Object Detection

Literature Survey

Mustafa Abdullah Hakkoz[*1]

Computer Science Engineering, Marmara University
Goztepe Campus, 34722, Kadikoy, Istanbul, Turkey

[*1]mustafahakkoz@marun.edu.tr

***Abstract*-Object detection, a computer vision topic specializing on predicting classes of objects in images or videos, gains popularity currently. Recent advances in deep learning and the increasing demand of industry require researchers to keep them updated. New domains of applications appear constantly; autonomous driving, virtual reality, augmented reality, video processing, aerial image processing, surveillance, manufacturing, all of them are fed by the development of object detection techniques. In this study, a complete review of the literature will be done, from the classical methods to deep learning-based approaches.**

Keywords- Object detection; literature survey; machine learning; computer vision; image processing; deep learning; CNN; one stage detectors; two stage detectors

I. INTRODUCTION

The task of locating and classifying an object in images or videos is crucial when a computer interacts with its environment. For example, in autonomous driving, the computer should run many tasks for car detection, pedestrian detection, traffic sign detection, road tracking, trace generation (Fig 1) etc. And almost all of them heavily rely on fast and accurate object detection methods.

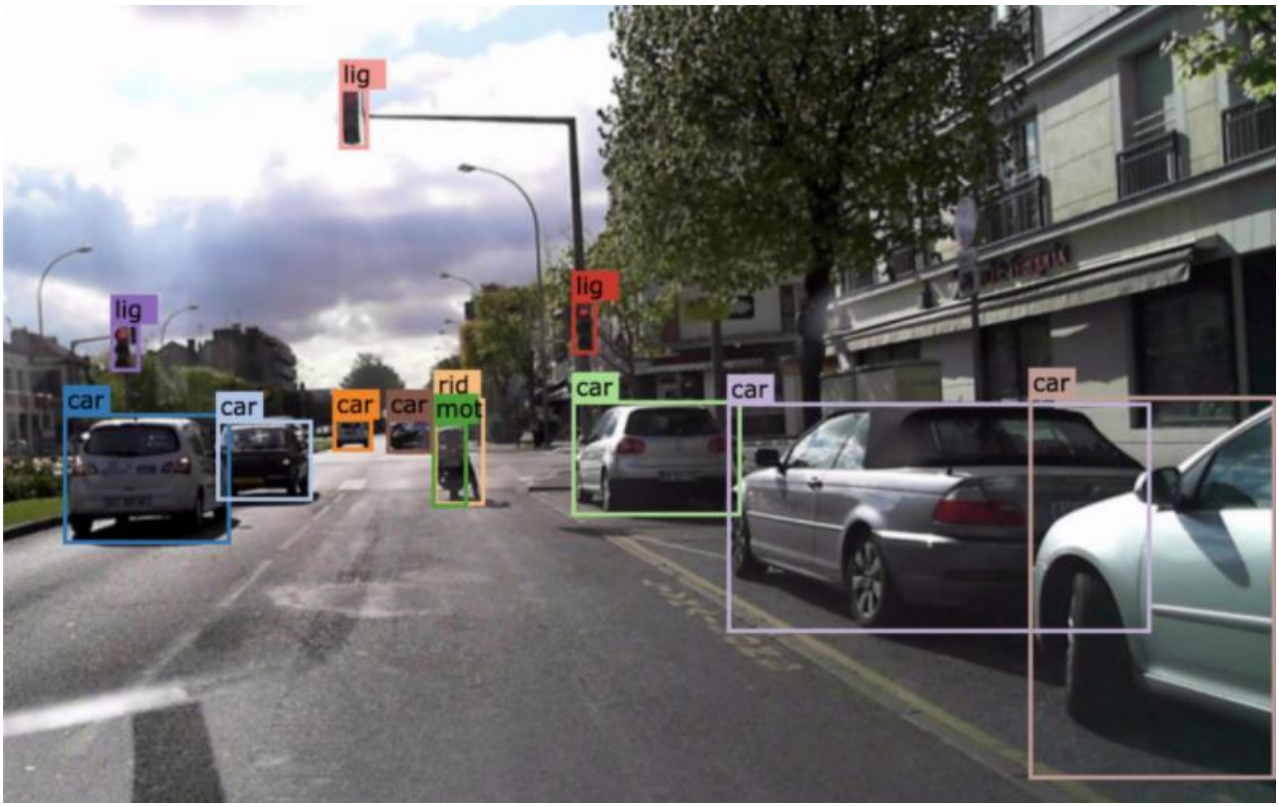

Fig. 1 In object detection all objects are marked by bounding boxes and class names. Original image from Berkeley DeepDrive [1].

In classical object detection pipelines, the system generally consists of multiple steps to handle multiple problems such as, image segmentations, region proposals, object localization, object classification along with pre-processing and post-processing steps. With the success of deep neural networks, these tasks are taken over by trained models separately or end-to-end designs.

In this survey, object detection techniques will be examined from classical to novel approaches along with evaluation methods, benchmarks, datasets and some important concepts offered by the researchers of the domains.

II. Classical Methods

A classical object detection pipeline consists of the steps preprocessing, region of interest (RoI) extraction, classification and verification or refinement. In the preprocessing phase, several methods to clean external effects and noises are performed such as exposure and gain adjustment, camera calibration, image rectification etc. Also, some approaches leveraging temporal information with a joint detection and tracking system like optical flow.

RoIs can be extracted using sliding windows technique with heuristics to reduce search space with predefined assumptions on ratio, size and location. Another alternative to generate RoIs is selective search [28]. Instead of exhaustive search over full image, selective search uses segmentations to extract object locations efficiently. It uses outputs of Felzenszwalb's algorithm [29] which give image segmentations and does a bottom-up hierarchical grouping by using region-based characteristics (Fig. 2).

Algorithm 1: Hierarchical Grouping Algorithm

DontPrintSemicolon **Input**: (colour) image
Output: Set of object location hypotheses L

Obtain initial regions $R = \{r_1, \cdots, r_n\}$ using Felzenszwalb and Huttenlocher (2004) Initialise similarity set $S = \emptyset$;
foreach *Neighbouring region pair* (r_i, r_j) **do**
  Calculate similarity $s(r_i, r_j)$;
  $S = S \cup s(r_i, r_j)$;

while $S \neq \emptyset$ **do**
  Get highest similarity $s(r_i, r_j) = \max(S)$;
  Merge corresponding regions $r_t = r_i \cup r_j$;
  Remove similarities regarding r_i : $S = S \setminus s(r_i, r_*)$;
  Remove similarities regarding r_j : $S = S \setminus s(r_*, r_j)$;
  Calculate similarity set S_t between r_t and its neighbours;
  $S = S \cup S_t$;
  $R = R \cup r_t$;
Extract object location boxes L from all regions in R;

Fig. 2 Detailed algorithm of Selective Search

The next step is classification of all candidates of RoIs. There are popular approaches on this step such as using cascades that discards false candidates (AdaBoost) in Viola-Jones algorithm [30] or using SVM with Histogram of Gradients (HoG) features [31]. HoG features are basically distributing gradient magnitudes over gradient directions bins and constructs feature vectors for a pixel cell (Fig. 3). These features can be used as inputs for machine learning classifiers.

