Efficient and Effective Object Detection and Recognition: from Convolutions to Transformers

Tianxiao Zhang

Submitted to the graduate degree program in Department of Electrical Engineering and Computer Science and the Graduate Faculty of the University of Kansas in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Committee members

_____ ,

_____ ,

_____ ,

_____ ,

_____ ,

_____ ,

_____ ,

_____ ,

Date defended: April 17, 2025

The Dissertation Committee for Tianxiao Zhang certifies
that this is the approved version of the following dissertation:

Efficient and Effective Object Detection and Recognition: from Convolutions to Transformers

_____

,

Date approved: _____

Abstract

With the development of Convolutional Neural Networks (CNNs), computer vision has entered a new era, significantly enhancing the performance of tasks such as image classification, object detection, segmentation, and recognition. Furthermore, the introduction of Transformer architectures has brought the attention mechanism and a global perspective to computer vision, advancing the field to a new level. The inductive bias inherent in CNNs makes convolutional models particularly well-suited for processing images and videos. On the other hand, the attention mechanism in Transformer models allows them to capture global relationships between tokens. While Transformers often require more data and longer training periods compared to their convolutional counterparts, they have the potential to achieve comparable or even superior performance when the constraints of data availability and training time are mitigated.

In this work, we propose more efficient and effective CNNs and Transformers to increase the performance of object detection and recognition. (1) A novel approach is proposed for real-time detection and tracking of small golf balls by combining object detection with the Kalman filter. Several classical object detection models were implemented and compared in terms of detection precision and speed. (2) To address the domain shift problem in object detection, we employ generative adversarial networks (GANs) to generate images from different domains. The original RGB images are concatenated with the corresponding GAN-generated images to form a 6-channel representation for cross-domain object detection. (3) A dynamic strategy for improving label assignment in modern object detection models is proposed. Rather than relying on fixed or statistics-based adaptive thresholds, a dynamic paradigm is introduced to define positive and negative samples. This allows more high-quality samples to be selected as positives, reducing the gap between classification and IoU scores and producing more accurate bounding boxes. (4) An efficient hybrid architecture combining Vision Transformers and convolutional layers is introduced

for object recognition, particularly for small datasets. Lightweight depth-wise convolution modules bypass the entire Transformer block to capture local details that the Transformer backbone might overlook. The majority of the computations and parameters remain within the Transformer architecture, resulting in significantly improved performance with minimal overhead. (5) An innovative Multi-Overlapped-Head Self-Attention mechanism is introduced to enhance information exchange between heads in the Multi-Head Self-Attention mechanism of Vision Transformers. By overlapping adjacent heads during self-attention computation, information can flow between heads, leading to further improvements in vision recognition.

Acknowledgements

I would like to thank my advisors professor Bo Luo and professor Guanghui Wang for instructing and supporting me during my Ph.D. career. I am thankful to professor Cuncong Zhong as my research advisor for collaborating with me on some projects. I would like to thank my committee: professor Bo Luo, professor Prasad Kulkarni, professor Fengjun Li, professor Cuncong Zhong, professor Huazhen Fang and professor Guanghui Wang for their valuable advice with my dissertation. I am grateful to my collaborators in my lab: Dr. Wenchi Ma, Dr. Kaidong Li, and Dr. Xiangyu Chen for their help with my research. I would like to express my gratitude to my parents for their endless support during the long research journey. I am also grateful to all my friends who encouraged and supported me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Computer vision has entered into the deep learning era since the success of AlexNet (Krizhevsky et al., 2012). Due to the inductive bias in CNNs, the computer vision tasks are soon dominated by CNNs and most computer vision tasks are significantly boosted and benefited from their light-weight small kernels. Transformer (Vaswani et al., 2017) was originally invented for natural language processing by capturing the long relationship between different work tokens. Vision Transformer (Dosovitskiy et al., 2020) introduces the Transformer architecture into vision recognition. In contrast to CNNs traversing the images with small kernels, the Vision Transformer model splits the image into equal-size non-overlapping image patches that are embedded as the tokens. The patch tokens are fed into the Transformer encoder for vision recognition. Even though Vision Transformer models lack the inductive bias, the global attention in Transformer encourages the patch tokens to have interactions with all other tokens and see better.

1.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) employ small kernels to efficiently traverse entire images to attain the features. Usually, the shallow convolutional layers focus on low-level features such as edges. With the layers going deeper, the size of the feature maps is reduced accordingly, and semantic information would be attained by the deep layers. The small kernels focus on the relationship of neighboring pixels of the feature maps, making CNNs highly efficient and suitable for images. In addition, pooling layers are harnessed in CNNs to reduce the size of the feature maps so that the pixels in the feature maps could have larger receptive fields with the layers going deeper

to obtain the semantic information in the images. The activation layers such as ReLU (Nair & Hinton, 2010) act as non-linear layers to introduce non-linearity into the CNNs. Multi-layer perceptrons (MLPs) are typically utilized as the final layers to aggregate all the neurons from the previous layers and finally output the predictions. A typical CNN architecture is demonstrated in Fig. 1.1.

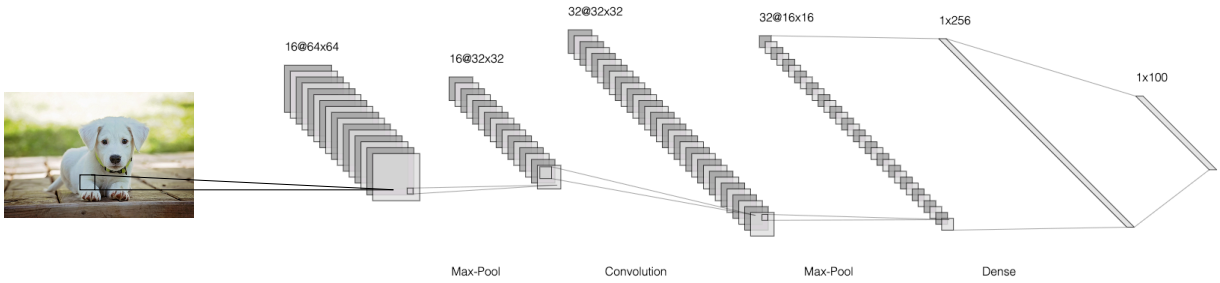

Figure 1.1: A typical CNN architecture

Object detection not only classifies the object categories but also localizes the objects by coordinates. Object detection is helpful for object tracking, segmentation, and image or video understanding. The object detectors could be categorized as two-stage detectors and one-stage detectors based on the number of stages. The two-stage detection models typically select object candidates that have high probabilities containing objects, and classify and further refine the coordinates of the object candidates in the second stage. One-stage detection models usually classify and predict the coordinates directly without candidate generation. Two-stage detectors usually have better performance than one-stage detectors, while one-stage detectors frequently have faster inference speed than two-stage models. Additionally, object detectors could also be classified as anchor-based models and anchor-free models based on the usage of anchors. Anchor-based models exploit the pre-defined anchors at each location so that the objects can select the positive anchors corresponding to them based on the thresholds. Anchor-free models usually utilize special points or anchor points to localize the objects.

A typical architecture for object detection is illustrated in Fig. 1.2. Usually, the detection model is comprised of the backbone, the neck, and the head. The backbone is typically harnessed to

extract the features from the images. The neck is utilized to merge different feature maps so that the semantic information from the deep feature maps can be employed to enhance the shallow feature maps. The head is to classify and localize the objects shown in the images. The classification and localization would be implemented in multiple feature maps with various scales so that the large feature maps could detect the small objects and the small feature maps could detect the large objects.

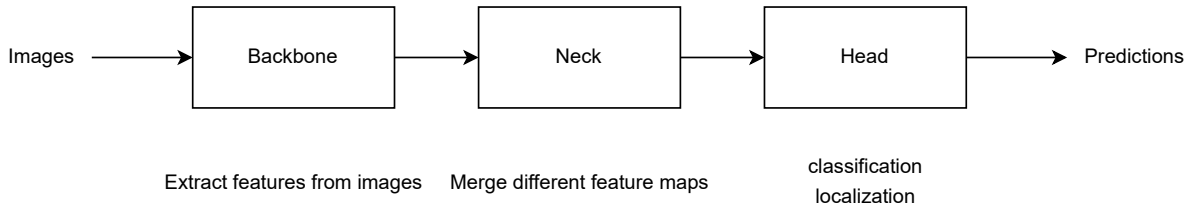

Figure 1.2: A typical Object detector

1.2 Vision Transformer

Transformer (Vaswani et al., 2017) was employed to obtain the relationship between the words in languages with the attention mechanism. Vision Transformer (Dosovitskiy et al., 2020) applied the Transformer architecture to vision recognition by treating each image patch as a token. The image is split into equal-size non-overlapping patches that are embedded as tokens. The positional embeddings are added to the patch tokens since the patch tokens would lose positional information. The positionally embedded patch tokens are fed into the Transformer encoder, which is comprised of several Transformer blocks that contain Multi-Head Self-Attention (MHSA) and Feed-Forward Networks (FFNs). The architecture for Vision Transformer is shown in Fig. 1.3.

Multi-Head Self-Attention utilizes attention mechanism to obtain the global relationship between tokens. Each token embedding would be linearly transformed to query (Q), key (K), and value (V). The weights for the values are calculated by the dot product of the current query and all keys. Then, the result for the current query is the summation of weighted values. A similar implementation is conducted for all queries. The queries, keys and values are typically divided into

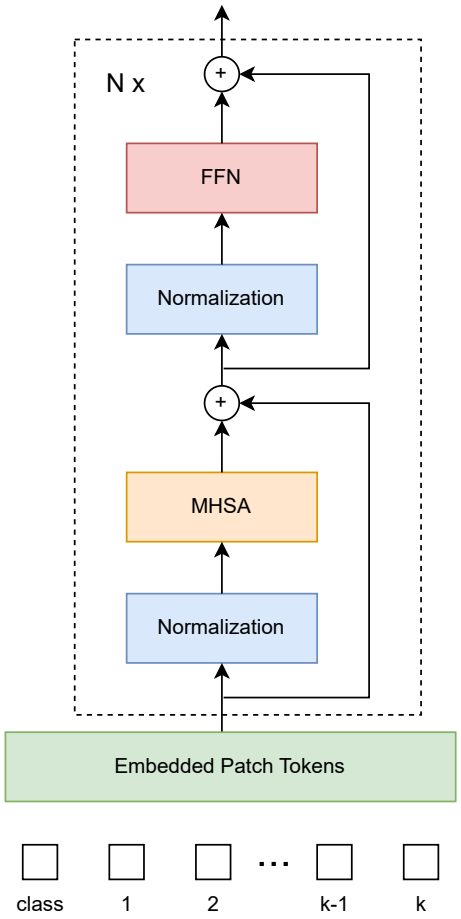

Figure 1.3: The architecture for Vision Transformer

multiple heads so that the self-attention is implemented in different heads independently to enhance the subspace representation. Feed-Forward Networks usually contain two layers of Multi-Layer Perceptrons (MLPs) to aggregate the information in each token embedding.

In the original Transformer (Vaswani et al., 2017) for language processing, the normalization is implemented after the Multi-Head Self-Attention and Feed-Forward Networks in each block. For Vision Transformers, the normalization is implemented before the Multi-Head Self-Attention and Feed-Forward Networks. The Vision Transformer model attaches a learnable class token to the patch tokens to learn the images, and the class token would be finally utilized to classify the images.

1.3 Convolutions v.s. Transformers

Convolutional Neural Networks (CNNs) employ small kernels to obtain the features by traversing the kernels across the images, and the receptive field is gradually enlarged by reducing the size of the feature size with the layers going deeper. For the pixels in the images, the neighboring pixels have stronger relationships than the distant pixels. Convolutional kernels could naturally obtain the relationship between the adjacent pixels. Nonetheless, the lack of global view for convolutional kernels is the main disadvantage for CNNs, even though the receptive field is increased by the pooling layers for the deep layers.

For the Transformer architecture, the self-attention mechanism calculates the relationship between each token and all other tokens for global view, and similar Transformer blocks are repeated multiple times to output the final results. Although the attention mechanism enables the Transformer models to have global view, the lack of inductive bias makes the Transformer models require more training data and training epochs to converge and achieve relatively great performance for vision tasks. However, if the training data is abundant and the training epochs are large enough, the Transformer models could yield comparable or even better performance than the convolutional counterparts.

Additionally, the complexity of Transformer is usually higher than that of convolutional models. The convolutional kernels are typically small, efficient, and effective. For Transformer models, the quadratic complexity in self-attention could limit the number of tokens so that the Transformer models would consume much more computational resources for high-resolution images.

According to the advantages and disadvantages of convolutional neural networks and Transformers, a hybrid architecture that combines the convolutions and Transformers may be a better choice so that the advantages from both architectures could be enhanced by each other. The combination should be flexible enough so that a similar hybrid structure of combination could also be applicable to other Transformer architectures. Moreover, the complexity of the hybrid models is also important since efficient hybrid models are more applicable to the real-world scenarios and the hybrid models that have much more complexity may not efficiently take advantage of the strength

from both architectures. In this work, a highly efficient and effective hybrid model is proposed for vision tasks with significant performance improvement.

1.4 The proposed works

The proposed works investigate efficient and effective object detection and recognition with convolutional neural networks and Vision Transformers. CNNs and Transformers have their own advantages and disadvantages, and sometimes they can be complementary to each other. How to take advantage of the strengths and reduce the effect of disadvantages could be one of the main factors in the success of the models.

1.4.1 CNN-based works

Although the current detection models have achieved excellent performance on some benchmarks and have been applied to multiple scenarios in real-world applications, there are still some drawbacks existing in the detection models.

The detection models can hardly detect extremely small objects shown in the images or videos, especially when the images or videos have large resolutions. Small object detection is a long-lasting problem since small objects only contain a few pixels and are easily ignored and treated as the background. Thus, detecting and tracking small objects such as golf balls are extremely difficult. Thus an approach is proposed to utilize the cropped patches containing the golf ball so that the golf ball could much easier be detected on the small patches. The patch locations are updated with the Kalman filter so that the patches move according to the golf balls in the videos.

The domain shift in the dataset is another problem in the detection models. For instance, if the detection models are trained with the dataset contains with all images of the day-time streets, the trained models would perform not well on the dataset with images of night-time streets, even though the scenery is the street for both datasets. The domain shift from day to night or night to day requires some techniques since the dataset in a specific domain might not be available or

might hard to be collected and labeled. Thus a strategy is proposed to employ GAN (Generative Adversarial Network) to generate the fake images of the target domain from the source domain and concatenate the real images from the source domain and their corresponding fake images from the target domain to form a six-channel representation. The six-channel representation is utilized to train and evaluate the detection models.

The definition of positive samples and negative samples is significant for a successful detection model. The anchor-based object detector is taken as an example to illustrate the problem. The IoU (Intersection over Union) threshold is important for dividing the positives and negatives. The previous detection models frequently utilize fixed thresholds (e.g., 0.5) for the division of positives and negatives. Nonetheless, some regular-shaped objects may have more positive samples corresponding to them, while some slender or small objects may correspond to less positive samples. The networks might focus on those regular-shaped objects and pay less attention to objects with non-regular shapes. In recent years, the detection models have utilized adaptive thresholds which are computed by the statistic parameters. Thus the IoU thresholds for objects with regular shapes are relatively high, and the IoU thresholds for objects with non-regular shapes are relatively low. The adaptive thresholds are more reasonable for the division of positives and negatives. However, the adaptive thresholds are not dynamically adjusted by the training status, and once the positives and negatives are defined for one image, they will not be changed during the training process. Based on the analysis, we proposed dynamic training for object detectors according to the predictions. If the predicted bounding boxes are less accurate, the samples might be divided as negatives even though the pre-defined anchors have high IoUs with the ground truth objects. Similarly, the samples might be divided as positives if the predicted bounding boxes are more accurate, although the pre-defined anchors have low IoUs with the ground truth objects. The proposed dynamic mechanism might select more high-quality samples as positives with the guide of predictions.

1.4.2 Transformer-based works

Vision Transformers have achieved tremendous progress since Transformer (Vaswani et al., 2017) was introduced to vision recognition (Dosovitskiy et al., 2020). The Vision Transformer models are becoming more and more friendly to vision tasks by improving the Transformer structure.

Vision Transformers' advantage is obtaining the global relationship between the patch tokens and the strength of CNNs to capture the images' local detail information. A hybrid architecture with Transformers and CNNs is usually friendly to vision tasks. However, flexibility and complexity are the main concerns for designing hybrid models so that the hybrid models could be flexible for most Vision Transformer architectures and would not increase too much complexity. A new hybrid architecture is proposed by combining light-weight depth-wise convolutions and Vision Transformer models. The proposed depth-wise convolution modules bypass the entire Transformer block containing Multi-Head Self-Attention module and Feed-Forward Network module so that the proposed convolutional modules are not involved in the inner structure of the Transformer models and the modules are flexible to be applied to most Vision Transformer models. In addition, the light-weight depth-wise convolution modules are extremely light-weight, and most parameters and computations are from the Transformer structure. The light-weight modules introduce tiny overheads that could be ignored, and the performance in the vision tasks is significantly enhanced.

The success of Transformer models is mainly from the effective information exchange in the Multi-Head Self-Attention mechanism. The effective information interaction between tokens makes the Transformer models have the global view of the input data. However, the attention is calculated independently in each head without the information of other heads. To further enhance the information interactions, a new strategy Multi-Overlapped-Head Self-Attention (MOHSA) is proposed. In the proposed MOHSA, the queries, keys and values in each head overlap with the queries, keys and values of two adjacent heads. Zero-padding is utilized for the first and last head with only one adjacent head. Some overlap paradigms are proposed to empirically illustrate the different effects of various overlap ratios. The performance for vision recognition is further enhanced with the proposed MOHSA.

Chapter 2

Efficient Golf Ball Detection and Tracking Based on Convolutional Neural Networks and Kalman Filter

Abstract

This chapter focuses on the problem of online golf ball detection and tracking from image sequences. An efficient real-time approach is proposed by exploiting convolutional neural networks (CNN) based object detection and a Kalman filter based prediction. Five classical deep learning-based object detection networks are implemented and evaluated for ball detection, including YOLO v3 and its tiny version, YOLO v4, Faster R-CNN, SSD, and RefineDet. The detection is performed on small image patches instead of the entire image to increase the performance of small ball detection. At the tracking stage, a discrete Kalman filter is employed to predict the location of the ball, and a small image patch is cropped based on the prediction. Then, the object detector is utilized to refine the location of the ball and update the parameters of the Kalman filter. In order to train the detection models and test the tracking algorithm, a collection of golf ball datasets is created and annotated. Extensive comparative experiments are performed to demonstrate the effectiveness and superior tracking performance of the proposed scheme.

2.1 Introduction

Golf is a popular sport that has attracted a huge number of participants and audiences. However, watching the dim, tiny, fast-moving golf balls flying around is not such a fun experience. Practicing

golf in the golf driving range causes problems to visualize the trajectory of the golf ball. In certain circumstances, the golf ball quickly flies beyond visual sight immediately after being hit. That is why golf is one of the sports that frequently utilize video analysis during the training session. However, detecting the fast-moving tiny golf ball is very challenging and problematic since the golf ball can fly extremely fast and quickly disappear from the camera's field of view. As soon as the golf ball flies farther from the camera, it becomes smaller and smaller in the image, which increases the difficulty in detection. Another huge challenge is caused by motion blur. As we know that, if the camera's frame rate is not high enough, a significant blur will appear in the image, making the ball hard to detect.

Most of the proposed methods could be classified as the sensor integration method (Umek et al., 2017), the traditional object tracking method (Lyu et al., 2015), and the traditional computer vision approach (Kim & Kim, 2011) (Woodward & Delmas, 2005). Umek *et al.* (Umek et al., 2017) proposed a sensor-integrated golf equipment. Without changing the functionality of a golf club, two orthogonal affixed train gage (SG) sensors, a 3-axis gyroscope, and an accelerometer were used to monitor the state and actions of the golf swing. Different types of golf swings and movements in the early phases of swing can be detected with strain gauge sensors. By collecting the returned data, the biofeedback application could be used to help golf beginners to learn repetitive swings. For the traditional object tracking approach, Lyu *et al.* (Lyu et al., 2015) proposed a real-time, high-speed moving ball shape object tracking algorithm using the fusion of multiple features. In the initial step, frame difference is obtained by subtracting the two consecutive frames. The image obtained is converted to a binary image, and an array of candidate objects is collected based on all the candidate contours. Then, the moments of all the contours are calculated. False candidates are eliminated by applying a multi-feature extraction method. Finally, the ROI is refined by the contour obtained in the previous step.

Using the traditional computer vision approach, Woodward *et al.* (Woodward & Delmas, 2005) proposed a low-cost golf ball tracking method. The images are taken by two calibrated stereo cameras and reconstructed in 3-D space. To realize VR golf, the authors only consider

recognizing the golf ball that is centimeter-wise from the camera. In the system, there are two dots, one blue and one yellow, attached to the golf club, and the golf ball is pink. The system requires an unchanged background. By subtracting the consecutive frame by the background, the golf ball could be tracked. While in reality, golf balls are a hundred meters away from the camera. The 3D reconstruction process is computationally intensive, which could not be implemented in real time. The authors labeled the golf ball and the club with different colors, by recognizing the color feature, the club and golf ball could be detected. Overall, none of these are efficient for real-time golf ball detection. Kim *et al*. (Kim & Kim, 2011) proposed a two-stage method for ball detection from images taken by the multi-exposure camera. They first estimate the region of the ball based on a threshold calculated using Otsu's method. Then, segment the ball through a labeling process.

Classical visual object tracking algorithms all fail in tracking the golf ball since the ball is too small in the image. In our experiments, we found that feeding high-quality images directly into the detection models led to significant computational load and terrible results. Inspired by (Lyu et al., 2015), we propose an accurate and real-time golf ball tracking approach based on the object detection and Kalman filter (Wu et al., 2017). Instead of using the entire image, we make detection from small cropped image patches to increase the detection accuracy without sacrificing speed. Extensive results demonstrate the effectiveness of the proposed approach.

2.2 Related Work

Artificial intelligence (AI) has significantly changed our lives in recent years. In the field of sports, AI technology is opening up a new path. NBA has established a data computing system that can mine data in games and model data through machine learning. AI is also starting to play the role of auxiliary training. Microsoft has developed a sports performance platform, a set of data management systems that analyzes the training and performance of athletes. Computer vision techniques can provide athletes with sports data analysis, help the coach to target weak areas, and improve sports performance, such as pose estimation, object detection, and object tracking. Golf

is one of the sports that frequently utilize video analysis and could be benefited significantly with the help of computer vision techniques. Pose estimation could assist in improving the action of the swing. Object tracking could help to track and plot the curve of the golf ball for further analysis.

To better assist golf players during training, Umek *et al*. (Umek et al., 2017) proposed a method based on sensor fusion to record the swing data. Wang *et al*. (Wang et al., 2018) proposed a high-speed stereo vision system under indoor lighting conditions using circle detection (Lu et al., 2017) to detect the golf ball and utilize the dynamic ROI for golf ball tracking. Lyu *et al*. (Lyu et al., 2015) proposed a golf ball tracking method based on multiple features. In real circumstances, the illumination variations, weather conditions, and background variations will complicate the detection and tracking of the golf ball. Inspired by the progress of state-of-the-art object detectors, which have demonstrated great capability in daily lives, such as automatic driving, and target tracking using surveillance cameras. We propose to solve this problem by using deep learning models.

Object detection and tracking are two classical computer vision problems. With the fast development of deep learning, convolutional neural network (CNN) based object detection and tracking approaches have drawn more attention over the classical approaches due to their unprecedented performance. Based on the tremendous development of computing power, especially GPU, which makes it possible to detect and track an object in real-time (Li et al., 2019a). Object tracking is the process of locating a moving object or (multiple objects) over time in consecutive video frames. Object tracking is widely applied in human-computer collaboration, traffic monitoring, and surveillance system. Classical tracking methods (Danelljan et al., 2014) (Sui et al., 2015) (Sui et al., 2016) require manual initialization with the ground truth in the first frame. In recent years, the tracking-by-detection methods have drawn more attention for their real-time applications with the fast development of GPUs and TPUs. Correlation filter-based trackers have also attracted a lot of attention due to their high-speed performance (Henriques et al., 2014) (Sui et al., 2018) (Sui et al., 2019). When it comes to this specific problem, with such a small and extremely fast-moving object, they all fail and would return useless information once failed during tracking.

Object detection focuses on detecting semantic instances for certain classes in videos or images. The fundamental purpose of object detection is to simultaneously localize and classify the objects shown in the images or videos. Object detection plays a crucial role in lots of practical applications, such as face recognition, pose estimation, medical diagnostics, etc. Object detection has a variety of applications, however, it also comes with challenges and problems. Some commonly seen challenges and problems are variations of object sizes, occlusions, viewpoints, and light conditions. A number of papers attempt to solve those problems, especially how to detect small objects in images or videos. The methods for object detection generally fall into either classical machine learning approaches (Dalal & Triggs, 2005) or deep learning approaches (Bochkovskiy et al., 2020) (Li et al., 2020a) (Lin et al., 2017a) (Lin et al., 2017b) (Ma et al., 2020b) (Redmon & Farhadi, 2017) (Redmon & Farhadi, 2018) (Zhang et al., 2018). In classical machine learning based approaches, the features are predefined, while deep learning based methods are able to perform end-to-end training without specifically defined features. In golf ball detection, the machine learning based approach does not show good performance, although the golf ball has a distinctive shape and color. In practical circumstances, there are many distracting objects in the scene, which could cause failure in target object detection. In addition, the background in real circumstances is not stationary, making it difficult to apply the background subtraction approach.

Deep learning based object detection models are usually categorized as one-stage methods and two-stage methods. For two-stage methods, the model proposes a set of candidates by selective search (Uijlings et al., 2013) or RPN (Ren et al., 2015), and then the classifiers will further refine the coordinates of the region proposals proposed in the first stage and simultaneously classify the object inside each bounding box. In contrast, the classifiers of one-stage models directly refine and classify the densely pre-defined anchors. Since the objects to be detected have various shapes and sizes, some object detectors such as SSD (Liu et al., 2016), FPN (Lin et al., 2017a), RetinaNet (Lin et al., 2017b) and RefineDet (Zhang et al., 2018) define anchors on various sizes of feature maps so that the anchors in large feature maps can recognize small instances and the anchors in small feature maps can detect large objects. Designing anchors on various sizes of feature maps to

detect instances with varying sizes and shapes is so effective that most modern detectors adopt this technique to design network architectures.

Recently, some anchor-free detectors, like CornerNet (Law & Deng, 2018), FSAF (Zhu et al., 2019), ExtremeNet (Zhou et al., 2019), CenterNet (Duan et al., 2019), and FCOS (Tian et al., 2019) are designed without any pre-defined anchor boxes. In addition, there is also some state-of-the-art such as (Zhang et al., 2020a) which combines anchor-based detectors and anchor-free detectors to adaptively train the samples. Two-stage object detectors often outperform one-stage object detectors while one-stage models have faster speed than two-stage models. Since the goal is to detect and track the golf ball in real time, the detectors chosen have to be both fast and accurate. Most of the one-stage methods have the advantage over two-stage methods in speed with the sacrifice of accuracy. To balance the trade-off, Faster R-CNN (Ren et al., 2015), YOLOv3 and its light version (Redmon & Farhadi, 2018), YOLOv4 (Bochkovskiy et al., 2020), SSD (Liu et al., 2016) and RefineDet (Zhang et al., 2018) are chosen in our study.

2.3 The Proposed Approach

We propose a two-stage scheme for golf ball detection and tracking. As shown in Fig. 2.1, we employ a Kalman filter to predict the estimated location of the golf ball in the next consecutive frame. The area centered at the estimated location is cropped and sent to the detector. In general, the image resolution for golf live TV shows is higher than 1080p, which is computationally intensive. To boost the accuracy and reduce the computational loads, we apply the discrete Kalman filter model for the location estimation. The proposed tracking approach depends on a recursive process of *a priori* estimation, object detection, and *a posteriori* estimation. The tracker is initialized in the first frame. In most golf practice driving ranges, the camera is set beside the player so that the full swing is recorded. In this scenario, the initial position of the golf ball is located in the lower center of the frame, which can be easily detected automatically by a detection model.

For the tracking, after the ball is detected on the cropped image patch, the "Time Update" in the Kalman filter will predict the coordinate of the golf ball in the next frame, which is denoted as

a priori estimation. The next image will be cropped with respect to the location that the Kalman filter predicts. Then the cropped image patch will be sent to the object detector. After that, based on the detection results, the "Measurement Update" in the Kalman filter will calculate *a posteriori* estimate of the current state which will be sent to the "Time update" in the Kalman filter for *a priori* estimation to predict the coordinate of the ball in the next frame. This loop will continue until all frames are detected.

2.3.1 Object Detection

The detection models we chose in this study are Faster R-CNN (Ren et al., 2015), YOLOv3 and its light version (Redmon & Farhadi, 2018), YOLOv4 (Bochkovskiy et al., 2020), SSD (Liu et al., 2016) and RefineDet (Zhang et al., 2018). Among these models, YOLOv3 (Redmon & Farhadi, 2018), YOLOv4 (Bochkovskiy et al., 2020), SSD (Liu et al., 2016), and RefineDet (Zhang et al., 2018) are considered as representatives of the one-stage object detection models, while Faster R-CNN (Ren et al., 2015) is a classical two-stage object detector. Object detection is to determine the location of the objects in the images or videos and simultaneously classify those objects in the image or videos.

YOLOv3 & YOLOv3 Tiny YOLOv3 (Redmon & Farhadi, 2018) predicts bounding boxes using pre-defined boxes with dimension clusters. YOLOv3 predicts multiple bounding boxes per grid cell. The prediction is based on the highest IoU between the predicted bounding box and the ground truth bounding box. Non-maximum suppression was applied when multiple bounding boxes appear in the image. The confidence score for the occurrence of the object in each bounding box is predicted using logistic regression. Inspired by the image pyramid, YOLOv3 adds several convolutional layers after the base feature extractor to make predictions at three different scales. Features are extracted from these three scales. YOLOv3 also adds a cross-layer connection between two prediction layers (except the output layer) and earlier finer-grained feature maps. YOLOv3 first up-samples the deep feature maps and then merges it with the previous features by

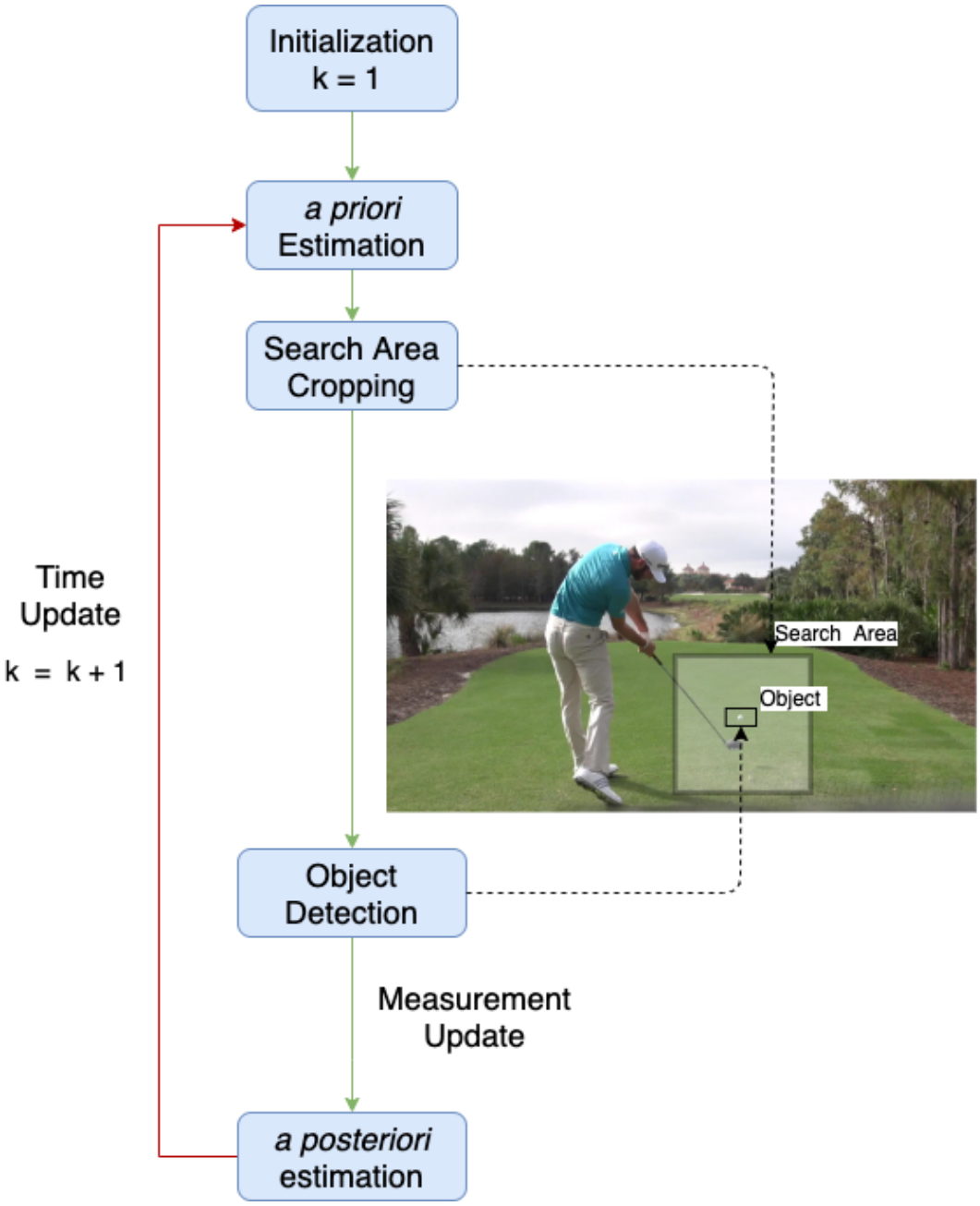

Figure 2.1: The flowchart of the proposed tracking strategy. We integrate the object detection and the Kalman filter together for object tracking. In the first frame of the video, we employ the detection model to localize the ball and its coordinates. Each time after the time update, *a priori* state estimate and *a priori* estimate error covariance will be utilized to perform the measurement update. After measurement update, *a posteriori* state estimate and *a posteriori* estimate error covariance will be employed to do the time update to predict *a priori* state estimate and *a priori* estimate error covariance in the next frame.

concatenation to better detect small objects. YOLOv3 provides high accuracy with relatively fast speed.

