IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 24 February 2023, accepted 2 May 2023, date of publication 8 May 2023, date of current version 12 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3273736

TOPICAL REVIEW

Object Detection and X-Ray Security Imaging: A Survey

JIAJIE WU, XIANGHUA XU, AND JUNYAN YANG
Department of Computer Science, Hangzhou Dianzi University, Qiantang, Hangzhou, Zhejiang 310018, China

Corresponding author: Jiajie Wu (wujiajie@hdu.edu.cn)

This work was supported by "Pioneer" and "Leading Goose" Research and Development Program of Zhejiang, China, under Grant 2022C03132.

ABSTRACT Security is paramount in public places such as subways, airports, and train stations, where security screeners use X-ray imaging technology to check passengers' luggage for potential threats. To streamline this process and make it more efficient, researchers have turned to object detection techniques with the help of deep learning. While some progress has been made, there are few comprehensive literature reviews. This paper provides a comprehensive overview of the standard object detection algorithms and principles in X-ray dangerous goods detection. The article begins by classifying and describing the more popular deep learning object detection techniques in detail and presenting the commonly used publicly available datasets and metrics. And then go on to summarize previous applications of deep learning techniques in X-ray dangerous goods detection, highlighting their successes and limitations. Finally, based on an analysis of the experimental results, it summarizes some of the limitations of deep learning in X-ray baggage detection thus far. It offers insights into the future of this exciting field. With this review, we hope to provide valuable insights and guidance for those seeking to improve public safety through X-ray imaging and deep learning technology.

INDEX TERMS Object detection, deep learning, x-ray image detection, baggage security, yolo.

I. INTRODUCTION

X-ray baggage security screening is essential to maintaining public safety in airports, train stations, and subways. In the past, this process was primarily performed by hand, relying on the experience and knowledge of security staff. However, this manual approach was prone to human error and could be impacted by factors such as fatigue and emotional turmoil [1]. A fully automated approach to X-ray baggage security screening is required to overcome these limitations. In this regard, deep learning object detection techniques offer new hope for achieving fully automated detection.

The task of object detection is one of the most fundamental tasks in the field of computer vision. Getting computers to recognize objects in their field of view, especially dangerous materials in luggage backpacks, has been difficult. With the rise of new generation deep learning techniques such as Convolutional Neural Network (CNN) and Transformer [2], [3], which completely replaced the original hand-made feature extractors [4], and have achieved stunning results.

The current mainstream trend is towards hybrid algorithms, as they combine the advantages of CNN and Transformer algorithms, with the former specializing in extracting texture constructs from images and the latter preferring to extract the contour features of the object [5]. Figure 1 lists some of the most representative algorithms in categories.

On the other hand, as a particular type, X-ray images have always been a focus of research in computer vision. However, the imaging principle of X-ray images differs from that of ordinary images in natural light, which leads to poor detection of ordinary algorithmic models on X-ray data sets [76]. An increasing number of researchers are conducting focused research and developing various algorithms to alleviate these problems:

- In terms of research content, it can be divided into two main points: Firstly, from a model perspective,

The associate editor coordinating the review of this manuscript and approving it for publication was Zhongyi Guo.

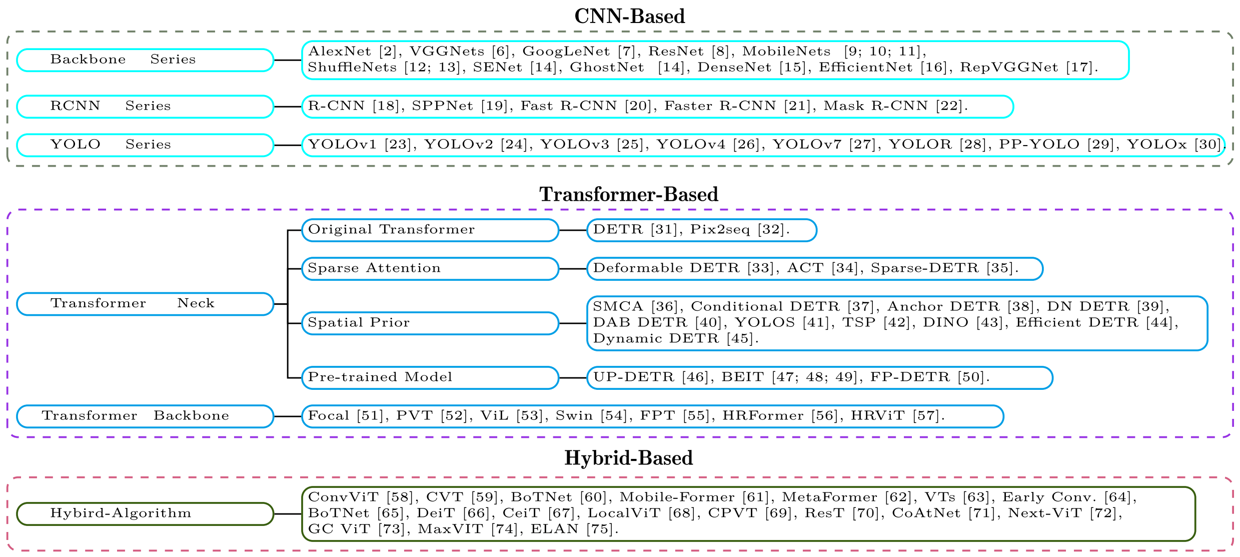

CNN-Based
Backbone Series
AlexNet [2], VGGNets [6], GoogLeNet [7], ResNet [8], MobileNets [9; 10; 11], ShuffleNets [12; 13], SENet [14], GhostNet [14], DenseNet [15], EfficientNet [16], RepVGGNet [17].
RCNN Series
R-CNN [18], SPPNet [19], Fast R-CNN [20], Faster R-CNN [21], Mask R-CNN [22].
YOLO Series
YOLOv1 [23], YOLOv2 [24], YOLOv3 [25], YOLOv4 [26], YOLOv7 [27], YOLOR [28], PP-YOLO [29], YOLOx [30].
Transformer-Based
Transformer Neck
Original Transformer
DETR [31], Pix2seq [32].
Sparse Attention
Deformable DETR [33], ACT [34], Sparse-DETR [35].
Spatial Prior
SMCA [36], Conditional DETR [37], Anchor DETR [38], DN DETR [39], DAB DETR [40], YOLOS [41], TSP [42], DINO [43], Efficient DETR [44], Dynamic DETR [45].
Pre-trained Model
UP-DETR [46], BEIT [47; 48; 49], FP-DETR [50].
Transformer Backbone
Focal [51], PVT [52], ViL [53], Swin [54], FPT [55], HRFormer [56], HRViT [57].
Hybrid-Based
Hybird-Algorithm
ConvViT [58], CVT [59], BoTNet [60], Mobile-Former [61], MetaFormer [62], VTs [63], Early Conv. [64], BoTNet [65], DeiT [66], CeiT [67], LocalViT [68], CPVT [69], ResT [70], CoAtNet [71], Next-ViT [72], GC ViT [73], MaxVIT [74], ELAN [75].

FIGURE 1. Overview of CNN-based, Transformer-based and hybrid algorithms.

it focuses on improving the accuracy of detection tasks under multiple overlapping obstacles or in pseudo-color images. Secondly, from a data perspective, it focuses on using deep learning techniques to synthesize better pseudo-color images similar to natural images and how to quickly and efficiently expand X-ray datasets to improve the final recognition accuracy.

- In terms of research approaches, there are three main types: image classification, object detection, and object segmentation. In practical applications, these three types of algorithms are often combined in order to improve the final recognition accuracy. The primary pieces of literature are shown in figure 2.

We are motivated by the fact that few articles have provided a detailed and comprehensive summary of deep learning algorithms and their applications in X-ray hazardous materials detection. This paper provides a comprehensive review of object detectors based on CNN, Transformers, and hybrid algorithms and a summary of their application to the X-ray image security screening field to fill this gap. The main contributions of the article are:

- An introduction to the popular object detection algorithms so far and an overview of their classification, including CNN-based, Transformer-based, and hybrid algorithms.
- A series of models for applying deep learning algorithms in X-ray baggage hazardous materials detection are described in detail.
- Experiments using different detection algorithms on an open dataset of X-ray baggage and giving meaningful analytical results.
- The article provides an outlook on the application of deep learning algorithms in the field of X-ray image security detection.

The other sections of this paper are organized as follows: Section 2 introduces the basic principles of object detection algorithms, the differences between CNN and Transformer algorithms, and the principles of X-ray imaging; Section 3 details each of the three types of object detection algorithms according to their algorithmic structure, namely, CNN-based, Transformer-based, and hybrid algorithms; Section 4 describes the application of deep learning to X-ray hazardous material detection, including classification, detection, and segmentation. In section 5, four algorithms (YOLOv5 [96], YOLOv7 [27], DINO [43], and Next-ViT [72]) are used to perform dangerous goods detection on the publicly available X-ray baggage dataset and give a meaningful analysis of the results; Section 6 provides a summary and outlook, showing the shortcomings of current deep learning algorithms applied in X-ray security screening and looking into the future.

II. BACKGROUND

A. DEEP LEARNING ARCHITECTURE

Object detection aims to localize and identify the targets in the given image. Locating the number and class of objects can become quite challenging due to the masking, exposure, and perspective of the objects in the image. It is necessary for the computer model to overcome these problems to the best of its ability and to consider timeliness [97]. Three common backbone architectures are listed below.

1) CNN-BASED BACKBONE

Krizhevsky et al. [2] won the 2012 ILSVRC (ImageNet Large-Scale Visual Recognition Challenge) due to its outstanding performance, quickly making CNN the first choice for handling various tasks in the field of computer vision.

Many classical CNN feature extraction networks have emerged, including VGG [6], GoogLeNet [7], ResNets [8],

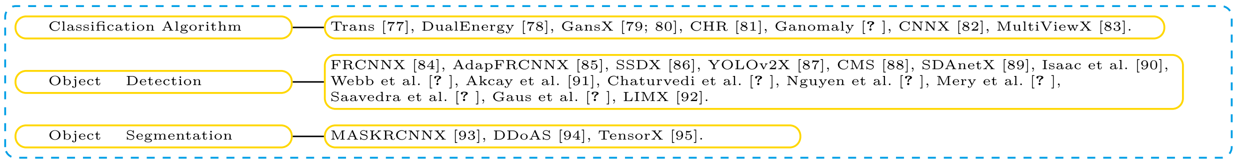

Classification Algorithm
Trans [77], DualEnergy [78], GansX [79; 80], CHR [81], Ganomaly [?], CNNX [82], MultiViewX [83].
Object Detection
FRCNNX [84], AdapFRCNNX [85], SSDX [86], YOLOv2X [87], CMS [88], SSDAnetX [89], Isaac et al. [90], Webb et al. [?], Akcay et al. [91], Chaturvedi et al. [?], Nguyen et al. [?], Mery et al. [?], Saavedra et al. [?], Gaus et al. [?], LIMX [92].
Object Segmentation
MASKRCNNX [93], DDoAS [94], TensorX [95].

FIGURE 2. **Overview of deep learning algorithms in the field of X-ray baggage dangerous goods detection.**

ResNeXt [98], CSPNet [99], EfficientNet [16]. These network structures are shown in figure 3.

Take VGG−16 as an example, and its specific structure is shown in figure 4.[1] The process of extracting features from the convolutional layer is shown in equation 1.

$$x_j^l = \sigma(\sum_{i=1}^{N^{l-1}} x_i^{l-1} \cdot w_{i,j}^l + b_j^l), \tag{1}$$

where x_j^l denotes the j_{th} feature of the l_{th} layer, N denotes the number of features, $w_{i,j}^l$ denotes the convolution kernel of the l_{th} layer, b_j^l denotes the corresponding bias term, and σ denotes the nonlinear function *ReLu*. It has been shown that CNN is not good at processing high-frequency noise in images, so they are more biased in extracting the texture features of images [100].

2) TRANSFORMER-BASED BACKBONE

The transformer was used to solve problems such as machine translation in the NLP domain [101], and its structure is shown in figure 5. After the great success of the Transformer-based model, [3] applied it to the image classification task and proposed the ViT model [3], which structure is shown in figure 6. Later, the ViT model and its variants are applied in various computer vision tasks, including object detection, scene segmentation, and so on [102], [103], [104], [105], [106].

A large part of the reason why transformer is so successful is attributed to the attention mechanism, namely the multi-head self-attentions (MSAs) [107]. Specifically, given the query matrix $\mathbf{Q} \in \mathbb{R}^{N \times D_k}$, the key matrix $\mathbf{K} \in \mathbb{R}^{M \times D_k}$, and the value matrix to be matched $\mathbf{V} \in \mathbb{R}^{M \times D_v}$, where N and M denote the lengths corresponding to $\mathbf{Q}$ and $\mathbf{K}$, and D_k and D_v denote the dimensions corresponding to $\mathbf{K}$ and $\mathbf{V}$. The computation process is as follows:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{QK}^\top}{\sqrt{D_k}}\right)\mathbf{V} = \mathbf{AV}, \tag{2}$$

where the attention matrix $\mathbf{A} = \text{softmax}\left(\frac{\mathbf{QK}^\top}{\sqrt{D_k}}\right)$. The dot product of $\mathbf{Q}$ and $\mathbf{K}$ divided by $\sqrt{D_k}$ can alleviate the gradient vanishing problem of the softmax function.

Besides, the training process of MSAs can be viewed as the process of smoothing the feature mapping space, as shown in figure 7. The subfigure (a) is the Loss-Landscape of ViT before smoothing, and (b) is the Loss-Landscape after smoothing using SAM [108]. The flatter Loss-Landscape, the better model performance and generalization ability. It is also shown in equation 2 that a positive average eigenvalue mapping enhances the performance of MSAs, while a negative value disrupts the optimization of the model. This provides a direction for optimizing ViT. The reason why ViT requires a large amount of data for pre-training to achieve better results is that a large amount of data can help the model suppress negative Hessian eigenvalues in the early stages of training to achieve the effect of smoothing Loss-Landscape and convexity of Loss [5], [109]. [110] shows that MSAs are good at extracting outline information of objects and are not good at processing low-frequency signals.

In vanilla ViT, if the pixels are directly processed using the attention mechanism as in NLP tasks, the computational complexity is a quadratic multiple of the image size, which is unacceptable for most image processing tasks. In addition, ViT with a fixed scale token is not fully applicable to vision tasks because the objects in the images are variable. A lot of improvements have been made to these flaws.

Taking Swin Transformer as an example, the appeal defect is solved by using the hierarchical feature maps obtained by downsampling operation, and the shifted window attention mechanism [54]. From figure 8, we can see that the spatial resolution of the hierarchical feature map in Swin Transformer is the same as that in ResNet, which can easily replace ResNet as the backbone in the network. The use of W-MSA and SW-MSA modules to implement the attention mechanism greatly reduces the computational resources required in the computation process through window exchange.

3) HYBRID BACKBONE

Hybrid frameworks are one of the current research hotspots [59], [60], [61], [72], [74], [111]. It has been shown that CNN will filter the low-frequency part of the image, and MSAs will filter the high-frequency part of the image, which is known that the high-frequency signal corresponds to the outline edge in the image, and the low-frequency part mostly corresponds to the background part [5].

The latest generation of the hybrid framework is to hybridize CNN with MSAs inside the stage [5], [72], not outside the stage, as shown in figure 9.

B. X-RAY IMAGING

The different imaging principles lead to different X-ray and natural light images, as shown in figure 10. The current

[1]Drawing tool from https://github.com/HarisIqbal88/PlotNeuralNet

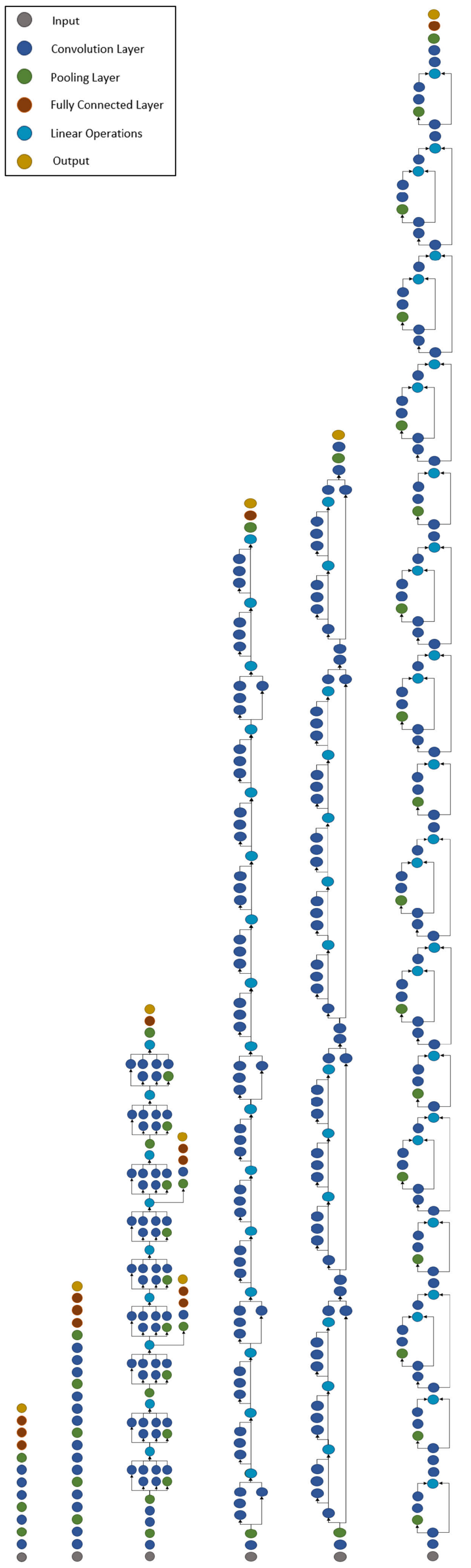


FIGURE 3. CNN-based backbone architecture [97]. From left to right in the figure are AlexNet, VGG−16, GoogLeNet, ResNet−50, CSPResNeXt−50, EfficientNet−B4.

