COLOR PALETTE GENERATOR

ANLING CHEN, MUHAMMAD TANVEER,
SOWJANYA SRITHARASARMA

PROJECT GOAL

We will create a color palette of detected pedestrians based on their clothing. We will be using pedestrian detection, clothing detection, and color recognition to achieve this goal.

REAL LIFE APPLICATIONS

- Our project uses pedestrian detection, clothing detection, and color recognition.
- Some real life applications of each one include:
    - **Pedestrian detection:** surveillance systems, automobile safety, and urban planning
    - **Clothing detection:** crowd monitoring, search and rescue operations, and targeted advertising
    - **Color recognition:** quality control in manufacturing, food quality and sorting, and traffic sign recognition
- Combining these three allow us to make advanced and context-aware applications in various real-life scenarios, such as:
    - Identify individuals based in emergency situations.
    - Detect pedestrians entering secure zones in high-secure buildings.
    - Analyze customer demographics, clothing styles, and color preferences to optimize inventory and plan marketing strategies for retailers.

PROJECT TIMELINE

PHASE 1

Pedestrian detection using OpenCV

PHASE 3

Color palette of pedestrian’s clothing

PHASE 2

Clothing detection with the isolated pedestrian

PEDESTRIAN DETECTION

- OpenCV provides a pre-trained pedestrian detector based on the HOG features and an SVM classifier.
    - A **HOG (Histogram of Oriented Gradients)** feature is a feature descriptor used for object detection and recognition (in this case, specifically people). Its feature descriptor represents an image’s gradient or edge orientation patterns as a histogram in machine learning models to recognize objects.
    - A **Support Vector Machine (SVM)** is a supervised machine learning algorithm used for classification and regression tasks. When combined with the HOG descriptor, the SVM classifier serves as the decision-making component that classifies image regions as either containing the object of interest (people) or not.

CALCULATING HOG FEATURES

1. **Preprocessing:** The input image is resized to a fixed aspect ratio (64x128).
2. **Calculate Gradient Images:** Horizontal and vertical gradients are computed using gradient kernels or operators, and the magnitude and direction of gradients are determined.
3. **Calculate Histogram of Gradients in 8x8 Cells:** The image is divided into 8x8 cells, which yields 128 numbers (2 values per pixel). These 128 values are then summarized into a 9-bin histogram of gradients, enhancing robustness to noise. The histogram is a vector of 9 bins representing angles from 0 to 160 degrees. Gradients are represented by unsigned angles and magnitudes.
4. **16x16 Block Normalization:** To reduce sensitivity to lighting variations, histograms are normalized over 16x16 blocks, creating a 36x1 vector.
5. **Calculate the Histogram of Oriented Gradients Feature Vector:** The normalized 36x1 vectors from different blocks are concatenated into a final feature vector of size 3780.

Calculating Gradient Images

The gradient is obtained by combining magnitude and angle from the image. Considering a block of 3x3 pixels, first Gx and Gy is calculated for each pixel. First Gx and Gy is calculated using the formula below for each pixel value.

$$G_x(r,c) = I(r,c+1) - I(r,c-1) \quad G_y(r,c) = I(r-1,c) - I(r+1,c)$$

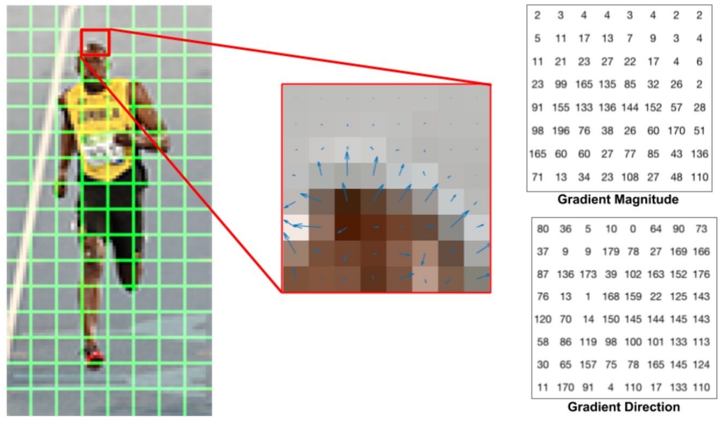
where r, c refer to rows and columns respectively. (Image by author)

After calculating Gx and Gy, magnitude and angle of each pixel is calculated using this formula:

$$Magnitude(\mu) = \sqrt{G_x^2 + G_y^2} \quad Angle(\theta) = |\tan^{-1}(G_y/G_x)|$$

Calculating Histogram of Gradients in 8x8 Cells

- After obtaining the gradient of each pixel, the gradient matrices (magnitude and angle matrix) are divided into 8x8 cells to form a block.
    - For each block, a 9-point histogram is calculated. A 9-point histogram develops a histogram with 9 bins and each bin has an angle range of 20 degrees (0, 20, 40, 60, …160).
    - The bins output the intensity of the gradient in that bin.



