Electrotechnics research project

Real-Time Detection of circuit components using YOLOv4 model

Software and Automation Engineering

2023-2024

Authors:	*Id :*
Alessio Pelusi	1109448
Lorenzo Serloni	1109161

Course Coordinator:

Prof. Simone Fiori

27th December 2023

Contents

1 Abstract

In this research, we explore the application of YOLOv4, a prominent deep learning methodology, for real-time detection and classification of electrical circuit components. Traditionally, circuit components, positioned variably in scanned images prevalent in electrical and electronics engineering, were treated individually using conventional methods. However, to enhance efficiency, we leverage the You Only Look Once (YOLOv4) approach for object detection. YOLOv4, designed for rapid object detection and classification, minimizes the need for extensive image data preprocessing and automates the feature extraction phase. Our objective is to achieve precise detection and classification of three distinct circuit components—resistor, inductor and capacitor—within circuit images. The implementation is carried out using the MATLAB programming language. To train our model, we curated a dataset comprising 400 circuit components images. Fine-tuning and adjustments were applied to a pre-trained YOLOv4 model, based on the YOLOv4 architecture, to align with project specifications. The training process spanned 90 epochs, and the model's efficacy was evaluated on diverse circuit components positions. The trained YOLOv4 model exhibited swift and high-performance detection of circuit components in real-time scenarios. Our proposed method showcases efficiency and effectiveness for real-time detection applications in electrical circuit analysis.

Keywords: *Electrical circuit components, Deep learning, YOLOv*4*, Object detection*.

2 Introduction

Object detection in images has wide applications across various fields, such as industrial inspection, autonomous driving, and medical imaging. In recent years, deep learning-based methods have achieved significant success in object detection tasks. Among these, the You Only Look Once (YOLO) algorithm has garnered increasing attention due to its real-time speed and competitive detection accuracy. The YOLO algorithm stands out as a state-of-the-art, single-stage object detector. Unlike two-stage detectors like Faster R-CNN, YOLO treats object detection as a regression problem, predicting spatially separated bounding boxes and associated class probabilities. This end-to-end modeling approach allows for extremely fast prediction speeds compared to previous methods. YOLOv3 and YOLOv4 have further improved the structure, achieving better accuracy and faster speeds. Notably, YOLOv4 continues these innovations and attains state-of-the-art results on COCO while being up to 3x faster than YOLOv3. The COCO dataset is a large-scale object detection, segmentation, and captioning dataset consisting of over 330,000 images containing 80 object categories. It is commonly used to evaluate the accuracy of object detection models. In this project, we applied the YOLOv4 tiny model pretrained with the COCO dataset for real-time detection of three main electrical components (capacitors, resistors, and inductors) from camera images. We initiated the project by collecting images of these components to create a robust dataset for training the network through transfer learning. The choice of the YOLOv4 tiny version was driven by its exceptional speed in inference, making it well-suited for real-time detection applications. Additionally, its compatibility with our limited computational capacity made it an ideal choice for the fine-tuning process. This project aims to develop a solution based on YOLOv4 for real-time recognition and classification of real electrical components. A dataset will be created using images of capacitors, resistors, and inductors. The YOLOv4 model, after adequate pre-training and adaptation, will undergo a training process. Finally, the model will be tested in real-time on various components to verify its reliability.

Possible real life application

Obviously, this type of detection has great potential to be deployed on edge devices such as smart cameras, enabling automation in industrial inspection processes. In industrial applications, object detection is commonly used during various quality control and manufacturing tasks

that involve visual inspection of components or products on assembly lines. Traditionally, such inspections are performed manually by human operators which can be an inefficient and labor-intensive process prone to errors. However, deploying a real-time object detection model like YOLOv4 on edge devices embedded within the manufacturing environment allows for automated visual inspections at scale. Smart cameras equipped with a trained YOLOv4 model could inspect electronics, detect defects, and even classify failure modes all in real-time as items move down the production line. Any defects could then trigger alerts to halt the line or remove faulty items, minimizing waste while also enhancing productivity. The high speeds of YOLOv4 make it well-suited for such time-critical industrial applications compared to other computer vision approaches. In the domain of electrical and electronic circuit design, the initial phase involves arranging physical components within the circuit, followed by a series of analyses. The arrangement of components can be done manually or through specific software. While the manual approach is considered traditional, it is often the preferred method initially. During circuit design, engineers make crucial decisions about the components, creating a schematic on paper and assigning numerical values to each element. Subsequently, mathematical calculations based on the circuit's objectives in the completed schematic are performed. If the results align with expectations, the circuit is prepared for testing, either through simulation programs or in a real laboratory setting. However, manually recreating a circuit on computer drawing software can be a laborious process, especially with complex circuits. The acquisition of an image of the circuit, automatic recognition of its components, and direct transfer to simulation programs through an interface or software offer a significant advantage in terms of time savings and workload alleviation. This simplifies component management during the design and testing process.

3 Material and Method

3.1 Data set

As the initial step of the study, a dataset containing 400 electrical circuit components was prepared. The dataset was created by combining a pre-existing dataset with additional images to achieve a sufficiently large number. Scanned images of 400 electrical components were collected to form a dataset for use in the study. Due to the high dimensions of the scanned images, they were initially resized and reduced. Three different components were defined to be detected in the circuit images: the resistor, inductor, and capacitor. **Figure 1** includes sample images of the circuit components in the dataset. As the study aims to identify circuit components in images, in contrast to classification studies, the localization of each circuit component is required. For this reason, the circuit components in the dataset images are labeled using the ImageLabeler program, a graphical image annotation tool available in MATLAB. **Figure 2** illustrates the labeling process of all components in the dataset images using the ImageLabeler program.