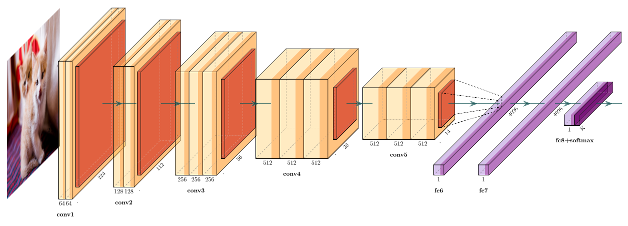


FIGURE 4. This is a diagram of VGG-16 architecture. In the diagram, "conv1" denotes the first convolutional layer, and "fc" denotes the fully connected layer. Specifically, conv1…5=Convolution+ReLu+MaxPooling, fc=FullyConected+ReLu.

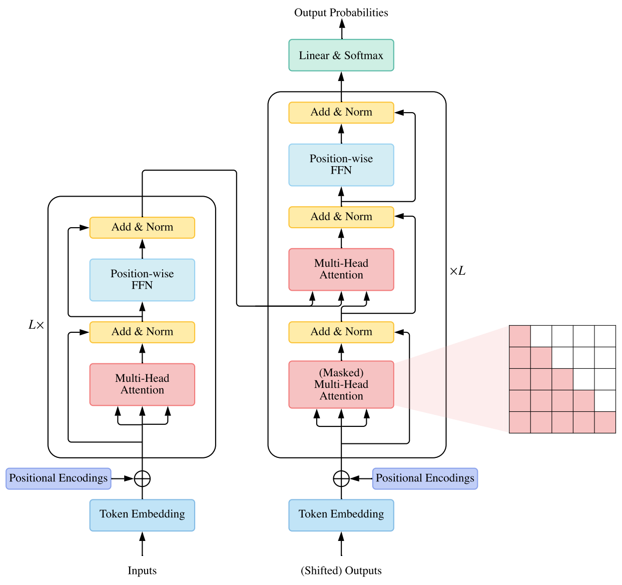


FIGURE 5. The transformer architecture in vanilla ViT [102].

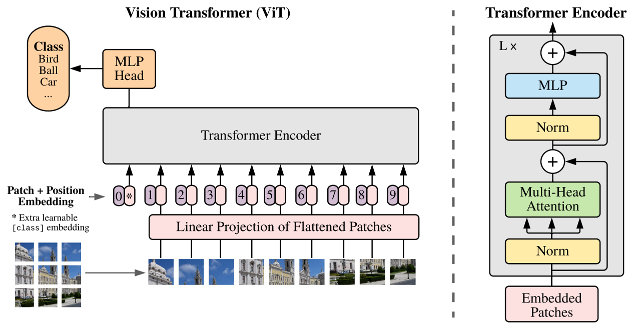


FIGURE 6. Overview of vanilla ViT architecture [3].

X-ray images are available in 2D and 3D. 3D images are usually baggage images scanned by CT (Computed Tomography) machines [112]. Because of their high price, the most common X-ray dangerous material images in the market are mainly 2D images, so the study in this paper mainly focuses on 2D images.

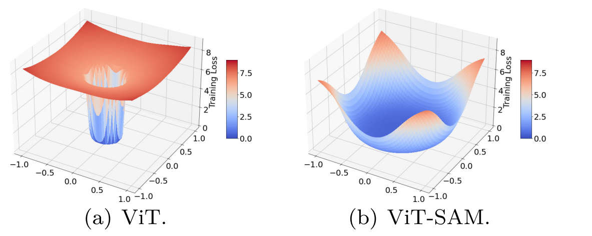


FIGURE 7. (a) is the Loss-Landscape of ViT-B; (b) is the Loss-Landscape after smoothing by SAM [109].

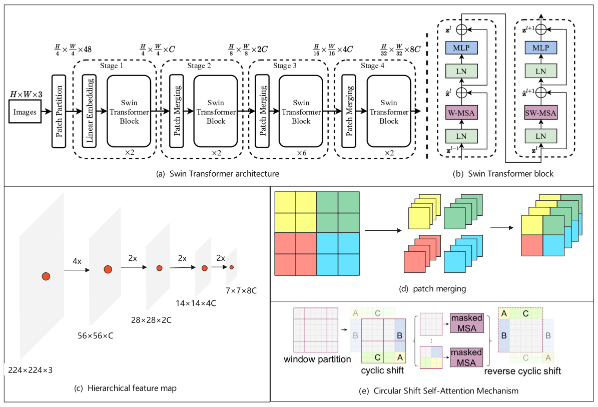


FIGURE 8. In this figure, (a) is the Swin Transformer architecture; (b) is the W-MSA and SW-MSA modules; (c) is the abstracted hierarchical feature graph; (d) represents the patch merging process, which is used to simulate the convolutional kernel operation in CNN; (e) is the cyclic shift self-attentive mechanism, which is used to simulate the Cross-Attention operation [54].

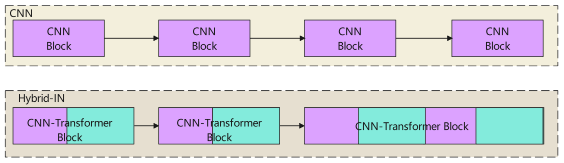


FIGURE 9. A mix of MSAs and Conv in the stage. The above diagram shows the regular CNN block. The diagram below shows the convergence of the Conv and MSAs layers within the stage.

Principle of X-ray imaging The main principle of X-ray imaging is that the X-ray tube produces a beam of light that can penetrate the scanned object. Different objects have different densities, and X-rays will attenuate differently at different material densities. This process can be expressed as equation 3:

$$I_x = I_0 e^{\mu x} \tag{3}$$

where I_x denotes the intensity of the X-ray at x, I_0 is the initialized intensity value, and μ denotes the linear attenuation coefficient based on the material thickness.

Currently, with the development of technology, X-ray machines are equipped with various energies that can produce a wide range of X-ray images for identifying the density and adequate atomic number of objects Z_{eff}. The intensity estimates can be found with Z_{eff} by [113], converting them to

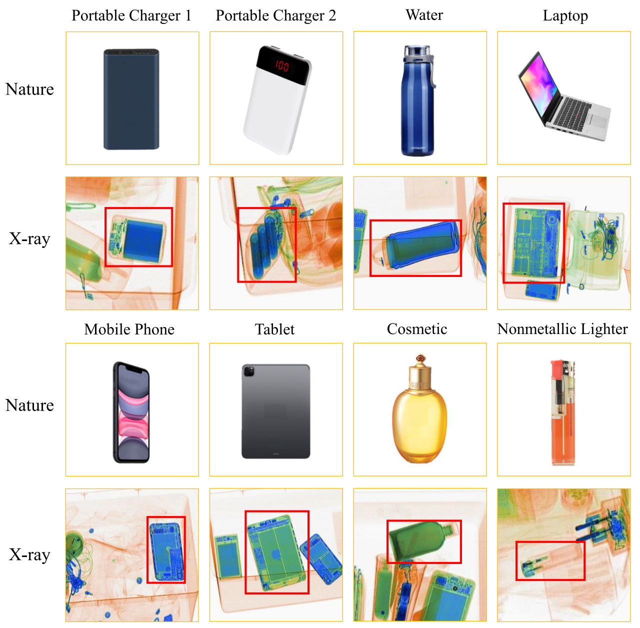


FIGURE 10. Different imaging demonstrations of the same objects under natural light and X-rays [92].

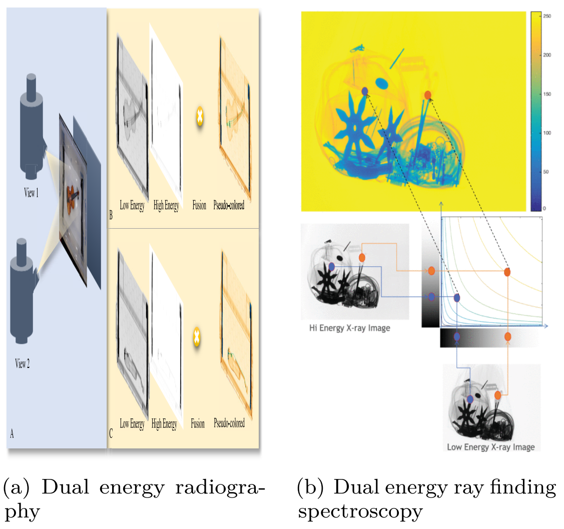


FIGURE 11. X-ray pseudo color map imaging process [116], and [114].

the corresponding pseudo-color images as shown in figure 11. In addition, it is also possible to form X-ray maps from multiple angles [76], [114], [115].

C. OBJECT DETECTION METRICS

The two most common metrics used in object detection tasks are as follows:

1) **GFLOPs**: Giga Floating-point Operations Per Second refers to the number of billion floating-point operations per second, often used to evaluate the performance of a model on GPU.
2) **mAP**: i.e., Mean Average Precision, was first proposed in the VOC competition. Among them, the calculation

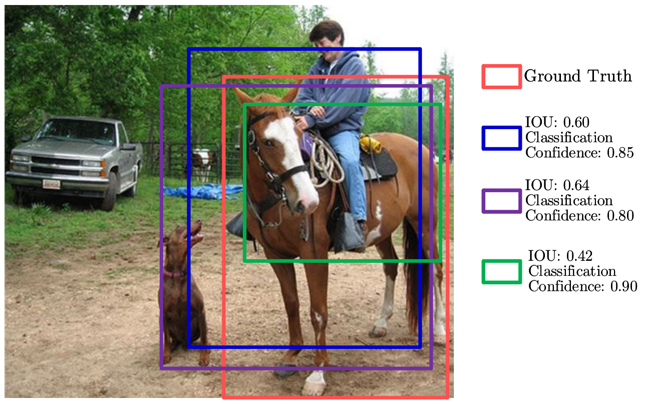


FIGURE 12. IOU example diagram. The ground truth in the figure is the complete one horse. The ratio of the green candidate box to the overlapping part of ground-truth is 0.42 [104].

of precision requires the participation of IOU (Intersection over Union), which is the ratio of the overlap area between the ground truth and the predicted bounding box to the union area, as shown in figure 12

When quantifying the prediction bounding box, a threshold must be set to determine if the detection is correct. If the IoU is greater than the threshold, it is classified as True Positive, while IoU below the threshold is classified as False Positive. If the model fails to detect objects in the ground truth, it is called a False Negative. Precision is used to measure the percentage of correct predictions. In contrast, recall measures the correct predictions relative to ground-truth and is calculated by referring to equation 4, equation 5:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} = \frac{True\ Positive}{All\ Observations} \tag{4}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} = \frac{True\ Positive}{All\ Ground\ Truth} \tag{5}$$

Based on equation 4 and equation 5, the average precision of each category is calculated separately, i.e., with N different recall rates, N precision rates are obtained. Moreover, mAP is the average of the precision of all categories, which can serve as a single metric for the final evaluation [117].

D. DATASETS

Four common datasets for object detection tasks are presented in this section, along with many X-ray baggage detection datasets.

1) OBJECT DETECTION DATASETS

Four standard public datasets for object detection tasks are described below, with detailed parameters in table 1.

a: PASCAL VOC [118]

The PASCAL VOC (Pascal Visual Object Classes) dataset refers to the dataset used in the challenge that started in 2005 and included four object class objects for classification and object detection tasks [117]. In 2007, the VOC07 dataset collected 5K training images and 12K objects with labels [119]. VOC12 expands the dataset to 11K training images, over 27K labeled objects, and 20 classifications and also includes tasks such as segmentation and action detection in 2012.

b: ILSVRC [120], [121]

ILSVRC (ImageNet Large Scale Visual Recognition Challenge) is a challenge that ran from 2010 to 2017. Two hundred of these categories were hand-picked for the object detection task and consisted of more than 500,000 images. Meanwhile, Meanwhile, ImageNet1000 is a subset of ImageNet with 1000 different object categories and a total of 1.2 million images. It provides a standardized benchmark for the ILSVRC image classification challenge.

c: MS-COCO [122]

The MS-COCO (Microsoft Common Objects in Context) dataset is the most commonly used dataset for the object detection task. Eighty target categories are used in the detection task, as shown in figure 13, corresponding to the recognition level of a young human 4-year-old child. It was launched in 2015, and its popularity has only grown since then. It has over 2 million instances with an average of 3.5 categories per image. In addition, it contains 7.7 instances per image, much more than other popular datasets. MS COCO also includes images from different perspectives.

d: Open Image [123]

Open Image is from Google. This dataset contains 9.2 million images, and each image is annotated at the image level with object bounding boxes and segmentation masks. Sixteen million bounding boxes, 600 categories, and an average of 8.3 object categories per image were annotated on 1.9 million images by Open Image for the object detection task.

2) X-RAY BAGGAGE DETECTION DATASETS

Six of these public datasets and one private dataset are described in detail below. The X-ray images in these six public datasets are shown in figure 14. In addition, table 2 summarizes additional X-ray security screening imaging datasets.

a: GDXray [140]

Grima X-Ray Dataset contains 19, 407 images of castings, welds, luggage, natural images, and backgrounds as shown in table 3. Although this dataset contains multi-view luggage detection images, it is not suitable for deploying contemporary large-scale deep learning algorithms on it due to the single scene.

TABLE 1. Detailed parameters for four types of object detection datasets [104].

DataSet	Category	Num. Of Pictures			Num. Of Marked Obj.		Total Num. (Train+Val)		
		Train	Val	Test	Train	Val	Images	Boxes	Boxes/Image
PASCAL VOC									
VOC07	20	2,501	2,510	4,952	6,301(7,844)	6,307(7,818)	5,011	12,608	2.5
VOC08	20	2,111	2,221	4,133	5,082(6,337)	5,281(6,347)	4,332	10,364	2.4
VOC09	20	3,473	3,581	6,650	8,505(9,760)	8,713(9,779)	7,054	17,218	2.3
VOC10	20	4,998	5,105	9,637	11,577(13,339)	11,797(13,352)	10,103	23,374	2.4
VOC11	20	5,717	5,823	10,994	13,609(15,774)	13,841(15,787)	11,540	27,450	2.4
VOC12	20	5,717	5,823	10,991	13,609(15,774)	13,841(15,787)	11,540	27,450	2.4
ILSVRC									
ILSVRC13	200	395,909	20,121	40,152	345,854	55,502	416,030	401,356	1.0
ILSVRC14	200	456,567	20,121	40,152	478,807	55,502	476,668	534,309	1.1
ILSVRC15	200	456,567	20,121	51,294	478,807	55,502	476,668	534,309	1.1
ILSVRC16	200	456,567	20,121	60,000	478,807	55,502	476,668	534,309	1.1
ILSVRC17	200	456,567	20,121	65,500	478,807	55,502	476,668	534,309	1.1
MS-COCO									
MS COCO15	80	82,783	40,504	81,434	604,907	291,875	123,287	896,782	7.3
MS COCO16	80	82,783	40,504	81,434	604,907	291,875	123,287	896,782	7.3
MS COCO17	80	118,287	5,000	40,670	860,001	36,781	123,287	896,782	7.3
MS COCO18	80	118,287	5,000	40,670	860,001	36,781	123,287	896,782	7.3
Open Images									
OICOD18	500	1,643,042	100,000	99,999	11,498,734	696,410	1,743,042	12,195,144	7.0

TABLE 2. Commonly used data sets for X-ray baggage detection of dangerous goods [125].

Datasets	Year	Classes	#Positive	#Negative	AnnotationType	Views	Public	Download
FSOD	2022	20	12,333	0	bbox	1	Y	[126]
EDS	2022	10	14,219	0	bbox	1	Y	[126]
Xray-PI	2022	12	2,409	0	bbox, mask	1	Y	[127]
PIXray	2022	12	5,046	0	bbox, mask	1	Y	[128]
CLCXray	2022	12	9,565	0	bbox	1	Y	[129]
HiXray	2021	8	45,364	0	bbox	1	Y	[130]
deei6	2021	6	7,022	0	bbox, mask	2	N	[131]
PIDray	2021	12	47,677	0	bbox, mask	1	Y	[132]
AB	2021	-	417	6,608	bbox	2	N	[133]
dbf4	2020	4	10,112	0	bbox, mask	4	N	[134]
OPIXray	2020	5	8,885	0	bbox	1	Y	[135]
SIXray	2019	6	8,929	10,500,302	bbox	1	Y	[136]
COMPASS-XP	2019	366	1,928	0	bbox	1	Y	[137]
TSADatasets	2019	4	182	13,586	bbox	-	N	[138]
dbf6	2018	6	11,627	0	bbox, mask	4	N	[91]
GDXray	2015	5	19,407	0	bbox	1	Y	[139]

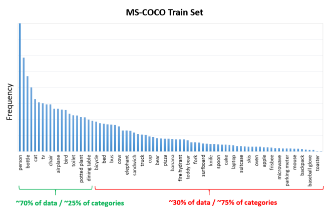
b: SIXray [81]

The dataset contains 1059231 X-ray images, divided into six categories: guns, knives, wrenches, pliers, scissors, and hammers, as shown in table 4. The SIXray dataset restores the actual scene and is a class of datasets with a severe imbalance between positive and negative samples, which is suitable for real-time detection.

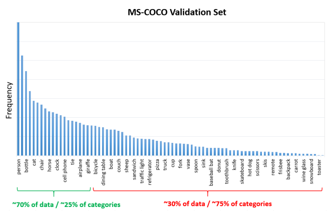
In the experimental part of the article, the authors set up three sub-datasets, SIXray10, SIXray100, and SIXray1000, where the numbers 10 and 100 in the first two sub-datasets represent the ratio of normal to prohibited items. For example, in the SIXray10 dataset, there are 98,219 images, of which 8,929 are prohibited items and 89,290 are normal items. The SIXray1000 sub-dataset is a mixture of 1,000 randomly selected prohibited images and 1,050,302 ordinary images.

c: Compass-XP Dataset [116]

The instances in this dataset are 501 instances drawn from 369 target classes of ImageNet, as shown in table 5. Moreover, it contains 1901 image pairs, and each pair contains



(a) COCO-Train dataset



(b) COCO-Val dataset

FIGURE 13. Data distribution of COCO training dataset and validation dataset [124].

