ISSN: 2347-6540

International Journal of Latest Research In Engineering and Computing (IJLREC)
Volume 11, Issue 1, Page No.6-12, 2024
www.ijlrec.com

OBJECT DETECTION USING YOLO: CHALLENGES, ARCHITECTURAL AND APPLICATIONS

Himanshu Singh[1], Nitin Shah[2], Ankit Singh Chauhan[3], Abhay Rawat[4]
Computer Science, Shivalik college of engineering, Dehradun, Uttarakhand, India[1,2,3,4]
singhhimanshu27803@gmail.com[1],nitinshah9894@gmail.com[2] ,ankitchauahan02@gmail.com[3], abhayrawat9389@gmail.com[4]

Abstract :- Object identification, an essential aspect of computer vision, has made notable progress due to the emergence of deep learning methods. YOLO (You Only Look Once) is a real-time object identification system recognized for its fast performance and precision. This abstract thoroughly summarizes YOLO, emphasizing its structure, training procedure, and uses. The YOLO architecture utilizes a single convolutional neural network (CNN) to predict bounding boxes and class probabilities at the same time, allowing for quick and accurate object localization. Training YOLO requires improving a predetermined loss function that merges localization and classification mistakes. The model's structure has been revised several times, with each new version fixing flaws and improving efficiency. YOLO has been widely used in several fields, such as driverless cars, surveillance, and healthcare. Its real-time operation makes it ideal for situations that need quick decision-making. Nevertheless, difficulties like managing tiny objects and occlusions remain, leading to continuous research endeavors to enhance YOLO and broaden its functionalities. This article is an introductory guide for scholars and practitioners who want to learn about the concepts and uses of YOLO in object detection tasks.

Keywords: Object Detection, YOLO, CNN

I. INTRODUCTION

Object detection is crucial in computer vision for many applications, such as autonomous driving, surveillance, picture retrieval, and augmented reality [1]. Object recognition traditionally depended on manually designed features and conventional machine learning techniques, which often faced challenges adapting to various datasets and encountered computational inefficiencies. Deep learning, specifically convolutional neural networks (CNNs), transformed the discipline by allowing direct end-to-end learning from raw visual input. YOLO is a leading object identification technique in deep learning, and it is known for its fast speed and high accuracy, among many others. YOLO takes a distinctive method by treating object recognition as a regression issue, where a single neural network predicts bounding boxes and class probabilities for all items in each picture at once. This architecture guarantees real -time performance and streamlines the pipeline, enhancing efficiency and reducing the likelihood of mistakes.
This study offers a detailed examination of YOLO, including its structure, training approach, and uses in many fields. We analyze the progression of YOLO over its several iterations, emphasizing enhancements and advancements included in each update. We also investigate the difficulties and constraints encountered by YOLO, such as dealing with tiny objects, occlusions, and size fluctuations, and review current research efforts focused on resolving these concerns. This study aims to provide academics, practitioners, and fans with a comprehensive knowledge of the fundamental concepts of YOLO and its importance in improving object detection. We want to stimulate more research and innovation in computer vision by clearly outlining its strengths, shortcomings, and prospective applications.

1.2 Object classification and localization

Object categorization and localization are critical challenges in computer vision, crucial for applications like autonomous driving, picture retrieval, and scene interpretation. Item classification identifies an item's category in an image, whereas object localization predicts the bounding box coordinates to outline the object's geographic extent accurately. These activities were considered distinct issues in the past, as object localization often depended on manually designed features and rule-based techniques to identify object borders [8]. Deep learning, namely convolutional neural networks (CNNs), has transformed the traditional approach by allowing simultaneous end -to- end learning of classification and localization problems. This article uses deep learning to explore ideas, methodology, and developments in object categorization and localization. We investigate how CNN-based structures have been modified and improved to tackle the difficulties associated with these tasks, such as size change, occlusion, and background clutter. A key method in this field is the Region-Based Convolutional Neural Network (R-CNN) family of techniques. These algorithms first identify regions containing objects before classifying them. Later versions like Faster R-CNN [9], and Mask R-CNN [10] have enhanced the process, significantly improving efficiency and precision. The Single Shot Multibox Detector (SSD) [11] and You Only Look Once (YOLO) series are significant paradigms that treat object recognition as a regression issue. They directly predict bounding box coordinates and class probabilities in one network run. These models provide exceptional real-time performance and are extensively used in applications requiring quick inference. We also examine the significance of benchmark datasets and evaluation criteria in objectively analyzing the performance of object categorization and localization algorithms. Datasets like COCO [12], PASCAL VOC [13], and ImageNet [14] have been essential in advancing advancement by offering uniform standards for comparison. This paper aims to provide a thorough overview of the latest approaches, obstacles, and future paths in object categorization and localization via deep learning. We want to clarify the fundamental ideas and methods to encourage more research and innovation in the crucial field of computer vision, leading to the development of more resilient and adaptable vision systems.