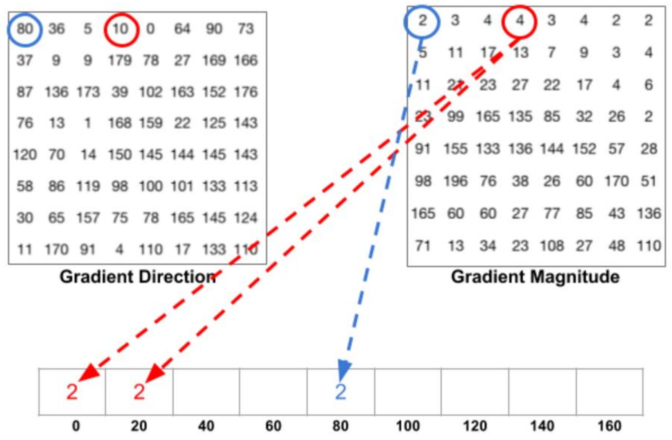
2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

Gradient Magnitude

80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

Gradient Direction

Center : The RGB patch and gradients represented using arrows. Right : The gradients in the same patch represented as numbers



80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

Gradient Direction

2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

Gradient Magnitude

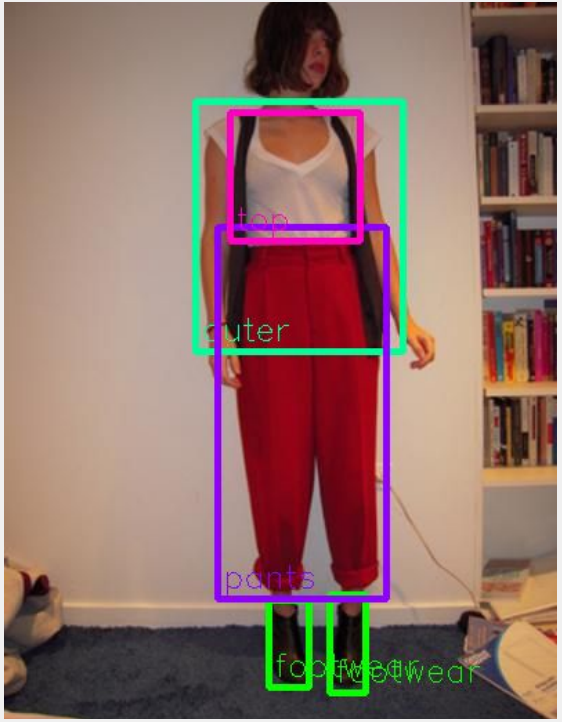
Histogram of Gradients

CLOTHING DETECTION

Completed using 2 technologies:

- **Ailia SDK** – an AI framework that houses various AI image detection models with a focus on speed and reduction of dependencies.

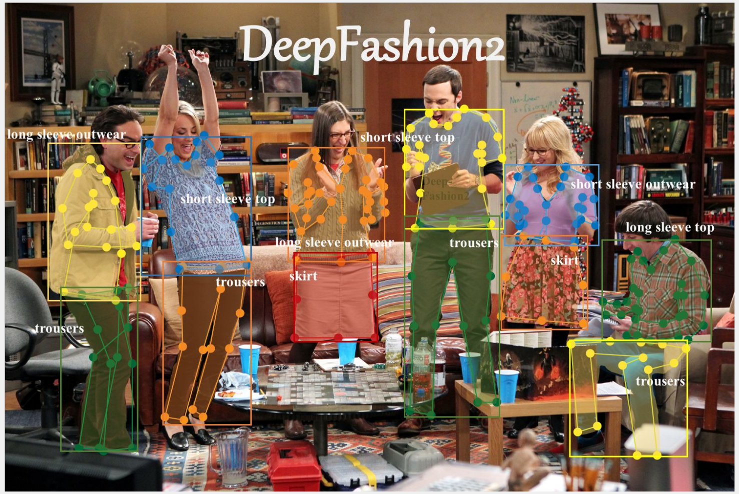
- **Clothing Detection (YOLOv3)** – a deep learning model trained to recognize clothing using DeepFashion2 and ModaNet datasets.