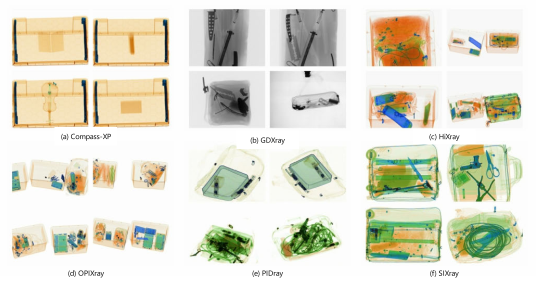


FIGURE 14. The six most common public datasets in X-ray baggage detection.

TABLE 3. GDXray Dataset [140].

Category	Series	ImageNum	Size (MB)
Casting	67	2,727	307.5
WeldSeams	3	88	209.4
Baggage	77	8,150	2,734.8
NaturalImages	13	8,290	191.9
Background	7	152	45.5
Total	167	19,407	3,489.0

X-ray images scanned with a Gilardoni FEP ME 536 with natural images taken with a Sony DSC-W800 digital camera. Each X-ray image package contains different image versions of low energy, high energy, material density, grayscale (a combination of low and high energy), and pseudo-color RGB, which are ideal for studying X-ray imaging principles. However, it is unsuitable for deep learning-based target recognition tasks.

d: OPIXray [141]

This dataset comes from real-time airport security data, manually annotated by professional security officers. It contains 8,885 X-ray images, divided into five categories: folding knives, straight knives, scissors, utility knives, and multi-tool knives, as described in table 6.

TABLE 4. SIXray Dataset [81].

Category	Classes	#Num.
Positive	Gun	3,131
	Knife	1,943
	Wrench	2,199
	Plier	3,961
	Scissor	983
	Hammer	60
Negative	-	1,050,302

e: PIDray [89]

This dataset covers many real-world scenarios for detecting prohibited items, especially intentionally hidden items. The PIDray dataset contains 12 categories with a total of 47,677 X-ray images, and each image comes with a high-quality annotated segmentation mask and bounding box. The test set composes of three classes according to the degree of masking: easy, medium, and challenging, as described in table 7.

f: HiXray [92]

This dataset contains eight categories: lithium-ion prismatic batteries, lithium-ion cylindrical batteries, water, laptops, cell phones, tablets, cosmetics, and non-metallic lighters, for a total of 102,928 contraband items, which is by far the most significant number of contraband items (2022) included in the dataset. The data was collected from accurate airport security checks and manually annotated by professional security screeners, as described in table 8.

g: DB

Durham Baggage (DB) Patch/Full Image. This database is private and not publicly available. The dataset includes pseudo-color 15,449 X-ray images from dual-energy four-view Smiths 6040i machines, including 494 cameras, 1596 ceramic knives, 3208 knives, 3192 guns, 1203 gun parts, 2390 laptops, and 3,366 benign images. The derived databases include DBP2 with DBP6 [142], that do the classification task, and DBF2 with DBF6 [84], [91].

III. OBJECT DETECTION

Detectors early can be classified into two categories according to the detection process: single-stage and two-stage. The latter uses more candidate frames than the former and may contain object suggestions for detection objects. However, as research progressed, introducing more advanced detectors broke the boundaries of single/dual-stage classification. Single/dual-stage classification using detectors alone has become inadequate. Therefore, instead of presenting the classification of detectors according to the single/dual-stage approach as other papers have done, this paper presents the

TABLE 5. COMPASS-XP Dataset [116].

	Category			Instance			ImagePair		
	Positive	Negative	Total	Positive	Negative	Total	Positive	Negative	Total
ImageNet	11	176	187	26	224	250	93	845	938
Custom	24	158	182	43	208	251	165	798	963
Total	35	334	369	69	432	501	258	1,643	1,901

TABLE 6. OPIXray Dataset [141].

Category	Train	Test	Total
FoldingKnife	1,589	404	1,993
StraightKnife	809	235	1,044
Scissor	1,494	369	1,863
UtilityKnife	1,635	343	1,978
Multi-toolKnife	1,612	430	2,042
Total	7,109	1,776	8,885

TABLE 7. PIDray Dataset [89].

Mode	Train	Test		
		Easy	Medium	Challenging
Num	29,457	9,482	3,733	5,005
Total	47,677			

TABLE 8. HiXray Dataset [92].

Category	Train	Test	Total
Li-ionPrismaticBattery	9,919	2,502	12,421
Li-ionCylindricalBattery	6,216	1,572	7,788
Non-metalLighters	706	177	883
Water	2,471	621	3,092
Laptop	8,046	1,996	10,042
Mobile	43,204	10,631	53,835
Tablet	3,921	997	4,918
Makeup	7,969	1,980	9,949
Total	82,452	20,476	102,928

classification according to model architecture categories [97], [104].

Three types of object detectors are presented in this section: CNN-based detector(section III-A), Transformer-based detector(section III-B), and hybrid detector(section III-C). Figure 15 lists the various models that are more mainstream in the field of object detection.

A. CNN-BASED DETECTOR

The two most common series of CNN-based detectors are the R-CNN series and the YOLO series [97], [104]. The former is a two-stage detector with slow speed and long training time but high accuracy. Meanwhile, the latter is a single-stage detector, characterized by relatively low accuracy and poor detection of small objects but faster detection. The most common object detection algorithms in both families are summarised in table 9.

1) RCNN SERIES DETECTORS

R-CNN [18] is the first to use CNN classification pre-training models (such as AlexNet [2]) to extract image features and implement the object detection task with the region candidate box selective search algorithm [155], as shown in figure 16. The R-CNN algorithm is divided into four steps as follows:

1) *Draw candidate regions* A selective search algorithm is used on the input image to select multiple high-quality candidate regions. These regions usually have different shapes and sizes. Each candidate region may contain a certain number of targets.
2) *Using CNN pre-training model to fine-tune* For fine-tuning, in the CNN model, the classification head after pre-training through the ImageNet1K dataset changed to 20 from 1000. After that, the candidate region in the original image, after changing the size, is input to the CNN model and using the pre-trained model to extract the features of the images in the regions.
3) *Training using SVM classifier* We are training the SVM classifier (containing two categories, i.e., positive and negative samples) using the features extracted by CNN to decide the target category in the region. When training the SVM, if the result belongs to the target classification, it is determined as a positive sample. Otherwise, it is a negative sample. It labels N candidate regions (typically 2K) with positive and negative samples using IOUs. Suppose the IOU of the candidate regions and the ground truth are more significant than 0.5. In that case, the sample is considered positive, and its category keeps the same as the ground-truth category. If the IOU is less than 0.3, it is considered a negative sample.
4) *Regression-based training bounding box* It used regression to fine-tune the location of the bounding box. A linear regression model is trained for each class to determine if the bounding box is optimal.

Afterward, a series of improvement algorithms were proposed based on R-CNN, and these improvements include two kinds: speed and network structure.

SPPNet [19] adds spatial pyramid pooling [156] to the top of the last convolutional layer of the CNN for the first time, generating fixed-length features for candidate regions of arbitrary size on the image, which speeds up the R-CNN evaluation.

Fast R-CNN [20] is an improvement of the training process of R-CNN and SPPNet. It achieves unified training

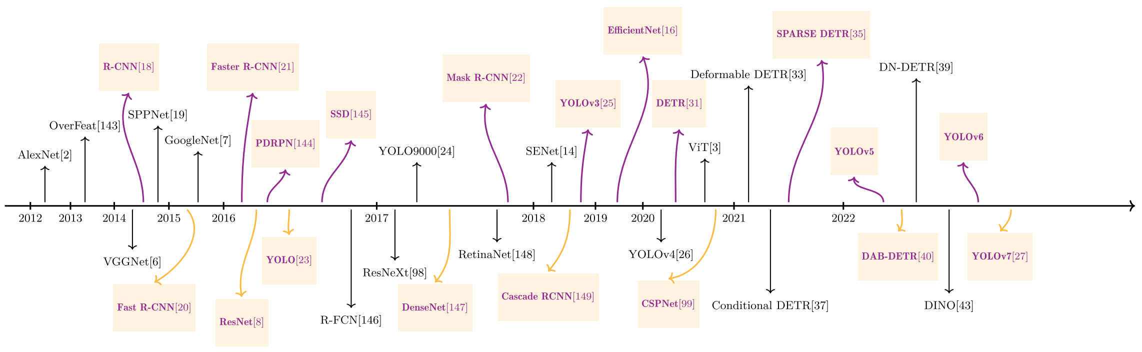


FIGURE 15. Timeline of object detection algorithm development.

TABLE 9. CNN-based detectors on the COCO 2017 val dataset.

Method	Epochs	GFLOPs	#Params(M)	mAP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Fast R-CNN [20]	-	-	-	19.7	35.9	-	-	-	-
Faster R-CNN [21]	36	180	42	40.2	61.0	43.8	24.2	43.5	52.0
Faster R-CNN+ [21]	108	180	42	42.0	62.1	45.5	26.6	45.4	53.4
Faster R-CNN+FPN [150]	-	-	-	36.2	59.1	39.0	18.2	39.0	48.2
Mask R-CNN [22]	36	739	82	46.3	64.3	50.5	-	-	-
Cascade Mask R-CNN [149]	36	260	44	41.0	61.7	44.9	-	-	-
HTC-R50-FPN [151]	-	-	-	43.6	60.0	41.5	20.4	40.7	51.2
YOLOv7 [27]	300	105	36.9	51.2	69.7	55.5	35.2	56.0	66.7
YOLOv7-E6E [27]	-	843	151.7	56.8	74.4	62.1	40.8	62.1	70.6
YOLOv6-L(v2.1) [152]	-	144	58.5	51.0	-	-	-	-	-
YOLOv5-L(r6.1) [96]	-	-	46.5	49.0	67.3	-	-	-	-
YOLOv4 [26]	-	143	64.4	49.7	68.2	54.3	32.9	54.8	63.7
YOLOv4-CSP [153]	-	120	52.9	47.5	66.2	51.7	28.2	51.2	59.8
YOLOR-u5 [28]	-	109	46.5	50.2	68.7	54.6	33.2	55.5	63.7
YOLOv3 [25]	-	157	63	38.5	-	-	-	-	-
YOLOX-M [30]	-	-	-	46.4	65.4	50.6	26.3	51.0	59.9
PPYOLOE-S [154]	-	17	7.9	42.7	60.5	46.6	23.2	46.4	56.9
PPYOLOE-X [154]	-	207	98.4	51.9	69.9	56.5	33.3	56.3	66.4

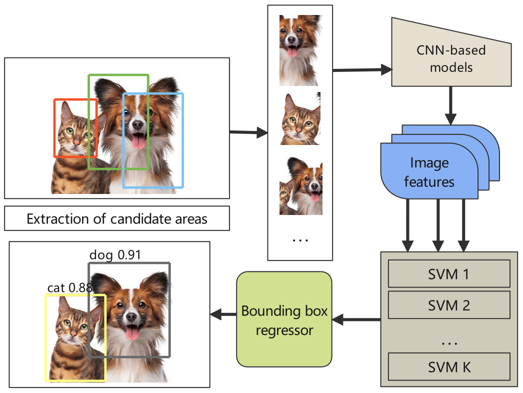


FIGURE 16. Overview of R-CNN architecture.

of softmax classifier, SVM, and bounding box regressor by region candidate box shared convolutional computation and adds a new layer, namely RoI (Region of Interest), to the network. Fast R-CNN significantly improves computational efficiency.

Faster R-CNN [21] uses the region recommendation network RPN (Region Proposal Network) to replace the selective search strategy in the previous network, which can generate a series of candidate boxes for any input image. From figure 17(b), we can see that the Faster R-CNN consists of four major components in general, which are:

- Backbone. It is used to extract features from images;
- Region Proposal Layer. This layer is the core layer of the network and consists of four parts, namely RPN (region proposed network), proposed layer, anchor target layer, and proposed target layer. (i) RPN is used to calculate and generate class prediction scores and boundary box regression coefficients; (ii) The proposed layer uses the anchor box generated by the anchor generator and trims the number of bounding boxes by applying non-maximum suppression (NMS) based on the foreground score. It also generates a converted bounding box

by applying the regression coefficient generated by RPN to the corresponding anchor box; (iii) The goal of the Anchor Target Layer is to select the anchors that can be used to train the RPN network; (ix) The Proposal Target Layer aims to select good RoIs from the list of RoIs output from the proposal layer. These RoIs will perform RoI pooling operations with the feature maps generated from the backbone and pass them to the rest of the RPN for computing the predicted category scores and bounding box regression coefficients;

- RoI Pooling Layer implements the spatial transformation of features, specifically, samples the input feature map gave the coordinates of the proposed bounding box of the region generated by the proposed target layer. These coordinates are usually not on integer boundaries and therefore require interpolation-based sampling;
- Classification Layer is used to obtain the output feature maps generated by the RoI Pooling Layer and perform convolution operations. The final output is realized by two fully connected layers, namely bbox_pred_net, and cls_score_net. The former can generate the class probabilities of each region suggestion, and the latter generates a set of class-specific bounding boxes.

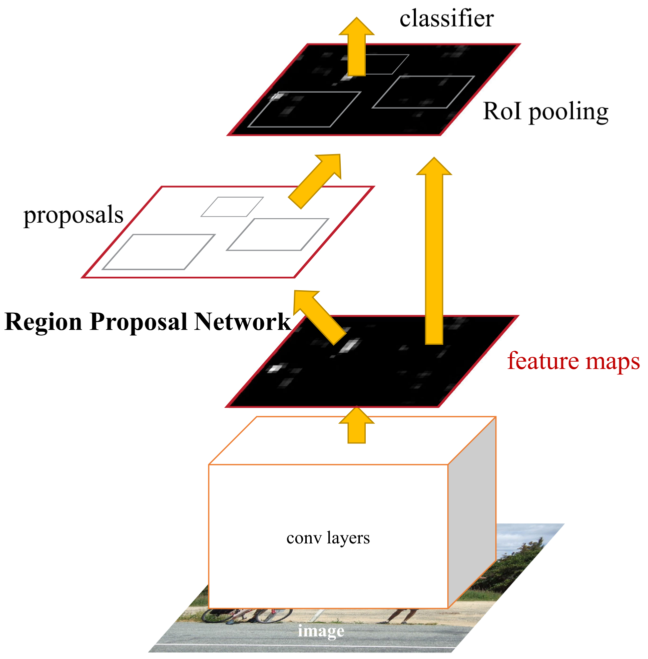
In general, Faster R-CNN is still the most widely used type of detector in the industry today.

R-FCN [146] is further improved the problem of position sensitivity of targets in images in different sub-networks of the Faster R-CNN. In Faster R-CNN, the image is processed in the first step of the backbone and then fed into the sub-network associated with RoI in the second step for processing. In the feature maps obtained after the first step of processing, the RoI is shared and insensitive to the object location. In the second step, the RoI is processed independently, i.e., they are sensitive to the target location.

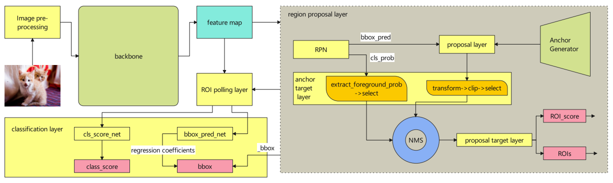
The R-FCN divides the RoI into $k \times k$ regions, which are mapped by a position-sensitive score map to each region, generating the corresponding response values. If all of this response value information is greater than the threshold of a certain category, then this region is judged as this category category. Otherwise, it is judged as the background category.

FPN [150] is a feature processing approach that is often applied in frameworks such as Faster R-CNN. FPN connects top-down side-by-side high-level features with low-resolution and high-semantic information to low-level features with high-resolution and low-semantic information so that features at all scales are rich in semantic information. This allows multi-scale feature representations with strong semantic information to be learned while improving computational efficiency.

Cascade R-CNN [149] is more like a training method. The essence is to cascade multiple R-CNN networks based on different IOU thresholds on the Faster R-CNN, i.e., the output of the previous R-CNN network is used as input to the latter R-CNN network, after which the results of the detection are optimized to keep rising the IOU threshold. Almost all



(a) Architecture [21].



(b) Details.

FIGURE 17. Overview of Faster R-CNN.

detectors based on R-CNN structures can use this cascading approach to improve detection accuracy.

Mask R-CNN [22] extends Faster R-CNN and can handle pixel-level segmentation of target instances. It classifies each pixel into multiple segments, uses Faster R-CNN for object framing, and adds an extra mask in the header. Mask R-CNN uses the previous RoIPool layer using a RoIAlign layer to avoid pixel-level misalignment due to spatial quantization. It is also among the most popular models in the R-CNN family.

HTC [151] is an improvement on the Cascade R-CNN and Mask R-CNN:

- By introducing the Interleaved Execution operation, the algorithm increases the information interaction between different branches within each stage, i.e., the information processed by the box branch is then passed to the mask branch for processing, eliminating the information gap between the training and testing processes.
- The algorithm adds an information flow connection between the mask branches of adjacent stages, allowing the mask branches of different stages to interact with each other.

- The algorithm will process the information for semantic segmentation.

The HTC algorithm is mainly used for semantic segmentation problems but can also be modified for object detection problems, e.g., HTC++ [157].