YOLOv3 tiny is a simplified version of YOLOv3. It reduced the number of convolutional layers to 7 and utilized 1×1 and 3×3 convolutional layers with fewer parameters. YOLOv3 tiny also utilized a pooling layer to achieve feature map size reduction, while YOLOv3 employed the convolutional layer with a step size of 2.

YOLOv4 YOLOv4 (Bochkovskiy et al., 2020) focuses on a bag of freebies that only changes the training process that might increase the training cost to enhance the performance of the detector without additional cost in inference time and the bag of specials that increases the accuracy of the detector with only a little increase of the inference time by post-processing. The backbone of YOLOv4 is CSPDarknet53 (Wang et al., 2020) which is an enhanced version of DarkNet53 used in YOLOv3. To increase the receptive field, YOLOv4 utilizes SPP (He et al., 2015) over the backbone network. In addition, instead of employing FPN (Lin et al., 2017a) to merge feature maps with different scales in YOLOv3, YOLOv4 exploits PANet (Liu et al., 2018) to combine different network levels. The detector head of YOLOv4 is still the same as YOLOv3. There are many other bag of freebies and bag of specials utilized in YOLOv4 to improve the performance of the detector. Some of them would increase little inference cost but improve the performance significantly (Bochkovskiy et al., 2020).

Faster R-CNN Faster R-CNN (Ren et al., 2015) is built on top of two networks, region proposal network (RPN) and detection network. RPN is responsible for generating candidate proposals from a series of densely sampled anchors with different sizes and ratios in each location on the feature maps. By default, there are 3 scales and 3 aspect ratios total 9 anchors at each feature map location. The output of the RPN is a bunch of rectangular proposals that may contain some objects in the image and associate with objectness scores. The proposals will later be processed by the classifiers to verify the occurrence of the objects and the regressors to further refine the bounding boxes of the region proposals. To be more accurate, The function of RPN is to judge if the anchors

contain some objects and refine the anchor boxes if they contain some objects. Then the region proposals with a high possibility to be foreground will be sent to the detection network to do the classification and bounding box regression.

For the detection network, Fast R-CNN (Girshick, 2015) is adopted. ROI pooling is leveraged to the region proposals that may contain some objects and then two branches are adopted, one branch is for refining the bounding boxes of the region proposals, and the other is to classify the objects that may be contained in the region proposals. To make this network fast and accurate, RPN shares the full-image convolutional features of all region proposals and by ROI pooling, the features of region proposals can be fixed and easily fed into the detection network. Due to the introduction of RPN, most background proposals are removed and only those proposals with high possibility containing some objects are fed into the detection network, thus the accuracy of Faster R-CNN is often better than that of one-stage object detectors which directly refines and classifies the dense anchors without selecting them. In the experiment, VGG-16 (Simonyan & Zisserman, 2014) is selected as the backbone network for Faster R-CNN.

SSD SSD (Liu et al., 2016) is abbreviated for Single Shot Multibox Detector which is a one-stage multiple-scale approach. SSD was utilized VGG-16 (Simonyan & Zisserman, 2014) as the backbone network. However, SSD (Liu et al., 2016) exploits feature maps with various scales to detect objects with different shapes and sizes. Although SSD achieved great performance on large objects, it shows drawbacks in detecting small objects. Lower level feature maps may contain more details while less semantic information, while deep feature maps may have more semantic information but fewer details. To better detect small objects such as golf balls, we have to extract and highlight the feature at a lower level of feature maps. Nonetheless, lacking enough semantic information in those shallower feature maps may prevent us from achieving good detection results for small objects.

RefineDet Similar to SSD (Liu et al., 2016), RefineDet (Zhang et al., 2018) also utilizes pre-defined bounding boxes on multi-scale feature maps and objectness scores associating with those

boxes to do the object detection. This network mainly consists of three modules: ARM, ODM, and TCB. ARM provides better initialization for regressors and can be treated as RPN in Faster R-CNN (Ren et al., 2015). ODM whose function is to detect the objects performs as one-stage multiple-scale detector SSD (Liu et al., 2016). TCB which is similar to the upsampling process in FPN (Lin et al., 2017a) connects ARM and ODM by combining the deep feature maps with the shallow feature maps via deconvolution. TCB serves as a bridge between ARM and ODM and it takes the feature maps associated with anchors and the transferred higher-level feature maps to increase detection accuracy. TCB just deconvolves higher-level feature maps to match the size and dimension of the current feature maps and then simply add the current feature map and the higher-level feature maps element-wise. Thus deep feature maps with more semantic information and shallow feature maps with more details can be combined together to enhance the performance of the detector.

Although it has two steps just like the two-stage object detectors, RefineDet is a one-stage detector since it closely combines the two steps without the process in two-stage detectors, such as RoIPooling. In addition, RefineDet utilizes VGG-16 (Simonyan & Zisserman, 2014) as the backbone which is the same as Faster R-CNN (Ren et al., 2015) and SSD (Liu et al., 2016). Employing more advanced networks as the backbone may boost the performance, but for convenience, VGG-16 was harnessed in our experiment.

2.3.2 Kalman Filter

The discrete Kalman filter is employed for location estimation. In Kalman filter, a motion state variable is defined as $x = \{a, b, u, v\}$, where $\{a, b\}$ could be the coordinates of the object, and $\{u, v\}$ could be the velocity of the object along the two directions. From (Welch et al., 1995), the state at each frame k is using equation 2.1 and the measurement is using equation 2.2. $w(k)$ and $v(k)$ are prediction noise and measurement noise, respectively, which are assumed normal distributions with covariance Q and R. The transition matrix $A \in n \times n$ calculates *a priori* estimate at the current state from the previous state. H is denoted as the observation matrix. The equations

are from (Welch et al., 1995).

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \tag{2.1}$$

$$z_k = Hx_k + v_k \tag{2.2}$$

The update of the Kalman filter includes two updates: Time Update and Measurement Update, as shown below.

Time Update:

2.3.2.1 a priori state estimate

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1} \tag{2.3}$$

2.3.2.2 a priori estimate error covariance

$$P_k^- = AP_{k-1}A^T + Q \tag{2.4}$$

Measurement Update:

2.3.2.3 Kalman Gain

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \tag{2.5}$$

2.3.2.4 a posteriori state estimate with measurement z_k

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \tag{2.6}$$

2.3.2.5 a posteriori estimate error covariance

$$P_k = (I - K_k H)P_k^- \tag{2.7}$$

$\hat{x}_k^-$ is *a priori* motion state estimate and $\hat{x}_k$ is *a posteriori* motion state estimate given the measurement information. P_k^- represents *a priori* estimate error covariance and *a posteriori* estimate

error covariance is represented by P_k. We refer to (Welch et al., 1995) for more details about those equations.

We integrate the object detectors and the Kalman filter together for golf ball tracking. First, we employ the detection model to localize the ball and its coordinates in the cropped first frame, whose center is the ground truth coordinate in the first frame. By utilizing the ground truth coordinate in the first frame, we can make sure that the cropped image of the first frame contains the golf ball. With the detection results in the first frame, the measurement update can be implemented easily. After that, the time update will predict the next coarse coordinate of the golf ball from the results of the previous measurement update. Then we crop the next frame centered at the predicted coarse coordinates. The cropped image is sent to the detector to do the detection again. The process will be repeated until the last frame has been detected.

2.4 Dataset

Deep learning based models need a large collection of data for training and validation, especially for data that are taken from different viewing points, light conditions, and backgrounds. Most of the data are collected from golf tournaments online and the rest are taken by us. The original data are in video format while we converted those videos into image sequences for labeling, training, and testing. The golf videos are in high resolution and taken using slow motion, to reduce motion blur. The whole dataset consists of 2169 golf images, where 1699 images are utilized as the dataset for detection and the rest 470 images are used for tracking. Details of the dataset are shown in Table 2.1.

Table 2.1: Dataset split used for training and test

	No. Images	No.Patches
Training Set	1356	11030
Test Set	343	2791
Tracking Dataset	470	3615

2.4.1 Detection Dataset

To train and validate the detection models, 1699 golf ball images are employed, and a great portion of them are collected from golf tournaments online and the rest are taken by ourselves. We manually label the bounding box of the ball in each image and crop out the 416×416 image patch that contains the ball as a training example. Since we only have a limited number of images, we further augment the dataset by shifting the location of the cropped patches. As shown in Fig. 2.2, based on the location of the ball, we shift the location to the up, down, left, and right by 100 pixels to generate 9 patch samples, with each patch the same size of 416×416. Then, we split the collected images into training and test sets at a ratio of 80:20. The split is based on the original images to ensure the augmented patches in the test set have never been seen in the training set.

2.4.2 Tracking Dataset

In order to test the tracking algorithm and detectors for unseen data, we collected another 17 short swing videos from the golf ball being hit until it flew out of view and 5 golf putting videos. We labeled the ground truth and generated the image patches in the same way as discussed above. We utilize the generated patches to test the generalization ability of the trained detection models on unseen data. The detection results are illustrated in Table 2.4. We use the entire video sequence to test the performance of the tracking algorithm. However, some of the short videos have only a few frames, we only choose 8 swing video sequences and 5 putting video sequences with relatively more frames so that we can evaluate the tracking performance, the selected videos and frame numbers are shown in Table 2.2. Please note that although the original images in the dataset are of high resolution, the golf balls are very small and only occupy a small number of pixels, most of them are only around 10×10 pixels, which makes them very hard to detect and track.

Figure 2.2: An original image and 9 generated training patches. We shift the location to the up, down, left, and right by 100 pixels to generate 9 patches based on the location of the ball. The size of each cropped image patch is 416×416.

Table 2.2: Tracking Dataset

	No. Images
Golf_Swing_1	19
Golf_Swing_2	33
Golf_Swing_3	16
Golf_Swing_4	28
Golf_Swing_5	13
Golf_Swing_6	35
Golf_Swing_7	22
Golf_Swing_8	97
Golf_Put_1	35
Golf_Put_2	33
Golf_Put_3	25
Golf_Put_4	35
Golf_Put_5	14
Total	405

2.5 Experimental Evaluations

In our experiments, YOLOv3 and YOLOv3 tiny are implemented using C++. The rest of the detectors are implemented using Python3 with OpenCV and PyTorch framework. The details are given below:

- **Faster R-CNN:** PyTorch, cuDNN, and CUDA
- **YOLOv4:** PyTorch, cuDNN, and CUDA
- **SSD:** PyTorch, cuDNN, and CUDA
- **YOLOv3 & YOLOv3 tiny:** Darknet, cuDNN, and CUDA
- **RefineDet:** PyTorch, cuDNN, and CUDA

The experiment is run on a single Nvidia Titan XP GPU, which has a memory of 12 GB. To thoroughly evaluate the performance, we will evaluate our detectors and tracker separately.

2.5.1 Detector Evaluations

Object detection includes both object localization and object classification. To thoroughly evaluate the performance of an object detector, object localization and classification are utilized as assessments. Commonly used classification evaluation metrics are accuracy, precision, and F1 score. Localization is actually a regression problem. For regression evaluation, MSE (Mean Squared Error) and RMSE (Root Mean Squared Error) are widely adopted. To better evaluate the performance of an object detector, AP score, which assesses both classification accuracy and localization accuracy, was employed in our experiment. The AP metrics in this chapter are AP-25 and AP-50. AP-25 represents AP at IoU threshold 0.25 and AP-50 stands for AP at IoU threshold 0.5. AP-50 is the commonly utilized metric for object detection. While in our experiment, we found that with the small size of the golf ball and error in labeling, AP-50 is too strict to evaluate our detectors. Thus AP-25 and AP-50 are both harnessed to evaluate our detectors.

We compared the performance of six classical detection models in the experiment: YOLOv3 & YOLOv3 tiny (Redmon & Farhadi, 2018), YOLOv4 (Bochkovskiy et al., 2020), Faster R-CNN (Ren et al., 2015), SSD (Liu et al., 2016) and RefineDet (Zhang et al., 2018). The performance of these detectors on the test set is shown in Table 2.3. It is obvious that Faster R-CNN and YOLOv4 achieve a similar mAP score that is much higher than other detectors. The inference time of Faster R-CNN and YOLOv4 is also very close to each other in this experiment. YOLOv3 tiny achieves the shortest inference time, which is 4 to 12 times shorter than other detectors.

Table 2.3: Detection Performance on the Test Set

Method	mAP@.25	mAP@0.5	Inference Time	fps
Faster R-CNN	98.3%	95.9%	36.00 ms	27.78
YOLOv4	99.3%	95.6%	35.84 ms	27.90
YOLOv3	95.6%	88.2%	17.96 ms	55.67
YOLOv3 tiny	92.3%	84.2%	2.79 ms	357.85
SSD	95.8%	78.3%	11.00 ms	90.91
RefineDet	90.0%	81.5%	20.00 ms	50

In order to test the generalization ability of our trained detection models for new data, we performed the same detection experiments on the image patches generated by all tracking sequences.

The results are shown in Table 2.4. It is evident that YOLOv4 achieves the best precision in terms of mAP score, while the result of Faster R-CNN is comparable to YOLOv4. However, the performance of all models drops by a large margin compared to the results from the test set. This is mainly due to two reasons: (1) the original training set is relatively small and training samples do not have sufficient representativeness; and (2) the distributions of the video dataset and the original training set are different because they are collected from different sources. Nonetheless, the overall performance is still acceptable and this can be demonstrated from the tracking performance. The inference time and detection speed of the tracking dataset remain the same as the test set as we are using the same size of image patches.

Table 2.4: Detector Performance on Tracking Dataset

Method	mAP@.25	mAP@0.5
Faster R-CNN	89.0%	84.0%
YOLOv4	92.0%	84.4%
YOLOv3	82.3%	71.4%
YOLOv3 tiny	83.4%	63.7%
SSD	84.8%	65.0%
RefineDet	82.0%	64.5%

2.5.2 Tracking Evaluation

We adopted the evaluation metrics from (Wu et al., 2013) to evaluate our tracking results. We use the selected 13 sequences in Table 2.2 to evaluate our algorithm. The results are evaluated in terms of CLE (center location error), precision, and SR (success rate). CLE is calculated by the Euclidean distance between the center location of the tracked object and the ground truth. Precision describes the percentage of the images whose predicted location is within a certain threshold of the ground truth location.

Another evaluation metric is the bounding box overlap. Tracking result is considered successful if $|\frac{x_t \cap x_g}{x_t \cup x_g}| > \theta$. $\theta \in [0,1]$. x_t is denoted as the tracking bounding box, while x_g is denoted as ground truth bounding box. The number of successful frames are counted in order to measure the performance of a sequence. Rather than assigning a certain threshold, we swept the threshold from

Table 2.5: Tracker Precision Rate Evaluation

Sequence	Faster R-CNN			YOLOv3			YOLOv3 tiny			SSD			RefineDet		
	CLE_1	CLE_2	CLE_5	CLE_1	CLE_2	CLE_5	CLE_1	CLE_2	CLE_5	CLE_1	CLE_2	CLE_5	CLE_1	CLE_2	CLE_5
Golf_Swing_1	31.6%	63.2%	89.5%	15.8%	36.8%	68.4%	15.8%	52.6%	68.4%	26.3%	57.9%	84.2%	5.3%	10.5%	21.1%
Golf_Swing_2	36.4%	75.8%	93.9%	30.3%	45.5%	90.9%	24.2%	48.5%	81.8%	15.2%	48.5%	90.9%	36.4%	78.8%	90.9%
Golf_Swing_3	31.3%	75.0%	81.3%	0%	18.8%	25.0%	31.3%	56.3%	56.3%	6.3%	6.3%	43.8%	12.5%	18.8%	43.8%
Golf_Swing_4	17.9%	32.1%	50.0%	7.1%	25.0%	57.1%	3.6%	21.4%	57.1%	0%	0%	0%	7.1%	21.4%	78.6%
Golf_Swing_5	15.4%	30.8%	61.5%	23.1%	46.2%	76.9%	0%	38.5%	92.3%	0%	30.8%	92.3%	7.7%	23.1%	53.9%
Golf_Swing_6	37.1%	65.7%	97.1%	0%	20.6%	82.4%	14.7%	29.4%	64.7%	5.7%	31.4%	88.6%	11.4%	34.3%	94.3%
Golf_Swing_7	36.4%	95.5%	100.0%	4.76%	33.3%	71.4%	23.8%	57.1%	95.2%	0%	0%	0%	22.7%	50.0%	86.4%
Golf_Swing_8	56.7%	81.4%	100.0%	17.5%	50.5%	92.8%	10.3%	39.2%	85.6%	9.3%	33.0%	97.9%	38.1%	68.0%	87.6%
Golf_Put_1	34.3%	82.9%	91.4%	2.9%	37.1 %	94.3%	5.71%	31.4%	88.6%	0%	8.6%	94.3%	2.9%	11.4%	94.3%
Golf_Put_2	36.4%	78.8%	100.0%	27.3%	66.7%	100.0%	12.1%	54.6%	97.0%	0%	9.1%	97.0%	6.1%	21.2%	97.0%
Golf_Put_3	36.0%	80.0 %	100.0%	12.4%	40.0%	100.0%	8.0%	32.0%	96.0%	0%	0%	96.0%	0%	32.0%	88.0%
Golf_Put_4	22.9%	71.4 %	100.0%	17.1%	42.9%	97.1%	2.9%	37.1%	91.4%	2.9%	11.4%	97.1%	5.7%	14.3%	85.7%
Golf_Put_5	57.1%	85.7%	100.0%	7.14%	42.9%	100.0%	14.3%	35.7%	92.9%	0%	14.3%	78.6%	7.1%	21.4%	78.6%
Average	34.6%	70.1%	89.6%	12.7%	38.9%	81.3%	12.8%	41.1%	82.1%	5.0%	19.3%	73.9%	12.5%	31.2%	76.9%

0 to 1 and used AUC (area under the curve) to evaluate the tracker.

Table 2.6: Tracker Success Rate Evaluation

Sequence	Faster R-CNN	YOLOv3	YOLOv3 tiny	SSD	RefineDet
Golf_Swing_1	54.0%	39.4%	36.8%	43.3%	15.0%
Golf_Swing_2	61.2%	48.2%	45.0%	54.0%	60.9%
Golf_Swing_3	55.6%	15.5%	36.7%	23.8%	27.2%
Golf_Swing_4	29.2%	27.4%	26.1%	0.0%	39.5%
Golf_Swing_5	41.8%	47.3%	51.5%	54.6%	36.8%
Golf_Swing_6	73.4%	52.0%	47.2%	55.8%	63.1%
Golf_Swing_7	67.1%	43.6%	55.4%	0.0%	58.9%
Golf_Swing_8	77.1%	60.9%	52.0%	61.0%	67.7%
Golf_Put_1	68.3%	62.2%	61.9%	56.2%	61.1%
Golf_Put_2	74.7%	72.4%	69.1%	54.1%	63.7%
Golf_Put_3	83.6%	73.9%	71.7%	63.9%	68.0%
Golf_Put_4	77.6%	71.4%	66.1%	62.2%	61.1%
Golf_Put_5	70.4%	67.9%	67.6%	58.4%	61.4%
Average	64.3%	52.5%	52.9%	41.8%	52.6%

2.5.3 Tracking Results

Since the detection results and the inference time of Faster R-CNN (Ren et al., 2015) and YOLOv4 (Bochkovskiy et al., 2020) are similar, we only compare the tracking results based on YOLOv3 and YOLOv3 tiny (Redmon & Farhadi, 2018), Faster R-CNN (Ren et al., 2015), SSD (Liu et al., 2016), and RefineDet (Zhang et al., 2018) in this experiment. Table 2.5 shows the precision

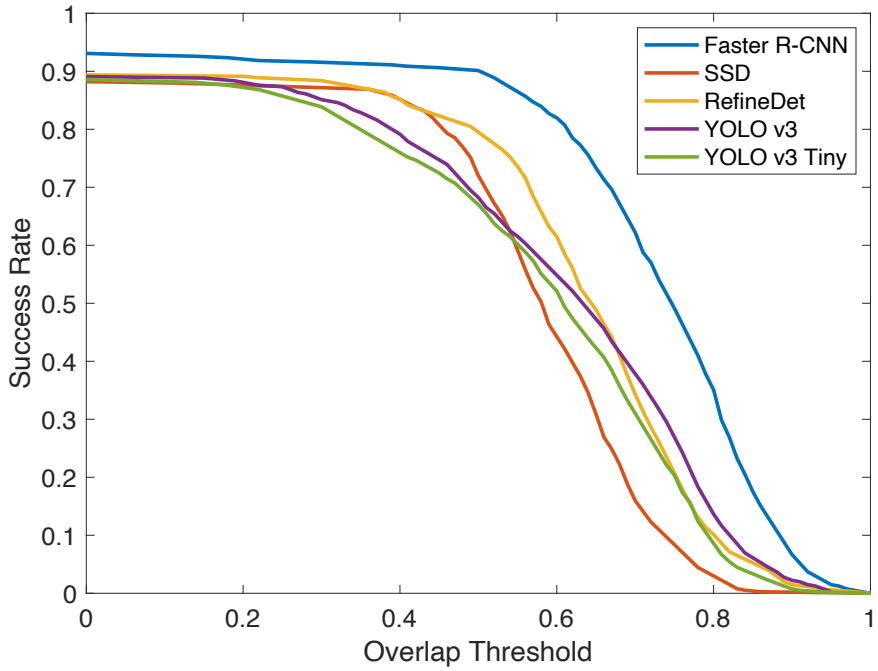

Figure 2.3: Success Rate. When the overlap of the ground truth bounding box and the detected bounding box is larger than some threshold in one frame, the tracking result in this frame is denoted as a success. When the overlap threshold increases, which means the accuracy requirement is higher, the success rate drops.

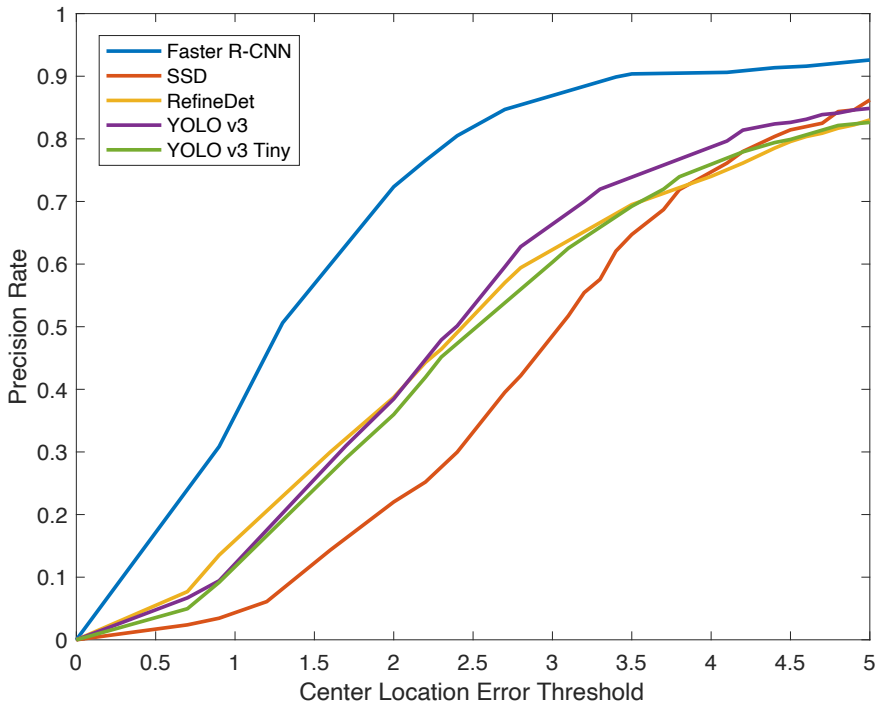

Figure 2.4: Precision Rate. Low CLE represents the overlap between the detected bounding box and the ground truth bounding box is high, which indicates high accuracy of the detection result. The precision rate increases as the CLE threshold increases.

results of all 5 models with CLE being 1, 2, and 5, respectively. It is obvious that Faster R-CNN has the best performance under the CLE metrics. Table 2.6 demonstrates the success rate of the trackers on all sequences. The overall success rate and precision rate of 5 models are depicted in Fig. 2.3 and Fig. 2.4, respectively. Fig. 2.3 and Fig. 2.4 demonstrate the performance of the trackers using the above evaluation metrics. According to Fig. 2.3 and Fig. 2.4, Faster R-CNN has the best performance for success rate and the precision rate at all overlap thresholds and center location error threshold. We can also see that the two-stage Faster R-CNN achieves the dominant performance in terms of the mAP score, while SSD does not seem to perform well. The performance of YOLOv3, YOLOv3 tiny, and RefineDet is similar. However, YOLOv3 and YOLOv3 tiny, SSD and RefineDet achieve competing accuracy with a much shorter inference time. The best-buy detector is YOLO v3 tiny, which achieved the highest speed of near 358 frames per second (fps) with reasonable accuracy.

Table 2.7 demonstrates the real frame rate of Faster R-CNN, SSD, and RefineDet in fps (frames per second) of different tracking sequences. Since YOLOv3 and YOLOv3 tiny are written in C++ using different computing platforms, we did not test their speed in this experiment. However, based on the detection speed, we infer the fps of YOLOv3 would be in the range of 40-55, and the fps of YOLOv3 tiny could be in the range of 100-180, making it an excellent model for real-world applications.

Fig. 2.5 shows the tracking results with Faster R-CNN of Golf_6 and Golf_7 video sequences, respectively. Fig. 2.6 illustrates the tracking results using Faster R-CNN of two putting sequences. We can see that the tracking results are visually very accurate. While most classical tracking algorithms based on the entire frames fail to track the ball due to the small size and fast movement of the ball.

2.6 Experiment and Error Analysis

In the previous report, most object detectors do not perform well on small object detection. One reason is that most detectors are trained on the datasets, like PASCAL VOC (Everingham et al.,

Figure 2.5: Two sample tracking results based on Faster R-CNN for swing test sequences Golf_6 (upper) and Golf_7 (lower). The white golf ball indicates the ground truth, and the red bounding boxes represent the tracking results.

Table 2.7: Tracker Frame Rate Evaluation (FPS)

Sequence	Faster R-CNN	SSD	RefineDet
Golf_Swing_1	24.27	42.30	38.78
Golf_Swing_2	24.93	49.65	43.84
Golf_Swing_3	24.94	46.80	34.22
Golf_Swing_4	20.63	46.16	30.83
Golf_Swing_5	24.10	43.22	38.79
Golf_Swing_6	24.83	38.19	43.91
Golf_Swing_7	24.33	34.98	42.00
Golf_Swing_8	25.64	49.76	43.49
Golf_Put_1	25.58	53.40	44.54
Golf_Put_2	26.18	56.61	46.42
Golf_Put_3	26.04	48.32	44.06
Golf_Put_4	26.39	56.03	45.34
Golf_Put_5	25.84	42.74	46.58
Average	24.90	46.78	41.75

2015) (Everingham et al., 2010) or COCO (Lin et al., 2014), where the size of the objects varies greatly, including plenty of large objects and middle-sized objects. The detectors are trained to detect objects of different sizes, rather than solely focusing on small objects. Therefore, the performance of small object detection is not appealing.

In this chapter, we only focus on detecting one small object-the golf ball. Thus, we can customize the detectors for this specific task. For instance, the anchor scales in the original Faster R-CNN paper are 128, 256, and 512, while we change them to 8, 16, and 32 in our experiment to better accommodate the small balls in the dataset. In addition, we perform the detection on the cropped patches instead of on the original high-resolution image, which makes the ball relatively “larger" and easier to detect than taking the entire image as input. To ensure the cropped patch contains the ball, we employ the Kalman filter to predict the location of the ball. As a result, we have achieved relatively good results in both detection and tracking.

The golf ball itself is an extremely small object. As it flies away from the camera, it becomes smaller and smaller, which results in significant detection errors. In addition, since the ball only occupies a small area, it is hard to label it accurately, and a one-pixel annotation error may have a huge influence on the precision evaluation of the detection results.

Figure 2.6: Two sample tracking results based on Faster R-CNN for putting test sequences Put_3 (upper) and Put_5 (lower). The white golf ball indicates the ground truth, and the red bounding boxes represent the tracking results.

Figure 2.7: The ground truth and the superimposed tracking results. The green bounding boxes are the ground truth and the red bounding boxes are the tracking results. Please note that there is one falsely tracked ball position.

As in the example shown in Fig. 2.7, the labeled ground truth is represented by the green bounding box, while the tracked results are in red. The size of the golf ball in the first frame is around 27×27. It was labeled as 26×26 with only 1-pixel error. Suppose the prediction is perfect, while the 1-pixel labeling error could lower the IOU by around 7.5%. When the ball is away from the camera, its size will decrease to 6×6 pixels. In this case, the 1-pixel error could result in 33% IOU drop. Based on this analysis, we can see that labeling errors or detection errors will have a significant influence on the IOU calculation of small objects, although this may not be an issue for large objects. Thus, it is reasonable to take the mAP score with IOU 0.25 into consideration for small objects.

2.7 Conclusion

The chapter has proposed and implemented a practical approach for real-time golf ball detection and tracking. The proposed solution is based on object detection and the discrete Kalman filter. For object detection, we have implemented the convolutional neural networks, including YOLOv3,

YOLOv3 tiny, YOLOv4, Faster R-CNN, SSD, and RefineDet. For tracking, we have tested 5 models and Faster R-CNN yields the best performance on success rate and precision rate, while YOLOv3 tiny achieves the fastest inference rate with competing accuracy. The other 3 models yield acceptable precision and frame rate. Extensive experimental evaluations have demonstrated the effectiveness of the proposed approach, while most classical methods may fail for the detection and tracking of such a small object. The performance could be further improved if a larger and more representative training dataset is available.

Chapter 3

Six-channel Image Representation for Cross-domain Object Detection

Abstract

Most deep learning models are data-driven and their excellent performance is highly dependent on the abundant and diverse datasets. However, it is very hard to obtain and label the datasets of some specific scenes or applications. If we train the detector using the data from one domain, it cannot perform well on the data from another domain due to domain shift, which is one of the big challenges of most object detection models. To address this issue, some image-to-image translation techniques have been employed to generate some fake data of some specific scenes to train the models. With the advent of Generative Adversarial Networks (GANs), we could realize unsupervised image-to-image translation in both directions from a source to a target domain and from the target to the source domain. In this study, we report a new approach to making use of the generated images. We propose to concatenate the original 3-channel images and their corresponding GAN-generated fake images to form 6-channel representations of the dataset, hoping to address the domain shift problem while exploiting the success of available detection models. The idea of augmented data representation may inspire further study on object detection and other applications.

3.1 Introduction

Computer vision has progressed rapidly with deep learning techniques and more advanced and accurate models for object detection, image classification, image segmentation, pose estimation, and tracking emerging almost every day (Ma et al., 2020a; Zhang et al., 2020c; Wu et al., 2019). Even though computer vision enters a new era with deep learning, there are still plenty of unsolved problems and domain shift is one of them. Albeit CNN models are dominating computer vision, their performances often become inferior when testing some unseen data or data from a different domain, which is denoted as domain shift. Since most deep learning models are data-driven and the high-accurate performance is mostly guaranteed by the enormous amount of various data, domain shift often exists when there are not enough labeled specific data but we have to test those kinds of data in the testing set. For instance, although we only detect cars on the roads, training the models on day scenes cannot guarantee an effective detection of cars in night scenes. We might have to utilize enough datasets from night scenes to train the models, nonetheless, sometimes the datasets from some specific scenes are rare or unlabeled, which makes it even more difficult to mitigate the domain shift effect.

To mitigate the situation where some kinds of training data are none or rare, The image-to-image translation that could translate images from one domain to another is highly desirable. Fortunately, with the advent of Generative Adversarial Networks (GANs) (Goodfellow et al., 2014), Some researchers aim to generate some fake datasets in specific scenes using GAN models to overcome the lack of data. With some unpaired image-to-image translation GAN models (i.e., CycleGAN (Zhu et al., 2017a)), it can not only translate images from the source domain to the target domain, but also translate images from the target domain to the source domain, and the entire process does not require any paired images, which make it ideal for real-world applications.

