

## Розділ I. Вибір кросплатформних середовища і засобів розробки

Реалізація роботи здійснювалась у одному з найпопулярніших середовищ розробки VS2019 на WPF мовою C# (платформа .NET Framework версія 4.7.1).

Використовував відкриту кросплатформну бібліотеку комп'ютерного бачення EmguCV – обгортка OpenCV на C#.

OpenCV (Open Source Computer Vision library) - бібліотека функцій та алгоритмів комп'ютерного зору, обробки зображень і чисельних алгоритмів загального призначення [1]. Бібліотека надає засоби для обробки і аналізу вмісту зображень, у тому числі розпізнавання об'єктів на фотографіях (наприклад, осіб і фігур людей, тексту тощо), відстежування руху об'єктів, перетворення зображень, застосування методів машинного навчання і виявлення загальних елементів на різних зображеннях [1].

Після створення проекту ПКМ на проект у вікні «Оглядач рішень» → Керування пакетами NuGet. Потім встановлюємо наступну обгортку OpenCV для C#.

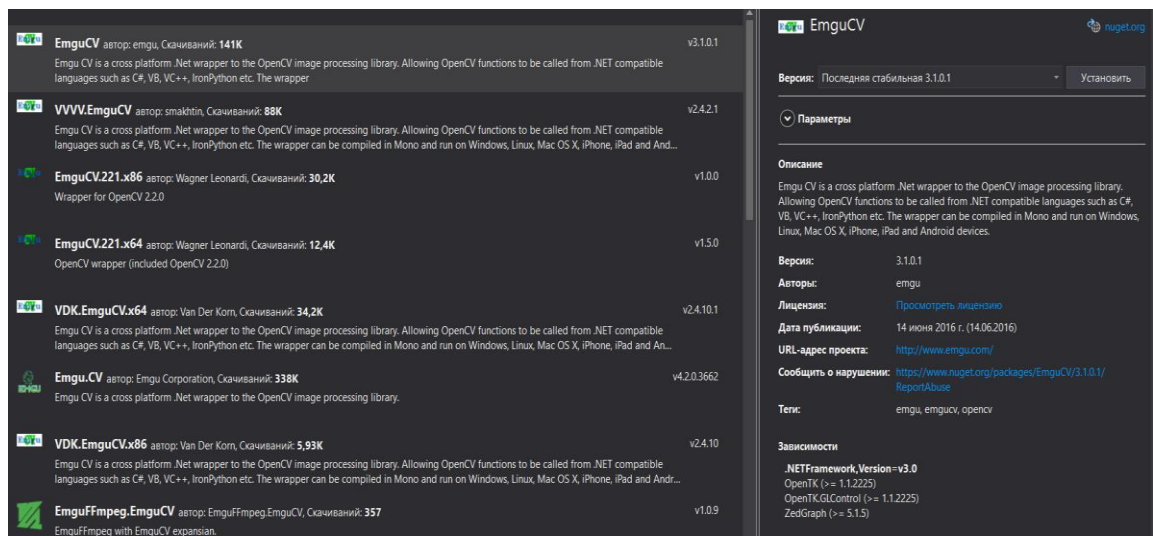


Рис. 1.1. Установлення EmguCV

Насправді вибір засобів не обмежений ніяк. Можна робити свої додатки, наприклад, на PHP з використанням GD, з використанням крос-платформного інструментарію Qt і на інших мовах програмування [2].

Далі завантажив “opencv-3.4.11” для створення зразків та каскаду [3]. Ця версія обґрунтована тим, що вона містить файли opencv\_createsamples, opencv\_traincascade і є простішою для використання. Також потрібно брати останню версію певної серії, оскільки бувають баги в попередніх версіях.

Теорія методу Віоли-Джонса чудово розписана та пояснюється у наступних статтях [4] [5] та відео [6].

## **Розділ II. Створення каскадів для розпізнавання дорожніх знаків**

### **2.1. Опис баз даних**

1) The German Traffic Sign Recognition Benchmark (GTSRB) - великий орієнтир класифікацій для багатьох категорій; багатокласний класифікатор [7].