1.3 How YOLO Algorithm works

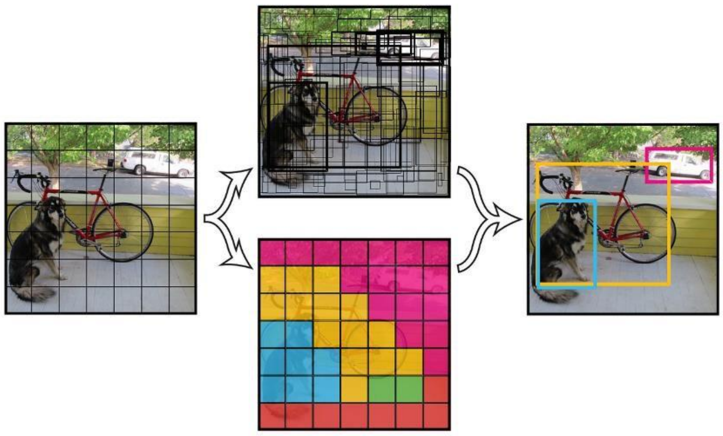

Figure 1: Working of YOLO

The YOLO algorithm is a regression-based method. The model forecasts the likelihood of item classes and provides bounding boxes that indicate the object's precise placement over the whole picture. The object's bounding boxes are defined by the coordinates bx and by, which indicate the centre of the box concerning the grid cell boundaries. The variables bw and bh reflect the width and height of an object concerning the whole picture, while the variable c denotes the item's class. YOLO processes the picture by partitioning it into a grid of dimensions S x S (3 x 3). Next, picture categorization and object localization methods are used on individual grids inside the image, assigning a label to each grid. The YOLO algorithm scans each grid to detect objects, determining their labels and bounding boxes. A grid without an item is labelled as zero. Each labelled grid is denoted as S.S and has 8 values: pc, bx, by, bw, bh, c1, c2, c3. The computer displays if a certain grid contains an item or not. If an object is present, the PC is given a value of
1; otherwise, it is allocated a value of 0. The parameters bx, by, bh, and bw represent the bounding box of a grid and are specified only when a valid item is present inside that grid. c1, c2, and c3 are classes. If the item is an automobile, then the values of c1, c2, and c3 are 0, 1, and 0 accordingly.

1.4 Challenges in object detection:

Using YOLO (You Only Look Once) for object detection offers many advantages, such as real-time processing [15] and detecting multiple objects [16] in a single pass. However, it also presents several challenges:

1. Accuracy vs. Speed Tradeoff: YOLO sacrifices some accuracy for speed. While it's faster than many other object detection algorithms, this can sometimes result in missed detections or misclassifications, especially for small or closely packed objects.
2. Localization Errors: YOLO sometimes struggles with accurately localizing objects, particularly when small, occluded, or clustered closely together. This can lead to bounding box errors where the predicted box does not tightly fit the object.
3. Limited Detection of Small Objects: Due to its single-pass nature, YOLO may have difficulty detecting small objects, especially if they are significantly smaller than the grid cell size used in the detection process.
4. Training Data Quality: Like any deep learning model, YOLO requires much more accurately annotated training data. Poor quality or insufficient training data can lead to subpar performance and generalization issues.

5. Generalization to Unseen Data: While YOLO can perform well on datasets like those it was trained on, its ability to generalize to unseen data, particularly in diverse environments or with uncommon objects, can be limited.
6. Handling Different Object Classes: YOLO may struggle with detecting rare or novel object classes, as it may not have enough examples of these classes in its training data to learn meaningful representations.
7. Fine-tuning and Transfer Learning: While YOLO comes pre-trained on large datasets like COCO, fine-tuning for specific tasks or domains may require additional expertise and effort to achieve optimal performance.
8. Resource Intensive: Training YOLO from scratch or even fine-tuning it on a new dataset can be computationally expensive and time-consuming, requiring powerful GPUs and substantial training time.
9. Hyperparameter Tuning: Like any deep learning model, YOLO has several hyperparameters that need to becarefully tuned to achieve optimal performance. This process can be challenging and may require extensive experimentation.
10. Post-Processing Overhead: While YOLO provides bounding box predictions directly, post-processing steps such as non-maximum suppression (NMS) are typically required to filter out redundant detections, which adds some overhead to the inference process.