The GAN models for image-to-image translation can generate the corresponding fake images of the target domain from the original images of the source domain in the training dataset, and we can utilize the GAN-generated images to train object detection models and test on images of target domain (Arruda et al., 2019). Since we expect to solve cross-domain object detection problems,

after pre-processing the data and generating the fake images with image-to-image translation models, the generated data has to be fed into the object detection models to train the model and the trained model could demonstrate its effectiveness through testing the data from the target domain. Employing GAN-generated fake images to train the detection models to guarantee the domain of the training data and testing data being the same illustrated the effectiveness of the approach and the detection performance was boosted for the scenario where the training data for the detection models is from one domain while the testing data is in another domain (Arruda et al., 2019).

Instead of simply utilizing the fake images to train the model, we propose to solve the problem from a new perspective by concatenating the original images and their corresponding GAN-translated fake images to form new 6-channel representations. For instance, if we only have source domain images but intend to test our model on unlabeled images in the target domain, what we did was training the image-to-image translation model with source domain data and target domain data. And then we could employ the trained image translation model to generate the corresponding fake images. Since some image-to-image translation models (Zhu et al., 2017a) could translate images in both directions, we are able to acquire the corresponding fake data for the data from both the source domain and target domain. Thus, both training images and testing images would be augmented into 6-channel representations by concatenating the RGB three channels of the original images with those from the corresponding fake images. Then we can train and test the detection models using available detection models, the only difference is the dimension of the kernel of the CNN models for detection in the first layer becomes 6 instead of 3. The process of training and testing the proposed method is depicted in Fig. 3.1.

3.2 Related Work

Image-to-image translation is a popular topic in computer vision (Xu et al., 2019b,a). With the advent of Generative Adversarial Networks (Goodfellow et al., 2014), it could be mainly categorized as supervised image-to-image translation and unsupervised image-to-image translation (Alotaibi, 2020). The supervised image-to-image translation models such as pix2pix (Isola et al.,

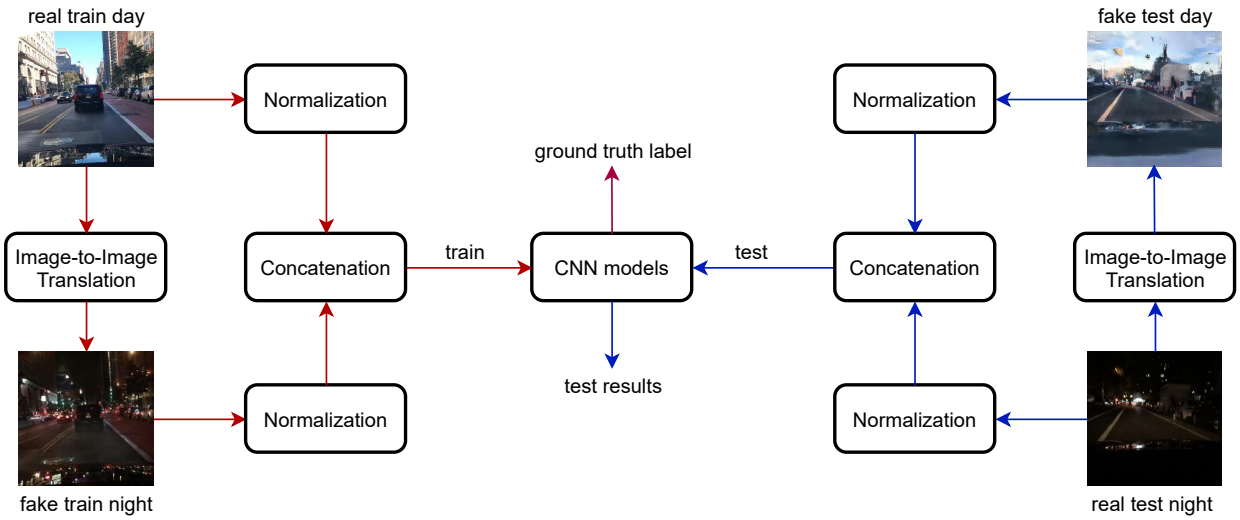

Figure 3.1: The flow chart of the proposed 6-channel image augmentation approach for training and testing CNN-based detection models.

2017) and BicycleGAN (Zhu et al., 2017b), require image pairs from two or more domains (i.e., the exact same image scenes from day and night), which are extremely expensive and unrealistic to be acquired in the real world. Perhaps the quality of the translated images is sometimes beyond expectations, they are not ideal for real-world applications.

The unsupervised image-to-image translation models can be divided as cycle consistency based models (i.e., CycleGAN (Zhu et al., 2017a), DiscoGAN (Kim et al., 2017), DualGAN (Yi et al., 2017)) which introduce cycle consistency losses, autoencoder based models (i.e., UNIT (Liu et al., 2017)) combined with autoencoder (Kingma & Welling, 2013), and recent disentangled representation models (i.e., MUNIT (Huang et al., 2018), DIRT (Lee et al., 2018)). Since the unsupervised image-to-image translation models only require image sets from two or more domains and do not necessitate any paired images, which are arduous to collect and annotate, they are often leveraged to generate some fake data in the target domain and applied to other computer vision tasks such as object detection and image classification. Among those unsupervised image-to-image translation models, CycleGAN (Zhu et al., 2017a) is frequently utilized as the image-mapping model to generate some fake data to be employed in some cross-domain problems (Inoue et al., 2018) (Arruda et al., 2019).

Object detection addresses the problem that detecting semantic instances on digital images or videos. The fundamental purpose of object detection is to classify the objects shown on the images or videos and simultaneously locate those objects by coordinates (Ma et al., 2020b). The applications of object detection are in various fields such as medical image analysis (Mo et al., 2018), self-driving cars, pose estimation, segmentation, etc.

From the perspective of stages, the object detectors are categorized into two types: one-stage detectors and two-stage detectors. For two-stage object detectors such as Faster R-CNN (Ren et al., 2016), MS-CNN (Cai et al., 2016), R-FCN (Dai et al., 2016), FPN (Lin et al., 2017a), these models are often comprised of a region proposal network as the first stage that selects the candidate anchors which have high probabilities to contain objects and a detection network as the second stage that classify the objects to be contained by these candidates and further do the bounding box regression for these candidates to refine their coordinates and finally output the results. For one-stage object detectors like SSD (Liu et al., 2016), YOLOv1-v4 (Redmon et al., 2016) (Redmon & Farhadi, 2017) (Redmon & Farhadi, 2018) (Bochkovskiy et al., 2020), RetinaNet (Lin et al., 2017b), these detectors often directly classify and regress the pre-defined anchor boxes instead of choosing some candidates. Thus the two-stage models often outperform the one-stage counterparts, while one-stage models frequently have a faster inference rate than two-stage approaches.

Due to the various sizes and shapes of the objects, some models (Liu et al., 2016) (Lin et al., 2017a) (Lin et al., 2017b) (Zhang et al., 2018) design anchor boxes on different levels of feature maps (the pixels on lower level feature maps have a small receptive field and the pixels on higher-level feature maps have large receptive field) so that the anchors on lower level features are responsible for the relative small objects and the anchors on higher-level features are in charge of detecting relatively large objects. The middle-sized objects are perhaps recognized by the middle-level feature maps.

The aforementioned detection models are anchor-based so we have to design pre-defined anchor boxes for these models. In recent years, some anchor-free models (Zhu et al., 2019) (Zhou

et al., 2019) (Duan et al., 2019) (Tian et al., 2019) (Law & Deng, 2018) are attracting great attention for their excellent performance without any pre-defined anchor boxes. Some of them are even dominating the accuracy on COCO benchmark (Lin et al., 2014). Since a large number of anchors has to be generated for some anchor-based models and most of them are useless because no object is contained in the majority of anchors, anchor-free models might predominate in the designs of object detectors in the future. Recently, the transformer (Vaswani et al., 2017) is applied successfully to object detection (Carion et al., 2020), which is an anchor-free model with attention mechanisms.

Nonetheless, many problems have not been well solved in this field, especially in cross-domain object detection. Since modern object detectors are based on deep learning techniques and deep learning is data-driven so that the performance of modern object detectors is highly dependent on how much annotated data can be employed as the training set. Cross-domain issues arise when there are not enough labeled training data that have the same domain as the testing data, or the dataset is diverse or composed of various datasets of different domains in both training and testing data.

Domain Adaptive Faster R-CNN (Chen et al., 2018) explores the cross-domain object detection problem based on Faster R-CNN. By utilizing Gradient Reverse Layer (GRL) (Ganin & Lempitsky, 2015) in an adversarial training manner which is similar to Generative Adversarial Networks (GAN) (Goodfellow et al., 2014), this chapter proposes an image-level adaptation component and an instance-level adaptation component, which augment the Faster R-CNN structure to realize domain adaptation. In addition, a consistent regularizer between those two components is to alleviate the effects of the domain shift between different datasets such as KITTI (Geiger et al., 2013), Cityscapes (Cordts et al., 2016), Foggy Cityscapes (Sakaridis et al., 2018), and SIM10K (Johnson-Roberson et al., 2016).

Universal object detection by domain attention (Wang et al., 2019) addresses the universal object detection of various datasets by attention mechanism (Vaswani et al., 2017). The universal object detection is arduous to realize since the object detection datasets are diverse and there exists

a domain shift between them. The paper (Hu et al., 2018) proposes a domain adaption module that is comprised of a universal SE adapter bank and a new domain-attention mechanism to realize universal object detection. (Inoue et al., 2018) deals with cross-domain object detection that instance-level annotations are accessible in the source domain while only image-level labels are available in the target domain. The authors exploit an unpaired image-to-image translation model (CycleGAN (Zhu et al., 2017a)) to generate fake data in the target domain to fine-tune the trained model which is trained on the data in the source domain. Finally, the model is fine-tuned again on the detected results of the testing data (pseudo-labeling) to make the model even better.

The study (Arruda et al., 2019) utilizes CycleGAN (Zhu et al., 2017a) as the image-to-image translation model to translate the images in both directions. The model trained on the fake data in the target domain has better performance than that trained on the original data in the source domain on testing the test data from the target domain. The dataset we employ in this chapter is from (Arruda et al., 2019) and we follow exactly the same pre-processing procedure to prepare the dataset. In the following, we will discuss our proposal that utilizes concatenated image pairs (real images and corresponding fake images) to train the detection model and compare it to the corresponding approach from (Arruda et al., 2019).

3.3 Proposed Approach

The framework of our proposed method is depicted in Fig. 3.1. In our implementation, we employ CycleGAN for image-to-image translation, which is trained with the data from the source domain (i.e., day images) and the data from the target domain (i.e., night images). First, the fake data (target domain) is generated from the original data (source domain) via the trained image-to-image translation model (i.e., generating the fake night images from the real day images). Then, the real and fake images are normalized and concatenated (i.e., concatenating two 3-channel images to form a 6-channel representation of the image). Finally, the concatenated images are exploited to train the CNN models. During the stage of testing, the test data is processed in a similar way as the training data to form concatenated images and sent to the trained CNN model for detection.

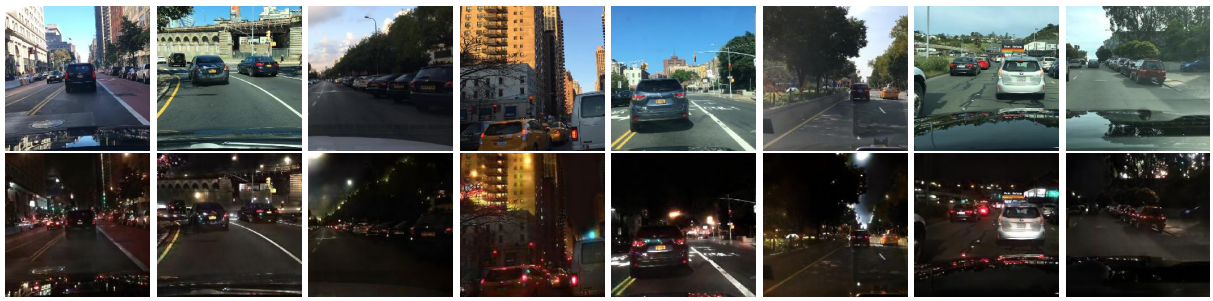

Figure 3.2: Several samples of original-day images (1st row) and their corresponding GAN-generated fake-night images (2nd row).

3.3.1 Image-to-Image Translation

To realize the cross-domain object detection, we have to collect and annotate the data in the target domain to train the model. While it is difficult to acquire the annotated data in the target domain, image-to-image translation models provide an option to generate fake data in the target domain.

In our experiment, we employed an unpaired image-to-image translation model: CycleGAN (Zhu et al., 2017a). CycleGAN is an unsupervised image-to-image translation that only requires images from two different domains (without any image-level or instance-level annotations) to train the model. Furthermore, unpaired translation illustrates that the images from two domains do not need to be paired which is extremely demanding to be obtained. Last but not least, the locations and sizes of the objects on the images should be the same after the image-to-image translation so that any image-level labels and instance-level annotations of the original images can be utilized directly on the translated images. This property is extraordinarily significant since most CNN models are data-driven and the annotations of the images are indispensable to successfully train the supervised CNN models (i.e., most object detection models). Unpaired image-to-image translation models such as CycleGAN (Zhu et al., 2017a) can translate the images in two directions without changing the key properties of the objects on the images. Thus the annotations such as coordinates and class labels of the objects on the original images can be smoothly exploited in the fake translated images. As manually annotating the images is significantly expensive, by image-to-image translation, the translated images would automatically have the same labels as their original counterparts, which to some extent makes manually annotating images unnecessary.

Figure 3.3: Several samples of original-night images (1st row) and their corresponding GAN-generated fake-day images (2nd row).

3.3.2 CNN Models

In Fig. 3.1, the CNN model can be any CNN-based object detection model, where the dimension of the convolutional kernel in the first layer is changed from 3 to 6. In our implementation, we employ Faster R-CNN (Ren et al., 2016) for detection, and we use ResNet-101 (He et al., 2016) as the backbone network for the detection model.

Faster R-CNN is a classic two-stage anchor-based object detector that is comprised of Region Proposal Network (RPN) and a detection network. Since it is an anchor-based model, we have to design some pre-defined anchor boxes on the feature maps. Typically, 9 anchors with 3 different sizes and 3 different aspect ratios are designed to act as the pre-defined anchor boxes on each location of the feature maps. The objective of RPN is to select some region proposals with a high probability of containing objects from the pre-defined anchors and further refine their coordinates. Each pre-defined anchor would be associated with a score indicating the probability of that anchor box containing an object. Only the anchor boxes with associated scores higher than some threshold can be selected as region proposals and those region proposals are further refined by RPN and later fed into the detection network.

The purpose of the detection network is to receive the region proposals selected and refined by RPN and finally do the classification for each rectangle proposal and bounding box regression to improve the coordinates of the box proposals. Since the region proposals may have various sizes and shapes, more accurately, the number of elements each proposal has might be varying. To guarantee the region proposals are fed into the fully connected layers effectively (the fully

connected layer needs the length of input data fixed), the ROI pooling layer is adopted to ensure the size of the input of each proposal to the detection network is fixed. The detection network is simply from Fast R-CNN (Girshick, 2015) that is to classify the object which might be contained by each region proposal and simultaneously refine the coordinates of the rectangle boxes. The output of the Faster R-CNN network is the class of the object each proposal might include and the coordinates of the bounding box for each refined proposal.

3.4 Experiments

In this section, the datasets and the experimental methodology and the parameter settings are elaborated. We conducted some of the experiments from (Arruda et al., 2019) for comparison.

3.4.1 Datasets

We employ the same dataset as (Arruda et al., 2019) in our experiments. The original datasets are from BDD100K (Yu et al., 2018) which is a large-scale diverse dataset for driving scenes. Since the dataset is extremely large and contains high-resolution images and various scenarios on the road and the weather conditions (sunny, rainy, foggy, etc.) (Arruda et al., 2019), the authors only choose the clear or partly cloudy day and night images to demonstrate the domain shift from day to night (Arruda et al., 2019). In addition, all selected images are cropped to 256×256 pixels with proper adjustment. There are a total 12,000 images left and processed (6,000 day images and 6,000 night images). After that, the images are randomly sampled and divided into four sets: train-day, train-night, test-day, and test-night, each of the sets contains 3,000 256×256 images. We harness the set of train-day and train-night to train the CycleGAN model and utilized the trained GAN model to generate fake train-night (from train-day), fake train-day (from train-night), fake test-night (from test-day), and fake test-day (from test-night). Now we have a total of 12,000 real images (3,000 for each set) and 12,000 fake images (3,000 for each set). Then, we can concatenate the real images and their corresponding fake images to generate 6-channel representations that

would be fed into the Faster R-CNN object detector. After choosing and processing the images, the car is the only object on the image to be detected. Some samples of real images and their corresponding GAN-generated fake counterparts are illustrated in Fig. 3.2 and Fig. 3.3.

3.4.2 Experimental Evaluations

Faster R-CNN model is implemented in Python (Yang et al., 2017) with Pytorch 1.0.0 and CycleGAN is implemented in Python (Zhu et al., 2020c) with PyTorch 1.4.0. All experiments are executed with CUDA 9.1.85 and cuDNN 7 on a single NVIDIA TITAN XP GPU with a memory of 12 GB.

The metric we employed is mean Average Precision (mAP) from PASCAL VOC (Everingham et al., 2015), which is the same metric employed in (Arruda et al., 2019). Since the car is the only object to be detected, the mAP is equivalent to AP in this dataset since mAP calculates the mean AP for all classes.

For CycleGAN, the parameters are default values in (Zhu et al., 2020c). For Faster R-CNN, similarly to (Arruda et al., 2019), we utilize pre-trained ResNet-101 (He et al., 2016) on ImageNet (Deng et al., 2009) as our backbone network. We select the initial learning rates from 0.001 to 0.00001 and the experiments are implemented separately for those chosen initial learning rates, but we do not utilize them all for each experiment since our experiments demonstrate that the higher the learning rate we select from above, the better the results would be. In each 5 epochs, the learning rate decays as 0.1 of the previous learning rate. The training process would be executed 20 to 30 epochs, but the results indicate that the Faster R-CNN model converges relatively early on the dataset. Training every 5 epochs, we record the testing results on test data, but we report the best one for each experiment. The model parameters are the same for 6-channel experiments and 3-channel experiments, except for 6-channel experiments, the kernel dimension of the first layer of the Faster R-CNN model is 6 instead of 3. And we just concatenate each kernel by itself to create 6-dimension kernels in the first layer of ResNet-101 backbone for 6-channel experiments. While for 3-channel experiments, we simply exploit the original ResNet-101 backbone as our initial training

parameters.

3.4.3 Experimental Results

First, we implemented the training and testing of the original 3-channel Faster R-CNN model, which is illustrated in Table 3.1. The test set is test-night data which is fixed. With different training sets, the detection results on test night are varying.

Table 3.1: 3-channel detection

Train set	mAP
train-day (3,000 images)	0.777
fake train-night (3,000 images)	0.893
train-night (3,000 images)	0.933
train-day + train-night (6,000 images)	0.941

From Table 3.1 we can see that, for testing the test-night set, the model trained on the fake-train night set is much better than that trained on the original train-day set, which corresponds to the results from (Arruda et al., 2019). These experimental results indicate that if the annotated day images are the only available training data while the test set contains only night images, we could leverage fake night images generated by the image-to-image translation models to train the CNN model. The results are excellent when the model is trained on the train-night set (without domain shift), indicating the domain shift is the most significant influence on the performance of the CNN model in this experiment.

Then we conduct the experiments for our proposed 6-channel Faster R-CNN model, which is shown in Table 3.2. The test data is comprised of test-night images concatenated with corresponding translated fake test-day images. The training sets in Table 3.2 have 6 channels. For instance, train-day in the table indicates train-day images concatenated with corresponding fake train-night images, and train-day plus train-night in the table represents train-day images concatenated with corresponding fake train-night images plus train-night images concatenated with corresponding fake train-day images.

From Table 3.1 and Table 3.2, it is noticeable that even though the model trained on train-day

Table 3.2: 6-channel detection

Train set	mAP
train-day (3,000 6-channel representations)	0.830
train-night (3,000 6-channel representations)	0.931
train-day + train-night (6,000 6-channel representations)	0.938

images concatenated with fake train-night images (6-channel) has a better result with AP 0.830 than that just training on train-day (3-channel) with AP 0.777, it is worse than the model only trained on fake train-night (3-channel) with AP 0.893.

To demonstrate if the 6-channel approach can improve the detection results in the situation where the training set and testing set do not have domain shift, we also performed the experiment that trains the model on the train-night set (3-channel) and tests it on the test-night set. From Table 3.1, the average precision is 0.933, which is pretty high since there is no domain shift between the training data and testing data. Accordingly, we did the corresponding 6-channel experiment which trains on train-night set concatenated with fake train-day set and tests it on test-night images concatenated with fake test-day images. From Table 3.2, the average precision of this 6-channel model is almost the same as its corresponding 3-channel model.

We increase the size of the training data by training the model with the train-day set plus the train-night set and testing it on test-night data. From Table 3.1 and Table 3.2, the result of the 6-channel model also performs similarly to its 3-channel counterpart. More experimental results are shown in Table 3.3, which are from the original 3-channel models. To remove the effect of domain shift, the training set and the testing set do not have domain shift (they are all day images or night images). From Table 3.3, it is obvious that the "quality" shift influences the performance of the models. For instance, the model trained on the original train-day (or train-night) set has better performance on the original test-day (or test-night) set than the GAN-generated fake day (or night) images. Similarly, the model which is trained on GAN-generated fake train-day (or fake train-night) set performs better on the GAN-generated fake test-day (or fake test-night) set than the original test-day (or test-night) set.

Table 3.3: 3-channel extra experiments

Train set	Test set	mAP
train-day	test-day	0.945
	fake test-day	0.789
fake train-day	fake test-day	0.914
	test-day	0.903
train-night	test-night	0.932
	fake test-night	0.859
fake train-night	fake test-night	0.924
	test-night	0.868

3.5 Conclusion

The study has evaluated a 6-channel approach to address the domain-shift issue by incorporating the generated fake images using image-to-image translation. However, we have not achieved the expected results. One possible reason is that the quality of the generated images is inferior compared to the original images, especially the fake day images generated from the data of night scenes, as illustrated in Fig. 3.2 and Fig. 3.3. If we merely concatenate the original high-quality images with their inferior counterparts, the model may treat the low-quality fake image channels as some kind of "noise", and thus, the model could hardly learn more useful information from the concatenated 6-channel representations. Another possible reason is that the domain shift issue may still exist in the combined 6-channel representations, which prevents the model from extracting useful information from the concatenated representations. Moreover, the dataset we used in the experiments only has limited samples, which are insufficient to train the model. We hope the idea of augmented data representation can inspire further investigations and applications.

Chapter 4

Dynamic Label Assignment for Object Detection by Combining Predicted IoUs and Anchor IoUs

Abstract

Label assignment plays a significant role in modern object detection models. Detection models may yield totally different performances with different label assignment strategies. For anchor-based detection models, the IoU (Intersection over Union) threshold between the anchors and their corresponding ground truth bounding boxes is the key element since the positive samples and negative samples are divided by the IoU threshold. Early object detectors simply utilize the fixed threshold for all training samples, while recent detection algorithms focus on adaptive thresholds based on the distribution of the IoUs to the ground truth boxes. In this chapter, we introduce a simple while effective approach to perform label assignment dynamically based on the training status with predictions. By introducing the predictions in label assignment, more high-quality samples with higher IoUs to the ground truth objects are selected as the positive samples, which could reduce the discrepancy between the classification scores and the IoU scores, and generate more high-quality boundary boxes. Our approach shows improvements in the performance of the detection models with the adaptive label assignment algorithm and lower bounding box losses for those positive samples, indicating more samples with higher-quality predicted boxes are selected as positives.

4.1 Introduction

Object detection is a fundamental problem in computer vision that simultaneously classifies and localizes all objects in images or videos (Nguyen et al., 2021; Zhang et al., 2019a; Dewi et al., 2020). With the fast development of deep learning, object detection has achieved great success and been applied to many real-world tasks such as object tracking (Bharati et al., 2018; Zhang et al., 2020b), image classification (Cen et al., 2021; Patel & Wang, 2022; Ma et al., 2022), segmentation (He et al., 2017, 2021), self-driving (Hemmati et al., 2022), and medical image analysis (Li et al., 2021a; Gosavi et al., 2022). Generally speaking, the detection models could be categorized as two-stage detectors and one-stage detectors. The two-stage detectors utilize Region Proposal Networks (RPN) (Ren et al., 2015) to select the anchors with high probability containing some objects and refine them. Those refined anchors are employed for classification and regression in the second stage. While one-stage detection models directly classify and regress the anchors to output the final results. The two-stage detection models often have higher accuracy while lower speed for inference compared to the one-stage detectors. RetinaNet (Lin et al., 2017b) discovers that the class imbalance of positives and negatives is the reason for the accuracy gap between two-stage models and one-stage models and proposes focal loss (Lin et al., 2017b) to solve this problem and close the accuracy gap between two types of models while still maintaining fast inference time of one-stage detectors. Recently, one-stage detection paradigms have a dominating influence for their high accuracy and low latency. Similar to image segmentation, object detection requires fine-grained details to accurately localize the bounding box of the object. Thus, recent object detectors frequently exploit Feature Pyramid Networks (FPN) (Lin et al., 2017a) so that large feature maps with fine details could recognize the small objects, while small feature maps with large receptive fields could detect the large objects.

Label assignment is to divide the samples into positives and negatives, which is essential to the success of object detection models. For anchor-based models, the core element for label assignment is the threshold for the division of positive and negative samples. After we calculate the Intersection over Union (IoU) between anchors and ground truth bounding boxes, the positive

samples are those anchors whose IoUs are larger than the threshold, while others are negatives or ignored. Sometimes the detection models design two thresholds, one for positive and the other for negative. The anchors whose IoUs are larger than the positive threshold are positive samples, while the anchors whose IoUs are smaller than the negative threshold are negative samples. Those anchors whose IoUs are between the positive threshold and negative threshold are ignored during the training process. Those early detection models (Ren et al., 2015; Liu et al., 2016) utilize fixed thresholds to divide the positives and negatives. However, the algorithms with the fixed threshold for dividing the positives and negatives ignore the differences between objects for their various shapes and sizes. For instance, some large or square-shaped objects frequently have more high-quality anchors corresponding to them, while some small and slender objects that have an extreme ratio of width and height often have low-quality anchors matched to them. Simply using a fixed IoU threshold for label assignment might deteriorate the performance of detecting objects without regular shapes. Although we could design more anchors with various shapes and sizes to alleviate the problem, the overhead and computational cost are also increased, which is not desired for the requirements that prefer low latency and less computational cost.

Recently, more and more adaptive label assignment strategies (e.g., ATSS (Zhang et al., 2020a)) have been proposed to adaptively calculate the threshold. These algorithms adaptively select positive samples and negative samples based on the IoU distribution between anchors and ground truth bounding boxes so that the ground truth bounding boxes that have more high-quality anchors corresponding to them will have a higher IoU threshold and those which have the most low-quality anchors corresponding to them will have a low IoU threshold, as illustrated in Figure 4.1. Nevertheless, adaptive assignment methods do not assign positives and negatives based on the predictions which are more accurate to represent the training status. Due to the discrepancy between the classification and localization, classification scores cannot precisely correspond to the localization quality, while NMS (non-maximum suppression) supposes that classification scores represent the localization quality and filters duplicates so that only the samples with high classification scores will be kept. However, if classification scores cannot accurately represent the

localization quality, the high-quality bounding boxes might be eliminated and some low-quality bounding boxes might be kept. However, fixed anchors cannot guarantee the quality of the predicted bounding boxes.

Therefore, introducing predictions to instruct label assignment is an effective approach to include the anchors, which could generate high-quality predictions as the positives. For some models using dynamic label assignment strategies, predictions are utilized to guide the label assignment during training since they represent the true training status. Nonetheless, the predictions in the early stage of training are inaccurate for both classification and bounding box regression; thus, they are inappropriate to instruct the label assignment if we directly apply them to dividing positive samples and negative samples. Adding distances to the ground truth centers as prior is proposed in the algorithm which utilizes predictions to weight the positive samples. Those prediction-based label assignment strategies exploit the distances or bounding boxes as the prior so that the positives are limited to the ground truth bounding boxes or those positive samples that are closer to the centers of the ground truth bounding boxes have more weights during the training process (Zhu et al., 2020a; Ge et al., 2021). The predictions (classification scores or predicted boxes) and the distances are two different "domains", so they could not be naturally combined. Thus, Autoassign (Zhu et al., 2020a) designed a center weighting module to solve this problem; however, the module may be sub-optimal due to the assumption that the samples closer to the centers of the ground truth would have more weights. MAL (Ke et al., 2020) employs "All-to-Top-1" strategy which includes enough anchors to learn the detector in the early training stage, and the number of anchors gradually decreases with the training process going on and finally only one optimal anchor is utilized. However, "All-to-Top-1" (Ke et al., 2020) reduces the number of anchors in the bag based on iterations instead of predictions. Thus, the training may not be optimal since the number of anchors in the bag is not controlled by predictions and might not satisfy the training status.

In this chapter, we propose a simple and effective method that directly combines the predicted IoUs between the predicted bounding boxes and the ground truth bounding boxes, and the anchor IoUs between the anchors and the ground truth bounding boxes. According to the adaptive mod-

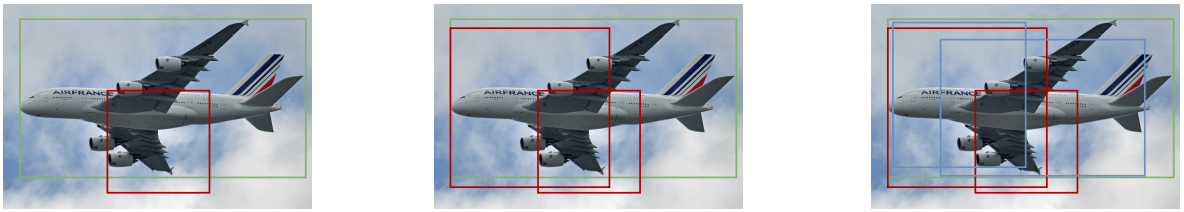

Figure 4.1: Illustration of label assignment strategies of RetinaNet, ATSS, and our method. The green box is the ground truth bounding box and the red and blue boxes are anchors and predicted boxes by corresponding anchors, respectively. Left: RetinaNet employs a fixed IoU threshold of 0.5 to select the positive anchors. If the IoU between the anchor (red box) and the ground truth (green box) is larger than 0.5, the anchor is denoted as positive to the ground truth. Middle: ATSS calculates the adaptive IoU thresholds based on the distribution of the anchors for each ground truth. If the ground truth has the most high-quality anchors (IoU to the ground truth is high), the IoU threshold for the ground truth is also high. Otherwise, the IoU threshold is low. Right: Our method also considers the predicted box (blue box) for each anchor (red box). Even though some anchors do not satisfy the IoU threshold, they might meet the threshold considering the predicted boxes, and the predicted boxes could assist the model in selecting the positive anchors more accurately.

els, the adaptive threshold is attained according to the statistical properties of the IoUs between the candidate anchors and the ground truth bounding boxes. Our method computes the distribution of the predicted IoUs and the anchor IoUs separately and then attains the combined parameters by simply adding them, respectively. Finally, the combined threshold is computed by the combined distribution parameters. Since the predictions are involved in the label assignment, soft targets (predicted IoUs between the predicted bounding boxes and the ground truth boxes) are more appropriate than the hard target (label 1) for positives in classification loss. QFL (Li et al., 2020b) and VFL (Zhang et al., 2021) are commonly utilized classification loss with soft targets. Both of them could further boost the performance of our proposed method. In addition, we replace the Centerness branch with the IoU branch for better accuracy.

In our work, we demonstrate the importance of using predictions to include samples with higher-quality predicted boxes as positives for label assignment and propose a simple and effective approach to introduce the predictions to the definition of positive and negatives. Our method could naturally introduce the predictions to label assignment algorithms and the anchors could perform

as prior for predictions. The experiments on COCO dataset (Lin et al., 2014) illustrate the effectiveness of our method without extra cost. Designing object detection models dynamically based on training status is a trend recently. While directly using the predictions is unreasonable due to the inaccurate predictions in the early stage of training. Thus, how to create prior to restrict the predictions is important but under-explored. Our idea of exploiting anchor IoUs as prior to combine with predicted IoUs could be a good choice and might inspire more related works.