- a) Містить більше 40 папок зі зображенням. Кожна папка має один вид дорожнього знаку з різною яскравістю та розміром;
- b) понад 50 000 зображень [7].

В кожній папці міститься файл .csv з описом «позитивних» зображень, а саме: Filename;Width;Height;Roi.X1;Roi.Y1;Roi.X2;Roi.Y2;ClassId

Filename – назва файлу зображення;

Width – ширина зображення;

Height – висота зображення;

Roi.X1, Roi.Y1, Roi.X2, Roi.Y2 - розташування знаку в межах зображення. Зображення містять межу навколо фактичного знаку 10 відсотків від розміру знаку, принаймні 5 пікселів;

ClassId - клас дорожнього знаку.

Більше інформації про цю базу даних міститься в файлі “Readme-Images.txt”.

2) The German Traffic Sign Detection Benchmark (GTSDB) являє собою оцінку виявлення з одним зображенням для дослідників, зацікавлених в області комп'ютерного зору та розпізнавання об'єктів [8]. У ньому представлені:

- a) 900 зображень (розподілених у 600 навчальних зображеннях та 300 зображеннях оцінювання);
- b) поділ на три категорії, що відповідають властивостям різних підходів виявлення з різними властивостями;
- c) онлайн-система оцінювання з негайним аналізом та ранжуванням поданих результатів [8].

В файлі «gt.txt» міститься опис об'єктів ідентично, як у файлах .csv бази даних GTSRB.

3) Файл “Test images 2017” - база «негативних» зображень. Там містить набір різних растрових зображень, в тому числі, на яких є дорожні знаки. Такі

зображення потрібно видалити, оскільки шукані об'єкти не можуть бути в базі «негативних» зображень.

## 2.2. Процес навчання

а) Підготовка даних.

“GTSRB\_Final\_Training\_Images.zip” можна скачати з [9], де містяться набори «позитивних» зображень. Як підготувати файли для створення зразків описується в [10].

В «FullIJCNN2013.zip», який можна скачати з [11], розташований тестовий набір даних та файл gt.txt з описом об'єктів на зображенні. За допомогою чого було уникнуто вирізання об'єктів з попередньої бази даних і вставлення в «негативні» зображення [8].

База «негативних» зображень скачена з [12].

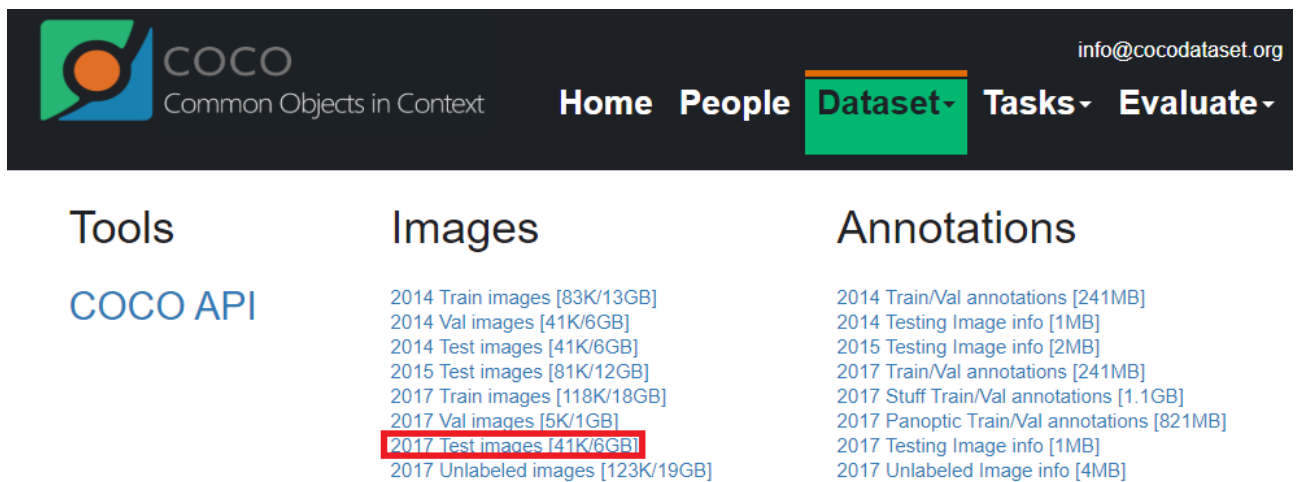


Рис. 2.1. Вибір бази даних з «негативними» зображеннями

Щоб створити bg.txt було розроблено програму для перейменування всіх файлів та сам запис у файл (додаток А).

