Faculty of Mathematics
and Computer Science

Machine learning course (ML)

# Object detection using deep neural networks

Comănac Dragoș-Mihail

*Department of Computer Science, Babeş-Bolyai University*
*1, M. Kogalniceanu Street, 400084, Cluj-Napoca, Romania*
*E-mail: dragos.comanac@stud.ubbcluj.ro*

**Abstract**

At the first glance, object detection may seem like an easy task: just classify and localize various objects in an image, but under this superficial description, there are hidden tens of years of research and complex methods that try their best to solve this field.

This kind of data understanding method such as object detection has various applications in the real world, hence the intense research interests involved.

Nevertheless, the research into this field is far from being complete, and we try to explore the recent milestone methods based on deep neural networks, which pushed object detection further than before and helped object detection to be an important use case in day to day applications.

We hope that this paper could serve as an introductory guide in the vast and complex field of object detection.

*Keywords:*
YOLO; R-CNN; RetinaNet; FCOS; TOOD

## 1. Introduction

Computer vision is a broad scientific field that deals with the extraction of meaningful information from visual data such as images or videos. As a consequence, over time, computer vision has emerged as a key field in the domain of artificial intelligence because it intends to replicate the functions of the human visual cortex. This is possible due to the continuous development of optical hardware, which nowadays can exceed the capabilities of the human eye, but probably more important are the huge quantities of data that are available to more and more people.

There are various methods that can be used in the field of computer vision, such as hand crafted features, but in this context of big data, the most relevant methods are related to machine learning, and more recently to deep learning which is better suited to visual data. Through supervised learning, deeper and deeper neural network models are now able to learn the complex patterns found in large amounts of data, thus they are a good fit for solving computer vision problems.

One such problem is object detection, which consists of locating and classifying objects in an image by producing a set of bounding boxes which contain both information about the spatial coordinates of the object, and information about it's type or class. This problem is relevant in many domains such as automotive, logistics or even assistive technologies, thus there is an intense ongoing research into this topic. Also, it has more potential than simple classification, because the objects are localized and this forces the learning algorithm to look for the very specific patterns that describe the objects, as opposed to classification where other patterns might be learned if the data distribution is unbalanced.

Given the potential of deep neural networks in solving the problem of object detection, the purpose of this paper is to study some recent architectural trends in building object detectors using deep convolutional neural networks (CNN). We also hope that this paper could serve as a guide into this vast and complex domain.

In what follows, we will describe the context of this paper and how it is related to the broader field of object detection in Section 2. Afterwards, we will perform a survey of some recent object detectors that are based on object detection in Section 3 and we compare their performances in Section 4.

## 2. Placement in the general field

The first object detection techniques heavily relied on handcrafted features. These methods were popular before 2014 and they collectively compose what is now the so called "traditional object detection period" [16]. The main innovation that took place in 2014 was that the deep neural networks reached a maturity such that they became a viable solution in general in the field of object detection. As such, the proposed object detectors since then are based on deep neural networks, hence they compose the "deep learning based detection period". The first type of solutions were quite limited, mostly due to the fact that the features were handcrafted, thus they are limited with respect to the numbers of patterns. Nevertheless, they still brought improvements that are still relevant and used by the modern detectors. Some of these innovations include regressing the bounding boxes or performing hard negative mining, which are used intensively in various object detection methodologies. Also, the sliding windows idea was developed during this time, consisting of having a windows that goes step by step over each region of the image and tries to see if there are any objects within that window. This practice as it is, is highly inefficient, but deep CNNs optimize this methodology.

Since the recent research interest has shifted completely on using deep neural networks, we are focusing on this kind of object detection solutions.

### *2.1. Deep neural networks*

The type of neural network that we are interested in falls into the category of supervised learning, which is a subfield of the more general area of machine learning. They are called deep neural networks because they have multiple hidden layers. The neural networks in the methods that are of interest for us use offline learning and are eager inductive learners, meaning that they are trained on a given dataset on which the correct output is known, in order to make predictions on a dataset on which the correct output is unknown.