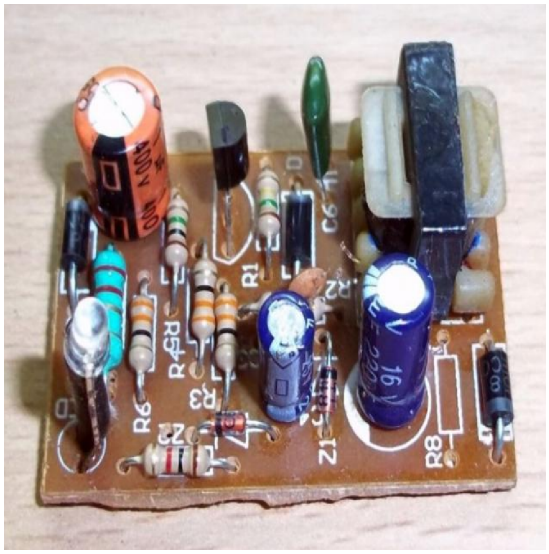


Figure 1

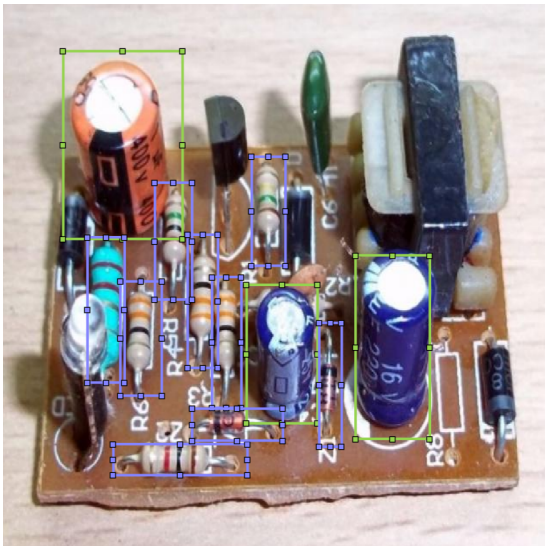


Figure 2

3.2 Deep Learning and Object Detection Networks

Deep learning is a newer field compared to artificial intelligence and machine learning. Unlike classical learning methods, deep learning is closer to the human brain as it consists of multi-layered neural networks **Figure 3**. Since humans are creatures in nature that excel at learning

and applying knowledge, recent studies focus on developing innovative learning algorithms similar to those used by the human brain. Considering the mentioned reasons, the deep learning method is often preferred for studies on object detection and image processing by researchers. Due to the widespread use of deep learning in different areas, various deep learning models specific to the field have been developed. Each of the developed models differs from each other and is used to solve problems in different areas. It is a great advantage to use deep learning approaches to classify images according to categories and detect objects in the image, as it exhibits a high-performance feature extraction ability. At the beginning of 2012, the convolutional neural network (CNN) architecture, a useful deep learning model in visual recognition, was developed for use in specific fields. CNN architectures are mostly preferred in the education and classification stages of image processing studies. While a basic CNN model is usually designed using convolution layers, activation function, ReLU layer, pooling layer, and fully connected layers as illustrated in **Figure 4**, YOLOv4 adopts a different approach. YOLOv4, short for "You Only Look Once" version 4, is a real-time object detection model that stands out for its fundamental architecture. YOLOv4 uses a single model that performs detection and classification in a single inference, making it more efficient for real-time applications. As YOLOv4 has been widely used in various areas, several models based on YOLOv4 have been developed, each with specific features. These YOLOv4 models differ in parameters such as the number of layers used, the order of the layers, and the learning rate, providing greater flexibility for specific applications. Furthermore, some users have customized YOLOv4 models to suit their needs, replacing default models with custom configurations. The main reasons for the increased use of YOLOv4 in many different areas can be explained as follows: it is easy to apply, fast in training and testing stages, and has efficient feature extraction stages occurring automatically through layers.

