**COS40007 – Artificial Intelligence for Engineering**

**Portfolio 6-Studio 6-1**

Student Name: Nguyen Dinh Dung

Student ID: 104772138

A black and red sign with white text

Description automatically generated

# Abstract

The creation and assessment of a deep learning model for graffiti detection in photos and real-time video data using YOLO v5 is presented in this portfolio contribution. The project exhibits a thorough comprehension of PyTorch setup, iterative optimization, evaluation metrics, model training, and data preprocessing. The source code, model results, and labeled datasets are arranged as follows in the submission:

* Annotation Conversion Function: To convert the supplied annotation format in training labels to the YOLO annotation format, a new function called convert\_annotations is built. This feature makes sure that bounding boxes are formatted and normalized correctly, which makes it easier to integrate them with the YOLO v5 training process.
* YOLO v5 Model Training: 400 randomly chosen photos from the training dataset are used to train the YOLO v5 model. The model can correctly identify graffiti in photos thanks to this training procedure, which makes use of the transformed annotations. The model's performance is improved with each iteration of the training process. Every iteration's trained models are saved as best.the week-06-portfolio/train/runs/train/graffiti\_detection\_iter\_X/weights/ folders, where X is the number of iterations.
* Iterative Training and Optimization: The YOLO v5 model is retrained using fresh sets of 400 training and 40 test photographs in each iteration of an iterative training procedure. This procedure keeps on until either all of the test images have been used for training and testing, or 80% of the test images have an IoU above 90%. The model from the previous stage serves as the pretrained model for each iteration, promoting progressive learning and performance improvement. Under the train directory, the results of each iteration—such as CSV files and example annotated images—are arranged in the appropriate iteration directories.
* Real-Time Video Detection: Graffiti in real-time video footage is detected using the final, optimized YOLO v5 model. After processing a variety of video inputs, the model recognizes and labels occurrences of graffiti with confidence ratings and bounding boxes. The model's real-time detection skills are illustrated using sample video sources from Pexels. The detection results are stored in the results directory, where they are arranged according to each video track in subfolders like track, track2, etc.

**Repository Structure and Access**

All project requirements, documentation, source code, and results are organized within the week-06-portfolio repository on GitHub. The following links provide access to each component:

**Requirements:** <https://github.com/dinhdung6/Yolo-graffiti-detection/blob/main/week-06-portfolio/Portfolio-week6.pdf>

**Documentation:** <https://github.com/dinhdung6/Yolo-graffiti-detection/tree/main/week-06-portfolio>

**Source Code:** <https://github.com/dinhdung6/Yolo-graffiti-detection/tree/main/week-06-portfolio/code>

**YAML Config:** <https://github.com/dinhdung6/Yolo-graffiti-detection/tree/main/week-06-portfolio/train/yaml>

**YOLO v5 Model Training and Results (train):** <https://github.com/dinhdung6/Yolo-graffiti-detection/tree/main/week-06-portfolio/train>

**Evaluation Images:** [https://github.com/dinhdung6/Yolo-graffiti-detection/tree/main/week-06-portfolio/train/evaluation\_images\_iter\_{1-30](https://github.com/dinhdung6/Yolo-graffiti-detection/tree/main/week-06-portfolio/train/evaluation_images_iter_%7b1-30))

**Evaluation Results CSV:** [https://github.com/dinhdung6/Yolo-graffiti-detection/blob/main/week-06-portfolio/train/evaluation\_results\_iter\_{1-30}.csv](https://github.com/dinhdung6/Yolo-graffiti-detection/blob/main/week-06-portfolio/train/evaluation_results_iter_%7b1-30%7d.csv)

**YOLO v5 Best Model on each Iteration**: [https://github.com/dinhdung6/Yolo-graffiti-detection/tree/main/week-06-portfolio/train/runs/train/graffiti\_detection\_iter\_{1-30}/weights](https://github.com/dinhdung6/Yolo-graffiti-detection/tree/main/week-06-portfolio/train/runs/train/graffiti_detection_iter_%7b1-30%7d/weights)

**Detection Results (results):** <https://github.com/dinhdung6/Yolo-graffiti-detection/tree/main/week-06-portfolio/results>

# Yolo Models deep learning

1. Write a function to convert given annotation format in training labels to YOLO annotation format.   
   I have included the convert\_annotations function to your code in order to change the training labels' provided annotation format to the YOLO annotation format.