б) Створення зразків і каскадів, використовуючи «позитивні» зображення на 1 «негативне»

Важливі параметри командного рядка для створення зразків за допомогою `opencv_createsamples` описується в джерелі [10].

При створенні зразків були проблеми з тим, що об'єкти на більшості зображеннях є завеликими, хоч при виконанні вам вивід помилки не допоможе, оскільки його немає.

Спершу досліджувались каскади з використанням лише позитивних зображень.

Також при дослідженнях з використанням лише «позитивних» зображень все-таки обов'язковими були «негативні», оскільки без них не вдасться створити каскаду. З'явиться помилка:

```
D:\MethodRecogniseRoadSigns\GTSRB\Final_Training>opencv_traincascade -data HaarCascade -vec samples(5000).vec -numPos 5000
-numStages 150 -w 15 -h 15 -mode ALL -precalcValBufSize 300 -precalcIdxBufSize 300
OpenCV: terminate handler is called! The last OpenCV error is:
OpenCV(3.4.11) Error: Bad argument (_cascadeDirName or _bgfileName or _vecFileName is NULL) in CvCascadeClassifier::train,
file C:\build\3_4_winpack-build-win64-vc15\opencv\apps\traincascade\cascadeclassifier.cpp, line 145
```

Рис. 2.2. Неправильне використання утиліти opencv\_traincascade

Для дослідження методу Віюлі-Джонса було створено різні .vec-файли з різною кількістю зразків (за замовчуванням створюються 1000).

Спершу було створено 5000 зразків з шириною та висотою 25 пікселів.

```
C:\Users\Бордан\Desktop\MethodRecogniseRoadSigns\GTSRB\Final_Training>opencv_createsamples -info Images\info.dat -vec
samples(5000).vec -bg negative.txt -num 5000 -w 25 -h 25
Info file name: Images\info.dat
Img file name: (NULL)
Vec file name: samples(5000).vec
BG file name: negative.txt
Num: 5000
BG color: 0
BG threshold: 80
Invert: FALSE
Max intensity deviation: 40
Max x angle: 1.1
Max y angle: 1.1
Max z angle: 0.5
Show samples: FALSE
Width: 25
Height: 25
Create training samples from images collection...
Done. Created 5000 samples
```

Рис. 2.3. Створення зразків

Схоже було утворено 7000, 10000 та 15000 зразків з шириною та висотою 25. З параметрами 35x35 та 15x15 було створено окремо 5000 і 7000 зразків. В результаті чого були отримані такі .vec файли лише з позитивними зразками:



GTSRB > Final_Training > Samples > -w 35 -h 35				Пошук: -w 35 -h 35
Ім'я	Дата змінення	Тип	Розмір	
 samples(5000).vec	14.05.2020 17:51	Файл VEC	11 968 КБ	
 samples(7000).vec	16.05.2020 9:10	Файл VEC	16 755 КБ	

Рис. 2.4. .vec-файли з позитивними зображеннями з висотою й шириною 35

px





GTSRB > Final_Training > Samples > -w 25 -h 25				Пошук: -w 25 -h 25
Ім'я	Дата змінення	Тип	Розмір	
 samples(5000).vec	08.05.2020 23:28	Файл VEC	6 109 КБ	
 samples(7000).vec	09.05.2020 15:04	Файл VEC	8 552 КБ	
 samples(10000).vec	10.05.2020 0:12	Файл VEC	12 217 КБ	
 samples(15000).vec	10.05.2020 13:38	Файл VEC	18 326 КБ	

