L-DETR: A Light-Weight Detector for End-to-End Object Detection With Transformers

TIANYANG LI1,2, JIAN WANG1, AND TIBING ZHANG1

1School of Computer Science and Technology, Northeast Electric Power University, Jilin 132012, China
2Jiangxi New Energy Technology Institute, Xinyu 338004, China

Corresponding author: Tianyang Li (tianyangli1985@gmail.com)

This work was supported by the Scientific Research Funds of Northeast Electric Power University under Grant BSZT07202107.

ABSTRACT

Now most high-performance models are deployed to the cloud, which will not only affect the real-time performance of the model, but also restrict the wide use of the model. How to designing a lightweight detector that can be deployed off-line on non-cloud devices is a promising way to get high-performance for artificial intelligence applications. Hence, this paper proposes a lightweight detector based on PP-LCNet and improved transformer named L-DETR. We redesign the structure of PP-LCNet and use it as the backbone in L-DETR for feature extraction. More-over, we adopt the group normalization in the encoder-and-decoder module and H-sigmoid activation function in the multi-layer perceptron to improve the accuracy of the transformer in L-DETR. The quantity of parameters of our proposed model is 26 percent and 46 percent of the original DETR with backbones of resnet50 and resnet18. The experimental results on several datasets show that this method is superior to the DETR model in object recognition, and the convergence speed is fast. Experimental results on multiple data sets have shown that our proposal has a higher performance than DETR models on object recognition and bounding box detection. The code is available at https://github.com/wangjian123799/L-DETR.git.

INDEX TERMS

Object detection, lightweight, DETR, PP-LCNet.

I. INTRODUCTION

Object detection is the task of detecting instances of objects of a certain class within an image. With the birth of deep learning methods, progress has been made in various fields, such as pedestrian prediction [1], object detection, object tracking and image synthesis [2]. Benefiting from the advance of deep learning, many Convolutional Neural Networks (CNN)-based depth detectors show excellent performance in object detection. With a large scale of parameters, this kind of detector can only be trained and deployed in the cloud environment, which makes it difficult to get high real-time performances with the network latency and bandwidth fluctuations. Deploying offline detectors on edge devices is a promising way to get high-performance for certain artificial intelligence applications that accomplish real-time object detection. However, these detectors may not be capable to accomplish the transfer learning and fine-tuning on edge devices due to computational limitations. Designing a lightweight detector that requires lower resource consumption and gets higher performances will be a key problem for breaking through the barrier to developing offline detectors.

Modern detectors indirectly solve this set prediction task by defining alternative regression and classification problems on a large number of proposals [3], anchors [4], or window centers [5]. The post-processing steps of approximate repeated prediction, the design of the anchor point set, and the heuristic method of assigning object frames to anchor points significantly affect their performance [6]. However, the rationality of artificial calibration of parameters is controversial, which leads to the uncertainty of model performances. Considering that the attention mechanism is a direct prediction of sets, more and more scholars have shifted their research focus to utilizing the attention mechanism to design detectors with high performance. For instance, ACT [7], an adaptive clustering converter, is presented by using the local sensitive hash method to cluster the query features adaptively and employs the essential prototype interaction to
clustering the features near the query key interaction. UL-DETR [8], an unsupervised pre-training method for object detection, is proposed to use the patch feature reconstruction branch related to patch detection to measure the importance of classification and localization in agent tasks. Moreover, the DETR [9] model, by using a transformer, is ingeniously designed to allow end-to-end object detection without manual post-processing. Despite the DETR has shown excellent performances in object detection, it cannot be trained and deployed on edge devices with its large-scale weights.

Many researchers have studied lightweight detectors which have the potential to accomplish real-time object detection on non-cloud devices. For example, the YOLOv3-based model [10] is designed to obtain a high real-time inference speed when detecting objects with some terminal units, but with lower accuracy than other algorithms. Furthermore, a lightweight model based on fast R-CNN [3] is proposed to perform target detection tasks in non-cloud devices. In addition, a lightweight CPU CNN [11] (named PP-LCNet) is presented, which has high performance on multiple tasks. Generally, lightweight detectors mainly have a compromise between efficiency and accuracy. Although above lightweight detectors have an efficient speed of inference, the accuracy is less than satisfactory for certain tasks.

