Machine Learning Final Project Proposal: Understanding and Extending YOLOX

YUZHOU QIN, SUSTech
SICHENG ZHOU, SUSTech

ACM Reference Format:
Yuzhou Qin and Sicheng Zhou. 2023. Machine Learning Final Project Proposal: Understanding and Extending YOLOX. 1, 1 (December 2023), 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 RESEARCH PROBLEM

1.1 Introduction to Object Detection and Tracking

Object detection and tracking are fundamental tasks in computer vision that play a crucial role in enabling machines to interpret and interact with the visual world. These tasks involve the identification and monitoring of objects within an image or a sequence of images, providing essential information for a myriad of applications ranging from autonomous vehicles and surveillance systems to augmented reality and human-computer interaction.

Object detection refers to the process of locating and classifying objects within an image, often bounding them with a rectangular box and assigning a corresponding label. Complementary to object detection, object tracking involves the continuous monitoring of identified objects across multiple frames or in a video stream. The objective is to maintain the identity of objects as they move and undergo changes in appearance, scale, or orientation.

The challenges in object detection and tracking are diverse, encompassing issues such as handling occlusions, variations in lighting conditions, and real-time processing requirements. As computer vision technology advances, researchers continually seek innovative approaches to improve the accuracy, efficiency, and robustness of these tasks.

1.2 Introduction to YOLOX

You Only Look Once X (YOLOX) stands as a significant milestone in the evolution of object detection algorithms within the field of computer vision.

Traditional object detection methods often face trade-offs between accuracy and speed, limiting their applicability in real-time scenarios. YOLOX adopts an "extreme" approach by focusing on a single detection level, streamlining the detection process for efficiency without compromising precision. This unique design allows YOLOX to achieve impressive real-time processing capabilities, making it well-suited for applications such as video analysis, autonomous vehicles, and interactive systems.

One of the distinguishing features of YOLOX is its anchor-free design, which eliminates the need for predefined anchor boxes. This design choice contributes to its adaptability and robustness across various object scales and aspect ratios. Additionally, YOLOX introduces the concept of a decoupled head, separating object classification and bounding box regression, facilitating more flexible model architecture customization.

YOLOX has gained attention for its real-time processing capabilities and competitive accuracy, making it a promising candidate for various applications.

2 RESEARCH GOALS AND OBJECTIVES

The overarching goal of this project is to comprehensively understand and extend the capabilities of the YOLOX object detection and tracking framework.

In this project, we will conduct a literature review on object detection and tracking methods, emphasizing the evolution of YOLOX and its comparative advantages over existing techniques. We will thoroughly study the research paper and codebase of YOLOX to gain a comprehensive understanding of its architecture, design philosophy, and implementation details. Then we will extend the applicability of YOLOX by training and testing the model on a different autonomous-driving-related dataset, evaluating the model's generalization capabilities across diverse datasets.

3 RELATED WORK

3.1 Object Detection

Object detection is a fundamental task in computer vision, playing a crucial role in various applications such as autonomous driving, surveillance, and human-computer interaction. Over the years, numerous methods have been proposed to address the challenges of accurately localizing and classifying objects within an image.

Classic approaches such as R-CNN (Region-based Convolutional Neural Network) and its variants, including Fast R-CNN and Faster R-CNN, laid the foundation for modern object detection techniques. These methods introduced the concept of region proposal networks and demonstrated significant improvements in both accuracy and efficiency.

The evolution of object detection continued with the advent of single-shot detectors (SSD) and You Only Look Once (YOLO) series. YOLO, in particular, introduced a real-time object detection paradigm by dividing the image into a grid and predicting bounding boxes and class probabilities directly. The trade-off between speed and accuracy became a central theme, leading to the development of YOLO variants like YOLOv2, YOLOv3, and the latest YOLOv4.

Authors' addresses: Yuzhou Qin, SUSTech, qinyz2021@mail.sustech.edu.cn; Sicheng Zhou, SUSTech, zhousc2021@mail.sustech.edu.cn.