4.2 Label Assignment in Object Detection

The label assignment is the core factor for the performance of the detection models, and how to divide positive samples and negative samples would determine how the networks learn and converge. For RPN training in Faster R-CNN (Ren et al., 2015), the pre-defined anchors whose IoUs with any ground truth bounding box higher than 0.7 will be defined as positives, and those whose IoUs with all ground truth objects lower than 0.3 are negatives. The anchors with IoUs between 0.3 and 0.7 are ignored during training, and the networks do not learn from those anchors. For SSD (Liu et al., 2016), the threshold of IoU between the positive anchor boxes and the ground truth objects is 0.5. RetinaNet (Lin et al., 2017b) assigns an anchor box to a ground truth object if the IoU between them is higher than or equal to 0.5, and assigns an anchor box as a negative sample if the IoUs between the anchor and all ground truth boxes are lower than 0.4. Others are ignored during training. The aforementioned strategies are traditional label assignment approaches with fixed thresholds for dividing positives and negatives. Even though those detection models with fixed thresholds are still effective for label assignment, they ignore the differences between various object samples for their shapes, sizes, and number of corresponding positive anchors. For instance, with fixed thresholds for label assignment, the objects with square-like shapes or large sizes might have more high-quality anchors corresponding to them so that they have more positive anchors during training, while some objects with slender shapes or small sizes might have the majority of low-quality corresponding anchor boxes; thus, they correspond to fewer positive anchors during training. So during training, the networks will be biased toward the objects with balanced ratios

of width and height or large sizes, and the performance of some slender or small objects will be compromised.

Recently, researchers focus on designing adaptive thresholds and gradually discard the fixed thresholds for label assignment. ATSS (Zhang et al., 2020a) computes the adaptive thresholds by calculating the mean and standard deviation according to the distribution of the IoUs between the candidate anchors and the ground truth objects. PAA (Kim & Lee, 2020) attains the anchor scores by combining the classification scores and localization scores, and then the selected anchor candidates which are based on top anchor scores are fitted into Gaussian Mixture Model (GMM) and the GMM is optimized by the Expectation-Maximization algorithm probabilistically. The candidate anchor boxes are separated by the boundary schemes as positives and negatives.

Using predictions to guide the label assignment could be more accurate since the pre-defined anchors might not accurately reflect the actual training status. Nevertheless, predictions in the early training stage are inaccurate and unreasonable to instruct the label assignment. FreeAnchor (Zhang et al., 2019b) exploits maximum likelihood estimation (MLE) to model the training process so that each ground truth could have at least one corresponding anchor with both high classification score and localization score. To solve the inaccurate predictions in the early epoch of training, the mean-max function is proposed so that almost all candidate anchors could be used in training the network and the best anchor could be selected from candidates when the training is sufficient, which is similar to "All-to-Top-1" mechanism in MAL (Ke et al., 2020). MAL (Ke et al., 2020) employs predictions from classification and localization as the joint confidence for the evaluation of the anchors. To alleviate the sub-optimal anchor-selection problem, MAL perturbs the features of selected anchors based on the joint confidence to suppress the confidence of those anchor candidates so that other anchors could have a chance to be selected and participate in the training process. In addition, MAL proposes "All-to-Top-1" strategy for anchor selection that includes enough anchors to be involved in training and gradually decreases the number of anchors in the bag to 1 with the training process going on. However, linearly decreasing the number of anchors in the bag is irrelevant to the prediction status and perturbing the features of selected anchors

introduces randomness which is also not correlated to the current predictions. Thus, this strategy might not be optimal to train the network.

Autoassign (Zhu et al., 2020a) introduces Center Weighting as the prior to address the unreasonable predictions in the early training phase, which indicates that the samples closer to the centers of ground truth would have more weights. Nonetheless, the prediction confidence and the distance to the ground truth centers are two different "domains" and the combination might be sub-optimal even though Autoassign models the distances with Gaussian-shape weighting function (Zhu et al., 2020a). Our model naturally combines the predicted IoUs (IoUs between the predicted boxes and the ground truth boxes) and the anchor IoUs (IoUs between the anchor boxes and the ground truth boxes) as the combined IoUs. Since they are both IoUs to the ground truth boxes, they could be naturally combined together without any complex functions or weights. In the early training stage, the anchor IoUs dominate the training process due to the random initialization. As the training process proceeds, the predicted IoUs will gradually dominate the combined IoUs, and more samples with high-quality bounding boxes will be selected as positives.

4.3 Proposed Approach

The adaptive label assignment strategies frequently divide the positive and negative samples by calculating the statistical parameters (e.g., mean and standard deviation) based on the candidate anchors or anchor bags which are selected according to the Euclidean distances between the centers of the anchors to the centers of the ground truth bounding boxes. After the candidate anchors are selected based on their positions to the ground truth boxes, the adaptive thresholds are computed based on the distribution of their IoUs to the corresponding ground truth bounding boxes. In the chapter, ATSS (Zhang et al., 2020a) is utilized as an example to illustrate the adaptive label assignment method and our proposed approach.

4.3.1 Revisit ATSS

ATSS (Adaptive Training Sample Selection) (Zhang et al., 2020a) makes an empirical analysis of the anchor-free approaches and anchor-based approaches and concludes that how to divide the positive and negative samples is the significant difference between anchor-based models and anchor-free models. Thus, ATSS proposes an algorithm that calculates the adaptive thresholds for defining the positives and negatives based on the candidate anchors which are selected according to the Euclidean distances between the centers of the anchors to the centers of the ground truth bounding boxes. The ATSS algorithm is shortly summarized as below:

For each GT (ground truth) box,

1: Compute the Euclidean distances between the centers of all anchors and the center of the GT.

2: Select k anchors with the smallest distances for each feature pyramid level; if the number of feature levels is L, the number of total candidate anchors corresponding to the GT is kL.

3: Compute the IoUs between kL candidate anchors and the GT box. Then calculate the mean and standard deviation of those IoUs.

4: The adaptive threshold is mean+std. If one anchor has IoU with the GT larger than or equal to mean+std, it is candidate positive of the GT, else it is negative.

5: Only the candidate positives whose centers are inside the GT box would be the final positives of the GT, others are negatives.

ATSS computes the threshold adaptively according to the shapes and sizes of the GT bounding boxes. If the GT boxes are large or square-like, the threshold will be higher since there are more high-quality anchors corresponding to them. If the GT boxes have slender shapes or small sizes, the threshold will be lower due to most low-quality anchors corresponding to them. Nevertheless, ATSS only computes adaptive thresholds according to the relationship between anchors and GT boxes. It merely relies on the anchors and ignores the actually predicted bounding boxes. In other

words, the anchor with the highest IoU to the GT box cannot guarantee its predicted bounding box also has the highest IoU to the GT among all positive anchors. Thus, some samples with high-quality predicted bounding boxes might be defined as negative samples whose classification target is 0. Thus, the performance of high-quality bounding boxes is affected. We will demonstrate it in the experiments. Using predicted information may improve the accuracy of defining the positives and negatives since predictions can represent the real training status of each sample. However, directly using the predictions might not be appropriate since the predictions in the early training stage are unreasonable to instruct the positive and negative definitions. Thus, we propose a simple and effective approach to address this problem by combining the predicted IoUs to the GT and the pre-defined anchor IoUs to the GT for each training sample.

4.3.2 Dynamic ATSS

We propose Dynamic ATSS, which introduces the predictions to the anchors for label assignment. In the early training phase, the predictions are inaccurate due to random initialization. Thus, the anchors would act as prior to instruct the label definition. The predictions gradually dominate the combined IoUs and would lead the label assignment with the training proceeding and the predictions being improved. The network structure is illustrated in Figure 4.2. The network structure is the same as ATSS, which has a CNN backbone (He et al., 2016), an FPN neck (Lin et al., 2017a) and a shared head which has two branches for classification and regression, respectively, while our proposed approach would extract the regression results and decode the regression offsets to the coordinates of bounding boxes, and finally calculate the IoUs between the decoded bounding boxes and the GTs. The predicted IoUs will be combined with anchor IoUs for selecting the positive samples.

$$CIoUs = PIoUs + AIoUs \tag{4.1}$$

$$mean(CIoUs) = mean(PIoUs) + mean(AIoUs) \tag{4.2}$$

$$std(CIoUs) = std(PIoUs) + std(AIoUs) \tag{4.3}$$

$$threshold(CIoUs) = mean(CIoUs) + std(CIoUs) \tag{4.4}$$

Equations (4.1)–(4.4) illustrate our proposed methods. $PIoUs$ is the predicted IoUs between the predicted bounding boxes and the ground truth boxes, and $AIoUs$ indicates the anchor IoUs between the pre-defined anchor boxes and the ground truth boxes. $CIoUs$ indicates the combined IoUs which is the summation of predicted IoUs and anchor IoUs. When calculating the mean and standard deviation for $CIoUs$, we compute them for $PIoUs$ and $AIoUs$ separately and sum them together, as illustrated in Equations (4.2) and (4.3). Finally, the thresholds of $CIoUs$ for defining the positives and negatives are computed by the summation of the mean and standard deviation of $CIoUs$. Since $PIoUs$ and $AIoUs$ are both IoUs to the ground truth bounding boxes, they can be naturally combined together by summation without designing any sophisticated formula (Zhu et al., 2020a) or reducing the number of positives during the training process (Ke et al., 2020). We also implement some experiments that down-weight the $AIoUs$ or up-weight the $PIoUs$ with the training proceeding. However, the simple addition of $PIoUs$ and $CIoUs$ could yield the best results, which is demonstrated in the experiments.

Why is utilizing predictions so important to guide the label assignment? The predictions are more accurate than the pre-defined anchors for defining the positives and negatives since we select the final results and implement the NMS algorithm based on the predicted results instead of the anchor boxes. We frequently design the detection models based on the assumption that the samples whose pre-defined boxes have high IoUs with the ground truth boxes are appropriate to be selected as positives or the samples whose centers are close to the centers of the ground truth objects are good candidates for positives. Once the positive samples are selected for each image, they would not be modified during the training process since the pre-defined anchor boxes or anchor points are fixed and they would not be changed according to the training status. Nevertheless, the samples with high-quality predictions might not frequently be those samples with high-quality anchor boxes or anchor points, although they have higher probabilities to generate high-quality predictions.

If we force the samples with high-quality anchor boxes or anchor points to be the positives through the entire training process, the network would focus on learning those samples even

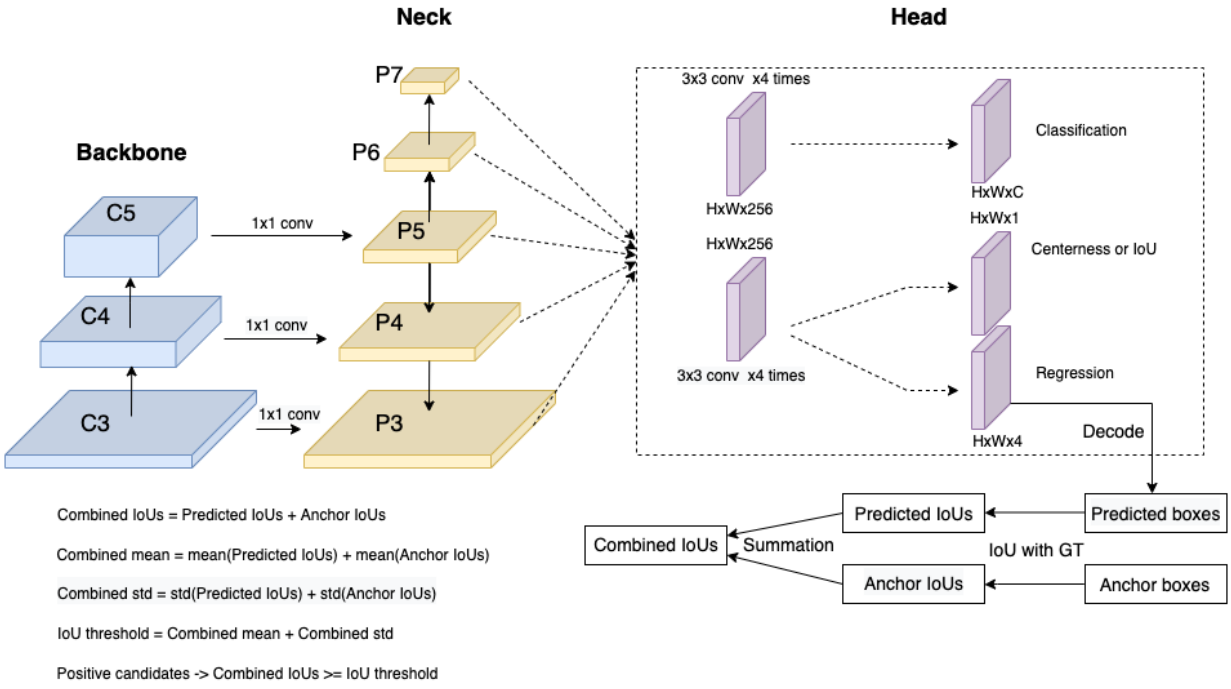

Figure 4.2: The network structure of our model. The structure of our model is the same as ATSS (Zhang et al., 2020a), which includes a CNN backbone (He et al., 2016), an FPN neck (Lin et al., 2017a), and a head that has two branches for classification and regression, respectively. Our approach would employ the predicted boxes, which are decoded from the regression branch. Then, the predicted IoUs and the anchor IoUs are attained by calculating the IoUs between the predicted boxes and the GTs, and the IoUs between the anchor boxes and the GTs, respectively. Finally, the Combined IoUs (CIoUs) are computed by summing the predicted IoUs and the anchor IoUs. The same calculation is implemented to attain the combined mean and combined std. The IoU threshold is computed by the summation of combined mean and combined std, and the positive candidates are defined as the samples whose combined IoUs are larger than or equal to the IoU threshold. The positive candidates are restricted inside the ground truth bounding boxes as the final positive samples.

though their predictions are not good enough and ignore those samples which could generate better-predicted results but may be assigned as negatives due to the relatively low-quality anchor boxes or anchor points. If the predictions are introduced to assist the definition of positive samples and negative samples, we could select more samples with high-quality predictions as positives and further improve those samples. From Table 4.1, simply adding predicted IoUs to anchor IoUs could yield better results and generate higher-quality predictions. The anchor IoUs are also necessary for our approach due to the random initialization of the network, and they can act as prior. In our method, the predictions and the prior are both IoUs to the ground truth bounding boxes; thus, they can be

naturally combined together by addition without any special design, which is shown in Figure 4.2.

4.3.3 Soft Targets for Classification Loss

With the emergence of focal loss (Lin et al., 2017b), most modern object detection models exploit focal loss for learning the class labels. Focal loss addresses the extreme imbalances between positive samples and negative samples during training and suppresses the majority of easy negative samples, which could dominate the training loss due to the extremely large number of those easy negatives. The focal loss function for positive samples ($y = 1$) and negative samples ($y = 0$) is shown in Equation (4.5).

$$\text{FL} = \begin{cases} -\alpha(1-p)^{\gamma} log(p), & y = 1 \\ -(1-\alpha)p^{\gamma} log(1-p), & y = 0 \end{cases} \tag{4.5}$$

Due to the introduction of predictions for label assignment, using soft target (predicted IoUs to the ground truth boxes) is more appropriate to rank the high predicted IoUs on top of other low predicted IoUs, which is utilized in GFL (Li et al., 2020b) and VFNet (Zhang et al., 2021). GFL is comprised of QFL and DFL for classification and regression, respectively. We employ QFL for classification in our model. QFL (Li et al., 2020b) for classification is illustrated in Equation (4.6).

$$\text{QFL} = \begin{cases} -|y-p|^{\beta}[ylog(p) + (1-y)log(1-p)], & y > 0 \\ -p^{\beta} log(1-p), & y = 0 \end{cases} \tag{4.6}$$

From the QFL equation, the cross-entropy loss is switched to the general form for positives ($y > 0$) since the soft targets are not equal to 1. In addition, the focal loss weights are also modified according to the soft targets.

Instead of down-weighting the losses when the classification predictions approach to the soft targets as used in QFL, VFNet (Zhang et al., 2021) exploits VFL (Zhang et al., 2021) that weights the positive losses with the soft targets to which those positives are assigned, which is demonstrated

in Equation (4.7). By changing the weights to the IoU targets for positives ($y > 0$), the losses of positive samples with higher IoU targets would also be higher so that the network could focus on learning those high-quality positives.

$$\text{VFL} = \begin{cases} -y[ylog(p) + (1-y)log(1-p)], & y > 0 \\ -\alpha p^{\gamma} log(1-p), & y = 0 \end{cases} \tag{4.7}$$

In the experiments, we could empirically illustrate that our proposed approach surpasses the same model using QFL or VFL in Table 4.1. In addition, by combining our proposed method with QFL or VFL, the performance of the detection model could be further improved.

4.4 Experiments

Implementation Details: The experiments are conducted on COCO dataset (Lin et al., 2014), where *train2017* is employed for training and *val2017* is for validation and ablation study. The training strategy exploits scheduler 1x (90 k iterations) for validation where the batch size is 16, and the initial learning rate is 0.01 and the learning rate is decreased by a factor of 10 at 60 k and 80 k iterations. The images are resized so that the maximum length of the longer side is 1333 and the length of the shorter side is 800. ResNet-50 (He et al., 2016) is utilized for all ablation study.

4.4.1 Ablation Study

In the experiments, ATSS algorithm (Zhang et al., 2020a) is selected as an example to illustrate the effectiveness of our proposed approach. We introduce the predicted IoUs to the ATSS algorithm to demonstrate the effectiveness of our approach.

4.4.1.1 The Effectiveness of Proposed Method

To prove the effectiveness of our proposed method, we did several experiments based on the ATSS algorithm. The experimental results are shown in Table 4.1.

Table 4.1: The effectiveness of the proposed method.

Model	AP	AP50	AP75	APs	APm	APl
ATSS	39.06	57.11	42.49	22.33	43.27	50.23
ATSS+CIoUs	**39.75**	57.43	43.08	23.03	43.83	52.27
ATSS+QFL	39.61	57.41	43.05	23.25	43.69	52.19
ATSS+VFL	39.65	57.38	43.39	23.45	43.53	52.19

From Table 4.1, ATSS combined with our proposed CIoUs (Combined IoUs) surpasses the same model with soft targets (QFL and VFL) for classification loss. Our simple modification can boost the original ATSS algorithm by around 0.7 AP on COCO *val2017* dataset, which demonstrates that using predictions could better guide the positive and negative definitions, and the anchor boxes are also necessary for instructing the label assignment, especially in the early stage of the training process. By simply combining them together, the model could yield great accuracy improvement. We simply introduce CIoUs into ATSS here, and the labeled target is still the hard target (1 for positives). In the following experiments, we will show that the performance could be further boosted with the soft target (QFL or VFL).

4.4.1.2 The Contribution of Each Element

In this section, we implement the experiments by removing some of the components. Through this ablation study, we can easily recognize the contribution of each element. The experimental results are demonstrated in Table 4.2.

In Table 4.2, similar to the aforementioned Equations (4.1)–(4.4), AIoUs represent the IoUs between the pre-defined anchor boxes and the ground truth bounding boxes. If only AIoUs are selected, the original ATSS is implemented. PIoUs indicate the IoUs between the predicted bounding boxes and the ground truth boxes. If both AIoUs and PIoUs are selected, our proposed Combined IoUs are implemented by summing the computed AIoUs and PIoUs. We can obviously notice that only employing PIoUs for label assignment significantly compromises the performance of the model from 39.06 AP to 29.39 AP, while simply adding the PIoUs to AIoUs for defining the pos-

itive samples and negative samples could yield around 0.7 AP improvement and the improvement happens in all metrics (AP, AP50, AP75, APs, APm, APl), which verifies that using predictions could include more candidates with high-quality predicted bounding boxes as the positive samples and finally improve the overall performance of the detection models.

Table 4.2: The ablation study for each element.

AIoUs	PIoUs	QFL	VFL	Centerness Branch	IoU Branch	AP	AP50	AP75	APs	APm	APl
✓				✓		39.06	57.11	42.49	22.33	43.27	50.23
	✓			✓		29.39	46.77	31.13	21.57	28.38	37.08
✓	✓			✓		39.75	57.43	43.08	23.03	43.83	52.27
✓	✓	✓		✓		40.07	57.46	43.73	23.47	44.30	52.60
✓	✓		✓	✓		39.83	57.45	43.15	22.75	44.22	52.88
✓	✓	✓			✓	**40.30**	57.49	44.00	22.85	44.48	53.71
✓	✓		✓		✓	40.15	57.37	43.64	23.51	44.09	53.21

Figure 4.3 illustrates the regression loss of the original ATSS and ATSS with our proposed combined IoUs. From Figure 4.3, the regression loss does not have too much difference in the early training phase for both models. While with the training process going on, our approach has lower regression loss than the original model, which indicates our model could select positive samples with higher-quality bounding boxes since more accurate predicted bounding boxes would yield lower regression loss. Additionally, the average precision for large objects (APl) is greatly improved by about 2%.

From Table 4.2, our proposed approach (AIoUs+PIoUs) could be further improved by soft targets (QFL and VFL). The original ATSS implements Centerness as the additional branch to weight the positive samples so that the samples closer to the centers of the GTs could have relatively higher weights than those far from the centers of the GTs. After switching the Centerness to IoU (predicting IoU instead of centerness), the performance could be boosted.

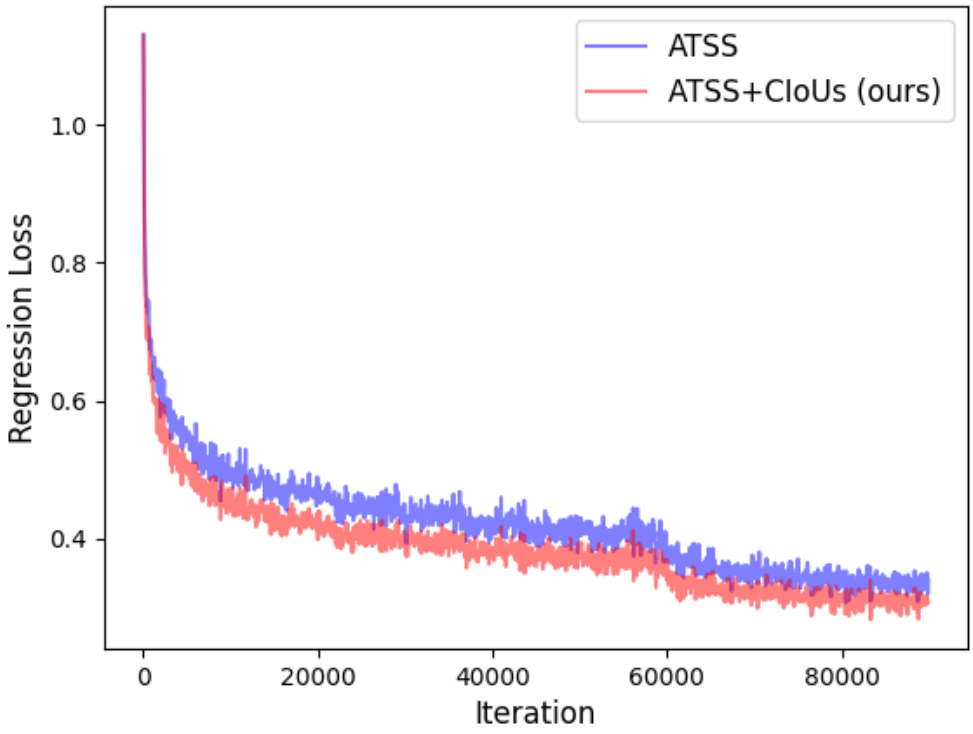

Figure 4.3: The regression loss of ATSS and ATSS+CIoUs.

4.4.1.3 Balancing Predicted IoUs and Anchor IoUs

We also investigate how to balance the predicted IoUs and the anchor IoUs when calculating the dynamic thresholds. The experiments are illustrated in Table 4.3.

From Table 4.3, the model with the ratio between PIoUs and AIoUs being 1:1 could yield the best accuracy. D_up indicates the dynamic weight gradually increases the weight with the training proceeding. D_down represents the dynamic weight that gradually decreases the weight with the training proceeding. In the experiment, we utilize $iteration/max\_iter$ as the D_up and $(1 - iteration/max\_iter)$ as the D_down. In the representation, $iteration$ is the current iteration and max_iter represents the total iterations the training would implement. The initial value of D_up is close to 0 and the final value of D_up would be close to 1 when the training is approaching the end. While the initial value of D_down is close to 1, the final value of D_down would be close to 0, which is the opposite of D_up.

Intuitively, the weight of the predicted IoUs should gradually increase since the predicted IoUs

Table 4.3: The ablation study on the weights for combination.

PIoUs	AIoUs	AP	AP50	AP75	APs	APm	APl
1	1	**39.75**	57.43	43.08	23.03	43.83	52.27
0.5	1	39.43	57.23	42.92	23.04	43.56	51.31
1.5	1	39.55	57.38	42.63	23.15	43.28	51.60
1	0.5	39.42	57.13	42.46	22.95	43.04	51.03
1	1.5	39.51	57.27	42.68	22.94	43.31	51.72
D_up	1	39.22	56.92	42.50	22.74	42.98	51.35
1	D_down	35.05	53.33	37.58	22.44	35.76	46.27
D_up	D_down	35.64	53.78	38.09	22.68	36.57	48.12

would be more and more accurate as the training proceeds. Thus, in our experiments, we apply D_up to the predicted IoUs and D_down to the anchor IoUs so that the predicted IoUs would gradually take over the label assignment and the anchor IoUs would gradually fade out of the label assignment. Nonetheless, introducing the dynamic weights to PIoUs and AIoUs does not improve the performance and applying D_down to AIoUs greatly worsens the accuracy of the detection model, which verifies that anchors still play a significant role in our method. Since PIoUs would gradually increase with the training proceeding due to the increasing localization accuracy of predicted boxes, dynamic weight has no positive effect to PIoUs. In our experiments, simple summation of PIoUs and AIoUs is applied and no weights are utilized for combining them. This ablation study also demonstrates that anchor IoUs (prior) and predicted IoUs could be naturally combined together without any sophisticated formula or weights, which is simpler than using a complicated formula to combine the center distance (prior) and the predictions (Zhu et al., 2020a) or "All-to-Top-1" (Ke et al., 2020) that gradually decreases the number of the anchors in the bag.

4.4.2 Application to the State-of-the-Art

The proposed approach combines the predicted IoUs with the anchor IoUs, so it could be directly applied to some state-of-the-art models that employ the ATSS algorithm for label assignment. GFLV2 (Li et al., 2021b) and VFNet (Zhang et al., 2021) are both state-of-the-art detection

models that employ ATSS algorithm for defining positives, negatives, and soft targets (IoUs) for the classification loss. Instead of directly predicting the offsets of the bounding boxes, GFLV2 (Li et al., 2021b) designs a distribution-guided quality predictor to estimate the localization quality via the bounding box distributions, and VFL (Zhang et al., 2021) makes a further bounding box refinement by employing a star-shaped box feature representation. We applied our method to GFLV2 (Li et al., 2021b) and VFL (Zhang et al., 2021), and the experimental results are shown in Table 4.4.

From Table 4.4, we can see that after applying our proposed method to the GFLV2 and VFNet, the two state-of-the-art models are further improved on COCO *val2017*. The improvement comes from the dynamic label assignment strategy (ATSS with our method), which selects samples with high-quality predicted boxes as the positives. Thus, the network could focus on those positives and further refine them. Figures 4.4 and 4.5 illustrate and compare the bounding box losses of GFLV2 to GFLV2+CIoUs, and VFNet to VFNet+CIoUs, respectively. Similar to the application of our method to ATSS, the models with our method yield lower bounding box losses with the training proceeding, which demonstrates that our proposed method could help the detection models to select higher-quality predicted bounding boxes that have lower bounding box losses.

Table 4.4: The application of the proposed approach to state-of-the-art models.

Model	**AP**	**AP50**	**AP75**	**APs**	**APm**	**APl**
GFLV2	40.6	58.1	44.4	22.9	44.2	52.6
GFLV2+CIoUs	**41.1**	58.6	44.8	23.7	44.3	53.9
VFL	41.3	59.2	44.8	24.5	44.9	54.2
VFL+CIoUs	**41.6**	59.5	44.9	24.3	45.1	54.6

4.4.3 Comparison to the State-of-the-Art

We apply our dynamic label assignment strategy (CIoUs+QFL+IOU branch) to the COCO *test-dev* (Lin et al., 2014) and compare them with the original ATSS. Table 4.5 illustrates the experimental results of the state-of-the-art models and our methods, where "MStrain" indicates multi-scale

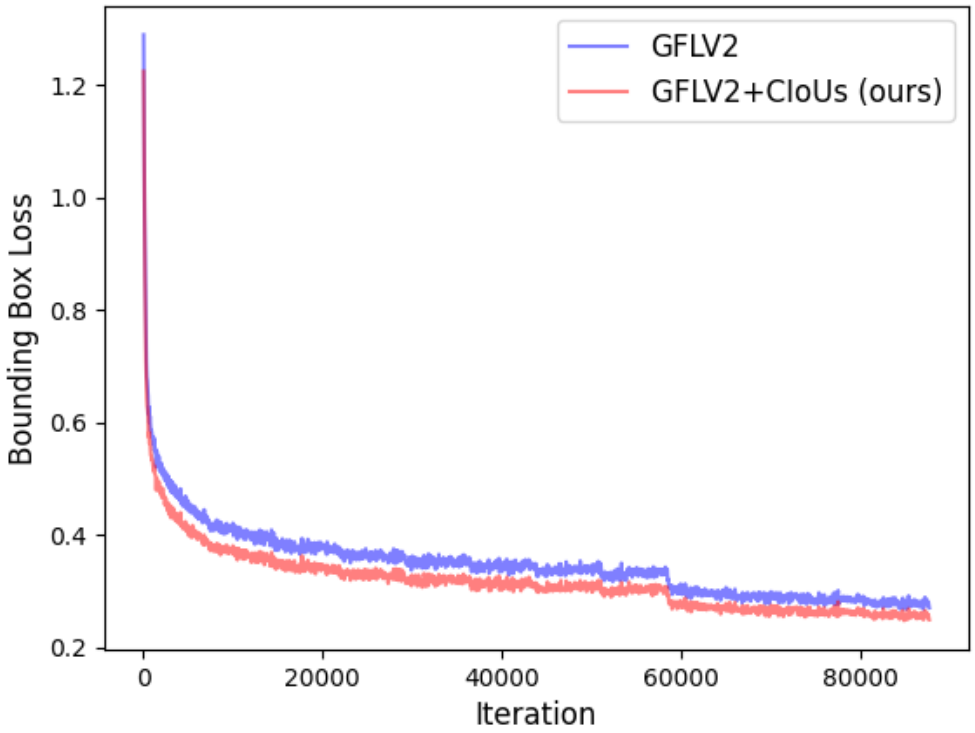

Figure 4.4: The bounding box loss of GFLV2 and GFLV2+CIoUs.

training strategy and "*" denotes our re-implementation. Scheduler 1x indicates 90 k iterations (12 epochs), and scheduler 2x represents 180 k iterations (24 epochs) on COCO *test-dev* benchmark.

Table 4.5 illustrates the experimental results of some state-of-the-art models and our methods. The NVIDIA Tesla P100 GPU is employed to test the inference time. From Table 4.5, we can see that the proposed method does not introduce extra cost and the real processing time for each image per GPU is the same as the original model (e.g., 80 ms indicates processing one image per GPU needs 80 milliseconds). Our Dynamic ATSS model could boost the original ATSS by around 1% in overall performance and 2% for large objects (APl). We also notice that our results are similar to state-of-the-art GFL (Li et al., 2020b), which is also based on ATSS for label assignment. While due to the introduction of distribution focal loss (DFL), GFL (Li et al., 2020b) predicts the bounding boxes multiple times for each sample, which would slightly increase the number of parameters. Our model is much simpler without introducing extra computational cost and parameters, and directly predicts the 4 offsets of the bounding boxes, just as the original ATSS.

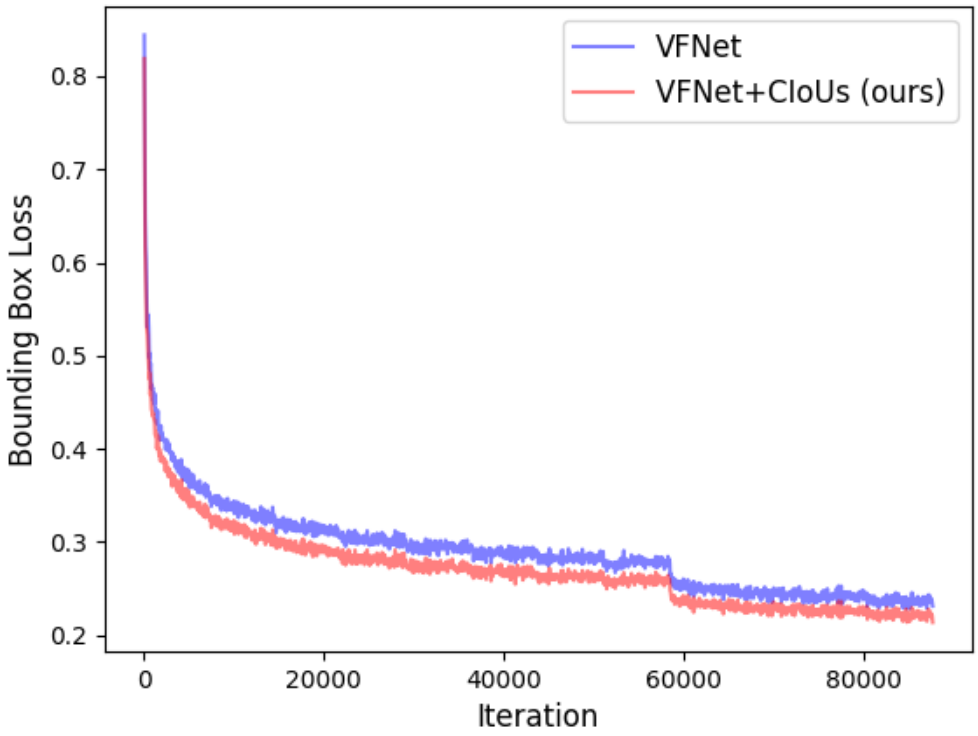

Figure 4.5: The bounding box loss of VFNet and VFNet+CIoUs.