### *2.2. One and two stage object detectors*

Broadly speaking, in the context of deep neural networks, the object detection problem mainly branches out in two-stage object detection which traditionally was slow, but accurate and one-stage object detection which initially was less accurate and fast, but recently they became better and better, even surpassing the performance of two-stage object detectors. Also, there were significant advances brought in two-stage detectors, that made them faster and very accurate.

Initially the task of object detection was decomposed into multiple tasks, that together made a pipeline, which is difficult to train. For example, region-based object detectors such as Regions with CNN features (R-CNN) [4] and its faster variants first generate bounding boxes through selective search, then a CNN extracts features that are further used in classifying the bounding boxes. All these steps slow down performance. This approach is called two-stage object detection and it opened the way into the deep learning era of object detection.

Their counterparts, the one stage, or single-shot, object detectors achieve real-time speed with decent accuracy because their detection pipeline consists only of one CNN that processes the image and directly outputs the predictions.

This approach used to have low accuracy due to the lack of large amounts of data, but recent advances have made the single-shot detectors rival the two-stage detectors in terms of accuracy, without losing speed because they benefit hardly from the end-to-end learning which allows the deep CNN to learn the various complex patterns directly from the data. Here, the You Only Look Once (YOLO) [10] method was one of the first methods to belong to the one-stage class of object detectors, hence it's success in achieving good performance, with little resources.

Also, more recently, another way of splitting the class of object detectors has emerged. Traditionally, anchors are a general way of simplifying the object detection task, and it can be applied in other tasks as well. The idea is that under ideal conditions, the machine learning algorithm can learn to predict in a specific area of an image any object with various shapes. But as it is usually the case, this proves to be difficult due to the lack of data and optimization problems. An elegant way to address this issue, is to use anchors which follow the divide et impera principle. Basically, instead of having only one predictor that focuses on various shapes, with anchors, there are several predictors that each can learn a specific shape.

For instance, if there are two object that are overlapping, and one is very long and thin, and the other is very short and wide, it would be hard for one predictor to handle both objects, but using two anchors, one for each object, it is easier. Another topic in which there is research interest invested into recently, is how positive and negative examples are defined. This is important to the learning process, because the positive examples contain the important information and they are crucial. Therefore, the trend is to build denser and denser object detectors in order to be able to cover as much of the image as possible and have a reasonable recall.

## 3. Survey of recent object detectors based on deep convolutional networks

In this section we will review some of the more recent object detection techniques. Since a lot of the recent research interest has been invested in one-stage object detectors, we will also focus mainly on this class of methods. Another reason for this is that this is commonly used in practical applications, because often the detection systems must be fast and one-stage detectors are build around speed. Thanks to recent advances, they are also as good, if not better in terms of accuracy than the two-stage object detectors.

### *3.1. General architecture*

The usual practice is to build one-stage object detectors using a single deep CNN. The main advantage of this approach is that it allows the object detection system to learn end-to-end the complex patterns directly from the data in order to predict the bounding boxes directly from the entire image. The downside is that it requires a considerable amount of data. Another approach is to use vision transformers, but this kind of method, even though it is one of the best when it comes to accuracy, it consumes a lot of resources because they usually have billions of parameters. In comparison, deep CNNs have usually millions of parameters, thus it is easier to deploy a CNN on an real-time industry application.

It is a common practice to split the neural network architecture in three parts, as we have depicted in Fig. 1. Each section of this general architecture are important and deserves comprehensive studies on their own. It is interesting to note, that this network architecture can be easily used with other tasks such as segmentation, only by changing the head. Another interesting use case is to use multiple heads, for different tasks, in order to form what is called a multi-task network.

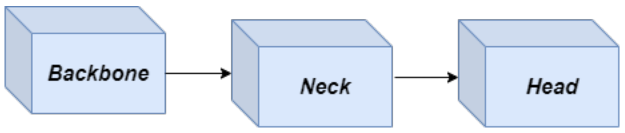

Fig. 1. General neural network architecture for one-stage object detectors

The role of the backbone is to extract feature maps from the original image, that describe for instance various shapes which can help in determining the location and class of the possible objects. Depending on the precision and speed requirements the specific design of the backbone can vary from small networks, used for speed, or large networks, used for better precision. The actual design can be custom, but if there is not enough data to accommodate the required size of the network, a common practice is to use transfer learning.