XXXX-XXXX/2023/12-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

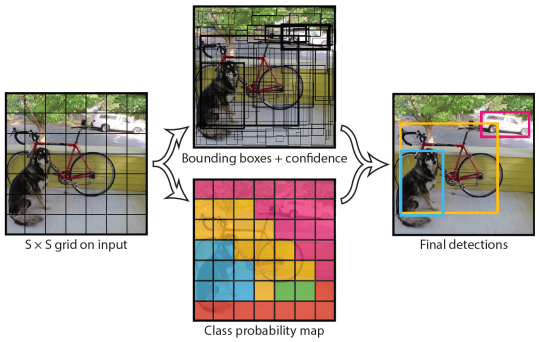

Fig. 1. YOLO Detection Process

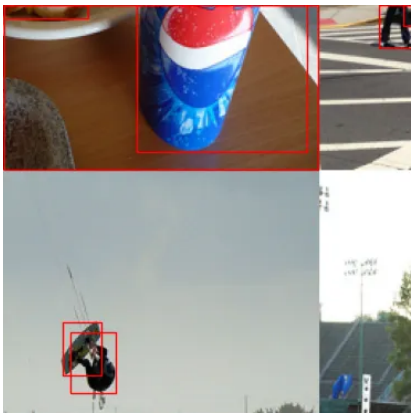

Fig. 2. Final Mosaic Image

3.2 Object Tracking

Object tracking extends the capabilities of object detection by enabling the continuous monitoring of objects across consecutive frames. Various tracking algorithms have been proposed to address the challenges of occlusion, scale variations, and appearance changes.

Traditional tracking methods include correlation filters, such as MOSSE (Minimum Output Sum of Squared Error), and Kalman filters, which maintain a dynamic state estimate for each object. More recently, deep learning-based trackers, such as GOTURN (Generic Object Tracking Using Regression Networks) and MDNet (Multi-Domain Network), have demonstrated superior performance in handling complex scenarios.

4 RESEARCH DESIGN AND METHODS

4.1 Literature Review

This section is the literation review we conducted on YOLOX. It should be noted that YOLOX is obtained by improving YOLOv3 as the baseline, therefore before take a deeper look into YOLOX, we need to investigate the methods used in YOLOv3

4.1.1 Overview. YOLO's CNN splits the input image into grids, and each grid predicts the bounding box and category. The overall process is shown in the figure 1.

In bounding box prediction, each grid should not only predict a bounding box position, but a corresponding confidence score as well. The bounding box is denoted by (x, y, w, h). The accuracy of the bounding box can be characterized by the intersection over union (IOU) of the predicted box and the ground truth. In classification, each grid has to predict the probability of which class the bounding box belongs to.

4.1.2 Data Augmentation. YOLOX uses two data augmentation methods: Mosaic and MixUp. Mosaic is to take 4 images and combine them into a single image, which is done by resizing each of the four images, stitching them together, and then taking a random cutout of the stitched images. MixUp averages two images together based on a weighting parameter λ:

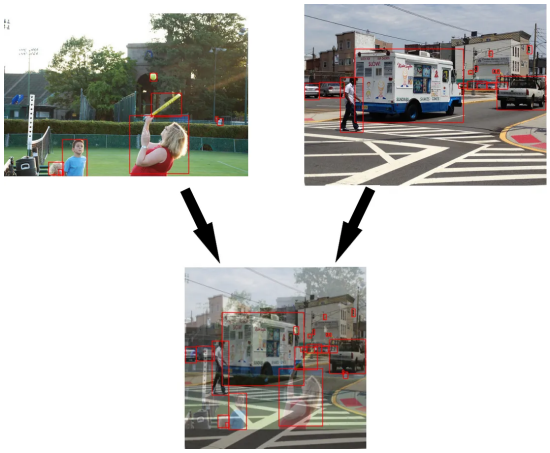

Fig. 3. MixUp

$$\tilde{x} = \lambda x_i + (1-\lambda)x_j, \quad \text{where } x_i, x_j \text{ are raw input vectors} \tag{1}$$

$$\tilde{y} = \lambda y_i + (1-\lambda)y_j, \quad \text{where } y_i, y_j \text{ are one-hot label} \tag{2}$$

Figure 2 and 3 are examples of those two augmentation methods. But the MixUp and Mosaic methods are closed for the last 15 epochs in YOLOX model, probably due to the fear of excessive data augmentation. Additionally, after using strong data augmentation, it is found that ImageNet pre-training is no more beneficial, so they train all the models from scratch.