Addressing these challenges often involves a combination of careful dataset curation, model tuning, architectural improvements, and post-processing techniques to achieve the desired level of performance for a specific application or use case.

1.4 Types of YOLO:

There are several versions of YOLO (You Only Look Once) developed over time, each with its own improvements and variations. Here are some notable types of YOLO:

1. YOLO v1 (You Only Look Once version 1):
• The original version was proposed by Joseph Redmon et al. in 2016.
• Pioneered the concept of real-time object detection in a single pass through the neural network.
• Divided the input image into a grid and directly predicted bounding boxes and class probabilities from this grid.

2. YOLO v2 (You Only Look Once version 2):
• Introduced by Joseph Redmon and Ali Farhadi in 2017.
• Addressed some of the limitations of YOLO v1, such as localization errors and struggles with small object detection.
• Utilized a different backbone network (Darknet-19) and incorporated anchor boxes to improve bounding box predictions.

3. YOLO v3 (You Only Look Once version 3:
• Released in 2018, YOLO v3 further improved upon its predecessors.
• Introduced several architectural changes, including the use of a variant of Darknet-53 as the backbone network, multi-scale detection, and feature pyramid networks (FPN).
• Achieved better accuracy and performance compared to YOLO v2.

4. YOLOv4:
• It is not an official release by the original authors but a significant community-driven improvement.
• Introduced by Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao in 2020.
• Implemented various enhancements, including the use of CSPDarknet53, improved data augmentation, and multi- level feature aggregation.
• Achieved state-of-the-art performance in terms of accuracy and speed.

5. YOLOv5:
• Developed by Ultralytics and released in 2020.
• Utilizes a different approach compared to previous versions by focusing on a simpler architecture based on the
PyTorch framework.
• Introduced new features such as automatic hyperparameter optimization, model scaling, and advanced data augmentation techniques.
• Gained popularity due to its ease of use and flexibility for customization.

2. YOLO ARCHITECTURE

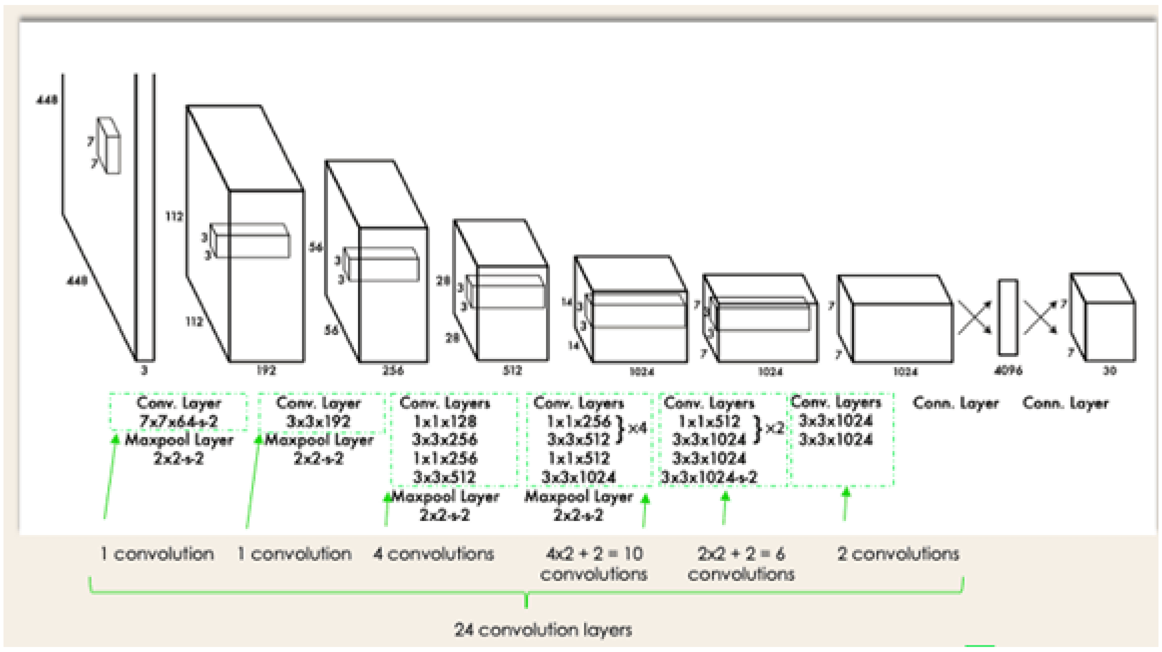
448
7
7
448
3
112
3
3
112
192
56
3
3
56
256
28
3
3
28
512
14
3
3
14
1024
7
3
3
7
1024
7
7
1024
4096
7
7
30
Conv. Layer 7x7x64-s-2 Maxpool Layer 2x2-s-2
Conv. Layer 3x3x192 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x128 3x3x256 1x1x256 3x3x512 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x256 3x3x512 }x4 1x1x512 3x3x1024 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x512 3x3x1024 }x2 3x3x1024 3x3x1024-s-2
Conv. Layers 3x3x1024 3x3x1024
Conn. Layer
Conn. Layer
1 convolution
1 convolution
4 convolutions
4x2 + 2 = 10 convolutions
2x2 + 2 = 6 convolutions
2 convolutions
24 convolution layers

