

Received June 17, 2021, accepted July 26, 2021, date of publication July 30, 2021, date of current version August 5, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3101297

Logo Detection With No Priors

DIEGO A. VELAZQUEZ^{1,2}, JOSEP M. GONFAUS¹, PAU RODRÍGUEZ³, F. XAVIER ROCA²,
SEIICHI OZAWA⁴, (Senior Member, IEEE), AND JORDI GONZÀLEZ²

¹Visual Tagging Services, Bellaterra, 08193 Catalonia, Spain

²Computer Vision Center, Bellaterra, 08193 Catalonia, Spain

³Element AI, Montréal, QC H2S 3G9, Canada

⁴Center for Mathematical and Data Sciences, Kobe University, Kobe 657-8501, Japan

Corresponding author: Diego A. Velazquez (diegoalejandro.velazquez@e-campus.uab.cat)


This work was supported in part by the Generalitat de Catalunya through the Industrial Doctorate Program under Grant 2020DI62, in part by the Spanish Ministry of Economy and Competitiveness (MINECO) and the European Regional Development Fund (ERDF) under Project TIN2015-65464-R and Project PID2020-120311RB-I00 / AEI / 10.13039/501100011033, and in part by NVIDIA Corporation.

ABSTRACT In recent years, top referred methods on object detection like R-CNN have implemented this task as a combination of proposal region generation and supervised classification on the proposed bounding boxes. Although this pipeline has achieved state-of-the-art results in multiple datasets, it has inherent limitations that make object detection a very complex and inefficient task in computational terms. Instead of considering this standard strategy, in this paper we enhance Detection Transformers (DETR) which tackles object detection as a set-prediction problem directly in an end-to-end fully differentiable pipeline without requiring priors. In particular, we incorporate Feature Pyramids (FP) to the DETR architecture and demonstrate the effectiveness of the resulting DETR-FP approach on improving logo detection results thanks to the improved detection of small logos. So, without requiring any domain specific prior to be fed to the model, DETR-FP obtains competitive results on the OpenLogo and MS-COCO datasets offering a relative improvement of up to 30%, when compared to a Faster R-CNN baseline which strongly depends on hand-designed priors.

INDEX TERMS Object detection, transformers, logo detection, deep learning, attention.

I. INTRODUCTION

The field of object detection has improved exponentially in recent years with the advent of R-CNN [1] and its several improvements, which eventually became the standard for object detection in the Machine Learning and Computer Vision communities. The key strategy of R-CNN is to combine proposal region generation with a supervised classification applied on those proposed bounding boxes. As a result, different variations of the R-CNN framework were proposed, resulting in a multitude of different approaches that can be considered to belong to the same family of one-stage object detectors: YOLO [2]–[4], CenterNet [5], SSD [6], RetinaNet [7], FoveaBox [8], Anchor Pruning [9], EfficientDet [10] to name a few. Without going into details about the differences in performances in terms of precision and speed of each model, it is safe to say that they all performed really well across different object detection benchmarks at the time of their publication. However all of these approaches are

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

constrained to the same limitations that are intrinsic to object detection when implemented as a supervised classification on proposed regions.

A. OBJECT DETECTION LIMITATIONS

The best-known object detection approaches mentioned above are very complex, they contain hundreds of hyper-parameters, lots of caveats and post processing steps. *But, why is all of this required?* Object detection is indeed a set-prediction problem in itself, where there exists an unordered set of bounding boxes that should be associated with a category label, and any detector is tasked with predicting said set. However, since neural networks exceed at classification and regression tasks, object detection problem has been treated as a box-classification problem.

So let us denote C as the set of possible classes, with the addition of 0 as the background class, and the set of boxes as \mathbb{B} in an image $I \in \mathbb{I}$. Then the problem involves learning a mapping ($f : \mathbb{I} \rightarrow 0, \dots, C^{|\mathbb{B}|}$) that correctly classifies each box as a background or object class. However, the number of boxes in an image belonging to a class $0, \dots, C$ is infinite,

so this box-classification modeling decision brings about a series of inherent limitations that we will now discuss.