2) YOLO SERIES DETECTORS

The YOLO series detectors received much attention once they were introduced. YOLO, or You Only Look Once, differs from R-CNN in that YOLO does not select targets by generating candidate frames but performs target prediction and recognition directly at the pixel level.[2]

YOLOv1 [23] grid the images and then use the grid for detection. Each grid is responsible for detecting targets that fall within its region. Each cell can detect multiple bounding boxes, and the bounding box is denoted by $p_{box} = (x, y, w, h, \alpha)$, where (x, y) denotes the center coordinates of the target object, (w, h) denotes the width and height, and the confidence $\alpha = p_{obj} \times IOU$, p_{obj} denotes the probability of having an object fall in the grid with probability. The final prediction $res = (p_{box}, \mathbf{C}_{lasses})$, where $\mathbf{C}$ denotes the number of target classifications in the dataset. The above procedure is shown in figure 18.

YOLOv2 [24] is an improvement on YOLOv1. The YOLO9000 model can detect 9000 target classes. YOLOv2 strives for a balance of speed and accuracy.

YOLOv3 [25] used Darknet-53 as the backbone for feature extraction compared to the previous two versions and achieved SOAT in the same period.

YOLOv4 [26] makes changes to the model structure. A BOF (Bag Of Freebies) strategy is used in the model, which can improve detection accuracy with only an increase in training cost and no impact on inference speed. The BOF strategy in YOLOv4 includes data augmentation, regularization, CmBN (Cross mini-Batch Normalization), CIoU-loss [162], and other techniques. BOS (Bag of specials) strategies, i.e., plug-in modules and post-processing methods that add only a small amount of inference time but can significantly improve object detection accuracy, are also used in YOLOv4. The former enhances certain models' properties, such as expanding the perceptual field, introducing attention mechanisms, or enhancing feature integration capabilities. At the same time, the latter can filter the model prediction results. BOS strategies in YOLOv4 include Mish activation [163], CSP (Cross-stage partial connections) [99], and other techniques. The architecture is shown in figure 19.

YOLOv5,[3]**YOLOv6**[4] only open the source code and no related academic paper. YOLOv5 retains much of the network structure of YOLOv4. YOLOv5 is one of the most used and popular object detection models within the industry, especially for applications in the field of real-time video detection.

[2] [158], [159], [160], [161] integrates the common YOLO series algorithms.

[3]https://github.com/ultralytics/yolov5

[4]https://github.com/meituan/YOLOv6

The backbone design idea of YOLOv6 is mainly derived from RepVGG [17], with the primary purpose of achieving more straightforward deployment and faster inference speed in the industry.

YOLOx [30] believes that YOLOv4 and YOLOv5 may have over-optimized the anchor-based detection method, so YOLOx chose to make improvements to YOLOv3. Previously, YOLOv5 achieved the best performance on the COCO dataset (48.2%AP). The improvements made by YOLOx include: (i)YOLOx changes the original Coupled Head to Decoupled Head, as shown in figure 20(b). Specifically, YOLOx decouples Cls, Reg, and IOUs, allowing the network to learn the categories and the corresponding coordinate regressions better; (ii)Detection is carried out in an anchor-free manner, i.e., by generating an a priori frame (anchor) of different sizes and aspect ratios at each position on a given feature map. The purpose of this is to:

1) Reducing the computational effort of the model, producing fewer prediction frames;
2) Mitigating positive and negative sample imbalances;
3) There is no need to design the parameters of the anchor manually.

YOLOR [28]'s encoder is used to learn both implicit and explicit knowledge representations, using implicit information to perform different tasks, and this technique is also integrated into YOLOv7.

YOLOv7 [27] increases the training cost in order to improve accuracy. By using the reparametrization trick, the inference cost is kept constant. In addition, YOLOv7's backbone is the basis of a cascade structure. Changes in network depth often bring about changes in width when the model is scaled and thus need to consider comprehensively when the model is scaled for evaluation. Several trainable Bag-of-Freebies methods are designed in the paper for solving the above problems, including planned reparametrization module design, dynamic label assignment strategy (coarse for auxiliary, fine for loss), batch normalization in topology, and EMA module usage. YOLOv7 can effectively reduce about 40% of the parameters and 50% of the computation of existing real-time object detectors and has faster inference and higher detection accuracy, with the structure shown in figure 21.

B. TRANSFORMER-BASED DETECTORS

The MSAs mechanism in the transformer allows for better extraction of contour information from the image. The main limitation of the Transformer is its high computational overhead, which is usually a quadratic amount of the input feature size. Common Transformer-based object detection algorithms are outlined in table 10.

DETR [31] is one of the first end-to-end transformer-based object detectors. It treats the object detection problem as an ensemble prediction problem. Unlike traditional object detectors, DETR learns anchors (not by hand) and does not use non-maximum suppression (NMS) post-processing. Instead, position-encoded "object queries" are fed to the

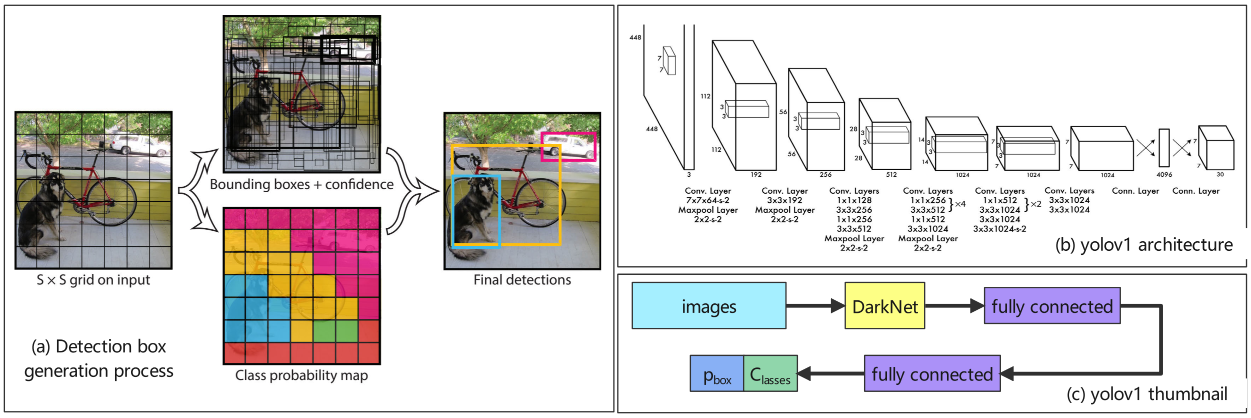

S × S grid on input
Bounding boxes + confidence
Class probability map
Final detections
(a) Detection box generation process
(b) yolov1 architecture
Conv. Layer 7x7x64-s-2 Maxpool Layer 2x2-s-2
Conv. Layer 3x3x192 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x128 3x3x256 1x1x256 3x3x512 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x256 3x3x512 ×4 1x1x512 3x3x1024 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x512 3x3x1024 ×2 3x3x1024 3x3x1024-s-2
Conv. Layers 3x3x1024 3x3x1024
Conn. Layer
Conn. Layer
images
DarkNet
fully connected
fully connected
pbox
Classes
(c) yolov1 thumbnail

FIGURE 18. Overview of YOLOv1 architecture [23].

TABLE 10. Transformer-based detectors on the COCO 2017 val dataset.

Method	Epochs	GFLOPs	#Params(M)	mAP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
DETR-R50 [31]	500	86	41	42.0	62.4	44.2	20.5	45.8	61.1
DETR-DC5 [31]	500	187	41	43.3	63.1	45.9	22.5	47.3	61.1
Deformable DETR [33]	50	78	34	39.7	60.1	42.4	21.2	44.3	56.0
Deformable DETR-DC5-R50-SS [33]	50	128	34	41.5	61.8	44.9	24.1	45.3	56.0
Deformable DETR-Iter [33]	50	173	40	43.8	62.6	47.7	26.4	47.1	58.0
Deformable DETR-Two-Stage [33]	50	173	40	46.2	65.2	50.0	28.8	49.2	61.7
Conditional DETR-R50 [37]	108	90	44	43.0	64.0	45.7	22.7	46.7	61.5
Conditional DETR-DC5-R101 [37]	108	262	63	45.9	66.8	49.5	27.2	60.3	63.3
DAB-DETR-R50 [40]	50	100	44	42.6	63.2	45.6	21.8	46.2	61.1
DAB-DETR-DC5-R101 [40]	50	296	63	46.6	67.0	50.2	28.1	50.5	64.1
DN-DETR-R50 [39]	50	94	44	44.1	64.4	46.7	22.9	48.0	63.4
DN-DETR-DC5-R101 [39]	50	282	63	47.3	67.5	50.8	28.6	51.5	65.0
DINO-4scale [43]	24	-	-	49.9	67.4	54.5	31.8	53.3	64.3
DINO-5scale [43]	36	-	-	51.0	69.0	55.6	34.1	53.6	65.6
Sparse-DETR-0.1 [35]	50	105	41	45.3	65.8	49.3	28.4	48.3	60.1
Sparse-DETR-0.5 [35]	50	136	41	46.3	66.0	50.1	19.0	49.5	60.8
EVA-Cascade R-CNN [164]	-	-	1,074	64.5	82.1	70.8	49.4	68.4	78.5
Swin-L-HTC++ [54]	-	1,470	284	57.1	82.1	70.8	49.4	68.4	78.5
SwinV2-G-HTC++ [157]	-	-	>=3,000	62.5	-	-	-	-	-

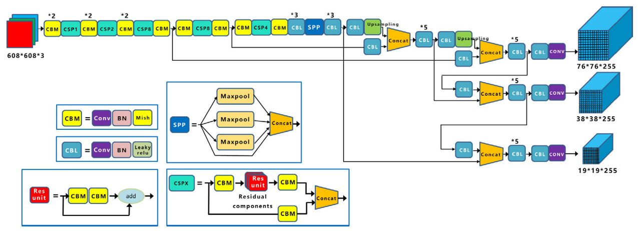
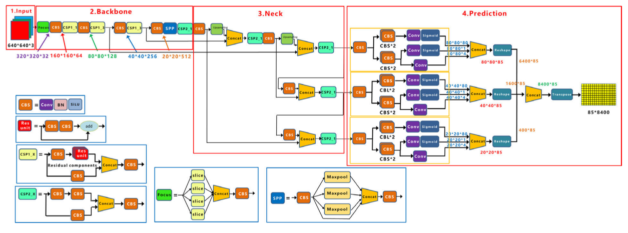


FIGURE 19. Overview of YOLOv4 architecture [26].

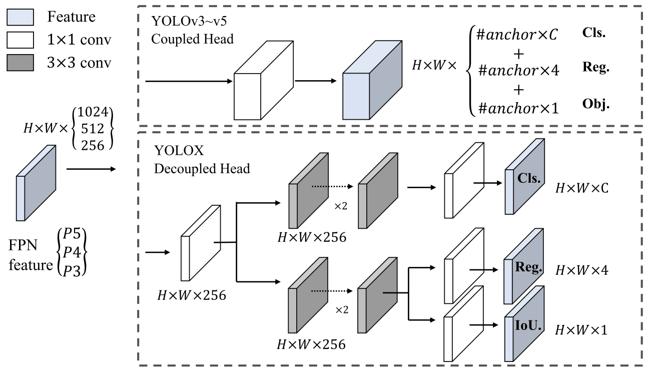
decoder for finding the features of an object in the image and decoding image features. The predictor produces detection results directly from the decoder's output queries, as figure 22 shows.

To avoid the problem of "object queries" in which the object cannot be matched accurately during the query process, DETR adds a particular class, the no object label ($\varnothing$), in addition to matching the regular class labels. In the training process, the Hungarian algorithm, a bivariate graph matching algorithm, is used to perform one-to-one matching of the ground-truth y_i with the predicted target $\hat{y}_{\sigma(i)}$, and the matching pair strategy $\hat{\sigma}$ with the loss function $\mathcal{L}_{\text{match}}$ as:

$$\begin{aligned} l\hat{\sigma} &= \underset{\sigma\in\mathfrak{S}_N}{\operatorname{argmin}} \sum_i^N \mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)}), \\ \mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)}) &= -\mathbb{1}_{\{c_i\neq\varnothing\}}\hat{p}_{\sigma(i)}(c_i) \\ &\quad + \mathbb{1}_{\{c_i\neq\varnothing\}}\mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)}). \end{aligned} \tag{6}$$



(a) YOLOx architecture



(b) Comparison of uncoupled head mechanisms

FIGURE 20. YOLOx architecture and its uncoupled head mechanisms [109].

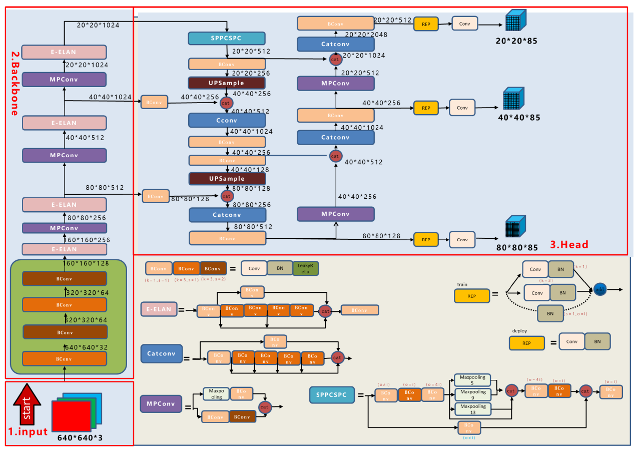


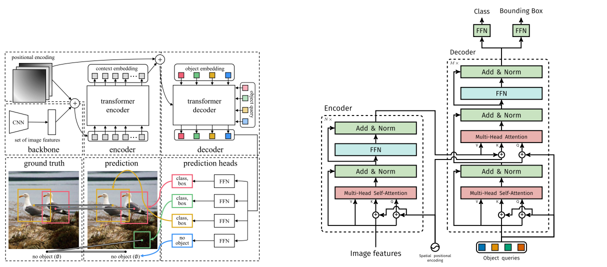
FIGURE 21. Overview of YOLOv7 architecture.

In addition, the Hungarian loss function $\mathcal{L}_{\mathrm{H}}$ includes class label loss and bounding box loss on all matching pairs ($c_i \neq \varnothing$):

$$\mathcal{L}_{\mathrm{H}}(y_i, \hat{y}_{\sigma(i)}) = \sum_{i=1}^{N}[-\log\hat{p}_{\hat{\sigma}(i)}(c_i) + \mathbb{1}_{\{c_i\neq\varnothing\}}\mathcal{L}_{\mathrm{box}}(b_i, \hat{b}_{\sigma(i)})]. \tag{7}$$

where $\mathcal{L}_{\mathrm{box}}$ denotes the bounding box loss and is calculated as:

$$\mathcal{L}_{box}(b_{\sigma(i)}, \hat{b}_i) = \lambda_{iou}\mathcal{L}_{iou}(b_{\sigma(i)}, \hat{b}_i) + \lambda_{L1}||b_{\sigma(i)} - \hat{b}_i||_1, \tag{8}$$



(a) Overview of the DETR framework (b) Detailed diagram of the DETR framework

FIGURE 22. DETR architecture [103], and [31].

where $\lambda_{iou}, \lambda_{L1} \in \mathbb{R}$ are hyperparameters, $\mathcal{L}_{iou}(\cdot)$ is calculated as:

$$\mathcal{L}_{iou}(b_{\sigma(i)}, \hat{b}_i) = 1 - \left(\frac{|b_{\sigma(i)} \cap \hat{b}_i|}{|b_{\sigma(i)} \cup \hat{b}_i|} - \frac{|B(b_{\sigma(i)}, \hat{b}_i) \setminus b_{\sigma(i)} \cup \hat{b}_i|}{|B(b_{\sigma(i)}, \hat{b}_i)|}\right). \tag{9}$$

where |.| denotes the set, the intersection or the union can be calculated by the linear function $b_{\sigma(i)}$ with min / max of $\hat{b}_i$, $B(b_{\sigma(i)}, \hat{b}_i)$ denotes the maximum bounding box containing $b_{\sigma(i)}, \hat{b}_i$ [31].

Although DETR was one of the first object detectors to use a transformer structure, its disadvantages include poor convergence and poor performance on high-resolution images, mainly due to:

1) **Encoder**: the input is an image feature extracted through the backbone (ResNet [8]), and the length and width of this feature are denoted by H and W, respectively. The complexity of the self-attention calculation (i.e., equation 2) grows quadratically with the feature pixel space, i.e., $O(H^2W^2C)$, where C is the feature dimension.
2) **Decoder**: the DETR requires the computation of both cross-attention and self-attention modules. In computing cross-attention, "object queries" are computed and extracted from the feature mapping output by encoder through the attention mechanism (i.e., equation 2), specifically, **Q** which is "object queries" and **K** is the feature mapping of the Encoder output. So the computational complexity of cross-attention grows linearly with the feature pixel space, i.e., $O(HWC^2 + NHWC)$, where N denotes the number of "object queries". When computing self-attention, "object queries" do **QKV** computations on each other, so the computational time complexity is $O(2NC^2 + N^2C)$ [33].

In response to the shortcomings of DETR, many algorithms have focused on improving the internal structure of the encoder and decoder, including the descending calculation of attention, improvements in the structure of "object queries", and the selection of the feature mapping part of the encoder output in the cross-attention calculation.