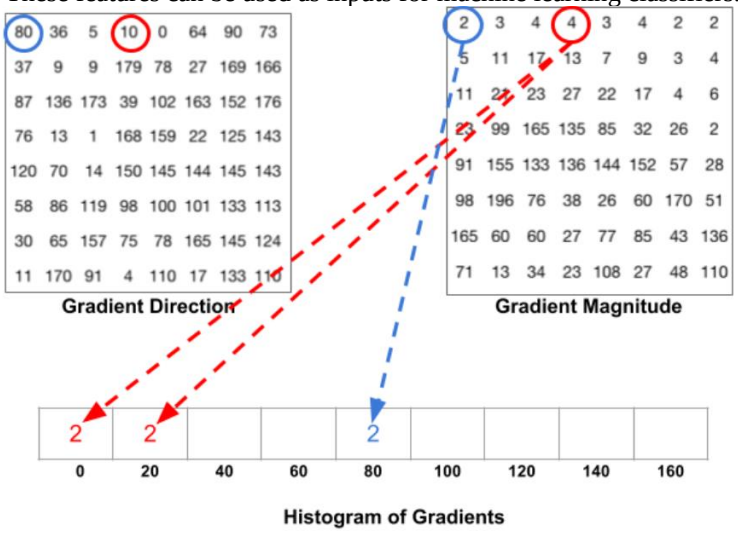

Fig. 3 How to split one gradient vector's magnitude if its degrees is between two degree bins. (Image source: [32])

III. Deep Learning Based Methods

With the rise of deep learning [2], most of computer vision tasks are taken over by them. CNN-based approaches are also very popular in object detection problem and researchers are focusing the accuracy and speed of these models. There are a lot of modifications on basic backbones and most of the ideas are coming from challenges like MS COCO [3], PAS-CAL VOC [4] and other datasets.

In this section backbone networks, modifications and some related concepts will be explained first, then single stage and two stage networks will be reviewed.

A. Backbone Networks

CNN is short for "Convolutional Neural Network" and inspired by how human visual cortex system works. It's based on convolution operation, a sliding window of multiplication-summation of predefined kernel (a.k.a. filter) and input image. Output image is generally much smaller and represents feature maps (Fig. 4).

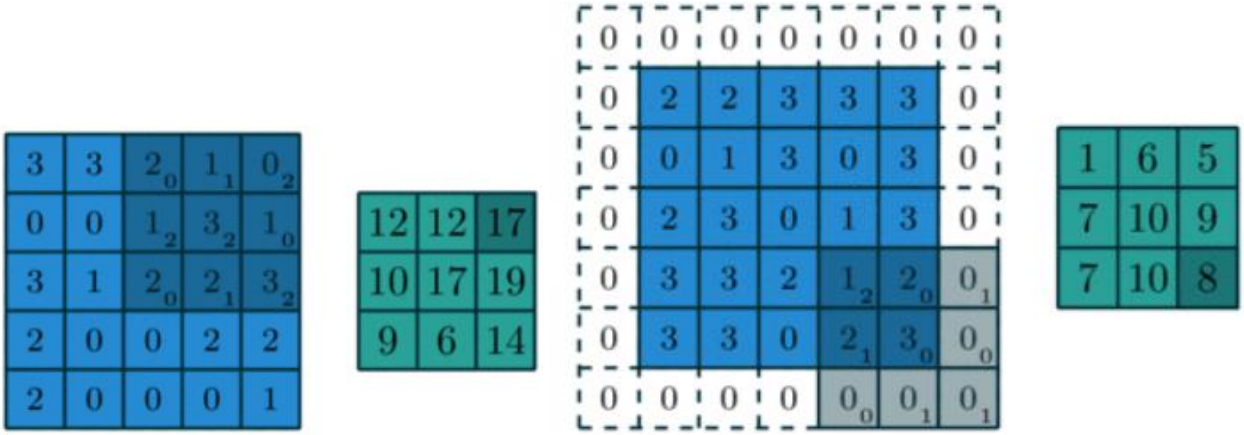

Fig. 4 On left 3x3 convolution with no padding, on right padding input image with zeros. Output shape didn't change since the operation on the right also uses stride of 2. (image source: lilianweng.github.io [5])

A lot of deep neural networks originally designed for classification tasks are adopted in object detection with extra modifications for domain specific difficulties encountered. The most popular architectures in increasing order of inference time are MobileNet [6], VGG [7], Inception [8], ResNet [9], Inception-ResNet [10], etc. These models are borrowed from ImageNet [11] competitions and mostly used as feature extractor in object detection pipelines (Fig. 4).

B. Objectness and Bounding Boxes

Objectness is defined by how likely an image region contains an image or not and it's a class-agnostic value. Region of Interests (RoIs) are object proposals gives us where to focus to find a proper object. And they generally marked by bounding boxes (Fig. 5). In general, models first make predictions for objectness of RoIs to separate objects from background, then classify them according to predefined classes. And finally, they may some elimination or refinement on bounding box scales and positions.

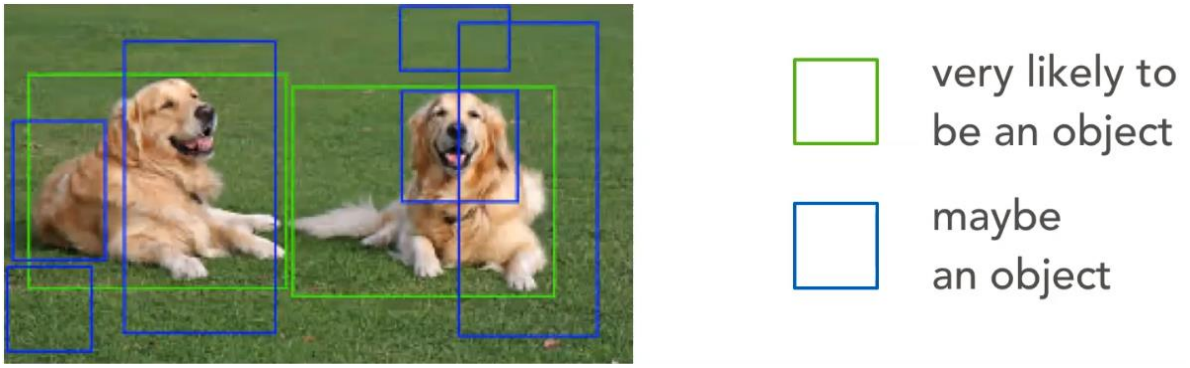

Fig. 5 Object proposals with bounding boxes. Green boxes are true positives while blue ones are false positives.

