IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 6, 2021, accepted November 8, 2021, date of publication November 23, 2021,
date of current version December 2, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3130129

Multi-Level Refinement Feature Pyramid Network for Scale Imbalance Object Detection

LUBNA AZIZ[1,2], MD SAH BIN HAJI SALAM[1], (Member, IEEE), USMAN ULLAH SHEIKH[3], (Senior Member, IEEE), SURAT KHAN[4], HUMA AYUB[5], AND SARA AYUB[3,4]

[1]Division of Artificial Intelligence, Faculty of Engineering, School of Computing, Universiti Teknologi Malaysia (UTM), Skudai, Johor 81310, Malaysia
[2]Department of Computer Engineering, Faculty of Information and Communication Technology, Balochistan University of Information Technology, Engineering and Management Sciences (BUITEMS), Quetta 87300, Pakistan
[3]Faculty of Engineering, School of Electrical Engineering, Universiti Teknologi Malaysia (UTM), Skudai, Johor 81310, Malaysia
[4]Department of Electrical Engineering, Faculty of Information and Communication Technology, Balochistan University of Information Technology, Engineering and Management Sciences (BUITEMS), Quetta 87300, Pakistan
[5]Department of Chemistry and Technology, Sardar Bahadur Khan Woman University, Quetta 86301, Pakistan

Corresponding authors: Lubna Aziz (engr.lubnaaziz@gmail.com) and Usman Ullah Sheikh (usman@fke.utm.my)

ABSTRACT Object detection becomes a challenge due to diversity of object scales. In general, modern object detectors use feature pyramid to learn multi-scale representation for better results. However, current versions of feature pyramid are insufficient to handle scale imbalance, as it is inefficient to integrate semantic information across different scales. Here, we reformulate feature pyramid construction as a feature reconfiguration process. We propose a detection network, Multi-level Refinement Feature pyramid Network, to combine high-level features (i.e., semantic information), middle-level feature and low-level feature (i.e., boundary information), in a highly-nonlinear yet efficient manner. A novel contextual features module is proposed, which consists of global attention and local reconfigurations. It efficiently gathers task-oriented contextual features across different scales and spatial locations (i.e., lightweight local reconfiguration and global attention). To evaluate significance of proposed model, we designed and trained end-to-end single stage detector called MRFDet by assimilating it into Single Shot Detector (SSD), and it achieved better detection performance compared to most recent single-stage objects detectors. MRFDet achieves an AP of 45.2 with MS-COCO and an improvement in *mAP* of 4.5% with VOC.

INDEX TERMS Object detection, feature pyramid, convolutional neural network, computer vision.

I. INTRODUCTION

Object detection becomes more challenging as the scale of object instances varies [1]–[3]. According to our best information so far, two strategies have been devised to resolve arising issues by this challenge. In the first strategy, the image pyramid is used to detect objects (i.e., a series of different sizes of input images) [1]. Due to computational complexity and increased memory requirements, this solution can only be exploited during testing. Thus, it dramatically drops the efficiency of the detector. The second strategy is based on the feature pyramid developed from the input image used for object detection [3], [4]. It can be exploited at both phases (testing and training phase) due to low memory requirements and computational cost. Furthermore, ''the feature pyramid module'' can be effectively incorporated into deep networks to create an end-to-end solution. However, the object detector based on pyramidal construction [3]–[6] yields promising results. But there are still some limitations due to the generation of feature pyramid from the intrinsic multi-scale pyramidal architecture of the backbone (that design for object classification task).

The feature pyramid models have two main limitations. First, single-level layers of the backbone network (i.e., designed for classification task) are used to generate feature maps that are not sufficiently descriptive for the object detection task. Second, a multi-level feature pyramid can produce a more descriptive feature-map, but it adds significant computational complexity. Primarily, the low-level features that are extracted from shallow layers are not very descriptive but helpful in object localization task. Moreover, the extraction of high-level features from deeper layers can be useful for the classification task. High-level features are appropriate for objects with intricate presence, while low-level features are

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

suitable for the objects with an uncomplicated appearance. In general, objects with similar sizes having different appearances, such as the appearance of a remote person, is more complicated than that of a similar traffic light. Thus, each feature-map in the pyramid based on single-level information can yield sub-optimal results (i.e., used for a specific range of object sizes).

The intuition behind this work is the use of information from the middle layers (i.e., it is expected to describe the mid-level representation of object parts and retain the spatial information as well) along with shallow and deep-level features. Medium-level features are necessary not only for initial low-level convolutional layers features (that encode basic visual geometry such as edges, circles, lines, corners) but also for high-level features that encode the high-level information used for object detection (category-level evidence). It is advantageous to have features of all levels for object recognition. The higher-level features are utilized for classification of the object, while low-level features are helpful for object localization. The most effective method to use middle-level features for object detection is still an open question.

This research work aims to build a more powerful feature pyramid with multiple levels for recognizing object instances of various scales with less computational effort, in order to avoid the previously mentioned constraints of existing methods. As depicted in Fig. 2, to accomplish this objective, we initially merged features from multiple layers (i.e., multi-level features) extracted through a backbone network such as VGG-16 and base features, and then fed them into a block of alternating residual standard convolution unit (RCU) to get more representative, multi-scale/level features. At this point, we compile the feature map of the same scales to develop the ultimate feature pyramid. Finally, the constructed feature pyramid is passed through Contextual features module to capture contextual information from a vast image region. In addition, each feature map contains layered information in the resulting feature pyramid. For this reason, we call our proposed network for building pyramids MRFPN (Multi-Level Refinement Feature Pyramid Network).

In this paper, a practical end-to-end single stage detector is designed and trained to assess the significance of our proposed Multi-level Refinement Feature Pyramid network. We call our model MRFDet (Multi-level Refinement Feature Detector) as it is constructed upon multi-level and multi-scale features network (MRFPN) integrate with the architecture of SSD [4]. MRFDet achieved the significant result (i.e., an Average Precision of 45.2), outperforming one-stage detectors on MS-COCO [7] and improvement of 4.5% in *mAP* PASCAL VOC07/12 benchmark datasets. The main contributions of this work are summarized as follows:

1) We proposed a multi-level refinement feature pyramid network (MRFPN) for object detection with less computational complexity. It exploits the features from multiple levels and recursively refines the shallow features to generate a middle level and more in-depth feature maps.
2) For the first time, Chained Parallel Pooling has been used during the construction phase of the feature pyramid to introduce more robustness and able to capture the contextual information from a vast image region, followed by prediction layers for object detection. For this purpose, the features are efficiently pool with several window sizes and merged with learnable weights and residual connections.
3) These design features result in extensive training and significant recognition performance; even input images are not high-resolution images, further improving the tradeoff between accuracy and speed.
4) With qualitative and quantitative observations, we prove that our MRFPN shows a significant improvement over conventional SSD [4] and M2Det [8]. MRFDet can be used for both datasets; i.e., PASCAL VOC 07/12 and MS COCO achieve state-of-the-art performance.

II. RELATED WORK

The sliding window has a rich and long history of perspective, beginning with the use of convolutional networks to recognize handwritten digits. However, the invention of enhanced object detector [9], integral channel features [10], and the HOG [11] led to more effective methods of face detection and pedestrian detection. The rebirth of deep learning exemplifies the sliding window in the realm of classic computer vision. In this section, we mainly discuss the difference and similarities of our model and some previous works in details.

A. ANCHOR-BASED OBJECT DETECTION

Anchor-based object detection framework further categories into two clusters: two-stage approaches with proposal determined and one-stage proposal free approaches. The **two-stage approach** in present-day object detection is a dominant metaphor. The Selective-Search [12] is a pioneering approach that comprises of two stages. In the first phase, a spare set of candidate proposal is generated that includes all objects, while the negative-locations are filtered. In contrast, the classification of background and foreground classes performed in the second stage. R-CNN [13] achieved a significant gain in accuracy by replacing the classifier of the second stage with a convolutional network. RCNN has improved in speed and accuracy over the years [2], [14] by using learned object proposals [15]–[17]. Region Proposal Networks (RPN) combines second stage classifier with proposal generation into a single convolutional network such as Faster R-CNN [16]. Various research works have been proposed to enhance its performance, including redesigning and reforming of architecture [3], [6], [18]–[20], attention and context mechanism [21], modified strategies in training and loss function [19], [22], feature fusion enrichment and enhancement [23], [24]. For two-stage detectors, the proposal is predicted using anchors as regression references and classification candidates. Such models achieve the highest rate of accuracy but are usually slow.