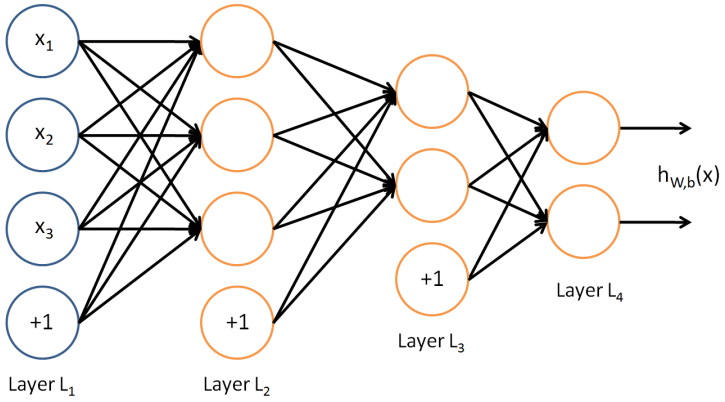


Figure 3

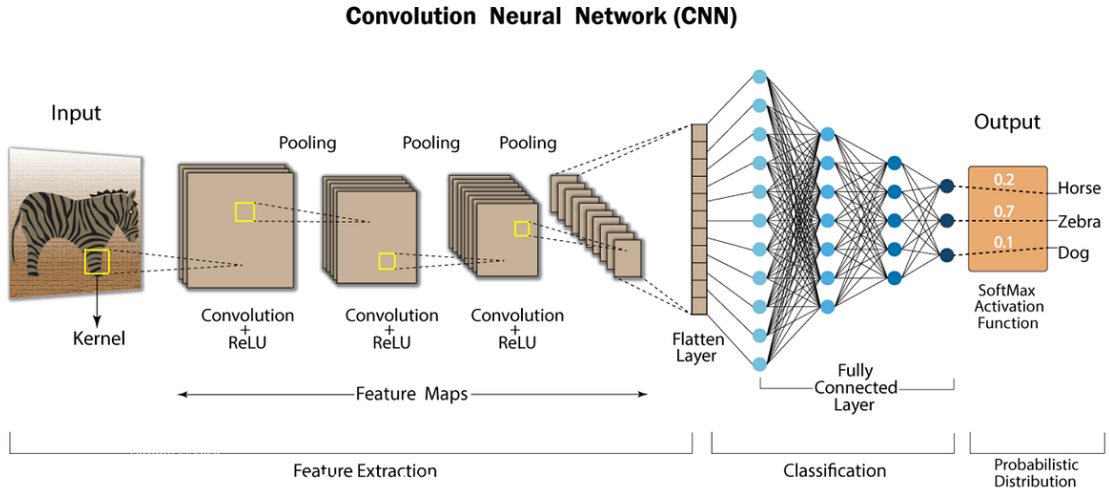


Figure 4

4 Proposed YOLO Method

4.1 How YOLO Technology Works

YOLO (You Only Look Once) is a model based on convolutional neural networks, in which the problems of object localization and classification in an image are treated for the first time as a single regression problem. This change in perspective introduced by the authors of YOLO [1](Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi) paved the way for a new approach to object detection, facilitating the subsequent development of increasingly accurate and high-performing models. YOLO is the first "one-stage" object detection system. It provides an innovative solution to the object detection problem by unifying the different components of object detection systems into a single network. This allows the network to reason simultaneously over the entire image rather than specific regions, ensuring a better understanding of the environment in which different object classes are located. Given S and B, the model divides the image into an $S \times S$ grid of cells. The grid cell containing the center of an object is defined as the cell responsible for it. Each grid cell predicts B bounding boxes and assigns a score called "Confidence Score" to each of them. This score indicates the probability that the bounding box contains an object and how accurate the dimensions and location of the bounding box are believed to be relative to the object. Regardless of the number B of generated bounding boxes, each grid cell predicts C classes with their respective conditional probabilities. At this point, for each bounding box, the model multiplies the conditional class probabilities (from the cell) by the confidence score of the bounding box. This yields a specific confidence score for each class for each bounding box. The resulting value encodes both the probability that an object of a given class appears in the examined bounding box and the accuracy with which the predicted bounding box delineates the spatial boundaries of the object.

4.2 YOLOv4: The Key Features

4.2.1 Backbone Network

The backbone network for an object detector is typically pretrained on ImageNet classification, implying that its weights are already adapted to discern pertinent features in an image. However, these weights are subsequently fine-tuned for the specific task of object detection. In our

[1]YOLO paper: https://arxiv.org/abs/1506.02640

discussion, we will assess the merits of the following backbones for the YOLOv4 object detector, as illustrated in **Figure 5**:

- CSPResNext50
- CSPDarknet53
- EfficientNet-B3

Table 1: Parameters of neural networks for image classification.

Backbone model	Input network resolution	Receptive field size	Parameters	Average size of layer output (WxHxC)	BFLOPs (512x512 network resolution)	FPS (GPU RTX 2070)
CSPResNext50	512x512	425x425	20.6 M	**1058 K**	31 (15.5 FMA)	62
CSPDarknet53	512x512	725x725	**27.6 M**	950 K	52 (26.0 FMA)	**66**
EfficientNet-B3 (ours)	512x512	**1311x1311**	12.0 M	668 K	11 (5.5 FMA)	26

Figure 5 (M in Parameters stands for million and K in Avarage size of layer output stands for kilobytes)

CSPResNext50 and CSPDarknet53 are both rooted in DenseNet, a design conceived to address challenges such as the vanishing gradient problem and to encourage effective feature propagation. Refer to **Figure 6-7** for a detailed exposition.

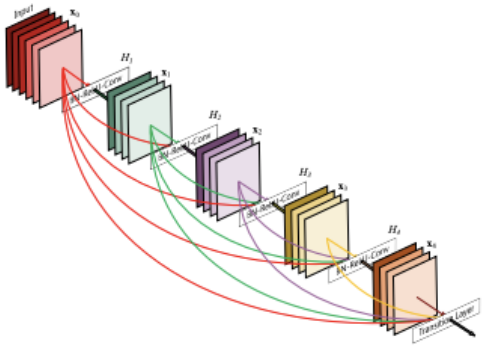
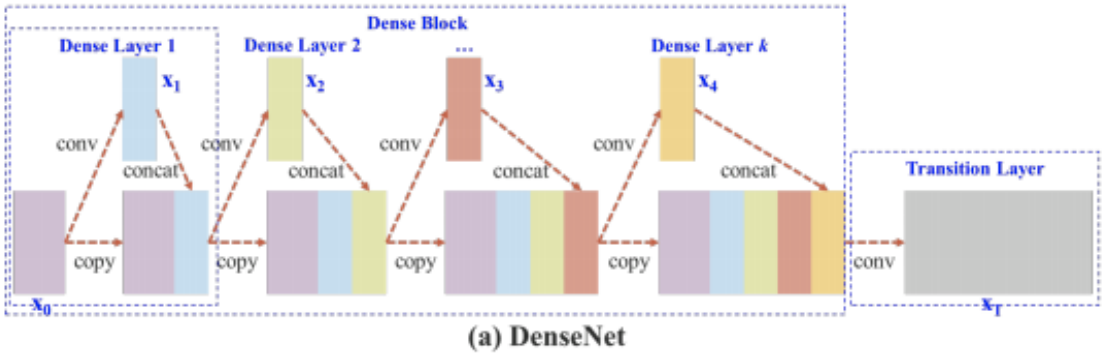


