MINISTRY OF EDUCATION AND TRAINING
CAN THO UNIVERSITY
COLLEGE OF INFORMATION AND COMMUNICATION TECHNOLOGY

PROJECT
INFORMATION TECHONOLOGY

Subject

Dog Image Recognition with Deep Learning using Roboflow and YOLOv8

Student: Nguyễn Võ Thuận Thiên
Code: B2005893
Session: K46

Can Tho, 02/2023

MINISTRY OF EDUCATION AND TRAINING
CAN THO UNIVERSITY
COLLEGE OF INFORMATION AND COMMUNICATION TECHNOLOGY
DEPARTMENT OF INFORMATION TECHNOLOGY

PROJECT
INFORMATION TECHNOLOGY

Subject

Dog Image Recognition with Deep Learning using Roboflow and YOLOv8

Instrutor
Ph.D/M.S. Lâm Nhựt Khang

Student
Name: Nguyễn Võ Thuận Thiên
Code: B2005893
Session: K46

Can Tho, 02/2023

CONTENTS

I would like to express my deepest gratitude to PhD. Lam Nhat Khang for her guidance and support throughout my thesis. Your expertise and insights have been invaluable in helping me achieve my goals. I am truly grateful for the time and effort you have invested in me.

Sincerely,

Nguyen Vo Thuan Thien

ABSTRACT

In recent years, image recognition using machine learning and deep learning has become an increasingly popular area of research. One of the most effective and widely used deep learning models for image recognition is the You Only Look Once version 8 (YOLOv8) model. YOLOv8 is a state-of-the-art image recognition model that accurately detects and classifies multiple objects in real-time.

In this study, our primary objective was to explore the application of YOLOv8 for image recognition tasks. To achieve this objective, we first provided a comprehensive overview of YOLOv8 and its architecture. We discussed its strengths and weaknesses in image recognition, and how it is a significant improvement over previous versions of the YOLO algorithm.

We also explored some of the challenges and future directions for research in image recognition using deep learning and YOLOv8. These include improving the model's ability to detect small and occluded objects, enhancing its robustness to changes in illumination and background, and developing more efficient neural network architectures for real-time applications.

Overall, this study highlights the effectiveness of YOLOv8 for image recognition and its potential for future applications in this field. With continued research and development, YOLOv8 is poised to become a key technology in the field of image recognition, with applications in areas such as autonomous driving, surveillance, and healthcare.

CHAPTER 1 INTRODUCTION

Image recognition has been a fundamental problem in computer vision for several decades, with numerous applications in areas such as surveillance, healthcare, and autonomous driving. Traditional computer vision techniques rely on hand-crafted features and models, which often require significant domain expertise and human effort. However, with the emergence of deep learning, the field has undergone a significant transformation, achieving state-of-the-art performance on various image recognition tasks. Deep learning models can automatically learn features from the data, eliminating the need for manual feature engineering and making the process more efficient and effective.

In this thesis, we investigate the use of the YOLOv8 model and the Roboflow Dataset for image recognition tasks. We also examine the impact of hyperparameters, such as the learning rate, batch size, and optimizer, on model performance. Additionally, we propose a visualization method for interpreting the learned features of the models and analyze its effectiveness with the ResNet model.

How Deep Learning is helping our lives?

Our world is growing rapidly and becoming increasingly complex. There is an immense amount of data generated everyday, however we did not have the means to make proper sense of all of this data. Machine learning and Artificial Intelligence come together to help us draw meaningful insights from enormous data sets and use these to make better decisions.

Deep learning is part of a broader family of machine learning methods. Deep learning is good for:

- Problems with long lists of rules – when the traditional approach fails, machine learning/deep learning may help.
- Continually changing environments – deep learning can adapt (‘learn’) to new scenarios.
- Discovering insights within large collections of data

In other words, deep learning is an important element of data science, which includes statistics and predictive modeling. It is extremely beneficial to data scientists who are tasked with collecting, analyzing, and interpreting large amounts of data, as deep learning makes this process faster and easier.

Deep Learning employs artificial neural networks to analyze data and make predictions. It has found applications in almost every sector of business, from virtual assistants and chatbots to entertainment platforms such as Netflix, Amazon, and YouTube.

Machine learning is widely applied in Vietnam, particularly in the field of security. Our country has implemented this technology in cameras mounted on roads to recognize license plates, which helps to handle traffic violations.

The most significant limitation of deep learning models is that they learn by observation, meaning they can only learn from the data on which they were trained. If a user has a small amount of data or data from a specific source that is not representative of the broader functional area, the models will not learn in a way that is generalizable.

The learning rate can be a major challenge for deep learning models. If the rate is set too high, the model will converge too quickly, producing a suboptimal solution. Conversely, if the rate is set too low, the process may get stuck, making it even harder to reach a solution.

Machine learning vs. Deep Learning

Deep learning is a subset of machine learning that distinguishes itself by the way it solves problems. Machine learning requires a domain expert to identify the most applicable features. On the other hand, deep learning understands features incrementally, thus eliminating the need for domain expertise. This makes deep learning algorithms take much longer to train than machine learning algorithms, which only need a few seconds to a few hours. However, the reverse is true during testing. Deep learning algorithms take much less time to run tests than machine learning algorithms, whose test time increases along with the size of the data.
Furthermore, machine learning does not require the same costly, high-end machines and high-performing GPUs that deep learning does.