The idea is to use an already existent network architecture that is trained on a large amount of diverse data. Usually, a network used for classification is used by replacing the last layers with the neck and head architecture specific for the given problem. In this way, the extracted knowledge, such as basic or complex shapes, from the larger source dataset is stored in the network and it can be reused on the target dataset. How many layers are cut from the image classifier neural network depends on how similar are the target and source dataset. Also, regarding training, the first step is to train the whole network with the backbone frozen in order to not break the knowledge stored in its weights and the second step is to train the whole network with the backbone unfrozen, but with a low learning rate, in order to better fit the target dataset.

For example a ResNet like architecture [5] can be used if high precision is needed, or a MobileNet like architecture [6, 14] is suitable if speed is of the essence. The ResNet architectures were able to bring deep learning to another level by introducing the residual block which allows a network to be very deep by optimizing how gradients flow. One residual block is formed by three convolution layers, which have an extra connection which adds the first layer to the output of the last. In this way even if there are not any gradients flowing on the main path, there will still be some on the side path and if the network does not need the extra connection, it can just learn the identity function. But, a lot of stacked residual blocks can slow down the network, and in comparison, MobileNets are fast because they use depthwise separable convolutions, which are optimized convolution by computing first a normal convolution, but on channels with kernels with one channel, then a pointwise convolution in order to control the output number of channels. This greatly reduces the number of operations. The usual practice when developing a new object detection technique is to use one of these architectures because in this way it can be fairly compared to other methods from the literature, because they also usually use one of the two.

The neck is an optional component and its main role is to be an intermediary between the backbone and the head, such that it further refines the features from the backbone in order to provide more information for the head. An example for the neck architecture is the Feature Pyramid Network [7]. The idea is that there are several feature maps extracted from the backbone that are provided to the head, in order to let the network to ”see” at several resolutions.

If the other two components are not necessarily that specific to object detection, the head is responsible for taking the output of the neck (or directly from the backbone if the neck does not exist), optionally refine them furthermore, and finally converting them to the desired representation. This is the part where YOLO method contributes the most. Basically, it describes how the head should convert a feature maps into a representation from which bounding boxes can be extracted.

## *3.2. You Only Look Once*

As we have mentioned earlier, YOLO is one of the first one-stage object detectors and it is a widely used method for performing object detection. In YOLO, the object detection problem is treated as a regression problem. There is only one neural network that predicts the bounding boxes and class probabilities from an image. This way, the network can benefit from using end-to-end learning, and the inference time is greatly reduced, thus achieving real-time performance.

Beside speed, the relatively easy to implement architecture is another reason for its popularity, or the fact that usually the neural networks that implement this architecture can be quite small, therefore they are suitable for deployment on devices with limited computing power.

The main idea behind YOLO is that in order to detect multiple objects, at various locations, the image is split in a grid in which each cell is responsible for detecting objects that appear inside it. In fact, this is the representation of the ground truth labels.

The ground truth is represented as a $C \times C \times B \times (5 + N)$ tensor. For each cell in the $C \times C$ grid, there are B anchor boxes associated. Each anchor box has the following parameters: $b_x$ and $b_y$ represent the center of the box and they are both relative to the cell responsible for predicting the object, meaning that they are divided to the size of the cell, $b_w$ and $b_h$ represent the width and height, c is the probability that the anchor predicts an object and $c_1, c_2, c_3, ...c_n$ represent the $N$ class probabilities. The actual choice for the size of the grid, the number of anchors or the number of classes is problem specific. They are important because they determine the maximum number of positives in an image. The anchors can be set manually, but the original authors used K-Means over the whole dataset in order to generate the anchors.