1) TOO MANY BOXES

As previously mentioned, the first problem is that there are infinite possible boxes in the image space. A solution is to approximate (*quantize*) this infinite space into a set of representative boxes. However, this approximation introduces a quantization error so the classification might be correct but the localization inaccurate. To address this, a new regression task is usually introduced: this regressor is responsible for predicting any quantization error and transforming inaccurate quantized boxes into better approximations, see Figure 1. Thus, in the end there is a switch to a proxy problem that now involves both a classification and a regression task.

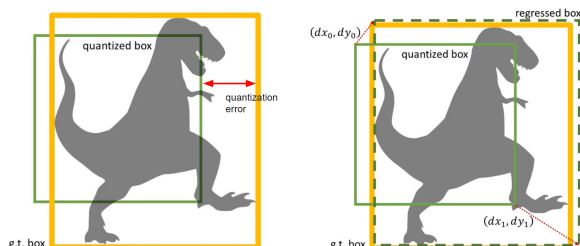


FIGURE 1. The quantization error that arises from quantizing the set of possible boxes (left) and its solution as a regression problem (right).

2) CLASSIFICATION RULE

Notice how the classification rule defined above ($f : \mathbb{I} \rightarrow 0, \dots, C^{|\mathbb{B}|}$), assumes that the model is able to predict the ground truth box perfectly. However it is extremely unlikely that the ground truth box will be within the set of quantized boxes. So, the classification rule is no longer defined for such quantized boxes, but instead a label assignment rule must be defined (e.g., IoU thresholds, centeredness, containment), like: if the quantized box has an IoU > 0.5 with the ground-truth bounding box, then it is not considered background.

3) REDUNDANT OUTPUTS

The implication of the previously defined assignment rule is that in fact now there may be many highly-overlapping boxes that will meet such criteria, leading to redundant detections. A set-level operation is then required to resolve this redundancy. The standard solution to this is decades old, it involves clustering this overlapping detections into a single one: typically Non-Maximum suppression (NMS) is used to this end.

4) FOREGROUND-BACKGROUND IMBALANCE

The last challenge is independent from the previous three, and it has to do with the inherent imbalance nature of the classification task at hand. Before quantization, detection is infinitely imbalanced, since there are infinite background boxes and only a few object boxes. And after quantization,

this problem is no longer intractable, since one typically ends up with $\sim 100k$ classification decisions per image, obviously better than infinite, but still extremely imbalanced. This is not only inefficient, since a system that expends a lot of computation classifying background boxes might be needlessly inefficient, but most importantly it hinders the learning process, since learning from imbalanced data is challenging.

This is usually handled with modified loss functions (e.g., focal loss [7]), cascade approaches (e.g., R-CNN family of detectors) or extreme quantization as the one seen in YOLO [2] where only 98 output boxes are possible, although this latter approach is less common.

In summary, the fact that the object detection problem is not usually treated as the prediction of a finite un-ordered set of boxes, introduces these surrogates (classification, regression) tasks which lead to several technical challenges. And the solutions to these issues lead to complex pipelines, typically not fully differentiable. Furthermore, the performance of said pipelines, relies heavily on domain specific priors introduced to guide the object detection process (e.g., anchors, labeling heuristics, NMS).

All the aforementioned issues are what our approach based on Detection Transformers (DETR) [11] will be trying to solve: by changing how the object detection problem is implemented, we will take advantage of a simpler, fully differentiable end-to-end method, which requires no priors and no post-processing. Since the basis of our approach is based on DETR, we will describe next its main properties when applied to object detection.