Рис. 2.5. .vec-файли з позитивними зображеннями з висотою й шириною 25

px



GTSRB > Final_Training > Samples > -w 15 -h 15				Пошук: -w 15 -h 15
Ім'я	Дата змінення	Тип	Розмір	
 samples(5000).vec	19.05.2020 23:07	Файл VEC	2 203 КБ	
 samples(7000).vec	19.05.2020 23:21	Файл VEC	3 084 КБ	

Рис. 2.6. .vec-файли з позитивними зображеннями з висотою й шириною 15

px

Аргументи командного рядка програми opencv\_traincascade, згруповані за цілями, які описані в [10] і [13].

Каскади були створені для всіх вище перелічених зразків. Приклад створення каскаду зі 7000 зразків:

```
C:\Users\Бордан\Desktop\MethodRecogniseRoadSigns\GTSRB\Final_Training>opencv_traincascade -data HaarCascade -vec samples(7000).vec -bg negative.txt -numPos 7000 -numNeg 1 -numStages 100 -w 25 -h 25 -mode ALL
PARAMETERS:
cascadeDirName: HaarCascade
vecFileName: samples(7000).vec
bgFileName: negative.txt
numPos: 7000
numNeg: 1
numStages: 100
precalcValBufSize[Mb] : 1024
precalcIdxBufSize[Mb] : 1024
acceptanceRatioBreakValue : -1
stageType: BOOST
featureType: HAAR
sampleWidth: 25
sampleHeight: 25
boostType: GAB
minHitRate: 0.995
maxFalseAlarmRate: 0.5
weightTrimRate: 0.95
maxDepth: 1
maxWeakCount: 100
mode: ALL
```

Рис. 2.7. Створення каскаду

Коли було створено 5000 та 7000 зразків, стало зрозуміло, що потрібно вводити параметри `-precalcValBufSize <precalculated_vals_buffer_size_in_Mb>` і `-precalcIdxBufSize <precalculated_idx_buffer_size_in_Mb>`, оскільки розміри файлів ідентичні, а кількість зразків суттєво відрізняється. Для перевірки цього було створено ще 10000 зразків (25x25) без задання цих параметрів та 15000 з ними. Зразки 15x15 та 35x35 на постійній основі створювались з цими параметрами.









Ім'я	Дата змінення	Тип	Розмір
 cascade(-w 15 -h 15, 5000pos, 1neg)	20.05.2020 9:58	XML Document	47 КБ
 cascade(-w 15 -h 15, 7000pos, 1neg)	20.05.2020 8:42	XML Document	92 КБ
 cascade(-w 25 -h 25, 5000pos, 1neg)	09.05.2020 14:17	XML Document	47 КБ
 cascade(-w 25 -h 25, 7000pos, 1neg)	10.05.2020 0:08	XML Document	48 КБ
 cascade(-w 25 -h 25, 10000pos, 1neg)	10.05.2020 13:42	XML Document	47 КБ
 cascade(-w 25 -h 25, 15000pos, 1neg)	14.05.2020 23:12	XML Document	139 КБ
 cascade(-w 35 -h 35, 5000pos, 1neg)	15.05.2020 23:31	XML Document	47 КБ
 cascade(-w 35 -h 35, 7000pos, 1neg)	19.05.2020 14:13	XML Document	94 КБ

Рис. 2.8. Файли .xml з 1 негативним зображенням

```
C:\Users\Бордан\Desktop\MethodRecogniseRoadSigns\GTSRB\Final_Training>opencv_traincascade -data HaarCascade -vec samples(15000).vec -bg negative.txt -numPos 15000 -numNeg 1 -numStages 300 -w 25 -h 25 -mode ALL -precalcValBufSize 300 -precalcIdxBufSize 300
PARAMETERS:
cascadeDirName: HaarCascade
vecFileName: samples(15000).vec
bgFileName: negative.txt
numPos: 15000
numNeg: 1
numStages: 300
precalcValBufSize[Mb] : 300
precalcIdxBufSize[Mb] : 300
acceptanceRatioBreakValue : -1
stageType: BOOST
featureType: HAAR
sampleWidth: 25
sampleHeight: 25
boostType: GAB
minHitRate: 0.995
maxFalseAlarmRate: 0.5
weightTrimRate: 0.95
maxDepth: 1
maxWeakCount: 100
mode: ALL
```

Рис. 2.9. Правильне створення каскаду

Якщо ПК піде у режим сну, після повернення до звичайного режиму, можна продовжити створення каскаду натиском на кнопку Enter.

