ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN CUỐI KỲ MÔN MÁY HỌC

Đề tài: Phát hiện động vật trong hình ảnh nương rẫy

Giảng viên hướng dẫn:

TS. Lê Đình Duy

Ths. Phạm Nguyễn Trường An

Sinh viên:

Họ tên: Hoàng Tuấn Anh

MSSV: 18520446

ꯑꯣꯏꯕ Tp. Hồ Chí Minh, 15/07/2020 ꯑꯣꯏꯔ

Phần I: Thông tin tóm tắt

Tên đề tài (tiếng Việt)	**Phát hiện động vật trong hình ảnh** - **Link GitHub (https://github.com/tuananh11052000/CS114.K21)** - **Link GGDrive (https://drive.google.com/drive/folders/14xDOTR1_mozZ-f5chpXtWloPD6NbC_r9?usp=sharing)**
Họ và tên **Ảnh**	Hoàng Tuấn Anh
Số buổi vắng	0
Số buổi đi trễ (không điểm danh)	0
Số lần Comment trên Google Classroom	0
Tóm tắt Bài tập Quá trình	Cách sử dụng google colab, các bài tập về Linear 3regression, Logistic Regression, GradientDescent, Data collection, crop image
Tóm tắt Bài tập Cuối kỳ	Sử dụng thuật toán rút trích đặc trưng Histogram of Oriented Gradients (HOG) và mô hình máy học Support Vector Machine (SVM), tiến hành phát hiện động vật (voi) kích thước ảnh 140x110 xuất hiện trong bức ảnh đầu vào.

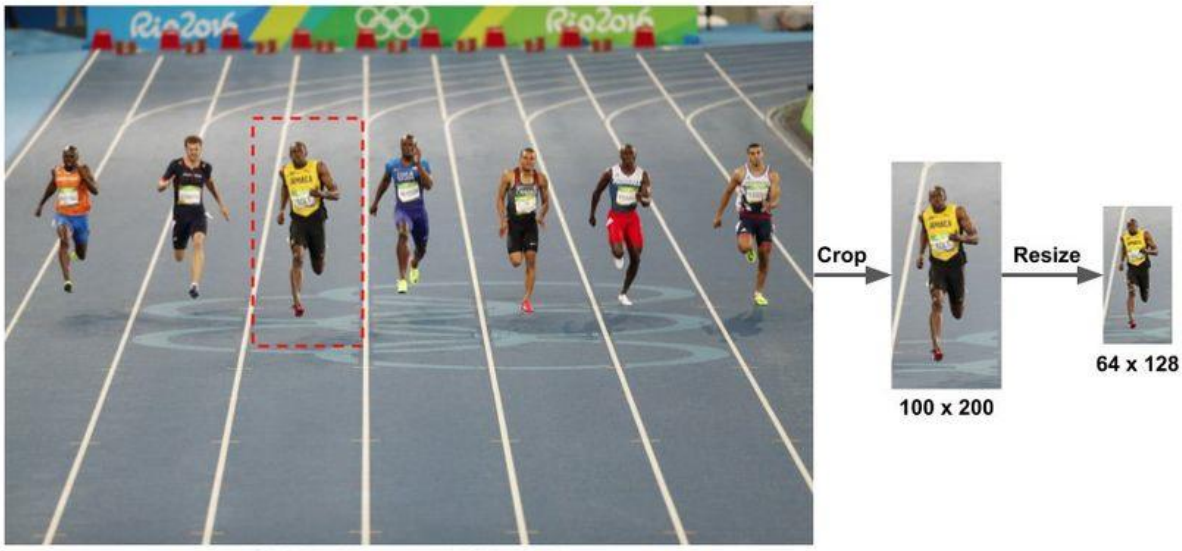
Phần 2: Báo cáo chi tiết đồ án cuối kỳ:

I. Các kiến thức liên quan:

1. Histogram of Oriented Gradients

1.1 Tiền xử lý

Trong bài toán này, để thuận tiện cho việc chia đều hình ảnh thành các khối, ô và tính toán đặc trưng ở các bước tiếp theo, chúng ta cần resize kích thước tất cả các hình ảnh trong tập dữ liệu về một kích thước chung.



Trong các ví dụ được trình bày trong bài viết này, kích thước chung cho 1 hình ảnh sẽ mặc định là 64x128. Trong đồ án, kích thước này sẽ là 140x110.

1.2 Tính Gradient

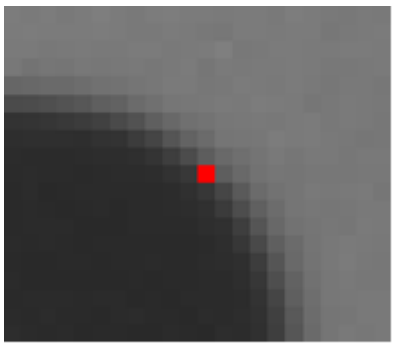
Đây là bước đầu tiên, được thực hiện bằng hai phép nhân chập ảnh gốc với 2 chiều, tương ứng với các toán tử lấy đạo hàm theo hai hướng Ox và Oy. Trong đó, 2 hướng tương ứng đó là:

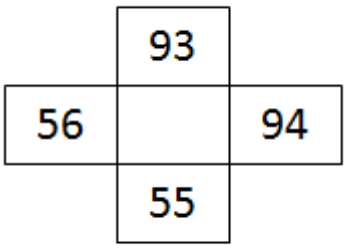
$$D_x = [-1\ 0\ 1] \text{ và } D_y = [1\ 0\ -1]^T,$$