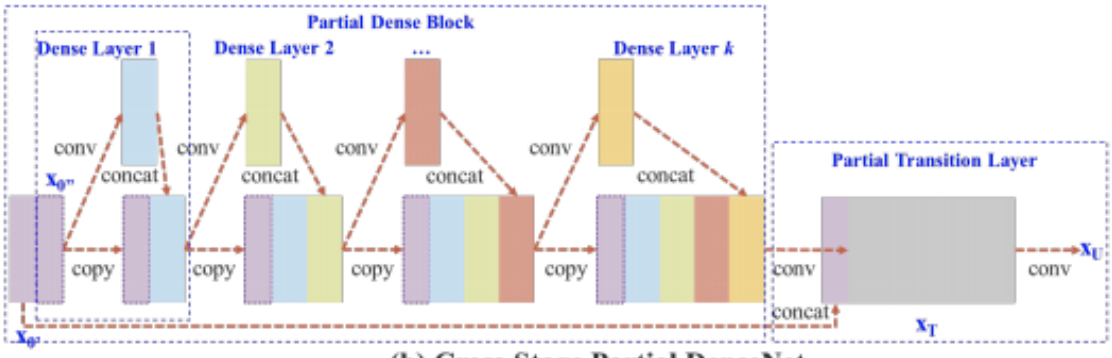
Figure 6



(a) DenseNet

Figure 7

In CSPResNext50 and CSPDarknet53, the DenseNet has undergone modifications to segregate the feature map of the base layer. This involves copying it and directing one copy through the dense block, while the other proceeds directly to the next stage. The objective of CSPResNext50 and CSPDarknet53 is to eliminate computational bottlenecks in DenseNet and enhance learning by transmitting an unedited version of the feature map, as depicted in **Figure 8**.



(b) Cross Stage Partial DenseNet

Figure 8

EfficientNet, a creation of Google Brain, was specifically designed to address the scaling challenges of convolutional neural networks. When scaling up a ConvNet, various decisions come into play, including input size, width scaling, depth scaling, and a combination of these factors. The EfficientNet paper posits the existence of an optimal point for each of these parameters, determined through a systematic search, as elucidated in **Figure 9**.

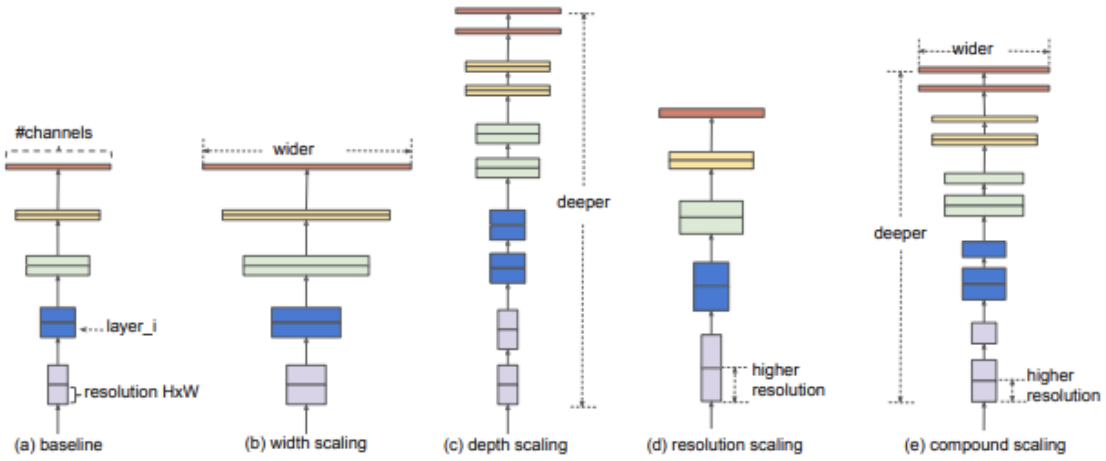


Figure 9

EfficientNet outperforms the other networks of comparable size on image classification. The YOLOv4 authors posit, however, that the other networks may work better in the object detection setting and decide to experiment with all of them.

4.2.2 YOLOv4 Neck

In the progression of object detection, the subsequent step involves amalgamating and interweaving the features generated in the ConvNet backbone in preparation for the detection phase. YOLOv4 contemplates several options for the neck, among which are:

- FPN
- PAN
- NAS-FPN
- BiFPN
- ASFF
- SFAM

These neck components typically traverse up and down among layers, selectively connecting only a subset of layers towards the conclusion of the convolutional network. Each of the P_1 entities mentioned above represents a distinct feature layer within the CSPDarknet53 backbone.

4.2.3 YOLOv4 Head

YOLOv4 utilizes the same YOLO head as YOLOv3 for detection, employing anchor-based detection steps and incorporating three levels of detection granularity **Figure 10-11**.