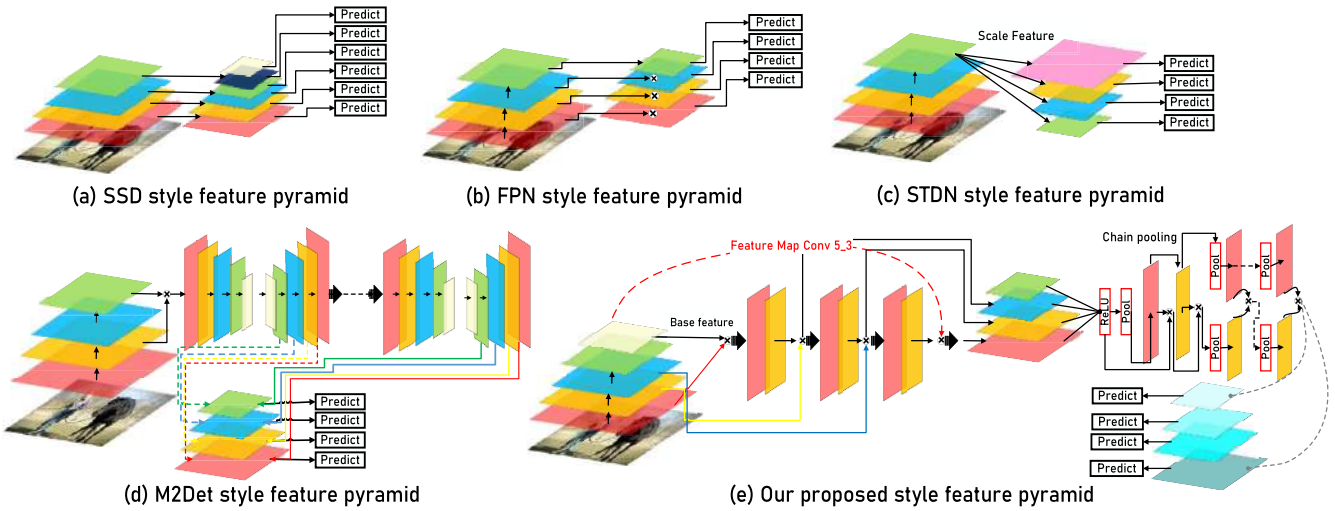

FIGURE 1. Demonstration of the five kinds of feature pyramid that are used in state-of-the-art models, a) SSD-style feature pyramid, b) feature pyramid used in FPN, c) feature pyramid used in STDN, d) feature pyramid used in M2Det, and e) our proposed multi-level multi-scale FPN.

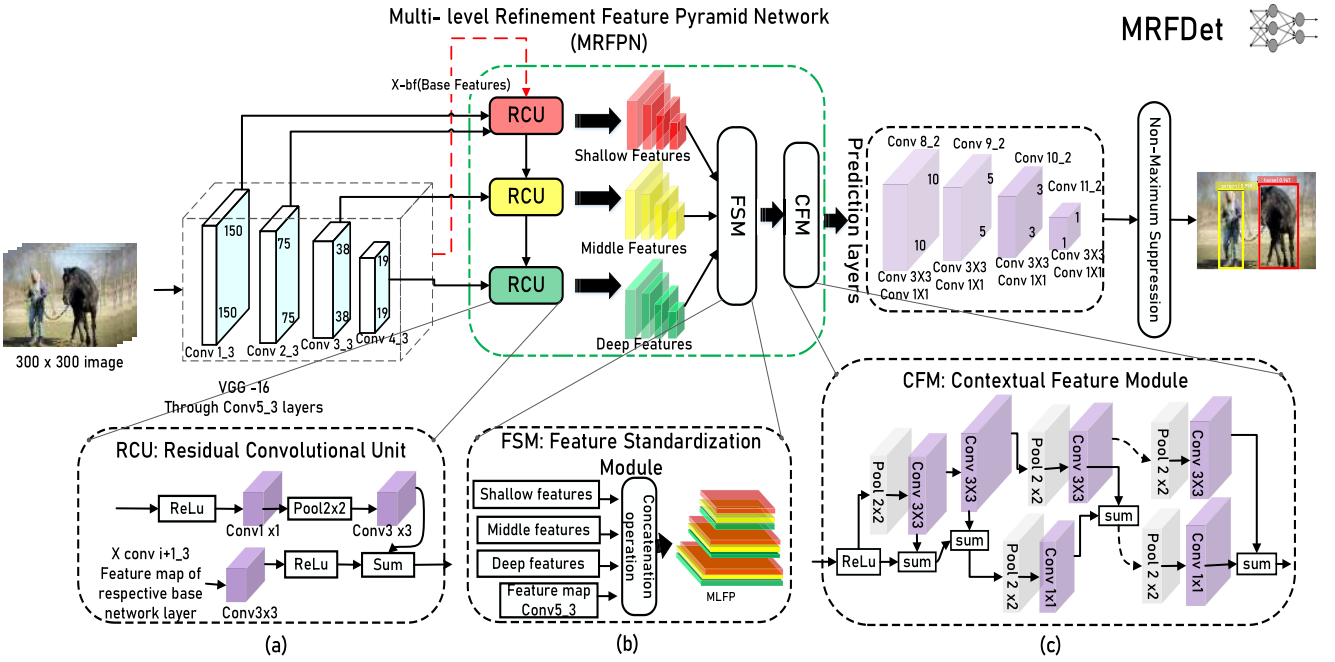

FIGURE 2. An outline of individual parts of our multi-path refinement feature pyramid design MRFDet (300 × 300). MRFDet utilizes the backbone and MLFPN frameworks to extract robust features to detect objects from the input image; it generates class category scores and dense bounding boxes. In MLFPN, the stack of RCU fuses base features and feature maps of corresponding layers of backbone to generate multi-scale multi-level features, and then the FSM module aggregates the features into multi-level multi-scale features pyramid. Finally, parallel pooling applies to increase the effectiveness of features against the appearance complexity variance of instances. And thus, we achieve fruitful end-to-end training of the whole system.

Due to high computational efficiency, **one-stage anchor-based detectors** have attained much attention. OverFeat [25] is the first modern deep learning based one-stage detector. SSD [4], [26] directly predict the object anchor box offset and category by spreading the anchor boxes on multi-scale layers within a ConvNet. Recent development shows that a plenty of work have been proposed to boost its performance in different aspects, such as multi-scale feature fusion [26], [27], training strategies (from scratch) [28], proposed new loss function [5], anchor matching and refinement [29], [30], and enrichment of features [31], [32]. One stage detector uses the anchor as a reference box for final selections.

B. ANCHOR-LESS EXPLORATIONS

The best known anchor-less detector could be YOLOv1 [33] with input image 448 × 448 and output of a 7 × 7 grid cell for the box prediction. YOLOv1 experiences from low recall as it used single point usages for bounding box prediction [34].

As a result, anchors are used to ensure high recall in YOLO9000 [34] and YOLOv3 [35]. Due to difficulties in detecting objects with multiple scales, some of detectors were considered inappropriate for generic object detection [36]. DenseBox (anchorless detector) [37] the image pyramid to detect objects with multiple scales that takes several seconds to process one image. RepPoints [38] uses a deformable convolution to create more precise features and represents an object as a set of sample points. FSAF [39] uses the anchor-free paradigm with the best feature prediction to train each instance. FCOS [36] uses a per-pixel prediction strategy and relies on centerness map to suppress poor-quality object detection. The CenterNet [40] represents each object through its characteristics at the center point. CornerNet [41] uses the Associative Embedding technique [42] to detect the bounding box of an object as a pair of key-points. Cascaded and central pooling is used to improve recall and precision in CornerNet. FoveaBox [43] proposed a technique with which the final class probability can be directly predicted by assigning objects to multiple adjacent pyramid levels.

C. FEATURE PYRAMID NETWORK/MULTI-LEVEL FEATURE PYRAMID

The effective representation of multi-scales features scales in object recognition is always the main hurdle to improving the detection accuracy. Most previous approaches to detection use a pyramid feature hierarchy extracted from backbone networks to make a prediction. As far as we know, two main strategies have been used to deal with scale-variance.

The first strategy is **featuring image pyramids** (i.e., input image with various sizes and resolutions) that is used to produce multi-scale semantic features. These semantic features further are used to separate prediction, and then all participate together to make the ultimate prediction. The feature of multi-scale images significantly improves the accuracy of recognition and localization precision, compared to single-scale images features such as used in [19] and SNIP [1]. Despite the increase in performance, feature image pyramid strategies are unable to gain popularity in the A.I. community and not plausible for real-time applications due to drastically high time and memory requirements. To remedy this problem, SNIP [1] used featured image pyramids only during the testing phase. ''In contrast, other models such as Fast R-CNN [14] and Faster R-CNN [16] did not to use this strategy by default''.

The second strategy is the **feature pyramid generation for object detection**, i.e., extracts feature from multiple layers of the network using a single-scale image. This method is considerably more cost-effective than the image pyramid approach in terms of computing effort and memory requirements and enables FPN to be provided in real-time applications both in training and in the test phase. In addition, it is flexible and fits easily into state-of-the-art detectors based on a deep neural network.'' As one of the pioneering works, Feature Pyramid Network (FPN) [3] has implemented a top-down pathway and side links to generate features pyramid that takes accuracy and speed into account. Following this idea, PANet [44] includes extra bottom-up path aggregation network on the top of FPN; STDL [45] exploits cross-scale features in scale-transfer module, M2Det [8] proposes the U-shape module to fuse the multi-scale feature; and G-FRNet [46] control the information flow across features uses gate units. Most recently, NAS-FPN [47] uses the neural architecture search to automatically design feature network topology''. Thousands of GPU hours are required during search in NAS-FPN and yielding an irregular feature network which is difficult to interpret. EfficientDet [48] proposes a weighted bidirectional feature network with customized compound scaling method for multi-scale feature fusion. Libra R-CNN [49] proposed a balance design comprises of simple components i.e., balanced feature pyramid, balance L1 loss, and IoU-balanced sampling, to solve the imbalance issue existing in the training process. The different flavors of feature pyramid have been shown in Fig. 1. In contrast, the purpose of this work is to understand whether single-phase detectors can match or outdo the precision of a two-phase detector at similar or faster speeds.