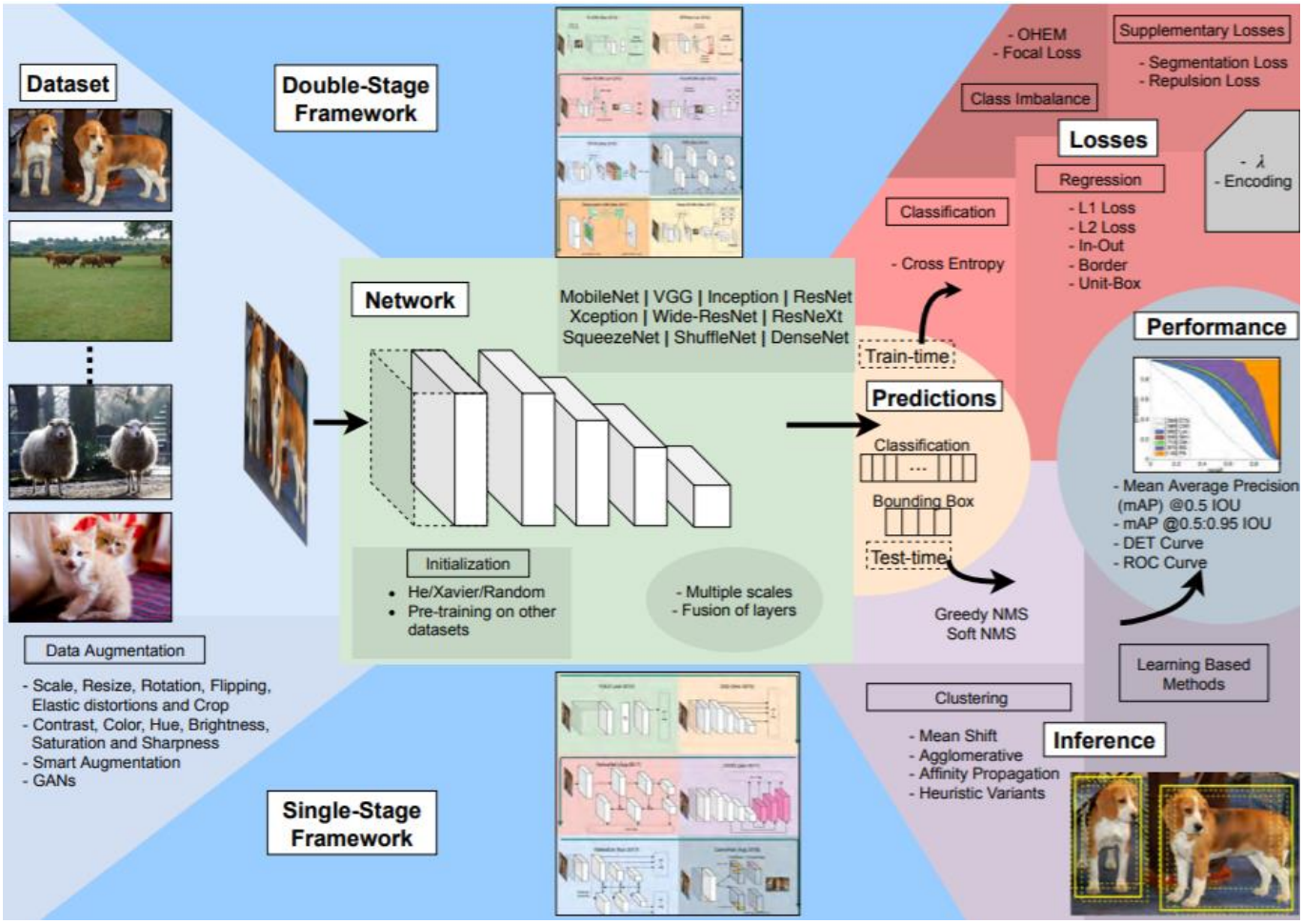

Fig. 6 A general view of object detection pipelines. Images are taken from dataset with data augmentation techniques then fed in to single-stage or double-stage frameworks. They use classical backbones and make predictions on class label and bounding box regressions. During training, several losses used for several tasks like classification, regression, segmentation etc. And during inference there may be some optimization tricks to increase accuracy like NMS and final performance is evaluated by standard metrics like mAP. (image source: [12])

C. Evaluation Metrics: IoU and mAP

Intersection over Union (IoU) or Jaccard Index is a similarity score between two bounding boxes. A detection is called true positive if an object proposal has IoU greater than certain threshold (usually 0.5) according to ground-truth box. The formula is defined in Fig. 7 and its values varies in range 0 to 1.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

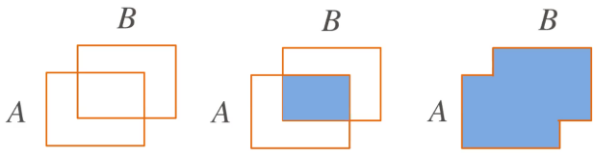

Fig. 7 Jaccard Index between A and B is intersection over union.

A common evaluation metric mean average precision (mAP) is used to measure the success of object detection models. It is a number between 0 and 100, higher value is better. To calculate it, first all detections from all input images will be combined and precision-recall (PR) curve will be drawn for each class. Average precision (AP) is the area under the PR curve. After calculating AP value for each class, second average will be calculated over all classes and it's called mean average precision (mAP).

While calculating simple precision, we need to detect true positives first. This is where IoU comes into play, if a predicted bounding box has IoU with ground-truth box greater than lambda threshold, it is considered as true positive. There are several implementations of mAP and most common one is COCO version. In COCO notation AP@.5 means average precision with

0.5 IoU threshold (also used in PASCAL VOC [17]) and there's no distinction between AP and mAP there. Simple AP notation is calculated over 10 different IoU thresholds from 0.5 to 0.95 and there are also other notations like AP^{small}, AP^{medium}, AP^{large} for varying object scales [13].

D. Anchors

While predicting a location of an object, we determine the size and location of the bounding box. This could be done by sliding window technique but again using different scales would be costly. To overcome this issue, generally bounding boxes with predefined scales (anchors) are used. Scales and sizes of anchors can be selected by manually according to task [14, 15], or can be generated by unsupervised methods by running on a dataset [16]. In Fig.8 predictions for 9 anchors, 3 sizes for 3 scales, are evaluated in each sliding window. For example, in object detection of the autonomous vehicle scenario anchors can be interpreted as; AR_2 is suitable for detecting cars, AR_3 is for pedestrians and AR_1 is more likely for general purposes. This idea is first introduced in Faster-RCNN [14], then adopted by almost all kinds of models, both single-stage and double-stage detectors [15,16].

$$x_a, y_a, w_a, h_a$$

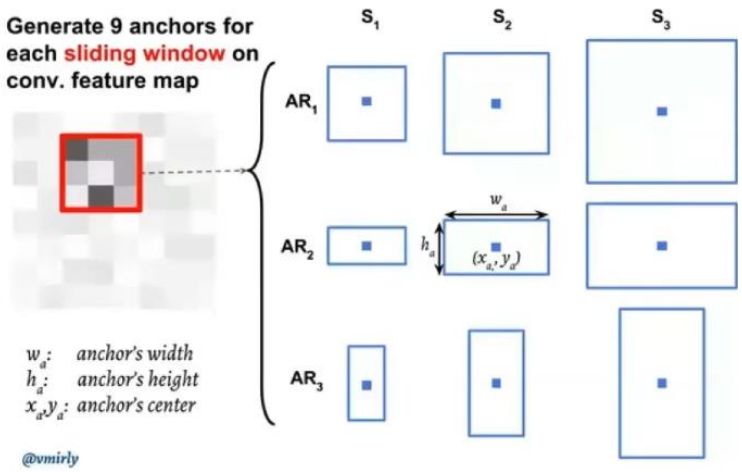

Fig. 8 Workings of anchors. 9 anchors declared at each spatial location of final feature map(s) and classification score for each anchor is predicted. Regression vector will include information about a score for objectness, a score for each class (n), 4 regression coordinates(x_a, y_a, w_a, h_a): 1 + n + 4 = 5 + n. And these scores are repetad for each anchors so final vector will be 9(5+n).*

E. Training Losses

Since DCNNs are mostly trained with SGD, loss function selection is crucial for optimizing. There are many of them in the literature and losses are generally defined for specific tasks such as objectness, classification and bounding box regression. There are also specialized versions of them to handle class imbalance.