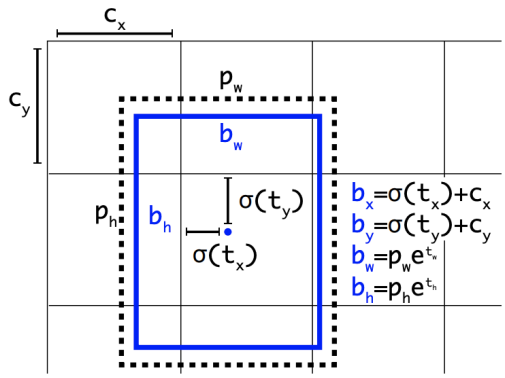


Figure 10

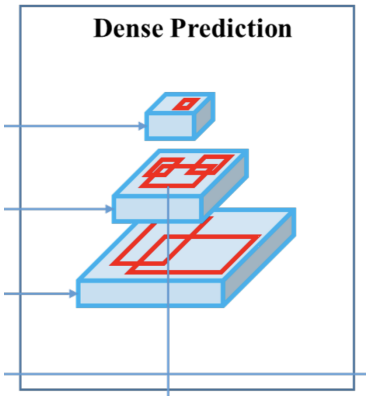
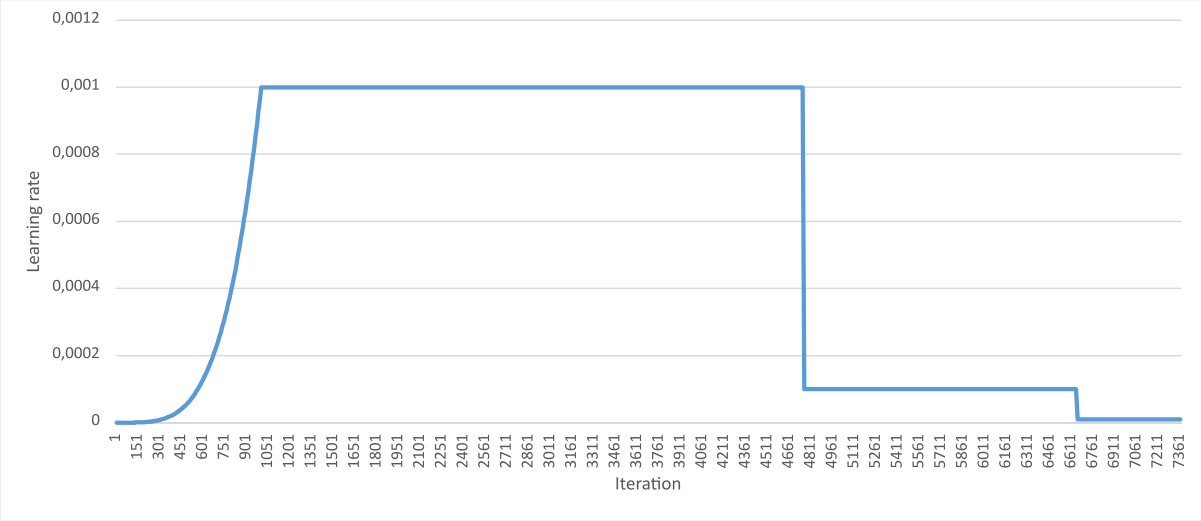


Figure 11

4.3 Training

4.3.1 Learning Rate

During the training of our YOLOv4 model on the electrical component dataset, we adhered to the recommended training process parameters provided by the MATLAB documentation for YOLOv4. This approach involved careful consideration of the learning rate schedule to ensure effective model training. To initiate the training process, we set a small learning rate of 1×10^{-15} for the initial epochs, following the guidance in the MATLAB documentation. Gradually, over the first 13 epochs, we increased the learning rate until it reached 0.001, aligning with the suggested strategy for stabilizing and advancing the learning process. Maintaining a learning rate of 0.001 from epoch 13 to 59 allowed the model to comprehensively explore and grasp the nuances of the electrical component dataset. However, as training progress showed signs of slowing between epochs 59 and 81, we strategically reduced the learning rate to 0.0001. This adjustment aimed to fine-tune the model's parameters, ensuring it captured intricate details. In the final 9 epochs, from 81 to 90, we employed an even smaller learning rate of 0.000001, in line with the MATLAB documentation's recommendation. This deliberate slowing down of the learning process during the concluding phase facilitated meticulous optimization of the model weights before reaching convergence.



4.3.2 Loss function

Regarding the loss components during training of our YOLOv4 model on the electrical component dataset, we closely monitored their behaviors alongside the total loss. The box loss graph showed a high initial value that rapidly decreased through around epoch 50, indicating the model was enhancing its localization abilities early on. Between epochs 50-80, the box loss reduction slowed noticeably before continuing a gradual decline, similar to observations for total loss. The object loss followed a comparable trend, sharply decreasing initially then leveling off from epochs 50-80 before continuing downwards. For class loss, the most fluctuation occurred but it still generally tracked downward. Its temporary spikes coincided with adjustments to the learning rate schedule. By tracking component losses together with total loss, we identified signs of training plateauing around epoch 50, similar to total loss. Accordingly, we lowered the learning rate to spur further optimization, driving all losses downward as seen in subsequent epochs. Thus, jointly analyzing the subdivision of losses provided confirmatory evidence that our scheduled lowering of learning rates successfully guided the YOLOv4 model towards iteratively minimizing errors on individual tasks essential for accurate object detection.

5 Results

In this study, a YOLOv4 model was developed for real-time detection and classification of electrical components (resistors, capacitors, and inductors). The key steps and outcomes are detailed below:

Dataset and Data Augmentation

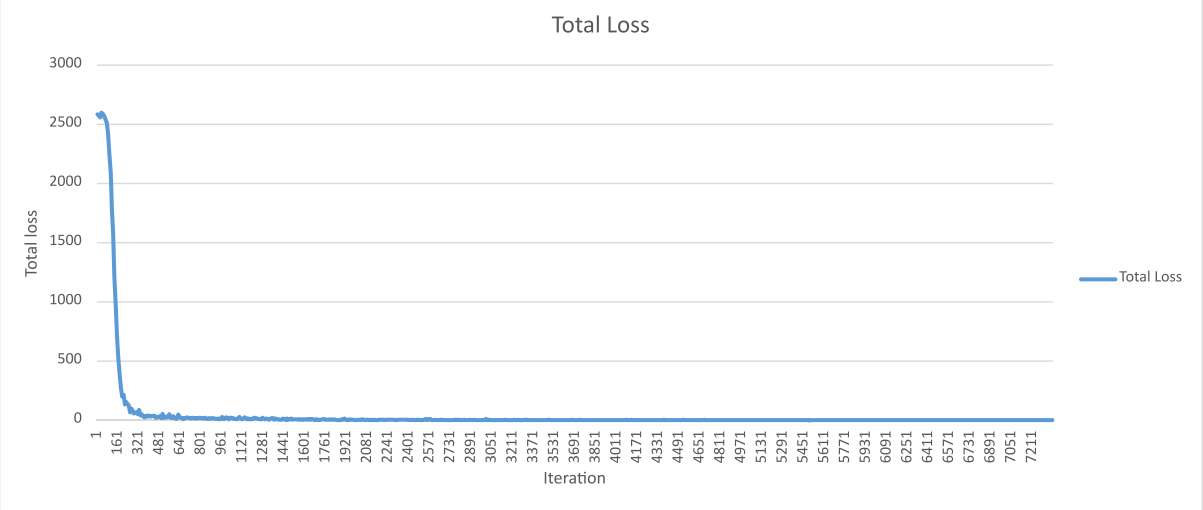
A dataset of 400 images containing capacitors, inductors, and resistors was curated and standardized. To mitigate overfitting, a data augmentation algorithm was employed, enhancing dataset diversity.

Training and Evaluation

The YOLOv4 tiny model underwent 90 epochs of training on the prepared dataset. Model evaluation focused on loss functions, including total loss, classification loss, and localization loss.

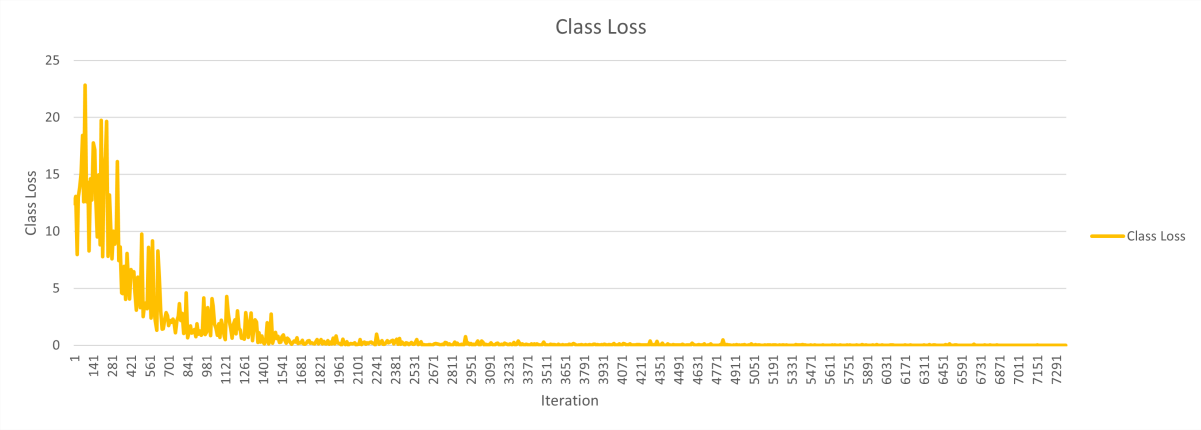
Total Loss

The overall model loss, as depicted in the first graph, reached 0.43339 after 90 epochs. This signifies a low overall loss, indicative of high real-time detection accuracy.



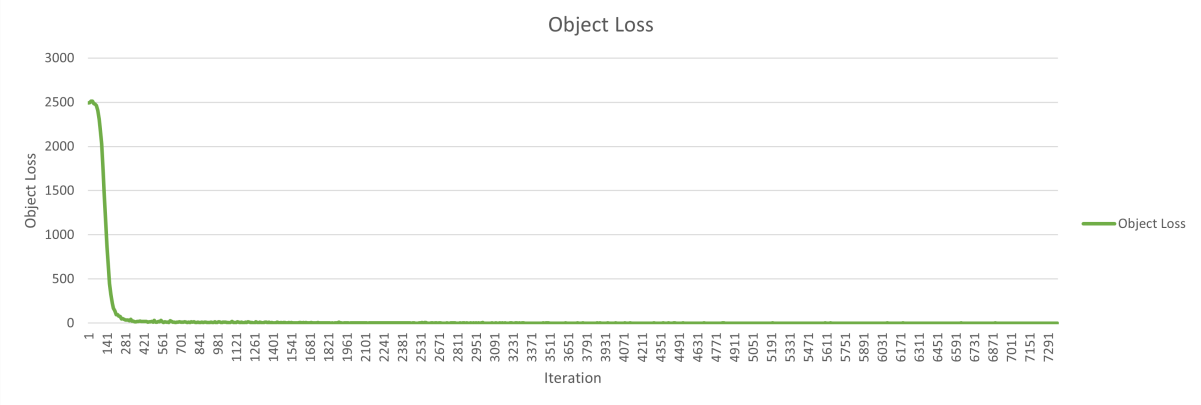
Class Loss

The class loss graph revealed a classification loss of 0.0054296 at the study's conclusion. The model effectively classified detected circuit components.