4.5 Conclusions

In this chapter, we have illustrated the advantages of using predictions for defining positive samples and negative samples for object detection models and proposed a simple and effective method to improve the performance of adaptive label assignment algorithms. By combining the predicted IoUs and anchor IoUs, the label assignment approaches could divide the positive and negative samples dynamically according to the predictions. The predicted IoUs could be combined with the anchor IoUs (prior) by summation, so the dynamic ATSS could implement label assignment based on training status and select the samples with higher-quality predicted bounding boxes as the positives. When applying the proposed approach to state-of-the-art detection models with the adaptive algorithm for label assignment, the performance of the models could be further improved.

The proposed method can select samples with high-quality predicted bounding boxes as the positive samples even though they have relatively low-quality anchors (with low IoUs between the anchors and the ground truth bounding boxes). While most detection models divide the positive

Table 4.5: Evaluation results on COCO test-dev.

Method	Backbone	Scheduler	Time	MStrain	AP	AP50	AP75	APs	APm	APl
FreeAnchor (Zhang et al., 2019b)	ResNet-101	2x	-	✓	43.1	62.2	46.4	24.5	46.1	54.8
FreeAnchor (Zhang et al., 2019b)	ResNeXt-101-32x8d	2x	-	✓	44.9	64.3	48.5	26.8	48.3	55.9
TridentNet (Li et al., 2019b)	ResNet-101	2x	-	✓	42.7	63.6	46.5	23.9	46.6	56.6
FCOS (Tian et al., 2019)	ResNet-101	2x	-	✓	41.5	60.7	45.0	24.4	44.8	51.6
FCOS (Tian et al., 2019)	ResNeXt-101-64x4d	2x	-	✓	44.7	64.1	48.4	27.6	47.5	55.6
SAPD (Zhu et al., 2020b)	ResNet-101	2x	-	✓	43.5	63.6	46.5	24.9	46.8	54.6
SAPD (Zhu et al., 2020b)	ResNet-101-DCN	2x	-	✓	46.0	65.9	49.6	26.3	49.2	59.6
RepPoints (Yang et al., 2019)	ResNet-101	2x	-		41.0	62.9	44.3	23.6	44.1	51.7
RepPoints (Yang et al., 2019)	ResNet-101-DCN	2x	-	✓	45.0	66.1	49.0	26.6	48.6	57.5
GFL (Li et al., 2020b)	ResNet-101	2x	-	✓	45.0	63.7	48.9	27.2	48.8	54.5
GFL (Li et al., 2020b)	ResNet-101-DCN	2x	-	✓	47.3	66.3	51.4	28.0	51.1	59.2
ATSS * (Zhang et al., 2020a)	ResNet-50	1x	80 ms		39.2	57.5	42.6	22.3	41.9	49.0
ATSS * (Zhang et al., 2020a)	ResNet-50-DCN	1x	96 ms		43.0	61.2	46.8	24.5	45.9	55.3
ATSS (Zhang et al., 2020a)	ResNet-101	2x	105 ms	✓	43.6	62.1	47.4	26.1	47.0	53.6
ATSS (Zhang et al., 2020a)	ResNet-101-DCN	2x	131 ms	✓	46.3	64.7	50.4	27.7	49.8	58.4
ATSS (Zhang et al., 2020a)	ResNeXt-64x4d-101	2x	191 ms	✓	45.6	64.6	49.7	28.5	48.9	55.6
ATSS (Zhang et al., 2020a)	ResNeXt-32x8d-101-DCN	2x	225 ms	✓	47.7	66.6	52.1	29.3	50.8	59.7
ATSS (Zhang et al., 2020a)	ResNeXt-64x4d-101-DCN	2x	236 ms	✓	47.7	66.5	51.9	29.7	50.8	59.4
Dynamic ATSS	ResNet-50	1x	80 ms		40.3	57.9	44.1	22.5	43.6	51.2
Dynamic ATSS	ResNet-50-DCN	1x	96 ms		44.4	61.9	48.6	25.1	47.8	58.1
Dynamic ATSS	ResNet-101	2x	105 ms	✓	44.7	62.5	48.9	26.7	48.3	55.7
Dynamic ATSS	ResNet-101-DCN	2x	131 ms	✓	47.3	65.0	51.7	28.3	50.8	60.4
Dynamic ATSS	ResNeXt-64x4d-101	2x	191 ms	✓	46.5	64.7	50.8	29.1	49.7	57.6
Dynamic ATSS	ResNeXt-32x8d-101-DCN	2x	225 ms	✓	48.6	66.7	53.0	29.9	51.5	61.8
Dynamic ATSS	ResNeXt-64x4d-101-DCN	2x	236 ms	✓	48.6	66.7	52.9	29.4	51.9	61.3

samples and negative samples based on fixed anchor boxes, those anchor boxes are always positive samples during the training process only when their IoUs to the ground truth boxes are relatively high, even though the predicted boxes are not that excellent. Our method introduces the training status (the predicted box for each sample) into the division of positives and negatives, those candidates that easily generate high quality predicted boxes are easier to be categorized as the positives, which would assist the network to focus on those high quality samples and generate more high quality bounding boxes. In addition, the proposed method does not introduce extra computational

costs and the inference time remains the same as the original methods, while our model increases the accuracy of the detection.

Since our method forces the network to pay much attention to the samples with high-quality predicted boxes, the improvement for AP50 and small objects is less than that for AP75 and large objects. We are working to improve the algorithm to further boost its performance for AP50 and small objects. Prediction-based label assignment algorithms for object detection are significant to select high-quality positive samples according to training status. We hope our simple and effective approach could inspire more work on designing dynamic object detectors based on training status.

Chapter 5

Depth-Wise Convolutions in Vision Transformers for Efficient Training on Small Datasets

Abstract

The Vision Transformer (ViT) leverages the Transformer's encoder to capture global information by dividing images into patches and achieves superior performance across various computer vision tasks. However, the self-attention mechanism of ViT captures the global context from the outset, overlooking the inherent relationships between neighboring pixels in images or videos. Transformers mainly focus on global information while ignoring the fine-grained local details. Consequently, ViT lacks inductive bias during image or video dataset training. In contrast, convolutional neural networks (CNNs), with their reliance on local filters, possess an inherent inductive bias, making them more efficient and quicker to converge than ViT with less data. In this chapter, we present a lightweight Depth-Wise Convolution module as a shortcut in ViT models, bypassing entire Transformer blocks to ensure the models capture both local and global information with minimal overhead. Additionally, we introduce two architecture variants, allowing the Depth-Wise Convolution modules to be applied to multiple Transformer blocks for parameter savings, and incorporating independent parallel Depth-Wise Convolution modules with different kernels to enhance the acquisition of local information. The proposed approach significantly boosts the performance of ViT models on image classification, object detection, and instance segmentation by a large margin, especially on small datasets, as evaluated on CIFAR-10, CIFAR-100, Tiny-ImageNet, and ImageNet for image classification, and COCO for object detection and instance segmen-

tation.

5.1 Introduction

Transformer models have demonstrated exceptional performance in Natural Language Processing (NLP) tasks by capturing long-range relationships through attention mechanisms (Vaswani et al., 2017). However, the direct application of Transformer models to vision tasks is less intuitive, as images are inherently interconnected, and pixels exhibit close relationships. Vision Transformer (ViT) (Dosovitskiy et al., 2020) addresses this challenge by dividing the image into fixed-size patches, linearly embedding each patch as a token. To capture 2D relationships among image tokens, positional embedding is introduced, compensating for the loss of 2D coordinate relationships in embedded image patches. ViT includes a learnable class token to interact with image patch tokens for image classification.

Despite its success, ViT often requires substantial data and longer training times due to the attention mechanism's computational demands. The attention mechanism calculates the dot product of embeddings for each token pair, necessitating more time to learn the inductive bias that neighboring pixels share stronger relationships. Global attention in ViTs treats all tokens equally, neglecting the fact that neighboring image patches have higher relationships. In contrast, Convolutional Neural Networks (CNNs) naturally possess inductive bias due to local filters. However, CNNs may have a lower upper bound than ViTs because of their limited global view. In essence, ViTs outperform CNNs when datasets are large enough and training times are sufficiently long, showcasing their superior performance under such conditions.

Image contents are inherently cohesive as a whole, and forcefully splitting them into patches can hinder the recognition process. Moreover, treating all patches equally in models like Vision Transformers (ViTs) sacrifices the inductive bias present in the images, requiring a more extensive training effort to converge. While some approaches involve overlapping patches, this introduces additional computational costs without fundamentally addressing the issue. In contrast, CNN models, by their nature, excel at filtering local pixels in a contiguous manner, which is crucial for image

recognition, particularly when dealing with relatively small objects. However, the lack of global views may restrict the performance of convolutional models, especially in scenarios with abundant training data. The key question becomes: How can we efficiently integrate these two approaches, leveraging convolutions to support Transformer models, ensuring rapid convergence, and achieving superior performance?

In this chapter, we introduce a straightforward yet effective method to seamlessly integrate convolutional and Transformer blocks, enabling the simultaneous learning of global and local information efficiently. Our approach leverages Depth-Wise Convolutions (Howard et al., 2017) to capture local information, while Transformer blocks are employed to capture global information. The Depth-Wise Convolutions serve as a shortcut, bypassing the entire Transformer block (attention+FFN). The final combination is achieved through summation, providing a unified representation of both Depth-Wise Convolutions and Transformer blocks. The Depth-Wise Convolutions are applied for each Transformer block, creating two paths after each block for the network to choose from. This design ensures a flexible and dynamic integration of the local and global features. Our method achieves a superior performance improvement with only a marginal increase in parameters and computations, particularly benefiting small datasets. Our approach enables small-size Transformer models to outperform some larger counterparts, showcasing their effectiveness and efficiency.

In summary, our contributions are outlined below.

- We propose an efficient and effective approach to combining Depth-Wise Convolutions and Transformer blocks, allowing simultaneous capture of local and global information with minimal additional parameters and computational load. The proposed lightweight Depth-Wise module bypasses entire Transformer blocks to attain fine-grained details that might be missed otherwise. This module does not alter the internal structure of MHSA and FFN, making it a plug-and-play component that can be utilized by most Transformer models. Our approach demonstrates superior performance in image classification, object detection, and instance segmentation.

- We developed two types of architectural variants. The first variant aims to reduce parameters

and floating-point operations (FLOPs) by utilizing the Depth-Wise module to bypass multiple Transformer blocks. The second variant seeks to improve performance by incorporating multiple independent parallel Depth-Wise modules, each dedicated to enhancing local information.

- We demonstrate that certain modules are dispensable when our approach is implemented in the training of Transformer models on small datasets. Furthermore, by applying our approach without these modules, we can reduce both parameters and FLOPs, while significantly enhancing the performance.

5.2 Related Work

5.2.1 Vision Transformers

ViT (Dosovitskiy et al., 2020) introduces the Transformer models into vision recognition by splitting the images into fixed-size patches and then tokenizing each patch into the token so that the image patches can be utilized in the attention module of Transformer models. Many variations and improvements have been proposed (Patel et al., 2022) (Chen et al., 2023) (Zhu et al., 2023) and applied to various vision tasks, such as point cloud completion (Wang et al., 2023) and crowd counting (Sajid et al., 2021). DeiT (Touvron et al., 2021a) employs distillation tokens for attention learning from the teacher models to the student models. CaiT (Touvron et al., 2021b) introduces LayerScale to effectively train the ViT models with deeper layers so that the performance of deep ViT models could be further boosted. Hierarchical Vision Transformer architecture (Wang et al., 2021) (Heo et al., 2021) (Chen et al., 2021a) (Chen et al., 2022b) (Chen et al., 2024) are designed to better suit vision tasks by reducing the size of feature maps as the network progresses deeper, resembling the structure of CNN architectures.

To reduce the computational cost, some window-based Vision Transformer models have been proposed. Swin Transformer (Liu et al., 2021) restricts the self-attention of the tokens on small windows so that the inductive bias could be slightly introduced while significantly reducing the computational costs with the sacrifice of the global views. To mitigate the limitation of lacking

global views, it also incorporates a shifted window mechanism, expanding the self-attention calculation to new shifted windows. Thus, the views of tokens are expanded. Other works, such as (Fang et al., 2022) (Chu et al., 2021a) (Huang et al., 2021), attempt to increase the receptive fields with cross-window interactions so that the information between the windows could be exchanged and the tokens could exchange the information with other windows.

5.2.2 Vision Transformers and Convolutions

CvT (Wu et al., 2021) designs a hierarchical Transformer architecture with a convolutional token embedding and a convolutional Transformer block utilizing a convolution projection to project the feature maps into query, key, and value. BoTNet (Srinivas et al., 2021) replaces the final three bottleneck blocks of the ResNet model with BoT blocks that contain MHSA layers so that the self-attention layer can aggregate the information attained by the convolutional layers. LocalViT (Li et al., 2021c) introduces the Depth-Wise Convolution into the Feed-Forward Networks in the Transformer block to add locality into the Transformer models. CMT (Guo et al., 2022) proposes a hybrid Transformer model to take advantage of Transformers and CNNs for global views and local features, respectively. MobileFormer (Chen et al., 2022c) designs efficient networks to integrate MobileNet (Howard et al., 2017) and Transformer blocks with a two-way bridge in between so that both local features and global interactions can be effectively communicated and fused.

DHVT (Lu et al., 2022) integrates the convolutions into MLP and patch embeddings to introduce the inductive bias into the Transformer model, and introduces a dynamic feature aggregation module in MLP and a "head token" in MHSA for diverse channel representation so that the gap between the Transformer models and CNN models could be eliminated. ViTAE (Xu et al., 2021) and its extension model ViTAEv2 (Zhang et al., 2023) utilize multiple dilated convolutions to downsample the feature maps and aid the MHSA module to attain the locality simultaneously. Mixformer (Chen et al., 2022a) parallelizes window-based self-attention and Depth-Wise Convolution to extend the receptive fields and designs bi-directional interactions to exchange information of channel and spatial dimensions between them. DMFormer (Wei et al., 2023) proposes a Dynamic

Multi-level Attention mechanism which is comprised of Depth-Wise Convolutions with multiple kernel sizes for various patterns and a gating mechanism for adaptability. ScopeViT (Nie et al., 2024) involves Depth-Wise Convolutions into Transformer architecture for scale-aware efficient training. DctViT (Su et al., 2024) proposes a hybrid structure with convolutions and Transformers for higher accuracy on multiple vision tasks. The hybrid structures are also applied to wetland classification (Jamali et al., 2023), salient object detection (Wang et al., 2022) (Liu et al., 2023b), referring image segmentation (Liu et al., 2023a), etc.

The computational structures of some previous models focus on integrating convolutional networks into the Multi-Head Self-Attention (MHSA) or Feed-Forward Network (FFN), making convolutional networks essential components of Transformer architectures. Additionally, the convolutional components in some of these studies are not necessarily lightweight. In contrast, our approach aims to efficiently combine Transformer blocks with convolutions while minimizing computational overhead. It is designed as a versatile and straightforward module that can be easily integrated into various Vision Transformer models.

5.3 Methodology

5.3.1 Vision Transformers

ViT (Dosovitskiy et al., 2020) introduces Transformers (Vaswani et al., 2017) into vision tasks by splitting the images into patches which are tokenized into tokens ($\mathbf{x}$). To preserve the positional relations of the image patches, learnable positional embeddings are added to each token to learn the 2D relations of the patches. The tokens and positional embeddings are illustrated in Eq. (5.1). $\mathbf{x_c}$ demonstrates the class token and $\mathbf{x_p}$ indicates the positional embeddings.

$$\mathbf{x^0} = (\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_l}; \mathbf{x_c}) + \mathbf{x_p} \tag{5.1}$$

$$\mathbf{x'^n} = \mathbf{x^n} + \mathrm{MHSA}(\mathrm{LayerNorm}(\mathbf{x^n})) \tag{5.2}$$

$$\mathbf{x^{n+1}} = \mathbf{x'^{n}} + \mathrm{FF}(\mathrm{LayerNorm}(\mathbf{x'^{n}})) \tag{5.3}$$

Eqs. (5.2) and (5.3) illustrate the Multi-Head Self-Attention (MHSA) layer and the feed-forward (FF) layer. The residual connection and pre-LayerNorm (Ba et al., 2016) are harnessed in both layers. The attention layer and feed-forward layer are formed as Transformer blocks and Transformer models are comprised of cascaded Transformer blocks. The class token is employed to classify the image and output the result.

ViT models leverage self-attention mechanisms to compute the similarity between each pair of patch tokens and then assign different weights to different tokens according to the similarities between the patch tokens. Nonetheless, ViT models often overlook the inductive bias inherent in images, where neighboring pixels or patches have more relations. This oversight can lead to slow convergence, requiring more training iterations to learn the inductive bias and demanding large datasets for optimal performance. In contrast, convolutions inherently possess an inductive bias due to local filters traversing the image, capturing local details. Recognizing the complementary nature of convolutions to Transformer models, particularly in scenarios with small datasets, we propose a lightweight approach using Depth-Wise Convolutions to enhance the convergence and performance of Vision Transformer models. This is particularly beneficial when training ViT models from scratch on limited datasets without additional assistance.

5.3.2 Our Approach

Convolutional kernels excel at capturing fine details in images, a capability lacking in ViT models. The challenge lies in determining how and where to incorporate these kernels. To maintain a lightweight design without significantly increasing parameters and computational demands, we select Depth-Wise Convolutions to filter the local details. We utilize the Depth-Wise Convolution as the shortcut to bypass the entire Transformer block. Since the patch tokens are flattened to 1D,

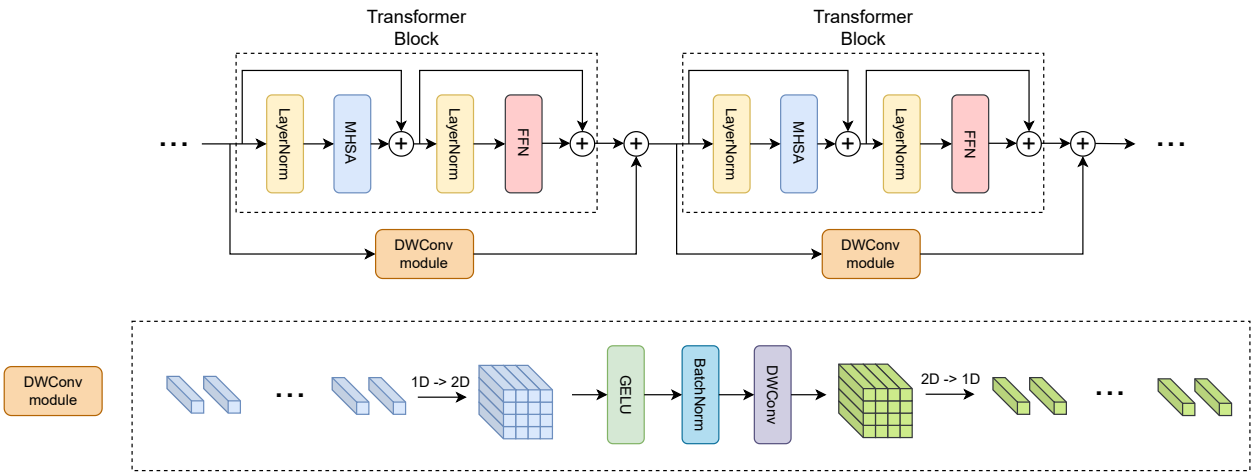

Figure 5.1: The architecture of our proposed method. The Depth-Wise Convolution module bypasses the entire Transformer block so that the local details can be attained and added to the output of the Transformer block. In the DWConv module, the 1D image patch tokens are first reshaped to 2D feature maps. If the class token exists, it would not be involved in the DWConv module and only image patch tokens are utilized to reconstruct the feature maps. Batch normalization and GELU activation are employed before the Depth-Wise Convolution. Finally, the feature maps would be reshaped to 1D tokens and added to the output of the Transformer block. The DWConv module is exploited in all Transformer blocks.

we have to reconstruct all patch tokens into 2D feature maps. The architecture of our proposed model is demonstrated in Fig. 5.1. The DWConv module is harnessed in all Transformer blocks as complementary components.

$\mathbf{x^n}$ from Eq. (5.2) is used to reshape the 1D tokens to 2D feature maps. The reshaped 2D feature maps are implemented GELU activation (Hendrycks & Gimpel, 2016) and batch normalization (Ioffe & Szegedy, 2015) before being fed into the Depth-Wise Convolution (DWConv). The kernel we utilize for Depth-Wise Convolution is 3×3. The 2D feature maps are reshaped to 1D patch tokens and finally the reshaped 1D patch tokens ($\mathbf{x_{1d}}^{\mathbf{n+1}}$) and the output of the Transformer block (Eq. (5.3)) are summed together. The summed result ($\mathbf{x_{ours}}^{\mathbf{n+1}}$) is utilized as the input to the next block. This process is illustrated below.

$$\mathbf{x_{2d}^n} = \text{Reshape}_{1d->2d}(\mathbf{x^n}) \tag{5.4}$$

$$\mathbf{x'_{2d}}^{\mathbf{n}} = \text{DWConv}(\text{BatchNorm}(\text{GELU}(\mathbf{x_{2d}}^{\mathbf{n}}))) \tag{5.5}$$

$$\mathbf{x_{1d}}^{\mathbf{n+1}} = \text{Reshape}_{2d->1d}(\mathbf{x'_{2d}}^{\mathbf{n}}) \tag{5.6}$$

$$\mathbf{x_{ours}}^{\mathbf{n+1}} = \mathbf{x}^{\mathbf{n+1}} + \mathbf{x_{1d}}^{\mathbf{n+1}} \tag{5.7}$$

The DWConv modules act as “supervisors" to supervise the Transformer blocks and they are complementary to each other. Each Transformer block is supervised by the DWConv modules to capture details that may be missed by the Transformer blocks. While the Transformer blocks play the main role in the architecture, the proposed lightweight DWConv modules are leveraged to retrieve local information, thereby enhancing the overall performance. Unlike some hybrid models that design complex hybrid architectures, our proposed approach demonstrates simplicity, effectiveness, and flexibility.

5.3.3 Architecture Variants

In addition to the base architecture, we have designed several variants based on the core structure, as illustrated in Fig. 5.2. In our base architecture, the Depth-Wise module bypasses each Transformer block. Additional variants are designed where the Depth-Wise module bypasses more Transformer blocks. These variants prove beneficial when working with Vision Transformers that have deeper layers, helping to reduce the number of parameters and computational costs.

Moreover, in Transformer architectures with multiple stages, the size of the feature maps is reduced and the dimension is increased in successive stages. To maintain the input and output sizes of the Depth-Wise module, we recommend limiting the bypass within each stage to prevent a Depth-Wise module from crossing stages when multiple Transformer blocks are bypassed. Alternatively, one Depth-Wise module can be used to bypass an entire stage, ensuring that each stage

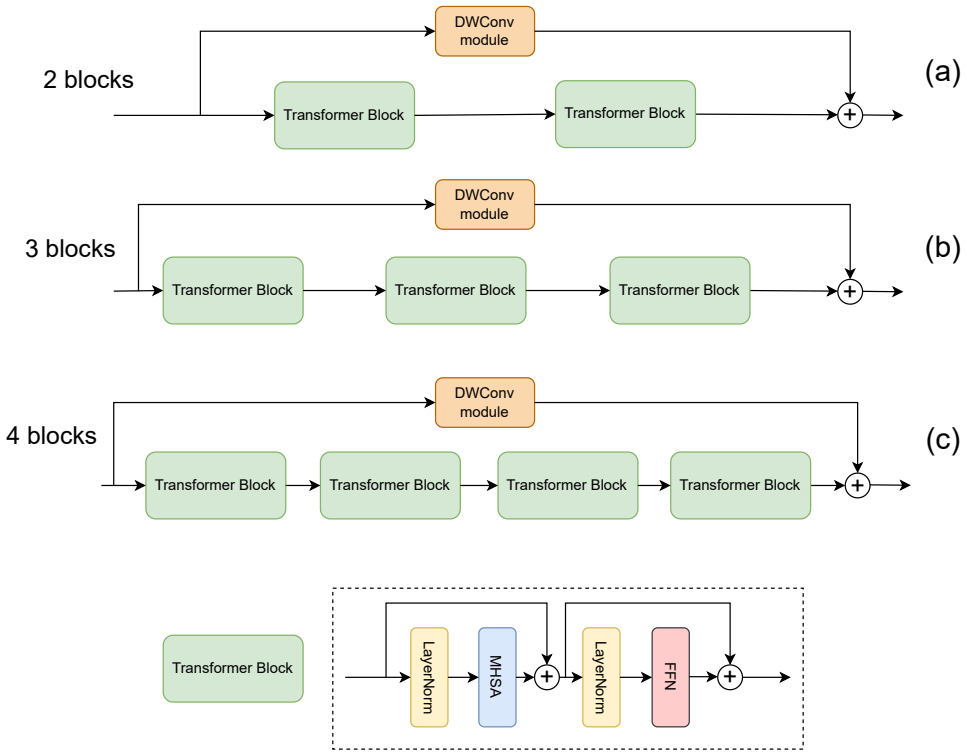

Figure 5.2: The architecture variants of our proposed approach involve bypassing multiple Transformer blocks. Structures (a), (b), and (c) represent the Depth-Wise module bypassing 2, 3, and 4 Transformer blocks, respectively. For Vision Transformer models with deeper layers, bypassing additional blocks may be a beneficial strategy to reduce both parameters and computational costs.

has only one corresponding Depth-Wise module for more efficient combinations.

Furthermore, the DWConv modules could operate in parallel with various kernel sizes to capture the local information independently, as illustrated in Fig. 5.3. In the experiments, we leverage parallel DWConv modules with different kernel sizes to demonstrate the performance of the variants. To further reduce the number of parameters and computational cost, multiple independent parallel DWConv modules could be combined with the aforementioned variants so that multiple Transformer blocks are contained by the DWConv modules. In Fig. 5.3, N Transformer blocks are encompassed in the DWConv modules and $N \geq 1$.

Not modifying the structures of MHSA and FFN makes our approach more flexible for use with most Transformer models, rather than being designed for specific ones. The proposed architecture variants illustrate the flexibility of our methods compared to some existing hybrid architectures that

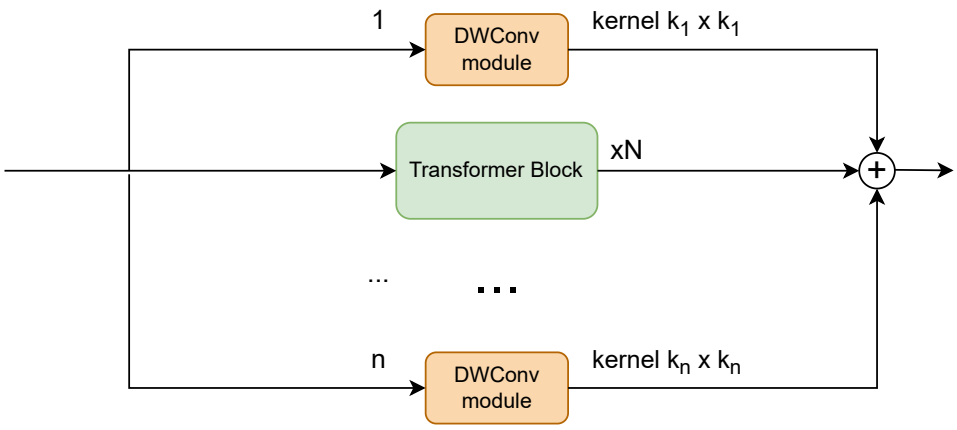

Figure 5.3: The architecture variants of our method involve multiple DWConv modules operating in parallel. These independent DWConv modules, each with different kernel sizes, run concurrently on Transformer blocks to capture local details simultaneously. This structure can be combined with previous variants shown in Fig. 5.2 to include N Transformer blocks in the DWConv modules.

combine convolutions and Transformers. The Transformers are greatly enhanced by the proposed DWConv module with minimal overhead. Additionally, the structure can be easily modified to further enhance the performance or save the parameters and computations with different variants.

5.3.4 Complexity Analysis

Our proposed approach is a lightweight module that is employed with each Transformer block. Unlike some models that insert convolutional layers inside the Transformer blocks, the proposed module is separable from the Transformer block, making it a plug-and-play module applicable to most existing Vision Transformer models. The increased parameters are dependent on the depths and dimensions of the Transformer models. Since our module is independent for each Transformer block without sharing parameters, deeper Transformer models could have more parameters introduced. However, the increased parameters are negligible compared to the Transformer backbone. For instance, the ViT-Tiny model used in our experiments has 12 blocks and a dimension of 192. With a depth-wise convolution of 3×3 kernel size, the increased parameters for the ViT-Tiny model are approximately $12 \times 192 \times (3 \times 3 + 1) = 23,040$ (0.023M) which is negligible compared

to the backbone with around 5.5 million parameters. Moreover, since the patch size is 16 and the images are resized to 224 and the size of the feature maps is 14×14, the increased calculations for ViT-Tiny model could be approximately calculated by $12 \times 192 \times (14 \times 14) \times (3 \times 3) = 4,064,256$ (0.004G), which is trivial compared to the total 1.26G FLOPs. In the aforementioned calculation, the number of parameters and calculations of BatchNorm are ignored since they are insignificant to the model.

In the experiments, sometimes our methods could even reduce the number of parameters and FLOPs considering some modules and positional embeddings could be removed for training the small dataset when our approach is applied to the Vision Transformer models. The increased number of parameters and FLOPs that are trivial to the models are highly dependent on the number of layers and dimensions of the models. Additionally, they also depend on which architecture variants are employed for the Vision Transformer models.

Some hybrid architectures merge convolutional networks into the Transformer architecture inefficiently, introducing significant parameters and computations as convolutional networks become essential components of Transformer structures. Additionally, these methods are often designed for specific Transformer architectures, making them impractical for other Transformer models. In contrast, our approach is designed to be easily incorporated into various Vision Transformer models. Complexity analysis shows that our approach introduces negligible overhead, with the majority of parameters and computations still coming from Transformer structures. However, the performance improvements are significant, especially on small datasets.

5.4 Experiments and Results

To verify the effectiveness and efficiency of our proposed approach, we select vanilla ViT (Dosovitskiy et al., 2020), CaiT (Touvron et al., 2021b), and Swin Transformer (Liu et al., 2021) for the experiments on three small datasets: CIFAR10 (Krizhevsky et al., 2009), CIFAR100 (Krizhevsky et al., 2009), and Tiny-ImageNet (Le & Yang, 2015). We also evaluated the model on a relatively large dataset: ImageNet-1K (Russakovsky et al., 2015). Additionally, COCO (Lin et al., 2014) is

utilized for the evaluation of object detection and instance segmentation.

5.4.1 Classification Performance on Small Datasets

CIFAR10 (Krizhevsky et al., 2009), CIFAR100 (Krizhevsky et al., 2009), and Tiny-ImageNet (Le & Yang, 2015) are exploited as small datasets for training and evaluating image classification tasks. The classification accuracy is defined as the ratio of correctly classified samples to the total number of samples. In our chapter, we use Top-1 accuracy for classification.

5.4.1.1 Experimental Settings

ViT. We select ViT-Tiny and ViT-Small to conduct the experiments for all datasets. The parameter settings of ViT models are followed by (Steiner et al., 2021). The dimensions of ViT-Tiny and ViT-Small are 192 and 384, respectively. The MLP ratio is 4 for both models which indicates the MLP dimensions are 768 and 1536 for tiny and small models, respectively. The numbers of heads for tiny and small models in Multi-Head Self-Attention are 3 and 6 respectively so the dimension for each head is 64 for tiny and small models. The depths are 12 for ViT-Tiny and ViT-Small.

CaiT. We choose CaiT-xxs12 and CaiT-xxs24 as the base models for the experiments. The dimensions of CaiT-xxs are 192 and the number of heads is 4. The MLP dimensions are 768 and the class depths are 2. The main depths for CaiT-xxs12 and CaiT-xxs24 are 12 and 24, respectively.

Swin Transformer. Swin-Tiny is selected for the experiments. For Swin-Tiny model, the drop path rate is 0.2 and the window size is 7; The depths and numbers of heads are (2, 2, 6, 2) and (3, 6, 12, 24) for each stage, respectively.

Experimental Parameters. All experiments are conducted using AdamW (Kingma & Ba, 2014) optimizer with 300 epochs and 20 epochs warmup. The weight decay is 0.05. The batch size for three small datasets is 128 with 4 NVIDIA P100 GPUs. The cosine decay learning rate scheduler is exploited. The base learning rate for Swin-Transformer on three small datasets is 2.5e-4, while the base learning rate for other experiments on small datasets is 5e-4. The images are resized to 224 and the patch size for both ViT and CaiT is 16.