4.1.3 From Sliding Window to Anchor-free. Sliding window is the most primitive object detection method. It means that given a fixed size window, it slides step by step from left to right and from top to bottom according to the set pace, and input each window into a convolutional neural network for prediction and classification. This method has two disadvantages. First, the window size is fixed, so it is not suitable for objects with large deformation. Second, there are many windows and large amount of computation.

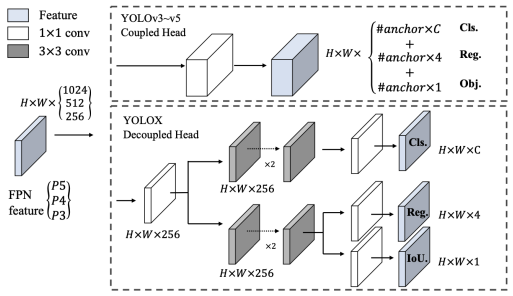

Fig. 4. Comparison between Coupled Head and Decoupled Head

Region proposal is the core idea in R-CNN family. RPN(Regional Proposal Network) is not responsible for image classification, it is only responsible for selecting candidate regions in the image that may belong to one class of the data set, and then the candidate regions generated by RPN are input into the classification network for final classification. However, this method still has the problem that a window can only detect a single object and cannot solve the multi-size problem.

The Anchor Box is defined by the aspect ratio of the border and the area of the border. According to the Anchor Box, a series of bounding boxes can be generated at any position of the image. YOLOv3 uses K-means clustering to learn different anchors from the training set. There are also some problems with anchor-based methods. Firstly, anchors are obtained by performing cluster analysis before training, which are domain-specific and less generic. Secondly, the anchor mechanism increases the complexity of the detection head, as well as the number of predictions per image.

YOLOX adopts the center-based anchor-free method, which uses the center point or the center target region to define the positive sample, and then predicts its distance to the four edges of the target. It assigns the center position of each object as a positive sample and predefines a scale range to specify the FPN level of each object.

4.1.4 From Coupled Head to Decoupled Head. The coupled head usually feeds the feature map output from the convolutional layer directly into several fully connected layers or convolutional layers to generate the output of the target location and category. YOLOv3 used coupled head. For the detection task of coco80, each anchor will generate a prediction result of $h \times w \times 85$ dimensions, of which obj (distinguish between foreground and background) occupies 1 channel, reg (coordinate) occupies 4 channels, and cls (predict which class among 80 classes) occupies 80 channels.

The decoupled head extracts the target position and category information separately, learns them separately through different network branches, and finally fuses them. YOLOX first uses 1×1 convolution to unify the original feature maps with different channel numbers to 256, and then uses two parallel branches, each of which uses 3×3 convolution.

The comparison between coupled head and decoupled head can be seen in Figure 4.

4.2 Code

We download the code from YOLOX repo on Github, a popular YOLOX open source repository which already have 8.7k stars. After installing the dependent packages following the instruction, we can either download the pretrained model and run the demo, or do the reproduction.

4.2.1 Demo. This repository provides an pretrained demo. After entering the program from the main function, the command line arguments are first parsed, then the model and weights are loaded, and then the image_demo or imageflow_demo function is called for object detection based on the input image or video path.

The image_demo function is used to work with a single image or an image directory. It first takes a list of images and then performs inference and visualization on each image. The imageflow_demo function is used to process a video or camera stream, and differs from processing a single image in that it should inference and visualize each image frame.

The inference function is the core of the model inference. It takes an image input, makes a prediction through the model, and returns the prediction and the image information. If the input is an image path, it will first read the image and then get the height and width of the image, as well as the original image. It then preprocesses and converts the image into a tensor, which is then passed through the model to make a prediction.

The visual function is used to visualize the predictions. It receives the prediction results and image information, and then draws the predicted bounding boxes and category labels on the original image.

4.2.2 Train on COCO Dataset. It provides YOLOX training code, allowing users to do train their own model on COCO dataset.

The main function configures the environment for distributed training, and creates a YOLOX trainer using the experiment configuration.