III. PROPOSED METHOD

A systematic overview of our proposed framework is shown in Fig. 2. Our framework is based on SSD with VGG-16 as backbone. The VGG-16 backbone network is used to generate base features. In addition, base features and feature maps of corresponding layers are used in construction phase of feature pyramid in multipath refinement feature pyramid network module (MRFPN). It consists of three modules such as Residual Convolutional Unit (RCU), Features Standardization Module (FSM), and Contextual Features Module (CFM)—a detailed description of three core modules with network configurations in MRFDet, as illustrated in the following sections.

1- RCU creates a set of multi-scale features that enrich the semantic information in the base feature and feature map of backbone layers.
2- FSM module assembles the features into a multi-level feature pyramid using a scaled feature chain operation.
3- Finally, CFM uses to capture background contextual information from a large area of the image. It uses multi-window sizes to pool the features and fuse them using learnable weights.

Finally, prediction layers produce dense bounding boxes and categories that are scored on learnable features, followed by non-maximum suppression (NMS) operation to get the final prediction similar to SSD.

A. MULTIPATH REFINEMENT FEATURE PYRAMID NETWORK (MRFPN)

As aforementioned, this scheme is used to generate multi-level feature map in order to detect objects with different scales. This schema generates a multi-level feature pyramid by merging low-level semantic features to medium and high-level features. It has three basic blocks i.e., RCU, FSM, and

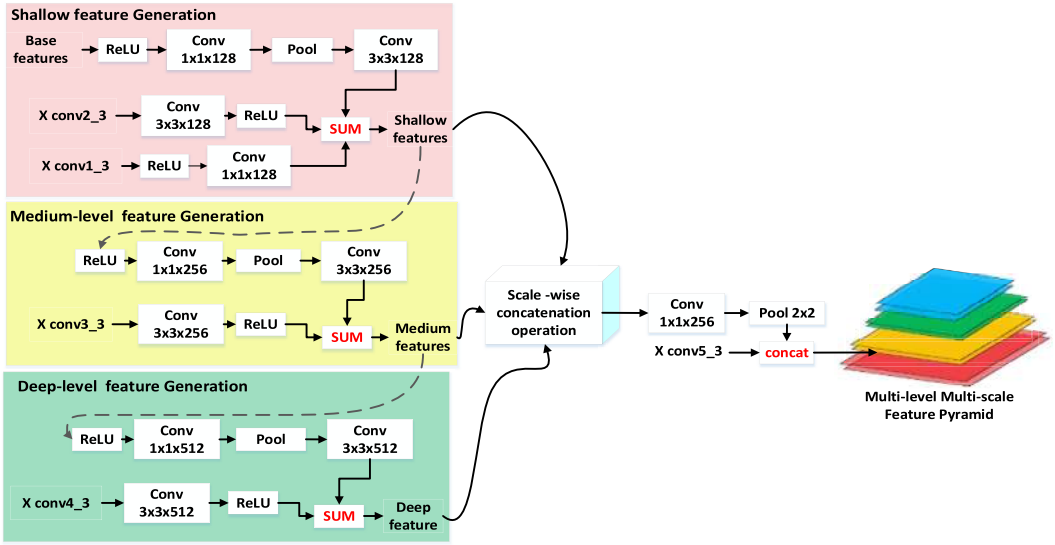

FIGURE 3. Structural detail of residual convolutional unit (RCU) with FSM. Upper block with red color: it generates low-level features and has three branches. Middle block with yellow color: it generates medium-level features and have two branches while bottom block with green color: it generates deep-level features and also have two branches just like second block. Finally, multi-level multi-scales feature pyramid is generated by FSM.

CFM as shown in Fig.2. Firstly, VGG-16 [50] generate based features that include multi-level semantic information for MRFPN. The first block of MRFPN comprises of a stack of RCUs that uses base feature and feature maps of four layers (i.e., Conv1_3 to Conv4_3) of the backbone network (VGG-16) to generates low, medium and high-level feature maps of different scales. In first RCU, base feature and feature map of Conv1_3, Conv2_3 of backbone are used to generates low-level features. While the output of first RCU is combined with feature map of Conv3_3 layer of backbone network in the second RCU to generate medium-level features. Similarly, high-level features are generated using output of second RCU and feature map of Conv4_3 layer of backbone network. Note that the first RCU has no previous knowledge of any other RCU and is therefore only learned from base features (X_{bf}).

The outputs of multi-level/scale features are calculated as:

$$\left\{x_1^l, x_2^l, x_3^l, \ldots\ldots, x_i^l\right\} = \begin{cases} RC_l\left(X_{bf}, X_{Conv(i+1)\_3}, , X_{Convi\_3}\right), & l = 1 \\ RC_l\left(x_i^{l-1}, X_{Conv(i+1)\_3}\right), & l = 2 \ldots L \end{cases} \quad (1)$$

where X_{bf} denotes the base feature, x_i^l denotes the features with the i^{th} scale in the l^{th} RCU, L denotes the number of RCUs, RC_l denotes the l^{th} RCU processing, and $X_{Conv(i+1)\_3}$ denotes a feature map of the $i+1$ layer of the backbone network. The feature standardization module is the second block of feature pyramid network. We used scale-wise concatenation operation to aggregates and up-samples the multi-scale features. Finally, Chained pooling is used to capture contextual background information from a huge image region. End-to-end training is done to efficiently train the entire network efficiently.

B. MRFDet

The architecture of MRFDet is illustrated in Fig. 2. The VGG16 backbone network develops base features that are exploited in the RCU stack to produce multi-scale feature maps. Multi-level feature fusion block concatenates the features map scale-wise to generate the feature pyramid. The prediction layer and NMS operation are applied to MLFP, similar to SSD except chained parallel pooling that is applied at the beginning of the Prediction layers block. It is worth considering here that our architecture is flexible and requires less computational effort, which makes it more convenient to adapt to real-time application.

1) RESIDUAL CONVOLUTIONAL UNIT (RCU)

The residual convolutional unit is first module of MRFPN that uses a set of adaptive convolutions to refine the base features and feature maps of backbone network and generates multi-level features. A optimized version of basic ResNet [18] block without batch normalization layers is used in the RCU. The detailed architecture of RCU is shown in Fig. 3. It has two branches, top branch comprises of two Conv layers with 1×1 and 3×3 filters and stride 2, while non-linearity has been added using pooling and ReLU (R) operations. The other branch has one 1×1 Conv layer, as shown in Fig. 2(a).

Finally, refined feature maps of backbone layers, i.e., X_{Conv1_3} and X_{Conv2_3} merges with output of top branch of RCU and construct low-level features as described in equation 2. While medium-level and deep-level features are built using X_{Conv3_3} and X_{Conv4_3} respectively, and the output of

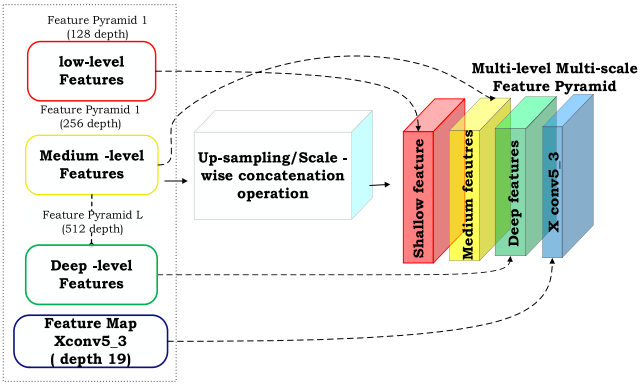

FIGURE 4. Illustration of feature standardization modules. Features with equivalent scaling and channel dimension are concatenated in FSM.

previous RCU block, as describe in equation 3 and equation 4. The corresponding backbone layer filter determines the filter size for each input path. Moreover, an additional Conv-layer 1×1 is used to improve ability to learn and to keep feature smooth [51]. In this way, multi-scale features of current level are generated. RC_1 generates low-level features, RC_2 generates medium-level features while RC_l generates deep level features.

$$X_L = R\left(Conv_{1\times1}\left(X_{Conv1\_3}\right)\right) + R\left(Conv_{3\times3}\left(X_{Conv2\_3}\right)\right) + Conv_{3\times3}\left(pool\left(Conv_{1\times1}\left(R\left(X_{bf}\right)\right)\right)\right) \quad (2)$$

$$X_M = R\left(Conv_{3\times3}\left(X_{Conv3\_3}\right)\right) + Conv_{3\times3}\left(pool\left(Conv_{1\times1}\left(R\left(X_L\right)\right)\right)\right) \quad (3)$$

$$X_H = R\left(Conv_{3\times3}\left(X_{Conv4\_3}\right)\right) + Conv_{3\times3}\left(pool\left(Conv_{1\times1}\left(R\left(X_M\right)\right)\right)\right) \quad (4)$$

2) FEATURES STANDARDIZATION MODULE

The second block of MRFPN is a features standardization module (FSM) that aggregates refine features generated by RCUs in the form of multi-level feature pyramid, shown in Fig. 2 (b) and Fig. 4. Initially, features with equivalent scales are concatenated with channel dimension. The resultant features can be represented as $X = (X_1, X_2, X_3, \ldots, X_i)$, where $X_i = \text{Cancat}\left(x_i^1, x_i^2, x_i^3, \ldots, x_i^L\right) \in \mathbb{R}^{W_i \times H_i \times C}$, refer to the i^{th} concatenated feature set. To generate the feature map with same dimensions, it uses convolutional adaptation and up-sample the smaller feature maps to larger feature maps. Finally, all feature maps are concatenated to generates multi-level feature pyramid containing features of multi-level depths of each scale.