Datasets

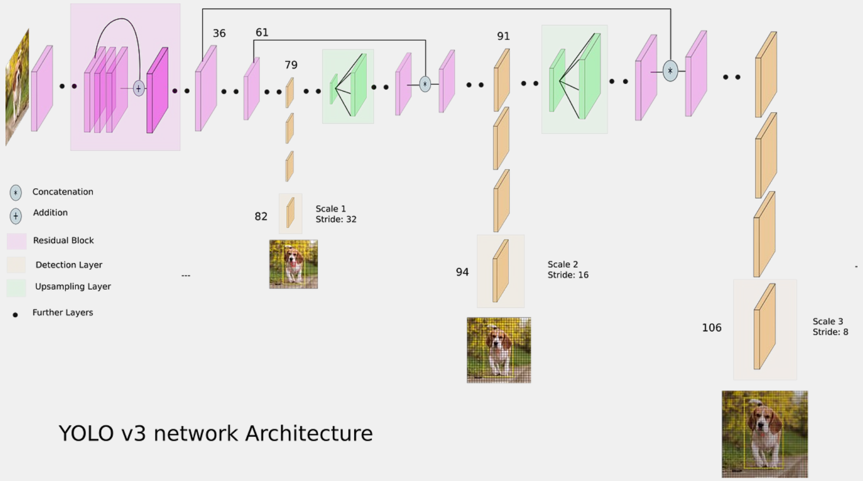
The Clothing Detection Model was trained on the following datasets: (25 categories and 801k items)

- **DeepFashion2:** a dataset that consists of images with clothing labeled in different variations with the images sourced from department stores and their customers
- **ModaNet:** a street fashion dataset with clothing labeled using color coded single polygons