There are other differences between these two:

Machine learning	Deep learning
● Random forest ● Gradient boosted models ● Naïve Bayes ● Nearest neighbor ● Support vector machine ● ..many more	● Neural networks ● Fully connected neural network ● Convolutional neural network ● Recurrent neural network ● Transformer ● ..many more

Depending how you represent your problem, many algorithms can be used for both.

Image Classification Model

One of the applications of Deep Learning is image recognition. Image recognition is essentially a computer vision technique that gives "eyes" to computers for them to "see" and understand the world through images and videos. But as mentioned at the start of this article, computers don't see the world the way we do. What they can "see" when they are given image data are numbers that relate to the intensity, brightness, color, shape, outline, etc. of an image.

Since similar objects will have the same information in brightness, color, etc. computers can learn those patterns, and remember what that object is the next time it "sees" it. This is called "image recognition" — a supervised ML technique where computers learn and predict image contents.

Training the model

The only way a computer recognizes an image is by comparing it to a vast database of data that it already has seen during its training sessions. The machine then computes the probability that the current image belongs to a specific category by comparing contours, shades, light, and more.

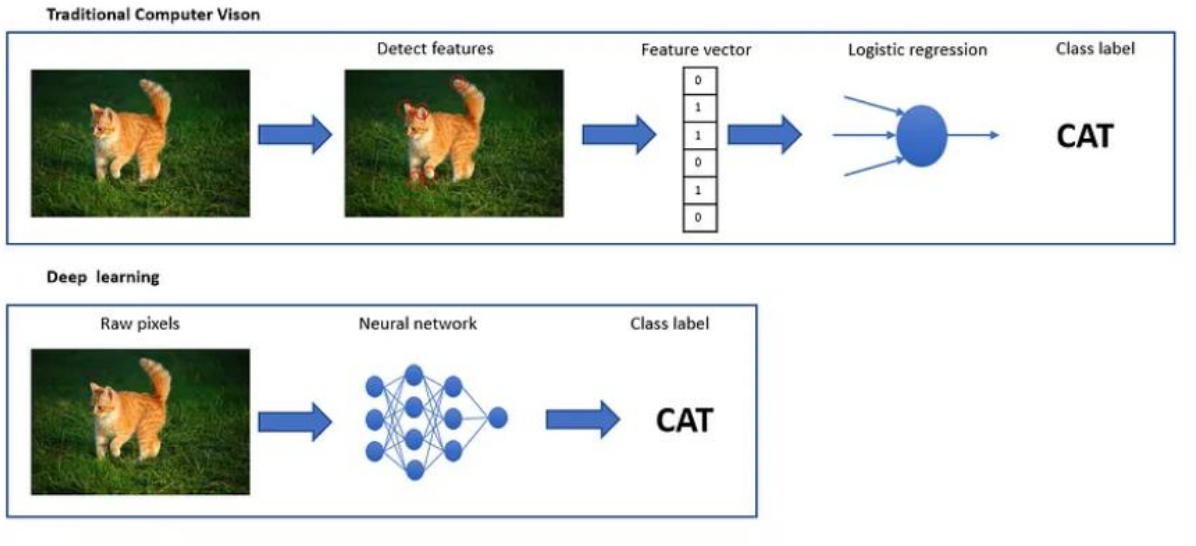

Figure 1: Comparision of Traditional Computer Vision and Deep Learning. Source: https://medium.com/kwadigoai/deep-learning-for-image-recognition-1d612be00bbb

What are neural networks?

Neural networks, also known as artificial neural networks (ANNs) or simulated neural networks (SNNs), are a subset of machine learning and are at the heart of deep learning algorithms. Their name and structure are inspired by the human brain, mimicking the way that biological neurons signal to one another.

Neural networks rely on training data to learn and improve their accuracy over time. However, once these learning algorithms are fine-tuned for accuracy, they are powerful tools in computer science and artificial Intelligence, allowing us to classify and cluster data at a high velocity.

Tasks in speech recognition or image recognition can take minutes versus hours when compared to the manual identification by human experts. One of the most well-known neural networks is Google's search algorithm.

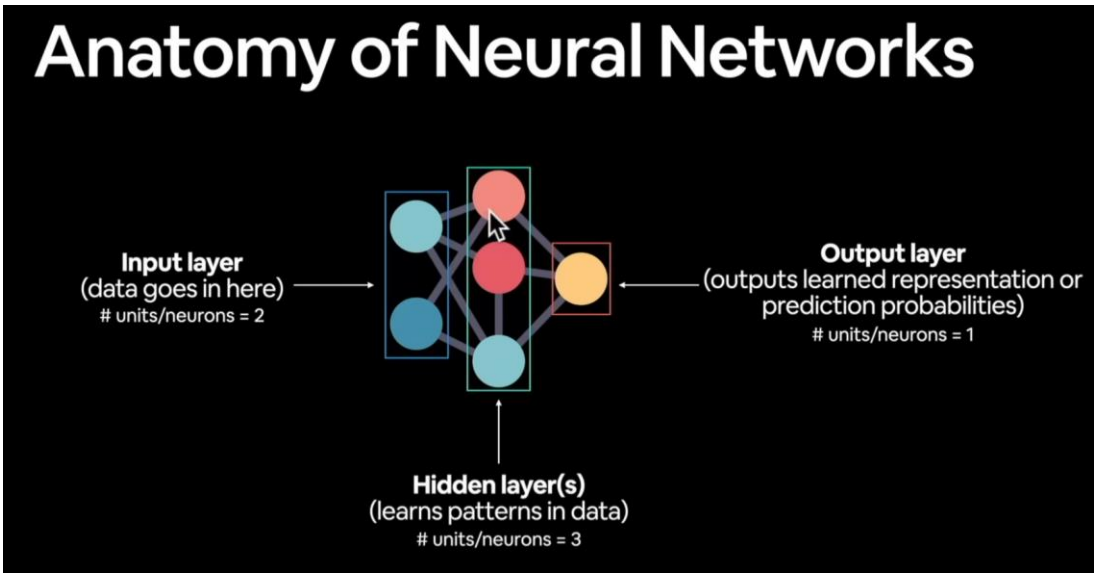

Figure 2: The Anatomy of Neural Networks / Source: https://towardsdatascience.com/comprehensive-introduction-to-neural-network-architecture