3) CONTEXTUAL FEATURES MODULE

The multi-level feature pyramid is fed to contextual features module (CFM) to generate more robust multi-level contextual features, as shown in Figs. 2(c) and 5. This module is used in place of last two prediction layers of SSD. It captures contextual background information from a huge image region. In order to keep the size of feature map same for addition,

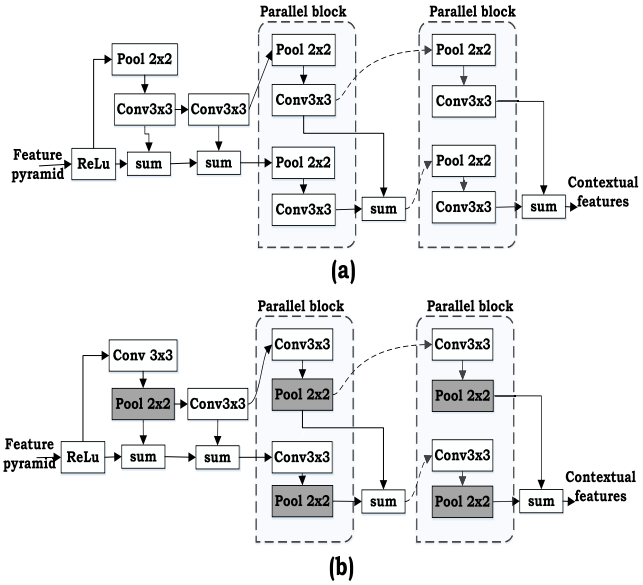

FIGURE 5. (a) Complete architecture of contextual feature module, multi-level feature pyramid is fed to CFM that contains four parallel branches by default, each branch contains Conv layer and pooling layer with different filters (i.e., 3×3 *and* 1×1). (b) Alternative architecture of contextual features module. Compare with Fig. 5 (a), position of pooling layer is exchange with Convolution layers marks with gray color.

padding and chain operation are carried out alternatively. A block of CFM contains two parallel branches of Conv layers and pooling layer. The output of Conv layer of previous block is fed to upper branch of next block, while the sum of output of previous block is fed to lower branch of next block. In contextual features module, we tested several combinations of parallel pooling blocks and observed that best result was obtains with four parallel pooling block combination in CFM [52]. An alternate architecture of chained parallel pooling block is shown in Fig. 5 (b). This alternate architecture is a modified version of the architecture shown in Fig. 5 (a) by interchanging the position of pooling layer and convolution layer in parallel pooling block. The convolution layer adapts to learn the input features and consider their importance before being fed to pooling layer. In our observation, this approach may sometimes perform a little better in some datasets compared to the original architecture.

4) OBJECTIVE LOSS FUNCTION

To handle the different object categories, the MRFDet training objective is derived from the multi-box objective [15], [53]. Let $x_{ij}^p = \{1, 0\}$ be an indicator for the agreement of i^{th} default box with the j^{th} ground truth box of class p. the overall objective loss function is a weighted sum of the loss of localization (loc) and confidence loss (conf):

$$L\left(x, c, p_{box}, g_{box}\right) = \frac{1}{N}\left(L_{conf}\left(x, c\right) + \alpha L_{loc}\left(x, p_{box}, g_{box}\right)\right) \quad (5)$$

where N is the number of matched default boxes, the loss is zero if $N = 0$. Localization loss is L1 smooth loss [54]

between the predicted box p_{box} and ground truth box (g_{box}) parameters. Similar to faster R-CNN [16], offsets regress for the center (cx, cy) of the default bounding box (b_{box}) and for its width (w) and height (h).

$$L_{loc}(x, , p_{box}, g_{box}) = \sum_{i\epsilon pos}^{N} \sum_{m\epsilon\{cx,cy,w,h\}} x_{ij}^{k} smooth_{L1}\left(p_{box}{}_{i}^{m} - \hat{g}_{box}{}_{j}^{m}\right) \tag{6}$$

$$\hat{g}_{box}{}_{j}^{cx} = \frac{\left(g_{box}{}_{j}^{cx} - d_i^{cx}\right)}{d_i^{w}} \quad \hat{g}_{box}{}_{j}^{cy} = \frac{\left(g_{box}{}_{j}^{cy} - d_i^{cy}\right)}{d_i^{h}} \tag{7}$$

$$\hat{g}_{box}{}_{j}^{w} = log\left(\frac{g_{box}{}_{j}^{w}}{d_i^{w}}\right) \quad \hat{g}_{box}{}_{j}^{h} = log\left(\frac{g_{box}{}_{j}^{h}}{d_i^{h}}\right) \tag{8}$$

The softmax loss over multiple classes confidence(c) is called confidence loss.

$$L_{conf}(x, c) = -\sum_{i\epsilon pos}^{N} log\hat{c}_i^{p} - \sum_{i\epsilon neg}^{N} log\hat{c}_i^{o}$$

and

$$\hat{c}_i^{p} = exp\left(c_i^{p}\right) \Big/ \sum_{p} exp\left(c_i^{p}\right) \tag{9}$$

and cross-validation is used to set the weight term α to 1.

5) NETWROK CONFIGURATION

We have assembled MRFDet with the VGG-16 backbone framework. Pre-trained backbone framework, i.e., VGG-16 (trained on ImageNet 2012 dataset [55]), is used to train the entire network. The default configuration of MRFPN contains three RCU, each with two branches except the first RCU; the first branch has two Conv layers of filters (1×1, and 3×3) with stride 2 and non-linearity incorporated through the ReLU and pooling operations, so it produces multi-scale features. The other branch has only a 1×1 Conv layer with a ReLU function. To decrease the parameters numbers, we use the Conv filter size less than 1024 to facilitate network training on the GPU. We use the same input sizes as it was used in the conventional SSD, RefineDet, and Retina Net such as 300, 512.

The last stage of the MRFPN consists of the contextual features module that comprises of parallel pooling unit that forms the chain. The contextual features module output is fed to the original SSD prediction layers as an input. We place six anchors with a total of three ratios for each pixel of pyramidal features. Then a probability rating of 0.05 is set as the threshold to filter out most of the low-scoring anchors. For more accurate boxes, post-processing is performed using soft non-maximum suppression NMS with a linear kernel [56]. Lowering the threshold to 0.01 can yield improve detection results; however, it significantly reduces the inference time. We don't see it as a pursuit of improving practical values. The SGD first uses a learning rate of 10-3, momenta of 0.9, and 0.0005 weight decay and batch size 32 to fine-tune the resulting model. The guidelines for learning rate and weight decay differ slightly for each data set. Complete training and testing code have been developed on TensorFlow.

C. EXPERIMENTAL RESULTS

To demonstrate the effectiveness of our approach, we carry out comprehensive experiments on two benchmark datasets of generic object detection, such as MS-COCO and PASCAL 07/12. PASCAL VOC07/12 includes 20 categories in 9,963 and 22,531 images, respectively. We compare with Fast-RCNN and Faster RCNN [16] in the Pascal data set. The previously trained backbone framework, i.e., VGG-16 is used to fine-tune the model. Localization and class confidences score is predicted using the MRFF block, pooling block, and prediction layer (i.e., Conv8_2, Conv9_2, Conv10_2, and Conv11_2). We set a learning rate of 0.0001 for the initial 5k iterations and afterwards keep training for the next 30k iterations with a learning rate 10-4 and 10-5. We use COCO-trainval35k for training, it contains 35 random subsets of 40k validation images and 80k training images. Our experimental section includes some sub-sections. (1) The detail of the implementation of the experiments is illustrated in the first section. (2) The comparison with the most modern models is explained in detail in the second section. (3) While the third section contains the ablation studies of MRFDet.

1) IMPLEMENTATION DETAIL

To analyze the performance of proposed scheme based on MRFDet, training process starts with warm-up strategy of 5k epochs with learning rate of 0.0001 than gradually decreases up to 10^{-4} and 10^{-5} at 15k epochs and end at 30k epochs. The TensorFlow platform is used to develop the MRFDet, with input size 300×300 and batch size of 32. Experiments are carried out on NVIDIA Geforce RTX 2060, CUDA 10.1 and CuDNN 7.5.0 with memory data rate of 14.00Gbps. The training phase of MRFDet with VGG-16 and input size is 300×300, and 512×512; the total cost of training is four days and a week, respectively.