B. OBJECT DETECTION TRANSFORMERS

The work presented here is based on Detection Transformers which takes a different approach than those based on R-CNN: the object detection problem is dealt as a set prediction problem directly, thus bypassing all the surrogate tasks mentioned above. DETR adopts an encoder-decoder architecture based on transformers [12], an architecture that has become very popular for sequence prediction. Also, DETR incorporates the transformer's self-attention mechanism to include global context from the image to its predictions, instead of predicting each object individually as most methods do. Furthermore, it helps to remove duplicate detections, since DETR explicitly models all pairwise interactions between elements in a sequence (of detections). More details can be found in the next section on related work.

Unfortunately, although the performance of DETR on large objects is much better in comparison to Faster R-CNN, its main limitation is that it struggles to find small objects. This is due to the lack of low level features that can be fed into the transformer. Therefore, in this paper we propose a procedure to extend DETR to improve its performance when localizing objects in quite small regions. In essence we introduce a Feature Pyramid (FP) Network which is able to capture the content of low level feature maps and feeds it into the model in a higher resolution. Since feeding lower level features to DETR will increase the complexity quadratically,

we also propose a strategy to learn to use higher resolution feature maps for smaller objects. We next describe the resulting architecture, called DETR-FP, which we evaluate on the MS-COCO [13] and QMUL-OpenLogo [14] datasets and compared to a competitive Faster R-CNN baseline. To summarize: In this work we expand DETR to incorporate multi-level information improving its performance on the detection of small objects and qualitative and quantitative results in the MS-COCO and the QMUL-OpenLogo benchmarks.

II. RELATED WORK

The work presented in this paper proposes a pure end-to-end solution to object detection using transformers [11] expanding on previous work by incorporating a Feature Pyramid (FP) network to the DETR architecture and benefiting from bipartite matching losses for set prediction, encoder-decoder architectures based on the transformer, parallel decoding, and other contributions from relevant object detection methods as described next.

A. OBJECT DETECTION WITH PRIORS

Standard object detection methods use deep learning to generate regions proposals and subsequently classify them, although unifying these two tasks would be more efficient.

The first deep learning method approaching region proposal and classification was R-CNN [1], which uses a selective search to generate the object proposals and a CNN on top to extract relevant features to be later classified by an SVM. Improvements of this architecture lead to the appearance of Fast-RCNN [15], where region proposals were extracted from features map of a CNN by applying the ROI Pooling operation and then classify each region with a fully connected network. Later, Faster-RCNN [16] solved the main draw-back of previous architectures by substituting the selective search by a dedicated CNN called Region Proposal Network, which learned, given a predefined set of anchor boxes, where objects are located in an image and their shape. Finally, Mask-RCNN [17], replaced the ROI Pooling with the ROI Align operation which removes the quantization present in the former, improving both segmentation and detection results. This family of RCNN detectors has been incrementally improved with recent works such as [18], [19] which guide the region anchors by learning where objects are likely to exist in an image. Reference [20] improves the Faster-RCNN detector using multi-scale convolution feature fusion to make the feature map contain more information, improving the detection of small objects.

All of the aforementioned methods are considered two-stage detectors, the first stage being the region proposal stage and the second one the classification and box refinement stage. One-stage detectors merge these two stages into one by directly predicting class probabilities and box position on image grid cells, given predefined anchors.

Notice that whether the method is one or two stage, it requires prior geometrical knowledge to be fed into the

model. Whether it be anchor boxes, their ratio and size, or a grid of possible object centers [5], [21]. This hand-crafted, prior information has a severe impact on model performance as shown in [22].

The closest works to the DETR approach but using priors are end-to-end set predictions for object detections [23] and instance segmentation [24]–[27]. They also use bipartite-matching losses with decode-encoder architectures, however they are also based on autoregressive models (RNNs).

Previous work using bipartite matching loss [2], [6], [28] modeled the relation between different predictions using convolutional or fully connected layers with a hand-designed NMS as a post processing step to improve their performance. Some other, more recent work [5], [7], [16] use non-unique assignment rules between ground truth and predictions together with an NMS.