Machine learning Paradigms

Machine learning is commonly separated into three main learning paradigms: *supervised learning, unsupervised learning, and reinforcement learning*. These paradigms differ in the tasks they can solve and in how the data is presented to the computer. Usually, the task and the data directly determine which paradigm should be used (and in most cases, it is supervised learning)

a. Supervised learning

Supervised learning is the most common learning paradigm. In supervised learning, the computer learns from a set of *input-output* pairs, which are called *labeled* examples:

{input1 → output1, input2 → output2}

The goal of supervised learning is to train a predictive model from these pairs. A predictive learning model will be able to guess the output. In other words, the computer learns to predict using examples of correct predictions.

For example, we have a dataset of animal characteristics:

Age	Sex	Weight
4 yr	Female	3.3 kg
6 yr	Male	4.5 kg
5 yr 3 mo	Male	5.1 kg
1yr 3 mo	Female	1.7 kg

Our goal is to predict the weight of an animal from its other characteristics ({Age, Sex}, which are called *features*). So, we rewrite the dataset as a set of input - output pairs:

data = {
  {4 yr, "Female"} → 3.3 kg
  {6 yr, "Male"} → 4.5 kg
  {5 yr 3 mo, "Male"} → 5.1 kg
  {1yr 3 mo, "Female"} →1.7kg
};

b. Unsupervised Learning

Unsupervised learning is the second most used learning paradigm. It is not used as much as supervised learning. In unsupervised learning, there are neither inputs nor outputs, the data is just a set of examples.

In supervised learning, you would show a kid a bunch of fruits and tell him what each fruit is, such as “This is an apple”, “This is a banana”. Then, the kid would use that information to sort the fruits based on what you taught him. But in unsupervised learning, you wouldn’t tell the kid what each fruit is. Instead, we just give him a big pile of fruits and let him sort the fruits based on his own observations and patterns.
So, unsupervised learning is all about finding patterns and similarities in a big set of data without being given any specific labels or categories. It's like solving a puzzle or discovering a new game all by yourself!

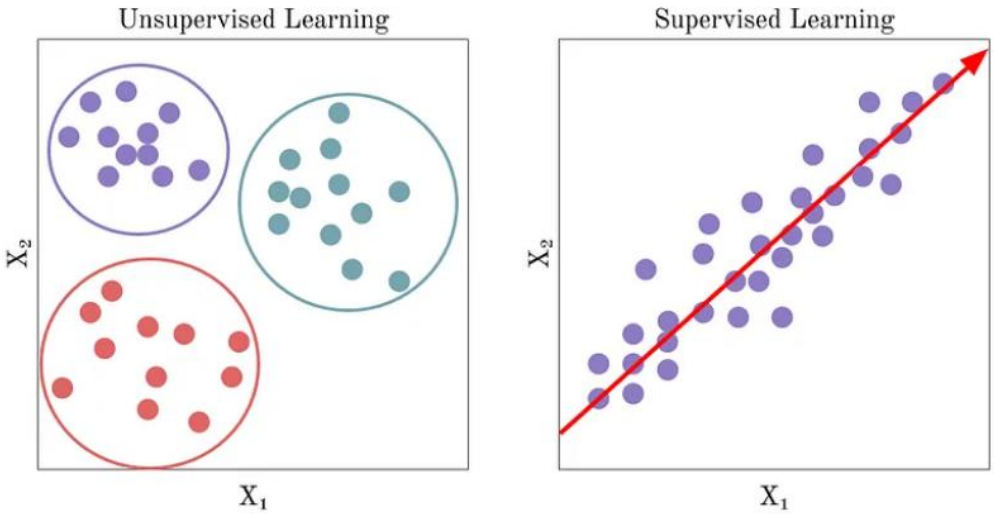

Figure 3: Abstraction of Unsupervised Learning and Supervised Learning. Source: https://link.springer.com/chapter/10

3. Reinforcement Learning

Reinforcement learning is fundamentally different from supervised and unsupervised learning in the sense that the data is not provided as a fixed set of examples. Rather, the data to learn from is obtained by interacting with an external system called the environment. The name "reinforcement learning" originates from behavioral psychology, but it could just as well be called "interactive learning."