2) COMPARISON WITH STATE-OF-THE-ART MODELS

a: MS-COCO

In Table 1 we analyze the test results of the proposed MRFDet with state-of-the-art object detectors. For these experiments we use a Multi-level Refinement Feature Pyramid Network with 3 RCUs, FSM and CFM blocks. Input image size, test strategies such as multi-scale technique, model speed, and test results are some of the significant parameters that are included in the comparison. The test results of MRFDet with MS COCO test-dev are portrayed in Table 1. In particular, MRFDet-300 with VGG-16 backbone has achieved an AP of 40.4 and thus surpassing most object detectors with enormous input sizes and more impressive backbones, e.g., Deformable R-FCN [57] has 37.5*AP* and Faster R-CNN with FPN is 36.2*AP*. Assembled with ResNet -101, MRFDet

TABLE 1. Detection accuracy of MRFDet 300 × 300 and 512 × 512 (input size) models' comparison with other state-of-the-art models in term of *mAP* percentage on MSCOCO test-dev set.

Models	Backbone	Input size	Multi-Scale	FPS	Avg. Precision, IOUs:			Avg. Precision, Area		
					0.5:0.95	0.5	0.75	S	M	L
Two-stage Detector:										
CoupleNet [62]	ResNey-101	~1000 × 600	--	8.2	34.4	54.8	37.2	13.4	38.1	50.8
Faster R-CNN [63]	Res101-FPN		--	6	36.2	59.1	39.0	18.2	39.0	48.2
Deformable R-FCN [57]	Inc-Res-v2		--	--	37.5	58.0	40.8	19.4	40.1	52.5
Cascade R-CNN [31]	Res101-FPN	~1280 × 800	--	7.1	42.8	62.1	46.3	23.7	45.5	55.2
SNIP [1]	DPN-98	--	--	--	45.7	67.3	51.1	29.3	48.8	57.1
One –stage detector:										
SSD300* [4]	VGG-16	300 × 300	--	43	25.1	43.1	25.8	6.6	25.9	41.4
DSSD321 [26]	ResNet-101	321 × 321	--	9.5	28.0	46.1	29.2	7.4	28.1	47.6
RetinaNet [5]	ResNet-101	~640 × 400	--	12.3	31.9	49.5	34.1	11.6	35.8	48.5
Refine Det 320 [30]	VGG-16	320 × 320	--	38.7	29.4	49.2	31.3	10.0	32.0	44.4
	ResNet-101		yes	--	38.6	59.9	41.7	21.1	41.7	52.3
M2Det [8]	VGG-16		--	33.4	33.5	52.4	35.6	14.4	37.6	47.6
	VGG-16		yes	--	38.9	59.1	42.4	24.4	41.5	47.6
	ResNet-101		--	21.7	34.4	53.5	36.5	14.8	38.8	47.9
	ResNet-101		yes	--	39.7	60.0	43.3	25.3	42.5	48.3
MRFDet (proposed)	**VGG-16**	**300 × 300**	**--**	**33.5**	**34.0**	**53.5**	**35.5**	**14.7**	**37.5**	**47.9**
	VGG-16	**300 × 300**	**yes**	**--**	**38.2**	**59.2**	**42.5**	**23.8**	**41.4**	**48.2**
	ResNet-101	**300 × 300**	**--**	**35.3**	**35.4**	**55.5**	**36.5**	**15.8**	**39.0**	**48.9**
	ResNet-101	**300 × 300**	**yes**	**--**	**40.4**	**61.0**	**45.2**	**25.3**	**41.7**	**49.8**
SSD512* [4]	VGG-16	512 × 512	--	22	28.8	48.5	30.3	10.9	31.8	43.5
DSSD [26]	ResNet-101	513 × 513	--	5.5	33.2	53.3	35.2	130	35.4	51.1
RetinaNet500 [5]	ResNet-101	~832 × 500	--	11.1	34.4	53.1	36.8	14.7	38.5	49.1
RefineDet512 [30]	VGG-16	512 × 512	--	22.3	33.0	54.5	35.5	16.3	36.3	44.3
	ResNet-101		yes	--	41.8	62.9	45.7	25.6	45.1	54.1
CornerNet [41]	Hourglass			4.4	40.5	57.8	45.3	20.8	44.8	56.7
	Hourglass		yes	--	42.1	57.8	45.3	20.8	44.8	56.7
M2Det [8]	VGG-16		--	18.0	37.6	56.6	40.5	18.4	43.4	51.2
	VGG-16		yes	--	42.9	62.5	47.2	28.0	47.4	52.3
	ResNet-101		--	15.8	38.8	59.4	41.7	20.5	43.9	53.4
	ResNet-101		yes	--	43.9	64.4	48.0	29.6	49.6	54.3
EfficientDet+BiFPN [48]	--		--	--	33.8	52.2	35.8	--	--	--
LibraRetinaNet [49]	ResNet-50-FPN	--	--	--	37.8	56.9	40.5	21.2	40.9	47.7
YOLOv4-SPP	D53	608 × 608	--	--	42.9	62.4	46.6	25.9	45.7	52.4
EtEOD [64]	ResNet-101	--	--	--	43.6	--	--	--	--	--
MRFDet (proposed)	**VGG-16**	**512 × 512**	**--**	**19.5**	**37.5**	**57.1**	**41.1**	**18.6**	**43.2**	**51.4**
	VGG-16	**512 × 512**	**yes**	**--**	**43.2**	**62.6**	**47.5**	**28.2**	**47.3**	**52.4**
	ResNet-101	**512 × 512**	**--**	**18.5**	**39.5**	**60.1**	**42.1**	**20.6**	**43.9**	**53.4**
	ResNet-101	**512 × 512**	**yes**	**--**	**45.2**	**64.0**	**49.0**	**30.1**	**50.3**	**55.1**
RetinaNet800	Res101-FPN	800 × 800	--	5.0	39.1	59.1	42.3	21.8	42.7	50.2
M2Det	VGG-16	800 × 800	--	11.8	41.0	59.7	45.0	22.1	46.5	53.8
YOLOv4-P5	CSP-P5	896 × 896	--		51.8	70.3	56.6	33.4	55.7	63.4
MRFDet (proposed)	VGG-16	800 × 800	--	10.5	43.5	60.1	45.7	22.5	46.9	54.1

can be further improved. To achieve an AP of 41.8, RefineDet [30] acquires the benefits of one and two-stage detector. Key-point regression is used in the CornerNet [41] to detect objects and borrows the advantages of Hourglass [58] and focal loss [5], thus gets AP of 42.1. Conversely, our proposed MRFDet is based on the original SSD regression method using multi-scale multi-level features and reached 45.2 AP, which is higher than any single-stage detectors.

TABLE 2. State-of-the-art COMPARISON with existing single and two stage detector on PASCAL VOC 07/12 test set. Our framework OUTPERFORMS FOR both input image size, such as 300 × 300 and 512 × 512. (Data: '07': Voc 2007 train-val, '07 + 12': VOC 2007 + VOC 2012 train-val, '07 + 12 + coco': first train on 07 + 12 then fine-tune on COCO trainval35K.)

Method	Dataset	*mAP*	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	Table	dog	horse	bike	person	Plant	sheep	sofa	train	T.V.
Fast -RCNN	07	66.9	74.5	78.3	69.2	53.2	36.6	77.3	78.2	82	40.7	72.7	67.9	79.6	79.2	73	69	30.1	65.4	70.2	75.8	65.8
	07+12	70	77	78.1	69.3	59.4	38.3	81.6	78.6	86.7	42.8	78.8	68.9	84.7	82.0	76.6	69.9	31.8	70.1	74.8	80.4	70.4
Faster -RCNN [63]	07	69.9	70	80.6	70.1	57.3	49.9	78.2	80.4	82	52.2	75.3	67.2	80.3	79.8	75	76.3	39.1	68.3	67.3	81.1	67.6
	07+12	73.2	76.5	79	70.9	65.5	52.1	83.1	84.7	86.4	52	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83	72.6
	07+12+COCO	78.8	84.3	82	77.7	68.9	65.7	88.1	88.4	88.9	63.6	86.3	70.8	85.9	87.6	80.1	82.3	53.6	80.4	75.8	86.6	78.9
SSD300 [4]	07	68	73.4	77.5	64.1	59.0	38.9	75.2	80.8	78.5	46	67.8	69.2	76.6	82.1	77	72.5	41.2	64.2	69.1	78	68.5
	07+12	74.3	75.5	80.2	72.3	66.3	47.6	83	84.2	86.1	54.7	78.3	73.9	84.5	85.3	82.6	76.2	48.6	73.9	76.0	83.4	74
	07+12+COCO	79.6	80.9	86.3	79.0	76.2	57.6	87.3	88.2	88.6	60.5	85.4	76.7	87.5	89.2	84.5	81.4	55.0	81.9	81.5	85.9	78.9
SSD512 [4]	07	71.6	75.1	81.4	69.8	60.8	46.3	82.6	84.7	84.1	48.5	75.0	67.4	82.3	83.9	79.4	76.6	44.9	69.9	69.1	78.1	71.8
	07+12	76.8	82.4	84.7	78.4	73.8	53.2	86.2	87.5	86.0	57.8	83.1	70.2	84.9	85.2	83.9	79.7	50.3	77.9	73.9	82.5	75.3
	07+12+COCO	81.6	86.6	88.3	82.4	76	66.3	88.6	88.9	89.1	65.1	88.4	73.6	86.5	88.9	85.3	84.6	59.1	85.0	80.4	87.4	81.2
TDFSSD300 [65]	07+12	79.8	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
TDFSSD512 [65]	07+12	81.7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
MRFDet300	**07**	**69.2**	**74.1**	**78.5**	**65.3**	**60**	**39.1**	**78.3**	**81.4**	**79.6**	**47**	**67.4**	**70.5**	**76.3**	**84.5**	**79**	**73.5**	**42.5**	**65.8**	**70.3**	**79**	**69.5**
	07+12	**75.7**	**76.3**	**80.3**	**74.8**	**67.3**	**47.8**	**84.1**	**84.6**	**87.3**	**54.9**	**79.6**	**75.2**	**85.7**	**86.7**	**84.2**	**76.9**	**49.2**	**74.6**	**78.0**	**82.4**	**75**
	07+12+COCO	**81.6**	**83**	**86**	**79.4**	**78.2**	**59.1**	**87.9**	**89**	**89.5**	**62.1**	**87.3**	**77.4**	**88.3**	**90.3**	**85.8**	**83**	**56.4**	**82.8**	**82.5**	**87.1**	**79.2**
MRFDet512	**07**	**72**	**77.1**	**82.9**	**70.0**	**61.2**	**47.4**	**83.0**	**85.3**	**85.7**	**49.7**	**77**	**69.2**	**84.5**	**85**	**81**	**77.6**	**46.0**	**70.3**	**70.1**	**79.5**	**73.6**
	07+12	**78.5**	**80.4**	**85.1**	**78.1**	**74.8**	**54.8**	**87.5**	**88.6**	**87.2**	**58.3**	**84.2**	**72.6**	**86.0**	**87.8**	**84.6**	**80.1**	**51.3**	**78.4**	**74.4**	**83.4**	**76.7**
	07+12+COCO	**84.6**	**86.3**	**89.3**	**83.4**	**78.1**	**68.9**	**89.8**	**89.3**	**92.1**	**66.9**	**89.6**	**76.7**	**87.5**	**90.0**	**86.8**	**85.3**	**60.2**	**88.1**	**81.3**	**90.8**	**82.1**