Therefore, improving the accuracy of lightweight detectors is a challenge for real-time object detection on edge devices. Considering that the DETR model has a good detection accuracy and the PP-LCNet is more efficient, combining their advances may design a lightweight detector with more precision, further contributing to real-time object detection. With this concern, this paper presents a CNN-based light-weight model (called the L-DETR), which may achieve offline real-time object detection on edge devices. The innovation content is as follows:

1) We design a new backbone of DETR based on the PP-LCNet to extract data features and greatly reduce the scale of parameters of DETR.

2) We further improve the transformer that is used as a component within the lightweight model. Specifically, considering the accuracy loss caused by the decrease in the number of parameters, we propose a new normalization strategy to improve the transformer module. To further alleviate the loss of accuracy caused by floating-point calculation on edge devices with low computing power, we improve the feedforward network (FFN) of DETR with the H-sigmoid [12] activation function.

The quantity of parameters of our proposed model is 26 percent and 46 percent of the original DETR with the backbone of resnet50 and resnet18. Experimental results have shown that our proposal has higher performance than the DETR model in object recognition and frame detection accuracy.

The rest of this paper is arranged as follows. The following section describes the related work. Section III explains the proposed approach in detail. Section IV provides the experimental results, and finally, a conclusion is drawn in Section V.

II. RELATED PRELIMINARY WORK
A. LIGHTWEIGHT MODEL FOR OBJECT DETECTION
Federated recently, object detection has ushered in a new era of rapid development, and converters based on the self-attention mechanism have been fully used in detectors. Generally, object detection algorithms based on converters are roughly divided into two categories. One is to directly use part or all of the converter as the backbone of feature extraction. For example, Swin transformer [13] and SOTA typically used backbone to extract features, which have high performances. The other uses CNN for feature extraction, followed by a transformer for encoding and decoding. For example, DETR [9], as the first end-to-end model, can be compared with Faster R-CNN [3], and its effect is even better on large objects. Despite both types of methods showing excellent performance, they require a lot of data and many parameters, allowing them to demonstrate superior performance only when deployed on the cloud. With the development of YOLO series technology, the number of network layers is getting larger along with more accuracy. Obviously, the increased accuracy is accompanied by an increasing number of parameters.

Therefore, when a model migrates from a cloud device to a non-cloud device, the most critical question is how to reduce the parameters of the model. Currently, there are many methods to solve this problem, which can be roughly divided into the compressed pre-training network or direct training small networks. The compressed pre-training network is mainly divided into four types: pruning [14], quantification [15], knowledge distillation [16], and low-rank decomposition [17].

Network pruning is an effective method to improve the model reasoning speed and reduce the model size, but pruning may not reduce the actual reasoning speed. Weight quantization technology reduces the storage space of model by lowering the floating-point number of parameters. It is a method of exchanging precision for speed. With the increase of quantization rate, the accuracy will decline. Knowledge distillation transfers knowledge from an extensive trained teacher network to many smaller and faster student models. While existing algorithms are not stable. The low-rank decompositio method mainly uses matrix decomposition to decompose the original convolution kernel in the deep neural network, but it will lead to a remarkable decline inaccuracy. These methods are difficult to carry out secondary training, which is unsuitable for specific application scenarios. To alleviate the above problems, the direct training of small networks used in this paper has been a popular method in recent years. Training small networks directly can significantly avoid the loss of network accuracy and improve the reasoning speed [18]. Moreover, researchers also try to promote the development
of activation functions and normalization in different applications directly to improve the model’s performance.

B. ACTIVATION FUNCTIONS

The activation function is an essential part of the neural network. A suitable activation function can significantly improve the performance of the training model. The Sigmoid [19] activation function can compress the input value to the output range of [0,1] in a great range. Its disadvantages are a large amount of calculation and the problem of gradient disappearance appears. The ReLU [20] activation function can be expressed in sparse numbers for neural networks and can more efficiently reduce the gradient and back-propagation, avoiding the problems of gradient explosion and gradient disappearance. Swish’s [21] activation function has no upper limit but has a lower limit and high smoothness. Compared with ReLU, it has a large amount of calculations and specific requirements for equipment.