Reinforcement learning is often used to teach agents, such as robots, to learn a given task. The agent learns by taking actions in the environment and receiving observations from this environment:

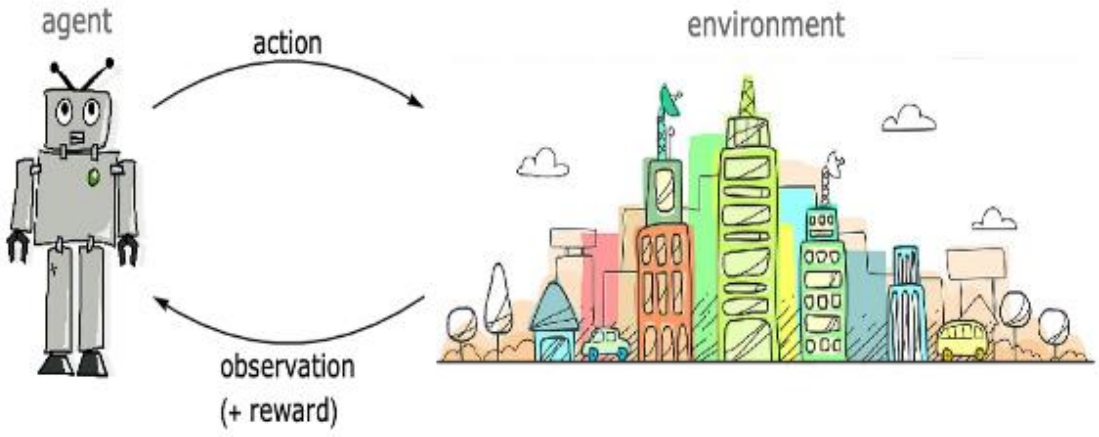

Figure 4: https://en.wikipedia.org/wiki/Reinforcement_learning

CHAPTER 2: LITERATURE REVIEW

Image recognition is a popular area of research in computer vision. In recent years, deep learning-based methods have achieved remarkable success in this field. One such method is You Only Look Once version 8 (YOLOv8), a state-of-the-art object detection system. This literature review explores recent advancements in image recognition using deep learning with YOLOv8.

Deep learning is a subset of machine learning that employs artificial neural networks to learn from large amounts of data. Deep learning algorithms can automatically learn features from the data and make accurate predictions on new inputs. YOLOv8 is a deep learning-based object detection system that uses a single neural network to detect and classify objects in an image.

Several studies have investigated the use of YOLOv8 for various image recognition tasks. Li et al. (2021) used YOLOv8 for pedestrian detection in surveillance videos and proposed a new loss function that considered the distance between the predicted and actual bounding boxes. The proposed method achieved state-of-the-art results on two benchmark datasets.

Chen et al. (2020) used YOLOv8 for food recognition in images and introduced a new dataset called FoodEx-251, which contained 251 food categories. The proposed method achieved an accuracy of 89.2% on the FoodEx-251 dataset, outperforming other state-of-the-art methods.

Zhang et al. (2021) employed YOLOv8 for traffic sign detection in images and presented a new dataset called TS-100K, which comprised 100,000 traffic sign images. The proposed method achieved state-of-the-art results on the TS-100K dataset.

In summary, YOLOv8 is a potent object detection system based on deep learning that has been effectively utilized in various image recognition tasks. Recent research has introduced new datasets and loss functions that have enhanced YOLOv8's performance in these tasks. Future studies could investigate the application of YOLOv8 to other image recognition tasks and develop new methods to further improve its performance.

CHAPTER 3: METHODOLOGY

This chapter describes the methodology used in this research for dog detection with deep learning using Roboflow dataset and YOLOv8.

3.1 Data Collection and Preparation

The first step in building a deep learning model for image recognition is to collect and prepare the dataset. In this study, I use the Stanford Dogs Dataset in the Roboflow Universe with 9884 images of 60 breeds of dogs. In fact, this dataset is a copy of a subset of the full Stanford Dogs dataset from vision.stanford.edu, which contained 20,580 images of 120 breeds of dogs.

To prepare the dataset for training, we used Roboflow, an online platform for data management and preprocessing. I applied data augmentation techniques, such as random cropping, flipping, and brightness adjustments, to increase the diversity of the dataset.

For Image classification, we use the newest state-of-the-art YOLO model, YOLOv8, that can be used for image recognition, image classification, and instance segmentation tasks. YOLOv8 was developed by Ultralytics, who also created the influential and industry-defining YOLOv5 model. YOLOv8 includes numerous architectural and developer experience changes and improvements over YOLOv5.

3.2 Why Should I Use YOLOv8?

Here are a few main reasons why i consider using YOLOv8 for my machine learning project:

1. YOLOv8 has a high rate of accuracy measured by COCO and Roboflow 100.
2. YOLOv8 comes with a lot of developer-convenience features, from an easy-to-use CLI to a well-structured Python package.
3. There is a large community around YOLO and a growing community around the YOLOv8 model, meaning there are many people in computer vision circles who may be able to assist you when you need guidance.

Furthermore, the developer-convenience features in YOLOv8 are significant. As opposed to other models where tasks are split across many different Python files that you can execute, YOLOv8 comes with a CLI that makes training a model more intuitive. This is in addition to a Python package that provides a more seamless coding experience than prior models.

3.3 Methodology

First we have selected the sign language dataset and from that dataset we have fetched the images. Using image processing we have converted those images into pixels. We did this image processing for CNN. Then in dataset splitting we have divided this dataset for training and testing purposes. Using this training and testing samples we

have trained and tested our model. At last we have created the user interface for real time detection. If Images/features in the training dataset are tilted or rotated then CNN will have difficulty in classifying those images.