Regarding the network design, the final layer has the same structure as the ground truths. One way this can be implemented is with a 1x1 (pointwise) convolution. In order to do this, the layer before the pointwise convolution should output a feature map with a spatial size of $C \times C$. The number of channels does not matter because it is controlled by the pointwise convolution which has $B * (5 + N)$ kernels in order to output a feature map of size $C \times C \times (B * (5 + N))$ which can be further reshaped to a tensor of shape $C \times C \times B \times (5 + N)$ in order to match the shape of the ground truth.

The success of the YOLO method is given by the robustness it has shown through time. Over the years, several methodologies have been proposed that use YOLO as the key element. Each brings different optimizations over the original method, but the main aspects are still relevant. For example, the second version of YOLO [11] mainly introduces the notion of anchors and a novel multi-scale training method in order to have good predictions across images of various scales. The third version of YOLO [12] represents mostly a bundle of small improvements.

In a more modern interpretation, YOLOX [2] turns the problem into an anchor free one, by following the modern guidelines in object detection.

### 3.3. *Region based convolutional neural networks*

This type of method falls into the category of two-stage object detectors. Traditionally, there was a clean separation in terms of performance between this type of object detectors and the one stage class of object detectors, but with time, both types of methods became better in what they lacked by bringing several optimizations over the original architecture. Two stage methods gained speed and one stage methods gained accuracy for example.

In Fig. 2 we can observe why this method is a two-stage type of method. Basically, the first stage is to extract from the input image a lot of region proposals or region of interests. Ideally, these region proposals contain objects of interest and will represent the final bounding box. Here we can observe the fundamental difference between two-stage methods and YOLO, or one-stage methods in general. In one-stage methods, the bounding boxes are obtained through regression, while in two-stage methods, they are extracted beforehand. It is important to note that R-CNN is agnostic to the way region proposals are extracted from the input image, as the authors mention. One way the region proposals can be extracted is by selective search, which is the method adopted originally by the authors and it does not involve any learning. Another way is to obtain them through a what is called a region proposal network that is a fully convolutional network capable of predicting both bounding boxes and objectness scores.

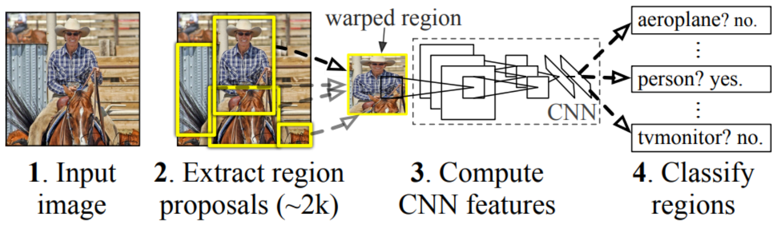

Fig. 2. Two stage pipeline used in R-CNN [4]

Region proposal is a complex domain and it deserves focused research on it's own. The idea is that it is crucial for two-stage method to have good proposals in order to achieve high performance. This could be seen as a downside compared to it's counterparts, the one-stage methods, because it adds a layer of extra complexity. Also, the density of region proposals is crucial, this also being related to a more general problem in object detection, that of defining positives. Thus, if there aren't enough proposals, the network performance will be limited in the sense that it will not be able to find all objects and the recall can be affected.

The second step is to take the region proposals and classify them. Optionally, the region proposals can be further refined before being classified. The classification can be done in several ways such as with artificial neural networks, convolutional neural networks or support vector machines.

The final predictions are composed from the bounding boxes given by the region proposals and the classification scores resulted from the second stage.

This method as it is can achieve good performance in terms of precision but it lacks speed. This is mostly due to the fact that many region proposals are actually overlapping, thus there are a lot of extra computations that have a negative impact on the speed. This was further optimized in Fast R-CNN [3] by introducing a convolutional neural network as a preliminary step before selective search in order to then compute region proposals on the processed feature maps using region of interest pooling. In this way, the feature extraction becomes a shared computation. Another optimization is proposed in Faster R-CNN [13] and it consists of replacing selective search with a region proposal network, that can learn to predict a set of object proposals, together with their objectness scores directly from the extracted feature maps, resulted by passing the input image through a convolutional neural network. These optimizations greatly reduce the inference time, and maintain good performance, but the method still lacks behind one-stage methods such as YOLO in terms of speed.