Якщо каскад ще не створився, а командний рядок був вимкнений із роботи, то можна задати повністю ідентичний рядок коду для запуску створення каскаду. Буде записано, наприклад,

**Stages 0-257 are loaded**

Рис. 2.10. Відновлення роботи opencv\_traincascade

і opencv\_traincascade відновить свою роботу.

Після того, як програма opencv\_traincascade закінчить свою роботу, підготовлений каскад буде збережений у cascade.xml файлі у –data папці. Інші файли в цій папці створені для випадку перерваного навчання, тому можна видалити їх після закінчення навчання.

Для кожного класифікатора, вираженого як сутність з підлеглими атрибутами було створене окреме XML-сховище. DOM (Document Object Model) тобто об'єктна модель документа такого класифікатора в загальному вигляді представлена нижче. Кореневим елементом є Сховище, довжина самого дерева дорівнює 8.

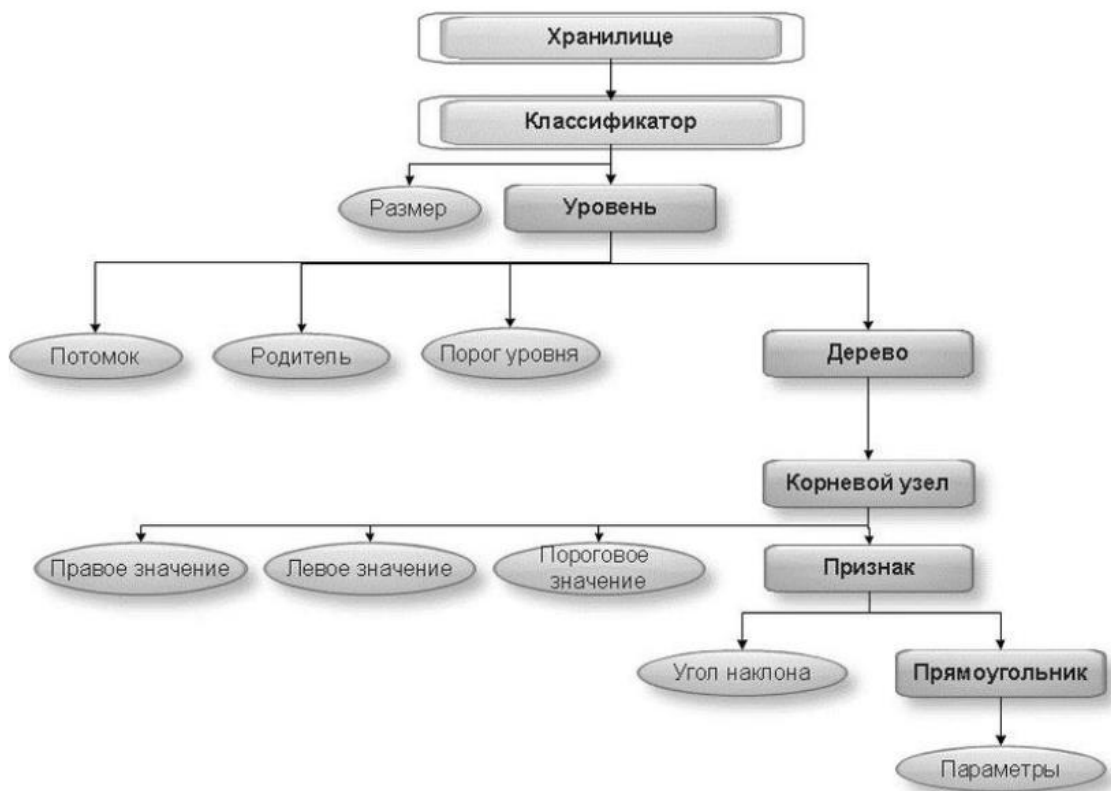


Рис. 2.11. Об'єктна модель класифікатора

На даному прикладі видно, яка інформація зберігається у використуваних XML. Це інформація про класифікатор (opencv-cascade-classifier) і його розмір (height, width), про використані рівні (stageParams), попередника, або батька даного рівня (weakClassifiers), наступного рівня або послідовника (internalNodes). Потім вибудовується дерево (stages) і його кореневи вузли (internalNodes). Далі описується про самі ознаки (features) в вузлах цього дерева, які задаються прямокутниками з параметрами (rects) з певним кутом нахилу (останнє число) (лістинг 2.1) [14]. Зібрані набори зображень переміщено в папку Images, а в папку HaarCascade погруповані різні каскади з різними параметрами висоти та ширини, в папку Samples поміщено папки з назвами, які відповідають розмірам зразків, де містяться відповідні .вес-файли.

Лістинг 2.1. Структура .xml-файлів [14].

```
<?xml version="1.0"?>

<opencv_storage>
<cascade type_id="opencv-cascade-classifier"><stageType>BOOST</stageType>
  <featureType>HAAR</featureType>
  <height>24</height>
  <width>24</width>
  <stageParams>
    <boostType>GAB</boostType>
    <minHitRate>9.9500000476837158e-01</minHitRate>
    <maxFalseAlarm>5.0000000000000000e-01</maxFalseAlarm>
    <weightTrimRate>9.4999999999999996e-01</weightTrimRate>
    <maxDepth>1</maxDepth>
    <maxWeakCount>100</maxWeakCount></stageParams>
  <featureParams>
    <maxCatCount>0</maxCatCount>
    <featSize>1</featSize>
    <mode>ALL</mode></featureParams>
  <stageNum>100</stageNum>
</stages>
  <_>
    <maxWeakCount>9</maxWeakCount>
