CS484 Project Progress Report Number Detection from UNO®Cards using YOLOv5

Efe Tarhan
22002840
Electrical and Electronics Engineering Department
Bilkent University
Ankara
efe.tarhan@ug.bilkent.edu.tr

Ege Yüceel
22002840
Electrical and Electronics Engineering Department
Bilkent University
Ankara
ege.yuceel@ug.bilkent.edu.tr

***Abstract*—In this project, we explore the application of YOLOv5 (You Only Look Once version 5) for detecting and classifying UNO®cards from images with varied backgrounds. By utilizing the "Uno Cards Dataset" and focusing on optimizing model hyperparameters, we achieved significant accuracy in identifying card numbers and symbols. The dataset comprises images annotated with value vectors for precise object localization, facilitating the model's training on distinguishing among 14 card classes, including color variations. Our results demonstrate YOLOv5's effectiveness in real-time object detection tasks within complex image environments, presenting a valuable approach for similar computer vision challenges.**

***Index Terms*—computer vision, deep learning, YOLOv5, detection, classification**

I. Introduction

Object detection and classification remain central challenges in the field of computer vision, with applications ranging from surveillance and security to aiding in the development of autonomous vehicles. In recent years, the advancement of deep learning algorithms has significantly enhanced the capability of computer vision systems to recognize and classify objects within images and real-time video feeds accurately. Among these advancements, the YOLO (You Only Look Once) series has emerged as a pivotal architecture due to its balance between speed and accuracy, making it particularly suitable for real-time applications.

The YOLO architecture has undergone several iterations, each improving upon its predecessor in terms of detection accuracy, speed, and model efficiency. The latest in this series, YOLOv5, introduces a range of enhancements, including architectural optimizations and training techniques, which further bolster its performance. Given YOLOv5's capabilities, this project explores its application in a niche yet practical scenario: detecting and classifying UNO cards from images. UNO, a popular card game with a diverse set of cards, presents a unique challenge for object detection systems due to the variety of colors, numbers, and symbols that must be accurately identified and classified.

This project aims to leverage YOLOv5's deep learning framework to develop a robust model capable of detecting and classifying UNO cards under varying conditions, including different backgrounds and card orientations. By utilizing a custom "Uno Cards Dataset", this study focuses on optimizing model hyperparameters and training techniques to achieve high accuracy in card detection and classification. The choice of UNO cards as the object of detection allows us to demonstrate YOLOv5's versatility and efficiency in handling complex image environments, thereby showcasing its potential applicability across a broader range of computer vision tasks.

In doing so, this research not only contributes to the ongoing development and understanding of YOLO architectures but also provides insights into the practical considerations and challenges involved in adapting deep learning models for specific, real-world applications. The following sections will detail the problem definition, dataset characteristics, model architecture, training process, and the results achieved through this project, offering a comprehensive overview of the application of YOLOv5 in the nuanced task of UNO card detection and classification.

II. Problem Definition

Our aim in this project is to detect and classify UNO®cards by the numbers and symbols on their faces on different background images. For this task we decided to use YOLOv5 [1] which is a powerful deep learning based tool for computer vision tasks. YOLOv5 is a deep learning model for object detection tasks, offering a balance between speed and accuracy. It's designed to recognize and localize objects within images quickly, making it suitable for real-time applications. YOLOv5 is the unofficial successor in the YOLO series, featuring improvements like enhanced model architecture and training procedures for better performance. It's widely used in various fields, such as surveillance, automotive, and retail, for tasks that require identifying and tracking objects. Throughout the project we aim to find optimal hyperparameters for the model in order to increase our performance of detection and

Fig. 1. A sample image from the UNO®Cards Dataset

classification. We will be using a custom dataset called The ”Uno Cards Dataset” [1] which can be found in the website Roboflow throughout the project.

III. Dataset

The ”Uno Cards Dataset” [2] used for the project contains the images of UNO®cards on various backgrounds. The input features of the dataset are raw RGB images that may or may not contain an UNO®card, whereas the outputs consist of a 5-dimensional value vector for each card in an image. This vector contains the categorical label of the card, the horizontal and vertical starting corner pixel of the image, and the horizontal and vertical pixel width of the image, which allows the algorithm to create a bounding box around the region of interest and assign a label to the card with a probability. It must also be noted that there can be multiple UNO®cards in an image, which will result in several feature vectors for each image. All images in the dataset have the size of 416x416 with RGB pixels. A sample image and its corresponding value object is shown in Figure 1.

```
0 0.32211538461538464 0.23677884615384615
    0.07692307692307693 0.08413461538461539
0 0.4026442307692308 0.2860576923076923
    0.08413461538461539 0.08653846153846154
13 0.48197115384615385 0.37259615384615385
    0.0889423076923077 0.08653846153846154
```

From the value vector presented in Figure 1, it is evident that all three UNO®cards depicted in the image have been successfully detected. The value ”13” represents the class label of the red UNO®card, which is marked with the number ”8” whereas the value ”0” represents the cards with value ”0”. This value mismatch is due to the alphanumeric order of the labels. The subsequent four numbers delineate the coordinates of the bounding box encapsulating the card's numeral on the image. The dataset contains 14 different class of cards where 9 of them can have 4 different colors whereas other 5 has a fixed appearance. Throughout this project we will be dealing with detecting and classifying the images with cards in this dataset.