C. NORMALIZATION

Since Batch Normalization (BN) [22] was proposed in 2015, it has been widely used in various networks. BN normalizes the features through the mean and variance calculated in the batch to simplify the optimization and make the very deep network converge. However, BN is sensitive to the size of batch sizes. Because the mean and variance calculated each time are on a batch, if batch sizes are too small, the calculated mean and variance are not enough to represent the whole data distribution. When running a very deep network, the requirements for equipment are high.

The calculation of Layer Normalization [23] and Instance Normalization [24] do not depend on the size of the batch. The former applies to the sequence model. Neuron inputs in the same layer have the same mean and variance, and different input samples have other mean and square. The latter is often used to generate the model, and its generation results mainly depend on the instance of an image. Therefore, using instance normalization to normalize H×W can accelerate the convergence of the model and maintain the independence between each image instance.

III. A LIGHT-WEIGHT DETECTOR BASED ON PP-LCNet AND TRANSFORMER

For balancing the efficiency and accuracy, a Light-Weight detector called L-DETR is designed based on DETR and PP-LCNet. It consists of two main parts, which are shown in Fig. 1. One part is the backbone of the model. It is an improved PP-LCNet that is used to extract data features. Compared to the DETR, L-DETR has fewer parameters with the new backbone. The other parts are an improved transformer. It is used to calculate the global information and make the final prediction. Its normalization and FFN are improved, which leads to an increase in the accuracy of frame detection. Next, more details of the backbone, transformer module, and FFN are discussed.

Where n is the number of layers, and position encoding includes spatial position coding and learnable position encoding.

A. BACKBONE BASED ON IMPROVED PP-LCNet

In Backbones are the key component for extracting data features. Practices show that models with large-scale parameters usually have high performance, which makes detectors quickly become very large. Obviously, designing a small and effective backbone is crucial for lightweight detectors. Hence, we study the PP-LCNet and improve it for use as a backbone in the proposed detector.

PP-LCNet uses DepthSepConv as the basic module. This module has no operations similar to shortcuts, so there are no additional operations such as concatenation or element-wise add. These operations will slow down the reasoning speed of the model and improve the accuracy of the small model. Finally, PP-LCNet uses several basic modules to stack a basement. Besides this basement, PP-LCNet also has the average pooling, flatten, and full connection layers.

In order to make PP-LCNet have richer expression ability and improve its calculation efficiency, we redesign the structure of PP-LCNet. Specifically, like original PP-LCNet, the improved network starts with the Stem Conv, following by five 3 × 3 DepthSepConv modules which are organized as three layers. The first layer is a module with output shape of 32 × 128 × 128. The second layer has two modules with output shape of 64 × 64 × 64 and the last also has two modules with output shape of 128 × 32 × 32. After these five 3 × 3 DepthSepConv modules, there are seven 5 × 5 DepthSepConv that are divided into two groups. One group has five modules with output shape of 256 × 16 × 16. The other has two modules with output shape of 512 × 8 × 8. At last, the improved model ends up with a 1280-dimensional 1 × 1 convolution. More details of the structure of improved PP-LCNet can be found in Fig.2.

Compared to the original PP-LCNet, the improved model has two main differences. One is that the improved model only uses five 3 × 3 DepthSepConv modules and removes
the layer of average group pooling (GAP) and full connection (FC). The other is that the output shape of different type of DepthSepConv (3 × 3 or 5 × 5) is enlarged. These differences lead to the improved model having a less scale of parameters, more calculation efficiency, and more plentiful data features. Especially, removing the GAP layer can capture an abundance of features of background and margin, which make the improved model more suitable for use as a backbone to abstract features. Next, we further introduce the improved transformer to get predictions.

B. IMPROVED TRANSFORMER
As we all know, the quality of the activation functions significantly influences the performance of the network. The output tensor of the transformer is transferred to a three-layer perceptron, linear projection layer with ReLU activation function, and hidden layer d-dimension. The three-layer perceptron is also used to predict the label of the class.