Learnable NMS methods [29], [30] and relation networks [31] use attention to model the relationship between predictions, by using direct set losses they do not require any post-processing steps. They do, however, require for hand-crafted prior information to be fed into the model (e.g., box proposals, anchors). DETR, in contrast, removes the need for these geometric priors or post processing steps by treating object detection as a set prediction problem.

B. SET PREDICTION WITH PRIORS

There is no canonical deep learning model to directly predict sets (of detected objects). The basic set prediction task is multi-label classification some examples of this for computer vision can be found in [32], [33]. However the one-vs-rest approach proposed in the aforementioned papers is not suitable for object detection due to the existing underlying structure between elements (i.e near identical boxes), which gives rise to near-duplicate detections.

Most current detectors use geometric prior knowledge to apply some sort of post-processing step, usually non maximum suppression to remove these near-duplicates, however set prediction is post-processing free. For constant size set prediction one could use dense fully connected networks [28], but given that the number of objects varies across images the problem cannot be approached in such a way. A general approach, suitable for variable length sets, is to use auto-regressive sequence models such as recurrent neural networks [34]. Whenever making set predictions the loss should be invariant to permutation in the predictions. A good solution is to design a loss based on the Hungarian algorithm [35] to match predictions and ground truth, in a one-to-one fashion, ensuring permutation-invariance.

C. DETECTION TRANSFORMERS

Transformers were introduced by Vaswani *et al.* [12] as a new attention-based building block for machine translation. Attention mechanisms [36] are neural network layers that aggregate information from the entire input sequence. Transformers implemented layers of self-attention which search

and update every item of a sequence by aggregating information from the entire sequence. One of the major advantages of attention-based models is their global computations and perfect memory, making them more suitable on long sequences than RNNs. In many problems transformers now replace RNNs in natural language processing, speech processing and computer vision [37]–[41].

Transformers were first used in auto-regressive models, following early sequence-to-sequence models [42], generating output tokens one by one. Due to the prohibitive inference cost (proportional to output length, and hard to batch) development of parallel sequence generation have been proposed in the domains of audio [43], machine translation [44], [45], word representation learning, and more recently speech recognition [46]. This work also combine transformers and parallel decoding for their suitable trade-off between computational cost and the ability to perform global computations.

Based on the encoder-decoder architecture of the transformers, Detection Transformers [11] were recently proposed as a simpler, fully differentiable end-to-end method, which requires no priors and no post-processing. In essence, for DETRs the object detection problem is dealt as a set prediction problem directly. Since the basis of our approach is based on DETR, we will describe next its main properties when applied to object detection.

DETR predicts all objects at once, and is trained in an end-to-end fashion, with a set loss function which performs bipartite matching between predicted and ground-truth objects. DETR does not require any hand-designed components that encode geometrical priors, such as non maximum suppression or spatial anchors. In addition, it does not require any custom layers (ROIAlign, ROIpool), thus making it reproducible in any deep learning framework, without the need to write custom GPU/CPU kernels for these custom layers to maximize performance. Finally, previous work on set-prediction was focused on autoregressive decoding with RNNs [23]–[27]. In contrast DETR matching function is permutation invariant, so is the transformer architecture in itself, thus allowing (non-autoregressive) parallel decoding [37], [44], [45], [47] of the predictions.

D. TOWARDS SMALL OBJECTS: LOGO DETECTION

The problem of logo recognition itself has a rich history of research. It is a challenging problem since logos are usually small and tiny differences in text or shape can represent widely different logos. In the 1990s, the question was explored primarily in the field of information retrieval use-cases. An image descriptor was developed with affine Transformations and stored in the Retrieval database [48]. There were also several methods focused on the neural networks [49], [50] but neither the networks were as deep nor the results were as impressive as in recent work.