Classification losses: Cross entropy (CE) is a classical approach for binary classification tasks and also used to learn objectness in detection pipelines. In Eq. 1 y is the label for a sample and if there's an object it becomes 1, otherwise 0. Also, p is a confidence score of the prediction.

$$CE(p,y) = \begin{cases} -log(p) & \text{if } y = 1 \\ -log(1-p) & \text{otherwise} \end{cases} \qquad (1)$$

There's also multi-variate version of CE (Eq. 2), $p_{o,c}$ predicts the probability of observation o being class c where c ∈ {1, …, C} and $y_{o,c}$ is 1 if an observation o belongs to class c, 0 otherwise. c=0 is a default value for background class which mean there's no object.

$$MCE(p,y) = -\sum_{c=0}^{C} y_{o,c} \log(p_{o,c}) \qquad (2)$$

Regression losses: Mean Saquared Error (MSE) also known as L2 loss can be calculated as:

$$\mathcal{L}_{MSE} = \frac{1}{M} \sum_{i=0}^{M} (y_i - \hat{y}_i)^2 \qquad (3)$$

Where y_i is the 4 -dimensional ground truth bounding box coordinates, and $\hat{y}_i$ is the predicted 4 -dimensional bounding box coordinates. L1 loss, on the other hand, known as Mean Absolute Error (MAE) can be calculated as following:

$$\mathcal{L}_{MAE} = \frac{1}{M}\sum_{i=0}^{M}|y_i - \hat{y}_i| \quad (4)$$

Finally, the smooth L1 loss, which takes advantage of the strengths of both L1 and L2 loss, is given below:

$$\mathcal{L}_{smooth} = \begin{cases} 0.5x^2 & |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases} \quad (5)$$

Where x is the difference between the predicted and ground truth values of each spatial point. Smooth L1 (or L2 in some versions) loss is more robust to outliers and used in Fast-RCNN [18].

Multi-task loss: Eq. 6 is a defacto equation used for classifying along with regression. It's basically sum of classification loss (i.e. MCE) and regression loss (i.e. L1 smooth) for each region proposal or reference box i. p_i and t_i are class probability and vector predicted for bounding box coordinates {x, y, w, h}. Star sign represent groundt truth values, p_i* is ground truth label for class and t_i* again ground truth vector {x*, y*, w*, h*} for bounding boxes. N_{cls} is number of classes while $N_{reg\ is}$ number of reference boxes (anchors). Regression part of the formula is generally multiplied by balancing parameter λ.

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*) \quad (6)$$

Coordinates of predicted and ground-truth bounding boxes (x, y, w, h, x*, y*, w*, h*) are normalized by anchor coordinates (x_a, y_a, w_a, h_a, x^*_a, y^*_a, w^*_a, h^*_a) to make learning easier (Fig. 6). Normalization and scaling operations are defined in Eq. 7. The whole multi-task loss process is first proposed in Fast-RCNN [18] and used widely in object detection community.

$$\begin{aligned} t_x &= \frac{x - x_a}{w_a}, \quad t_y = \frac{y - y_a}{h_a} \\ t_w &= log\frac{w}{w_a}, \quad t_h = log\frac{h}{h_a} \\ t_x^* &= \frac{x^* - x_a}{w_a}, \quad t_y^* = \frac{y^* - y_a}{h_a} \\ t_w^* &= log\frac{w^*}{w_a}, \quad t_h^* = log\frac{h^*}{h_a} \end{aligned} \quad (7)$$

Class imbalance loss: For both of single and double-stage detectors, there are a lot of predictions for background but a little number of objects (Fig 3). To handle the imbalance between positive and negative classes, first approach to use is Online Hard Negative Mining (OHEM) [19]. Hard negatives are negative predictions with low scores and they contribute learning process more than easy negatives. During learning process, for example between the epochs of neural network training, we can eliminate easy negatives of previous epoch and only use hard negative and positive samples in next epochs. OHEM partly solves the imbalance problem since it has ratio of negatives to positives 3:1.

On the other hand, focal loss proposed in [20] reshapes the loss function so that it can reduce the weight of the easy negative samples and the model can focus the training on the hard negative samples. Focal loss is defined as follows:

$$FL(p,y) = \begin{cases} -\alpha_t(1-p)^\gamma log(p) & \text{if } y = 1 \\ -\alpha_t p^\gamma log(1-p) & \text{otherwise} \end{cases} \quad (8)$$

Where $(1-p)^\gamma$ is a modulating factor and $\gamma \geqslant 0$ is a tunable focusing parameter. Authors of the focal loss paper experiment with values of $\gamma \in [0,5]$. Parameter α is weighting factor between [0,1]. Focal loss (Eq. 8) was found to increase the performance by 3.2 mAP points on MS COCO, in comparison to OHEM on a ResNet50-FPN backbone and 600 pixel image scale [19] (Fig. 9).

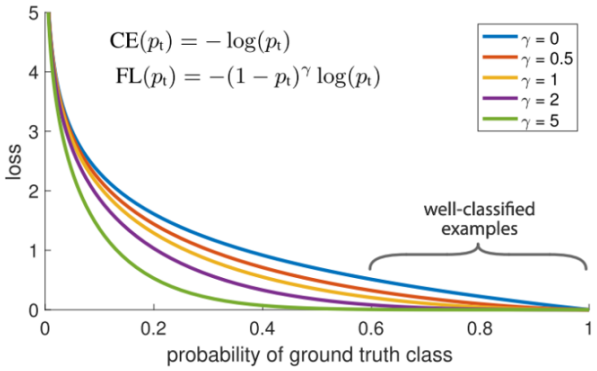

Fig. 9 Effects of gamma parameter on focal loss (Image source: [20]).

F. Inference concept: Greedy and Soft NMS

Object detection models produces high number of overlapping bounding boxes for a single object (Fig. 10). An elimination process is required in inference step and greedy Non-Maximum Suppression (greedy NMS) is most frequent technique [21]. In this approach, the bounding box with the highest confidence is selected and all other boxes having IoU higher threshold λ_{nms} are eliminated. This process will be repeated until all of the boxes are covered. In soft NMS version, instead of eliminating boxes, their scores are reduced. The advantage of soft NMS over greedy one is, it can handle overlapping objects of the same class (Fig. 10). It improved the performance by 1.1 mAP for COCO dataset [22]. Both procedures can be seen in Fig. 11.

Rather than using NMS as post-processing step, there are implementations of learnable NMS [23] which penalizes double detections in loss function. They propose pure CNN implementation of it (GossipNet) and it worked better than greedy NMS and they obtained a performance gain of 0.8 mAP on COCO dataset.