Deformable DETR [33] is one of the most widely used improved algorithms for DETR. Its most significant

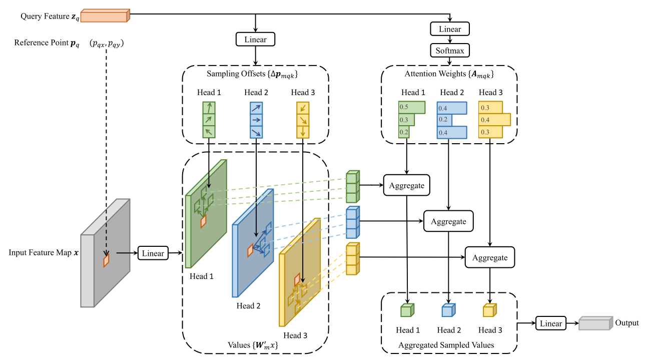

Query Feature z_q
Reference Point p_q (p_{qx}, p_{qy})
Linear
Sampling Offsets $\{\Delta p_{mqk}\}$
Head 1
Head 2
Head 3
Softmax
Attention Weights $\{A_{mqk}\}$
Aggregate
Input Feature Map x
Values $\{W'_m x\}$
Aggregated Sampled Values
Output

FIGURE 23. **The Deformable Attention Module [33].**

contribution is to improve the performance and accuracy of DETR in detecting small objects using a multi-level variable attention mechanism. In the case where 10× is smaller than the original DETR training epochs, the computational complexity is $O(2N_qC^2+\min(HWC^2, N_qKC^2))$, an inference speedup of 1.6 times.

The Deformable Attention Module in Deformable DETR only on a few key sampling points near the reference point rather than the entire feature mapping map. This reduces the dimensionality of **K** in equation 2 and thus reduces the computational complexity, as shown in figure 23.

The calculations of deformable attentional characteristics are given by:

$$\text{DeformAttn}(z_q, \boldsymbol{p}_q, \boldsymbol{x}) = \sum_{m=1}^{M} \boldsymbol{W}_m \big[\sum_{k=1}^{K} A_{mqk} \cdot \boldsymbol{W}'_m \boldsymbol{x}(\boldsymbol{p}_q + \Delta \boldsymbol{p}_{mqk}) \big]. \quad (10)$$

where $\boldsymbol{x} \in \mathbb{R}^{C\times H\times W}$ denotes the feature mapping output by the encoder, $\boldsymbol{p}_q$ denotes the 2D reference point, z_q denotes the q_{th} content feature, A_{mqk} denotes the attention weight of the m^{th} attention head at the k^{th} sampling point, and $\Delta\boldsymbol{p}_{mqk}$ denotes the sampling offset with respect to the reference point. From equation 10, it can be seen that the 2D reference points are involved in the computation as part of the cross-attention query.

According to equation 10, the formula for multi-level variable attentional features can be written as:

$$\text{MSDeformAttn}(z_q, \hat{\boldsymbol{p}}_q, \{\boldsymbol{x}^l\}_{l=1}^{L}) = \sum_{m=1}^{M} \boldsymbol{W}_m \cdot \big[\sum_{l=1}^{L}\sum_{k=1}^{K} A_{mlqk} \cdot \boldsymbol{W}'_m \boldsymbol{x}^l(\phi_l(\hat{\boldsymbol{p}}_q) + \Delta \boldsymbol{p}_{mlqk}) \big]. \quad (11)$$

where $\{\boldsymbol{x}^l\}_{l=1}^{L}$ is the l_{th} level input feature map, $\hat{\boldsymbol{p}}_q \in [0, 1]^2$ denotes the normalized coordinates of the reference point corresponding to each query element q, $\Delta\boldsymbol{p}_{mlqk}$ denotes the sampling offset, and A_{mlqk} denotes the attention weight of the m^{th} attention head under the l^{th} level input feature map with respect to the k^{th} sampling point.

In addition, Deformable DETR provides the so-called "two-stage" selection strategy as a candidate strategy. The $top-K$ feature values output from the last layer of the encode is selected as prior knowledge to strengthen the queries in the decoder.

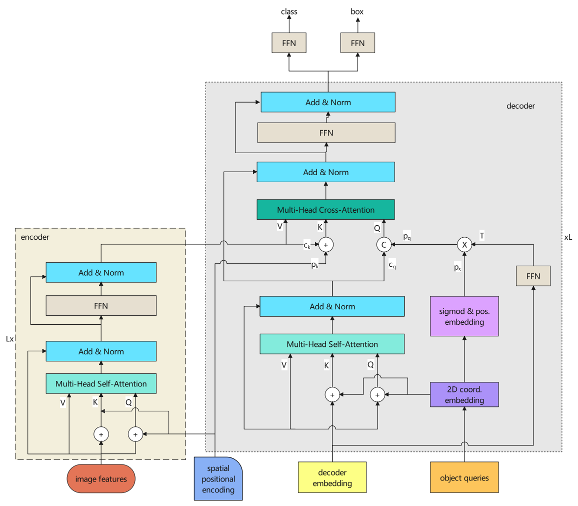


FIGURE 24. **Overview of Conditional DETR architecture.**

Conditional DETR's [37] most significant contribution is the different treatment of content queries in the decoder (the output of self-attention in the decoder) from location space queries, as shown in figure 24. From the figure, the **K** in the decoder cross-attention is generated by the position embedding rvp_k and the encoder output content embedding $\mathbf{c}_k$, respectively. Moreover, **Q** includes the spatial location embedding $\mathbf{p}_q$ in addition to the content embedding $\mathbf{c}_q$ formed by the decoder self-attention.

It is generated by first normalizing the reference point s and mapping it into a 256-dimensional sinusoidal position embedding $\mathbf{p}_s$ (keeping the same generation as $\mathbf{p}_k$), after which the $\mathbf{p}_s$ formed by the reference point s is transformed in the embedding space using $\boldsymbol{T}$ to obtain the spatial position embedding $\mathbf{p}_q$ in **Q**, i.e., $\mathbf{p}_q = \boldsymbol{T} \cdot \mathbf{p}_s$. Since the decoder's embedding includes the position information, it can be used by $\boldsymbol{T} = FFN(decoderembedding)$.

DAB-DETR [40] is based on Conditional DETR, combining reference points with "object queries" to form a 4D learnable probe frame, (x, y, w, h), where (x, y) is the center coordinate point of the probe frame, and (w, h) is the width and height of the probe frame, as shown in figure 25.

DAB-DETR updates the detection frame as a learnable parameter in the model. In the decoder, the self-attention module is used for query updates, and the cross-attention module is used for feature detection.

Given the q_{th} probe frame $A_q = (x_q, y_q, w_q, h_q)$, the position query P_q is generated by $P_q = \text{MLP}(\text{PE}(A_q))$, where PE denotes the position encoder that generates the sine embedding, calculated as:

$$\begin{aligned}\text{PE}(A_q) &= \text{PE}(x_q, y_q, w_q, h_q)\\ &= \text{Cat}(\text{PE}(x_q), \text{PE}(y_q), \text{PE}(w_q), \text{PE}(h_q)).\end{aligned} \quad (12)$$

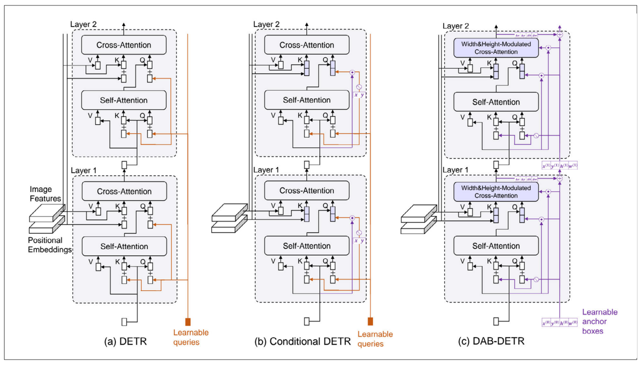


FIGURE 25. The difference between DAB-DETR, DETR, and Conditional DETR architectures [40].

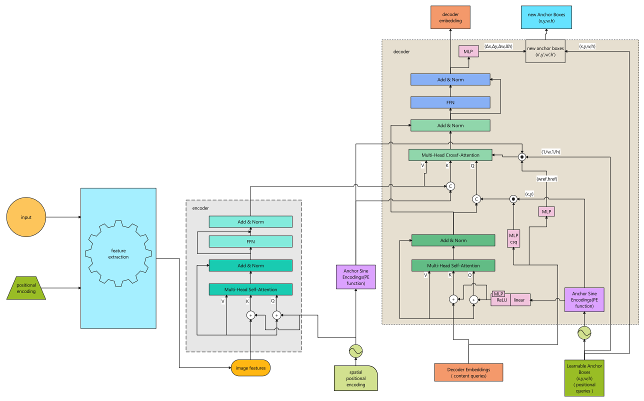


FIGURE 26. Overview of DAB architecture.

where Cat denotes the splicing function. We also use Self-Attn: $\mathbf{Q}_q = C_q + P_q, \mathbf{K}_q = C_q + P_q, \mathbf{V}_q = C_q$ to calculate the self-attention module.

Use the following formula to calculate the cross-attention module:

$$\begin{aligned}\text{Cross-Attn: } Q_q &= \text{Cat}(C_q, \text{PE}(x_q, y_q) \cdot \text{MLP}^{(\text{csq})}(C_q)),\\ K_{x,y} &= \text{Cat}(F_{x,y}, \text{PE}(x, y)), \quad V_{x,y} = F_{x,y}. \quad (13)\end{aligned}$$

where $F_{x,y}$ represents the image features at the point (x, y).

figure 26 shows all the above processes.

DN-DETR [39] changes the training way of DETR. Based on the previous framework, DN-DETR improves the bivariate bipartite graph-matching strategy in the original DETR. DN-DETR adds noise to the ground truth. The training model re-learns the ground truth, which can substantially reduce the matching instability caused by the Hungarian algorithm, thus speeding up the convergence. The decoder is comprises two parts, the noise reduction module and the matching module. The two modules were previously trained collaboratively through a complex masking mechanism, and the "no-object" class in the original DETR was eliminated from the model.

SPARSE DETR [35] improves on the encoder in DETR; previous algorithms have mainly been improving on the decoder, but SPARSE DETR offers a different perspective. SPARSE DETR proposes a DAM (Decoder cross-Attention

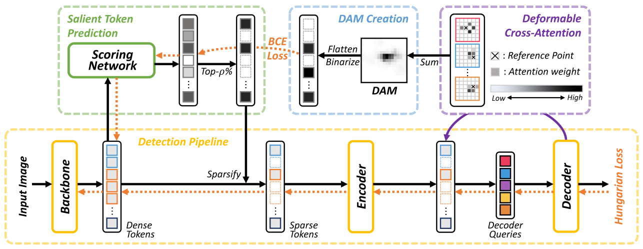


FIGURE 27. Training score network using DAB in SPARSE DETR [35].

Map) to construct a score network for the encoder part, which filters the features that can participate in the calculation of the cross-attention part of the decoder, as shown in figure 27.

DINO [43] has summarised its previous work and upgraded DETR, including three improvements on the encoder and decoder.

1) DINO builds on the previous DN-DETR by using a contrast learning strategy for the noise reduction module and re-adding the "no-object" category;
2) The model uses a Mixed Query Selection strategy to select the output portion of the encoder dynamically. When making a selection, the model retains only the a priori position information and not the content information, as the feature content information at this point can mislead the decoder into making a wrong selection;
3) When iteratively updating the probe box, DINO updates the i_{th} and $(i+1)_{th}$ layers using the parameters of the i_{th} layer. It will make better use of the previous position information, and the iterative formula is equation 14.

$$\begin{aligned}\Delta b_i &= \text{Layer}_\text{i}(b_{i-1}), \quad b_i' = \text{Update}(b_{i-1}, \Delta b_i),\\ b_i &= \text{Detach}(b_i'), \quad b_i^{(\text{pred})} = \text{Update}(b_{i-1}', \Delta b_i). \quad (14)\end{aligned}$$

where b_{i-1} denotes the $(i-1)_{th}$ input box, $b_i^{(\text{pred})}$ denotes the prediction box to be obtained, and b_i' denotes that it is not involved in the backpropagation calculation.

C. HYBRID MODELS

By hybrid detector in this paper, we mean a detector with both CNN and MSAs layers in the backbone. In terms of the training process, the algorithm is generally trained in a pre-training dataset (generally ImageNet-1K dataset [121]) for classification or others. After obtaining the pre-training model, the powerful representation capability of the model is used to fine-tune it in the downstream object detection task. Some of the most common algorithms in this series are listed in table 11.

ConvViT [58]was an early use of the MSAs layer to simulate CNN layer operations in order to improve the representation capability of the model, which inspired the development of subsequent models. ConvViT is influenced by [167] and [168], i.e. MSAs can simulate arbitrary convolution layers as long as they have enough heads. It developed a new self-attentive layer, the GPSA (Gated Positional Self-Attention) layer. The GPSA has a strong induced bias similar to the

TABLE 11. The hybrid architecture detector on the COCO 2017 val dataset. "P-Epochs" indicates the number of times the model was pre-trained.

Method	P-Epochs	GFLOPs	#Params(M)	mAP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Next-ViT-S-MaskRCNN1X [72]	300	290	52	45.9	68.3	50.7	-	41.8	-
Next-ViT-B-MaskRCNN2X [72]	300	340	65	47.2	69.6	51.6	-	42.8	-
GC ViT-S [73]	300	866	108	52.4	71.0	57.1	-	-	-
GC ViT-B [73]	300	1,018	146	52.9	71.7	57.8	-	-	-
Pix2seq-R50 [32]	300	-	37	43.0	61.0	45.6	25.1	46.9	59.4
Pix2seq-R101-DC5 [32]	300	-	57	45.0	63.2	48.6	28.2	48.9	60.4
MaxViT-S-Cascade MaskRCNN [74]	300	595	107	53.1	72.5	58.1	-	45.4	-
MaxViT-B-Cascade MaskRCNN [74]	300	856	157	53.4	72.9	58.1	-	45.7	-
ConvMAE-Mask-RCNN [165]	1,600	900	104	53.2	-	-	-	-	-
InternImage-B-Mask-RCNN1X [166]	300	501	115	48.8	71.0	53.9	-	44.0	-
InternImage-XL-Mask-RCNN1X [166]	300	1,782	387	55.3	74.5	60.2	-	48.0	-

convolutional layer, and its role is to replace the original SA (Self-Attention) layer in the ViT. Specifically, the GPSA layer is initialized to simulate the localization of the convolutional layer. Then it is freed from localization by adjusting the gating parameters so that each head of the MSAs targets fixed content information, giving it the ability to give extra attention to different locations. The whole process is like this:

$$\mathrm{GPSA}_h(\boldsymbol{X}) := \mathrm{normalize}\left[\boldsymbol{A}^h\right]\boldsymbol{X}\boldsymbol{W}^h_{val} \tag{15}$$

$$\boldsymbol{A}^h_{ij} := (1-\sigma(\lambda_h))\,\mathrm{softmax}\left(\boldsymbol{Q}^h_i\boldsymbol{K}^{h\top}_j\right) + \sigma(\lambda_h)\,\mathrm{softmax}\left(\boldsymbol{v}^{h\top}_{pos}\boldsymbol{r}_{ij}\right). \tag{16}$$

where $\sigma : x \mapsto {}^1/(1+e^{-x})$ denotes the activation function, $\boldsymbol{Q}^h_i$ denotes the query matrix of the i_{th} patches under the h_{th} self-attentive head, $\boldsymbol{v}^{h\top}_{pos}$ denotes the trainable embedding, and $\boldsymbol{r}_{ij}$ denotes the relative position encoding.

CVT [59] improves the efficiency of ViT-like models by introducing convolution into the transformer through the CTE (Convolutional Token Embedding) layer and the CP (Convolutional Projection) layer. The two modules are the CTE (Convolutional Token Embedding) layer and the CP (Convolutional Projection) layer, whose primary function is to downsample to enrich the feature map's representation. Replace the original position linear projection in ViT. Specifically, the token is mapped into the 1D space by `Flatten(Conv2d(Reshape2D(`x_i`),` s`))`. where x_i denotes the token to be mapped, `Conv2d` denotes the depth-separable convolution, i.e., `Depth-wise Conv2d` $\rightarrow$ `BatchNorm2d` $\rightarrow$ and `Point-wise Conv2d`, s means convolution kernel. As shown in figure 28 (b).

BoTNet [60] replaces the BottleNeck 3 × 3 convolutional layer in ResNet [8] with MSAs. CMT modules consider CNN to capture local information in images and MSAs to extract global correlation information. It is combining the two yields a model that is both efficient and accurate.

Mobile-Former [61] aims to improve recognition accuracy in lightweight applications. The model communicates MobileNet [10] with the transformer in parallel in both

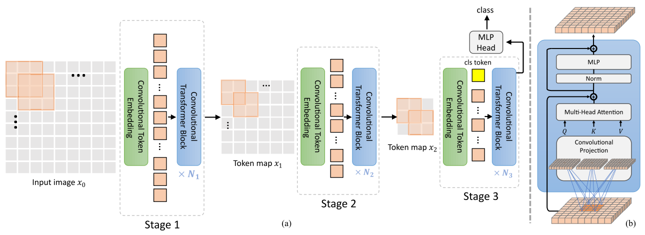


FIGURE 28. (a): The CVT framework, (b): Detail of CTB (Convolutional Transformer Block) [59].

directions. Due to efficiency issues, the transformer in Mobile-Former only uses a minimal number of Tokens, which affects the contribution of MHSA to the model's accuracy.

MetaFormer [62] is an abstraction framework that abstracts the transformer encoder into two components, the mutable component responsible for attention, the token-mixer, and the other remaining invariant components (such as MLP and residual concatenation).

Next-ViT [72] is contributed to both academic research and industrial deployments. Industrial deployments are very demanding in terms of model execution time and computational resources, which requires the model to achieve a specific scaling ratio to ensure that the model is optimal. Next-ViT comprises four stages, P2, P3, P4, and P5, each consisting of two core modules, the NCB (Next Convolution Block) and the NTB (Next Transformer Block). The NCB is used to calculate local information, and the NTB is used to calculate global information.