The trainer first performs some initialization, including the initialization of the model, optimizer, data loader, and the recovery of some training states. The optimizer is initialized and the resume_train method is called to check whether the specified checkpoint file is present. The no_aug variable is then set depending on whether the training phase without using data augmentation has been reached, and the data loader is obtained. It then initializes a data prefetcher, which speeds up data loading. Next, it gets the number of iterations per training cycle and gets the learning rate scheduler. If the occupy parameter is set, the function calls the occupy_mem method to occupy some memory. If distributed training is used, the model is wrapped as a DDP object. If the exponential moving average of the model is used, a ModelEMA object is created and the number of updates is set.

The model performs the following steps in each iteration: preprocessing of the data, forward propagation of the model, back propagation of the loss, parameter update of the optimizer, update of the EMA model, update of the learning rate, and update of the measurer. First, take the next batch of input and target data from the prefetcher and transfer them to the specified data type. Next, it turns off the gradient computation on the target data and then preprocesses the input data and uses the context manager to perform the

Classification	Version
CPU	Intel(R) Xeon(R) Gold 5320 CPU @ 2.20GHz
GPU	A100
Memory	125G
GPU Driver	525.125.06
CUDA	12.0.

Table 1. Hardware and Software Configurations

forward propagation of the model, which enables mixed-precision training. It then takes the total loss from the output of the model. After that, the gradient of the optimizer is then cleared, and the gradient scaler scales and backpropagates the gradient of the loss. If the exponential moving average of the model is used, the model updates the EMA model and updates the learning rate of the learning rate scheduler.

4.2.3 Evaluation. The evaluate method is also provided to evaluate the performance of the model in terms of inference time, NMS time, and average precision. The evaluation function transforms the data in the data loader to the specified tensor type and then uses the model for inference. As we iterate over the data, we also record the inference time and non-maximum suppression (NMS) time and store these times in a tensor.

4.3 Improvement

YOLOX, as an object detection algorithm, has some problems and challenges in its design and implementation which we can improve.

First, YOLOX uses an Anchor-free design, which means that it directly predicts the bounding boxes of objects instead of using predefined anchors like some other object detection algorithms. However, its disadvantage is that it is not suitable for general object detection, it is suitable for multi-scale object detection, small object detection, etc., and its accuracy is lower than that of the anchor-based algorithm.

Secondly, the design of YOLOX includes elements such as decoupling head, data augmentation, Anchor Free, and sample matching. These designs may increase the complexity of the model and make the model more difficult to understand and implement. Moreover, these designs may also affect the performance of the model as they may introduce additional computational overhead and potential errors.

Finally, the performance improvement of YOLOX is inextricably linked to sample matching. This may mean that the performance of YOLOX may suffer if the method of sample matching is not sufficiently optimized. Therefore, understanding and optimizing methods for sample matching is key to understanding and improving YOLOX's performance.

5 INITIAL RESULTS

At this stage, we have deployed our environment on the cluster in HPC Lab. The hardware and software configuration is given in table 1.

We have successfully run the inference part of the model, and we are now reproducing the training part. However, in the original configuration of the training part, 300 epochs need to be trained, and each epoch needs to be trained for 14786 iterations, which takes a long time. As of Dec 8^{th}, our model has been trained for 103 epochs.

6 STAFFING PLAN

Yuzhou Qin. Responsible for conducting the literature review, summarizing key insights, and understanding the YOLOX codebase.

Sicheng Zhou. Collaborates with the lead researcher in understanding the YOLOX codebase and contributing to the reproduction efforts. Takes an active role in preparing the autonomous-driving-related dataset for testing.

7 TIMELINE

Week 1-2: Background Research and Literature Review. Conduct an in-depth literature review on YOLOX, focusing on the original research paper, relevant conference proceedings, and open-source contributions. Summarize key insights, strengths, and weaknesses identified from the literature.

Week 3-4: Code Understanding and Reproduction. Thoroughly study the YOLOX codebase, gaining a clear understanding of its architecture, key components, and implementation details. Begin the process of code reproduction or successful code execution on the original dataset used in the YOLOX research paper.

Week 5: Dataset Preparation and Initial Testing. Prepare an autonomous-driving-related dataset for testing YOLOX, ensuring compatibility with the framework. Train and test the YOLOX model on the selected dataset, using the reproduction code.

Week 6: Evaluation and Final Report. Evaluate the YOLOX model's performance based on benchmarking metrics, including accuracy, precision, and recall. Compile final results and insights into a comprehensive report. Discuss any challenges encountered during the reproduction and testing process.