Some practices indicate that the H-Sigmoid activation function has higher performance than ReLU with some deep networks. Hence, we use H-Sigmoid to replace the ReLU. The H-Sigmoid uses the ReLU6 function to simulate the sigmoid function. There are some advantages to using H-Sigmoid in the transformer. First, the H-Sigmoid function can offset the excessive linear growth of the ReLU activation function, and this unrestricted growth may affect the stability of the model. Next, a good numerical resolution is also achieved when the floating-point number of non-cloud devices is below the low accuracy. Finally, the H-Sigmoid activation function has less computation and can prevent gradient explosion and gradient disappearance. The activation function formulas of ReLU6 and H-Sigmoid and their derivatives are shown in Equations (1) and (2).

\[
\text{ReLU} = \min (6, \max (0, x)) \in [0, 6] \quad (1)
\]

\[
\text{HardSigmoid} = \frac{\text{ReLU} (x + 3)}{6} \in [0, 1] \quad (2)
\]

In addition, for improving training stability of transformer, we further investigate normalization functions. The transformer used in DETR adopts the layer normalization for normalizing a tensor with the shape \( [W^*H, b, C] \) to \( [W^*H, 1, C] \) in the direction of the channel. One of advantages of layer normalization (LN) is that it can be normalized within a single piece of data without batch training. Hence, it can be well used RNN without depending on batch size and the length of input sequence and have positive effects. While, it does not have much more effects on CNN. Considered to approximate the mean and standard deviation of the whole data with less computational cost and keep much background information of images, we employ group normalization to get more effects with CNN. The number of groups is adjustable, which provides the feasibility for the model to find a more suitable method. It calculates the mean value of \( (C_i/G)^{\star}H^{\star}W \) by grouping in the channel direction and then normalizing each group. Like layer normalization, its calculation is independent of batch sizes and is not constrained. The group normalization formula is shown in Equations (3-5):

\[
\tilde{x}_i = \frac{1}{\sigma_i} (x_i - u_i) \quad (3)
\]

where \( x \) represents the feature computed by a layer, and \( i \) is an index. In the case of 2D images, \( i = (i_b, i_c, i_w) \) is a 4D vector indexing the features in \( (b, C, H^{\star}W) \) order, where \( b \) is the batch axis, \( C \) is the channel axis, and \( H^{\star}W \) is the produce of spatial height and width axes. The mean \( u \) and \( \sigma \) standard deviation (std) are computed by:

\[
u_i = \frac{1}{m} \sum_{k \in S_i} x_K, \quad \sigma_i = \sqrt{\frac{1}{m} \sum_{k \in S_i} (x_K - u_i)^2} \quad (4)
\]

With \( \varepsilon \) as a small constant. \( S_i \) is the set of pixels in which the mean and std are computed, and \( m \) is the size of this set.

\[
S_i = \left\{ K | K_N = i_N, \quad \frac{K_C}{C/G} = \left[ \frac{i_C}{C/G} \right] \right\} \quad (5)
\]

Here, \( G \) is the number of groups, \( C/G \) is the number of channels per group, \( [\cdot] \) is the floor operation, and \( \left[ \frac{i_C}{C/G} \right] \) means that the indexes \( i \) and \( K \) are in the same group of channels.

IV. EXPERIMENT AND ANALYSIS
This section introduces experimental results and analysis of our proposed L-DETR model. The purpose of our experiment is to verify our proposal by using different data sets on two different devices. Experimental results show that the accuracy of the L-DETR model is better than DETR in Object detection.

A. IMAGE DATASETS
The experiment will be carried out from three aspects: First, we show the generalization ability of L-DETR in unbalanced
data sets. The unbalanced data (COCO-01, COCO-02) used in this paper is obtained by clipping the COCO2017 data set. The inconsistency between the training set and the validation set is regarded as noise. Next, the target detection ability of L-DETR is verified by using the data containing multiple categories. The data (COCO-03, COCO-04, MVI-01, Person) is obtained after clipping COCO2017 and UA-DETRAC and the self-collected pedestrian data set. The MVI-01 data is used to verify the engineering ability of the model. Finally, the experiments are carried out on different devices to verify the compatibility of L-DETR.

Considering the limitation of the computational power of the experimental equipment, we collect several small datasets from COCO2017 and UA-DETRAC for meeting the different needs of achieving our experiments.

1) Small datasets from COCO2017: We collect five small datasets from COCO2017, named COCO-01, COCO-02, COCO-03, COCO-04, and COCO-05. Images in COCO-01, COCO-02, and COCO-05 are selected randomly, which means that the types contained in the training set and the validation set will be inconsistent. The purpose is to verify the generalization ability of L-DETR. The training set of COCO-01 and COCO-05 contains 2000 pictures, and the validation set includes 500 images. The training set of COCO-02 contains 2200 pictures, and the validation set includes 800 images. COCO-03 has four categories (car, bus, airplane, boat). Its training set contains 1868 pictures, and its validation set contains 462 pictures. COCO-04 has eight categories (cat, dog, horse, sheep, cow, elephant, bear, giraffe). Its training set contains and validation set includes 1937 pictures and 718 images separately.