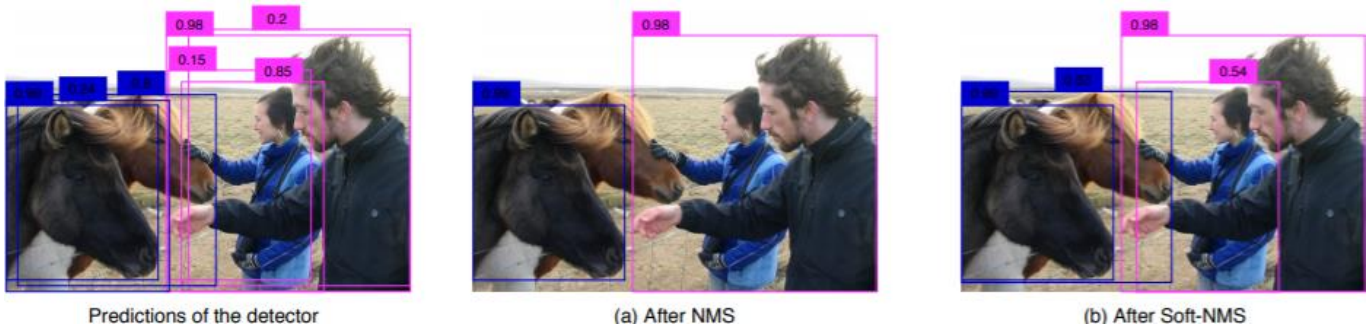

Fig. 10 The scenario of overlapping objects of same classes. Blue boxes belong to horse class, pink boxes belong to person class. On the left side, there are several bounding boxes for each object. In (a) greedy NMS handles the situation but removes the objects behind. In (b) soft NMS covers all of the objects succesfuly.

Algorithm 1 Non-Max Suppression

Input : $\mathcal{B} = \{b_1, .., b_N\}$, $\mathcal{S} = \{s_1, .., s_N\}$, N_t
$\mathcal{B}$ is the list of initial detection boxes
$\mathcal{S}$ contains corresponding detection scores
N_t is the NMS threshold

begin
  $\mathcal{D} \leftarrow \{\}$
  while $\mathcal{B} \neq$ *empty* **do**
    $m \leftarrow$ argmax $\mathcal{S}$
    $\mathcal{M} \leftarrow b_m$
    $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{M}$; $\mathcal{B} \leftarrow \mathcal{B} - \mathcal{M}$
    for b_i *in* $\mathcal{B}$ **do**
      if $iou(\mathcal{M}, b_i) \geq N_t$ **then**
        $\mathcal{B} \leftarrow \mathcal{B} - b_i$; $\mathcal{S} \leftarrow \mathcal{S} - s_i$
      end NMS
      $s_i \leftarrow s_i f(iou(\mathcal{M}, b_i))$ Soft-NMS
    end
  end
  return $\mathcal{D}, \mathcal{S}$
end

Fig. 11 Difference between greedy and soft NMS algorithms is shown in red and green boxes. Soft version reduces scores of overlapping bounding boxes instead of removing them. (original figure is taken from [24])

G. Multiscale Detection

Usually, detection of small objects suffers more than detections of large objects. To handle this issue, different scales of same input images (image pyramids) can be used but it's a computationally costly operation. Another approach, using outputs of different features maps, is used in SSD (Single Shot Detection) paper [25]. In general, deep layers has higher semantic information lower resolutions comparing to shallow layers so, pixels of small objects can be lost while going deeper in network. So shallow layers are more suitable for detecting small objects. To enrich these layers with the semantic information of latter layers, a fusion operation can be implemented. In FPN (Feature Pyramid Network) [27] and RetinaNet [20], a top-

down network parallel to main network with lateral connections which runs simple up-sampling operation. It takes output layer of main network, up-sample it and add them to relevant layer in main network to transfer the semantic information from deeper layers to shallower layers. For final prediction, some may use the only the finest layer of top-down network [26] but FPN and RetinaNet uses all layers of it. Approaches explained here for multiscale detection can be seen in Fig. 12.

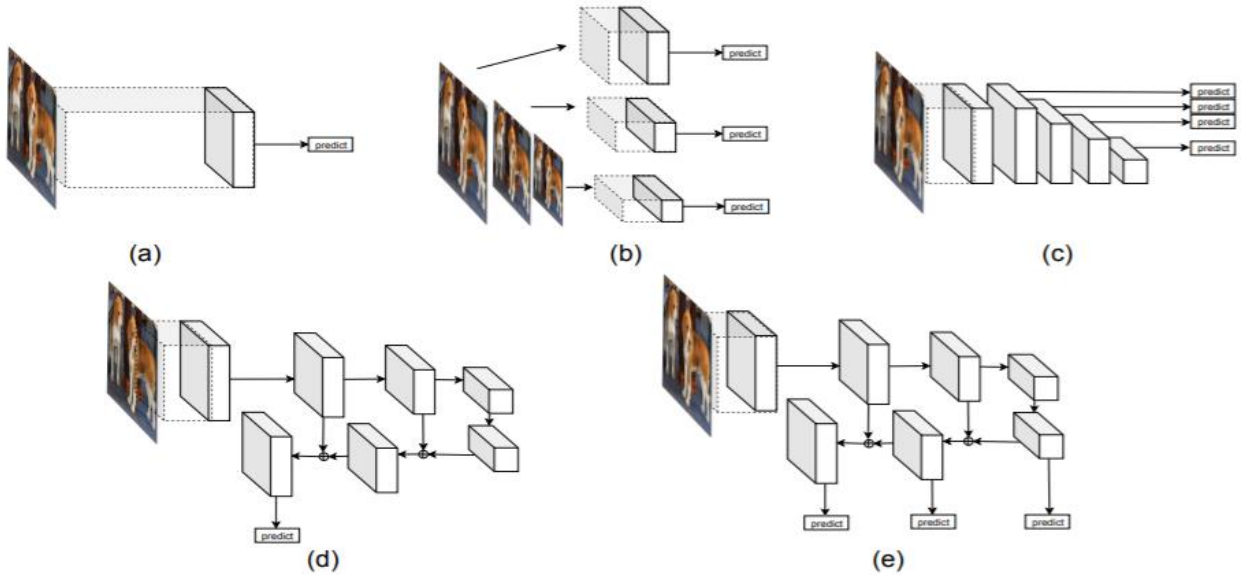

Fig. 12 Modifications of backbone networks to handle multiscale detection problem. (a) simple network, (b) image pyramids, (c) using different feature maps for prediction, (d) top-down network to enrich layers main network with lateral connection but using only finest layer as output, (e) same approach but using several layers as output. (original figure from [12])

H. Double-stage Frameworks

The process of detection objects can be split into two steps: proposing regions and object classification & bounding box regression. The purpose of first step is presenting class-agnostic candidate boxes for next step. And then in the next step, classifier tries to assign class to each proposal and fine-tune the location of bounding boxes. The most famous double stage detectors are proposed under R-CNN family.

R-CNN: R-CNN [33] is short for "Region-based Convolutional Neural Networks". The main idea is composed of two steps. First, using selective search, it identifies a manageable number of bounding-box object region candidates (~2k candidates per image). And then it extracts CNN features from each region independently for classification. These feature vectors are then consumed by a binary SVM trained for each class independently. To reduce the localization errors, a regression model is trained to correct the predicted detection window on bounding box correction offset using CNN features (Fig 12).

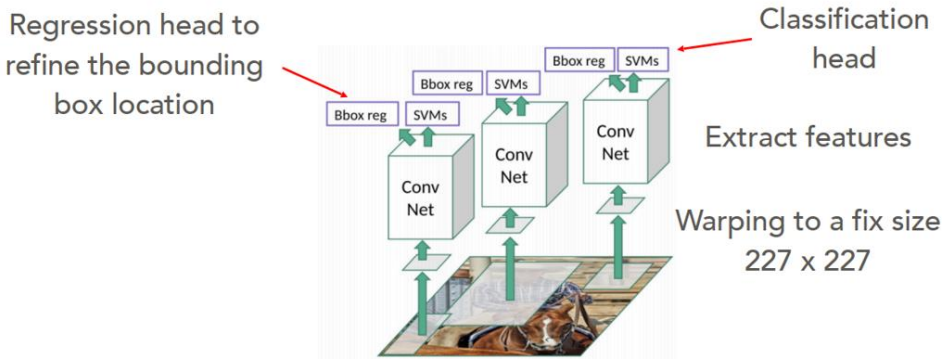

Fig. 13 R-CNN architecture (original figure is taken from Girshick et al. [33]).