In the 2000's, improved image descriptors became feasible with the introduction of SIFT and similar methods [51]–[53]. These methodologies were used to better represent images for recognizing logos [54]–[58]. Apart from SIFT, Other

approaches have also been explored by the community, metric learning [59], [60], using min-hashing [61] and bundling features for improved search [62]. Most of those approaches required complex pipelines for preprocessing images.

Recent work in logo recognition utilizes deep neural networks that offer superior performance with end-to-end automation of the pipeline. Broadly speaking, the following approach is prevalent: an image is fed into a convolutional neural network and a classifier predicts [14], [63]–[68]. All the aforementioned works require the use of prior information.

To the best of our knowledge, Detection Transformers had not been applied yet to logo detection due to their problems in detecting small objects. In order to solve this, we next describe the proposed method in order to extend the DETR architecture for logo detection without the use of geometrical priors, such as non maximum suppression or spatial anchors.

III. METHODOLOGY

In this work we improve the DETR architecture for small objects with the use of a Feature Pyramid Network, thus addressing one of the main drawbacks of the DETR architecture. In this section we will describe the basic DETR architecture, the criterion used for training, and the improvements we made in order to improve the performance for small objects.

A. PREDICTION OF SETS

One of the main challenges addressed in DETR is to treat object detection as a set-prediction problem in a non-autoregressive manner. In order to do this DETR predicts a pre-defined fixed-size of N predictions per image and pads the ground truth labels with as many \emptyset (no object) as necessary to reach N labels.

Let us denote \hat{y} as the set of prediction and y as the set of padded ground truth. Then to find a bipartite matching between these two sets we need to search for a permutation of N elements with the lowest cost:

$$\hat{\sigma} = \arg \min_{\sigma \in N} \sum_i^N \mathcal{L}_{match}(y_i, \hat{y}_{\sigma(i)}) \quad (1)$$

where $\mathcal{L}_{match}(y_i, \hat{y}_{\sigma(i)})$ is a pair-wise matching cost between ground truth y_i and a prediction with index $\sigma(i)$. This function minima is computed efficiently finding an optimal matching using the Hungarian algorithm [35]. The matching also takes into account the similarity of the predicted and ground truth boxes. We can view \hat{y} as a vector that contains a class label c and a bounding box b . For an index prediction $\sigma(i)$ the probability class is defined as $\hat{p}_{\sigma(i)}(c_i)$ and the predicted box as $\hat{b}_{\sigma(i)}$. Thus $\forall c_i \notin \emptyset \mathcal{L}_{match}(y_i, \hat{y}_{\sigma(i)})$ can be defined as:

$$\mathcal{L}_{match}(y_i, \hat{y}_{\sigma(i)}) = \hat{p}_{\sigma(i)}(c_i) + \mathcal{L}_{box}(b_i, \hat{b}_{\sigma(i)}) \quad (2)$$

After the optimal matching is computed, the next step is to compute the actual loss for all pairs matched in Eq. (1). The loss is defined as a linear combination of negative

log-likelihood for class prediction and a box loss:

$$\mathcal{L}_{\text{Hungarian}}(y, \hat{y}) = \sum_{i=1}^N -\log \hat{p}_{\hat{\sigma}(i)}(c_i) + \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)}) \quad (3)$$

where $\hat{\sigma}$ is the optimal assignment computed in Eq. (1). When $c_i \in \emptyset$ The log probability term is down weighted by a factor of 10 to alleviate class imbalance.