Table 5.1: The experimental results of ViT-Tiny and ViT-Small on small dataset (PE = Positional Embedding)

Model	CIFAR-10			CIFAR-100			Tiny-ImageNet		
	Accuracy	Params	FLOPs	Accuracy	Params	FLOPs	Accuracy	Params	FLOPs
ViT-Tiny	94.01	5.5M	1.26G	73.68	5.5M	1.26G	59.00	5.6M	1.26G
ViT-Tiny w/o PE	87.83 (-6.18)	5.5M	1.26G	64.41 (-9.27)	5.5M	1.26G	53.15 (-5.85)	5.5M	1.26G
ViT-Tiny (**ours**)	**96.41** (+2.40)	5.5M	1.26G	**78.05** (+4.37)	5.6M	1.26G	**64.10** (+5.10)	5.6M	1.26G
ViT-Tiny w/o PE (**ours**)	**96.32** (+2.31)	5.5M	1.26G	**77.31** (+3.63)	5.5M	1.26G	**63.57** (+4.57)	5.5M	1.26G
ViT-Small	95.09	21.7M	4.61G	73.97	21.7M	4.61G	60.90	21.7M	4.61G
ViT-Small w/o PE	89.27 (-5.82)	21.6M	4.61G	65.68 (-8.29)	21.6M	4.61G	53.98 (-6.92)	21.7M	4.61G
ViT-Small (**ours**)	**97.02** (+1.93)	21.7M	4.62G	**80.01** (+6.04)	21.7M	4.62G	**66.86** (+5.96)	21.8M	4.62G
ViT-Small w/o PE (**ours**)	**96.96** (+1.87)	21.6M	4.62G	**80.14** (+6.17)	21.7M	4.62G	**66.59** (+5.69)	21.7M	4.62G

Table 5.2: The ablation study of ViT-Tiny (accuracy)

Method	CIFAR-10	CIFAR-100	Tiny-ImageNet
shortcut	87.50	65.10	52.15
kernel 3	96.32	77.31	63.57
kernel 5	96.26	**78.71**	63.67
kernel 7	96.26	78.69	63.95
kernel 3+5	**96.52**	78.63	64.00
kernel 3+5+7	96.39	78.00	**64.27**

Data Augmentation. Most regularization and augmentation settings follow (Liu et al., 2021), including color jitter, Auto-Augment (Cubuk et al., 2018), random erasing (Zhong et al., 2020), MixUp (Zhang et al., 2017), CutMix (Yun et al., 2019). All experiments are trained from scratch on each dataset without the assistance of an extra dataset.

5.4.1.2 Vision Transformer

The vanilla ViT model splits the image into small patches which are embedded as tokens for Transformer blocks. Since the tokens are 1-dimensional without the 2-dimensional positional information, the vanilla ViT model utilizes a positional embedding which would be added to all tokens to learn the 2-dimensional positional relationship between tokens. Since convolutions with

zero padding could encode the positional information (Chu et al., 2021b), we also applied our approach to the ViT model without positional embeddings. The experimental results are illustrated in Table 5.1.

From Table 5.1 we observe that removing the positional embeddings significantly deteriorates the performance of ViT-Tiny and ViT-Small, highlighting the importance of positional embeddings in Vision Transformers. When our method is applied to ViT models, there is a substantial improvement in performance, regardless of the presence of positional embeddings. For ViT-Tiny, the increased accuracy for CIFAR-10, CIFAR-100, and Tiny-ImageNet are around 2%, 4%, and 5%, respectively. For ViT-Small, the performance boost for CIFAR-10, CIFAR-100, and Tiny-ImageNet are nearly 2%, 6%, and 6%, respectively. Additionally, our method without positional embeddings even slightly reduces the number of parameters with much better accuracy. More importantly, the accuracy of ViT-Tiny with our proposed DWConv surpasses that of vanilla ViT-Small which has almost 4x the number of parameters and FLOPs by large margins, demonstrating the efficiency and effectiveness of our proposed method.

Moreover, extra experiments are implemented with different kernel sizes and parallel DWConv modules, as illustrated in Table 5.2. Directly applying a shortcut connection with positional embedding without any modules to bypass the Transformer blocks significantly reduces the accuracy. The possible reason for that might be the low-level input embeddings for the Transformers, which is different from the high-level input features attained from CNNs (Ma et al., 2021). Some large kernel sizes or parallel DWConv modules could boost the performance with a slightly higher number of parameters and FLOPs.

5.4.1.3 CaiT

CaiT introduces LayerScale to improve the performance of deeper layer transformer models by multiplying a learnable diagonal matrix (Touvron et al., 2021b) by each residual block. The class attention is introduced before the final classifier to convert patch embeddings into the final class embeddings. Talking heads attention (Shazeer et al., 2020) is utilized in the model for

Table 5.3: The experimental results of CaiT-xxs12 and CaiT-xxs24 on small dataset (LS = Layer-Scale, TH = Talking Head, PE = Positional Embedding)

Model	CIFAR-10			CIFAR-100			Tiny-ImageNet		
	Accuracy	Params	FLOPs	Accuracy	Params	FLOPs	Accuracy	Params	FLOPs
CaiT-xxs12	92.02	6.4M	1.30G	73.43	6.4M	1.30G	59.17	6.5M	1.30G
CaiT-xxs12 w/o-(LS, TH, PE)	87.45 (-4.57)	6.4M	1.28G	70.32 (-3.11)	6.4M	1.28G	60.62 (+1.45)	6.4M	1.28G
CaiT-xxs12 w/o-(LS, TH, PE) (**ours**)	**96.43** (+4.41)	6.4M	1.29G	**81.72** (+8.29)	6.4M	1.29G	**70.47** (+11.30)	6.4M	1.29G
CaiT-xxs24	93.89	11.8M	2.53G	74.84	11.8M	2.53G	60.97	11.8M	2.53G
CaiT-xxs24 w/o-(LS, TH, PE) (**ours**)	**97.28** (+3.39)	11.8M	2.51G	**82.83** (+7.99)	11.8M	2.51G	**70.64** (+9.67)	11.8M	2.51G

Table 5.4: The accuracy for different blocks bypassed by DWConv module with CaiT

Method	CIFAR-10	CIFAR-100	Tiny-ImageNet
xxs12 (1 block)	**96.43**	**81.72**	**70.47**
xxs12 (2 blocks)	96.16	80.67	68.76
xxs12 (3 blocks)	95.60	79.50	67.61
xxs12 (4 blocks)	94.80	78.61	67.49
xxs24 (1 block)	**97.28**	82.83	**70.64**
xxs24 (2 blocks)	97.00	**82.90**	70.07
xxs24 (3 blocks)	96.84	81.90	69.57
xxs24 (4 blocks)	96.51	80.16	68.61

further improvement of the performance. However, LayerScale (Touvron et al., 2021b) and talking heads attention (Shazeer et al., 2020) are not necessary when our proposed approach is applied to CaiT model on a small dataset. Moreover, talking heads attention is extremely time-consuming for small dataset training in our experiments. Thus, LayerScale and talking heads attention are removed when our method is applied to CaiT-xxs12 and CaiT-xxs24, which is demonstrated in Table 5.3, where “LS", “TH" and “PE" illustrate LayerScale, talking heads attention and positional embeddings. Similar to the vanilla ViT model, the positional embeddings are not necessary when our method is introduced to the small dataset training.

It is evident from Table 5.3 that removing LayerScale, talking heads attention, and positional embeddings reduces the accuracy for small datasets except Tiny-ImageNet. When our DWConv modules are applied to CaiT models, the accuracy is significantly boosted for small datasets. For

Table 5.5: The experimental results of Swin-Tiny on small datasets

Model	CIFAR-10			CIFAR-100			Tiny-ImageNet		
	Accuracy	Params	FLOPs	Accuracy	Params	FLOPs	Accuracy	Params	FLOPs
Swin-Tiny	93.59	27.5M	4.51G	78.75	27.6M	4.51G	68.24	27.7M	4.51G
Swin-Tiny w/o shift-window	93.36 (-0.23)	27.5M	4.51G	78.51 (-0.24)	27.6M	4.51G	68.30 (+0.06)	27.7M	4.51G
Swin-Tiny (**ours**)	**96.92** (+3.33)	27.6M	4.52G	**83.92** (+5.17)	27.6M	4.52G	**71.96** (+3.72)	27.7M	4.52G
Swin-Tiny w/o shift-window (**ours**)	**97.18** (+3.59)	27.6M	4.52G	**83.38** (+4.63)	27.6M	4.52G	**72.36** (+4.12)	27.7M	4.52G
Swin-Tiny kernel 3+5 (**ours**)	**97.06** (+3.47)	27.7M	4.56G	**83.84** (+5.09)	27.8M	4.56G	**72.74** (+4.50)	27.8M	4.56G

Tiny-ImageNet, the accuracy of CaiT-xxs12 with our proposed DWConv is tremendously improved by around 11% with less number of parameters and FLOPs since the aforementioned modules are eliminated when our approach is applied to CaiT models. Similar to ViT models, CaiT-xxs12 with our method has much higher accuracy than the original CaiT-xxs24 and almost half of the number of parameters and FLOPs compared to CaiT-xxs24 model. In addition, CaiT-xxs12 with our DWConv modules even has much better performance than the original Swin-Transformer (as shown in Table 5.5) that has almost 4x the number of parameters and FLOPs than CaiT-xxs12.

To verify the architecture variant that multiple Transformer blocks are bypassed by the proposed DWConv modules, more experiments are conducted with CaiT, as demonstrated in Table 5.4. The number of blocks indicates how many Transformer blocks are supervised by the DWConv modules in the architecture. The performance drops when more blocks are supervised by DWConv modules, but the accuracy is still much higher than the original models. The variant is appropriate when the layers are deeper to reduce the number of parameters and FLOPs while still maintaining relatively high accuracy.

5.4.1.4 Swin Transformer

The architecture of Swin Transformer consists of four stages with hierarchical feature maps. The size of the feature maps is reduced by 2 on each side by merging adjacent image patches in the following successive stage. Shifted window-based self-attention (Liu et al., 2021) is proposed

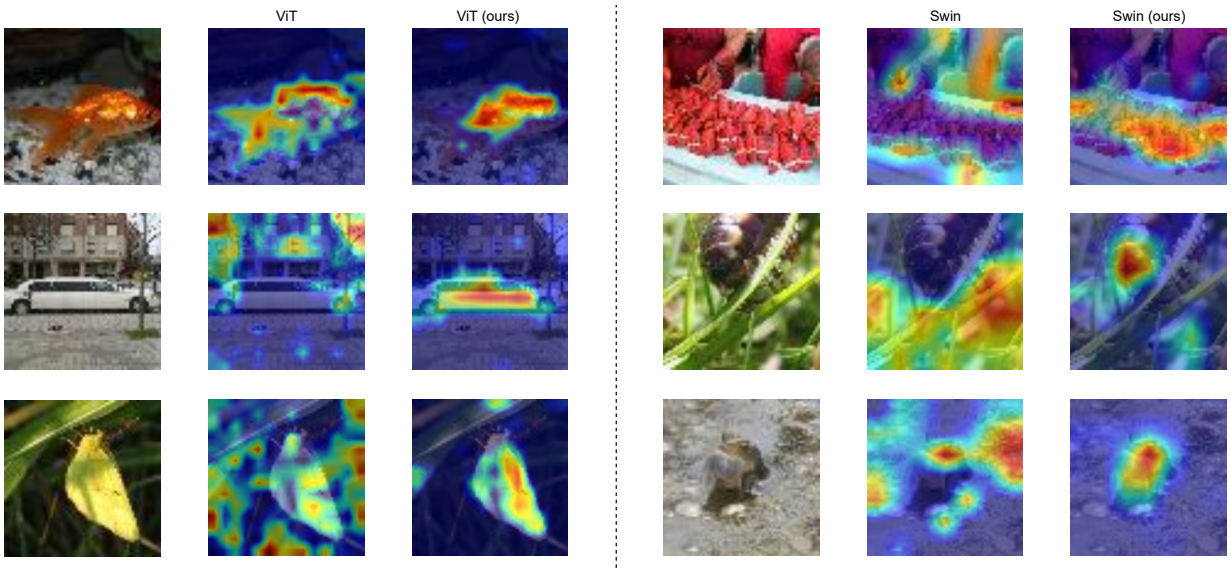

Figure 5.4: Some Grad-CAM visualization with ViT-Tiny and Swin-Tiny models. The vanilla Transformer models tend to capture the global information, as illustrated in the CAM visualizations. With our method, the models are able to capture both local details and global perspectives, particularly when dealing with smaller objects. Please note that the original images in the figure are from the Tiny-ImageNet dataset with a low resolution of 64×64 pixels. Thus, they appear blurred when enlarged.

to extend the view of the tokens instead of limiting the view of the tokens in the windows they are assigned. In the experiments, we investigate the effectiveness of our method applied to Swin Transformer, which is demonstrated in Table 5.5.

The shifted window approach does not have too much effect on the performance of small datasets. Swin Transformer model with our method has much better accuracy than the original model with negligible parameter overhead. In addition, “kernel 3+5" means parallel DWConv modules have kernel size 3 and 5, respectively. Independent parallel DWConv modules with different kernels increase the accuracy in some cases, but the number of parameters and computations would be slightly increased.

We also utilize GRAD-CAM (Selvaraju et al., 2017) to visualize the focus areas of the models, as depicted in Fig. 5.4. The Transformer models exhibit global views of images, while Transformer models might overlook some objects due to a lack of local information, especially when the objects are relatively small. With our method, both global and local information could be captured and

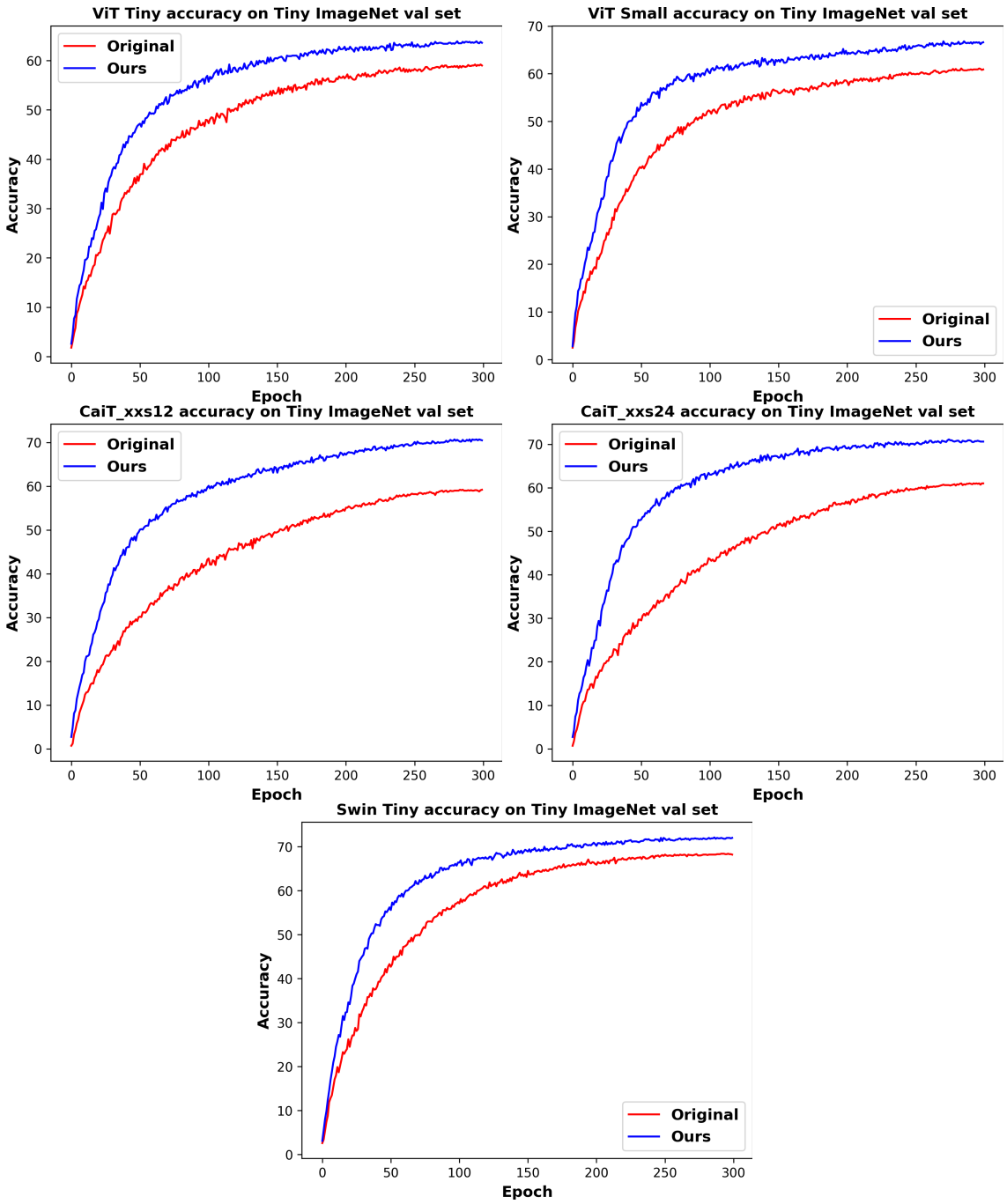

Figure 5.5: The accuracy for val set during the training on Tiny-ImageNet for 300 epochs. The blue curves indicate our method and the red curves are from the original models. The accuracy for val set is recorded for each epoch. The convergence of the models with our approach is much faster than the original models.

enhanced by each other, which could improve the performance of the models.

The convergence of our approach is significantly faster than the original models, which is

demonstrated in Fig. 5.5. The accuracy on val set is recorded for each epoch on Tiny-ImageNet for all models. Our method exhibits much higher performance and considerably faster convergence speed. Our approach could reach a similar accuracy at around or less than 100 epochs while the original models require 300 epochs to attain the same accuracy. Similar performance curves are observed for CIFAR-10 and CIFAR-100.

5.4.2 Classification Performance on ImageNet-1K

In addition to the small datasets, we have also evaluated the models on a relatively large dataset, ImageNet-1K (Russakovsky et al., 2015), to further verify the effectiveness of our approach. ImageNet-1K (Russakovsky et al., 2015) contains nearly 1.3 million images for training and 50k images for validation. For ImageNet-1K (Russakovsky et al., 2015), the batch size is 1024 with 8 NVIDIA V100 GPUs, and the base learning rate is 1e-3.

We utilize CaiT (Touvron et al., 2021b) and Swin Transformer (Liu et al., 2021) to illustrate the performance of our approach on ImageNet. The kernel size for our method is 3×3 and the DWConv module is applied to each Transformer block. When our approach is applied to the models, the positional embeddings and talking heads attention in CaiT are retained for better accuracy, while LayerScale is eliminated. For Swin Transformer, our approach is directly applied to the model without any other changes. The experimental results are demonstrated in Table 6.8. We employ top-1 accuracy to measure the performance of the models and the results of ResNet models are extracted from (Wightman et al., 2021). The performance of CaiT and Swin-Transformer on ImageNet-1K is further boosted (up to 2%) by our method.

Moreover, in comparison to the convolutional counterparts like ResNet (He et al., 2016), our approach still has superior performance with insignificant parameters and FLOPs overhead. Especially when the layers of the Transformer models go deeper (e.g., CaiT-xxs24), the improvement is even higher on ImageNet.

We also visualize the convergence curve of CaiT-xxs24 on ImageNet. As illustrated in Fig. 5.6, the convergence rate of our approach is much faster than the original model by a large margin when

Table 5.6: The performance on ImageNet-1K

Model	Accuracy	Params	FLOPs	Year
ResNet-18 (He et al., 2016) (Wightman et al., 2021)	71.5	11.7M	1.8G	2016
ResNet-34 (He et al., 2016) (Wightman et al., 2021)	76.4	21.8M	3.7G	2016
ResNet-50 (He et al., 2016) (Wightman et al., 2021)	80.4	25.6M	4.1G	2016
SE-ResNet-50 (Hu et al., 2018) (Wightman et al., 2021)	80.0	28.1M	4.1G	2018
DeiT-Ti (Touvron et al., 2021a) (Zhang et al., 2023)	72.2	5.7M	1.3G	2021
Visformer-Ti (Chen et al., 2021b)	78.6	10.3M	1.3G	2021
PVT-Tiny (Wang et al., 2021)	75.1	13.2M	1.9G	2021
DeiT-S (Touvron et al., 2021a) (Zhang et al., 2023)	79.9	22.1M	4.6G	2021
PVT-Small (Wang et al., 2021)	79.8	24.5M	3.8G	2021
MSG-T (Fang et al., 2022)	80.9	28M	4.6G	2022
DiT-B1 (Ma et al., 2023)	79.9	30.3M	2.0G	2023
Visformer-S (Chen et al., 2021b)	81.5	40.2M	4.9G	2021
CaiT-xxs12	74.85	6.6M	1.3G	-
CaiT-xxs24	77.66	11.9M	2.5G	-
Swin-Tiny	81.14	28.3M	4.5G	-
CaiT-xxs12 (**ours**)	**75.89** (+1.04)	6.6M	1.3G	-
CaiT-xxs24 (**ours**)	**79.66** (+2.00)	12.0M	2.5G	-
Swin-Tiny (**ours**)	**81.73** (+0.59)	28.3M	4.5G	-

the epoch is less than 100. This experiment indicates that our proposed approach could achieve higher accuracy with significantly faster convergence speed on a relatively large dataset.

5.4.3 Object Detection and Instance Segmentation

In addition to Image Classification, we apply the proposed approach to object detection and instance segmentation and conduct the experiments on COCO dataset (Lin et al., 2014) with Mask RCNN (He et al., 2017) and Cascade Mask R-CNN (Cai & Vasconcelos, 2018). The backbone utilized in the experiments is Swin-Tiny (Liu et al., 2021). The models are trained from scratch without pre-trained backbones.

For the experimental settings, we employ AdamW (Kingma & Ba, 2014) as the optimizer and

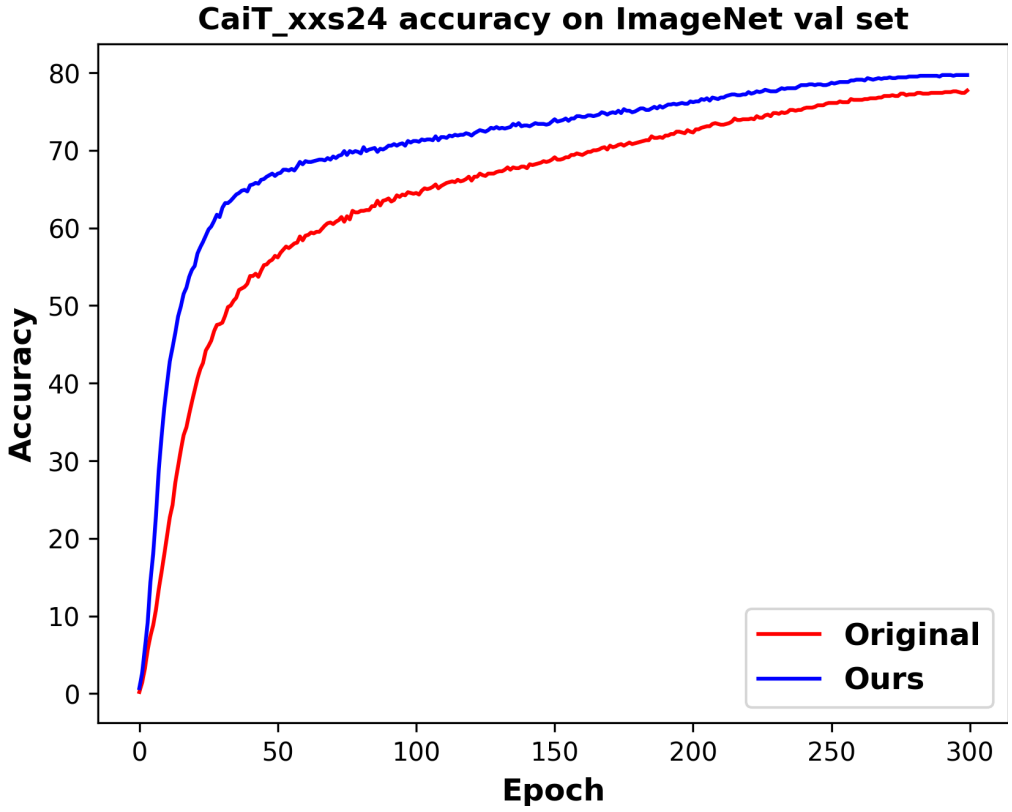

Figure 5.6: The accuracy of CaiT-xxs24 for val set during the training on ImageNet for 300 epochs. The blue curve demonstrates our method. The convergence rate for our approach is much faster than the original model.

the warmup iterations are 500. We utilize four NVIDIA P100 GPUs to train the model with 2 samples for each GPU. The initial learning rate is set at 5e-5 and the learning rate is reduced by 10 at epochs 9 and 12, respectively. The image scale for the experiments is 1333×800. The total epochs for the experiments are 12.

The results for the experiments of object detection and instance segmentation are illustrated in Table 5.7, where “Cas Mask-RCNN" stands for “Cascade Mask-RCNN". From the definition in COCO (Lin et al., 2014), “mAP" refers to the average precision results that are averaged over all classes. The average precision is calculated by averaging the results over IoU thresholds from 0.5 to 0.95 with a step of 0.05. Additionally, “AP_{50}" represents the average precision computed using only the IoU threshold of 0.5 and “AP_{75}" indicates the average precision computed using only the IoU threshold of 0.75. Additionally, the visualization of the original method and our

Table 5.7: Experiments for Object Detection and Instance Segmentation

Model	Object Detection			Instance Segmentation		
	mAP	AP_{50}	AP_{75}	*mAP*	AP_{50}	AP_{75}
Mask-RCNN	28.2	48.3	29.3	27.4	45.6	28.7
Mask-RCNN (**ours**)	**30.6**	**50.2**	**32.6**	**28.9**	**47.4**	**30.6**
Cas Mask-RCNN	35.3	52.5	38.0	31.4	49.8	33.7
Cas Mask-RCNN (**ours**)	**36.2**	**53.0**	**39.1**	**32.1**	**50.6**	**34.5**

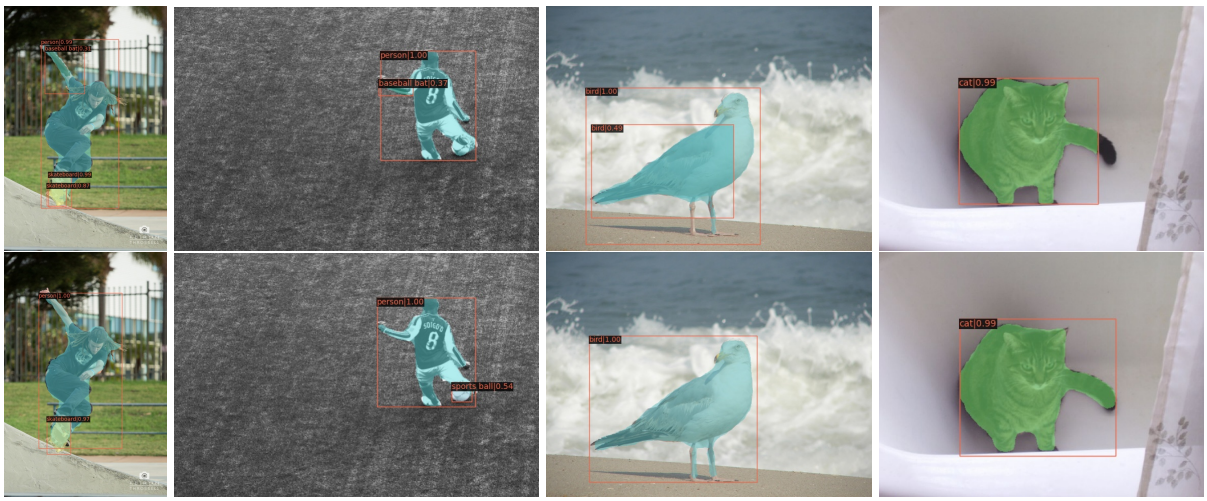

Figure 5.7: The visualization of object detection and instance segmentation between the original method (the first row) and ours (the second row) with Mask-RCNN. The results demonstrate that our approach better detects small objects and produces more accurately predicted boundaries for the objects.

proposed approach with Mask-RCNN (He et al., 2017) is illustrated in Fig. 5.7. In Fig. 5.7, the first row demonstrates the visualization results of the original model and the second row indicates the visualization results of our proposed model.

The experimental results clearly show that our approach improves the performance of backbone networks for object detection and instance segmentation, demonstrating the effectiveness of our proposed method across various vision tasks. The effectiveness likely stems from the fine-detailed information captured by the proposed DWConv module. Object detection and instance segmentation require detailed information for predicting object boundaries and pixel-level labels, respectively. Vision Transformers might lack the ability to capture extensive fine-detailed information, especially when used as the backbone. Our proposed DWConv module complements this

limitation with minimal overhead.

5.4.4 Analysis

The proposed DWConv module, which bypasses the entire Transformer block, demonstrates higher accuracy for image classification, object detection, and instance segmentation when the models are trained from scratch. Additionally, this module enables Transformer models to achieve a much faster convergence rate for image classification, especially on relatively small datasets. Furthermore, our approach can significantly enhance small-size Transformer models, even surpassing large-size original Transformer models with substantially more parameters and computations on small datasets for image classification. Our approach illustrates both the effectiveness and efficiency of Vision Transformer models.

Although our architecture performs well on relatively small datasets due to the inductive bias introduced by the DWConv module, the improvement might not be as pronounced when the dataset is relatively large. The abundant data can mitigate the drawbacks of Transformer models. In addition, since the proposed DWConv module is lightweight and plug-and-play, it may not show significant improvement when models have a large number of parameters and computations. A large number of parameters and computations can increase the representation ability of Transformer models and potentially remedy the lack of inductive bias, albeit inefficiently. However, our proposed models enhance both the effectiveness and efficiency of Transformer models, achieving higher accuracy than some large models, despite having significantly fewer parameters and computations.

Moreover, our method may not show significant improvement for transfer learning. One strength of our proposed approach is that our light-weight module can be utilized in most Transformer models, potentially enhancing performance, particularly on small datasets, when training from scratch. The experiments in this chapter are all conducted with training from scratch. The possible reason for the lack of significant improvement in transfer learning is that pre-trained models already possess substantial representation ability, reducing the necessity for the inductive bias

introduced by our approach. Thus, our proposed module may not provide much additional benefit when pre-trained models are applied to other tasks or datasets for fine-tuning.

5.5 Conclusion

In this chapter, we have presented a straightforward yet impactful approach that utilizes Depth-Wise Convolution modules to bypass Transformer blocks, enabling Vision Transformer models to capture both global and local information with minimal overhead. Extensive experimental evaluations show that small Transformer models, when equipped with our method, outperform larger Transformer models with significantly more parameters and FLOPs on small datasets for image classification. Our approach also significantly improves performance on ImageNet-1K (Russakovsky et al., 2015) for classification and COCO (Lin et al., 2014) for object detection and instance segmentation when trained from scratch. Additionally, we introduce several architecture variants tailored to different models and objectives. We anticipate that our method will inspire further research on Vision Transformers, particularly in the context of small datasets.

Chapter 6

Improving Vision Transformers by Overlapping Heads in Multi-Head Self-Attention

Abstract

Vision Transformers have made remarkable progress in recent years, achieving state-of-the-art performance in most vision tasks. A key component of this success is due to the introduction of the Multi-Head Self-Attention (MHSA) module, which enables each head to learn different representations by applying the attention mechanism independently. In this chapter, we empirically demonstrate that Vision Transformers can be further enhanced by overlapping the heads in MHSA. We introduce Multi-Overlapped-Head Self-Attention (MOHSA), where heads are overlapped with their two adjacent heads for queries, keys, and values, while zero-padding is employed for the first and last heads, which have only one neighboring head. Various paradigms for overlapping ratios are proposed to fully investigate the optimal performance of our approach. The proposed approach is evaluated using five Transformer models on four benchmark datasets and yields a significant performance boost.

6.1 Introduction

Since the introduction of Transformer (Vaswani et al., 2017) from language to vision, Vision Transformers have gradually dominated many vision tasks such as image recognition (Dosovitskiy et al., 2020) (Liu et al., 2021) (Chen et al., 2023) and object detection (Carion et al., 2020) (Ma et al., 2021) (Zhao et al., 2024) (Hou et al., 2025). Due to the global attention mech-

anism in Transformers, Vision Transformers could yield better performance in vision tasks when the training data is abundant and the training epochs are enough. Although the convergence rate of Vision Transformers is frequently slower than their Convolutional counterparts, global information communication is one of the key elements for the success of Transformer models.

Multi-Head Self-Attention (MHSA) (Vaswani et al., 2017) is the core module for Transformer models so that the self-attention mechanism can be implemented in multiple heads to learn different representations. The self-attention mechanism is originally designed for processing language data to capture the long-range relationship between words in language. When self-attention is applied to vision tasks for image classification and recognition, each image patch token could interact with all other patch tokens for the global views of the image, which is significant for the excellent performance of Vision Transformers. MHSA represents that the queries, keys and values are split into different heads and the self-attention is computed in each head independently. Transformer models with Multi-Head Attention frequently yield better performance than those with Single-Head Attention. Thus, most Transformer models utilize Self-Attention with multiple heads for better representation learning and performance. Even though MHSA boosts the performance of the Transformer models, the relationships and interactions between different heads are rarely investigated.