Figure 2: Architecture of the YOLO [18]

The YOLO (You Only Look Once) architecture revolutionized object detection with its real-time processing capabilities. Unlike traditional object detection methods that apply a classifier to various regions of an image, YOLO [19] views object detection as a regression problem to spatially separated bounding boxes and associated class probabilities in a single inference pass.

Key Components:

1. Input Processing: YOLO divides an input image into an S × S grid. Each grid cell predicts a fixed number of bounding boxes and their corresponding confidence scores and class probabilities.
2. Feature Extraction: A convolutional neural network (CNN), often based on Darknet, extracts feature from the input image. The feature maps capture semantic information about objects at different scales.
3. Bounding Box Prediction: Each grid cell predicts bounding boxes using regression, indicating the location of objects. These bounding boxes are represented as (x, y, w, h), where (x, y) denotes the center coordinates of the box relative to the grid cell, and (w, h) represents the width and height of the box relative to the entire image.
4. Class Prediction: YOLO predicts class probabilities for each bounding box. It assigns a confidence score to each bounding box, representing the likelihood of containing an object and how accurate the box is in terms of localization.
5. Non-max Suppression: To eliminate duplicate detections, YOLO employs non-maximum suppression. It removes low-confidence detections and selects the most confident ones among overlapping bounding boxes.

Limitations:

1. Localization: Accuracy: YOLO may struggle with precisely localizing small objects or objects with intricate shapes.
2. Difficulty with Small Objects: Detecting small objects in high-resolution images can be challenging for YOLO due to the coarse feature maps.
3. Handling of Overlapping Objects: YOLO may struggle with accurately detecting overlapping objects, especially if their bounding boxes intersect significantly. Overall, the YOLO architecture offers a compelling solution for object detection in various scenarios, effectively balancing speed, and accuracy.

3. POPULAR DATASET AND CHARACTERISTICS

The MicroSoft Common Objects in Context (MSCOCO) dataset is one of the standards and most popular datasets for computer vision tasks. This dataset was primarily designed to experiment with image/object classification, detection, and instance

segmentation tasks using ML/DL-based approaches. This dataset comes with fewer categories; however, it comprises more instances in each category. Specifically, it includes 91 categories of objects like person, dog, train, and other commonly encountered objects. In addition to the large number of instances in each category, multiple instances with different characteristics are observed per image [17]. Pascal Visual Object Classes (Pascal VOC) [13] is another benchmarking dataset for visual object classification, segmentation, and detection. The dataset community has constantly contributed every year, starting with 4 classes in 2005 and continuing to 20 classes in 2007, making it competitive with recent advancements. The various classes of Pascal VOC are presented in Table 1. Approximately
11,530 images are in the training dataset, containing 27,540 Region of Interests (RoI) and 6929 segmentations. Over recent years, the commonly used metric for evaluation in object detection has been average precision (AP), which can be defined as the average detection precision under various recalls and evaluated in a class-specific manner. To compare the performance of all the categories of objects, an average of all the object categories, i.e., mean Average Precision (mAP) is used as a final metric for evaluation in the object detection and related fields

Table 1: PASCAL VOC classes

Vehicles	Household	Animals
Bus	Potted Plant	Dog
Aeroplane	Bottle	Bird
Bicycle	Chair	Cat
Bus	Plant	Dog
Motorbike	TV	Sheep
Train		

4. CONVOLUTIONAL NEURAL NETWORKS AND PRE-TRAINED MODELS

Conventional machine learning methods depend entirely on manually constructed feature extraction and subsequent feature selection to perform prediction or classification tasks. Typically, these algorithms dedicate significant effort to selecting the optimal feature extraction approach. Researchers, manufacturers, and academics actively engage in deep learning to address these limitations. Popular deep learning techniques/models comprise deep neural networks, Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN) and their architectural variations like Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), Generative Adversarial Networks (GAN), and various types of autoencoders. CNN is often used to efficiently collect spatial and contextual information with reduced parameters because of the inefficiency of scaling in deep neural networks.