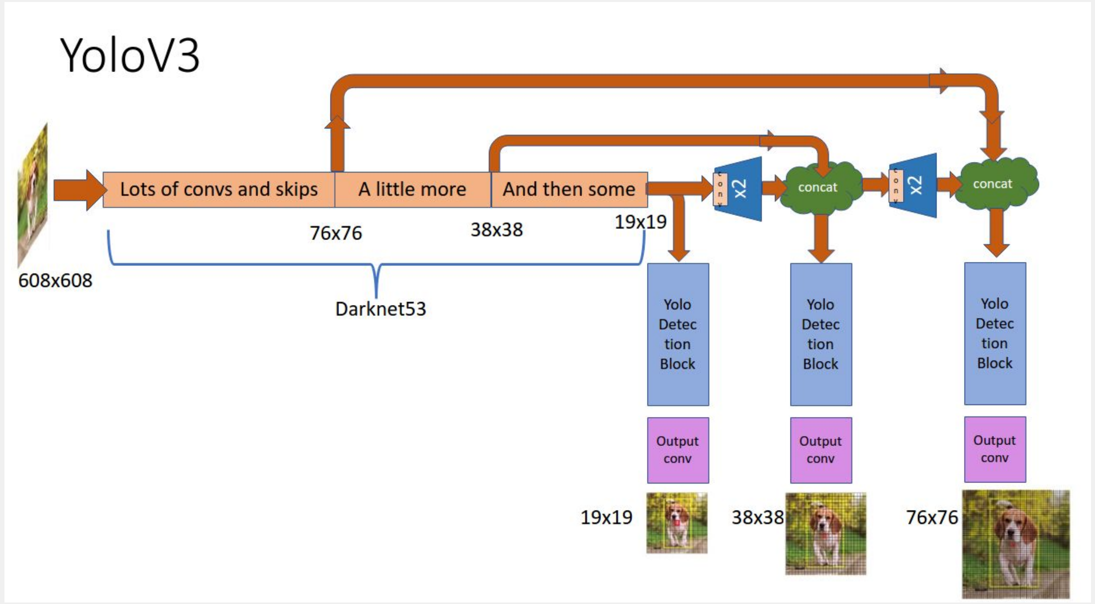
YOLOv3

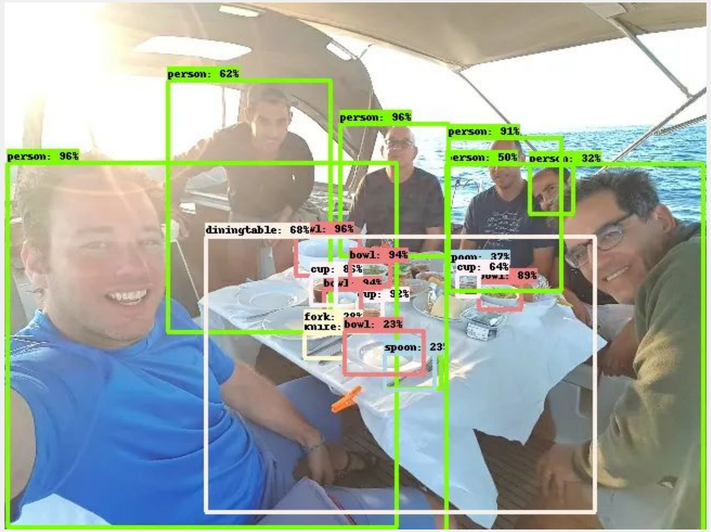
- Deep Convolutional Neural Network
- Real-time object detection algorithm that identifies specific objects in videos, live feeds, or images
- Features learned by the convolutional layers are passed onto a classifier which makes the detection prediction.
    - 1x1 convolutions
    - Size of feature map = size of prediction map



YOLO v3 network Architecture

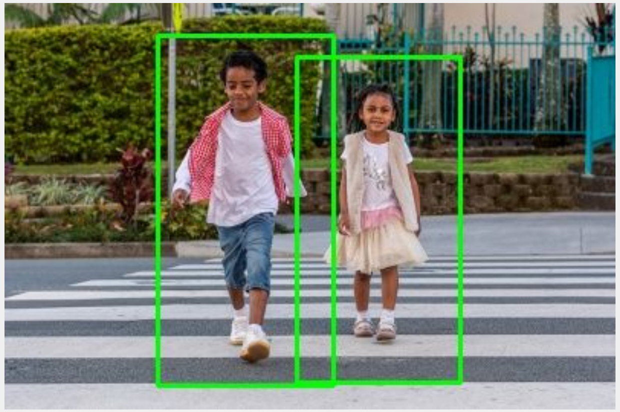
YOLOv3

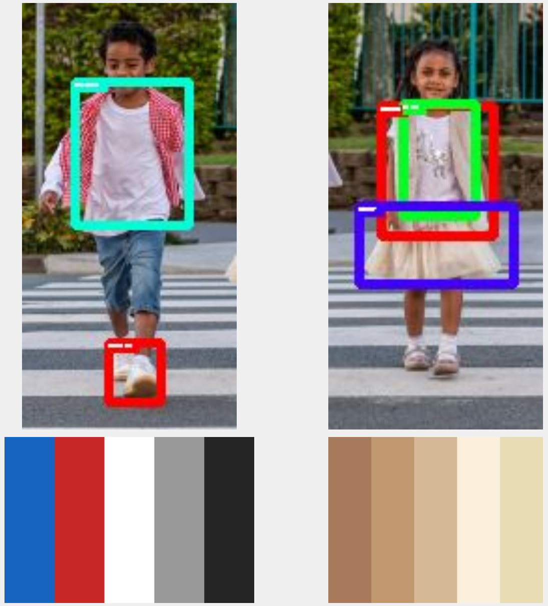

YoloV3
608x608
Lots of convs and skips
A little more
And then some
76x76
38x38
19x19
Darknet53
conv
x2
concat
conv
x2
concat
Yolo Detection Block
Yolo Detection Block
Yolo Detection Block
Output conv
Output conv
Output conv
19x19
38x38
76x76


person: 62%
person: 96%
person: 96%
person: 91%
person: 50%
person: 32%
diningtable: 68%
bowl: 96%
bowl: 94%
cup: 86%
bowl: 94%
cup: 92%
spoon: 37%
cup: 64%
bowl: 89%
fork: 28%
knife
bowl: 23%
spoon: 23%