Looking through the R-CNN learning steps, you could easily find out that training an R-CNN model is expensive and slow, as the following steps involve a lot of work:

- Running selective search to propose 2000 region candidates for every image.
- Running the CNN every image region (N images * 2000).
- Warping region candidates into fixed size causes distortions heavily.

- The whole process involves three models separately without much shared computation: the convolutional neural network for image classification and feature extraction; the top SVM classifier for identifying target objects; and the regression model for tightening region bounding boxes.

Fast R-CNN: To make R-CNN faster, Girshick (2015) improved the training procedure by unifying three independent models into one jointly trained framework and increasing shared computation results, named Fast R-CNN [34]. Instead of extracting CNN feature vectors independently for each region proposal, this model aggregates them into one CNN forward pass over the entire image and the region proposals share this feature matrix. Then the same feature matrix is branched out to be used for learning the object classifier and the bounding-box regressor. In conclusion, computation sharing speeds up R-CNN (Fig. 14).

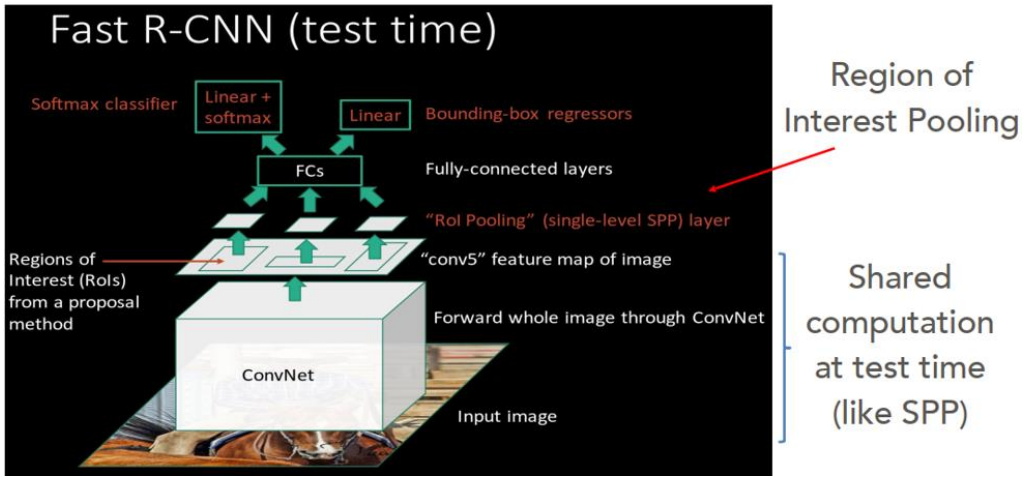

Fig. 14 The architecture of Fast R-CNN (Image source: [34]).

Fast R-CNN is much faster in both training and testing time. However, the improvement is not dramatic because the region proposals are generated separately by another model (selective search again) and that is very expensive. But using RoI Pooling method accuracy is increased around 6 points in mAP.

RoI Pooling is a type of max pooling to convert features in the projected region of the image of any size, h x w, into a small fixed window, H x W. The input region is divided into H x W grids, approximately every subwindow of size h/H x w/W. Then apply max-pooling in each grid (Fig. 15).

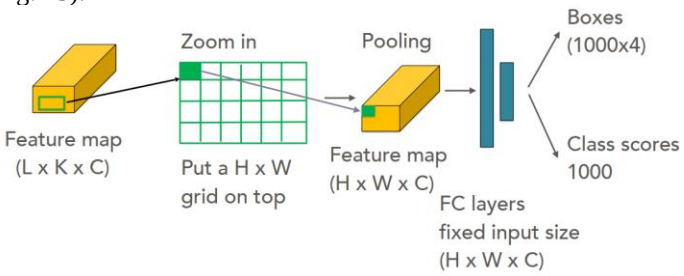

Fig. 15 RoI Pooling method puts a grid on regions of interests to produce fixed sized inputs for fully connected layers.

Faster R-CNN: An intuitive speedup solution is to integrate the region proposal algorithm into the CNN model. Faster R-CNN (Ren et al., 2016 [35]) is doing exactly this: construct a single, unified model composed of RPN (region proposal network) and fast R-CNN with shared convolutional feature layers. RPN slides a small n x n spatial window over the conv feature map of the entire image. At the center of each sliding window, it predicts multiple regions of various scales and ratios simultaneously. An anchor is a combination of (sliding window center, scale, ratio). For example, 3 scales + 3 ratios => k=9 anchors at each sliding position. Then it feeds these RoIs to detection heads (Fig. 17).

Comparing each version of R-CNNs, speed is improving drastically while accuracy stays same (Fig. 16).

	R-CNN	Fast R-CNN	Faster R-CNN
Test time (per image)	50 sec	2 sec	0.2 sec
Speedup	1x	25x	250x
mAP	66.0	66.9	66.9

Fig. 16 Performance comparison of R-CNN family in COCO challenge.

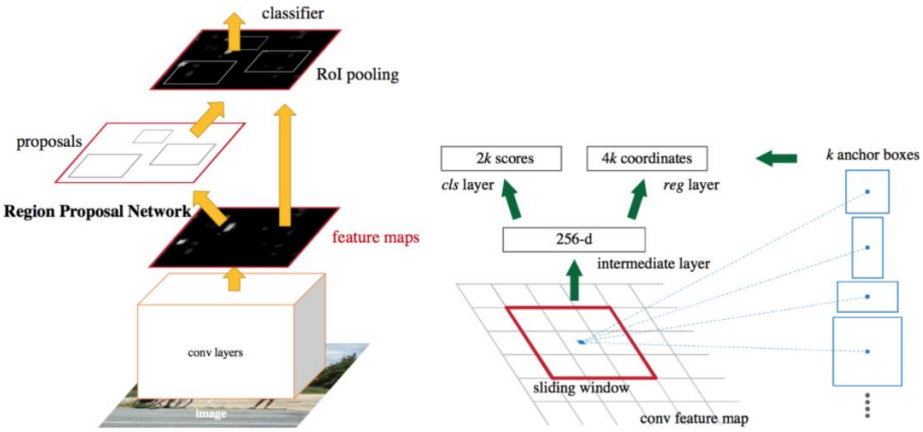

Fig. 17 Architecture of Faster R-CNN on left and achor boxes on right.

İ. Single-stage Frameworks

This part focuses on one-stage models for fast detection, including SSD, RetinaNet and models in the YOLO family. These models skip the explicit region proposal stage but apply the detection directly on dense sampled areas. Two-stage detectors can achieve high accuracy but could be too slow for certain applications such as autonomous driving. One-stage detectors are faster and simpler, but might potentially drag down the performance a bit.