## *3.4. RetinaNet*

RetinaNet [8] is an important milestone for one-stage object detectors because the authors introduce a novel way of computing the loss, namely the focal loss. This approach proposes to bridge the precision gap between one-stage and two-stage object detectors. After this point, one-stage methods seem to dominate the literature.

The authors argue that the main disadvantage the one-stage methods had against two-stage ones, was the imbalance between the background and foreground classes which heavily impacted dense object detectors during training. Two-stage methods address this issue by carefully designing the region proposal stage. In contrast, dense object detectors have to predict many more regions. The aim of the Focal loss is to mitigate this problem by focusing on the hard examples, rather than leaving the large amount of easy predictions overwhelm the training.

The class imbalance problem impacts the neural network learning in several ways. If class imbalance is not considered, the network learning will eventually enter a plateau due to the fact that it can detect easy classes and the overall performance will be good. Also, small losses from a large amount of easily predicted objects can overwhelm the gradients, and the loss from misclassified hard examples will not have the desired impact. For example, if there are two classes, the loss of a high confidence negative class is $-log(0.95)$ for example and the loss of a confident positive class is also $-log(0.95)$. Hence, if the classes are imbalance, they still contribute the same in the overall loss, and it is easy to see why multiple examples from the overrepresented class dominate the gradients.

The traditional way to solve this was to perform some kind of dataset alteration such as data augmentation in order to introduce more meaningful examples from the underrepresented class. It is not trivial to introduce such methods, because the new data must follow the same distribution as the original one, but the main innovation of focal loss is that it solves class imbalance without changing the dataset.

Mathematically speaking the authors name this loss function a ”dynamically scaled cross entropy loss”. Traditionally, when it comes to classification, the cross entropy loss was a popular choice, but as we have previously mentioned, the classical formula does not account for the heavy class imbalance. In Fig. 3 we can see the exact difference between the cross entropy loss and focal loss.

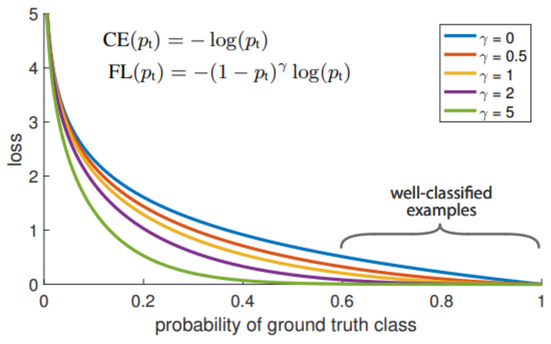

Fig. 3. Cross entropy and focal loss [8]

It is easy to notice that the cross entropy loss function is just a particular case of the focal loss function when $\gamma$ is zero. The key-point to note is the $\gamma$ parameter which is the power of the extra term. Other kinds of losses try to lower the loss from outliers, but the focal loss has the opposite role, that of giving a small total value in the final loss of a large number of easy examples. In this way, the gradients are not dominated by a large number of small losses from the easy examples. As such, in simpler terms, the main drawback of cross entropy is that it treats all classes the same, and focal loss treats them differently.

Geometrically speaking, we can tell from the graph that for the cross entropy (blue line) the loss is low for probabilities larger than 60%. By increasing $\gamma$, this probability is relaxed, and for $\gamma = 5$ if the probability is above 30% the loss is low. This is good, because in this way the easy examples have a lower impact in the total loss. Another way of seeing this is that a model using the standard cross-entropy, would still penalize examples with a high probability, but with focal loss, the model would not try to bring the probabilities close to one, but focus on improving the hard examples, which will have a larger magnitude in the final loss.

The formula as it is in Fig. 3 only handles the problem of balancing easy and hard examples, but the authors introduce another parameter $\alpha_t$ and the formula becomes:

$$FL(p_t) = -\alpha_t \cdot (1 - p_t)^\gamma \cdot log(p_t) \tag{1}$$

Essentially, $\alpha_t$ is a weighting factor that controls the impact on the loss of each class. For instance, each $\alpha_t$ can take as value the inverse of class frequency.

