+	+	+
trunk	+	+	+	+	+	+	+
blur	-	-	-	-	-	+	-
medianblur_k1	-	-	-	-	-	-	-
medianblur_k3	-	-	-	-	-	+	-
medianblur_k5	-	+/-	-	+	+/-	+	-
otsu+blur	-	+/-	-	+	+/-	+	-
otsu	-	+/-	-	+	+	+	-
laplassian+blur	-	-	-	-	-	-	-
sobel_x(sobel_y)	-	-	-	-	-	-	-
otsu+gaussblur	-	+/-	-	+	+	+	-

otsu+bilateria lf	-	+/-	-	+	+/-	+	-
----------------------	---	-----	---	---	-----	---	---

При проведении дополнительных исследований было установлено, что комбинация otsu+bilateral_filter+closing (Рис. 4) хорошо идет для размытия лишних символов и там остаются только цифры для распознавания.

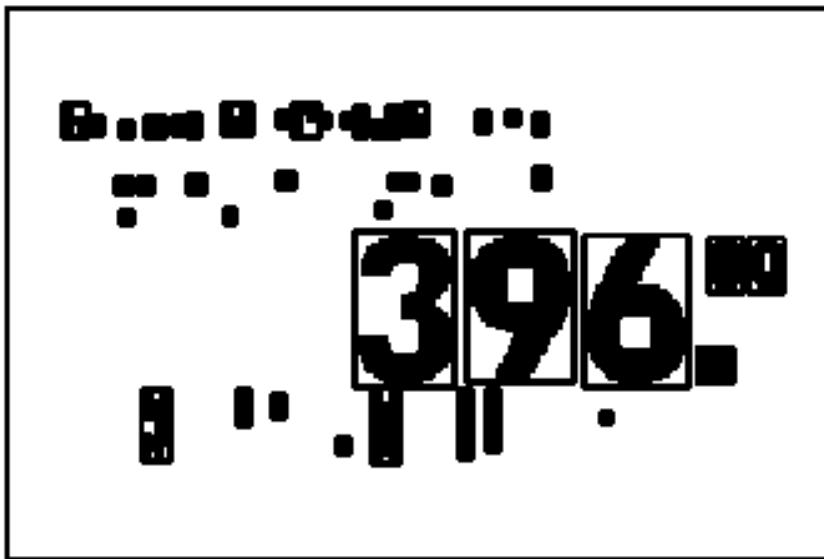


Рис. 4 Otsu+bilateral+closing, картинка сверху

Для каскада Хаара также было проведено исследование(см. Таблица 5), в котором исследовалось качество. Для тестирования было взято 522 «плохих» и 340 «хороших» фотографий. Коэффициент ложного срабатывания(FA) был взят равным 0.5, также количество итераций было взято равным 20(значения по

умолчанию). Для определения качества срабатывания была взята выборка из 60 фотографий с ценником, не входящая в обучающую выборку (см. Рис. 5).

Высота	Ширина	Время	Попадания	%
30	20	2 ч 37 мин 5 сек	29/60	48,3
20	15	49 мин 27 сек	8/60	13,3
40	20	8 ч 27 мин 25 сек	42/60	70

Таблица 5: Таблица сравнения каскада Хаара в зависимости от па-



Рис. 5 Полученный результат при обучении каскада Хаара параметров

В качестве параметров были взяты размеры шаблонов, как видно, при увеличении размеров шаблонов качество увеличивается, но и время тоже увеличивается.

Заключение

В ходе производственной практики были выполнены следующие задачи:

1. Изучить основы библиотеки компьютерного зрения OpenCV
2. Выявить наиболее оптимальные фильтры для предобработки изображений для распознавания объектов
3. Подготовить (разметить) данные для обучения каскада Хаара.
4. Обучить каскад Хаара и с помощью него найти объекты на изображении.

Также были получены следующие навыки:

1. Программирование на языке Python
2. Была освоена библиотека OpenCV

Литература

1. Официальный сайт библиотеки OpenCV <http://docs.opencv.org/3.3.0/>
2. Официальный сайт Python 3.6 <https://docs.python.org/3/tutorial/>
3. Paul Viola and Michael J. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511. IEEE, 2001.
4. Разработка системы распознавания текста на кассовых чеках URL: <http://www.azoft.ru/blog/razrabotka-sistemy-raspoznavaniya-teksta-na-kassovyh-chekah/>

Приложение 1

Название класса	Описание класса	Рисунок
Класс 1	Хорошо различимое наименование товара и цена на него, выполнено по шаблону сети, в которой произведён мониторинг	
Класс 2	Есть наименование и цена, но выполнено не по шаблону сети, а от руки	

Класс 3	Есть только цена, написанная от руки и наклеенная на товар	 <p>Дата: 2017-06-08 13:03:29 Id: 442</p>
Класс 4	Выполнено по шаблону, но плохо различима цена и/или наименование	 <p>Дата: 2017-06-08 12:36:57 Id: 173</p>

Класс 5

Ценники, не входящие ни в один из предыдущих классов, или объединяющие сразу несколько из предыдущих классов