YOLO: The YOLO model ("You Only Look Once"; Redmon et al., 2016 [36]) is the very first attempt at building a fast real-time object detector. Because YOLO does not undergo the region proposal step and only predicts over a limited number of bounding boxes, it is able to do inference in real-time. They started by dividing the image into a S × S grid and assuming B bounding boxes per grid. Each cell containing the center of an object instance is responsible for the detection of that object. Each bounding box predicts 4 coordinates, objectness and class probabilities. This reframed the object detection as a regression problem. To have a receptive field that covers the whole image they included a fully connected layer in their design towards the end of the network (Fig. 18).

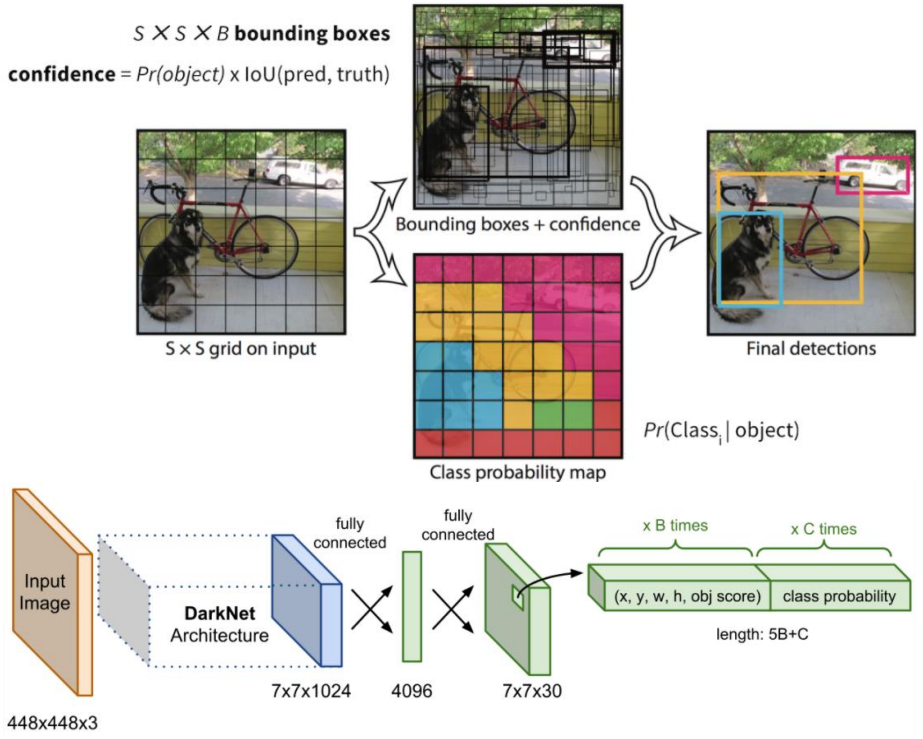

Fig. 18 The workflow of YOLO model. (Image source: original paper [36]).

As a one-stage object detector, YOLO is super-fast, but it is not good at recognizing irregularly shaped objects or a group of small objects due to a limited number of bounding box candidates.

SSD: The Single Shot Detector (SSD; Liu et al, 2016 [37]) is one of the first attempts at using convolutional neural network's pyramidal feature hierarchy for efficient detection of objects of various sizes. SSD uses the VGG-16 model pre-trained on ImageNet as its base model for extracting useful image features. On top of VGG16, SSD adds several conv feature

layers of decreasing sizes. They can be seen as a pyramid representation of images at different scales. Intuitively large fine-grained feature maps at earlier levels are good at capturing small objects and small coarse-grained feature maps can detect large objects well. In SSD, the detection happens in every pyramidal layer, targeting at objects of various sizes (Fig. 19).

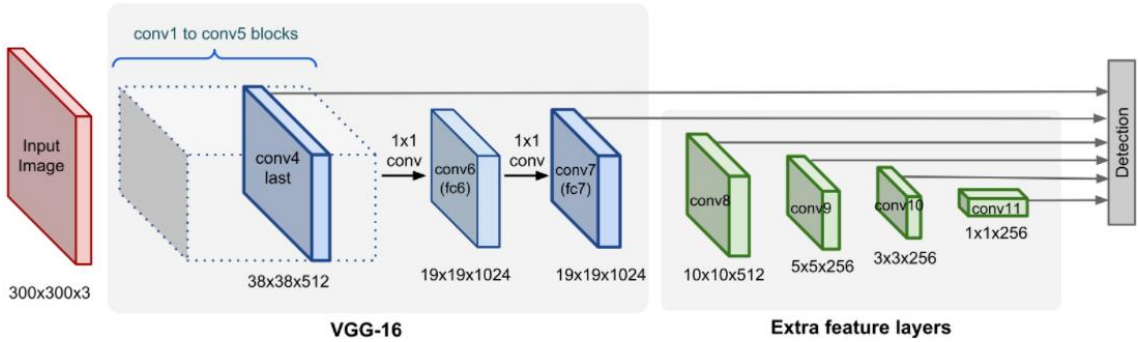

Fig. 19 The model architecture of SSD (image source: [38]).

Improvements on YOLO: A variety of modifications are applied to make YOLO prediction more accurate and faster, including:

- In YOLOv2/YOLO9000 [38] several tricks to increase accuracy are implemented: Batch normalization, high resolution images, using anchors, k-means clustering of anchor box dimensions, direct location prediction, adding fine-grained features from earlier layers, image pyramids and light weighted backbone model (DarkNet-19) [39].
- In YOLOv3 [40] these design tricks are included: Logistic regression for confidence scores instead of softmax, residual DarkNet-53 as backbone network, feature pyramid for multi-scale prediction, skip-layer concatenation like FPN and RetinaNet networks [39].

RetinaNet: The RetinaNet (Lin et al., 2018 [20]) is a one-stage dense object detector. Two crucial building blocks are featurized image pyramid and the use of focal loss. As explained in chapter *G. Multiscale Detection*, featurized image pyramid is similar to feature pyramid of SSD but has improvements like concatenating top-bottom and bottom-top pathways to increase the accuracy of multiscale detection. Additionally, focal loss is a loss function which is specialized for class-imbalanced situations and explained in the chapter *E. Training Losses.* By using these two design tricks, RetinaNet achieves top performance on single-stage detectors in 2017 (Fig. 20).

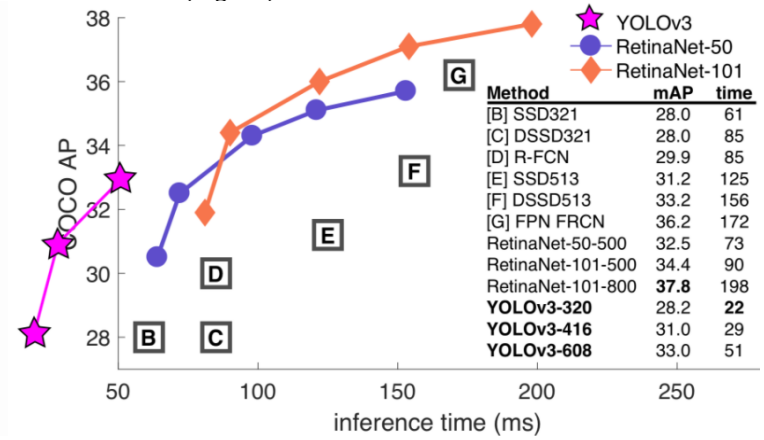

Fig. 20 Performance comparison of RetinaNet, YOLOv3, SSD, FPN and other one-stage detectors.