    <stageThreshold>-5.0425500869750977e+00</stageThreshold>
    <weakClassifiers>
      <_>
        <!-- Ознаки першого класифікатора -->
        <internalNodes>
```



```

0 -1 0 -3.1511999666690826e-02</internalNodes>
<leafValues>
2.0875380039215088e+00 -2.2172100543975830e+00</leafValues></_>
</weakClassifiers></_>
<!-- решта поверхів-->
</stages>
<features>
<_>
<rects>
<_>
6 4 12 9 -1.</_>
<_>
6 7 12 3 3.</_></rects></_>
<tilted>0</tilted></_>
<!--Решта прямокутників-->
</features></cascade>
</opencv_storage>

```

Для різних версій OpenCV ця структура може відрізнятись.

с) Створення каскаду, використовуючи 5000 позитивних та 3000 негативних зображень

Тут потрібно створювати .vec-файл, в якому кількість «позитивних» і «негативних» зображень значно більша від кількості відповідних зображень при створенні каскаду, інакше отримаємо помилку на певному етапі (stage), так як метод намагається на кожному кроці тренування вибрати випадковим чином numPos та numNeg зображень з навчальних наборів і якщо зображень мало, він не може сформувати відповідні набори для кроків навчання.

```

Командний рядок
D:\MethodRecogniseRoadSigns\GTSRB\Final_Training>opencv_traincascade -data HaarCascadeNegative -vec samples(5000,2000).vec -bg negative(2000).vec -numPos 5000 -numNeg 2000 -numStages 300 -w 15 -h 15 -mode ALL -precalcValBufSize 500 -precalcIdxBufSize 500
PARAMETERS:
cascadeDirName: HaarCascadeNegative
vecFileName: samples(5000,2000).vec
bgFileName: negative(2000).vec
numPos: 5000
numNeg: 2000
numStages: 300
precalcValBufSize[Mb]: 500
precalcIdxBufSize[Mb]: 500
acceptanceRatioBreakValue: -1
stageType: BOOST
featureType: HAAR
sampleWidth: 15
sampleHeight: 15
boostType: GAB
minHitRate: 0.995
maxFalseAlarmRate: 0.5
weightTrimRate: 0.95
maxDepth: 1
maxWeakCount: 100
mode: ALL

===== TRAINING 0-stage =====
BEGIN
POS count : consumed 5000 : 5000
NEG count : acceptanceRatio 2000 : 1
Precalculation time: 14.219

+-----+-----+
| N | HR | FA |
+-----+-----+
| 1 | 1 | 1 |
+-----+-----+
| 2 | 1 | 1 |
+-----+-----+
| 3 | 0.9958 | 0.453 |
+-----+-----+
END>
Training until now has taken 0 days 0 hours 0 minutes 49 seconds.

===== TRAINING 1-stage =====
BEGIN
OpenCV Error: Bad argument (Can not get new positive sample. The most possible reason is insufficient count of samples in given vec-file.) in CvCascadeImageReader::PosReader::get, file C:\buildslave64\win64_amd64\master_PackSlave-win64-vc14-shared\opencv\apps\traincascade\imagestorage.cpp, line 157