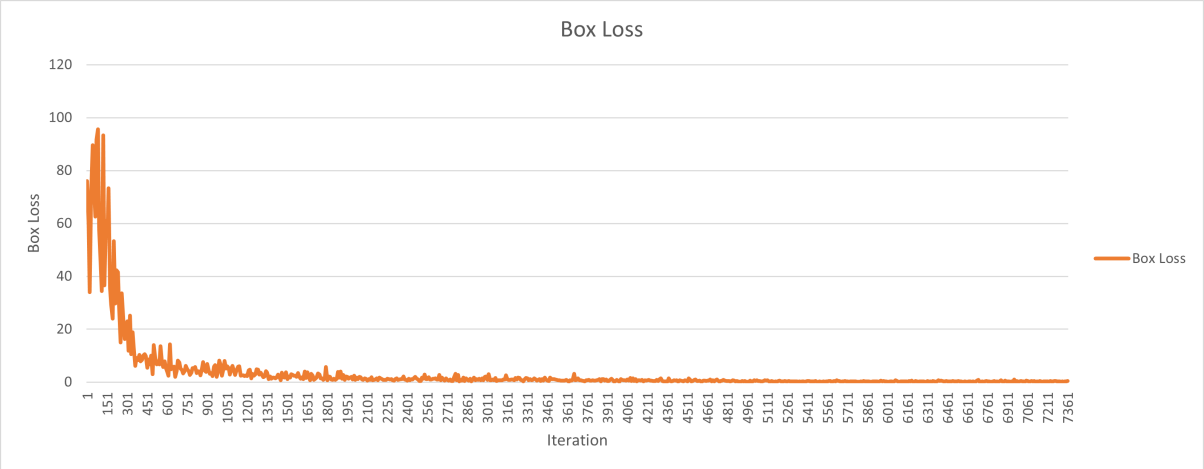
Object Loss

The object loss graph demonstrated a gradual decrease, reaching a minimum of 0.041081 after 90 epochs. This suggests improved accuracy in classifying circuit components.



Box Loss

Similarly, the box loss graph exhibited a gradual decline, reaching a minimum of 0.38688 at the study's conclusion. The model improved in predicting bounding boxes around circuit components.



Real-Time Detection

Figures 12 and 13 showcase real-time detection on frames captured from the webcam. The model accurately identified components, drawing colored rectangles and providing labels of the detected components. The YOLOv4 model displayed promising performance in real-time detection and classification of circuit components. It achieved low total loss, effective classification, and accurate localization in hand-drawn circuit diagrams.