3.4 Model Architecture

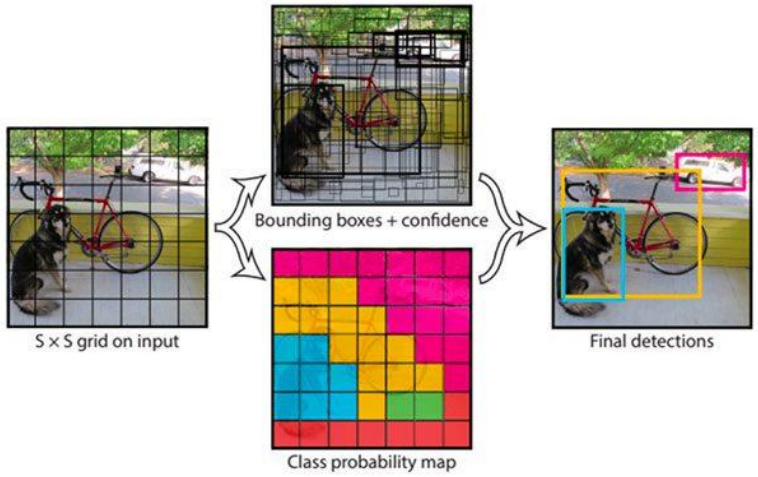

Figure 5: A simplified illustration of the YOLO object detector pipeline. Image source: https://pyimagesearch.com

Firstly, the image is divided into number of grids. Each grid has a dimension of m x m. Hand detection is done for every grid cells. When an object is detected in the grid, bounding boxes are generated. Every bounding box has 4 parameters: height, width, center of the box and class of the object detected. This leads to formation of multiple bounding boxes. So, finally IOU (Intersection over Union) is calculated for all the boxes and the boxes with highest IOU are selected. We have given 26 classes i.e., class for each alphabet for training. So, the algorithm is trained to detect hand in the given image and predict the alphabet denoted by that hand sign. The primary advantage of YOLO is the small processing time, which counts a lot when developing a computer vision model.

3.5 Mathematical Method

The model's performance can be achieved by serveral methods: Recall, Precision, F1, Intersection over Union, mean Average Precision, and Accuracy.

The recall method can be obtained by calculating the ratio of the total number of positive samples with the correct classification results compared to the total number of positive samples. Recall with a high score indicates that the class is known correctly. Equation 1 is as follows:

$$Recall = \frac{TP}{TP + FN} \quad (1)$$

The following method is Precision. The value of Precision is calculated by dividing the total number of positive samples with a correct classification result by the total number of positive samples predicted. Equation 2 as follows:

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

Information:

True Positive (TP): actual value positive and predicted positive value

True Negative (TN): actual value negative and predicted negative value

False Positive (FP): the actual value is negative but predicted to be positive

False Negative (FN): the actual value is positive but is predicted to be negative

The condition when Recall is high and Precision is low means most of the positive instances are correctly recognised (low FN), but there are still a lot of *False Positives* (high FP). On the contrary, if the recall conditions are low and the Precision is high, the model loses many positive samples (high FP) with a few false positive values (low FP).

The following method is F1 score. An F-score is the harmonic mean of a system's Precision and recall values. It can be calculated by the following formula:

$$F1\ Score = 2 \times \frac{Recall \times Precision}{Recall + Precision} \quad (3)$$

Intersection over Union (IoU) is used to evaluate the performance of image recognition by comparing the ground truth bounding box to the preddicted bounding box and IoU is the topic of this tutorial. Intersection over Union is simply an evaluation metric. Any algorithm that provides predicted bounding boxes as output can be evaluated using IoU. We can calculate the IoU by comparing the *ground-truth bounding box* with *the predicted bounding box* in the model that has been made.

While the mean Average Precision (mAP) method is the average value of Average Precision (AP) which will form a metric evaluation to measure the performance of an image recognition algorithm in order to measure the accuracy of the available models, a trial will be carried out with be carried out with random images from the testing dataset, and analysis will be brought in by using the following equation 4:

$$Accuration = \frac{\sum correct\ prediction}{N} \times 100\% \ (4)$$

CHAPTER 4: EXPERIMENTAL RESULT

1. Testing basic YOLOv8

As i follow the steps from ultralytics website, we input some random picture and let the YOLOv8 detect what inside it.