T là phép toán chuyển vị ma trận.
Khi đó, bạn có thể tính được Gradient bao gồm hai thành phần là cường độ (Gradient Magnitude) và hướng (Gradient Derection) theo công thức sau:

$$\text{Cường độ: } |G| = \sqrt{I_x^2 + I_y^2}$$
$$\text{Hướng: } \theta = \frac{\arctan I_x}{I_y}$$

Ví dụ: Giả sử ta có một điểm ảnh như sau





Chúng ta sẽ áp dụng các công thức trên để tính được gradient của điểm ảnh này:

$$I_x = I * D_x = [56\ x\ 94] * \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} = [38]$$

$$I_y = I * D_y = \begin{bmatrix} 93 \\ y \\ 55 \end{bmatrix} * [1\ 0\ -1] = [38]$$

$$\text{Cường độ: } |G| = \sqrt{I_x^2 + I_y^2} = \sqrt{38^2 + 38^2} \approx 53{,}74$$

$$\text{Hướng: } \theta = \frac{\arctan I_x}{I_y} = \frac{\arctan(38)}{38} \approx 2.33$$

1.3 Tính vector đặc trưng cho từng ô (cells)

Để tính toán vector đặc trưng cho từng ô (cell), chúng ta cần chia hình ảnh thành các block, mỗi block lại chia đều thành các cell. Để xác định được số block, chúng ta sẽ sử dụng công thức sau:

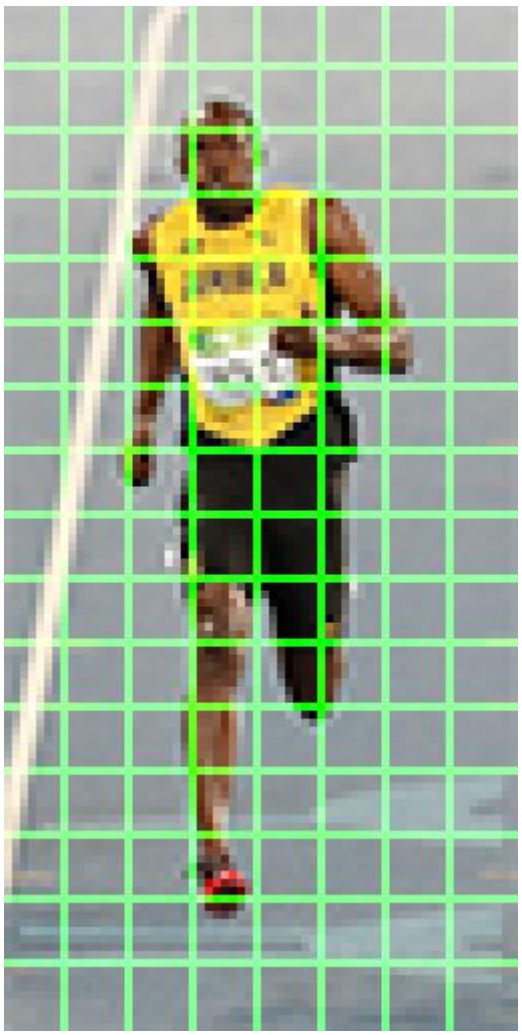
$$n_{block\_image} = \left(\frac{W_{image} - W_{block} * W_{cell}}{W_{cell}} + 1\right) * \left(\frac{H_{image} - H_{block} * H_{cell}}{H_{cell}} + 1\right)$$

trong đó:

W_{image}, W_{block}, W_{cell} : lần lượt là chiều rộng của ảnh, khối, ô

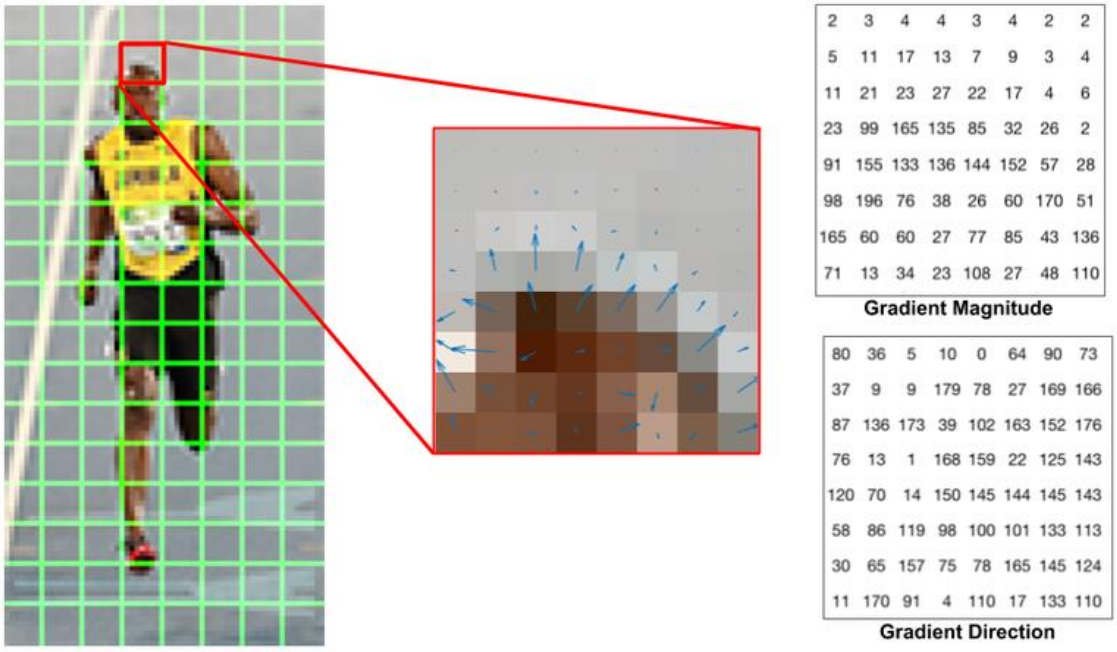
H_{image}, H_{block}, H_{cell} : lần lượt là chiều dài của ảnh, khối, ô