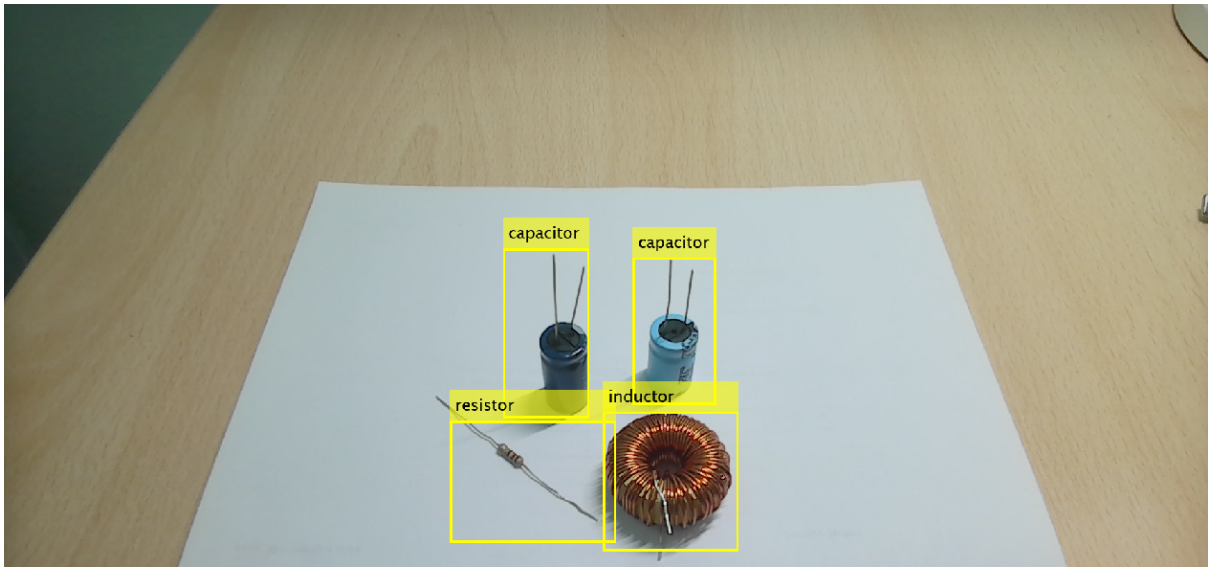


Figure 12

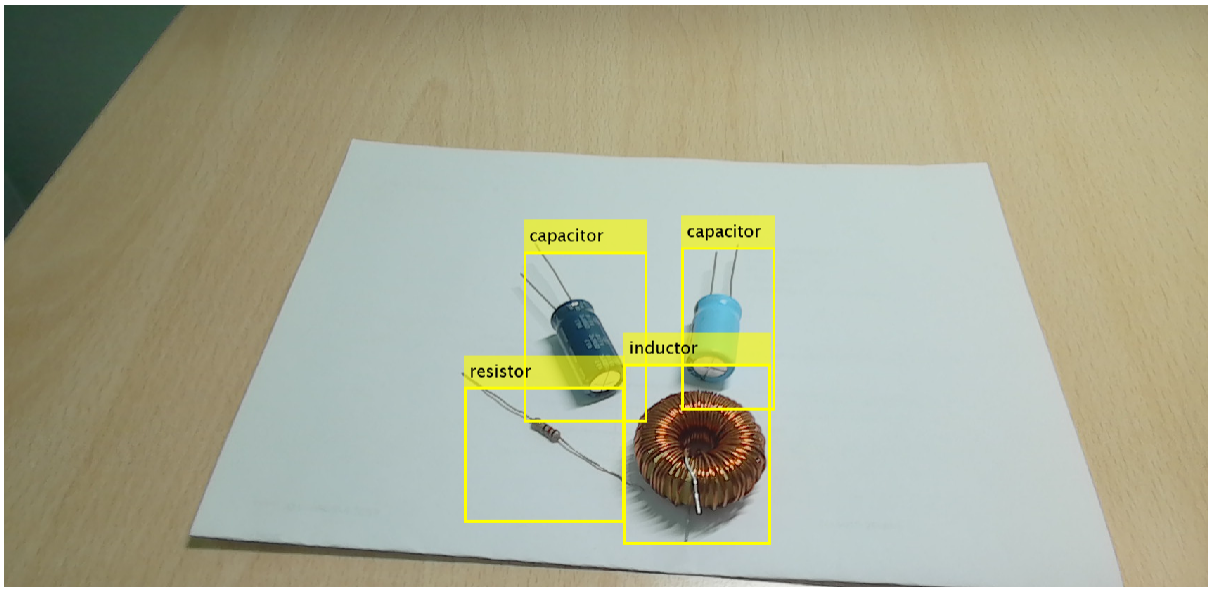


Figure 13

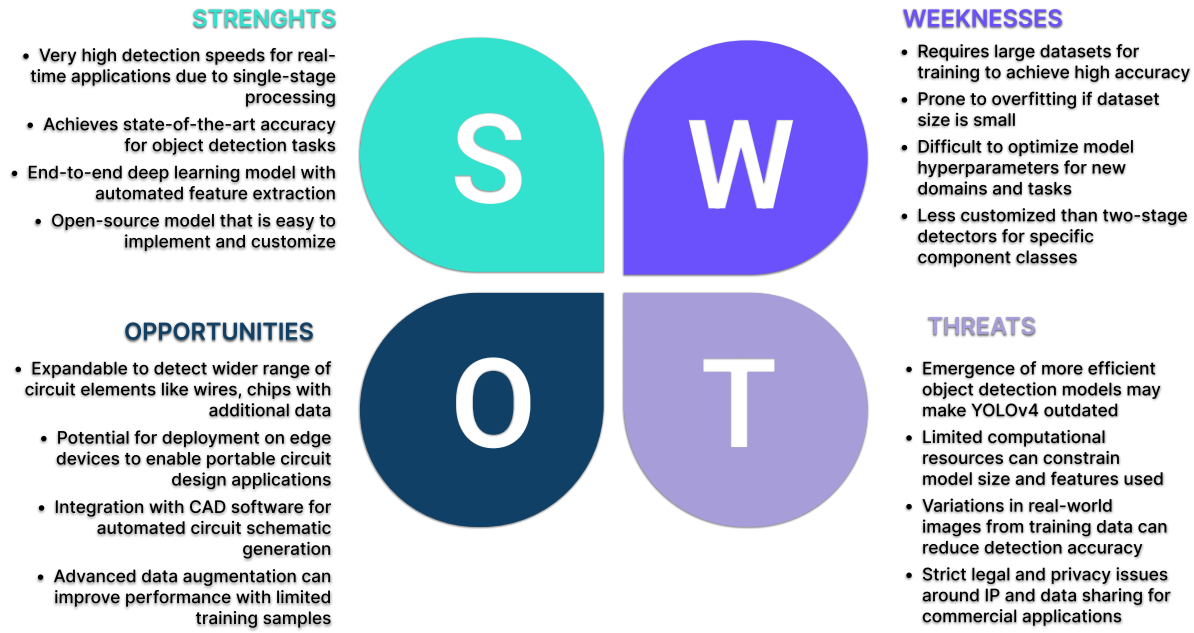
6 Conclusion

This study has successfully tackled the challenging task of fine-tuning the YOLOv4 model for real-time recognition of resistors, capacitors, and inductors in electrical circuits. Through a meticulous fine-tuning process, we have significantly enhanced the model's identification capabilities, achieving robust and precise performance even in dynamic situations.

The use of YOLOv4, with its advanced architecture, has proven crucial in attaining high accuracy in detecting specific components in electrical circuits. The training phase, fueled by an extensive dataset containing resistors, capacitors, and inductors in various configurations, enabled the model to accurately learn the distinctive features of each component.

Implementing a fine-tuning process further optimized the model's capabilities, improving its sensitivity and specificity in real-world scenarios. Analysis of loss functions provided valuable insights into the training progression, contributing to a better understanding of the model's behavior under different conditions.

Our research results demonstrate that YOLOv4, when properly trained and refined, can be successfully employed for real-time recognition applications of electronic components, making a significant contribution to the fields of automation and circuit analysis. This study establishes a solid foundation for future implementations and developments in automatic recognition of electrical components, paving the way for new possibilities in integrating artificial intelligence into electronic engineering.



7 Bibliography Sitography

- https://blog.roboflow.com/a-thorough-breakdown-of-yolov4/
- https://dergipark.org.tr/en/download/article-file/1662507
- https://amslaurea.unibo.it/19637/1/main.pdf
- https://it.mathworks.com/help/vision/ug/object-detection-using-yolov4-deep-learning.html
- https://arxiv.org/abs/1506.02640
- https://it.mathworks.com/help/vision/ug/getting-started-with-yolo-v4.html
- https://github.com/matlab-deep-learning/pretrained-yolo-v4
- https://universe.roboflow.com/mathew-tomy-wip4s/passive-component-detection/browse