TABLE 3. Ablation experiments regarding design of MRFDet on MS-COCO-miniVal set. The results shows the improvement in detection accuracy.

Module Variations	Cases							
+ 3RCU		*						
+3RCU+ 2(conv3)			*					
+3RCU+ 2(conv3) + 1(conv1)				*	*	*	*	*
+Base feature with feature map of layers					*	*	*	*
+FSM unit						*	*	*
+CFM (4)							*	*
ResNet101								*
AP	25.3	27.6	30.7	30.9	33.0	34.2	34.6	**36.2**
AP_{50}	44.7	46.4	52.3	52.1	52.4	53.7	53.9	**54.7**
AP_{Small}	7.2	8.9	15.2	15.4	15.3	15.7	15.8	**16.9**
AP_{medium}	27.0	29.9	36.8	37.6	39.2	39.7	38.2	**41.2**
AP_{large}	40.9	47.6	46.3	45.7	50.0	51.0	49.0	**51.6**

TABLE 4. Standard pattern of MRFPN in MRFDet with backbone-VGG-16, 300 × 300 image size.

Variant of Channels	RCU_s	RCU_m	RCU_d	Contextual feature module (2)	Parameters (M)	AP	AP_{50}	AP_{75}
	128	128	128	256	57.8	32.4	53.1	36.1
	256	256	256	256	80.2	35.3	54.4	37.0
	512	512	512	256	165.5	36.1	54.3	37.9
	128	256	512	256	118.2	36.3	54.6	38.2

The parameter of comparison between different sophisticated models is speed of single scale inference methods as different strategies and tools are used. Furthermore, the increase in performance of MRFDet is not entirely due to depth of the model or obtained parameters. In comparison with modern one- and two-stage detectors, we find that 201M parameters are generated in CornerNet with Hourglass and 205M parameters are generated in Mask R-CNN [6] (ResNet-101-32 × 8d-FPN [59]). While our model generates 146M parameters.

b: PASCAL VOC2007/2012

The test images from PASCAL VOC 07/12 are used to compare the performance of our proposed model with most advanced one and two-stage detectors, as depicted in Table 2. The standard configuration of MRFDet contains three RCUs with a feature standardization module and a contextual feature module as shown in Fig. 2. While conventional SSD prediction layers are used to compute confidence score and localization. The "Xavier" method [60] is used to initialize the parameters of all prediction layers. The backpropagation is used to learn the scaling, since the feature norm is scaled to 20 at each location in the feature map using L2 normalization technique [61]. Our model is trained on PASCAL VOC 07/12 and further fine-tune on the MS-COCO trainval35k in order to achieve better results.

To correctly assess the class confidence score and location, we cannot use the correction rate of image classification because each image has multiple objects under multiple categories in the object detection problem. In PASCAL VOC 07/12 test set, *mAP* is used as the evaluation index of accuracy, and frames per second are used as the evaluation index

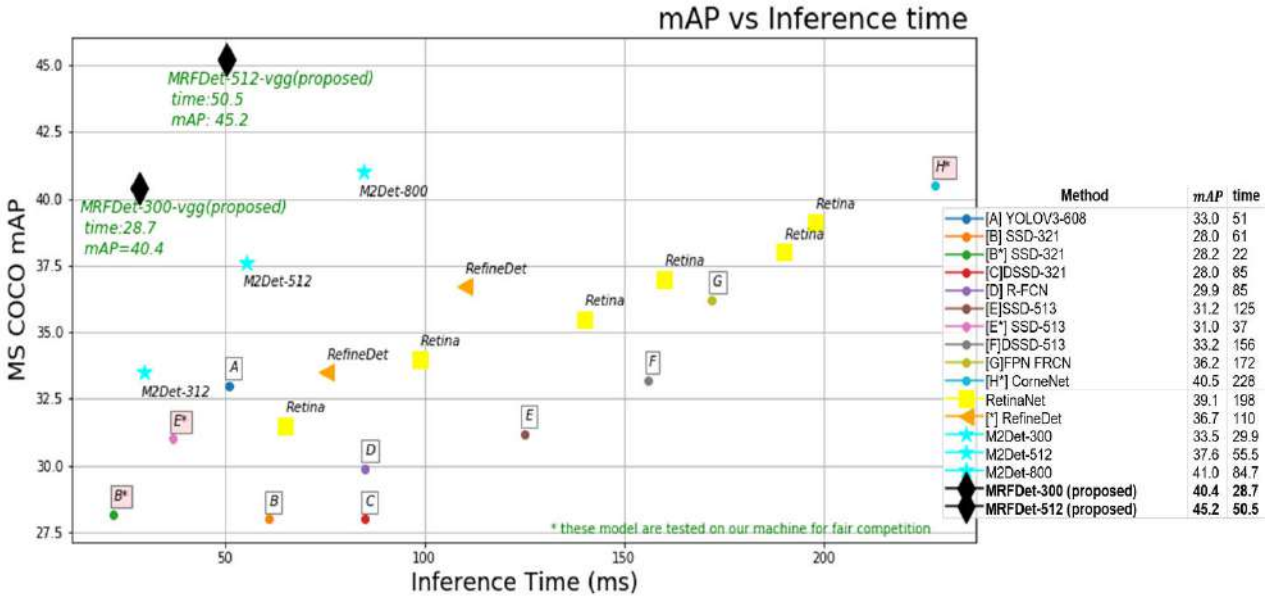

FIGURE 6. MS-COCO test-dev: graph between speed (ms) versus accuracy (*mAP*).

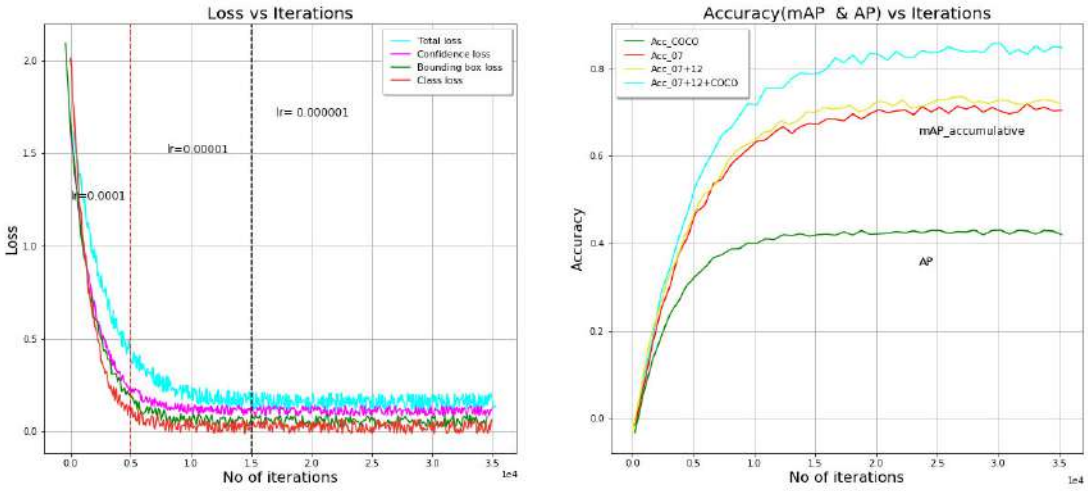

FIGURE 7. (a) Loss vs. iteration graph during training class loss, bounding box loss, confidence loss and total loss with three learning rates (0.0001 lr : 00 <= $iteration$ <= 500; 0.00001 lr : 5k < $iteration$ <= 15k; 0.000001 lr : 15k < $iteration$ <= 35k). (b) Accuracy vs. iterations graph during training (AP: on MS COCO dataset; *mAP* : Pascal Voc 07, 07+12, 07 + 12 + coco).