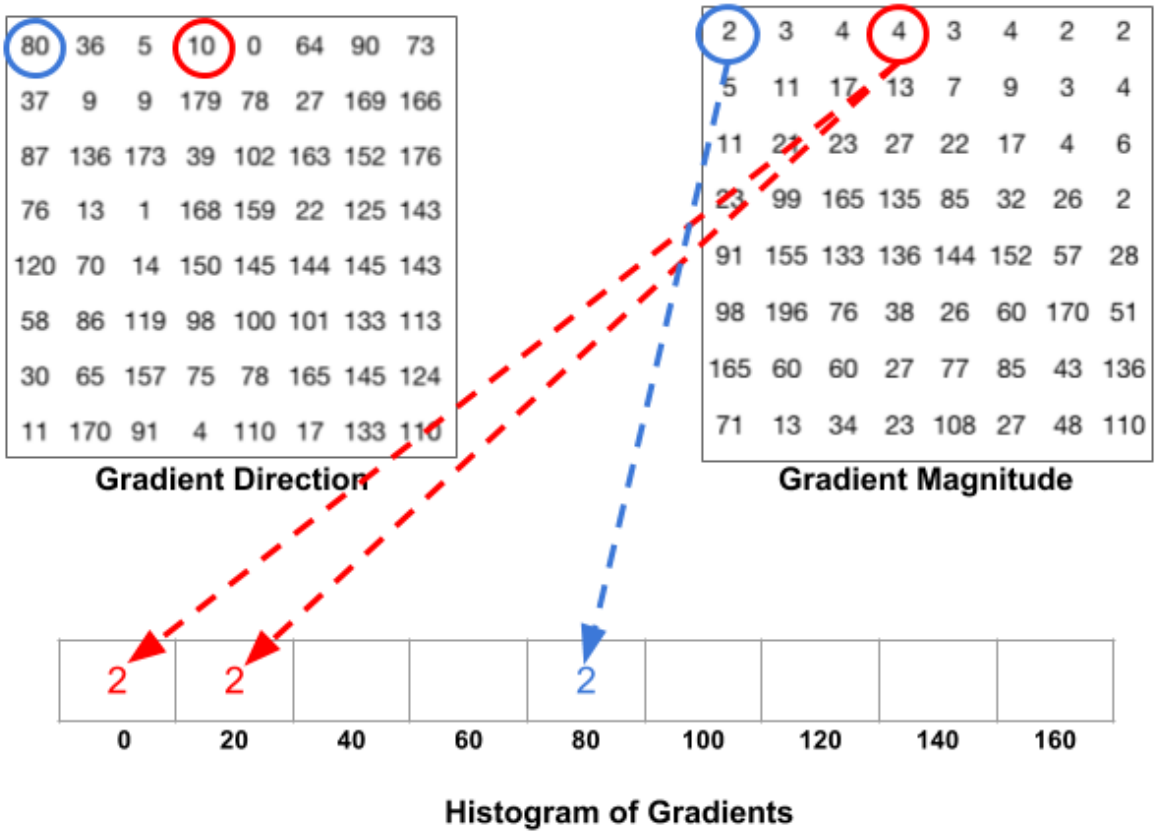
Ví dụ: Trong trường hợp này, hình ảnh của chúng ta có kích thước là 64x128, ta sẽ chia mỗi hình ảnh thành các block có kích thước 16x16. Mỗi block sẽ bao gồm 4 cell, mỗi cell có kích thước là 8x8.



Tiếp theo, tiến hành tính toán đặc trưng HOG tại mỗi cell *sử dụng không gian hướng 9 bin*, trường hợp "**unsigned-HOG**". Hướng gradient sẽ chạy trong khoảng 0 độ đến 180 độ, trung bình 20 độ mỗi bin.

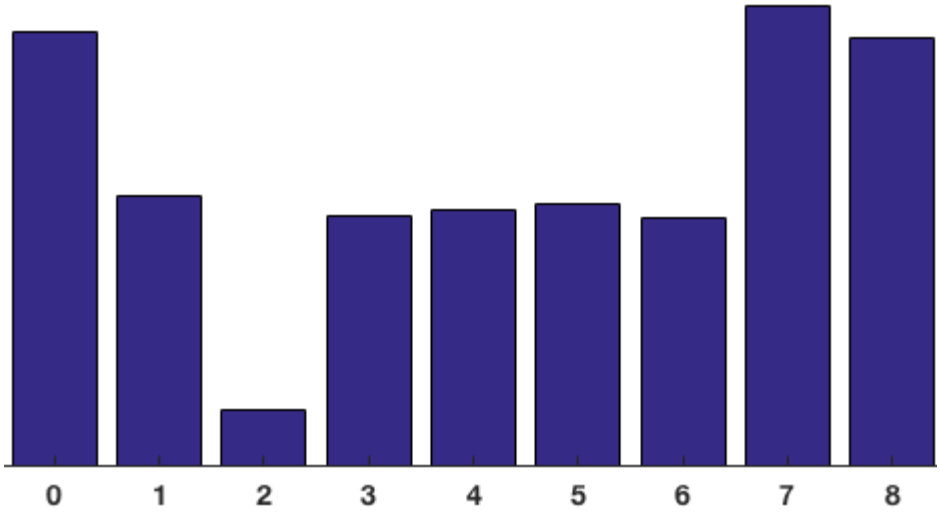
Tại mỗi cell, xây dựng một biểu đồ cường độ gradient bằng cách vote các pixel vào biểu đồ. Trọng số vote của mỗi pixel phụ thuộc hướng và cường độ gradient (được tính toán từ bước 2) của pixel đó. Ví dụ:





Như trong hình ảnh trên, đầu tiên là pixel có bao quanh màu xanh lam. Nó có hướng 80 độ và cường độ là 2, vì vậy ta thêm 2 vào bin thứ 5 (hướng 80 độ). Tiếp theo là pixel có bao quanh màu đỏ. Nó có hướng 10 độ và cường độ 4. Vì không có bin 10 độ, nên ta vote cho bin 0 độ và 20 độ, mỗi bin thêm 2 đơn vị.