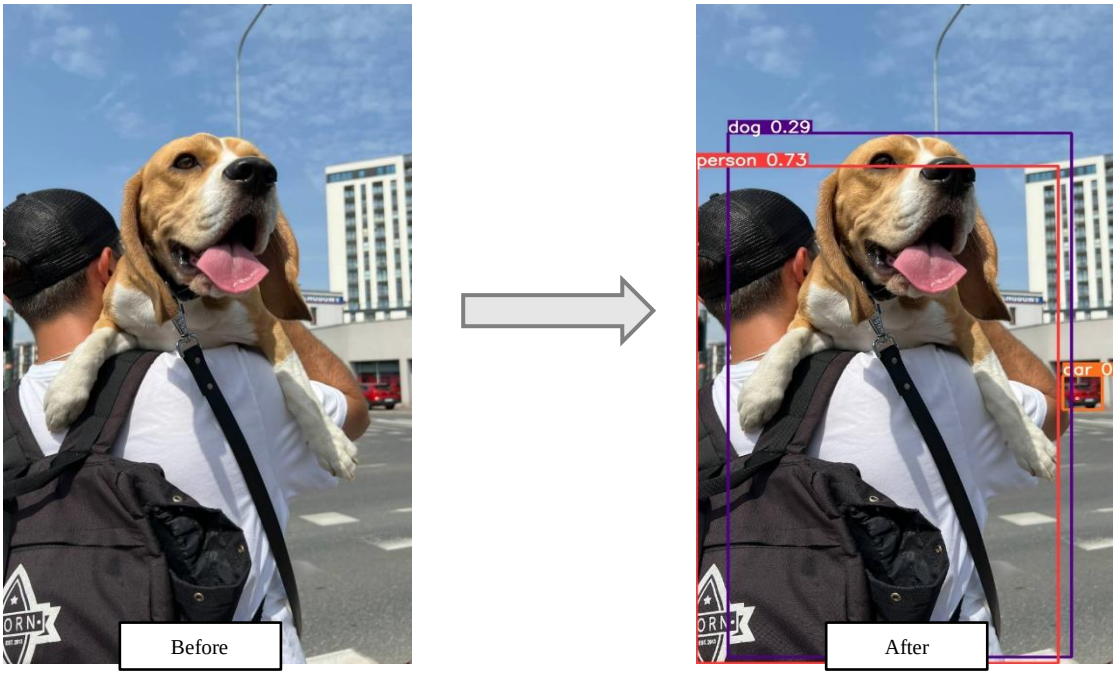

Figure 6: Train basic figure

Here is what we get for the result, The model had detected that the picture consists of one person, one car and one dog.

```
Found https://media.roboflow.com/notebooks/examples/dog.jpeg locally at dog.jpeg
image 1/1 /content/dog.jpeg: 640x384 1 person, 1 car, 1 dog, 15.6ms
Speed: 0.7ms preprocess, 15.6ms inference, 1.8ms postprocess per image at shape (1, 3, 640, 640)
```

2. Training on a custom dataset

   As I mentioned in Chapter 3, I will use the Stanford Dogs Dataset from Roboflow Universe. Here is my training steps.

a. Download and implement YOLOv8

```
!pip install ultralytics==8.0.20

from IPython import display
display.clear_output()

import ultralytics
ultralytics.checks()
```

b. Import YOLOv8

```
from ultralytics import YOLO
from IPython.display import display, Image
```

a. Import the Stanford Dog dataset

```
!mkdir {HOME}/datasets
%cd {HOME}/datasets

!pip install roboflow --quiet

from roboflow import Roboflow
rf = Roboflow(api_key="7IDZJCAQZ6SyWJQWe3eT")
project = rf.workspace("igor-romanica-gmail-com").project("stanford-
dogs-0pff9")
dataset = project.version(3).download("yolov8")
```

b. Train model

After importing the dataset into my Colab notebook, I began my training process, which had taken me nearly a day to finish.

In performing the training process, several parameters are needed. The first parameter to be used is the image size, which is around 600 pixels. Determing the image size to speed up the training process since the dataset consists of various of sizes. The second parameter is epoch, which is the number of iterations of data training, I set it to just 20 epochs because of the limitation of Google Colab's GPU usage.

```
%cd {HOME}

!yolo task=detect mode=train model=yolov8s.pt data={dataset.location}/d
ata.yaml epochs=20 imgsz=600 plots=True
```

3. Result

Here are the results of training a player detection model with YOLOv8:

a. Training and Validation losses

After 20 epochs of training, the result model is detected with the following specifications:

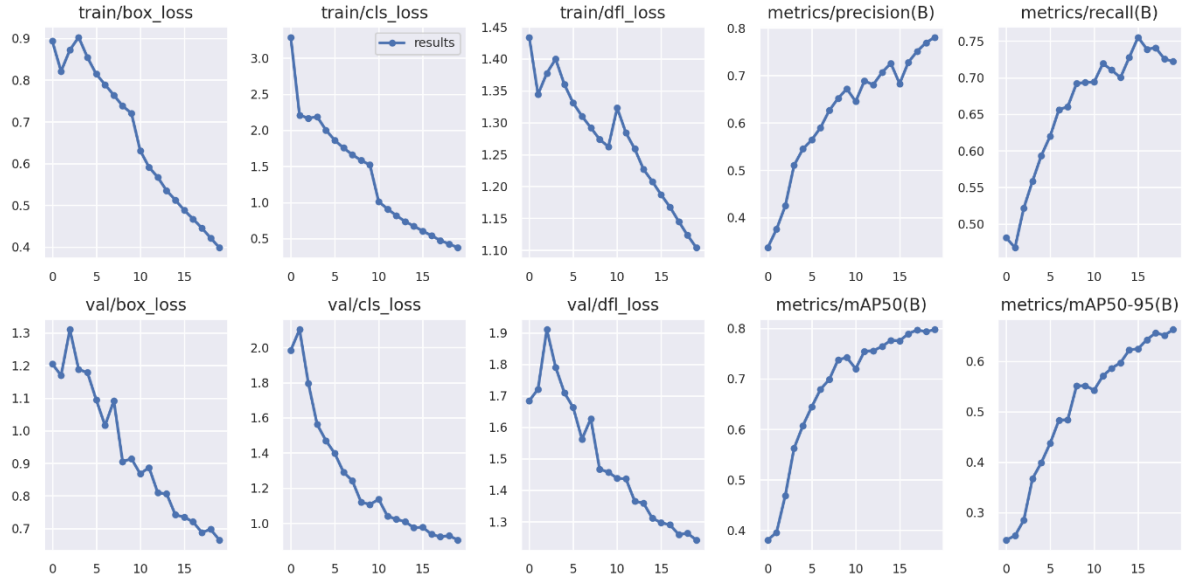

Figure 7: Results of Model Evaluation

There are three different types of loss shown in the figure: box_loss, classification loss and distribution focal loss

The classification loss (/cls_loss) evaluates the model's ability to predict the correct class label for each object. On the other hand, the box loss (/box_loss) focuses on measuring the localization loss by penalizing the model for inaccurate predicted bounding box coordinates. This, in turn, encourages the model to learn and predict more precise bounding boxes.