NCB follows the abstract design of MetaFormer and uses MHCA (Multi-Head Convolutional Attention) as a token-mixer, where the convolutional attention mechanism can learn the relationship between different tokens through the trainable parameter W in the local perceptual field. The formula is as follows:

$$\mathrm{CA}(z) = \mathrm{O}(W, (T_m, T_n)) \;\; where \;\; T_{\{m,n\}} \in z. \tag{17}$$

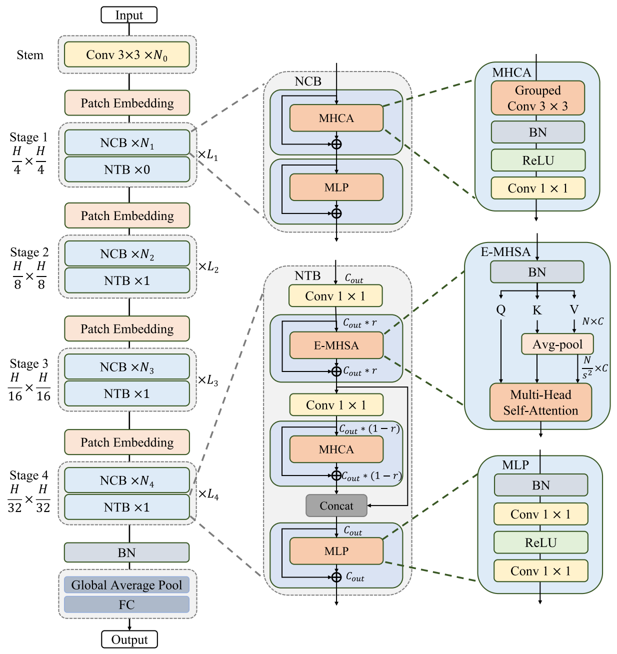


FIGURE 29. Overview of Next-ViT architecture [72].

where T_m and T_n denote the adjacent token in the input feature z, and O denotes the inner product operation.

NTB uses E-MHSA (Efficient Multi-Head Self Attention) to capture low-frequency signals, such as background and other global information, where the SA operator is used to reduce the spatial attention computational complexity. It can be expressed as equation $\mathrm{SA}(X) = \mathrm{Attention}(X \cdot W^Q, \mathrm{P}_s(X \cdot W^K), \mathrm{P}_s(X \cdot W^V))$, where $\mathrm{Attention}(Q, K, V) = \mathrm{softmax}(\frac{QK^T}{d_k})V$ and P_s means average pooling ($stride = s$), i.e., reducing the computational complexity of attention by downsampling. The overall framework is shown in figure 29.

ELAN [75] consists of three modules, and in this paper, we only discuss the core module called ELAB (Efficient Long-range). The ELAB module is referenced from the Swin Transformer model [54], which uses Shift-Convolution layers to extract local structural information from images and GMSA (Group-wise Multi-scale Self-Attention) module to extract global information. The Shift-Convolution layer consists of a shift operator and a 1×1 convolution. The primary function is to shift the first four groups of the input features, which have been divided into five equal groups, in the left, right, top, and bottom directions to ensure that the 1×1 convolution yields information about the surrounding pixels. The GMSA deals with group window-based self-attentive mechanisms, with the freedom to adjust the window size within each group [169]. When computing SA (Self Attention), the ASA (Accelerated Self-Attention) mechanism is used, except that the LN (Layer Normalization) in the transformer is replaced with Batch Normalization to ensure that the SA between groups is computed without additional overhead. The SA is computed in two Gaussian spaces instead of three, thus saving a 1×1 convolution in each SA. In addition, ELAN uses tricks such as shared SA scores and circular shifts along the diagonal to speed up the model's computation according to its network characteristics. It is worth mentioning that the current backbone of YOLOv7 [27] is the basis of the ELAN extension.

The ideas of the GC ViT [73], MaxViT [74], ConvMAE [165], and InternImage [166] models are borrowed from the architectural design of the Swin Transformer [54]. The accuracy of the object detection algorithm can be improved by generating multi-level feature maps in stages to extract information about objects contained in images with different resolutions in space.

Pix2seq [32] algorithm is designed to take advantage of the transformer's ability to process NLP sequences by serializing the bbox and class of the indicated object in the image to make predictions on the serialized bbox and class. The Pix2seq is essentially a generative self-supervised learning algorithm [170].

IV. X-RAY BAGGAGE DETECTION WITH DEEP LEARNING

X-Ray baggage detection is a task that, at this stage, is mainly carried out manually. There is a massive market for deep learning in this task. According to different detection methods, X-ray dangerous goods detection algorithms mainly include conventional image analysis, machine learning, and deep learning algorithms. This paper focuses on deep learning algorithms. Moreover, three types of supervised learning algorithms, classification, detection, and segmentation, are used for the introduction [76], [171]. Table 12 demonstrates the application of deep learning algorithms.

A. CLASSIFICATION

Classification algorithms were one of the first algorithms to emerge in this field. In simple terms, the need is fulfilled by determining the prohibited items' presence during the security screening process. The limitation of this algorithm is that it treats dangerous goods detection as a simple classification problem, and the final result does not accurately detect the type and location of hazardous materials.

Akcay et al. [77] used CNN to classify the dataset through migration learning. Solving a binary classification problem like the presence or absence of firearms demonstrated that using CNN is more effective than traditional machine learning algorithms like SVM.

Rogers et al. [78] first use of dual-energy X-ray pictures for imaging detection. High-energy and low-energy X-ray images captured by the dual-energy X-ray machine were used as asingle channel (H), dual channel ($\{H, -\log H\}$, $\{-\log H, -\log L\}$) and four-channel ($\{-\log L, L, H, -\log H\}$) as different the input channels to train the VGG-19 network for classification.

Zhao et al. [79], [80] introduces GAN [172] to the X-ray imaging detection task by a three-stage learning method for classification learning. The input X-ray dataset is first

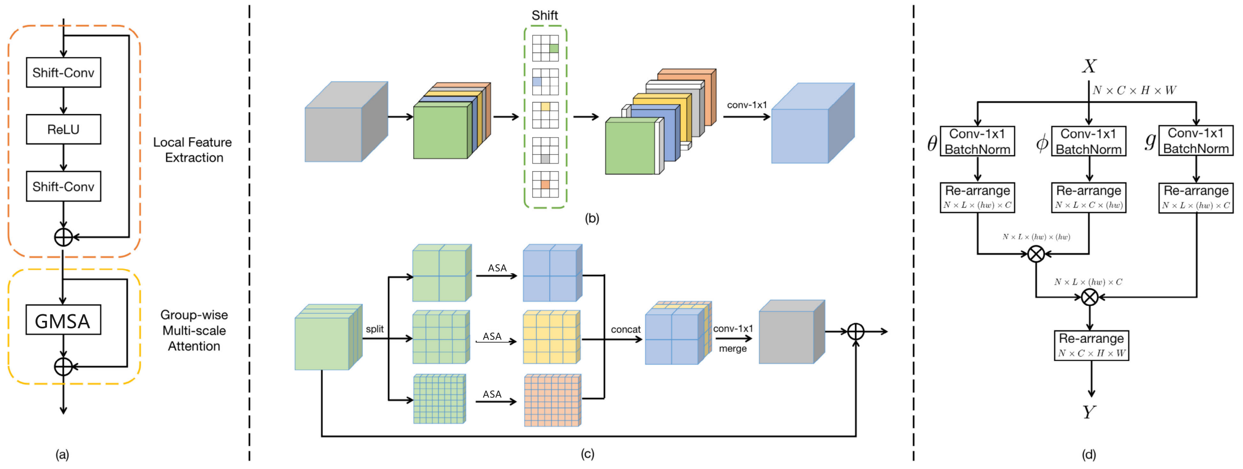

Shift-Conv
ReLU
Shift-Conv
Local Feature Extraction
GMSA
Group-wise Multi-scale Attention
(a)
Shift
conv-1x1
(b)
split
ASA
concat
conv-1x1 merge
(c)
X
$N \times C \times H \times W$
θ Conv-1x1 BatchNorm
ϕ Conv-1x1 BatchNorm
g Conv-1x1 BatchNorm
Re-arrange $N \times L \times (hw) \times C$
Re-arrange $N \times L \times C \times (hw)$
Re-arrange $N \times L \times (hw) \times C$
$N \times L \times (hw) \times (hw)$
$N \times L \times (hw) \times C$
Re-arrange $N \times C \times H \times W$
Y
(d)

FIGURE 30. (a)GLAB framework, (b)Shift-Convolution layer, (c)GMSA layer, (d) Calculation of ASA [75].

TABLE 12. Deep learning algorithms in the field of X-ray baggage detection. (If not specified, ACC, mAP, and mIOU in the last column correspond to the performance metrics of the classification, detection, and segmentation algorithms in the first column).

Tasks	Literature	Year	Datasets	Num. Pictures	Methods	ACC(%)/mAP(%)/mIOU(%)
Classification	[77]	2016	Private	6,997	AlexNet	99.0
	[78]	2017	Private	120,000	VGG	99.5
	[79; 80]	2018	-	-	GANs+KNN	91.1
	[81]	2019	SIXray10	98,219	ResNet50+CHR	77.9
			SIXray100	901,829		57.9
			SIXray1000	1,051,302		37.0
			SIXray10	98,219	DenseNet+CHR	79.6
			SIXray100	901,829		59.9
			SIXray1000	1,051,302		48.4
Detection	[84]	2017	DBF2/6	11,627	SW-CNN+ResNet-101	77.6
					RCNN+VGG16	77.9
					Faster RCNN+VGG16	88.3
					R-FCN+ResNet101+ResNet-101	85.6
	[138]	2019	TSADatasets	13,786	SSD-InceptionV2	75.2
					Faster-RCNN-ResNet101	91.7
					Faster-RCNN-InceptionResNetV2	94.1
	[85]	2020	TSADatasets-Dataset A	6000(HC)+35000(SOC)	Faster RCNN+MatchInstancesImages	94.1
			TSADatasets-Dataset B	19000(HC)+70000(SOC)		95.8
	[88]	2019	GDXray	8,150	CMST	94.5(mAP)/96.4(mIOU)
			SIXray	1,059,231		93.7(mAP)/96.9(mIOU)
	[89]	2021	PIDray	47,677	SDANet+ResNet-101-FPN	61.6
	[92]	2021	HiXray	45,364	SSD+LIM	73.1
					FCOS+LIM 7	77.3
					YOLOv5+LIM	83.2
	[90]	2022	deei6	7,022	conditional GAN	67.5
Segmentation	[93]	2019	Private	3,534	Mask R-CNN+ResNet101	97.9
					Dual CNN+ResNet18	66(ACC)
	[95]	2020	GDXray	8,150	TST	96.72(mAP)
			SIXray	1,059,231		95.16(mAP)
			OPIXray	8,885		75.32(mAP)
			COMPASS-XP	11,568		58.42(mAP)
			Combined Dataset	1,087,833		46.57(mAP)
	[94]	2022	PIXray	5,046	DDoAS	76.3

classified and labeled by the angular information of the foreground objects extracted from the input images. GAN generates new objects, and finally, a small classification network is used to confirm whether the generated images belong to the correct class.

CHR [81] model performs classification detection on the SIXray dataset [81]. Since the SIXray dataset is constructed to simulate a realistic environment where a significant imbalance in security screening data occurs, the CHR model copes with the class imbalance by extracting image features from

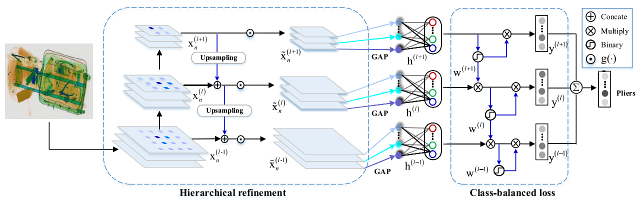


FIGURE 31. Overview of CHR architecture [81].

three consecutive layers. Specifically, the backward layer in the CHR model is upsampled and connected to the preceding layer, as shown in figure 31.

In the figure, the $g()$ function is used to remove redundant information from the feature map by feeding the advanced features into three layers of $(\{h(\tilde{x}_n^{(l-1)}), h(\tilde{x}_n^{(l)}), h(\tilde{x}_n^{(l+1)})\})$ for differentiation classification. A multi-level strategy is used in the CHR model to extract the features of the objects better.

Caldwell et al. [83] investigates the generalization ability of models trained with different datasets from various scanners. The authors created training and test samples from single or multiple domains to investigate the effect of migration between other models. The limitation is that migration learning is still challenging due to the scanners' unknown parameters and the CNN's ability to generalize to unseen target datasets.

B. DETECTION

Most existing X-ray dangerous goods detection algorithms use a CNN framework to detect the type and location of dangerous goods in luggage. The limitation of this algorithm is that the detection results rely heavily on the texture information of the detected objects and do not fully use the shape contour information of the objects. It leads to the fact that the actual detection results do not achieve the desired results.

Akcay et al. [84] detects and identifies imaging on the DBF2/6 using the Faster R-CNN [21] algorithm. The mAP on the DBF6 reached 88.3%.

Sigman et al. [85] proposes a semi-supervised domain adaptation learning algorithm, the Background Adaptive FRCNN (Background Adaptive Faster R-CNN) algorithm. The authors assume that, in reality, there are no dangerous goods in the security-checked images, and this assumption aims to obtain the dataset more quickly. The algorithm has two domains: a manually collected domain with hazards and a real-world domain without hazards. In addition, two domain discriminators are trained using adversarial, one for discriminating the target offer frame and the other for discriminating the image features. Only the background area outside the target proposal region and ground truth is extracted for features when training on a manual dataset. It allows the model to identify the hazards better, as the background features of the images with hazards (manual dataset) will match the features of the images without hazards (real dataset).

Subramani et al. [86] trained on the SIXray10 dataset using the SSD [145] and RetinaNet [148] detectors, with mAPs of 60.5% and 60.9%, respectively. [87] used YOLOv2 [24] to achieve an average accuracy of 94.5% and a recall of 92.6% on the unpublished dataset SASC.

Hassan et al. [88] uses a cascaded multi-scale structure to form RoI after extracting tensors from different angles of the object. The mAP reached over 96% on both the GDXray and SIXray.

Wang et al. [89] designed a Selective Dense Attention Network, SDANet, which constructs a strong baseline on the PIDray, which consists of a dense attention module and a dependency refinement module, as shown in figure 32. SDANet uses the attention mechanism to focus on target objects in complex contexts in a multi-level feature pyramid graph. The final APs on easy, medium, and hard are 71.2%, 64.2%, and 49.5%, respectively.

Tao et al. [92] proposes the LIM, Lateral Inhibition Module, which is a module that ignores task-irrelevant information and focuses only on recognizable features when objects overlap each other. Specifically, LIM is a carefully designed flexible add-on module that minimizes the flow of noisy information through the Bidirectional Propagation module and activates the boundaries of the most recognizable features from four directions through the Boundary Activation module, as shown in figure 33. LIM achieved 83.2% and 90.6% mAP on the HiXray and OPIXray.

Isaac et al. [90] has used the conditional GAN model as well as the FFL technique to analyze four types of images, including high-energy, low-energy, and effective ray-Z images, as well as pseudo-color images synthesized

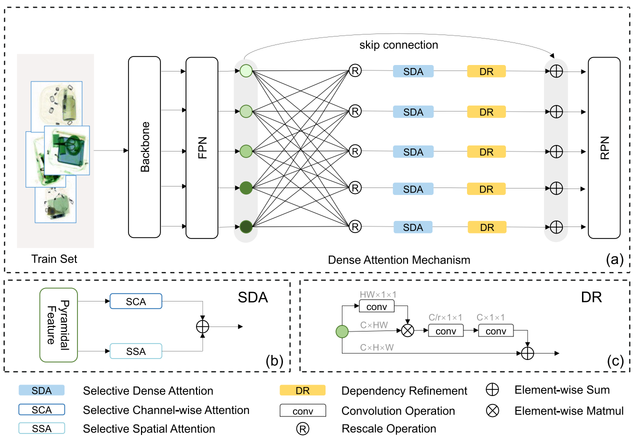


FIGURE 32. Overview of SDANet framework [89].

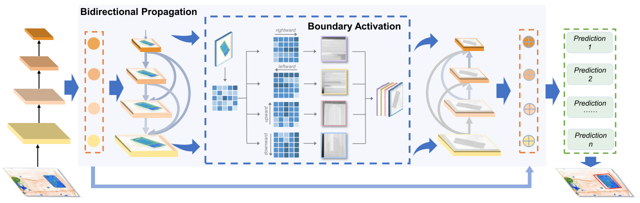


FIGURE 33. Overview of LIM Detailed Architecture [92].

from these three types of images. The results show that the pseudo-color maps synthesized by the conditional GAN model achieve significant results on the private dataset deei6.

C. SEGMENTATION

Object segmentation algorithms segment images into multiple sub-regions. In X-ray images with highly overlapping targets, segmentation algorithms often rely on additional information to complete the segmentation, such as the contour information of hazardous materials. It brings additional computational effort while the hazardous material profile information dominates the final segmentation result.

Gaus et al. [93] uses a dual convolutional neural network architecture to detect automatic anomalies in x-ray images. The paper uses R-CNN [18], mask R-CNN [22], and RetinaNet-like detection networks to provide object localization for specific target object classes. Specifically, the images are segmented using mask R-CNN to initialize the RoI, followed by a negative/positive bifurcation of the previous RoI by a network such as RetinaNet, with a segmentation accuracy of 97.6%.

Hassan et al. [95] segmented the targets on the images by extracting the structural tensor from different angles, and finally achieved a segmentation mAP of 96.7%/96.16%/75.32%/58.4% on GDXray/SIXray/OPIXray/Compass-XP respectively.

Ma et al. [94] addresses the problem of inaccurate identification of different contraband or dangerous goods due to differences in appearance. The model named DDoAS consists of two modules: DDoM, which accurately infers contraband information from a dense overlapping background by means of dense backlinks, and ADM, which aims to improve the low learning efficiency due to differences in shape and size between different contraband items. The limitation of the DDoAS algorithm is that the model uses additional optical information (object edges and vertices) to assist in verification, which makes it challenging to detect contraband with poor edge information, such as small folding knives.