## *3.5. Fully convolutional One-Stage Object Detection*

We have previously stated the importance of anchors in solving the object detection problem, but their main drawback is that they introduce extra hyper-parameters that impact the performance and are hard to tune. Furthermore, anchor-based solutions heavily rely on the number and shape of anchors, meaning they don't scale well, especially when the object shapes have a high variance. Hence recent researches propose anchor free solutions as a better alternative to the classical approach using anchors, hence removing the hassle of tuning extra hyper-parameters and some extra computations involving anchors.

Fully Convolutional One-Stage Object Detection (FCOS) [15] is one of the first anchor-free solutions that do not require complicated post-processing and has a good recall. Before FCOS, object detection was one of the few computer vision tasks that deviated from the "fully convolutional per-pixel prediction" due to the dependency on anchors. As such, this is exactly what FCOS tries to do, to solve object detection in a similar fashion to semantic segmentation and it manages to be better than anchor-based detectors in terms of performance.

The authors argue that there are several benefits from using a fully convolutional anchor free neural network. Firstly, the object detection is now modeled similarly to other tasks, thus various ideas can be reused and generalised to instance-wise prediction problems. Also, as we have already mentioned, a lot of extra complexity and computations are removed.

Fig. 4 presents how exactly the predictions are made in per-pixel fashion. First of all, the ground truths are represented by the top-left and bottom-right corners of the bounding box, together with the class index. The idea is that there are several feature maps, each with a stride $s$, and each point with coordinates $x, y$ can be remapped at the original scale as $\frac{s}{2} + x \cdot s, \frac{s}{2} + y \cdot s$. A positive is a point $x, y$ that falls into a ground truth bounding box, otherwise is seen as a negative and it is labeled as background. As in Fig. 4, the distances to the left, top, right and bottom sides are also regressed directly from the point. In this way, the bounding box is regressed directly in a similar fashion to semantic segmentation, as opposed to anchor-based object detectors, which regress the bounding boxes with anchors as reference. It is possible that a point may fall into multiple bounding boxes, and in this case, the bounding box with the smallest area is chosen as the regression target. These kind of points are called ambiguous samples and they are reduced significantly when using multiple levels for prediction.

Having this setup, the authors argue that one of the reasons this works better than the anchor-based ideas is that FCOS uses as many foreground samples as possible to train the network, as opposed to anchor-based methods that

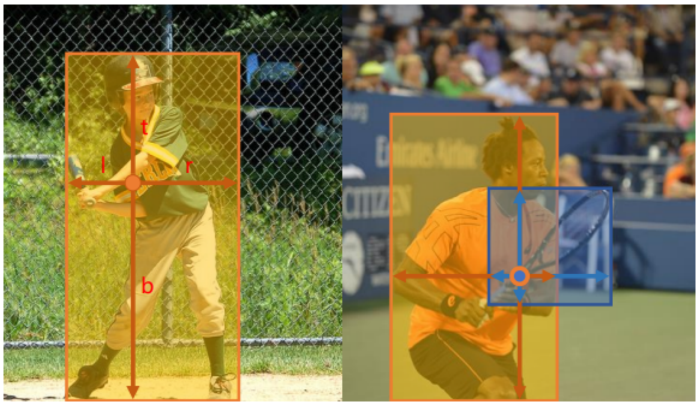

Fig. 4. How FCOS makes predictions [15]

train only on boxes that have the IOU with the anchors above a certain threshold. Also, another point the authors make is that FCOS has nine times fewer outputs variables that other anchor-based methods.

The authors notice that in order to bridge the performance gap between FCOS and anchor-based methods, a center-ness branch must be introduced. The cause is that predictions on multiple levels cause low-quality predictions in locations that are far from the center of the object. As such, another layer, without hyper-parameters is introduced parallel to the classification branch, that predicts the ”center-ness” of a location. Essentially, it is a value between zero and one, that is trained using binary cross-entropy and it is used to down-weight the confidences of boxes that are far from the center of an object, thus the low-quality prediction are filtered by non-maximum suppression. The authors note that this is very important to achieve good performance. An alternative would be to have an extra hyper-parameter that controls the portion of the ground truth box that is seen as a positive, and actually both methods combined yield the best results.