for real-time detection. Precision (P) and recall (R) for each class are used to draw P-R curve. The formula of precision and recall is as follows:

$$P = {}^{TP}/_{TP+FP'}, \quad R = {}^{TP}/_{TP+FN'} \tag{10}$$

where FP represents the numbers of false-positive predicted samples, TP represents the number of the true-positive predicted samples, and FN represents the numbers of false-negative predicted samples. The formula of the *mAP* and *AP* are given below:

$$AP = \int_0^1 p(R)dR \tag{11}$$

$$mAP = \sum_{i=1}^{N} \frac{AP(i)}{N} \tag{12}$$

The frame per second is defined as the number of pictures that recognize in one second by a detector. FPS rate above 24 is considered as smooth. As is clear from Table 2, our proposed model MRFDet- 300 (low resolution) is already performing better than Fast R-CNN. When input size is increased to 512 × 512 for further training, it is even more accurate and outperforms the Faster R-CNN by 2.7% mAP. If we train the MRFDEt-300 with additional data (i.e., 07 + 12), we observe that MRFDet-300 is already 2.1% better than SDD and Faster R-CNN and MRFDet-512 by 3.6% better. We get our best results

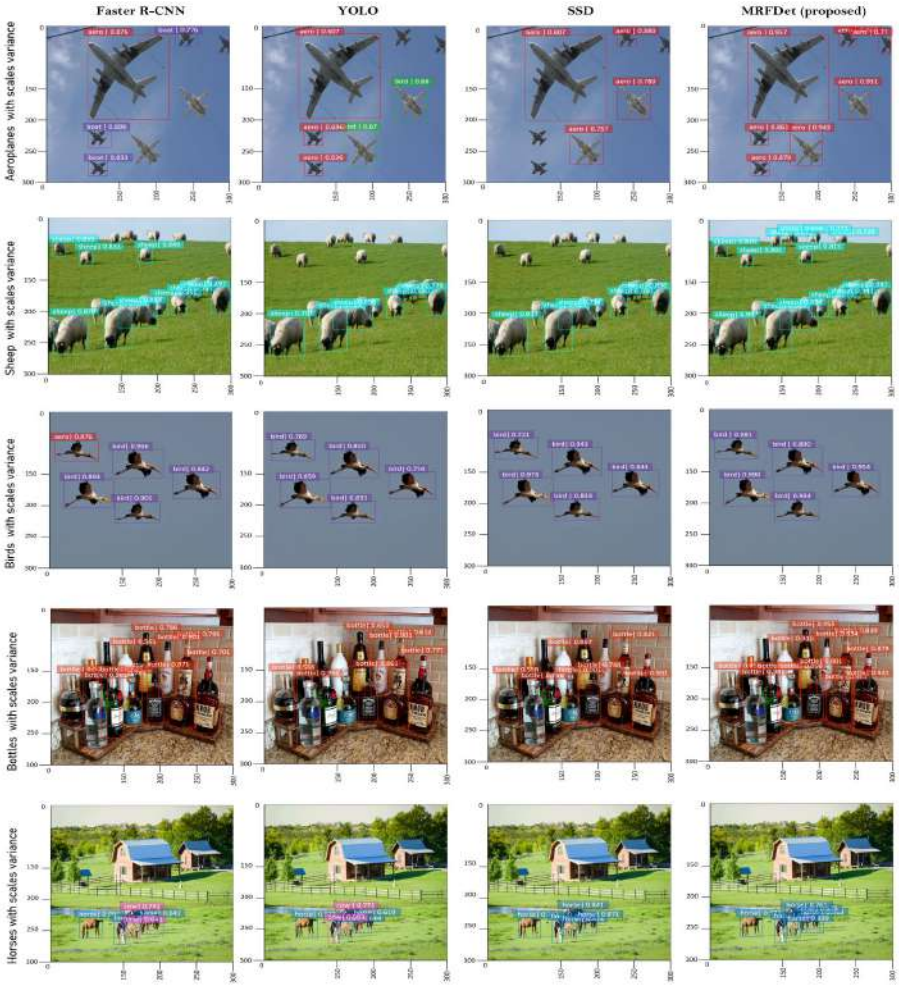

FIGURE 8. Comparison of results with other modern models.

after fine-tuning our model on MS-COCO trainval35k that is 84.6%mAP. MRFDet is particularly sensitive to the size of bounding boxes, and due to multi-level feature pyramid with several scales, offers significantly better performance with small objects. Those small objects have semantic information in the MLFP that help in detecting small objects.

3) ABLATION STUDIES ON MS-COCO

In this section, we review the effectiveness of each module configuration of MRFDet on detection performance. We are trying three different designs of RCU in our MRFPN. First, a simple design of RCUs upgrades the AP to up to three units, as shown in the third section of Table 3. In the first branch of each RCU, an additional conv layer with a 3×3 filter is used, which improves the AP to 3.1. Finally, increasing the conv layer in the second branch of the RCU delivers the best results in the 30.9 AP performance. Although increments in conv-layers can improve detection accuracy, the redundant use of the basic function in each RCU increases the number of parameters. For this reason, features of different layers of backbone networks are used to construct low, medium, and high-level features. While the necessary location information is obtained through the embedded basic feature. The Feature Standardization Module (FSM) improved all scoring measurements, as shown in the seventh column of Table 3. Next, we analyzed the effect of our proposed contextual features module on recognition performance. Additional block of parallel pooling improves performance significantly. But we've seen the best results with four consecutive blocks. Strong base functions provide a noticeable AP gain, such as using ResNet-101 as the backbone instead of VGG-16, it generated an AP gain of 2% as shown in Table 3.

4) VARIANTS OF MLFPN

It has been observed that multi-scale features are very effective in detection tasks. It remains to be seen to what extent MRFPN has made the improvement? And how to design RCU and contextual feature module? And how many blocks of RCU ought to be fine? A combination of variants of standard pattern is examined, such as VGG-16 as backbone network and input size 300×300 and tune the no of RCUs, adaptation of internal channels of each RCU and architecture of contextual feature module. As appeared in Table 4,

FIGURE 9. Qualitative results of the proposed framework. Our method works well with occlusion, interclass interference and cluster background.

different configuration of RCUs and contextual features modules are used to assess the performance of proposed MRFDet on COCO mini-Val set.

Stacking more Conv layers in RCUs and in the contextual features module gives more boost in terms of accuracy. The increase in the number parameter remains comparable while adapting a combination of three RCUs with 512, 256, or 128 channels respectively.

5) SPEED

In the context of comparing inference speed of MRFDet with latest models, we conclude that reduced version of VGG-16 [50] (without F.C. layers) speed up the base feature extraction process. The inference time of an image is a sum of NMS time and CNN time of 1000 images and is divided by 1000, and batch size is set to 1. Specifically, we assembled the MRFDet with VGG-16 (reduced version) and proposed the faster version of MRFDet with input size 320 × 320, and the standard and accurate version of MRFDet with input size 512 × 512. Taking advantage of our proposed MRFPN framework and one-stage detector, MRFDet has significantly improved the speed and accuracy curve compared to other advanced methods. MRFDet can achieve precise detection results with high speed based on the optimization of TensorFlow. The speed of SSD321-ResNet101, SSD513-ResNet101, M2Det-VGG16, RefineDet512-ResNet101, RefineDet320-ResNet101, and

CornerNet are tested on our device for fair comparison. It comes to the conclusion that MRFDet performs far better in term of accuracy and efficiency.

IV. DISSCUSSION

According to our observations, the detection accuracy of MRFDet has improved mainly due to the proposed MRFPN and Contextual features module. Firstly, we merge the multilayered feature maps and backbone base-features through alternate blocks of RCU and MLFF modules to extract more robust multi-scale multi-level features. Finally, constructed features are fed into CFM to minimize the parameters and add more robustness to object detection. In contrast, existing detectors [3], [26], [30] are only used with an increase in the depth of the layers of the backbone or extra layers. Therefore, predominant detection performance has been achieved through our proposed method. In particular, multi-level multi-scale features are used to demonstrate better performance in dealing with appearance-complexity variation across objects instances. Our proposed MRFPN can learn effective features for detecting an object with large variations in appearance and scales. E.g., the input image contains person, vehicles and traffic signal of different sizes. Some of the findings are as follows: 1) A larger size person has higher and stronger activation value at feature map as compared to a smaller one. 2) While the small-sized person, traffic signal, and the vehicle have substantial activation values with the same scale feature map. 3) Conversely, the individual, vehicle and traffic signals have an extremely robust activation value on the feature maps of the significant level, the middle level and the lowest level. From our observations, the proposed method can effectively learn sensitive features to deal with variations in scale and complexity of appearance across object instances. It is essential to use multi-level/scale features to identify objects of comparable size but different in appearance.