IV. Model Architecture

YOLO v5, is an advanced deep learning algorithm used for object detection tasks, part of a series of YOLO algorithms that prioritize speed and accuracy. The architecture of YOLO v5 can be divided into several key components, each with a specific function that contributes to the model's overall performance. The model will be explained using the documentation provided by Utralytics [3].

The backbone of YOLO v5, typically CSPDarknet53, uses a series of convolutional layers to extract features from input images. The CSP (Cross Stage Partial) structure optimizes the Darknet53 by dividing the feature map of the base layer into two parts and then merging them through a cross-stage hierarchy. This method reduces the computational cost and enhances the learning capability of the network. The convolutional layers within this part vary in filter size and stride, enabling the extraction of features at different scales and complexities. Batch normalization and Leaky ReLU activation functions follow these convolutional layers to stabilize the learning process and introduce non-linearity.

Following the Backbone is the Neck, which is tasked with feature fusion and refinement. The Neck of YOLO v5 employs an innovative combination of FPN (Feature Pyramid Network) and PAN (Path Aggregation Network) for feature fusion and enhancement. FPN improves the network's ability to detect objects at different scales by creating a pyramid of feature maps at multiple levels and then upscaling and merging these maps to ensure rich semantic information at all levels. PAN further refines this process by enhancing the flow of information in both bottom-up and top-down directions, ensuring that features from all levels are effectively aggregated and utilized. This structure involves additional convolutional layers for processing the feature maps, allowing for the detailed detection of objects regardless of their size.

The third critical component is the Head, which is the final part of the model. The detection head of YOLO v5 is responsible for the final prediction of bounding boxes, object classes, and confidence scores. It employs convolutional layers designed to map the enriched feature maps to a tensor containing the predictions. This tensor is organized such that each cell corresponds to a region in the input image and contains information about bounding boxes (coordinates and size), objectness score (the probability that an object exists within the bounding box), and class probabilities (the likelihood of each object class being present). The design allows for parallel processing of all grid cells in an image, significantly speeding up the detection process. The full architecture of the YOLOv5 model is shown in Figure 2.

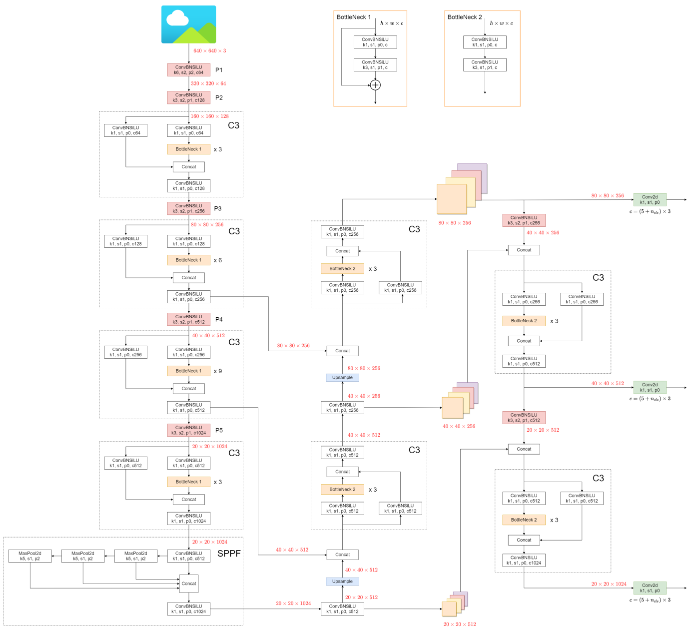

Fig. 2. Model Arcihtecture of YOLO v5 [3]

The YOLO v5 architecture incorporates several advancements and optimizations over its predecessors, enhancing its capabilities. One of the key capabilities is its speed, enabling real-time object detection, which is crucial for applications requiring instant feedback, such as autonomous vehicles and surveillance systems. Additionally, YOLO v5 introduces improvements in accuracy, thanks to optimizations in model architecture and training procedures. It also offers better scalability and flexibility, allowing for customization and adaptation to various object detection tasks with different complexity levels. Furthermore, YOLO v5 supports a wide range of image sizes, enabling it to maintain high performance across different resolutions, which is particularly beneficial for applications with varying input image quality.