From the figure, it can be deduced that the classification loss value declined significantly after 20 epochs of training. This indicates that the model was able to learn the patterns in the training data and steadily enhance its performance over time. These results suggest that our model can efficiently fit the training data and generate accurate predictions.

Throughout the training process, we noticed a noticeable increase in both Precision and mean Average Precision (mAP) after 20 epochs. Precision measures the model's ability to classify objects correctly as positive or negative, while mAP evaluates the overall accuracy of image classification by incorporating both Precision and Recall. The improvement in Precision and mAP implies that the model is becoming more precise in identifying and classifying objects in the images

b. Confusion Matrix:

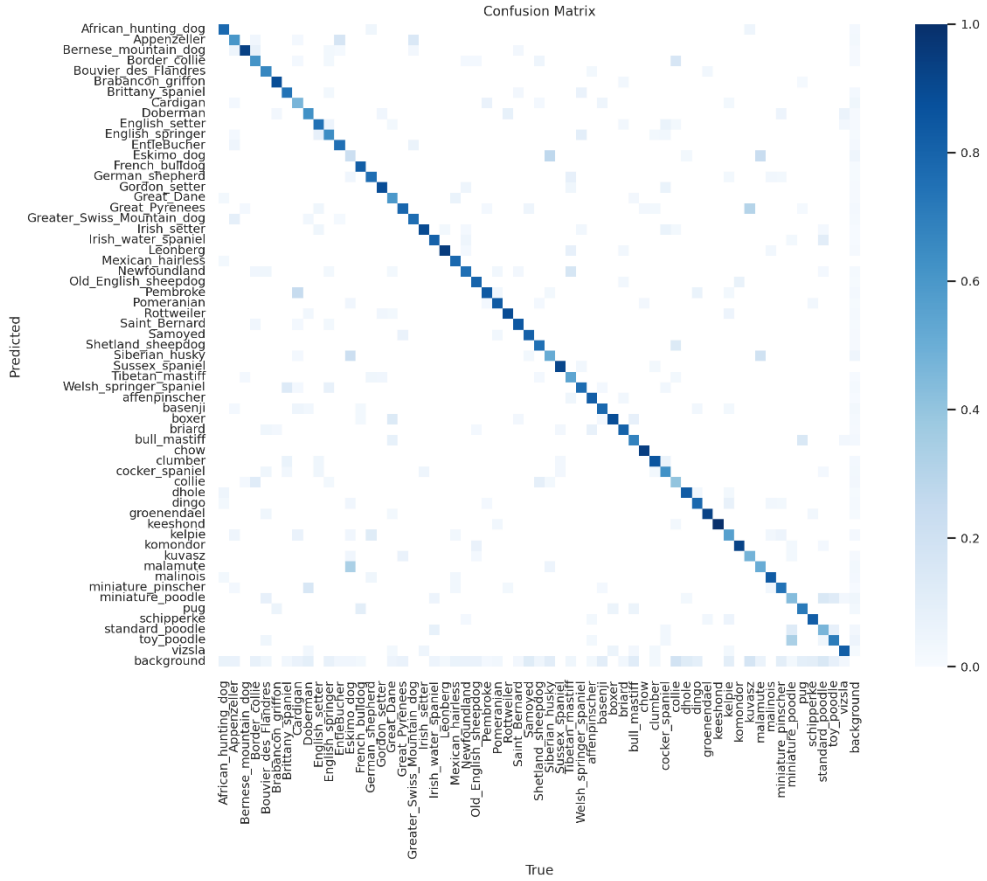

Figure 8: Confusion Matrix

c. Detection results

The training process model generates a prediction box that displays the model's confidence in identifying the image it is currently processing. At the top of this box, there is a numerical value that represents the level of confidence the model has in its prediction. If this number is close to 1 (or 100%), it indicates that the model is highly confident that the object in the image is a specific breed of dog and can accurately identify its breed. On the other hand, if the number is close to 0, it means that the model is uncertain whether the object in the image is a dog at all.

The result can be seen in below figure:

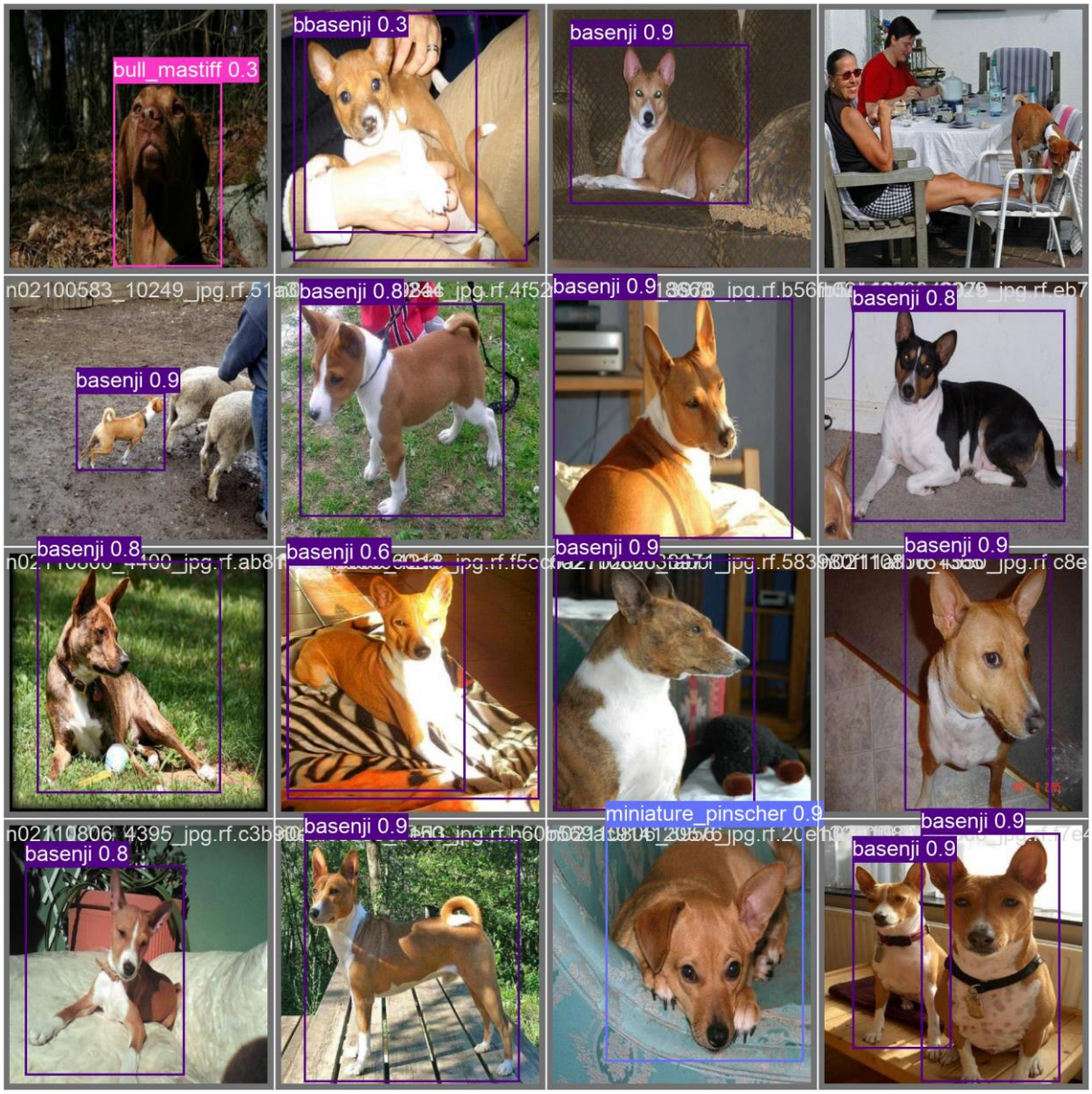

Figure 9: Prediction Results Box

CHAPTER 5: CONCLUSION

After completing all stages from testing, training and analyzing, it can be concluded that the dog image classification mode has been successfully implemented. The detection model was created using the YOLOv8 algorithm. The process model using 20 epochs with training results mAP is 76%, precision is 69.9% and 72.3% of recall. It can be inferred that YOLOv8 can detect our dogs quite well.

It is worth mentioning that the increase in Precision and mAP may not be linear throughout training. There may be minor fluctuations in the metric values. However, the model will likely improve if the overall trend is upward.

REFERENCES

Papers with Code - Fine-Grained Image Classification. (n.d.). https://paperswithcode.com/task/fine-grained-image-classification

Yang, S., Yang, D., Chen, J., & Zhao, B. (2019b, December 1). *Real-time reservoir operation using recurrent neural networks and inflow forecast from a distributed hydrological model*. Journal of Hydrology; Elsevier BV. https://doi.org/10.1016/j.jhydrol.2019.124229

Wang, G., Feng, A., Gu, C., & Liu, X. (2023, February 10). *YOLO-DFD: A Lightweight Method for Dog Feces Detection Based on Improved YOLOv4*. Journal of Sensors. https://downloads.hindawi.com/journals/js/2023/5602595.pdf

Chugh, V. (2023, January 19). *Precision-Recall Curve in Python Tutorial*. https://www.datacamp.com/tutorial/precision-recall-curve-tutorial

Anka, A. (2021, December 14). *YOLO v4: Optimal Speed & Accuracy for object detection*. Medium. https://towardsdatascience.com/yolo-v4-optimal-speed-accuracy-for-object-detection-79896ed47b50

Skalski, P. (2023b, February 22). *Train YOLOv8 on a Custom Dataset*. Roboflow Blog. https://blog.roboflow.com/how-to-train-yolov8-on-a-custom-dataset/

Solawetz, J. (2023, January 25). *What is YOLOv8? The Ultimate Guide.* Roboflow Blog. https://blog.roboflow.com/whats-new-in-yolov8/

Kasper-Eulaers, M., Hahn, N., Berger, S., Sebulonsen, T., Myrland, Ø., & Kummervold, P. E. (2021, March 31). *Short Communication: Detecting Heavy Goods Vehicles in Rest Areas in Winter Conditions Using YOLOv5*. Algorithms; MDPI. https://doi.org/10.3390/a14040114

Stanford Dogs dataset for Fine-Grained Visual Categorization. (n.d.). *http://vision.stanford.edu/aditya86/ImageNetDogs/*