A screen shot of a computer program

AI-generated content may be incorrect.

1. Train and create a YOLO model by randomly taking 400 images from train data which can detect graffiti in the image

My code does the following actions to train and generate a YOLO model for graffiti detection using 400 randomly chosen training images:

* Selecting random images

A screenshot of a computer program

AI-generated content may be incorrect.

* Creating the Yaml configuration

A computer screen shot of a program

AI-generated content may be incorrect.

* Training

A computer screen with white text

AI-generated content may be incorrect.

1. Randomly take 40 images from test data and compute IoU for each and generate a CSV file containing 3 columns [image\_name, confidence value, IoU value]. If no graffiti is detected for an image then its IoU will be 0.  
     
   Your code does the following to calculate the Intersection over Union (IoU) for each of the 40 randomly chosen test photos and assess how well the trained YOLO model performs on them:

* Select test images:

A black screen with white text

AI-generated content may be incorrect.

* Evaluating the model and compute IOU:

A screen shot of a computer program

AI-generated content may be incorrect.

A computer screen shot of a program

AI-generated content may be incorrect.

A computer screen shot of a program code

AI-generated content may be incorrect.

1. Until IoU value of 80% images in your test data is over 90% or all images are utillised for training and testing purpose, you need to iteratively train and test the model with a new set of 400 training and 40 test images. Make sure you use the model of previous iteration as the pre-trained model for new iteration.

I use a while-loop to iteratively train and test the YOLO model until at least 80% of the test photos have an IoU higher than 90%. This approach is repeated until the stopping criteria are satisfied. Here's a detailed explanation of how this is accomplished:

* Initializing the Threshold

A screenshot of a computer program

AI-generated content may be incorrect.

* Iterative Training and Evaluation Loop

A computer screen with many colorful text

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

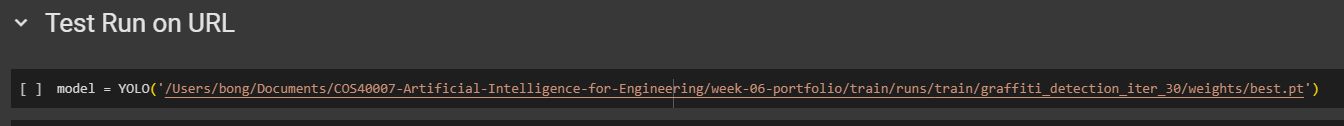
A screenshot of a computer program

AI-generated content may be incorrect.

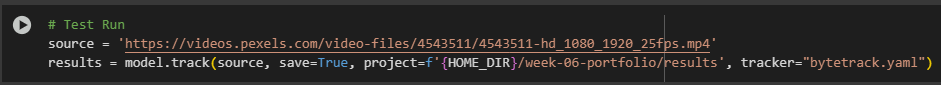
1. Use your final model to detect graffiti in real-time video data.

The following actions are taken by my code in order to implement the final trained YOLO model for real-time graffiti detection in video data:

* Load the trained model



* Testing on sample video from pexels providing in the requirements



* Retrieve video from Pexels through its API
  + Function to extracting video id from provided URL in requirements

A screen shot of a computer code

AI-generated content may be incorrect.

* + Function to retrieve the video stream url from Pexels

A screen shot of a computer program

AI-generated content may be incorrect.

* Retrieve video and use model to predict and tracking

A screenshot of a computer screen

AI-generated content may be incorrect.