The success of Transformer models mainly lies on the effective information exchange between different tokens so that each token can have global views of the context information. Due to the superior performance of MHSA, most Transformer models utilize MHSA by default. However, the queries, keys and values are divided for each head without overlapping and there is no information exchange when the attention is computed in each head. In other words, when calculating the attention in the current head, it does not have the information in other heads. Although the tokens will be processed by linear projections after the attention, the information exchange is only limited to each token.

In this chapter, we claim that information exchange during attention calculation in each head can improve the performance of Vision Transformers. This can be realized by overlapping queries,

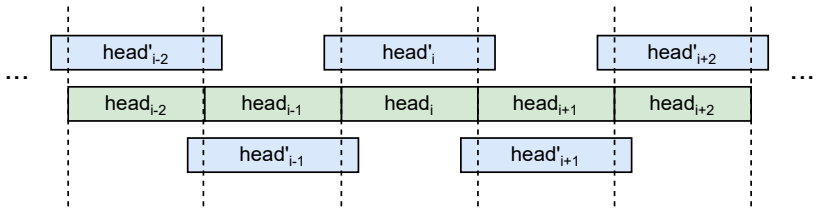

Figure 6.1: The proposed multi-overlapped-head method (blue) vs the original multi-head method (green). Instead of hard division of the heads, our approach softly splits the heads by overlapping each head with its neighboring heads.

keys and values in each head with queries, keys and values of neighboring heads. For this purpose, we propose Multi-Overlapped-Head Self-Attention (MOHSA) to improve the Multi-Head Self-Attention mechanism by overlapping the heads so that Q, K, and V in each head are overlapped by Q, K, and V of their neighboring heads when the attention is calculated, as illustrated in Fig. 6.1. By overlapping the heads, the information in other heads could also be involved in the calculation of the attention in the current head. Since the overlapping would slightly increase the dimension of the tokens after concatenating tokens from different heads, the linear projections would decrease the dimension of the tokens to the original size. The information communication between the heads could yield better performance for Vision Transformers. Moreover, we design various paradigms for overlapping ratios to investigate the best performance of our proposed approach. We explore several Vision Transformer models on CIFAR-10 (Krizhevsky et al., 2009), CIFAR-100 (Krizhevsky et al., 2009), Tiny-ImageNet (Le & Yang, 2015) and ImageNet-1k (Russakovsky et al., 2015) to verify the effectiveness of our proposed MOHSA.

Our major contributions are summarized below.

- We propose Multi-Overlapped-Head Self-Attention (MOHSA) and demonstrate that Vision Transformer models could be improved by overlapping the queries, keys and values of the current head with the queries, keys and values of adjacent heads when the attention is computed. Several Vision Transformer models are exploited on various datasets to demonstrate the effectiveness of our proposed method.
- Various variants based on the overlap dimensions are proposed to thoroughly investigate the

optimal performance of MOHSA. The proposed MOHSA could be integrated into Vision Transformer models to enhance their performance with negligible overhead.

- To the best of our knowledge, our work is the first to investigate the overlapping approach between different heads for MHSA when the attention is calculated.

6.2 Related Work

6.2.1 Vision Transformers

Vision Transformer (Dosovitskiy et al., 2020) utilizes the image patches which are embedded as tokens and a class token as the token inputs for the Transformer encoder to recognize the images. Swin-Transformer (Liu et al., 2021) reduces the computational cost by limiting the implementation of attention calculation in each window and expands the view of the tokens by shifting the windows. Shuffle Transformer (Huang et al., 2021) proposes spatial shuffle to exchange information across windows. MSG-Transformer (Fang et al., 2022) harnesses MSG tokens to represent local windows and the information communication between windows is implemented with MSG tokens. PVT (Wang et al., 2021) designs a Vision Transformer with a hierarchical structure without convolutions by shrinking the feature maps gradually. CaiT (Touvron et al., 2021b) and DeepViT (Zhou et al., 2021) investigate Vision Transformers with deeper layers. A multi-layer dense attention decoder was proposed in (Patel et al., 2024) for segmentation. MobileViT series (Mehta & Rastegari, 2021) (Mehta & Rastegari, 2022) (Wadekar & Chaurasia, 2022) and EdgeViTs (Pan et al., 2022) explore Vision Transformers on mobile-level applications. Some works (Wu et al., 2021) (Xiao et al., 2021) (Guo et al., 2022) (Ibtehaz et al., 2024) (Zhang et al., 2025) introduce convolutions into the Vision Transformers to take advantage of both convolutions and Transformers. PiT (Nguyen et al., 2024) explores the possibility of using individual pixels instead of patches as the tokens for Vision Transformers. ViTAR (Fan et al., 2024) adjusts Vision Transformers on various image resolutions. PartialFormer (Vo et al., 2024) proposes partial attention for more efficient Vision Transformers on various vision tasks.

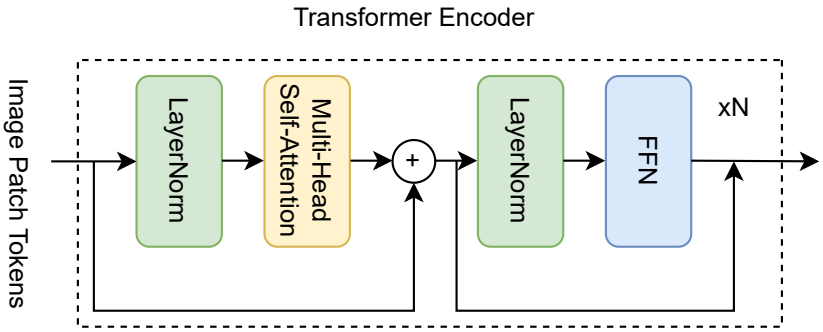

Figure 6.2: The Transformer encoder for a typical Transformer model. The Transformer encoder is exploited in Vision Transformer for image classification. N indicates the number of layers for the Transformer encoder.

6.2.2 Attention Heads Interaction

DeepViT (Zhou et al., 2021) explores a deeper layer Vision Transformer by proposing re-attention that mixes the attention maps among all heads with a learnable matrix before multiplying with the values. The re-attention is similar to talking-heads attention (Shazeer et al., 2020) which is originally employed for language tasks and also utilized by CaiT (Touvron et al., 2021b). Talking-heads attention (Shazeer et al., 2020) applies learnable linear projections to the heads in Multi-Head Attention before and after the softmax function to exchange the information for the attention maps between heads in the attention module.

The aforementioned methods mix attention maps using linear projections, allowing information exchange between heads only after the attention maps are calculated. However, the mixed attention maps are then applied to values that do not have information exchange between heads. In contrast, we propose a method that enables communication between heads during the attention computation by overlapping the heads with Q, K, and V. Although this approach introduces a slight increase in computation and parameters, it significantly enhances the performance of Vision Transformer models across various datasets.

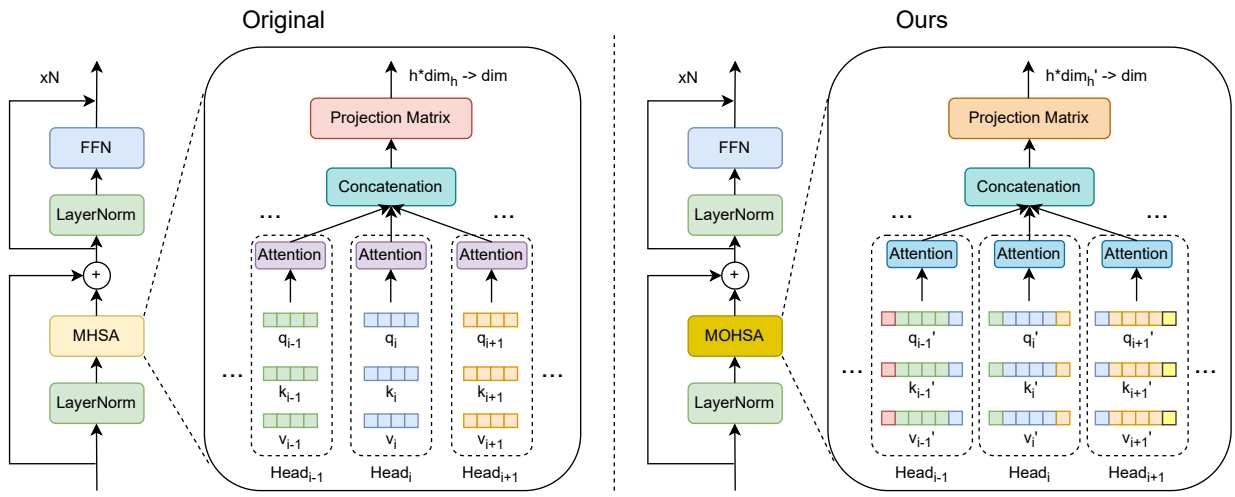

Figure 6.3: Our proposed Multi-Overlapped-Head Self-Attention. MHSA represents the original implementation of Multi-Head Self-Attention with hard division of heads and MOHSA indicates our proposed Multi-Overlapped-Head Self-Attention with soft division of heads. In the original Vision Transformer (left), Q, K, and V are split for different heads and the attention is computed for each head independently. To exchange the information between heads when the attention is calculated, we propose to overlap Q, K, V with Q, K, and V in adjacent heads (right). Since overlapped heads would slightly increase the number of dimensions, the projection matrix would project the concatenated heads to the original token dimension.

6.3 Approach

6.3.1 Multi-Head Self-Attention

Transformer (Vaswani et al., 2017) has an attention mechanism to compute the long-range relationship between tokens. The attention calculation includes queries, keys, values and the dimension of queries and keys d_k. The queries and keys are implemented dot product to calculate the weights which are utilized to the values to compute the final results. The matrix format of attention calculation (Vaswani et al., 2017) is illustrated in Eq. (6.1). $\sqrt{d_k}$ is employed to prevent the large value input for the softmax function after computing the dot product (Vaswani et al., 2017).

$$Attention(Q,K,V) = Softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{6.1}$$

Multi-Head Self-Attention (Vaswani et al., 2017) is utilized in the Transformer model for

better performance so that different heads can learn different representations, which is better than one-head attention (Vaswani et al., 2017). The attention in each head is illustrated in Eq. (6.2). Q, K, and V are divided as Q_i, K_i, and V_i in each head respectively and the attention calculation is implemented in each head independently for different aspects of learning.

$$Head_i = Attention(Q_i, K_i, V_i) \tag{6.2}$$

$$MHSA(Q, K, V) = Concate(Head_1, ..., Head_n)W \tag{6.3}$$

Finally, MHSA could be represented as Eq. (6.3). The results of all heads are concatenated and W is the projection matrix.

Vision Transformer (Dosovitskiy et al., 2020) is the Transformer model applied to vision tasks, mainly image classification and recognition. For Vision Transformer, only the encoder is utilized for feature extraction, as demonstrated in Fig. 6.2. The images are divided as same-size patches which are embedded as tokens for the Vision Transformer. Layer Normalization (Ba et al., 2016) is often employed as the normalization before MHSA and FFN. Vision Transformer (Dosovitskiy et al., 2020) could be applied to vision tasks and achieve significant performance on vision tasks due to the effective information exchange for attention mechanism. The self-attention mechanism calculates the dot product of each token with all other tokens effectively exchanging the information between them. In vision tasks like image classification and recognition, each patch token has a global view of the image, which is important for each patch to attain the context information.

Nonetheless, the attention is implemented independently in each head and one head does not have the attention information of other heads when the attention is computed. Although the projection matrix is implemented after all the heads are concatenated, only the information in each token is exchanged. There is no information exchange when computing the attention in each head. Thus we propose to overlap the information of neighboring heads to enhance the information exchange when the attention is computed in each head.

6.3.2 Multi-Overlapped-Head Self-Attention

Based on the aforementioned analysis, we have proposed a simple yet effective approach that improves Multi-Head Self-Attention to enhance the performance of Vision Transformer. To make information exchange between heads, we exploit soft division instead of hard division when Q, K, and V are divided into different heads. The process could be illustrated in Eq. (6.4) for Q, K, and V respectively. We utilize "part" to illustrate partial overlapping with adjacent heads in Eq. (6.4).

$$\begin{aligned} Q_i' &= Concate(part(Q_{i-1}), Q_i, part(Q_{i+1})) \\ K_i' &= Concate(part(K_{i-1}), K_i, part(K_{i+1})) \\ V_i' &= Concate(part(V_{i-1}), V_i, part(V_{i+1})) \end{aligned} \tag{6.4}$$

In Eq. (6.4), Q_i, K_i and V_i are the original hard division results for $Head_i$, which is shown in Eq. (6.2). For soft division, Q_i', K_i' and V_i' also include partial information in their neighboring heads so that Q, K, V are overlapped with two adjacent heads, which is demonstrated in Fig. 6.3. The left figure indicates the original implementation of the hard division of Q, K, V to different heads and the right figure represents our proposed implementation of the soft division of Q', K', V' to different heads. The Q_i, K_i, and V_i would overlap with two adjacent heads to construct Q_i', K_i', and V_i' for calculating the attention. For the first head and the last head which only have one adjacent head, zero padding is utilized to construct two neighboring heads for the first and the last head.

$$Head_i' = Attention(Q_i', K_i', V_i') \tag{6.5}$$

$$MOHSA(Q,K,V) = Concate(Head_1', ..., Head_n')W' \tag{6.6}$$

After the attention for the overlapped Q', K', and V' is calculated for each head, the results are concatenated together, as illustrated in Eq. 6.5-6.6. The overlapped heads would slightly increase the dimension of the tokens after concatenation. Thus the projection matrix W' would project the concatenated dimension ($h * dim_h'$) to the original token dimension (dim) so that the token could

Table 6.1: The variants for overlapping ratios

Methods	Description
inc (x layers)	increase overlap dim by 1 every x layers
dec (x layers)	decrease overlap dim by 1 every x layers
0-indexed	start (end) overlap dim from (to) 0
1-indexed	start (end) overlap dim from (to) 1

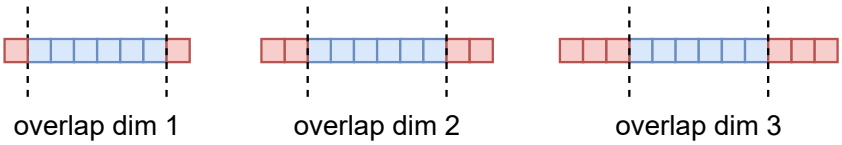

Figure 6.4: The illustration of the overlap dimensions. The blue parts demonstrate the original non-overlapping heads and the red parts indicate the overlapped parts from adjacent heads. The number of overlap dimensions is the overlap dimension of one side adjacent head.

be fed into the next layer with the same dimension. The projection matrix W in Eq. (6.3) projects the concatenated non-overlapped heads ($h * dim_h$) to the original token dimension (dim). Since the dimension of overlapped heads dim'_h is slightly larger than the dimension of non-overlapped heads dim_h and the number of heads h is unchanged, the projection matrix W' in our proposed MOHSA would have slightly more parameters than the projection matrix W in the original MHSA.

6.3.3 Overlapping Ratios

The overlapping ratios for the overlapped heads are crucial to the effectiveness of the proposed approach. In the experiments, we design several paradigms for the overlapping ratios. In this work, we utilize overlap dimensions to demonstrate the overlapping ratios, as illustrated in Fig. 6.4. From Fig. 6.4, the number of overlap dimensions for each head is the overlap dimension of its one-side adjacent head. In Fig. 6.4, the blue sections are the original parts of the heads and the red sections are overlapped parts with two adjacent heads. For the first and the last head which have only one side adjacent head, zero padding is exploited for the missing neighboring heads.

In addition to the fixed overlap dimensions for all layers, we also implement some variants of

Table 6.2: The parameter settings for the models

Models	heads	depths	MLP ratio
ViT-Tiny	12	12	4
ViT-Small	12	12	4
CaiT-xxs12	4	12 (2)	4
CaiT-xxs24	4	24 (2)	4
Swin-Tiny	(3, 6, 12, 24)	(2, 2, 6, 2)	4

the overlapping ratios by changing the overlap dimensions according to the depths of the layers. In this chapter, one layer includes the attention module and the FFN module, as illustrated in Fig. 6.2 which represents one layer of the Transformer encoder. The variants for the overlapping ratios are demonstrated in Table 6.1. In Table 6.1, “inc (x layers)" represents the overlap dimension is increased by 1 every x layers, and “dec (x layers)" indicates the overlap dimension is decreased by 1 every x layers, which is a reverse process of "inc (x layers)" based on the depths of the layers. In addition, “0-indexed" illustrates the overlap dimension starts from 0 for “inc" and ends at 0 for “dec", and “1-indexed" demonstrates the overlap dimension starts from 1 for “inc" and ends at 1 for “dec". For instance, inc (2 layers) with “0-indexed" (represented by “inc-0 (2)" in the experiments) for a total of 12 layers has overlap dimensions (0, 0, 1, 1, 2, 2, 3, 3, 4, 4, 5, 5) for layer from 1 to 12, and dec (1 layer) with “1-indexed" (represented by “dec-1 (1)" in the experiments) for total 12 layers has overlap dimensions (12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1) for layer from 1 to 12.

6.4 Experiments

In the experiments, we select three representative models to investigate the proposed approach: vanilla ViT (Dosovitskiy et al., 2020), deeper layer Transformer CaiT (Touvron et al., 2021b), and window-based hierarchical Transformer Swin-Transformer (Liu et al., 2021). More specifically, ViT-Tiny (Dosovitskiy et al., 2020), ViT-Small (Dosovitskiy et al., 2020), CaiT-xxs12 (Touvron et al., 2021b), CaiT-xxs24 (Touvron et al., 2021b) and Swin-Tiny (Liu et al., 2021) are selected to explore the effectiveness of the proposed approach. The parameter settings for the models are

Table 6.3: The ablation study on ViT for CIFAR-10

Methods	ViT-Tiny			ViT-Small		
	Accuracy	Params	FLOPs	Accuracy	Params	FLOPs
original	84.31	5.5M	1.3G	86.53	21.7M	4.6G
fixed 1	84.94	5.6M	1.3G	86.98	21.8M	4.7G
fixed half	85.34	6.0M	1.5G	86.84	23.4M	5.3G
inc-0 (1)	85.54	5.8M	1.4G	**87.30**	22.3M	4.9G
inc-0 (3)	84.83	5.6M	1.3G	87.24	21.8M	4.7G
inc-0 (6)	**85.65**	5.5M	1.3G	86.74	21.7M	4.6G
inc-1 (1)	85.35	5.9M	1.5G	86.98	22.4M	4.9G
inc-1 (3)	84.31	5.7M	1.3G	86.83	21.9M	4.7G
inc-1 (4)	85.18	5.6M	1.3G	86.81	21.9M	4.7G
dec-0 (1)	85.43	5.8M	1.4G	86.67	22.3M	4.9G
dec-0 (3)	84.81	5.6M	1.3G	86.35	21.8M	4.7G
dec-1 (1)	85.25	5.9M	1.5G	86.87	22.4M	4.9G
dec-1 (3)	85.12	5.7M	1.3G	86.84	21.9M	4.7G

shown in Table 6.2. For Swin-Tiny, the number of heads and depths are varied based on the stages. CaiT-xxs12 and CaiT-xxs24 have two extra class layers. The models will be trained and evaluated on CIFAR-10 (Krizhevsky et al., 2009), CIFAR-100 (Krizhevsky et al., 2009), Tiny-ImageNet (Le & Yang, 2015) and ImageNet-1k (Russakovsky et al., 2015).

The experiments are implemented by 100 epochs with 20 epochs warmup. The optimizer is AdamW (Kingma & Ba, 2014). The experiments for CIFAR-10 (Krizhevsky et al., 2009), CIFAR-100 (Krizhevsky et al., 2009), Tiny-ImageNet (Le & Yang, 2015) are run on 4 NVIDIA P100 GPUs with total batch size 128. The experiments for ImageNet-1k (Russakovsky et al., 2015) are conducted on 4 NVIDIA V100 GPUs with a total batch size of 512. The initial learning rate for ViT and CaiT is 0.0005. For Swin-Tiny, the initial learning rate for Tiny-ImageNet is 0.00025, and for ImageNet is 0.0005. The images are resized to 224 for the experiments. Some other settings follow the settings in Swin-Transformer (Liu et al., 2021).

Table 6.4: The ablation study on CaiT-xxs12 for CIFAR-10

Methods	Accuracy	Params	FLOPs
original	77.04	6.4M	1.3G
fixed 1	78.14	6.4M	1.3G
inc-0 (1)	78.79	6.5M	1.4G
inc-0 (2)	78.63	6.5M	1.3G
inc-0 (3)	**80.04 (+3.00)**	6.4M	1.3G

6.4.1 CIFAR-10

CIFAR-10 (Krizhevsky et al., 2009) includes 50k training images and 10k testing images. Among all 60k images, there are 10 classes and each class has 6k images. The ablation study for ViT-Tiny and ViT-Small on CIFAR-10 is demonstrated in Table 6.3. In Table 6.3, “fixed 1" represents the overlap dimension is 1 for all layers, and “fixed half" indicates the overlap dimension is half of the head dimension for all layers. “inc” indicates the overlap dimension is increased with the depths of the layers increasing and “dec" represents the overlap dimension is decreased with the layers going deeper. The two numbers following “inc” or “dec", for instance, “inc-0 (3)" illustrates the overlap dimension is increased by 1 every 3 layers and it is “0-indexed". More details about the variants of overlapping ratios can be found in Sec. 6.3.3.

The best accuracy for ViT-Tiny is 85.65 for variant “inc-0 (6)" which increases the overlap dimension by 1 every 6 layers with “0-indexed" and the best accuracy for ViT-Small is 87.30 for variant “inc-0 (1)" which increases the overlap dimension by 1 every 1 layer with “0-indexed". For ViT-Tiny, “fixed half" demonstrates higher accuracy than “fixed 1". For ViT-Small, “fixed half" has lower accuracy than “fixed 1".

Although the overlapped heads would slightly increase the computations and parameters, from the experimental results, we can see that the increased numbers of parameters and FLOPs are negligible. Table 6.4 illustrates the experimental results of CaiT-xxs12 on CIFAR-10. The best result 80.04 significantly boosts the performance of CaiT-xxs12 by 3% with negligible overhead.

Table 6.5: The ablation study on ViT for CIFAR-100

Methods	ViT-Tiny			ViT-Small		
	Accuracy	Params	FLOPs	Accuracy	Params	FLOPs
original	61.88	5.5M	1.3G	63.78	21.7M	4.6G
fixed 1	62.17	5.6M	1.3G	63.90	21.8M	4.7G
fixed half	62.61	6.0M	1.5G	63.76	23.5M	5.3G
inc-0 (1)	62.86	5.8M	1.4G	64.83	22.3M	4.9G
inc-0 (2)	**63.01**	5.7M	1.3G	64.21	22.0M	4.7G
inc-1 (1)	62.80	5.9M	1.5G	64.49	22.4M	4.9G
inc-1 (2)	62.77	5.7M	1.4G	64.24	22.1M	4.8G
dec-0 (1)	62.45	5.8M	1.4G	64.62	22.3M	4.9G
dec-1 (3)	62.25	5.7M	1.3G	**64.97**	22.0M	4.7G

6.4.2 CIFAR-100

CIFAR-100 (Krizhevsky et al., 2009) includes 50k training images and 10k testing images. There are 100 classes and each class has 600 images. Table 6.5 demonstrates the experiments of ViT-Tiny and ViT-Small on CIFAR-100. The best result for ViT-Tiny is 63.01 with “inc-0 (2)" which increases the overlap dimension by 1 every 2 layers with “0-indexed" and for ViT-small is 64.97 with “dec-1 (3)" which decreases the overlap dimension by 1 every 3 layers with “1-indexed". Both results improve the models by more than 1% than the original models. Similar to the results on CIFAR-10, “fixed half" has higher accuracy for ViT-Tiny, and “fixed 1" illustrates better performance for ViT-Small. In addition, “fixed half" does not demonstrate effectiveness for ViT-Small.

Table 6.6 illustrates the performance of CaiT on CIFAR-100. For CaiT-xxs12, using 1 overlap dimension for all layers could boost the accuracy by nearly 2%. For CaiT-xxs24, the accuracy is significantly enhanced by 5% with “inc-1 (3)" which increases the overlap dimension by 1 every 3 layers with “1-indexed". Additionally, “inc-1 (3)" is much better than using half dimension of the head as the overlap dimension which utilizes more parameters and FLOPs.

Table 6.6: The ablation study on CaiT for CIFAR-100

Models	Methods	Accuracy	Params	FLOPs
CaiT-xxs12	original	50.95	6.4M	1.3G
	fixed 1	**52.94 (+1.99)**	6.4M	1.3G
CaiT-xxs24	original	50.94	11.8M	2.5G
	fixed half	52.47	12.7M	3.1G
	inc-1 (1)	55.84	12.2M	2.8G
	inc-1 (3)	**55.94 (+5.00)**	11.9M	2.6G

6.4.3 Tiny-ImageNet

Tiny-ImageNet (Le & Yang, 2015) includes 100k training images and 10k validation images. There are 200 classes in the dataset. Each class has 500 images for training and 50 images for validation. The experimental results on Tiny-ImageNet are demonstrated in Table 6.7. The accuracy of ViT-Tiny could be enhanced by 1.61% utilizing “inc-1 (2)" that increases the overlap dimension by 1 every 2 layers with “1-indexed". ViT-Small is boosted by 1.27% with “inc-1 (1)" that increases the overlap dimension by 1 every 1 layer with “1-indexed". Moreover, using half of the head dimension as the overlap dimension is better than using 1 as the overlap dimension for all layers for ViT-Tiny, while it is not the case for ViT-Small.

For CaiT-xxs12 and CaiT-xxs24, “fixed 1" that utilizes fixed overlap dimension 1 for all layers could significantly improve the accuracy by 2.39% and 3.58%, respectively. Furthermore, the accuracy is enormously enhanced by 7.41% for CaiT-xxs24 with “inc-1 (1)" which increases the overlap dimension by 1 every 1 layer with “1-indexed". The effectiveness of our method is significant on CaiT models and we can ignore the slightly increased parameters and computations.

For Swin-Tiny, “fixed 1" paradigm could greatly enhance the accuracy by 1.23%, and the performance could be further boosted by a large margin with “dec-1 (1)" that decreases the overlap dimension by 1 every 1 layer with “1-indexed". The effectiveness of overlapped heads is also demonstrated on the Transformer model with window-based hierarchical architecture.

Table 6.7: The experimental results on Tiny-ImageNet

Models	Methods	Accuracy	Params	FLOPs
ViT-Tiny	original	50.79	5.6M	1.3G
	fixed 1	51.44	5.6M	1.3G
	fixed half	52.32	6.0M	1.5G
	inc-0 (1)	51.45	5.9M	1.4G
	inc-1 (2)	**52.40 (+1.61)**	5.7M	1.4G
	dec-0 (1)	52.21	5.9M	1.4G
	dec-1 (2)	52.20	5.7M	1.4G
ViT-Small	original	54.53	21.7M	4.6G
	fixed 1	55.15	21.8M	4.7G
	fixed half	54.65	23.5M	5.3G
	inc-0 (2)	55.30	22.0M	4.7G
	inc-1 (1)	**55.80 (+1.27)**	22.4M	4.9G
	dec-1 (3)	55.46	22.0M	4.7G
CaiT-xxs12	original	42.66	6.5M	1.3G
	fixed 1	**45.05 (+2.39)**	6.5M	1.3G
CaiT-xxs24	original	42.46	11.8M	2.5G
	fixed 1	46.04	11.8M	2.6G
	inc-0 (2)	46.98	12.0M	2.7G
	inc-1 (1)	**49.87 (+7.41)**	12.2M	2.8G
Swin-Tiny	original	57.04	27.7M	4.5G
	fixed 1	58.27	27.8M	4.5G
	inc-0 (6)	58.67	27.8M	4.5G
	inc-1 (3)	58.58	28.1M	4.6G
	dec-0 (1)	58.59	28.0M	4.7G
	dec-1 (1)	**58.90 (+1.86)**	28.1M	4.7G

6.4.4 ImageNet

ImageNet-1k (Russakovsky et al., 2015) includes 1k classes with more than 1 million training images and 50k validation images. By training and testing on ImageNet, we could investigate the effectiveness of our proposed approach on large datasets. The experimental results on ImageNet

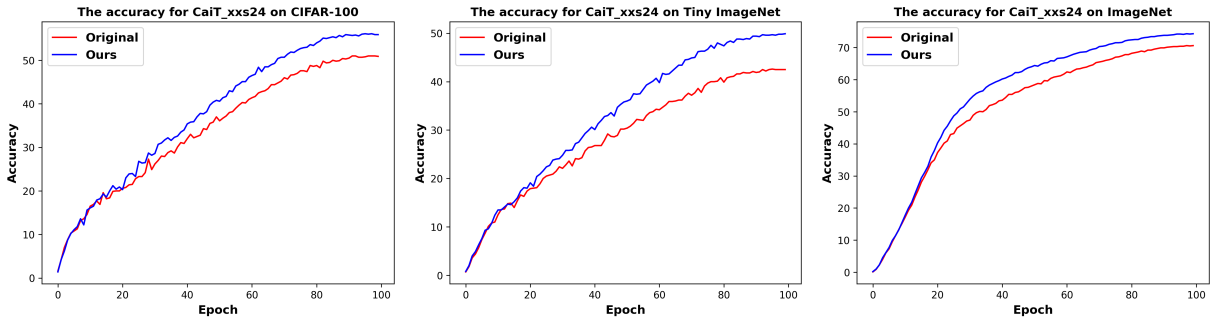

Figure 6.5: The accuracy comparison during the training process. The accuracy comparison between the original method and our proposed approach for CaiT-xxs24 on val or test set of CIFAR-100, Tiny-ImageNet, and ImageNet is illustrated from left to right. The blue curve demonstrates our approach with the best performance and the red curve indicates the original method.

are demonstrated in Table 6.8. “inc-0 (1)" and “inc-1 (1)" indicate the overlap dimension is increased by 1 every 1 layer with “0-indexed" and “1-indexed", respectively. “dec-0 (1)" and “dec-1 (1)" represent the overlap dimension is decreased by 1 every 1 layer with “0-indexed" and “1-indexed", respectively. Almost all models equipped with our proposed MOHSA have significant improvements on ImageNet.

ViT-Tiny could be boosted by more than 1% with “fixed half" that uses half of the head dimension as the overlap dimension for all layers. ViT-Small is enhanced by 0.79% with “dec-0 (1)" with negligible overhead. Additionally, utilizing 1 as the overlap dimension for all layers is not effective for ViT-Tiny on ImageNet. But using 1 as the overlap dimension is better than using half head dimension as the overlap dimension for all layers for ViT-Small.

Our approach has even more performance enhancement on CaiT models. CaiT-xxs12 is greatly improved by 1.17% with “inc-1 (1)" and the accuracy of CaiT-xxs24 is significantly increased by 3.70% with “inc-0 (1)". For CaiT-xxs24, the accuracy is boosted by more than 3% by using only 1 as the overlap dimension for all layers, which illustrates the effectiveness and efficiency of our proposed method by remarkably boosting the performance of the models with minimum overhead.

For Swin-Tiny, simply using 1 as the overlap dimension is not effective on ImageNet. Even though increasing the overlap dimension to half of the head dimension improves the accuracy of Swin-Tiny, it achieves almost the same result as the paradigms of varying the overlap dimension

Table 6.8: The experimental results on ImageNet

Models	Methods	Accuracy	Params	FLOPs
ViT-Tiny	original	69.25	5.7M	1.3G
	fixed 1	69.11	5.8M	1.3G
	fixed half	**70.31 (+1.06)**	6.2M	1.5G
	inc-0 (1)	69.96	6.0M	1.4G
	inc-1 (1)	69.70	6.1M	1.5G
	dec-0 (1)	69.71	6.0M	1.4G
	dec-1 (1)	69.34	6.1M	1.5G
ViT-Small	original	77.24	22.0M	4.6G
	fixed 1	77.57	22.1M	4.7G
	fixed half	77.40	23.8M	5.3G
	inc-0 (1)	77.87	22.6M	4.9G
	inc-1 (1)	77.91	22.8M	4.9G
	dec-0 (1)	**78.03 (+0.79)**	22.6M	4.9G
	dec-1 (1)	77.99	22.8M	4.9G
CaiT-xxs12	original	67.56	6.6M	1.3G
	fixed 1	67.61	6.6M	1.3G
	fixed half	68.30	7.0M	1.6G
	inc-0 (1)	68.63	6.7M	1.4G
	inc-1 (1)	**68.73 (+1.17)**	6.7M	1.4G
	dec-0 (1)	68.49	6.7M	1.4G
	dec-1 (1)	68.25	6.7M	1.4G
CaiT-xxs24	original	70.56	11.9M	2.5G
	fixed 1	73.64	12.0M	2.6G
	fixed half	73.79	12.8M	3.1G
	inc-0 (1)	**74.26 (+3.70)**	12.4M	2.8G
	inc-1 (1)	74.19	12.4M	2.8G
	dec-0 (1)	74.08	12.4M	2.8G
	dec-1 (1)	74.17	12.4M	2.8G
Swin-Tiny	original	78.52	28.3M	4.5G
	fixed 1	78.44	28.4M	4.5G
	fixed half	78.70	30.4M	5.0G
	inc-0 (1)	78.70	29.4M	4.7G
	inc-1 (1)	**78.72 (+0.20)**	29.6M	4.7G
	dec-0 (1)	78.64	28.6M	4.7G
	dec-1 (1)	78.61	28.8M	4.7G

Table 6.9: The ablation study on applying overlap to Q, K, V

Overlap	ViT-Tiny			CaiT-xxs12		
	Accuracy	Params	FLOPs	Accuracy	Params	FLOPs
None	69.25	5.7M	1.3G	67.56	6.6M	1.3G
Q, K	69.15	5.7M	1.3G	66.85	6.6M	1.3G
V	69.62	6.0M	1.4G	68.53	6.7M	1.3G
Q, K, V	**69.96**	6.0M	1.4G	**68.63**	6.7M	1.4G

by the depths of the layers which have fewer parameters and computations.