IV. Conclusions

In the object detection literature, major challenges are scale variance, rotational variance, domain adaptation, object localizations, occlusions, detecting small objects and researchers proposed several tricks explained in this report. There are also more recent works that are not covered in this article like CornerNet [41], CenterNet [42] which uses new ideas like heatmaps and embeddings, Mask R-CNN [43] for image segmentation tasks, methods that use graph networks, adversarial networks and learnable NMS [44].

This report is written for the final project of CSE4084 Multimedia Systems course in Marmara University and aiming to cover foundational papers of object detection literature. Mainly classical approaches, deep learning detectors along with their design tricks are explained by using original papers and web blogs. Main accomplishments of this project are learning and examining the architectures, algorithms of design trick / heuristics and gaining knowledge on state-of-art results. It can be pointed out that working this project was very helpful on introducing object detection domain and can be expanded in to video object detection, image segmentation, object tracking, 3D scenes and even more on autonomous driving domains as future work.

V. Revised Project Plan

Week 1 (Nov 16-22): Literature search and investigating approaches
Week 2 (Nov 23-29): Literature search and investigating approaches
Week 3 (Nov 30-Dec 6): Main concepts in object detection models
Week 4 (Dec 7-13): Midterms
Deadline (Dec 14): Midterm Report
Week 5 (Dec 14-20): Main concepts in object detection models
Week 6 (Dec 21-27): Single-stage networks
Week 7 (Dec 28-Jan 3): Double-stage networks
Week 8 (Jan 4-10): Classical approaches
Week 9 (Jan 11-17): Conclusions
Week 10 (Jan 18-22): Preparing Final Report and Presentation
Deadline (Jan 22): Final Report and Presentation

References

[1] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Berkeley DeepDrive. [Online], Available: https://bdd-data.berkeley.edu/ (Date of Access 14 / 12 /2020)

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, L´eon Bottou, and Kilian Q. Weinberger, editors, Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States, pages 1106–1114, 2012. URL http://papers.nips.cc/paper/4824- imagenet-classification-with-deepconvolutional-neural-networks

[3] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Doll´ar, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In Computer Vision - ECCV 2014 -13th European Conference, Zurich, Switzerland, September 6-12, 2014, pages 740–755, 2014.

[4] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. International Journal of Computer Vision (IJCV), 88(2):303–338, 2010.

[5] Lil'log, Object Detection for Dummies Part 2: CNN, DPM and Overfeat. [Online], Available: https://lilianweng.github.io/lil-log/2017/12/15/object-recognition-for-dummies-part-2.html (Date of Access 15 / 12 /2020)

[6] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. CoRR, abs/1704.04861, 2017. URL http://arxiv.org/abs/1704.04861

[7] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. CoRR, abs/1409.1556, 2014. URL http://arxiv.org/abs/1409.1556

[8] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al. Going deeper with convolutions. In IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015, pages 1–9, 2015.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas,NV, USA, June 27-30, 2016, pages 770–778, 2016.

[10] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inceptionv4, inception-resnet and the impact of residual connections on learning. In AAAI, volume 4, page 12, 2017.

[11] Jia Deng, Wei Dong, Richard Socher, LiJia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA, pages 248–255, 2009.

[12] Agarwal, Shivang, Jean Ogier Du Terrail, and Frédéric Jurie. "Recent advances in object detection in the age of deep convolutional neural networks." arXiv preprint arXiv:1809.03193 (2018).

[13] COCO Common Objects in Context, Detection Evaluation. [Online], Available: https://cocodataset.org/#detection-eval (Date of Access 15 / 12 /2020)

[14] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards realtime object detection with region proposal networks. In Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada, pages 91–99, 2015.

[15] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, ChengYang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, pages 21–37, 2016.

[16] Redmon, J., and A. Farhadi. "YOLO9000: Better, faster, stronger. arXiv 2016." arXiv preprint arXiv:1612.08242 (2016).

[17] Mark Everingham, John Winn, The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Development Kit. [Online], Available http://host.robots.ox.ac.uk/pascal/VOC/voc2012/htmldoc/devkit_doc.html (Date of Access 15 / 12 /2020)

[18] Ross Girshick. Fast r-cnn. In IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015, pages 1440–1448, 2015.

[19] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas,NV, USA, June 27-30, 2016, pages 761–769, 2016.

[20] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Doll´ar. Focal loss for dense object detection. In IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017, pages 2999–3007. IEEE Computer Society, 2017.
[21] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA, volume 1, pages 886–893, 2005.
[22] Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S Davis. Soft-nms—improving object detection with one line of code. In IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017, pages 5562–5570, 2017
[23] Jan Hendrik Hosang, Rodrigo Benenson, and Bernt Schiele. Learning non-maximum suppression. In 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, pages 6469–6477, 2017.
[24] Sambasivarao. K, Non-maximum Suppression (NMS), Medium. [Online], Available: https://towardsdatascience.com/non-maximum-suppression-nms-93ce178e177c (Date of Access 15 / 12 /2020)
[25] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, ChengYang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, pages 21–37, 2016.
[26] Abhinav Shrivastava, Rahul Sukthankar, Jitendra Malik, and Abhinav Gupta. Beyond skip connections: Top-down modulation for object detection. CoRR, abs/1612.06851, 2016. URL http://arxiv.org/abs/1612. 06851.
[27] Tsung-Yi Lin, Piotr Doll´ar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, volume 1, page 4, 2017.
[28] Radu Timofte and Luc Van Gool. “Sparse Flow: Sparse Matching for Small to Large Displacement Optical Flow”. In: Proc. of the IEEE Winter Conference on Applications of Computer Vision (WACV). 2015.
[29] Felzenszwalb, P.F., Huttenlocher, D.P. Efficient Graph-Based Image Segmentation. International Journal of Computer Vision 59, 167–181 (2004). https://doi.org/10.1023/B:VISI.0000022288.19776.77
[30] P. A. Viola, M. J. Jones, and D. Snow. “Detecting pedestrians using patterns of motion and appearance”. In: International Journal of Computer Vision (IJCV) 63(2) (2005), pp. 153–161.
[31] N. Dalal and B. Triggs. “Histograms of oriented gradients for human detection”. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). 2005.
[32] Learn OpenCV, Histogram of Oriented Gradients. [Online], Available: https://www.learnopencv.com/histogram-of-oriented-gradients (Date of Access 05 / 02 /2021)
[33] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014, pages 580–587, 2014.
[34] Ross Girshick. Fast r-cnn. In IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015, pages 1440–1448, 2015.
[35] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards realtime object detection with region proposal networks. In Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada, pages 91–99, 2015.
[36] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas,NV, USA, June 27-30, 2016, pages 779–788, 2016.
[37] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, ChengYang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, pages 21–37, 2016.
[38] Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. In 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, pages 6517–6525. IEEE Computer Society, 2017.
[39] Lil’Log, Object Detection Part 4: Fast Detection Models. [Online], Available: https://lilianweng.github.io/lil-log/2018/12/27/object-detection-part-4.html (Date of Access 07 / 02 /2021).
[40] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. CoRR, abs/1804.02767, 2018. URL http://arxiv.org/abs/1804.02767
[41] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8 - 14, 2018, 2018.
[42] Duan, Kaiwen, et al. "Centernet: Keypoint triplets for object detection." *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019.
[43] He, Kaiming, et al. "Mask r-cnn." *Proceedings of the IEEE international conference on computer vision*. 2017.
[44] Hosang, Jan, Rodrigo Benenson, and Bernt Schiele. "Learning non-maximum suppression." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.