```

Рис. 2.12. Помилка створення каскаду через неправильну кількість зображень



Також може бути виконане досрочне завершення створення каскаду, тобто буде створено менша етапів. Це стається, якщо похибка менша порогу. В командному рядку буде наступне повідомлення: «Train dataset for temp stage can not be filled. Branch training terminated».

Також при створенні каскаду потрібно виділити, як можна більше системної пам'яті для швидшого обчислення з негативними зображеннями.

32бітний Windows може максимально працювати з 4 ГБ оперативної пам'яті, а максимальна кількість пам'яті на один процес, який він дає змогу виділити ~2 ГБ, якщо процес вимагатиме більше пам'яті, то операційна система просто його переб'є [13].

Для 64бітних систем практика показує, що операційній системі зазвичай краще лишати 1-2 ГБ оперативки, щоб вона просто не зависла і режим багатозадачності працював.

Оскільки для навчання застосовуються два буфери, тому загальне використання пам'яті є сумою двох значень.

Час виконання поверхів при переданні різної кількості пам'яті буферам:

- 1) 300 і 300 МБ – 1 день 3 години
- 2) 1 ГБ і 1 ГБ – 2 дні
- 3) 2 ГБ і 2ГБ – 3 дні

Варто брати до уваги, що ці значення задавались для одного й того ж каскаду на різних поверхах і чим більший поверх, тим довше він створюється, тобто ніяким змін на швидкість виконання каскаду виділення пам'яті не впливає. Це все залежить від індивідуальних характеристик комп'ютера. Це також було помітно при створенні каскаду, використовуючи 10000 «позитивних» зображень та 1 «негативне».

### **2.3. Підготовка нових наборів даних**

Для того, щоб тренувати класифікатор, потрібні були великі набори зображень «позитивних», і великі набори зображень «негативних», грубо кажучи, шуканий об'єкт і його фон. При склеюванні виходить новий набір зображень [14]. Під склеюванням мається на увазі, що вирізають шуканий об'єкт і вставляють його у будь-яке місце на зображенні, де немає об'єктів з даної сфери.

В результаті навчання отримав багато класифікаторів, що в теорії мало б визначати будь-який дорожній знак, якщо він не буде пошкодженим. Для цього використовується AdaBoost, який вбудований в бібліотеку EmguCV.