V. EXPERIMENT

In this section, to test the accuracy of standard models in detecting X-ray images without modifying the original structure and to provide directions for subsequent research, we select the four most common models among the three types of algorithms for experimentation. Specifically, these are: YOLOv5,[5] YOLOv7 [27], DINO [43], and NextViT [72].

The data sets used in this experiment are the processed SIXray[6] and PIDray.[7] We have marked them as *SIXray*$_p$ and *PIDray*$_p$ respectively. *SIXray*$_p$ contains five classes, namely Gun, Knife, Pliers, Scissors, Wrench, and *PIDray*$_p$ contains 12 classes, namely Baton, Bullet, Gun, Hammer, HandCuffs, Knife, Lighter, Pliers, Powerbank, Scissors, Sprayer,

[5]https://github.com/ultralytics/yolov5

[6]https://universe.roboflow.com/object-detection/ugku

[7]https://universe.roboflow.com/object-detection/security_xray

TABLE 13. ***SIXray*$_p$ and *PIDray*$_p$ Dataset.**

	#Num. TrainingSet	#Num. TestingSet	Classes
$SIXray_p$	17K	840	5
$PIDray_p$	9.4K	422	12

TABLE 14. **YOLOv5 detects basic results for the *PIDray*$_p$ dataset.**

Class	Images	Labels	P	R	mAP@.5	mAP@.5:.95
all	422	641	0.942	0.89	0.921	0.677
Baton	422	23	0.97	0.957	0.969	0.739
Bullet	422	23	0.95	0.998	0.995	0.73
Gun	422	28	0.903	0.964	0.985	0.735
Hammer	422	75	0.984	0.92	0.974	0.694
HandCuffs	422	12	0.946	0.998	0.995	0.767
Knife	422	76	0.915	0.846	0.887	0.686
Lighter	422	77	0.918	0.58	0.654	0.403
Pliers	422	127	0.998	0.942	0.982	0.755
Powerbank	422	44	0.826	0.773	0.804	0.557
Scissors	422	7	0.998	0.805	0.862	0.613
Sprayer	422	77	0.912	0.922	0.955	0.713
Wrench	422	72	0.986	0.977	0.993	0.729

Wrench, as shown in table 13 We fine-tuned the training data set directly and tested the results on the testing data set.

As the detection process is real-time, the YOLOv5 and YOLOv7 models are used in preference to the detection process, and the results are shown in table 14, table 15, table 16, table 17 respectively. These tables show that YOLOv7 is more accurate in recognition than YOLOv5, and v7 has a more incredible inference speed than the other models due to the use of techniques such as model re-parameterization. The four models' visual comparison results are shown in figure 34. As can be seen in subplot (c), the Transformer-based and hybrid models do not work well in X-ray image detection.

The main reason for this is that the MSAs mechanism learns the contour information of the object. However, in the X-ray image, the hazardous object to be detected is covered or obscured by a large number of other objects, which leads to confusing feature information obtained by MSAs and cannot correctly distinguish the exact location of the object; on the contrary, the CNN learns more information about the texture of the object from the pseudo-color image, which helps identify the type of object. Inspired by several models, DINO incorporates a variety of factors that facilitate improved recognition accuracy but does not perform specific optimizations and has a lower mAP than other models in this area. In addition, Next-ViT, as a hybrid model, combines the advantages of both Conv and MSAs. However, as it is not an end-to-end detection model and its structural construction does not apply to X-ray images with many overlapping objects, it has no particular advantages regarding the accuracy and operational efficiency. A hybrid model more suitable for X-ray images should be designed.

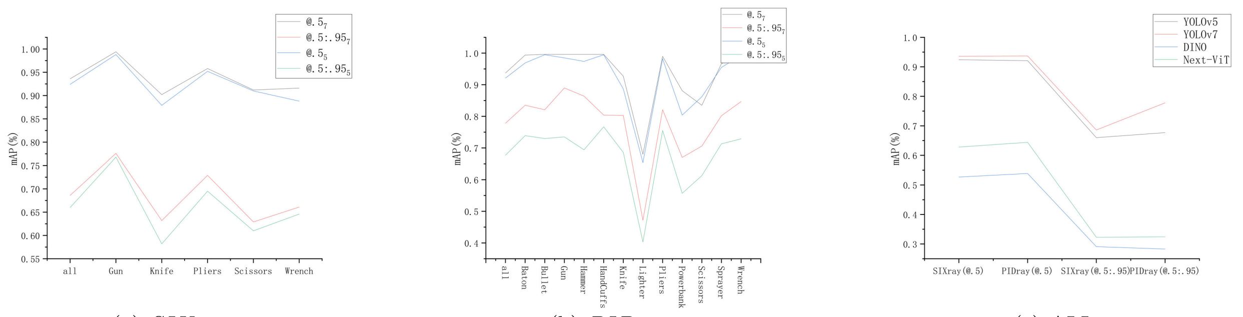


FIGURE 34. **(a) Represents the detailed identification results of the YOLOv5 versus YOLOv7 for each category on the *SIXray$_p$* dataset. "@.5$_5$" and "@.5$_7$" denote the value of mAP 50 under YOLOv5 and YOLOv7, respectively, and so on; (b) Representation of the detailed identification results of the YOLOv5 versus YOLOv7 models for each category on the *PIDray$_p$* dataset; (c) indicates the comparison mAP results of the four models YOLOv5, YOLOv7, DINO, and Next-ViT.**

TABLE 15. **YOLOv5 detects basic results for the *SIXray$_p$* dataset.**

Class	Images	Labels	P	R	mAP@.5	mAP@.5:.95
all	840	1,586	0.926	0.881	0.924	0.66
Gun	840	432	0.962	0.965	0.988	0.768
Knife	840	200	0.904	0.804	0.879	0.582
Pliers	840	562	0.937	0.916	0.952	0.695
Scissors	840	109	0.931	0.862	0.91	0.61
Wrench	840	283	0.897	0.859	0.888	0.646

TABLE 16. **YOLOv7 detects basic results for the *PIDray$_p$* dataset.**

Class	Images	Labels	P	R	mAP@.5	mAP@.5:.95
all	422	641	0.956	0.899	0.937	0.778
Baton	422	23	0.998	0.956	0.994	0.835
Bullet	422	23	0.982	0.998	0.996	0.821
Gun	422	28	0.984	0.998	0.996	0.89
Hammer	422	75	0.998	0.987	0.996	0.864
HandCuffs	422	12	0.978	0.998	0.996	0.804
Knife	422	76	0.959	0.816	0.927	0.803
Lighter	422	77	0.937	0.584	0.68	0.472
Pliers	422	127	0.998	0.971	0.99	0.821
Powerbank	422	44	0.878	0.75	0.881	0.67
Scissors	422	7	0.827	0.857	0.835	0.706
Sprayer	422	77	0.933	0.899	0.967	0.802
Wrench	422	72	0.998	0.965	0.99	0.847

TABLE 17. **YOLOv7 detects basic results for the *SIXray$_p$* dataset.**

Class	Images	Labels	P	R	mAP@.5	mAP@.5:.95
all	840	1,586	0.947	0.888	0.936	0.686
Gun	840	432	0.982	0.97	0.994	0.776
Knife	840	200	0.929	0.845	0.902	0.632
Pliers	840	562	0.96	0.918	0.958	0.729
Scissors	840	109	0.939	0.85	0.912	0.629
Wrench	840	283	0.924	0.855	0.916	0.661

VI. CONCLUSION

This paper reviews the more popular deep learning object detection algorithms of recent years. Also, it summarises the application of deep learning to the field of X-ray baggage dangerous goods detection. While many models have temporarily solved some of the problems in this area, huge limitations remain:

1) The pseudo-color pictures formed by dual-energy X-ray still do not work well with modern detection models and must be modified in depth to obtain more reasonable results.
2) The timeliness of the algorithm is a factor that must be considered at the moment.
3) In reality, if prohibited goods are in luggage, they will inevitably be wrapped in layers. The resulting X-ray images can be extreme, with objects stacked on top of each other over a large area. The accuracy of existing models for identification may need to be higher.
4) The current X-ray baggage image dataset is still small and of low quality, which affects the training of deep learning models.

In response to the above challenges, we offer the following suggestions:

1) Using image translation or style transfer techniques to generate corresponding natural light images from X-ray images, expanding the X-ray baggage dataset.
2) The use of image pairs formed by high- and low-energy rays, combined with images in natural light, enriches the color of X-ray images and brings them closer to natural light images.
3) Reduce the cost of 3D CT scan recognition technology by converting 2D algorithms to 3D algorithms to recognize stacked layers that are difficult to recognize in the 2D case.
4) Image feature extraction and synthesis using a Diffusion model more advanced than GAN to generate high-quality X-ray images containing prohibited items.
5) Although most prohibited items are masked, they do not change their original shape excessively when exposed to X-ray. They can still be identified using contour information through a rational algorithm design. One of the future directions in X-ray dangerous goods detection is using hybrid algorithms that combine texture features

and contour information of prohibited items for identification.

6) In order to make fair comparisons, evaluation criteria must be established on public datasets.

REFERENCES

[1] A. Schwaninger, A. Bolfing, T. Halbherr, S. Helman, A. Belyavin, and L. Hay, "The impact of image based factors and training on threat detection performance in X-ray screening," Tech. Rep., 2008.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, 2012, pp. 1–9.

[3] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16 × 16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.

[4] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1, Dec. 2001. p. I.

[5] N. Park and S. Kim, "How do vision transformers work?" 2022, *arXiv:2202.06709*.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[9] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[10] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

[11] A. Howard, M. Sandler, B. Chen, W. Wang, L.-C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le, "Searching for MobileNetV3," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1314–1324.

[12] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6848–6856.

[13] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "ShuffleNet V2: Practical guidelines for efficient CNN architecture design," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 116–131.

[14] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.

[15] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "GhostNet: More features from cheap operations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1580–1589.

[16] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.

[17] X. Ding, X. Zhang, N. Ma, J. Han, G. Ding, and J. Sun, "RepVGG: Making VGG-style ConvNets great again," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 13733–13742.

[18] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2014.

[20] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.

[21] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[22] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2961–2969.

[23] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[24] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 7263–7271.

[25] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*.

[26] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*.

[27] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," 2022, *arXiv:2207.02696*.

[28] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, "You only learn one representation: Unified network for multiple tasks," 2021, *arXiv:2105.04206*.

[29] X. Long, K. Deng, G. Wang, Y. Zhang, Q. Dang, Y. Gao, H. Shen, J. Ren, S. Han, E. Ding, and S. Wen, "PP-YOLO: An effective and efficient implementation of object detector," 2020, *arXiv:2007.12099*.

[30] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "YOLOX: Exceeding YOLO series in 2021," 2021, *arXiv:2107.08430*.

[31] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Computer Vision—ECCV 2020*. Glasgow, U.K., Aug. 2020, pp. 213–229.

[32] T. Chen, S. Saxena, L. Li, D. J. Fleet, and G. Hinton, "Pix2seq: A language modeling framework for object detection," 2021, *arXiv:2109.10852*.

[33] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable DETR: Deformable transformers for end-to-end object detection," in *Proc. Int. Conf. Learn. Represent.*, 2021, pp. 1–16.

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

[35] B. Roh, J. Shin, W. Shin, and S. Kim, "Sparse DETR: Efficient end-to-end object detection with learnable sparsity," 2021, *arXiv:2111.14330*.

[36] P. Gao, M. Zheng, X. Wang, J. Dai, and H. Li, "Fast convergence of DETR with spatially modulated co-attention," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 3621–3630.

[37] D. Meng, X. Chen, Z. Fan, G. Zeng, H. Li, Y. Yuan, L. Sun, and J. Wang, "Conditional DETR for fast training convergence," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 3651–3660.

[38] Y. Wang, X. Zhang, T. Yang, and J. Sun, "Anchor DETR: Query design for transformer-based detector," in *Proc. AAAI Conf. Artif. Intell.*, 2022, vol. 36, no. 3, pp. 2567–2575.

[39] F. Li, H. Zhang, S. Liu, J. Guo, L. M. Ni, and L. Zhang, "DN-DETR: Accelerate DETR training by introducing query DeNoising," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 13619–13627.

[40] S. Liu, F. Li, H. Zhang, X. Yang, X. Qi, H. Su, J. Zhu, and L. Zhang, "DAB-DETR: Dynamic anchor boxes are better queries for DETR," 2022, *arXiv:2201.12329*.

[41] Y. Fang, B. Liao, X. Wang, J. Fang, J. Qi, R. Wu, J. Niu, and W. Liu, "You only look at one sequence: Rethinking transformer in vision through object detection," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 26183–26197.

[42] Z. Sun, S. Cao, Y. Yang, and K. Kitani, "Rethinking transformer-based set prediction for object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 3611–3620.

[43] H. Zhang, F. Li, S. Liu, L. Zhang, H. Su, J. Zhu, L. M. Ni, and H.-Y. Shum, "DINO: DETR with improved denoising anchor boxes for end-to-end object detection," 2022, *arXiv:2203.03605*.

[44] Z. Yao, J. Ai, B. Li, and C. Zhang, "Efficient DETR: Improving end-to-end object detector with dense prior," 2021, *arXiv:2104.01318*.

[45] X. Dai, Y. Chen, J. Yang, P. Zhang, L. Yuan, and L. Zhang, "Dynamic DETR: End-to-end object detection with dynamic attention," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 2988–2997.

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

[47] H. Bao, L. Dong, S. Piao, and F. Wei, "BEiT: BERT pre-training of image transformers," 2021, *arXiv:2106.08254*.

[48] Z. Peng, L. Dong, H. Bao, Q. Ye, and F. Wei, "BEiT v2: Masked image modeling with vector-quantized visual tokenizers," 2022, *arXiv:2208.06366*.

[49] W. Wang, H. Bao, L. Dong, J. Bjorck, Z. Peng, Q. Liu, K. Aggarwal, O. K. Mohammed, S. Singhal, S. Som, and F. Wei, "Image as a foreign language: BEiT pretraining for all vision and vision-language tasks," 2022, *arXiv:2208.10442*.

[50] W. Wang, Y. Cao, J. Zhang, and D. Tao, "FP-DETR: Detection transformer advanced by fully pre-training," in *Proc. Int. Conf. Learn. Represent.*, 2021, pp. 1–14.

[51] J. Yang, C. Li, P. Zhang, X. Dai, B. Xiao, L. Yuan, and J. Gao, "Focal self-attention for local-global interactions in vision transformers," 2021, *arXiv:2107.00641*.

[52] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 568–578.

[53] P. Zhang, X. Dai, J. Yang, B. Xiao, L. Yuan, L. Zhang, and J. Gao, "Multi-scale vision longformer: A new vision transformer for high-resolution image encoding," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 2998–3008.

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

[55] D. Zhang, H. Zhang, J. Tang, M. Wang, X. Hua, and Q. Sun, "Feature pyramid transformer," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2020, pp. 323–339.

[56] Y. Yuan, R. Fu, L. Huang, W. Lin, C. Zhang, X. Chen, and J. Wang, "HRFormer: High-resolution vision transformer for dense predict," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 7281–7293.

[57] J. Gu, H. Kwon, D. Wang, W. Ye, M. Li, Y.-H. Chen, L. Lai, V. Chandra, and D. Z. Pan, "Multi-scale high-resolution vision transformer for semantic segmentation," 2021, *arXiv:2111.01236*.

[58] S. d'Ascoli, H. Touvron, M. L. Leavitt, A. S. Morcos, G. Biroli, and L. Sagun, "ConViT: Improving vision transformers with soft convolutional inductive biases," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 2286–2296.

[59] H. Wu, B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, and L. Zhang, "CvT: Introducing convolutions to vision transformers," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 22–31.

[60] A. Srinivas, T.-Y. Lin, N. Parmar, J. Shlens, P. Abbeel, and A. Vaswani, "Bottleneck transformers for visual recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 16519–16529.

[61] Y. Chen, X. Dai, D. Chen, M. Liu, X. Dong, L. Yuan, and Z. Liu, "Mobile-former: Bridging MobileNet and transformer," 2021, *arXiv:2108.05895*.

[62] W. Yu, M. Luo, P. Zhou, C. Si, Y. Zhou, X. Wang, J. Feng, and S. Yan, "MetaFormer is actually what you need for vision," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 10819–10829.

[63] B. Wu, C. Xu, X. Dai, A. Wan, P. Zhang, Z. Yan, M. Tomizuka, J. Gonzalez, K. Keutzer, and P. Vajda, "Visual transformers: Token-based image representation and processing for computer vision," 2020, *arXiv:2006.03677*.

[64] T. Xiao, M. Singh, E. Mintun, T. Darrell, P. Dollár, and R. Girshick, "Early convolutions help transformers see better," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 30392–30400.

[65] A. Srinivas, T.-Y. Lin, N. Parmar, J. Shlens, P. Abbeel, and A. Vaswani, "Bottleneck transformers for visual recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*. Washington, DC, USA: IEEE Computer Society, Jun. 2021, pp. 16514–16524.

[66] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 10347–10357.

[67] K. Yuan, S. Guo, Z. Liu, A. Zhou, F. Yu, and W. Wu, "Incorporating convolution designs into visual transformers," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 579–588.

[68] Y. Li, K. Zhang, J. Cao, R. Timofte, and L. Van Gool, "LocalViT: Bringing locality to vision transformers," 2021, *arXiv:2104.05707*.

[69] X. Chu, Z. Tian, B. Zhang, X. Wang, and C. Shen, "Conditional positional encodings for vision transformers," 2021, *arXiv:2102.10882*.

[70] Q. Zhang and Y.-B. Yang, "ResT: An efficient transformer for visual recognition," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 15475–15485.