2) Small datasets from UA-DETRAC: This data set is obtained by using road surveillance cameras to collect vehicles and dynamically mark 8250 vehicles and 1.21 million object boxes. Vehicles are divided into four categories: car, bus, van, and other. We collect 2648 pictures as the training set and 662 as the validation set, called as MVI-01. The purpose of using this data set is to verify the application capability of L-DETR.

3) Self-collected data set (Person): We use a camera to collect data set called “Person”. Its training set includes 1343 images and validation sets has 521 images. Most of the images of the data set coincide with many people and object boxes, and it is used to verify the bounding box detection ability of the proposed model.

B. DETAILS OF MODELS

To analyze the performance of our proposal by contrast to the DETR model, we firstly establish a DETR model with the transformer that has three layers of encoder and decoder, and full connection with 256 hidden units. Moreover, the dimension of FFN of the established DERT is 1024 and the number of headers used by the multi-head self-attention mechanism is 4.

Typically, the established DETR model can use three backbones that are resnet50, resnet34, and resnet18. We take the established DETR with three backbones as three basic references to L-DETR. Besides, we also design models for analyzing the performance of improved PP-LCNet and transformer. First, A model that uses the original PP-LCNet as a backbone to replace the resnet50 is designed. Then, we use the improved PP-LCNet as the backbone of DETR to study its performance. Finally, the improved PP-LCNet combined with an improved transformer by using H-Sigmoid and L-DETR are established. It should be noted that, like backbones in DETR, the original PP-LCNet is pre-trained and the improved PP-LCNet also uses part of the parameters of the pre-trained model. It should be noted that the quantity of parameters of L-DETR is 26 percent and 46 percent of the original DETR with the backbone of resnet50 and resnet18.

In terms of classification accuracy, we use the average classification error rate of 50 iterations of the training set for comparison. In terms of edge detection, we use the highest value in the 45th to 50th iterations for comparison.

C. EXPERIMENTAL RESULTS

We implement four sets of experiment to evaluate the performance of L-DETR. In the first set of experiments, all mentioned models are implemented on GPUs with the random data set COCO-5 for evaluating performance of classification. Except for L-DETR model, the learning rate of other models is the same. The learning rate of backbones (lr_b) is set to $1 \times 10^{-5}$ and the learning rate of transformer (lr) is set to $1 \times 10^{-4}$. L-DETR model has been tested for many times, the learning rate of other models is the same. The learning rate of backbones (lr_b) is set to $4 \times 10^{-5}$ and the learning rate of transformer (lr) is set to $5 \times 10^{-6}$. Main results of first set of experiments on the validation set are shown in Fig.3. As the Fig.3 (a) shown, the DETR model with the original PP-LCNet (backbone) has poor performance in classification compared to DERT models with the backbone of resnet50, resnet34 and resnet18. This result indicates that, compared with resnet50, resnet34 and resnet18, the ability of capturing features of the original PP-LCNet is weak. While, Fig.3 (b) has shown that the improved PP-LCNet is better, but is still not comparable to resnet50, resnet34 and resnet18. Consequently, it is necessary to further improve transformer to use the insufficient features more efficient.

From Fig.3 (c), we can see that there are improvements in classification by combining improved PP-LCNet and improved transformer with the H-sigmoid function. There is a significant fluctuation, which implies that with a small date set and H-sigmoid function, the model may be overfitted. Hence, using normalization to improve stability and alleviate overfitting may be a promising way. The results shown in Fig.3 (d) have exactly proved the significance effect of group normalization. We can see from that, with the improved PP-LCNet and transformer, L-DERT not only has better stability, but also has better performance than DERT models with different backbones.