1 Input Image

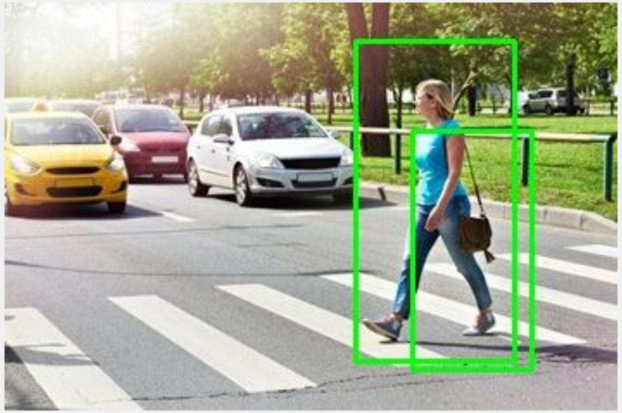
3 Clothing Detection





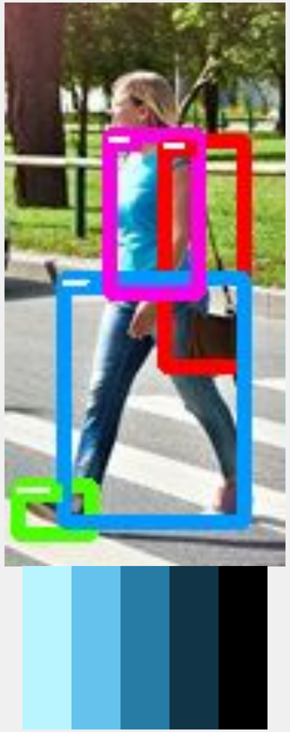
2 Pedestrians Detected

1 Input Image



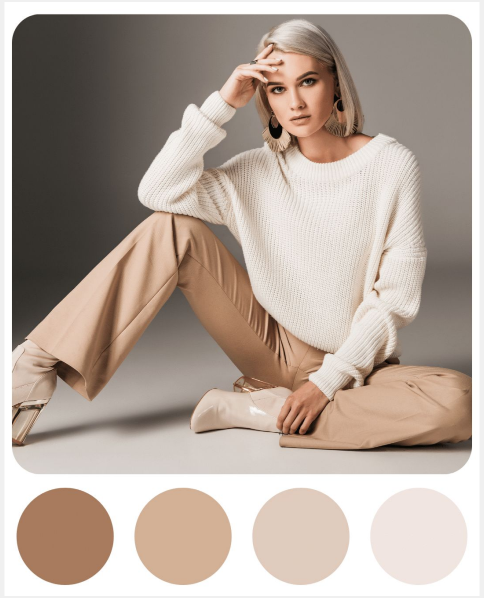
2 Pedestrians Detected

3 Clothing Detection



Color Palette Generation

The next step after clothing detection would be to generate a color palette from the pedestrians clothing



REFERENCES

- HOG Detector (Histogram of Oriented Gradients):
  - Original Research Paper: https://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf
  - https://learnopencv.com/histogram-of-oriented-gradients/
  - https://towardsdatascience.com/hog-histogram-of-oriented-gradients-67ecd887675f

- SVM (Support Vector Machine):
  - https://www.techtarget.com/whatis/definition/support-vector-machine-SVM

- OpenCV Pedestrian Detection Library:
  - https://pyimagesearch.com/2014/11/10/histogram-oriented-gradients-object-detection/

- DeepFashion2:
  - https://github.com/switchablenorms/DeepFashion2

- Clothing Detection Model:
  - https://github.com/axinc-ai/ailia-models/tree/master/deep_fashion/clothing-detection

THANK YOU

OTHER ALGORITHMS

- Haar cascade classifier
  - Haar cascades are machine learning object detection algorithms, proposed by Paul Viola and Michael Jones in 2001, commonly used for applications like facial recognition and object identification.

HAAR CASCADE OBJECT DETECTION

- Applications: Facial Recognition, Robotics, Autonomous Vehicles, Image Search, Agriculture
- Haar cascades are machine learning object detection algorithms, proposed by Paul Viola and Michael Jones in 2001, commonly used for applications like facial recognition and object identification. The process involves four key stages:
  - Calculating Haar features
  - Creating integral features
  - Adaboost training
  - Implementing cascading classifiers

HAAR CASCADE CLASSIFIER

1. **Calculating Haar features**
   a. Haar features are computed on adjacent rectangular regions within a detection window, representing patterns like edges or textures.
   b. Integral images are used to speed up Haar feature calculations.
2. **Creating integral features**
   a. Integral images expedite Haar feature calculations by creating array references for sub-rectangles, reducing the number of operations.
3. **Adaboost training**
   a. Adaboost selects and trains the best features, combining weak learners to form a strong classifier.
   b. Weak learners are created by moving a window over the input image and computing Haar features, comparing them to a threshold.
4. **Implementing cascading classifiers**
   a. Cascade classifiers consist of stages, each with a collection of weak learners.
   b. Stages reject negative samples quickly to improve efficiency.
   c. Maximizing a low false negative rate is crucial.