## Джерела

1. Shin Shi «Emgu CV Essentials» year 2013, Packt Publishing Ltd.
2. Стаття про підготовку проекту під OpenCV: <https://habr.com/ru/post/135244/>.
3. Сайт для скачування OpenCV:  
<https://github.com/opencv/opencv/releases/tag/3.4.11>
4. Rapid Object Detection using a Boosted Cascade of Simple. Date of Conference: 8 14 Dec. 2001 Date Added to IEEE Xplore: 15 April 2003 Print ISBN: 0-7695-1272-0 Print ISSN: 1063-6919 INSPEC Accession Number: 7176899 DOI: 10.1109/CVPR.2001.990517 Publisher: IEEE Conference Location: Kauai, HI, USA, USA: <https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf>
5. Стаття про теорію методу Віоли-Джонса для розпізнавання облич:  
<https://habr.com/ru/post/133826/>.
6. Відео з тлумаченням про метод Віоли-Джонса:  
<https://www.youtube.com/watch?v=oRdFLHZsgFM&t=1290s>
7. Відомості про базу даних зі зображеннями, які містять лише 1 дорожній знак:  
<http://benchmark.ini.rub.de/?section=gtsrb&subsection=news>
8. Відомості про базу даних зі зображеннями, які містять дорожні знаки та фон:  
<http://benchmark.ini.rub.de/?section=gtsdb&subsection=news>.
9. Сайт для скачування бази даних зі зображеннями, які містять по одному дорожньому знаку:  
<https://sid.erda.dk/public/archives/daaeac0d7ce1152aea9b61d9f1e19370/published-archive.html>
10. Офіційний сайт OpenCV:  
[https://docs.opencv.org/master/dc/d88/tutorial\\_traincascade.html](https://docs.opencv.org/master/dc/d88/tutorial_traincascade.html)
11. Сайт для скачування бази даних зі зображеннями, які містять дорожні знаки та фон:  
<https://sid.erda.dk/public/archives/ff17dc924eba88d5d01a807357d6614c/published-archive.html>
12. Сайт для скачування «негативних» зображень:  
<https://cocodataset.org/#download>
13. Опис створення каскаду: <http://note.sonots.com/SciSoftware/haartraining.html>
14. Стаття про метод Віоли-Джонса для розпізнавання емоцій:  
<https://habr.com/ru/post/134857/>

## Додатки

### Додаток А

```
using System;
using System.IO;

namespace RenameImagesANDCreateNegImagesFile
{
    class Program
    {
        static void Main(string[] args)
        {
            while(true)
            {
                Int32 i = 0;
                Console.WriteLine("Input action:\n" +
                    "1 - rename negative images\n" +
                    "2 - create negative images file");
                String st = Console.ReadLine();
                String nameFile, path;
                Byte.TryParse(st, out Byte j);
                switch (j)
                {
                    case 1:
                        {
                            try
                            {
                                Console.Write("Input absolute path to directory with
negative images: ");

                                path = Console.ReadLine();
                                DirectoryInfo images = new DirectoryInfo(path);
                                Console.WriteLine("Results:");
                                foreach (FileInfo fileInfo in images.GetFiles())
                                {
                                    i++;
                                    nameFile = $"{path}\\{i}.jpg";

                                    File.Move(fileInfo.FullName, nameFile);
                                    //Console.WriteLine($"{fileInfo.Name} renamed to
{i}.jpg");
                                }
                                Console.WriteLine("Images successfully renamed");
                            }
                            catch(Exception ex)
                            {
                                Console.WriteLine($"Exception: {ex.Message}");
                            }
                            break;
                        }
                    case 2:
                        {
                            Console.Write("Input absolute path to directory with negative
images: ");

                            path = Console.ReadLine();
                            Console.Write("Input absolute path to and name of negative
images file with extension (it can't be create):");
                            nameFile = Console.ReadLine();
                            try
                            {
                                using (StreamWriter sw = File.CreateText(nameFile))
                                {
                                    DirectoryInfo images = new DirectoryInfo(path);
                                    Console.WriteLine("File context:");
                                    foreach (FileInfo fileInfo in images.EnumerateFiles())
                                    {
                                        sw.WriteLine($"{images.FullName}\\{fileInfo.Name}");
                                    }
                                }
                            }
                        }
                }
            }
        }
    }
}
```

```

Console.WriteLine($"{images.FullName}/{fileInfo.Name}");
    }
    Console.WriteLine("Text file successfully recorded");
}
catch (Exception ex)
{
    Console.WriteLine($"Exception: {ex.Message}");
}
break;
}
default:
{
    Console.WriteLine("Input 1 or 2");
    break;
}

}
Console.WriteLine("Press some key in order to continue\n");
Console.ReadKey(true);
}

}
}
}

```