Experimental results shown in Fig.3 have testified to the superiority of L-DERT in classification. Despite L-DERT
a) The DETR model uses four different backbones to compare the classification losses. Resnet50 has the most parameters, which leads to the best model effect. The PPLCNet has the least parameters, which leads to the poor overall model effect. (b) On the DETR model, the improved PPLCNet reconstructed by our method is used as the backbone. Compared with the DETR using PPLCNet as the backbone, our method improves the performance of the model. (c) By combining the improved PPLCNet and the improved transformer with H-Sigmoid function, the classification has been improved. Because the data used in the experiment is less, the model produces overfitting. (d) The L-DETR model proposed in this paper has better classification accuracy under the condition of greatly reducing the overall parameters of the model.

**FIGURE 3.** Experimental results of models for classification. (a) The DETR model uses four different backbones to compare the classification losses. Resnet50 has the most parameters, which leads to the best model effect. The PPLCNet has the least parameters, which leads to the poor overall model effect. (b) On the DETR model, the improved PPLCNet reconstructed by our method is used as the backbone. Compared with the DETR using PPLCNet as the backbone, our method improves the performance of the model. (c) By combining the improved PPLCNet and the improved transformer with H-Sigmoid function, the classification has been improved. Because the data used in the experiment is less, the model produces overfitting. (d) The L-DETR model proposed in this paper has better classification accuracy under the condition of greatly reducing the overall parameters of the model.

**TABLE 1.** Bounding box detection results of DETR and L-DETR on different data sets.

| Model   | Data      | Learning rate (lr, lrb) | Class error | Bbox(iou =0.5) |
|---------|-----------|-------------------------|-------------|---------------|
| DETR    | COCO-01   | lr=4 \times 10^{-5},   | 93.084      | 0.0012        |
|         |           | lrb=5 \times 10^{-6}   |             |               |
| DETR    | COCO-02   | lr=4 \times 10^{-5},   | 92.204      | 0.0013        |
|         |           | lrb=5 \times 10^{-6}   |             |               |
| DETR    | COCO-03   | lr=4 \times 10^{-5},   | 97.769      | 0.0043        |
|         |           | lrb=5 \times 10^{-6}   |             |               |
| L-DETR  | COCO-01   | lr=4 \times 10^{-5},   | 78.95       | 0.0085        |
|         |           | lrb=5 \times 10^{-6}   |             |               |
| L-DETR  | COCO-02   | lr=4 \times 10^{-5},   | 81.373      | 0.0064        |
|         |           | lrb=5 \times 10^{-6}   |             |               |
| L-DETR  | COCO-03   | lr=4 \times 10^{-5},   | 29.917      | 0.0820        |
|         |           | lrb=5 \times 10^{-6}   |             |               |

Having better performance in classification, the improvement in bounding box detection is rarely small. We further conduct another set of experiments to analyze the capability of bounding box detection of L-DETR.

The second set of experiments we implemented is used to analyze the bounding box detection capability of L-DETR. All experiments are running on GPUs. COCO-01, COCO-02 and COCO-03 are used in these experiments. Considering that the results in fig.3 have shown that DETR model with resnet50, resnet34 and resnet18 have similar performances within 50 epoch, we only use the DETR model with resnet18 to contrastively analyze the capability of L-DETR. The learning rate of both models are same, in which the learning rate of backbones (lrb) is set to 4 \times 10^{-3} and the learning rate of transformer (lr) is set to 5 \times 10^{-6}. The experimental results on validation sets are shown in Table 1.

COCO-01 and COCO-02 are unbalanced data. It was evident that our L-DETR has a significant decrease in classification error rate for this unbalanced data, and its effect on bounding box detection is better than DETR with resnet18. It indirectly proves that the generalization ability of L-DETR is better. On COCO-03, which clips four classes from the data set COCO2017, our model converges faster than DETR and has dramatically improved bounding box detection.

Although the classification error rate decreases irregularly and the improvements in bounding box detection value are not too much, results in table 1 have shown the superiority of L-DETR on classification and the bounding box detection.