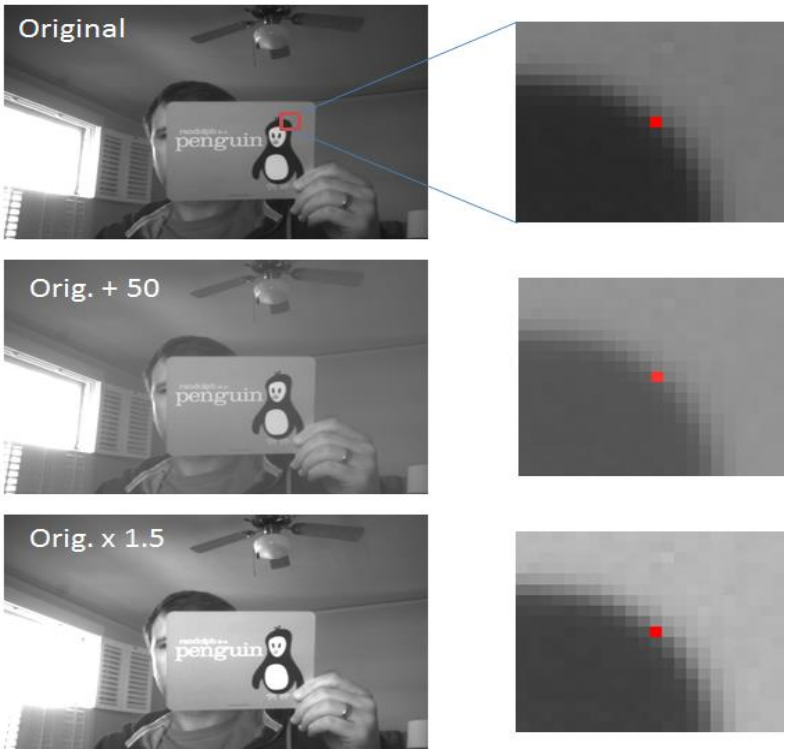
Sau khi vote hết các pixel trong một cell kích thước 8x8 vào 9 bin, ta có thể thu được kết quả như sau:



1.4 Chuẩn hóa khối (blocks)

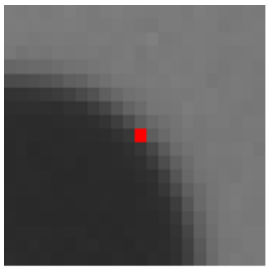
Để tăng cường hiệu năng nhận dạng, các histogram cục bộ sẽ được chuẩn hóa về độ tương phản bằng cách tính một ngưỡng cường độ trong một khối và sử dụng giá trị đó để chuẩn hóa tất cả các ô trong khối. Kết quả sau bước chuẩn hóa sẽ là một vector đặc trưng có tính bất biến cao hơn đối với các thay đổi về điều kiện ánh sáng.

Đầu tiên, hãy xem xét ảnh hưởng của việc chuẩn hóa tới các vector gradient trong ví dụ sau:



Trong hình ảnh trên, trường hợp đầu tiên là một ô của hình ảnh ban đầu. Trường hợp thứ hai, tất cả các giá trị pixel đã được tăng lên 50. Trong trường hợp thứ ba, tất cả các giá trị pixel được nhân với 1.5. Dễ dàng thấy được, trường hợp thứ ba hiển thị độ tương phản gia tăng. Ảnh hưởng của phép nhân là làm các điểm ảnh sáng trở nên sáng hơn nhiều, trong khi các điểm ảnh tối chỉ sáng hơn một chút, do đó làm tăng độ tương phản giữa các phần sáng và tối của hình ảnh.

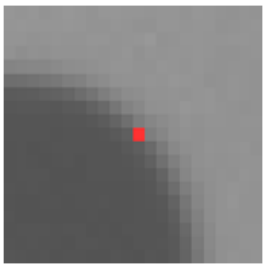
Hãy nhìn vào các giá trị pixel thực tế và sự thay đổi của vector gradient của ba trường hợp trên trong hình ảnh sau:



	93	
56		94
	55	

$$\nabla f = \begin{bmatrix} 38 \\ 38 \end{bmatrix}$$

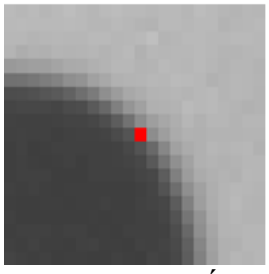
$$|\nabla f| = \sqrt{(38)^2+(38)^2} = 53.74$$