Popular deep learning techniques/models comprise deep neural networks, Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN) and their architectural variations like Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), Generative Adversarial Networks (GAN), and various types of autoencoders. CNN is often used to efficiently collect spatial and contextual information with reduced parameters because of the inefficiency of scaling in deep neural networks. Fundamental transformation unit that converts an input volume into an output volume. The spatial extent of the convolution process is a hyperparameter referred to as a receptive field or filter size. The filter size used for convolution on the input is crucial for obtaining valuable feature information. The size of the output volume is determined by other hyperparameters, including depth, stride, and zero padding [20]. A dot product is calculated between the weights (w) and the spatial input (x) in neural networks, followed by the addition of a bias term (b) and the application of a non-linear activation function (f). The convolution procedure for textual and visual input may b e represented by the following equations.

The output of the ith neuron in layer l is represented by y l. The filter size in textual input is denoted by d, while the filter width and filter height in visual input are represented by d1 and d2, respectively.

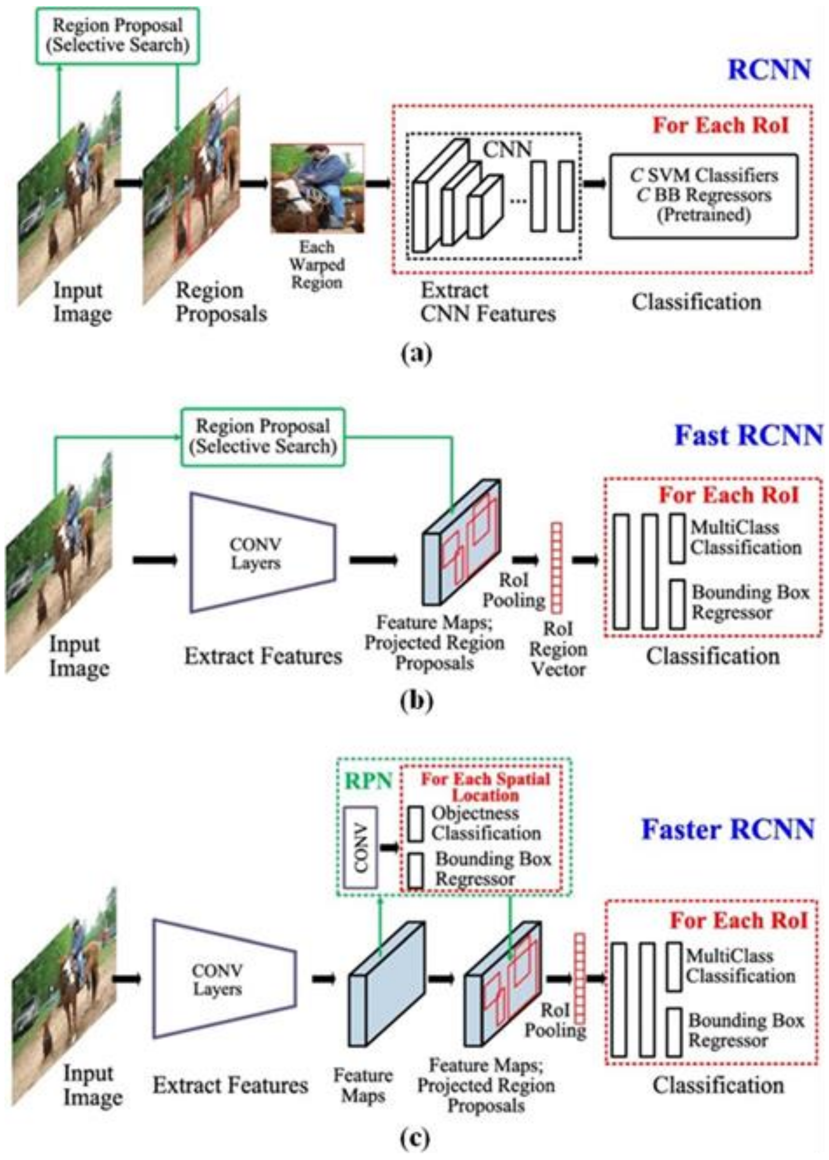

Figure 3: Object detectors with two stages (a) RCNN (b) Fast-RCNN (c) Faster-RCNN