In the previous two sets of experiments, we use a greater learning rate in DERT models. To eliminate the interference of learning rate and further verify the proposed model, we design the third set of experiments on application data sets COCO-03, MVI-01, and Person with different learning rates. The comparative experimental results are shown in Table 2. It can be seen from COCO-03 with four categories that the convergence speed of the L-DETR model is still better than DETR. On MVI-01 and person with a similar background, the value of our model in edge detection is higher than DETR, and the classification error rate is lower than DETR. Our
Table 2. Different learning rates.

| Model | Data    | Learning rate (lr, lrb) | Class error (iou=0.5) |
|-------|---------|-------------------------|-----------------------|
| DETR  | COCO-03 | $10^{-5}, 1 \times 10^{-4}$ | 94.301, 0.0077        |
| DETR  | MVI-01  | $10^{-5}, 1 \times 10^{-4}$ | 22.545, 0.0152        |
| DETR  | Person  | $10^{-5}, 1 \times 10^{-4}$ | 96.703, 0.0025        |
| L-DETR | COCO-03 | $10^{-5}, 1 \times 10^{-4}$ | 29.917, 0.0085        |
| L-DETR | MVI-01  | $10^{-5}, 1 \times 10^{-4}$ | 21.442, 0.0057        |
| L-DETR | Person  | $10^{-5}, 1 \times 10^{-4}$ | 63.884, 0.0140        |

Table 3. 50 Iterative experiments were performed on two different devices.

| Model | Data    | Learning rate (lr, lrb) | Class error (50 epoch) |
|-------|---------|-------------------------|------------------------|
| L-DETR | COCO-01 | $10^{-5}, 1 \times 10^{-4}$ | 78.95, 0.0085         |
| L-DETR | COCO-02 | $10^{-5}, 1 \times 10^{-4}$ | 81.373, 0.0064        |
| L-DETR | COCO-03 | $10^{-5}, 1 \times 10^{-4}$ | 29.917, 0.00820       |
| L-DETR | COCO-04 | $10^{-5}, 1 \times 10^{-4}$ | 49.855, 0.00697       |
| L-DETR | COCO-01 | $10^{-5}, 1 \times 10^{-4}$ | 81.26, 0.0096         |
| L-DETR | COCO-02 | $10^{-5}, 1 \times 10^{-4}$ | 80.855, 0.00999       |

The model performs better than DETR in classification error rate and edge detection on different data sets.

To verify the equipment compatibility of L-DETR, we conducted 50 iterative experiments on L-DETR on two different devices. One is a device containing only two GPUs, and the other is a device with a combination of CPU and GPU. The learning rate set in the experiment is the transformer’s learning rate (lr) to $5 \times 10^{-6}$ and the backbone’s (lrb) to $4 \times 10^{-5}$. The experimental results are shown in Table 3.

When DETR and L-DETR use the same transformer’s learning rate (lr) to $5 \times 10^{-6}$, the backbone’s (lrb) to $4 \times 10^{-5}$ compared on the CPU + GPU device, the value obtained by DETR in bounding box detection is 0.0101, while the value obtained by L-DETR is 0.0210. The experimental results of the classification error rate are shown in Fig. 3. Even on different devices, our model is better than DETR in classification error rate and edge detection.

Since the backbone of L-DETR is proposed by devices focusing on CPU, we can see that the edge detection of L-DETR on CPU + GPU has been improved. The classification error rate has increased, which may be due to the problem of the device. The performance of the GPU devices on CPU + GPU is not as good as two GPU devices. The time of training 50 epoch on CPU + GPU is only one-quarter of training 50 epoch on two GPUs. This situation is pronounced with the increase in the amount of data. The parameters of L-DETR proposed by us are only 45.7% of those with resnet18 as the backbone and 26.1% of those with resnet50 as the backbone. Even on CPU + GPU devices, the time for our model to run 50 iterations is about half that of DETR with resnet50 as the backbone.

V. CONCLUSION

Object detection is one of the foundation tasks of computer vision. Designing a lightweight detector that can be deployed off-line on edge devices is a promising way to get high-performance for certain artificial intelligence applications. Therefore, this paper proposes a lightweight detector based on PP-LCNet and transformer, named L-DETR. More specifically, we redesign the structure of PP-LCNet and use it as the backbone. In addition, the transformer which is an important component of L-DETR is improved by using the H-Sigmoid activation function and group normalization. Multiple sets of experiments have shown the superiority of our proposed model in both classification and bounding box detection.