The second part of the matching cost and the Hungarian loss is the box loss. These boxes are predicted directly in contrast with many other detectors that make the predictions with respect to some initial guesses. This, however, gives rise to the problem of the relative scaling of the loss, since the ℓ_1 loss will have different scales for small and large boxes, even if their relative error is similar. In DETR this problem is mitigated by linearly combining the aforementioned ℓ_1 loss and the generalized IoU loss [32] that is scale invariant. Thus the box loss $\mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)})$ is defined as:

$$\mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)}) = \lambda_{\text{iou}} \mathcal{L}_{\text{iou}}(b_i, \hat{b}_{\sigma(i)}) + \lambda_{L_1} \left\| b_i - \hat{b}_{\sigma(i)} \right\|_1 \quad (4)$$

where λ_{iou} and λ_{L_1} are hyperparameters that control the weight of each term in the loss, and $\mathcal{L}_{\text{iou}}(b_i, \hat{b}_{\sigma(i)})$ is the generalized IoU loss defined as:

$$\mathcal{L}_{\text{iou}}(b_i, \hat{b}_{\sigma(i)}) = 1 - \left(\frac{|b_{\sigma(i)} \cap \hat{b}_i|}{|b_{\sigma(i)} \cup \hat{b}_i|} - \frac{|B(b_{\sigma(i)}, b_i) \setminus b_{\sigma(i)} \cup \hat{b}_i|}{|B(b_{\sigma(i)}, \hat{b}_i)|} \right) \quad (5)$$

where $|\cdot|$ means ‘‘area’’, $B(b_{\sigma(i)}, b_i)$ means the largest box containing $b_{\sigma(i)}, b_i$. The areas are computer based on min / max of linear functions of the box coordinates and the union and intersection of box coordinates are used as shorthands for the boxes themselves.

B. THE CORE DETR ARCHITECTURE

The DETR architecture is rather simple, therefore, unlike many modern detectors, it can be implemented in any deep learning framework in just a few hundred lines, without the need for huge configuration files such as those seen in Detectron2 [69] and without the need to implement custom layers.

The DETR architecture is mainly composed of 3 components. A conventional CNN backbone that given the initial image and yields an activation map $f \in \mathbb{R}^{C \times H \times W}$. This feature map is then fed to a 1×1 convolution in order to reduce the channel dimension from C to d creating a new feature map $z_0 \in \mathbb{R}^{d \times H \times W}$. Since the encoder expects a sequence the z_0 activation map spatial dimensions are collapsed into one, thus resulting in a $d \times HW$ feature map. The transformer architecture, however, is permutation invariant, so this feature map input must be added to positional embeddings [40], [70], which are basically sine waves at different frequencies. This is the same trick used in the original transformer paper [12] but adapted to images. Each encoder layer has a standard architecture and consists of a multi-head attention module and a feed forward network (FFN).

The decoder follows the standard architecture of the transformer. It uses self-attention and encoder-decoder attention mechanisms to transform N embeddings of size d . These embeddings are called object queries, which are learned positional embeddings that are tasked with *looking* at different regions in the feature map to find objects. These embeddings are transformed by the decoder and then decoded independently into a set of box coordinates and class labels by small feed forward network, resulting in N final predictions. Notice how, the DETR transformer is extremely similar to the standard transformer architecture.

This unique ability that the transformer has, making use of these self- and encoder-decoder attention mechanisms, allows the model to reason globally about all the objects, while being able to use the whole image as context.

C. FEATURE PYRAMID NETWORKS + DETR

As we briefly mentioned in the introduction one of the flaws of DETR is that it while exceeding on localizing large objects, it struggles to find small ones. This is due to the lack of low level features that can be fed into the transformer. This problem is addressed by adding a feature pyramid (FP) network [71] that allows us to capture the content of low level feature maps and feeds it into the model in a higher resolution. However, using low level features in this architecture is very computationally expensive, so a design decision must be made, as described next.

Let us denote the feature map of the last level of an FPN as $f \in \mathbb{R}^{C \times H \times W}$. This feature map is then projected into d dimensions using a 1×1 convolution and flattened into $d \times HW$ in order to be fed into the transformer. Notice that the complexity of the self-attention of the encoder is $\mathcal{O}(d^2HW + d(HW)^2)$, while $\mathcal{O}(d(HW)^2)$ is the complexity of computing attention weights for only one head. Feeding lower level features to the model will increase this complexity quadratically, since a feature map from the second to last level of the FPN has a size of $C \times 2H \times 2W$.