	143	
106		144
	105	

$$\nabla f = \begin{bmatrix} 38 \\ 38 \end{bmatrix}$$

$$|\nabla f| = \sqrt{(38)^2+(38)^2} = 53.74$$



	140	
84		141
	83	

$$\nabla f = \begin{bmatrix} 57 \\ 57 \end{bmatrix}$$

$$|\nabla f| = \sqrt{(57)^2+(57)^2} = 80.61$$

- Các con số trong các ô là giá trị pixel của các điểm ảnh lân cận điểm ảnh được đánh dấu màu đỏ.
- Delta F là đạo hàm theo riêng hai hướng của điểm ảnh ([Ix, Iy])
- | Delta F| là giá trị cường độ điểm ảnh (Gradient Magnitude), tính theo công thức (*)

Trong trường hợp một và hai, giá trị cường độ vector gradient của chúng tương đương nhau, nhưng trong trường hợp thứ ba, cường độ vector gradient đã tăng lên 1.5 lần. Nếu chia ba vector bằng độ lớn tương ứng, ta sẽ nhận được các kết quả tương đương cho cả ba trưởng hợp. Vì vậy, trong ví dụ trên, chúng ta thấy rằng bằng cách chia các vector gradient theo độ lớn của chúng, chúng ta có thể biến chúng thành bất biến để thay đổi độ trương phản.

Có nhiều phương pháp có thể được dùng để chuẩn hóa khối. Gọi v là vector cần chuẩn hóa chứa tất cả các histogram của một khối. ‖v(k)‖ là giá trị chuẩn hóa của v theo các chuẩn k=1, 3 và e là một hằng số nhỏ. Khi đó, các giá trị chuẩn hóa có thể tính bằng một trong những công thức sau:

$$\text{L2-norm: } f = \frac{\boldsymbol{v}}{\sqrt{\|\boldsymbol{v}_2\|^2+e^2}}$$

$$\text{L1-norm: } f = \frac{\boldsymbol{v}}{\|\boldsymbol{v}_1\|+e}$$

$$\text{L1-sqrt: } f = \sqrt{\frac{\boldsymbol{v}}{\|\boldsymbol{v}_1\|^2+e}}$$

Ghép các vector đặc trưng khối sẽ thu được vector đặc trưng R-HOG cho ảnh. Số chiều vector đặc trưng ảnh tính theo công thức :

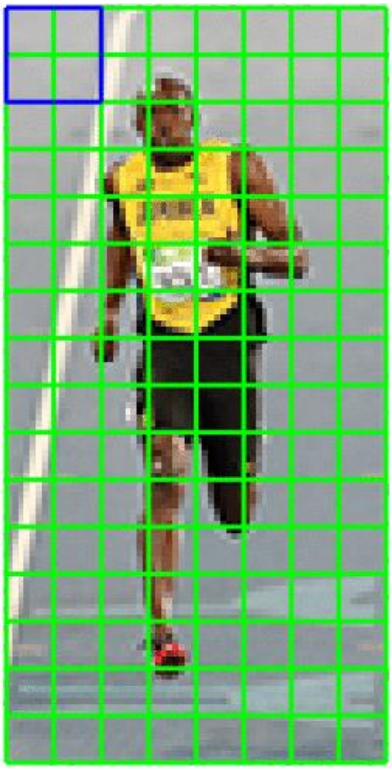
$$size_{image} = n * size_{block}$$

trong đó :

n là số khối của hình ảnh.

$size_{block}$ là số chiều của vector đặc trưng khối

1.5 Tính toán vector đặc trưng HOG



- Với mỗi hình ảnh kích thước 64x128, chia thành các block 16x16 chồng nhau, sẽ có 7 block ngang và 15 block dọc, nên sẽ có 7x15 = 105 blocks.
- Mỗi block gồm 4 cell. Khi áp dụng biểu đồ 9-bin cho mỗi cell, mỗi block sẽ được đại diện bởi một vector có kích thước 36x1.
- Vì vậy, khi nối tất cả các vector trong một block lại với nhau, ta sẽ thu được vector đặc trưng HOG của ảnh có kích thước 105x36x1 = 3780x1.

2. Support Vector Machine:

- Là thuật toán thường được áp dụng trong bài toán phân lớp, áp dụng cho cả dữ liệu tuyến tính và không tuyến tính.
- Khi biểu diễn các điểm dữ liệu lên không gian nhiều chiều, có vô số các mặt phẳng có thể chia các điểm dữ liệu theo phân lớp của chúng, nhưng vấn đeef đặt ra đâu mới là mặt phẳng phân chia tốt nhất. Support Vector Machine Learning có thể hiểu đơn giản là một thuật toán tìm ra một hay nhiều siêu mặt phẳng phù hợp nhất để chia các điểm dữ liệu theo phân lớp của chúng một cách phù hợp nhất.
- Khoảng cách từ một điểm (vector) có toạ độ x_0 tới *siêu mặt phẳng* (*hyperplane*) có phương trình: $\mathbf{w}^T\mathbf{x} + b = 0$ được xác định bởi:

$$\frac{|\mathbf{w}^T\mathbf{x}_0 + b|}{||\mathbf{w}||_2}$$

Trong đó $||\mathbf{w}||_2 = \sqrt{\sum_{i=1}^{d} w_i^2}$ với d là số chiều của không gian.

- Với mặt phần chia như trên, *margin* được tính là khoảng cách gần nhất từ 1 điểm tới mặt đó (bất kể điểm nào trong hai classes):

$$\text{margin} = \min_n \frac{y_n(\mathbf{w}^T\mathbf{x}_n + b)}{||\mathbf{w}||_2}$$

- Bài toán tối ưu trong SVM chính là bài toán tìm w và b sao cho margin này đạt giá trị lớn nhất.