[71] Z. Dai, H. Liu, Q. V. Le, and M. Tan, "CoAtNet: Marrying convolution and attention for all data sizes," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 3965–3977.

[72] J. Li, X. Xia, W. Li, H. Li, X. Wang, X. Xiao, R. Wang, M. Zheng, and X. Pan, "Next-ViT: Next generation vision transformer for efficient deployment in realistic industrial scenarios," 2022, *arXiv:2207.05501*.

[73] A. Hatamizadeh, H. Yin, J. Kautz, and P. Molchanov, "Global context vision transformers," 2022, *arXiv:2206.09959*.

[74] Z. Tu, H. Talebi, H. Zhang, F. Yang, P. Milanfar, A. Bovik, and Y. Li, "MaxViT: Multi-axis vision transformer," 2022, *arXiv:2204.01697*.

[75] X. Zhang, H. Zeng, S. Guo, and L. Zhang, "Efficient long-range attention network for image super-resolution," 2022, *arXiv:2203.06697*.

[76] S. Akcay and T. Breckon, "Towards automatic threat detection: A survey of advances of deep learning within X-ray security imaging," *Pattern Recognit.*, vol. 122, Feb. 2022, Art. no. 108245.

[77] S. Akcay, M. E. Kundegorski, M. Devereux, and T. P. Breckon, "Transfer learning using convolutional neural networks for object classification within X-ray baggage security imagery," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 1057–1061.

[78] T. W. Rogers, N. Jaccard, and L. D. Griffin, "A deep learning framework for the automated inspection of complex dual-energy X-ray cargo imagery," *Proc. SPIE*, vol. 10187, pp. 106–117, May 2017.

[79] Z. Zhao, H. Zhang, and J. Yang, "A GAN-based image generation method for X-ray security prohibited items," in *Proc. Chin. Conf. Pattern Recognit. Comput. Vis. (PRCV)*. Springer, 2018, pp. 420–430.

[80] J. Yang, Z. Zhao, H. Zhang, and Y. Shi, "Data augmentation for X-ray prohibited item images using generative adversarial networks," *IEEE Access*, vol. 7, pp. 28894–28902, 2019.

[81] C. Miao, L. Xie, F. Wan, C. Su, H. Liu, J. Jiao, and Q. Ye, "SIXray: A large-scale security inspection X-ray benchmark for prohibited item discovery in overlapping images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2119–2128.

[82] T. Morris, T. Chien, and E. Goodman, "Convolutional neural networks for automatic threat detection in security X-ray images," in *Proc. 17th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2018, pp. 285–292.

[83] M. Caldwell, M. Ransley, T. W. Rogers, and L. D. Griffin, "Transferring X-ray based automated threat detection between scanners with different energies and resolution," in *Proc. SPIE*, vol. 10441, 2017, pp. 130–139.

[84] S. Akcay and T. P. Breckon, "An evaluation of region based object detection strategies within X-ray baggage security imagery," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 1337–1341.

[85] J. B. Sigman, G. P. Spell, K. J. Liang, and L. Carin, "Background adaptive faster R-CNN for semi-supervised convolutional object detection of threats in X-ray images," *Proc. SPIE*, vol. 11404, pp. 12–21, May 2020.

[86] M. Subramani, K. Rajaduari, S. D. Choudhury, A. Topkar, and V. Ponnusamy, "Evaluating one stage detector architecture of convolutional neural network for threat object detection using X-ray baggage security imaging," *Revue d'Intell. Artificielle*, vol. 34, no. 4, pp. 495–500, Sep. 2020.

[87] Z. Liu, J. Li, Y. Shu, and D. Zhang, "Detection and recognition of security detection object based on YOLO9000," in *Proc. 5th Int. Conf. Syst. Informat. (ICSAI)*, Nov. 2018, pp. 278–282.

[88] T. Hassan, S. H. Khan, S. Akcay, M. Bennamoun, and N. Werghi, "Deep cmst framework for the autonomous recognition of heavily occluded and cluttered baggage items from multivendor security radiographs," *CoRR*, vol. 14, p. 17, Dec. 2019.

[89] B. Wang, L. Zhang, L. Wen, X. Liu, and Y. Wu, "Towards real-world prohibited item detection: A large-scale X-ray benchmark," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 5412–5421.

[90] B. K. S. Isaac-Medina, N. Bhowmik, C. G. Willcocks, and T. P. Breckon, "Cross-modal image synthesis within dual-energy X-ray security imagery," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2022, pp. 333–341.

[91] S. Akcay, M. E. Kundegorski, C. G. Willcocks, and T. P. Breckon, "Using deep convolutional neural network architectures for object classification and detection within X-ray baggage security imagery," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 9, pp. 2203–2215, Sep. 2018.

[92] R. Tao, Y. Wei, X. Jiang, H. Li, H. Qin, J. Wang, Y. Ma, L. Zhang, and X. Liu, "Towards real-world X-ray security inspection: A high-quality benchmark and lateral inhibition module for prohibited items detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 10923–10932.

[93] Y. F. A. Gaus, N. Bhowmik, S. Akçay, P. M. Guillén-Garcia, J. W. Barker, and T. P. Breckon, "Evaluation of a dual convolutional neural network architecture for object-wise anomaly detection in cluttered X-ray security imagery," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2019, pp. 1–8.

[94] B. Ma, T. Jia, M. Su, X. Jia, D. Chen, and Y. Zhang, "Automated segmentation of prohibited items in X-ray baggage images using dense de-overlap attention snake," *IEEE Trans. Multimedia*, early access, May 11, 2022, doi: 10.1109/TMM.2022.3174339.

[95] T. Hassan and N. Werghi, "Trainable structure tensors for autonomous baggage threat detection under extreme occlusion," in *Proc. Asian Conf. Comput. Vis.*, 2020, pp. 1–16.

[96] Ultralytics. (2022). *YOLOv5*. [Online]. Available: https://github.com/ultralytics/yolov5/releases/tag/v6.1

[97] S. S. A. Zaidi, M. S. Ansari, A. Aslam, N. Kanwal, M. Asghar, and B. Lee, "A survey of modern deep learning based object detection models," *Digit. Signal Process.*, vol. 126, Jun. 2022, Art. no. 103514.

[98] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1492–1500.

[99] C.-Y. Wang, H.-Y. M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh, "CSPNet: A new backbone that can enhance learning capability of CNN," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 390–391.

[100] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel, "ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness," 2018, *arXiv:1811.12231*.

[101] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.

[102] T. Lin, Y. Wang, X. Liu, and X. Qiu, "A survey of transformers," 2021, *arXiv:2106.04554*.

[103] Y. Liu, Y. Zhang, Y. Wang, F. Hou, J. Yuan, J. Tian, Y. Zhang, Z. Shi, J. Fan, and Z. He, "A survey of visual transformers," 2021, *arXiv:2111.06091*.

[104] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," *Int. J. Comput. Vis.*, vol. 128, pp. 261–318, Feb. 2020.

[105] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, "Transformers in vision: A survey," *ACM Comput. Surv.*, vol. 54, no. 10, pp. 1–41, Jan. 2022.

[106] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu, Z. Yang, Y. Zhang, and D. Tao, "A survey on vision transformer," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 1, pp. 87–110, Jan. 2023.

[107] D. Zhou, Z. Yu, E. Xie, C. Xiao, A. Anandkumar, J. Feng, and J. M. Alvarez, "Understanding the robustness in vision transformers," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 27378–27394.

[108] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur, "Sharpness-aware minimization for efficiently improving generalization," 2020, *arXiv:2010.01412*.

[109] X. Chen, C.-J. Hsieh, and B. Gong, "When vision transformers outperform ResNets without pre-training or strong data augmentations," 2021, *arXiv:2106.01548*.

[110] M. M. Naseer, K. Ranasinghe, S. H. Khan, M. Hayat, F. S. Khan, and M.-H. Yang, "Intriguing properties of vision transformers," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 23296–23308.

[111] J. Guo, K. Han, H. Wu, Y. Tang, X. Chen, Y. Wang, and C. Xu, "CMT: Convolutional neural networks meet vision transformers," 2021, *arXiv:2107.06263*.

[112] F. L. Roder, "Explosives detection by dual-energy computed tomography (CT)," *Proc. SPIE*, vol. 182, pp. 171–178, Oct. 1979.

[113] B. Abidi, Y. Zheng, A. Gribok, and M. Abidi, "Screener evaluation of pseudo-colored single energy X-ray luggage images," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR) Workshops*, Sep. 2005, p. 35.

[114] D. Mery, D. Saavedra, and M. Prasad, "X-ray baggage inspection with computer vision: A survey," *IEEE Access*, vol. 8, pp. 145620–145633, 2020.

[115] D. Mery, *Computer Vision for X-Ray Testing*, vol. 10. Cham, Switzerland: Springer, 2015.

[116] M. Caldwell and L. D. Griffin, "Limits on transfer learning from photographic image data to X-ray threat detection," *J. X-Ray Sci. Technol.*, vol. 27, no. 6, pp. 1007–1020, Jan. 2020.

[117] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Sep. 2010.

[118] M. Everingham and J. Winn, "The PASCAL visual object classes challenge 2012 (VOC2012) development kit," Pattern Anal. Stat. Model. Comput. Learn., Tech. Rep., 2012, pp. 1–45.

[119] M. Everingham and J. Winn, "The PASCAL visual object classes challenge 2007 (VOC2007) development kit," Univ. Leeds, Leeds, U.K., Tech. Rep., 2007.

[120] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[121] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[122] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2014, pp. 740–755.

[123] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, A. Kolesnikov, T. Duerig, and V. Ferrari, "The open images dataset V4," *Int. J. Comput. Vis.*, vol. 128, no. 7, pp. 1956–1981, 2020.

[124] M. Akbari, A. Banitalebi-Dehkordi, and Y. Zhang, "EBJR: Energy-based joint reasoning for adaptive inference," 2021, *arXiv:2110.10343*.

[125] NeelBhowmik. (2022). *X-Ray Datasets*. [Online]. Available: https://github.com/NeelBhowmik/xray

[126] (2022). *FSOD EDS*. [Online]. Available: https://github.com/DIG-Beihang/XrayDetection#x-ray-fsod

[127] LPAIS. (2022). *Xray-Pi*. [Online]. Available: https://github.com/LPAIS/Xray-PI

[128] (2022). *Pixray*. [Online]. Available: https://github.com/Mbwslib/DDoAS

[129] (2022). *CLCXray*. [Online]. Available: https://github.com/GreysonPhoenix/CLCXray

[130] (2022). *HiXray*. [Online]. Available: https://github.com/DIG-Beihang/XrayDetection

[131] N. Bhowmik, Y. F. A. Gaus, and T. P. Breckon, "On the impact of using X-ray energy response imagery for object detection via convolutional neural networks," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2021, pp. 1224–1228.

[132] LPAIS. (2022). *Pidray*. [Online]. Available: https://github.com/bywang2018/security-dataset

[133] M. Naji, A. Anaissi, A. Braytee, and M. Goyal, "Anomaly detection in X-ray security imaging: A tensor-based learning approach," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2021, pp. 1–8.

[134] B. K. S. Isaac-Medina, C. G. Willcocks, and T. P. Breckon, "Multi-view object detection using epipolar constraints within cluttered X-ray security imagery," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 9889–9896.

[135] LPAIS. (2022). *OPIXray*. [Online]. Available: https://github.com/OPIXray-author/OPIXray

[136] (2022). *SIXray*. [Online]. Available: https://github.com/MeioJane/SIXray

[137] (2022). *Compass-XP*. [Online]. Available: https://zenodo.org/record/2654887/#%5C.YUtGVHVKikA

[138] K. J Liang, J. B. Sigman, G. P. Spell, D. Strellis, W. Chang, F. Liu, T. Mehta, and L. Carin, "Toward automatic threat recognition for airport X-ray baggage screening with deep convolutional object detection," 2019, *arXiv:1912.06329*.

[139] LPAIS. (2022). *GDXray*. [Online]. Available: https://domingomery.ing.puc.cl/material/gdxray/

[140] D. Mery, V. Riffo, U. Zscherpel, G. Mondragón, I. Lillo, I. Zuccar, H. Lobel, and M. Carrasco, "GDXray: The database of X-ray images for nondestructive testing," *J. Nondestruct. Eval.*, vol. 34, no. 4, pp. 1–12, 2015.

[141] Y. Wei, R. Tao, Z. Wu, Y. Ma, L. Zhang, and X. Liu, "Occluded prohibited items detection: An X-ray security inspection benchmark and de-occlusion attention module," in *Proc. 28th ACM Int. Conf. Multimedia*, Oct. 2020, pp. 138–146.
[142] M. E. Kundegorski, S. Akçay, M. Devereux, A. Mouton, and T. P. Breckon, "On using feature descriptors as visual words for object detection within X-ray baggage security screening," Tech. Rep., 2016.
[143] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "OverFeat: Integrated recognition, localization and detection using convolutional networks," 2013, *arXiv:1312.6229*.
[144] L. Zhang, L. Lin, X. Liang, and K. He, "Is faster R-CNN doing well for pedestrian detection?" in *Proc. Eur. Conf. Comput. Vis.* Springer, 2016, pp. 443–457.
[145] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2016, pp. 21–37.
[146] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 1–9.
[147] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.
[148] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.
[149] Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving into high quality object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6154–6162.
[150] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2117–2125.
[151] K. Chen, W. Ouyang, C. C. Loy, D. Lin, J. Pang, J. Wang, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, and J. Shi, "Hybrid task cascade for instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4974–4983.
[152] PaddlePaddle. (2022). *YOLOv6-L (V2.1)*. [Online]. Available: https://github.com/meituan/YOLOv6/releases/tag/0.2.1
[153] C.-Y. Wang, A. Bochkovskiy, and H.-Y.-M. Liao, "Scaled-YOLOv4: Scaling cross stage partial network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 13029–13038.
[154] S. Xu, X. Wang, W. Lv, Q. Chang, C. Cui, K. Deng, G. Wang, Q. Dang, S. Wei, Y. Du, and B. Lai, "PP-YOLOE: An evolved version of YOLO," 2022, *arXiv:2203.16250*.
[155] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, Feb. 2013.
[156] K. Grauman and T. Darrell, "The pyramid match kernel: Discriminative classification with sets of image features," in *Proc. 10th IEEE Int. Conf. Comput. Vis. (ICCV)*, vol. 1, Oct. 2005, pp. 1458–1465.
[157] Z. Liu, H. Hu, Y. Lin, Z. Yao, Z. Xie, Y. Wei, J. Ning, Y. Cao, Z. Zhang, L. Dong, F. Wei, and B. Guo, "Swin transformer v2: Scaling up capacity and resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 12009–12019.
[158] D. Thuan, "Evolution of YOLO algorithm and YOLOv5: The state-of-the-art object detention algorithm," Tech. Rep., 2021.
[159] G. Chaucer. (2022). *YOLOU: United, Study and Easier to Deploy*. [Online]. Available: https://github.com/jizhishutong/YOLOU
[160] PaddlePaddle. (2022). *YOLOSeries*. [Online]. Available: https://github.com/nemonameless/PaddleDetection_YOLOSeries
[161] Iscyy. (2022). *YOLOAir: Makes Improvements Easy Again*. [Online]. Available: https://github.com/iscyy/yoloair
[162] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU loss: Faster and better learning for bounding box regression," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 7, pp. 12993–13000.
[163] D. Misra, "Mish: A self regularized non-monotonic activation function," 2019, *arXiv:1908.08681*.
[164] Y. Fang, W. Wang, B. Xie, Q. Sun, L. Wu, X. Wang, T. Huang, X. Wang, and Y. Cao, "EVA: Exploring the limits of masked visual representation learning at scale," 2022, *arXiv:2211.07636*.
[165] P. Gao, T. Ma, H. Li, Z. Lin, J. Dai, and Y. Qiao, "ConvMAE: Masked convolution meets masked autoencoders," 2022, *arXiv:2205.03892*.
[166] W. Wang, J. Dai, Z. Chen, Z. Huang, Z. Li, X. Zhu, X. Hu, T. Lu, L. Lu, H. Li, X. Wang, and Y. Qiao, "InternImage: Exploring large-scale vision foundation models with deformable convolutions," 2022, *arXiv:2211.05778*.
[167] J.-B. Cordonnier, A. Loukas, and M. Jaggi, "On the relationship between self-attention and convolutional layers," 2019, *arXiv:1911.03584*.
[168] S. Li, X. Chen, D. He, and C.-J. Hsieh, "Can vision transformers perform convolution?" 2021, *arXiv:2111.01353*.
[169] B. Yang, L. Wang, D. Wong, L. S. Chao, and Z. Tu, "Convolutional self-attention networks," 2019, *arXiv:1904.03107*.
[170] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, and J. Tang, "Self-supervised learning: Generative or contrastive," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 1, pp. 857–876, Jan. 2023.
[171] D. Mery and C. Pieringer, *Computer Vision for X-Ray Testing*, 2nd ed. Cham, Switzerland: Springer, 2021.
[172] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 1–9.

JIAJIE WU received the M.S. degree in computer science from the Shanxi University of Finance and Economics, Taiyuan, China, in 2017. He is currently pursuing the Ph.D. degree in computer science with Hangzhou Danzi University. He focuses on the field of object detection and X-ray imaging security detection.

XIANGHUA XU received the Ph.D. degree in computer science from Zhejiang University, Hangzhou, China, in 2005. He is currently a Professor with the School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou. He has authored or coauthored over 100 peer-reviewed journals and conference papers. His current research interests include computer vision and parallel and distributed computing. He was a recipient of the Best Paper Award at the 2012 IEEE International Symposium on Workload Characterization.

JUNYAN YANG received the B.S. degree in civil engineering from Beijing Forestry University, Beijing, China, in 2021. He is currently pursuing the M.S. degree in computer science with Hangzhou Danzi University. He focuses on the field of computer vision and X-ray imaging security detection.