The architectural architecture of CNNs includes convolutional, pooling, and Rectified Linear Units (ReLU), working together as a

5. CONCLUSION AND FUTURE SCOPE

This paper explores the basics of CNN algorithms and gives an overview of YOLO's real-time object detection method. CNN models are really good at spotting objects and dealing with image quirks like glare. They're super useful for things like finding defects or creating educational apps. YOLO stands out from other CNN algorithms for a few reasons. It's easy to set up and train because of its simple loss function, and it can train the whole model at once, which is pretty cool. YOLOv2, the second version, is especially impressive because it's really fast but still accurate at detecting objects.

Overall, YOLO is great at recognizing different objects, and it's a top choice for real-time object detection. This suggests that the future of object detection is bright and full of possibilities.

REFERENCES

[1] P. Viola and M. Jones, "Robust real-time object detection", International Journal of Computer Vision, vol. 4, pp. 34-47, 2001

[2] P. Venkateswari, E. Jebitha Steffy and N. Muthukumaran, "License Plate cognizance by Ocular Character Perception", International Research Journal of Engineering and Technology, vol. 5, no. 2, pp. 536-542, February 2018.

[3] Liang M, Hu X (2015) Recurrent convolutional neural network for object recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3367–3375.

[4] T. Gao Gould and D. Koller, "Region-based segmentation and object detection", Advances in neural information processing systems, pp. 655-663, 2009.

[5] B. Renuka, B. Sivaranjani, A. Maha Lakshmi and N. Muthukumaran, "Automatic Enemy Detecting Defense Robot by using Face Detection Technique", Asian Journal of Applied Science and Technology, vol. 2, no. 2, pp. 495-501, April 2018.

[6] A. Aruna, Y. Bibisha Mol, G. Delcy and N. Muthukumaran, "Arduino Powered Obstacles Avoidance for Visually Impaired Person", Asian Journal of Applied Science and Technology, vol. 2, no. 2, pp. 101-106, April 2018.

[7] Mohamed Aladem et al., "A Combined Vision-Based Multiple Object Tracking and Visual Odometry System", IEEE Sensors Journal, vol. 19, 2019.

[8] Thai LH, Hai TS, Thuy NT (2012) Image classification using support vector machine and artificial neural network. Int J Inform Technol Comput Sci 4(5):32–38

[9] Shaoqing Ren et al., "Faster R-CNN: Towards real time object detection with region proposal networks", IEEE Transactions on pattern analysis and machine intelligence, 2016.

[10] Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 580–587

[11] Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, Berg AC (2016) Ssd: single shot multibox detector. European Conf Comput Vis 2016:21–37.

[12] Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL (2014) Microsoft coco: common objects in context. In: European Conf Comput Vis, pp 740–755.

[13] Everingham M, Eslami SA, Van Gool L, Williams CK, Winn J, Zisserman A (2015) The pascal visual object classes challenge: a retrospective. Int J Comput Vis 111(1):98–136

[14] Wang CY, Mark Liao HY,Wu YH, Chen PY, Hsieh JW, Yeh IH (2020) CSPNet: a new backbone that can enhance learning capability of CNN. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, pp 390–391

[15] Agarwal S, Terrail JO, Jurie F (2018) Recent advances in object detection in the age of deep convolutional neural networks. arXiv preprint arXiv:1809.03193.

[16] Rather AM, Agarwal A, Sastry VN (2015) Recurrent neural network and a hybrid model for prediction of stock returns. Expert Syst Appl 42(6):3234–3241

[17] Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL (2014) Microsoft coco: common objects in context. In: European Conf Comput Vis, pp 740–755.

[18] Choi H, Ryu S, Kim H (2018) Short-term load forecasting based on ResNet and LSTM. In IEEE international conference on communications, control, and computing Technologies for Smart Grids (SmartGridComm), pp 1-6.

[19] Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: unified, real-time object detection. In proceedings of the IEEE conference on computer vision and pattern recognition, pp 779-788

[20] Bengio Y, Courville AC, Vincent P (2012) Unsupervised feature learning and deep learning: a review and new perspectives. CoRR, abs/1206.5538, 1(2665).