The contributions of this paper are as follows:

1) The overall parameters of the model are significantly reduced, which can effectively solve the limitation that can only be deployed in the cloud. We use a lightweight network to replace the original backbone, suitable for the model with CNN as the backbone to extract features. Further thinking, whether we use the engineering ticks in the lightweight network to the network with transformer as the backbone and combine the transformer’s modeling ability of global information with the advantages of CNN, we guess that it can reduce not only the overall parameters but also improve the reasoning speed of the model;

2) Our experiments show that different activation functions significantly impact different normalization methods. Some activation functions or normalization methods can get good results alone, but the results become uncertain after combining the two. Moreover, it is straightforward to improve the model’s overall performance through activation functions and normalization methods, and it can provide new ideas for different practical applications;

3) In the experiment, we found that when DETR uses a 2-layer encoder and a 4-layer decoder, the effect is higher than that of a 3-layer encoder and a 3-layer decoder, and found that the decoder part of the transformer module has a significant impact on loss. In the decoder part, the first multi-head self-attention significantly impacts loss. We wonder if...
we can separate this part from the decoder and set the number of layers to 1 to 2. The model can reduce the parameters and losses and maintain the original accuracy through the above methods. It provides a reference for building a similar DETR model structure.

In the future, we will further optimize the structure of L-DETR and improve its performance.

REFERENCES

[1] A. K.-F. Lui, Y.-H. Chan, and M.-F. Leung, “Modelling of destinations for data-driven pedestrian trajectory prediction in public buildings,” in Proc. IEEE Int. Conf. Big Data (Big Data), Dec. 2021, pp. 1709–1717.

[2] A. T. Gnanha, W. Cao, X. Mao, S. Wu, H. S. Wong, and Q. Li, “The residual generator: An improved divergence minimization framework for GAN,” Pattern Recognit., vol. 121, Jan. 2022, Art. no. 108222.

[3] S. Ren, K. He, and R. Girshick, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in Proc. Adv. Neural Inf. Process. Syst., vol. 28, 2015, pp. 91–99.

[4] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, “Focal loss for dense object detection,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 2980–2988.

[5] Z. Tian, C. Shen, H. Chen, and T. He, “FCOS: Fully convolutional one-stage object detection,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 9627–9636.

[6] S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Z. Li, “Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 9759–9768.

[7] M. Zheng, P. Gao, R. Zhang, K. Li, X. Wang, H. Li, and H. Dong, “End-to-end object detection with adaptive clustering transformer,” 2020, arXiv:2011.09315.

[8] Z. Dai, B. Cai, Y. Lin, and J. Chen, “UP-DETR: Unsupervised pre-training for object detection with transformers,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2021, pp. 1601–1610.

[9] N. Carion, F. Massa, and G. Synnaeve, “End-to-end object detection with transformers,” in Proc. Eur. Conf. Comput. Vis. Cham, Switzerland: Springer, 2020, pp. 213–229.

[10] J. Redmon and A. Farhadi, “YOLOv3: An incremental improvement,” 2018, arXiv:1804.02767.

[11] C. Cui, T. Gao, S. Wei, Y. Du, R. Guo, S. Dong, B. Lu, Y. Zhou, X. Lv, Q. Liu, X. Hu, D. Yu, and Y. Ma, “PP-LCNet: A lightweight CPU convolutional neural network,” 2021, arXiv:2109.15099.

[12] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, and V. Vasudevan, “Searching for MobileNetV3,” in Proc. IEEE Int. Conf. Comput. Vis., Oct. 2019, pp. 1314–1324.

[13] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, “Swin transformer: Hierarchical vision transformer using shifted windows,” in Proc. IEEE/CVF Conf. Comput. Vis. (ICCV), Oct. 2021, pp. 10012–10022.

[14] C.-H. Tu, J.-H. Lee, Y.-M. Chan, and C.-S. Chen, “Pruning depthwise separable convolutions for MobileNet compression,” in Proc. Int. Joint Conf. Neural Netw. (IJCNN), Jul. 2020, pp. 1–8.

[15] I. Hubara, M. Courbariaux, and D. Soudry, “Binarized neural networks,” in Proc. Adv. Neural Inf. Process. Syst., vol. 29, 2016, pp. 1–9.

[16] H. Felix, W. M. Rodrigues, D. Macedo, F. Simoes, A. L. I. Oliveira, V. Teichrieb, and C. Zanchettin, “Squeezed deep 6DoF object detection using knowledge distillation,” in Proc. Int. Joint Conf. Neural Netw. (IJCNN), Jul. 2020, pp. 1–7.