Another advantage of this method is that it can be extended in Region Proposal Networks.

### 3.6. *Task-aligned One-stage Object Detection*

As we have mentioned, object detection is basically a composed task, consisting of the simpler classification and localization. The general approach in solving this composed task, is to solve the individual simpler tasks separately. In the context of deep neural networks, this is achieved by having two parallel branches in the head, one for classification and one for localization. While this works reasonably well, it has its flaws. The main disadvantage is that the two tasks do not work collaboratively in this way, thus bounding boxes with good localization might be discarded during non-maximum suppression only because of the bad score.

Having these arguments in mind, Task-aligned One-stage Object Detection (TOOD) [1] proposes to actively align the two tasks. In this way, a if a predicted bounding box has a good confidence score, it should have a good localization also.

In Fig. 5 we can see the two main components that achieve the alignment. Firstly, the authors design a task-aligned head (T-head). This can be used in any object detector following the backbone-neck-head architecture, which is a great advantage. The other important component is the task alignment learning (TAL), which is independent of T-head and can also be applied to other object detectors, either anchor-based or anchor-free. Actually, these components can be extended to other tasks as well, but not all tasks might be compatible, and it might not scale well with the number of tasks aligned.

The T-head is built to ensure the interaction of the classification and localization branches, as opposed to conventional object detection head for one-stage architectures. The idea is that there is a single branch that makes the predictions using a task-aligned predictor (TAP). The authors note that the problem with this would be that the two task might need to focus on different areas and features of the object, therefore they propose an attention layer to mitigate this issue. Afterwards, the spatial distributions are adjusted, considering both tasks. This adjustment is done using alignment maps which are learned.

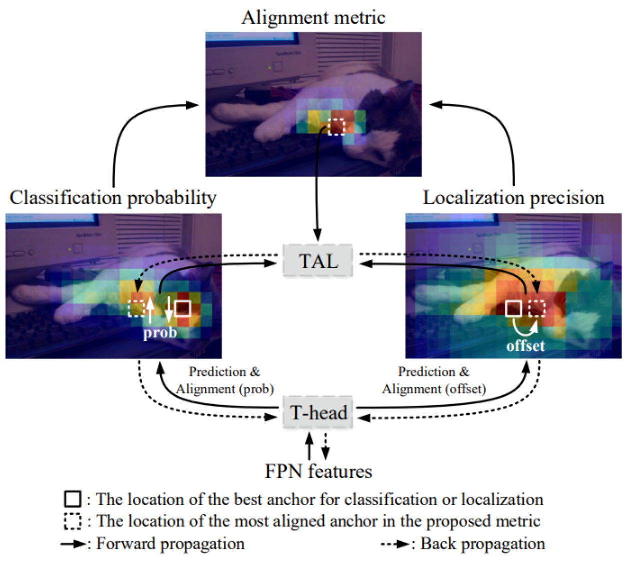
Alignment metric
Classification probability
Localization precision
TAL
prob
offset
Prediction & Alignment (prob)
Prediction & Alignment (offset)
T-head
FPN features
: The location of the best anchor for classification or localization
: The location of the most aligned anchor in the proposed metric
: Forward propagation
: Back propagation

Fig. 5. TOOD learning [1]

While T-head can work quite well on its own, the authors also introduce TAL in order to further guide the network towards aligned predictions. It is mainly composed of selecting qualitative anchors based on some metric and it considers simultaneously anchor assignment and weighting. Therefore, in order to work well with non-maximum suppression, a high confidence box should have good localization, and a low confidence box should have a bad localization, and be discarded. To achieve this, the authors introduce an anchor alignment metric that dynamically encourages the network to focus on qualitative anchors in terms of alignment optimization: $s^\alpha * u^\beta$ where s is the classification score and u is the IOU score, and $\alpha$ and $\beta$ control the importance of each task. As such, for each instance, the first $m$ anchors having the largest anchor alignment metric are chosen. Also, the loss is modified to include this anchor alignment metric.