II. Chi tiết đề tài:

1. Thu thập dữ liệu:
   - Mô tả về bộ dữ liệu:
     - ✓ Dữ liệu bao gồm 2200 ảnh với hai nhãn.
     - ✓ Ảnh có kích thước 140x110 px.
     - ✓ Bộ dữ liệu là tự thu thập.
     - ✓ Đối với ảnh nhãn 0, lựa chọn, thu thập những ảnh với góc chụp chú voi ở phía bên hông. Ảnh voi nằm chính giữa ảnh. Mỗi ảnh chỉ chữa một chú voi duy nhất và nó không bị che khuất với các vật thể khác.
     - ✓ Đối với nhãn 1 thu thập được tổng cộng 1927 ảnh, là những ảnh không xuất hiện voi, toàn bộ đều là ảnh nương rây, núi rừng với độ sáng tuỳ ý.
   - Cách thu thập dữu liệu:
     - ✓ Dữ liệu được thu thập theo hai nhãn:

       + Nhãn 0: Những hình ảnh có xuất hiện voi.

       + Nhãn 1: Những hình về ruộng đồng, nương rẫy không xuất hiện voi.

     - ✓ Đối với hình ảnh ở nhãn 1 thu thập bằng hai cách sau:
       - Thu thập bằng hình ảnh sẵn có trên google: Tìm kiếm bằng google hình ảnh những chú voi đảm bảo sau khi cắt xong vẫn thu được một hình ảnh với toàn bộ các bộ phận của chú voi và quan trọng hơn là tỉ lệ chiều dài và rộng của tấm ảnh phải bằng hoặc gần bằng với tỉ lệ 140x110 (Nhằm mục đích sau khi resize thì ảnh chú voi không bị biến dạng hoặc chỉ bị biến dạng chút ít.
       - Thu thập thông qua các video: Đấy là cách thu thập chính cho bài toán này (Chiếm 90% tổng số lượng ảnh trong nhãn 0). Thực hiền tìm kiếm những video thế giới động vật có xuất hiện nhiều voi, lựa chọn những đoạn video xuất hiện voi với góc chụp ngang thân voi, độ sáng tốt, voi không bị che khuất bởi các vật thể khác. Hình ảnh đảm bảo toàn bộ các bộ phận của voi nằm gọn trong bức ảnh, với khoảnh cách

với lề giao động trong khoảng từ 3% đến 5% chiều rộng của bức ảnh (đối với lề trái và lề phái) và 3% đến 5% chiều cao (đối với lề trên và lề dưới).

- ❖ Lưu ý với cả hai cách thu thập trên: Ảnh đảm bảo màu của voi không được khớp với màu nền, nếu bị khớp, việc trích suất ở bức tiếp theo sẽ rất khó khăn.

Sau khi có được bộ ảnh thoả yêu cầu, tiến hành tăng cường độ đa dạng của dữ liệu bằng cách đổi chiều bức ảnh(bằng cách lật bức ảnh lại). Bước này giúp nhân đôi số lượng ảnh có trong bộ dữ liệu.

- ✓ Thu thập dữ liệu với nhãn 1:
  - o Xuất phát từ mục đích của bài toán là phát hiện voi vào phá nương rẫy. Bộ dữ liệu cho nhãn 1 là những hình ảnh về nương rẫy (không xuất hiện voi). Từ đặc điểm đó của nhãn 1, em chọn cách thu thập như sau: Đầu tiên thực hiện thu thập những hình ảnh về nương rẫy với kích thước ảnh lớn, rõ nét với số lượng là 100 bức.
  - o Sau khi có được bộ các bức ảnh từ cách trên, resize toàn bộ ảnh về kích thước nhỏ hơn (với chiều rộng là 500px, chiều cao tính theo tí lệ nhằm không làm biến dạng cách thành phần đối tượng trong bức ảnh). Sau đó, xây dựng một tool viết bằng python, opencv, hỗ trợ cắt bức ảnh ra thành nhiều phần khác nhau với kích thước mỗi khung hình là 140x110 px. Bước nhảy theo trục x của mỗi khung hình được xét bằng 80px, theo trục y là 80px (Do việc thu thập được các bức hình lớn là tương đối dễ, cho nên ở đây ta có thể xét bước nhảy hơi lớn một chút để tạo ra sự khác biệt giữa các khung hình liền kề).

2. Xữ lý dữ liệu:
   - ✓ Sau khi có được bộ các bức ảnh từ các cách thu thập trên, tiến hành resize lại toàn bộ bộ dữ liệu, chuyển chúng về kích thước 140x110 px. Cùng với bước resize, ta tiến hành chuyển ảnh từ ảnh ban đầu là ảnh màu sang ảnh xám. Với mục đích làm giảm kích thước bộ dữ liệu, giảm thời gian tiêu hao khi train.
   - ✓ Tiến hành đọc ảnh. Sau bước này chúng ta nhận được một mảng hai chiều (140,110).
   - ✓ Cùng lúc với bước đọc ảnh, ta tiến hành gán nhãn cho bộ dữ liệu. Do khi lưu trữ bộ dataset trên google driver, các ảnh với nhãn tương ứng sẽ được lưu trong một folder riêng, ta có thể tận dụng điều này để lấy được nhãn của dữ liệu, bằng cách tách nhãn từ đường link trỏ tới phần tử dữ liệu khi tiến hành duyệt. Các nhãn tương ứng được lưu trong một mảng với index bằng với index của ảnh nạp vào mảng khi đọc.
   - ✓ Tiến hành rút trích đặc trưng bằng HOG với các tham số: mỗi cell có kích thước (2,2), mỗi block có kích thước 2x2 cells, lưu vào trong một

mảng. Ứng với mỗi bức ảnh, trải qua bước rút trích đặc trưng chúng ta thu được một vector đặc trưng một chiều với kích thước 42432 phần tử.