V. Training & Results

In the training setup provided for YOLOv5, the model is configured to train for 20 epochs with a batch size of 32. The initial learning rate is set to 0.01, which is a critical parameter influencing how quickly the model learns from the training data. Beyond these fundamental settings, other important hyperparameters include the momentum at 0.937 and weight decay at 0.0005, both of which play significant roles in the optimization process. Parameters such as warmup epochs and various loss gain settings like box, cls, and obj loss gains are also specified, though these are set to their default values to ensure a balanced approach to learning object detection.

The training results displayed indicate a generally successful learning process. Specifically our mAP (mean Average Precision) at IoU (Intersection over Union) success rate increases up to 85%

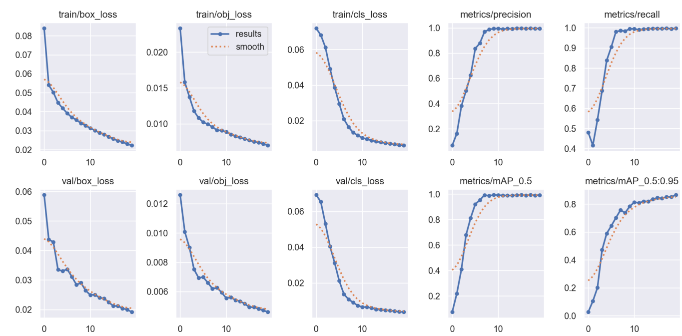

Fig. 3. Performance of the model for different metrics throghout training epochs.

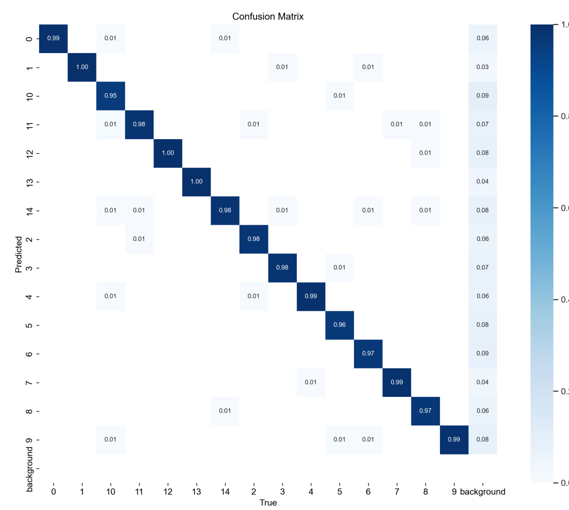

Fig. 4. Confusion matrix of the results obtained by evaluating trained model on the test set.

Similarly, as the epochs progress, there is a consistent decrease in loss for both training and validation sets up to 0.022, which suggests that the model is effectively learning and generalizing well.

Moreover, the precision which is defined as the ratio of correctly identified positive cases to the total predicted as positive, reaches up to more than 95% while this is also the case for recall, that is, the ratio of correctly identified positive cases to the actual total positive cases. However, these results reflect the performance in validation and training data. In order to establish a more reliable and independent result, we have tested the model with images from the test set and discussed the results.

After the training procedure the model is evaluated using the test set. From the obtained results a confusion matrix is created. The occurrence probabilities obtained using the frequencies can be observed from the confusion matrix given in Figure 4.

Based on the data extracted from the confusion matrix, it's evident that the true positive rates for each class, representing the numbers and symbols on the UNO cards, are exceptionally high, falling within the range of 0.98 to 0.99. This remark-

Fig. 5. Models detection and classification performance on random data selected from the validation set created for training

able performance metric highlights the deep learning model's profound ability to accurately detect and classify the various elements on the UNO cards. The high true positive rates indicate that the model has successfully learned the nuanced differences between each number and symbol, managing to distinguish them with a high degree of accuracy.

The significance of these true positive rates cannot be overstated, as they serve as a clear indicator of the model's effectiveness in recognizing and categorizing each card's unique features. In the context of machine learning, especially within the realm of image classification and pattern recognition, achieving such high true positive rates is a noteworthy accomplishment. It suggests that the model is not only adept at identifying the presence of specific numbers and symbols but also at minimizing the instances of false negatives and false positives.

Some detection and classification examples of the model on the validation set can be observed from Figure5 that are chosen randomly from the validation set during training.

VI. Conclusion and Future Work

So far, we have trained a YOLOv5 model with specific hyperparameters, and our results, as reported in the previous section, indicate success. We have made significant progress. Our next step is to investigate hyperparameter search methods and apply them to our model to identify optimal hyperparameters for our training.

References

[1] "Ultralytics YOLOv5 Architecture," 2024. [Online]. Available: https://docs.ultralytics.com/tr/yolov5/

[2] A. Crawshaw, "Uno cards dataset." [Online]. Available: https://public.roboflow.com/object-detection/uno-cards

[3] G. Jocher, "Comprehensive Guide to Ultralytics YOLOv5." [Online]. Available: https://github.com/ultralytics/ultralytics/blob/main/docs/en/yolov5/tutorials/architecture_description.md