## 4. Comparison between object detectors in terms of performance

In this section we aim to give a quantitative comparison on standard object detection datasets of the methods described in Section 3.

To measure an object detection system performance usually mean average precision (mAP) is the most relevant measure. Over time, this mAP measure became the standard when it comes to measuring the quality of an object detection system.

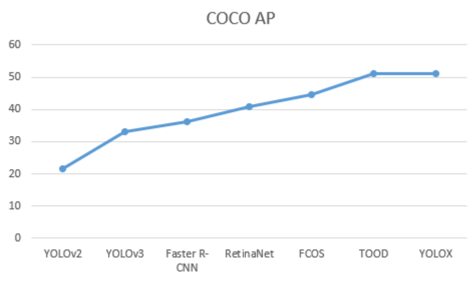

Fig. 6. Performances over the COCO dataset in terms of AP

As the technology advanced, researchers were able to train on larger and larger datasets, thus the Microsoft Common Objects in Context (COCO) dataset [9] with 80 classes became the new standard that new object detection methodologies use in order to be relevant in the field. Also, the authors provide an evaluation server which allows a better comparison between object detectors. The idea is that the labels for the test set are not publicly available, only the images can be downloaded. Therefore, the researchers that propose new object detection methods, have to upload on the evaluation server their results on the test set, and get back a detailed mAP evaluation for various thresholds and for small, medium and large objects.

As such, we perform a comparison on the complex standard COCO dataset in Fig. 6.

## 5. Conclusions

In conclusion, object detection has come a long way and several important advances have been made, such as end-to-end learning, focal loss, anchor-free methods or the alignment of tasks that have pushed further and further the performance of object detectors. Looking towards the future, we believe that many of these ideas can be integrated together to create other object detection methodologies that surpass existing ones.

## References

[1] Feng, C., Zhong, Y., Gao, Y., Scott, M.R., Huang, W., 2021. TOOD: Task-aligned One-stage Object Detection, in: IEEE/CVF International Conference on Computer Vision (ICCV), IEEE Computer Society. pp. 3490–3499.

[2] Ge, Z., Liu, S., Wang, F., Li, Z., Sun, J., 2021. YOLOX: Exceeding YOLO Series in 2021. ArXiv abs/2107.08430.

[3] Girshick, R., 2015. Fast r-cnn, in: Proceedings of the IEEE international conference on computer vision, pp. 1440–1448.

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

[5] He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep Residual Learning for Image Recognition, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778.

[6] Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., 2017. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. CoRR abs/1704.04861.

[7] Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S., 2017a. Feature pyramid networks for object detection, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2117–2125.

[8] Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P., 2017b. Focal loss for dense object detection, in: Proceedings of the IEEE international conference on computer vision, pp. 2980–2988.

[9] Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft COCO: Common Objects in Context, in: European conference on computer vision, Springer. pp. 740–755.

[10] Redmon, J., Divvala, S., Girshick, R., Farhadi, A., 2016. You Only Look Once: Unified, Real-Time Object Detection, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779–788.

[11] Redmon, J., Farhadi, A., 2017. YOLO9000: Better, Faster, Stronger. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) , 6517–6525.

[12] Redmon, J., Farhadi, A., 2018. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767 .

[13] Ren, S., He, K., Girshick, R.B., Sun, J., 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence 39, 1137–1149.

[14] Sandler, M., Howard, A.G., Zhu, M., Zhmoginov, A., Chen, L.C., 2018. MobileNetV2: Inverted Residuals and Linear Bottlenecks. IEEE/CVF Conference on Computer Vision and Pattern Recognition , 4510–4520.

[15] Tian, Z., Shen, C., Chen, H., He, T., 2019. FCOS: Fully Convolutional One-Stage Object Detection, in: Proceedings of the IEEE/CVF international conference on computer vision, pp. 9627–9636.

[16] Zou, Z., Shi, Z., Guo, Y., Ye, J., 2019. Object Detection in 20 Years: A Survey. ArXiv abs/1905.05055.