V. CONCLUSION

Multi-level refinement feature pyramid network is proposed to identify the objects with different scales and complex appearances. The proposed strategy consists of three modules, a stack of three residual convolutional units is used in the first module to construct the multi-scale multi-level features using feature maps of backbone network and base features (i.e., constructed from VGG-16). The second module is used to standardize multi-level features after up/down-sampling with similar scale in the form of feature pyramid. Finally, contextual features module is used to strengthen the resulting feature pyramid for object identification. We achieve significantly improved scores compared to other single-stage detector on MS-COCO dataset (i.e., 45.2 AP with multi-scale inference strategy). The effectiveness of proposed architecture is demonstrated using the results of ablation studies. However, there is still a room for improvement in detection, such as GAN can be used to reconstruct high resolution deep features or to optimize the upper sample layer using interpolation techniques.

REFERENCES

[1] B. Singh and L. S. Davis, "An analysis of scale invariance in object detection–SNIP," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3578–3587.
[2] 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. 2015.
[3] T.-Y. Lin, P. Dollar, 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.
[4] 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.*, Oct. 2016, pp. 21–37.
[5] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.
[6] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," in *Proc. ICCV*, Oct. 2017, pp. 2961–2969.
[7] 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.*, Sep. 2014, pp. 740–755.
[8] Q. Zhao, T. Sheng, Y. Wang, Z. Tang, Y. Chen, L. Cai, and H. Ling, "M2det: A single-shot object detector based on multi-level feature pyramid network," in *Proc. AAAI Conf. Artific. Intel.*, 2019, pp. 9259–9266.
[9] 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)*, Dec. 2001, pp. 1–9.
[10] P. Dollar, Z. Tu, P. Perona, and S. Belongie, "Integral channel features," in *Proc. Brit. Mach. Vis. Conf.*, 2009, p. 91.
[11] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2005, pp. 886–893.
[12] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, Apr. 2013.
[13] 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.
[14] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.
[15] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, "Scalable object detection using deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 2147–2154.
[16] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.
[17] P. O. Pinheiro, R. Collobert, and P. Dollar, "Learning to segment object candidates," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 1990–1998.
[18] 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.
[19] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 761–769.
[20] R. Vaillant, C. Monrocq, and Y. L. Cun, "Original approach for the localisation of objects in images," *IEE Proc.-Vis., Image Signal Process.*, vol. 141, no. 4, pp. 245–250, Aug. 1994.
[21] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick, "Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2874–2883.
[22] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang, "Acquisition of localization confidence for accurate object detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 784–799.
[23] T. Kong, A. Yao, Y. Chen, and F. Sun, "HyperNet: Towards accurate region proposal generation and joint object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 845–853.
[24] T. Kong, F. Sun, A. Yao, H. Liu, M. Lu, and Y. Chen, "RON: Reverse connection with objectness prior networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5936–5944.
[25] 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*.

[26] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "DSSD: Deconvolutional single shot detector," 2017, *arXiv:1701.06659*.
[27] F. Sun, T. Kong, W. Huang, C. Tan, B. Fang, and H. Liu, "Feature pyramid reconfiguration with consistent loss for object detection," *IEEE Trans. Image Process.*, vol. 28, no. 10, pp. 5041–5051, May 2019.
[28] Z. Shen, Z. Liu, J. Li, Y.-G. Jiang, Y. Chen, and X. Xue, "DSOD: Learning deeply supervised object detectors from scratch," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy, Oct. 2017, pp. 1919–1927.
[29] T. Kong, F. Sun, H. Liu, Y. Jiang, and J. Shi, "Consistent optimization for single-shot object detection," 2019, *arXiv:1901.06563*.
[30] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li, "Single-shot refinement neural network for object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4203–4212.
[31] Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving into high quality object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6154–6162.
[32] J. Guo, K. Han, Y. Wang, C. Zhang, Z. Yang, H. Wu, X. Chen, and C. Xu, "Hit-detector: Hierarchical Trinity architecture search for object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11405–11414.
[33] 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.
[34] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 7263–7271.
[35] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*.
[36] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully convolutional one-stage object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9627–9636.
[37] L. Huang, Y. Yang, Y. Deng, and Y. Yu, "DenseBox: Unifying landmark localization with end to end object detection," 2015, *arXiv:1509.04874*.
[38] Z. Yang, S. Liu, H. Hu, L. Wang, and S. Lin, "RepPoints: Point set representation for object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9657–9666.
[39] C. Zhu, Y. He, and M. Savvides, "Feature selective anchor-free module for single-shot object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 840–849.
[40] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," 2019, *arXiv:1904.07850*.
[41] H. Law and J. Deng, "CornerNet: Detecting objects as paired keypoints," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 734–750.
[42] A. Newell, Z. Huang, and J. Deng, "Associative embedding: End-to-end learning for joint detection and grouping," 2016, *arXiv:1611.05424*.
[43] T. Kong, F. Sun, H. Liu, Y. Jiang, L. Li, and J. Shi, "FoveaBox: Beyound anchor-based object detection," *IEEE Trans. Image Process.*, vol. 29, pp. 7389–7398, 2020.
[44] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8759–8768.
[45] P. Zhou, B. Ni, C. Geng, J. Hu, and Y. Xu, "Scale-transferrable object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 528–537.
[46] M. A. Islam, M. Rochan, N. D. B. Bruce, and Y. Wang, "Gated feedback refinement network for dense image labeling," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3751–3759.
[47] G. Ghiasi, T.-Y. Lin, and Q. V. Le, "NAS-FPN: Learning scalable feature pyramid architecture for object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 7036–7045.
[48] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 10781–10790.
[49] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, "Libra R-CNN: Towards balanced learning for object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 821–830.
[50] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[51] M. Lin, Q. Chen, and S. Yan, "Network in network," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2014.
[52] L. Aziz, M. S. B. H. S. Fc, and S. Ayub, "Multi-level refinement enriched feature pyramid network for object detection," *Image Vis. Comput.*, vol. 115, Nov. 2021, Art. no. 104287.
[53] C. Szegedy, S. Reed, D. Erhan, D. Anguelov, and S. Ioffe, "Scalable, high-quality object detection," 2014, *arXiv:1412.1441*.
[54] L. Zhang, L. Lin, X. Liang, and K. He, "Is faster R-CNN doing well for pedestrian detection," in *Proc. Eur. Conf. Comput. Vis.*, Oct. 2016, pp. 443–457.
[55] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, and A. C. Berg, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.
[56] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis, "Soft-NMS—Improving object detection with one line of code," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5561–5569.
[57] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 764–773.
[58] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, Oct. 2016, pp. 483–499.
[59] S. Xie, R. Girshick, P. Dollar, 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.
[60] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, vol. 2010, pp. 249–256.
[61] W. Liu, A. Rabinovich, and A. C. Berg, "ParseNet: Looking wider to see better," 2015, *arXiv:1506.04579*.
[62] Y. Zhu, C. Zhao, J. Wang, X. Zhao, Y. Wu, and H. Lu, "CoupleNet: Coupling global structure with local parts for object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 4126–4134.
[63] 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.
[64] J. Wang, L. Song, Z. Li, H. Sun, J. Sun, and N. Zheng, "End-to-end object detection with fully convolutional network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 15849–15858.
[65] H. Pan, J. Jiang, and G. Chen, "TDFSSD: Top-down feature fusion single shot MultiBox detector," *Signal Process., Image Commun.*, vol. 89, Nov. 2020, Art. no. 115987.

LUBNA AZIZ received the bachelor's and master's degrees in computer engineering from the Balochistan University of Information Technology, Engineering and Management Sciences (BUITMES), Quetta, Balochistan, Pakistan, in 2008 and 2017, respectively. She is currently pursuing the Ph.D. degree with the University of Technology Malaysia (UTM), Johor Bahru, Malaysia. She has been on study leave from BUITEMS, since 2019. Her research interests include image processing (biomedical imaging), machine learning, deep learning, and computer vision, especially object recognition.

MD SAH BIN HAJI SALAM (Member, IEEE) received the Bachelor of Science degree from the University of Pittsburgh, PA, USA, and the Ph.D. degree in computer science from the UTM Faculty of Engineering, School of Computing, University of Technology Malaysia. He is currently the Head of VICUBELA, University Technology Malaysia. His research interests include language processing and image processing.

USMAN ULLAH SHEIKH (Senior Member, IEEE) received the Ph.D. degree in image processing and computer vision from Universiti Teknologi Malaysia, in 2009. He is currently a Senior Lecturer with the Department of Computer and Electrical Engineering, Faculty of Engineering, Universiti Teknologi Malaysia. His research interests include computer vision, machine learning, and embedded systems design.

SURAT KHAN received the B.S. degree from the Balochistan University of Engineering and Technology Khuzdar, Pakistan, the M.S. degree from UET Peshawar, Pakistan, and the Ph.D. degree in management science and technology from BUPT, Beijing, China, in 2012. He has 24 years of professional experience with Siemens, PTCL, and BUITEMS.

HUMA AYUB received the B.S. and M.S. degrees from Balochistan University. She is currently pursuing the Ph.D. degree with Sardar Bahadur Khan Women's University, Balochistan. She has also been a member of SBK, since 2018. She worked for ten years as a Research Assistant with PCIR, Pakistan.

SARA AYUB (Member, IEEE) received the M.S. degree in signal and image processing from NUST, Pakistan, and the Ph.D. degree from the Faculty of Engineering, UTM. She is currently working as an Assistant Professor with BUITEMS.