Table 6.9 illustrates the ablation study on applying overlap paradigm “inc-0 (1)" to Q, K, or V by using ViT-Tiny and CaiT-xxs12 on ImageNet. From the experimental results shown in Table 6.9, applying our approach to V has more effect on boosting the performance, and applying the overlapping approach to Q, K, and V could yield the best performance.

6.4.5 Analysis

In the experiments, we utilize different variants of our proposed MOHSA on various Vision Transformer models and datasets to illustrate the effectiveness of MOHSA. Overall, the enhancement of the models on various datasets is remarkable with such insignificant overhead. Our proposed approach might be exploited as a plug-and-play method for Vision Transformer models and might enhance the performance of those models with minimum extra cost. For different variants of our proposed method, the results manifest some differences. For a fixed paradigm that the overlap dimension is the same for all layers, the performance could be enhanced by only 1 overlap dimension in most cases and increasing the overlap dimension for fixed mode cannot guarantee better results. Additionally, the variants of varying overlap dimensions based on the depths of the layers demonstrate superior performance than the fixed paradigm on various models and datasets in most cases. Compared to the fixed paradigm with a high overlap dimension, varying overlap dimensions with the depths of the layers could save the number of parameters and computational costs.

To illustrate the accuracy during training, Fig. 6.5 manifests the accuracy of CaiT-xxs24 on the

val or test set with the training proceeding between the original models and our proposed models. “inc-1 (3)", “inc-1 (1)" and “inc-0 (1)" are selected as ours for comparison for CIFAR-100, Tiny-ImageNet, and ImageNet, respectively. The accuracy between the original models and our models demonstrates almost no difference at the early stage of training, while diverges significantly with the training proceeding.

Moreover, the best improvements for various datasets are mostly CaiT models. The possible reason might be that CaiT (Touvron et al., 2021b) utilizes talking-heads attention (Shazeer et al., 2020) before and after the softmax function. Equipped with our MOHSA, CaiT might have more effective information exchange between different heads and illustrate superior performance.

6.5 Conclusion

In this chapter, we have proposed a simple yet effective module MOHSA, to improve the original MHSA in Vision Transformers by overlapping the heads, allowing for information exchange during attention calculation in each head. Extensive evaluations and comparisons with the state-of-the-art on multiple datasets have demonstrated the superior performance of the proposed approach. To the best of our knowledge, this is the first work to propose overlapping heads and achieve significant enhancement across various datasets for different Vision Transformer models. We hope our work will inspire the community to further explore the structure of Vision Transformers.

Chapter 7

Conclusion and Future Works

In this work, multiple object detection and recognition methods are proposed, leveraging both convolutional neural networks and Transformers. (1) A golf ball dataset is created and annotated, and a novel golf ball detection and tracking approach is introduced using the Kalman filter. (2) Generative Adversarial Networks (GANs) are employed to generate images from a different domain, which are then concatenated with images from the original domain to form a 6-channel representation for cross-domain object detection. (3) A dynamic label assignment strategy is proposed for object detection models, combining predictions with anchors to better identify high-quality positive samples, thereby improving detection performance. (4) Additionally, a highly efficient hybrid architecture is introduced by integrating depth-wise convolutions with Vision Transformers. This architecture enables flexible and efficient integration with minimal overhead, enhancing the performance of Vision Transformers on vision tasks. (5) Furthermore, a novel Multi-Overlapped-Head Self-Attention mechanism is proposed to facilitate greater information exchange in multi-head self-attention, leading to improved accuracy in Vision Transformer models.

In the future, the golf ball dataset could be expanded to include more diverse scenarios and use cases. The Kalman filter-based tracking method could also be extended to other sports, such as soccer, for ball tracking. Moreover, since diffusion models (Ho et al., 2020) are capable of generating high-quality images, they may serve as a more effective alternative to GANs for synthesizing cross-domain images. These diffusion models could be leveraged to generate images from different domains to enhance cross-domain object detection. Additionally, the proposed dynamic training mechanism, which incorporates the current training status with model predictions, could potentially be applied to other detection models to further improve training efficiency and

performance.

Transformers are from natural language processing to obtain the long-range relationship between tokens in languages. Since the proposed method could integrate the light-weight depth-wise convolutions via bypassing the entire Transformer blocks without modifying the inner structures of the Transformer model. Could one-dimensional convolutions be applied to the Transformer for natural language processing with the same pattern? Since neighboring word tokens also have some meaningful relationships, will one-dimensional convolutions in large language models increase efficiency when the models are trained? Applying the proposed hybrid architecture to large language models could be a good point for future work to extend the work to language tasks. Moreover, the proposed Multi-Overlapped-Head Self-Attention mechanism demonstrates superior performance for vision recognition. Will the mechanism be applicable to large language models? It is also a good point to extend the proposed mechanism to language tasks in the future.

In object detection and recognition, commonly encountered objects are typically easy for models to recognize. However, training models to perceive complex relationships between objects and to interpret the visual world more holistically remains an active area of research. Recently, Multimodal Large Language Models (MLLMs) have demonstrated the ability to generate fine-grained descriptions of visual data. Despite this progress, these models can sometimes hallucinate, producing inaccurate or nonsensical descriptions. Additionally, their reliance on high-quality data and substantial computational resources limits their broader applicability. Fortunately, tools like ChatGPT (Achiam et al., 2023) can be used to generate high-quality data, significantly reducing the cost of manual labeling. Moreover, as more advanced pretrained models become open source, the computational burden can be further alleviated through parameter-efficient fine-tuning techniques. Nonetheless, there remains a need to explore more accurate data labeling methods and more efficient fine-tuning strategies to enable more applications of these models in diverse real-world scenarios.

Furthermore, open-world object detection and recognition are also a promising direction for future research. Unlike current models, which often suffer from catastrophic forgetting when

learning new information, humans are capable of continuously acquiring new knowledge in an open-world setting without forgetting what they have already learned. Designing mechanisms that enable models to recognize unknown objects and incrementally learn from them—while preserving previously acquired knowledge—remains a key challenge and an active area of research.

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. (2023). Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Alotaibi, A. (2020). Deep generative adversarial networks for image-to-image translation: A review. *Symmetry*, 12(10), 1705.

Arruda, V. F., Paixão, T. M., Berriel, R. F., De Souza, A. F., Badue, C., Sebe, N., & Oliveira-Santos, T. (2019). Cross-domain car detection using unsupervised image-to-image translation: From day to night. In *2019 International Joint Conference on Neural Networks (IJCNN)* (pp. 1–8).: IEEE.

Ba, J. L., Kiros, J. R., & Hinton, G. E. (2016). Layer normalization. *arXiv preprint arXiv:1607.06450*.

Bharati, S. P., Wu, Y., Sui, Y., Padgett, C., & Wang, G. (2018). Real-time obstacle detection and tracking for sense-and-avoid mechanism in uavs. *IEEE Transactions on Intelligent Vehicles*, 3(2), 185–197.

Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*.

Cai, Z., Fan, Q., Feris, R. S., & Vasconcelos, N. (2016). A unified multi-scale deep convolutional neural network for fast object detection. In *European conference on computer vision* (pp. 354–370).: Springer.

Cai, Z. & Vasconcelos, N. (2018). Cascade r-cnn: Delving into high quality object detection.

In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 6154–6162).

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). End-to-end object detection with transformers. In *European conference on computer vision* (pp. 213–229).: Springer.

Cen, F., Zhao, X., Li, W., & Wang, G. (2021). Deep feature augmentation for occluded image classification. *Pattern Recognition*, 111, 107737.

Chen, B., Li, P., Li, B., Li, C., Bai, L., Lin, C., Sun, M., Yan, J., & Ouyang, W. (2021a). Psvit: Better vision transformer via token pooling and attention sharing. *arXiv preprint arXiv:2108.03428*.

Chen, Q., Wu, Q., Wang, J., Hu, Q., Hu, T., Ding, E., Cheng, J., & Wang, J. (2022a). Mixformer: Mixing features across windows and dimensions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 5249–5259).

Chen, X., Hu, Q., Li, K., Zhong, C., & Wang, G. (2023). Accumulated trivial attention matters in vision transformers on small datasets. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* (pp. 3984–3992).

Chen, X., Liu, J., Wang, Y., Brand, M., Wang, G., Koike-Akino, T., et al. (2024). Superlora: Parameter-efficient unified adaptation of multi-layer attention modules. *arXiv preprint arXiv:2403.11887*.

Chen, X., Qin, Y., Xu, W., Bur, A. M., Zhong, C., & Wang, G. (2022b). Improving vision transformers on small datasets by increasing input information density in frequency domain. In *IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, volume 2.

Chen, Y., Dai, X., Chen, D., Liu, M., Dong, X., Yuan, L., & Liu, Z. (2022c). Mobile-former: Bridging mobilenet and transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 5270–5279).

Chen, Y., Li, W., Sakaridis, C., Dai, D., & Van Gool, L. (2018). Domain adaptive faster r-cnn for object detection in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3339–3348).

Chen, Z., Xie, L., Niu, J., Liu, X., Wei, L., & Tian, Q. (2021b). Visformer: The vision-friendly transformer. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 589–598).

Chu, X., Tian, Z., Wang, Y., Zhang, B., Ren, H., Wei, X., Xia, H., & Shen, C. (2021a). Twins: Revisiting the design of spatial attention in vision transformers. *Advances in Neural Information Processing Systems*, 34, 9355–9366.

Chu, X., Tian, Z., Zhang, B., Wang, X., Wei, X., Xia, H., & Shen, C. (2021b). Conditional positional encodings for vision transformers. *arXiv preprint arXiv:2102.10882*.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., & Schiele, B. (2016). The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3213–3223).

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., & Le, Q. V. (2018). Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*.

Dai, J., Li, Y., He, K., & Sun, J. (2016). R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems* (pp. 379–387).

Dalal, N. & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1 (pp. 886–893).: Ieee.

Danelljan, M., Shahbaz Khan, F., Felsberg, M., & Van de Weijer, J. (2014). Adaptive color at-

tributes for real-time visual tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1090–1097).

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248–255).: Ieee.

Dewi, C., Chen, R.-C., Liu, Y.-T., Liu, Y.-S., & Jiang, L.-Q. (2020). Taiwan stop sign recognition with customize anchor. In *Proceedings of the 12th International Conference on Computer Modeling and Simulation* (pp. 51–55).

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., & Tian, Q. (2019). Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 6569–6578).

Everingham, M., Eslami, S. A., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2015). The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111(1), 98–136.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2), 303–338.

Fan, Q., You, Q., Han, X., Liu, Y., Tao, Y., Huang, H., He, R., & Yang, H. (2024). Vitar: Vision transformer with any resolution. *arXiv preprint arXiv:2403.18361*.

Fang, J., Xie, L., Wang, X., Zhang, X., Liu, W., & Tian, Q. (2022). Msg-transformer: Exchanging local spatial information by manipulating messenger tokens. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 12063–12072).

Ganin, Y. & Lempitsky, V. (2015). Unsupervised domain adaptation by backpropagation. In *International conference on machine learning* (pp. 1180–1189).: PMLR.

Ge, Z., Wang, J., Huang, X., Liu, S., & Yoshie, O. (2021). Lla: Loss-aware label assignment for dense pedestrian detection. *arXiv preprint arXiv:2101.04307*.

Geiger, A., Lenz, P., Stiller, C., & Urtasun, R. (2013). Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11), 1231–1237.

Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 1440–1448).

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems* (pp. 2672–2680).

Gosavi, D., Cheatham, B., & Sztuba-Solinska, J. (2022). Label-free detection of human coronaviruses in infected cells using enhanced darkfield hyperspectral microscopy (edhm). *Journal of Imaging*, 8(2), 24.

Guo, J., Han, K., Wu, H., Tang, Y., Chen, X., Wang, Y., & Xu, C. (2022). Cmt: Convolutional neural networks meet vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 12175–12185).

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 2961–2969).

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9), 1904–1916.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

He, L., Lu, J., Wang, G., Song, S., & Zhou, J. (2021). Sosd-net: Joint semantic object segmentation and depth estimation from monocular images. *Neurocomputing*, 440, 251–263.

Hemmati, M., Biglari-Abhari, M., & Niar, S. (2022). Adaptive real-time object detection for autonomous driving systems. *Journal of imaging*, 8(4), 106.

Hendrycks, D. & Gimpel, K. (2016). Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Henriques, J. F., Caseiro, R., Martins, P., & Batista, J. (2014). High-speed tracking with kernelized correlation filters. *IEEE transactions on pattern analysis and machine intelligence*, 37(3), 583–596.

Heo, B., Yun, S., Han, D., Chun, S., Choe, J., & Oh, S. J. (2021). Rethinking spatial dimensions of vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 11936–11945).

Ho, J., Jain, A., & Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33, 6840–6851.

Hou, X., Liu, M., Zhang, S., Wei, P., Chen, B., & Lan, X. (2025). Relation detr: Exploring explicit position relation prior for object detection. In *European Conference on Computer Vision* (pp. 89–105).: Springer.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 7132–7141).

Huang, X., Liu, M.-Y., Belongie, S., & Kautz, J. (2018). Multimodal unsupervised image-to-image translation. In *Proceedings of the European Conference on Computer Vision (ECCV)* (pp. 172–189).

Huang, Z., Ben, Y., Luo, G., Cheng, P., Yu, G., & Fu, B. (2021). Shuffle transformer: Rethinking spatial shuffle for vision transformer. *arXiv preprint arXiv:2106.03650*.

Ibtehaz, N., Yan, N., Mortazavi, M., & Kihara, D. (2024). Acc-vit: Atrous convolution's comeback in vision transformers. *arXiv preprint arXiv:2403.04200*.

Inoue, N., Furuta, R., Yamasaki, T., & Aizawa, K. (2018). Cross-domain weakly-supervised object detection through progressive domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5001–5009).

Ioffe, S. & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning* (pp. 448–456).: pmlr.

Isola, P., Zhu, J.-Y., Zhou, T., & Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1125–1134).

Jamali, A., Roy, S. K., & Ghamisi, P. (2023). Wetmapformer: A unified deep cnn and vision transformer for complex wetland mapping. *International Journal of Applied Earth Observation and Geoinformation*, 120, 103333.

Johnson-Roberson, M., Barto, C., Mehta, R., Sridhar, S. N., Rosaen, K., & Vasudevan, R. (2016). Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? *arXiv preprint arXiv:1610.01983*.

Ke, W., Zhang, T., Huang, Z., Ye, Q., Liu, J., & Huang, D. (2020). Multiple anchor learning for

visual object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 10206–10215).

Kim, J.-S. & Kim, M.-G. (2011). Automatic high-speed flying ball detection from multi-exposure images under varying light conditions. In *Proceedings of the 10th International Conference on Virtual Reality Continuum and Its Applications in Industry* (pp. 549–552).

Kim, K. & Lee, H. S. (2020). Probabilistic anchor assignment with iou prediction for object detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXV 16* (pp. 355–371).: Springer.

Kim, T., Cha, M., Kim, H., Lee, J. K., & Kim, J. (2017). Learning to discover cross-domain relations with generative adversarial networks. *arXiv preprint arXiv:1703.05192*.

Kingma, D. P. & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kingma, D. P. & Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.

Law, H. & Deng, J. (2018). Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 734–750).

Le, Y. & Yang, X. (2015). Tiny imagenet visual recognition challenge. *CS 231N*, 7(7), 3.

Lee, H.-Y., Tseng, H.-Y., Huang, J.-B., Singh, M., & Yang, M.-H. (2018). Diverse image-to-image translation via disentangled representations. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 35–51).

Li, B., Wu, W., Wang, Q., Zhang, F., Xing, J., & Yan, J. (2019a). Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4282–4291).

Li, K., Fathan, M. I., Patel, K., Zhang, T., Zhong, C., Bansal, A., Rastogi, A., Wang, J. S., & Wang, G. (2021a). Colonoscopy polyp detection and classification: Dataset creation and comparative evaluations. *arXiv preprint arXiv:2104.10824*.

Li, K., Ma, W., Sajid, U., Wu, Y., & Wang, G. (2020a). Object detection with convolutional neural networks. In *Deep Learning in Computer Vision* (pp. 41–62). CRC Press.

Li, X., Wang, W., Hu, X., Li, J., Tang, J., & Yang, J. (2021b). Generalized focal loss v2: Learning reliable localization quality estimation for dense object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 11632–11641).

Li, X., Wang, W., Wu, L., Chen, S., Hu, X., Li, J., Tang, J., & Yang, J. (2020b). Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. *arXiv preprint arXiv:2006.04388*.

Li, Y., Chen, Y., Wang, N., & Zhang, Z. (2019b). Scale-aware trident networks for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 6054–6063).

Li, Y., Zhang, K., Cao, J., Timofte, R., & Van Gool, L. (2021c). Localvit: Bringing locality to vision transformers. *arXiv preprint arXiv:2104.05707*.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017a). Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2117–2125).

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017b). Focal loss for dense object

detection. In *Proceedings of the IEEE international conference on computer vision* (pp. 2980–2988).

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision* (pp. 740–755).: Springer.

Liu, F., Kong, Y., Zhang, L., Feng, G., & Yin, B. (2023a). Local-global coordination with transformers for referring image segmentation. *Neurocomputing*, 522, 39–52.

Liu, M.-Y., Breuel, T., & Kautz, J. (2017). Unsupervised image-to-image translation networks. In *Advances in neural information processing systems* (pp. 700–708).

Liu, S., Qi, L., Qin, H., Shi, J., & Jia, J. (2018). Path aggregation network for instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 8759–8768).

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). Ssd: Single shot multibox detector. In *European conference on computer vision* (pp. 21–37).: Springer.

Liu, Y., Xiong, Z., Yuan, Y., & Wang, Q. (2023b). Transcending pixels: boosting saliency detection via scene understanding from aerial imagery. *IEEE Transactions on Geoscience and Remote Sensing*.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., & Guo, B. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 10012–10022).

Lu, C., Xia, S., Huang, W., Shao, M., & Fu, Y. (2017). Circle detection by arc-support line segments. In *2017 IEEE International Conference on Image Processing (ICIP)* (pp. 76–80).: IEEE.

Lu, Z., Xie, H., Liu, C., & Zhang, Y. (2022). Bridging the gap between vision transformers and convolutional neural networks on small datasets. *Advances in Neural Information Processing Systems*, 35, 14663–14677.

Lyu, C., Liu, Y., Li, B., & Chen, H. (2015). Multi-feature based high-speed ball shape target tracking. In *2015 IEEE international conference on information and automation* (pp. 67–72).: IEEE.

Ma, W., Li, K., & Wang, G. (2020a). Location-aware box reasoning for anchor-based single-shot object detection. *IEEE Access*, 8, 129300–129309.

Ma, W., Tu, X., Luo, B., & Wang, G. (2022). Semantic clustering based deduction learning for image recognition and classification. *Pattern Recognition*, 124, 108440.

Ma, W., Wu, Y., Cen, F., & Wang, G. (2020b). Mdfn: Multi-scale deep feature learning network for object detection. *Pattern Recognition*, 100, 107149.

Ma, W., Zhang, T., & Wang, G. (2021). Miti-detr: Object detection based on transformers with mitigatory self-attention convergence. *arXiv preprint arXiv:2112.13310*.

Ma, Y., Fei, Z., & Huang, J. (2023). Dit: Efficient vision transformers with dynamic token routing. *arXiv preprint arXiv:2308.03409*.

Mehta, S. & Rastegari, M. (2021). Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. *arXiv preprint arXiv:2110.02178*.

Mehta, S. & Rastegari, M. (2022). Separable self-attention for mobile vision transformers. *arXiv preprint arXiv:2206.02680*.

Mo, X., Tao, K., Wang, Q., & Wang, G. (2018). An efficient approach for polyps detection in endoscopic videos based on faster r-cnn. In *2018 24th International Conference on Pattern Recognition (ICPR)* (pp. 3929–3934).: IEEE.

Nair, V. & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)* (pp. 807–814).

Nguyen, C. C., Tran, G. S., Burie, J.-C., Nghiem, T. P., et al. (2021). Pulmonary nodule detection based on faster r-cnn with adaptive anchor box. *Ieee Access*, 9, 154740–154751.

Nguyen, D.-K., Assran, M., Jain, U., Oswald, M. R., Snoek, C. G., & Chen, X. (2024). An image is worth more than 16x16 patches: Exploring transformers on individual pixels. *arXiv preprint arXiv:2406.09415*.

Nie, X., Jin, H., Yan, Y., Chen, X., Zhu, Z., & Qi, D. (2024). Scopevit: Scale-aware vision transformer. *Pattern Recognition*, 153, 110470.

Pan, J., Bulat, A., Tan, F., Zhu, X., Dudziak, L., Li, H., Tzimiropoulos, G., & Martinez, B. (2022). Edgevits: Competing light-weight cnns on mobile devices with vision transformers. In *European Conference on Computer Vision* (pp. 294–311).: Springer.

Patel, K., Bur, A. M., Li, F., & Wang, G. (2022). Aggregating global features into local vision transformer. *arXiv preprint arXiv:2201.12903*.

Patel, K., Li, F., & Wang, G. (2024). Multi-layer dense attention decoder for polyp segmentation. In *Proceedings of the 2024 14th International Conference on Biomedical Engineering and Technology* (pp. 115–120).

Patel, K. & Wang, G. (2022). A discriminative channel diversification network for image classification. *Pattern Recognition Letters*, 153, 176–182.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779–788).

Redmon, J. & Farhadi, A. (2017). Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 7263–7271).

Redmon, J. & Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 91–99.

Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6), 1137–1149.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3), 211–252.

Sajid, U., Chen, X., Sajid, H., Kim, T., & Wang, G. (2021). Audio-visual transformer based crowd counting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 2249–2259).

Sakaridis, C., Dai, D., & Van Gool, L. (2018). Semantic foggy scene understanding with synthetic data. *International Journal of Computer Vision*, 126(9), 973–992.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision* (pp. 618–626).

Shazeer, N., Lan, Z., Cheng, Y., Ding, N., & Hou, L. (2020). Talking-heads attention. *arXiv preprint arXiv:2003.02436*.

Simonyan, K. & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Srinivas, A., Lin, T.-Y., Parmar, N., Shlens, J., Abbeel, P., & Vaswani, A. (2021). Bottleneck transformers for visual recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 16519–16529).

Steiner, A., Kolesnikov, A., Zhai, X., Wightman, R., Uszkoreit, J., & Beyer, L. (2021). How to train your vit? data, augmentation, and regularization in vision transformers. *arXiv preprint arXiv:2106.10270*.

Su, K., Cao, L., Zhao, B., Li, N., Wu, D., Han, X., & Liu, Y. (2024). Dctvit: Discrete cosine transform meet vision transformers. *Neural Networks*, 172, 106139.

Sui, Y., Wang, G., Tang, Y., & Zhang, L. (2016). Tracking completion. In *European Conference on Computer Vision* (pp. 194–209).: Springer.

Sui, Y., Wang, G., & Zhang, L. (2018). Joint correlation filtering for visual tracking. *IEEE Transactions on Circuits and Systems for Video Technology*.

Sui, Y., Zhang, S., & Zhang, L. (2015). Robust visual tracking via sparsity-induced subspace learning. *IEEE Transactions on Image Processing*, 24(12), 4686–4700.

Sui, Y., Zhang, Z., Wang, G., Tang, Y., & Zhang, L. (2019). Exploiting the anisotropy of correlation filter learning for visual tracking. *International Journal of Computer Vision*, 127(8), 1084–1105.

Tian, Z., Shen, C., Chen, H., & He, T. (2019). Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 9627–9636).

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., & Jégou, H. (2021a). Training data-efficient image transformers & distillation through attention. In *International conference on machine learning* (pp. 10347–10357).: PMLR.

Touvron, H., Cord, M., Sablayrolles, A., Synnaeve, G., & Jégou, H. (2021b). Going deeper with image transformers. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 32–42).

Uijlings, J. R., Van De Sande, K. E., Gevers, T., & Smeulders, A. W. (2013). Selective search for object recognition. *International journal of computer vision*, 104(2), 154–171.

Umek, A., Zhang, Y., Tomažič, S., & Kos, A. (2017). Suitability of strain gage sensors for integration into smart sport equipment: A golf club example. *Sensors*, 17(4), 916.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Vo, X.-T., Nguyen, D.-L., Priadana, A., & Jo, K.-H. (2024). Efficient vision transformers with partial attention. In *European Conference on Computer Vision* (pp. 298–317).: Springer.

Wadekar, S. N. & Chaurasia, A. (2022). Mobilevitv3: Mobile-friendly vision transformer with simple and effective fusion of local, global and input features. *arXiv preprint arXiv:2209.15159*.

Wang, C.-Y., Mark Liao, H.-Y., Wu, Y.-H., Chen, P.-Y., Hsieh, J.-W., & Yeh, I.-H. (2020). Cspnet: A new backbone that can enhance learning capability of cnn. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (pp. 390–391).

Wang, Q., Liu, Y., Xiong, Z., & Yuan, Y. (2022). Hybrid feature aligned network for salient object detection in optical remote sensing imagery. *IEEE transactions on geoscience and remote sensing*, 60, 1–15.

Wang, S., Xu, Y., Zheng, Y., Zhu, M., Yao, H., & Xiao, Z. (2018). Tracking a golf ball with high-speed stereo vision system. *IEEE Transactions on Instrumentation and Measurement*, 68(8), 2742–2754.

Wang, W., Xie, E., Li, X., Fan, D.-P., Song, K., Liang, D., Lu, T., Luo, P., & Shao, L. (2021). Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 568–578).

Wang, X., Cai, Z., Gao, D., & Vasconcelos, N. (2019). Towards universal object detection by domain attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 7289–7298).

Wang, Y., Zhu, H., & Wang, G. (2023). Pst-net: Point cloud completion network based on local geometric feature reuse and neighboring recovery with taylor approximation. In *2023 International Joint Conference on Neural Networks (IJCNN)* (pp. 1–8).: IEEE.

Wei, Z., Pan, H., Li, L., Lu, M., Niu, X., Dong, P., & Li, D. (2023). Dmformer: Closing the gap between cnn and vision transformers. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 1–5).: IEEE.

Welch, G., Bishop, G., et al. (1995). An introduction to the kalman filter.

Wightman, R., Touvron, H., & Jégou, H. (2021). Resnet strikes back: An improved training procedure in timm. *arXiv preprint arXiv:2110.00476*.

Woodward, A. & Delmas, P. (2005). Computer vision for low cost 3-d golf ball and club tracking. In *Proc. Image and Vision Computing New Zealand* (pp. 11–15).

Wu, H., Xiao, B., Codella, N., Liu, M., Dai, X., Yuan, L., & Zhang, L. (2021). Cvt: Introducing convolutions to vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 22–31).

Wu, Y., Lim, J., & Yang, M.-H. (2013). Online object tracking: A benchmark. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2411–2418).

Wu, Y., Sui, Y., & Wang, G. (2017). Vision-based real-time aerial object localization and tracking for uav sensing system. *IEEE Access*, 5, 23969–23978.

Wu, Y., Zhang, Z., & Wang, G. (2019). Unsupervised deep feature transfer for low resolution image classification. In *Proceedings of the IEEE International Conference on Computer Vision Workshops* (pp. 0–0).

Xiao, T., Singh, M., Mintun, E., Darrell, T., Dollár, P., & Girshick, R. (2021). Early convolutions help transformers see better. *Advances in neural information processing systems*, 34, 30392–30400.

Xu, W., Keshmiri, S., & Wang, G. (2019a). Stacked wasserstein autoencoder. *Neurocomputing*, 363, 195–204.

Xu, W., Shawn, K., & Wang, G. (2019b). Toward learning a unified many-to-many mapping for diverse image translation. *Pattern Recognition*, 93, 570–580.

Xu, Y., Zhang, Q., Zhang, J., & Tao, D. (2021). Vitae: Vision transformer advanced by exploring intrinsic inductive bias. *Advances in neural information processing systems*, 34, 28522–28535.

Yang, J., Lu, J., Batra, D., & Parikh, D. (2017). A faster pytorch implementation of faster r-cnn. *https://github.com/jwyang/faster-rcnn.pytorch*.

Yang, Z., Liu, S., Hu, H., Wang, L., & Lin, S. (2019). Reppoints: Point set representation for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 9657–9666).

Yi, Z., Zhang, H., Tan, P., & Gong, M. (2017). Dualgan: Unsupervised dual learning for image-to-image translation. In *Proceedings of the IEEE international conference on computer vision* (pp. 2849–2857).

Yu, F., Xian, W., Chen, Y., Liu, F., Liao, M., Madhavan, V., & Darrell, T. (2018). Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2(5), 6.

Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., & Yoo, Y. (2019). Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 6023–6032).

Zhang, H., Cisse, M., Dauphin, Y. N., & Lopez-Paz, D. (2017). mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.

Zhang, H., Wang, Y., Dayoub, F., & Sunderhauf, N. (2021). Varifocalnet: An iou-aware dense object detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 8514–8523).

Zhang, H., Zhou, X., Lan, X., Li, J., Tian, Z., & Zheng, N. (2019a). A real-time robotic grasping approach with oriented anchor box. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51(5), 3014–3025.

Zhang, Q., Xu, Y., Zhang, J., & Tao, D. (2023). Vitaev2: Vision transformer advanced by exploring inductive bias for image recognition and beyond. *International Journal of Computer Vision*, (pp. 1–22).

Zhang, S., Chi, C., Yao, Y., Lei, Z., & Li, S. Z. (2020a). Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 9759–9768).

Zhang, S., Wen, L., Bian, X., Lei, Z., & Li, S. Z. (2018). Single-shot refinement neural network for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4203–4212).

Zhang, T., Xu, W., Luo, B., & Wang, G. (2025). Depth-wise convolutions in vision transformers for efficient training on small datasets. *Neurocomputing*, 617, 128998.

Zhang, T., Zhang, X., Yang, Y., Wang, Z., & Wang, G. (2020b). Efficient golf ball detection and tracking based on convolutional neural networks and kalman filter. *arXiv preprint arXiv:2012.09393*.

Zhang, X., Wan, F., Liu, C., Ji, R., & Ye, Q. (2019b). Freeanchor: Learning to match anchors for visual object detection. *arXiv preprint arXiv:1909.02466*.

Zhang, X., Zhang, T., Yang, Y., Wang, Z., & Wang, G. (2020c). Real-time golf ball detection and tracking based on convolutional neural networks. In *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)* (pp. 2808–2813).: IEEE.

Zhao, C., Sun, Y., Wang, W., Chen, Q., Ding, E., Yang, Y., & Wang, J. (2024). Ms-detr: Efficient detr training with mixed supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 17027–17036).

Zhong, Z., Zheng, L., Kang, G., Li, S., & Yang, Y. (2020). Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34 (pp. 13001–13008).

Zhou, D., Kang, B., Jin, X., Yang, L., Lian, X., Jiang, Z., Hou, Q., & Feng, J. (2021). Deepvit: Towards deeper vision transformer. *arXiv preprint arXiv:2103.11886*.

Zhou, X., Zhuo, J., & Krahenbuhl, P. (2019). Bottom-up object detection by grouping extreme and center points. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 850–859).

Zhu, B., Wang, J., Jiang, Z., Zong, F., Liu, S., Li, Z., & Sun, J. (2020a). Autoassign: Differentiable label assignment for dense object detection. *arXiv preprint arXiv:2007.03496*.

Zhu, C., Chen, F., Shen, Z., & Savvides, M. (2020b). Soft anchor-point object detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16* (pp. 91–107).: Springer.

Zhu, C., He, Y., & Savvides, M. (2019). Feature selective anchor-free module for single-shot object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 840–849).

Zhu, J.-Y., Park, T., Isola, P., & Efros, A. A. (2017a). Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision* (pp. 2223–2232).

Zhu, J.-Y., Park, T., & Wang, T. (2020c). Cyclegan and pix2pix in pytorch. *https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix*.

Zhu, J.-Y., Zhang, R., Pathak, D., Darrell, T., Efros, A. A., Wang, O., & Shechtman, E. (2017b). Toward multimodal image-to-image translation. In *Advances in neural information processing systems* (pp. 465–476).

Zhu, L., Wang, X., Ke, Z., Zhang, W., & Lau, R. W. (2023). Biformer: Vision transformer with bi-level routing attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 10323–10333).