3. Chọn model :

- Với mỗi điểm dữ liệu của chúng ta sau khi thực hiện xong các bước ở trên, ứng với mỗi điểm dữ liệu, ta có thể biểu diễn chúng trên không gian n chiều (cụ thể ở đây là không gian 42432 chiều. Từ đây ta nhận thấy rằng, bộ dữ liệu hiện tại của chúng ta có số chiều rất lớn.
- Bài toán của chúng là dạng classification.
- Support vector machine learning là một kĩ thuật phân lớp khá phổ biến, với ưu điểm là có khả năng tính toán hiệu quả khi không gian có số lượng chiều của dữu liệu lớn. Ngoài ra khả năng áp dụng Kernel mới cho phép linh động giữa các phương pháp tuyến tính và phi tuyến tính từ đó khiến cho hiệu suất phân loại lớn hơn.
- Dựa vào những đặc điểm trên, chúng ta sẽ chọn SVM là model cho bài toán.

4. Train model:

- Sau khi có được bộ dữ liệu là một mảng các vector đặc trưng đã được gán nhãn và chọn được model, ta tiến hành train.
- Tham số C (độ lớn của tham số này quyết định mức độ fit của mô hình). Sử dụng một hàm được cung cấp bởi sklearn để xác định giá trị C phù hợp với model.

```
from sklearn.model_selection import GridSearchCV
# defining parameter range
param_grid = {'C': [0.01, 0.1, 1, 10, 100],
              'gamma': [1, 0.1, 0.01, 0.001, 0.0001],
              'kernel': ['linear']}
grid = GridSearchCV(SVC(), param_grid, refit = True, verbose = 3)
# fitting the model for grid search
grid.fit(x_train, y_train)
# print best parameter after tuning
print(grid.best_params_)
# print how our model looks after hyper-parameter tuning
print(grid.best_estimator_)
```

Kết quả:

```
{'C': 0.1, 'gamma': 1, 'kernel': 'linear'}
SVC(C=0.1, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=1, kernel='linear',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

- Các tham số truyền vào mô hình là C (độ lớn cảu tham số này quyết định mức độc fit của mô hình) được xét bằng 0.1, kernal được xét bằng ‘linear’, “gamma” được xét bằng 1.

```
model = SVC(kernel="linear", C=0.1, gamma=1, probability=False, random_state=42)
model.fit(x_train, y_train)
```

5. Đánh giá model:

- Đối với bộ dữ liệu test, chúng ta sử dụng hàm random để lấy ta 15% dữ liệu trong bộ dữ liệu ban đầu. Sau khi random ta có được 69 nhãn 0 và 296 nhãn 1. Do vậy, một model có thể sử dụng được nếu accurancy của nó phải lớn hơn 81.1 %.
- Trước tiên sử dụng cách tính score là tính số lượng nhãn dự đoán đúng trên tổng số nhãn. Tại bước này sử dụng một hàm của thư viện sklearn cung cấp để tính:

```
from sklearn.metrics import accuracy_score
pred = model.predict(x_test)
print('acc: ', accuracy_score(pred, y_test))
```

Ta thu được kết quả như sau:
Đối với model sử dụng SVM:

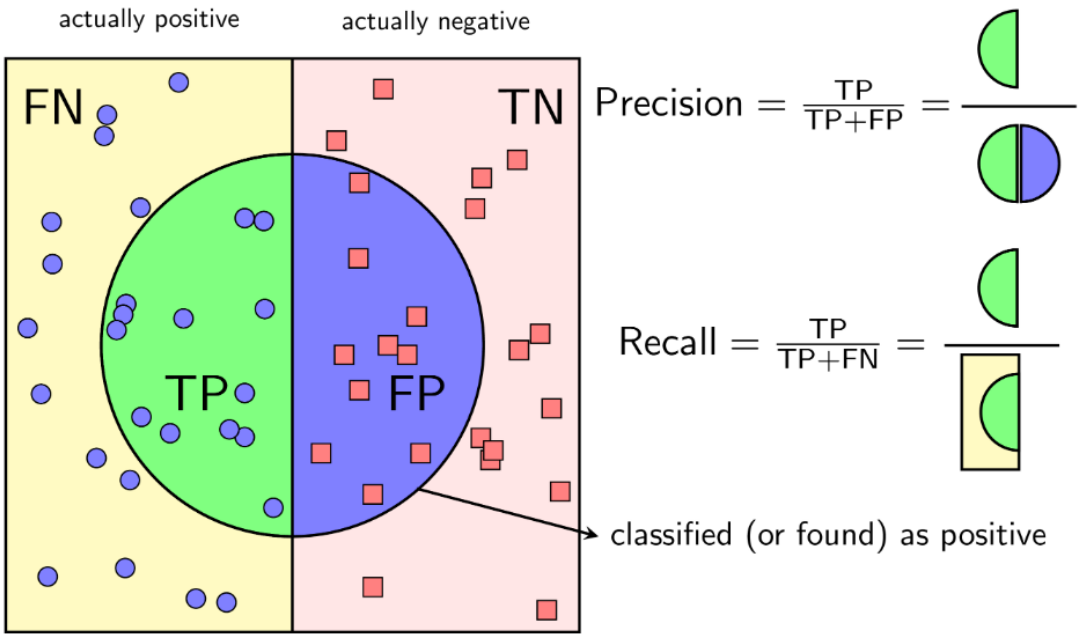
```
acc:  0.9972527472527473
```

Đối với model sử dụng KNN:

```
acc:  0.8186813186813187
```

Con số này tuy cao nhưng chưa thể đánh giá là model này có đánh giá tốt hay không.

- Đối với mục tiêu bài toán, chúng ta cần xác định xem đối với khả năng dự đoán nhãn 0 là tốt hay không, nếu sử dụng cách trên thì không thể biết được. Và nhằm so sánh tính thực tiễn của hai model khác nhau. Ta sử dụng một cách khác đó là tính thông qua precision và recall.



Đối với mỗi nhãn, ta có thể tính được hai tham số là precision và recall với công thức như trên. Precision phản ảnh tỉ lệ giữa số lần dự đoán đúng nhãn 0 trên cho tổng số nhãn 0. Còn recall phản ảnh tỉ lệ giữa dự đoán đúng nhãn 0 trong tất cả các nhãn được dự đoán là 0.
Sau khi có được hai tham số đó, ta tiến hành tính tham số F của model đối với nhãn 0 bằng công thức tham số F này phản ảnh đúng khả năng dự đoán một nhãn nào đó, ngay cả khi dự đoán toàn bộ các nhãn đều là nhãn đó:

$$F_1 = 2\frac{1}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}} = 2\frac{\text{precion} \cdot \textit{recall}}{\text{precision} + \text{recall}}$$

Kết quả trên model sử dụng SVM:

```
[ ] from sklearn.metrics import classification_report
    target_names = ['class 0', 'class 1']
    print(classification_report(y_test, pred, target_names = target_names))
```

```
              precision    recall  f1-score   support

     class 0       1.00      0.99      0.99        69
     class 1       1.00      1.00      1.00       295

    accuracy                           1.00       364
   macro avg       1.00      0.99      1.00       364
weighted avg       1.00      1.00      1.00       364
```

Nhận thấy rằng model dự đoán tốt trên cả hai nhãn với bộ test hiện tại.

6. Xây dựng một ứng dụng:

- Sau khi xây dựng được một mô hình dự đoán phục vụ mục đích của đề tài, ta chuyển đến bước prediction trong 7 bước xây dựng một mô hình máy học.
- Tại bước này chúng ta cho phép truyền vào một bức ảnh bất kỳ, kết quả đầu ra là một dự đoán là có hay không sự xuất hiện của voi trong bức ảnh.
- Với ý tưởng sử dụng một ô cửa trượt kích thước 140x110 px, tiến hành di chuyển khung cửa trượt trên toàn bộ ảnh, nhằm tìm ra vị trí có xuất hiện voi.
- Ứng với mỗi vị trí của khung cửa trượt, tiến hành đọc ảnh, rút trích đặc trưng và đưa vào mô hình để sự đoán.

III. Ưu nhược điểm của model

1. Ưu điểm:
   - ✓ Là phương pháp dễ tiếp cận và phát triển cho những người chưa có nhiều kiến thức về Máy học. Các hàm huấn luyện có sẵn trong thư viện, nhiều nguồn hướng dẫn vì HOG và SVM là những phương pháp, thuật toán khá phổ biến.
2. Nhược điểm:

Do đây là lần đầu tiếp cận với machine learning nên đồ án vướng phải khá nhiều nhược điểm:

- ➢ Quá trình trích suất đặc trưng rất tốn thời gian đặc biệt là đối với những hình ảnh có độ phức tạp lớn.
- ➢ Những hình dạng của con voi xuất hiện trong dataset rất đa dạng, làm cho việc dự đoán gặp rất nhiều sai sót.
- ➢ Hướng xây dựng ứng dụng chưa phù hợp, do mất thời gian khá lâu để có thể đưa ra được kết quả.
- ➢ Do hạn chế kiến thức, đồ án chỉ thu thập ảnh chụp bên hông chú voi, nhắm thuận lợi cho việc xây dựng model. Bởi vậy, với hình ảnh một chú voi khi chụp ở một góc độ khác thì không thể nhận ra được.

3. Hướng phát triển trong tương lai:

- ➢ Mô hình có khả năng phát hiện và khoanh vùng xuất hiện voi trong một bức ảnh lơn, do vậy trong tương lai em nghĩ nó có thể phát triển thành một ứng dụng thời gian thực đáp ứng được mong muốn của đề tài.
- ➢ Trong tương lai, nếu có thể, em sẽ thực hiện bổ sung thêm nhiều hình dáng khác của con voi(không chỉ riêng những hình ảnh chụp bên hông chú voi) nhằm phát hiện được sự xuất hiện của chúng khi chúng ở bất kỳ tư thế nào.

IV. Tổng kết:

Với vị trí là một sinh viên khoa khác, em cảm thấy rất vui và hào hứng khi tìm hiểu về machine learning.Trải qua thời gian học, em đã có cái nhìn tổng quan về lĩnh vực máy học. Biết được các bước cơ bản để xây dựng một ứng dụng máy học.Biết cách đặt câu hỏi cho một vấn đề mà mình chưa hiểu rõ. Trong tương lai. Đặc biệt nhất là khả năng tự học, tự tìm hiểu và sử dụng tài liệu tiếng anh.

Nguồn tài liệu tham khảo:

[1] "machinelearning căn bản," [Online]. Available: https://machinelearningcoban.com/2017/04/09/smv/.

[2] "sklearn," [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html.

[3] "geek for geek," [Online]. Available: https://www.geeksforgeeks.org/svm-hyperparameter-tuning-using-gridsearchcv-ml/.

[4] "learn opencv," [Online]. Available: https://www.learnopencv.com/histogram-of-oriented-gradients/.

[5] "viblo," [Online]. Available: https://viblo.asia/p/tim-hieu-ve-phuong-phap-mo-ta-dac-trung-hog-histogram-of-oriented-gradients-V3m5WAwxZO7.