In order to alleviate this cost, we take the approach shown in Figure 2. Where we crop the second to last level of the FPN into four equally sized $C \times H \times W$ patches and feed them all into the transformer. The object queries learn to *query* higher resolution feature maps for smaller objects. It is worth mentioning that this approach can be repeated for every level in the pyramid at the cost of a considerable memory increase due to the increase in size in the computational graph needed for backward propagation. In this work we only use the last two levels of the pyramid.

IV. EXPERIMENTS

We compare the performance of DETR-FP against a strong Faster-RCNN baseline in the task of Logo detection, concretely in the QMUL-Openlogo benchmark [14]. The dataset is comprised of 27,083 images from 352 logo classes, built by aggregating and refining seven other logo datasets [55], [63], [64], [68], [72]–[74]. We use weights from ImageNet and MS-COCO [13] in different experiments for

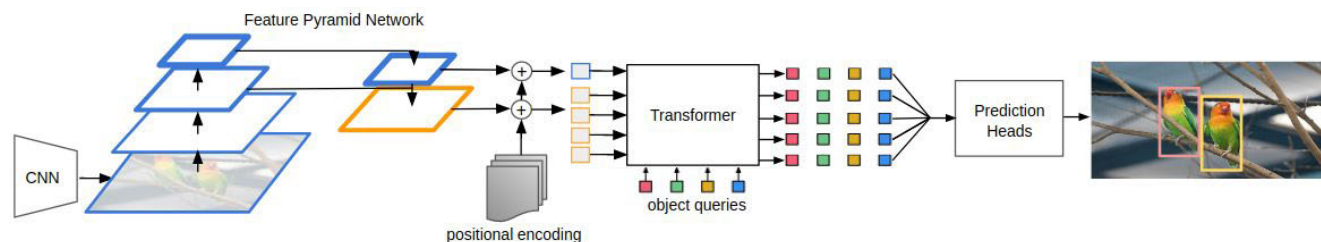


FIGURE 2. The DETR-FP approach to improve the localization of small objects. We introduce a feature pyramid and split the second to last level of the pyramid into four patches of the same size as the smallest feature map in the pyramid. This is all fed into the DETR pipeline and the predictions of each query for each feature map are concatenated.

both architectures. The QMUL-OpenLogo dataset has three different settings in order to reproduce the aforementioned real world scenario situation, where new classes constantly have to be learned. There are three settings, but only the first one is fully supervised, where every logo classes contains 70% of training split and 30% of evaluation split. Note that this data split is enforced by the challenge organizers in order to compare each method in the benchmark fairly.

All the experiments are run on eight GeForce GTX 1080 TI GPUs, using Distributed Data Parallel from PyTorch [75].

The DETR-FP pipeline is extended from the original DETR work. All of the DETR-FP models are trained using AdamW [76] with improved weight decay handling, set to 10^{-4} , gradient clipping is applied with a maximal gradient norm of 0.1 in order to stabilize training. All models were trained with $N = 100$ object queries.

1) BACKBONE AND TRANSFORMER

We use the same ResNet-50 backbone from Torchvision for all experiments, with weights from either ImageNet or MS-COCO, depending on the setting and a batch size of one image per GPU. Backbone batch normalization weights and statistics are frozen during training as it has been widely adopted in object detection. Two separate learning rates are used for the backbone and the transformer, 5^{-06} and 5^{-05} respectively. The transformer weights are initialized with Xavier initialization [77].

2) FASTER-RCNN BASELINE

We take a Faster-RCNN model, provided by the Detectron2 framework [69] with weights from either ImageNet or Ms-COCO depending on the setting, and fine tune it on the OpenLogo dataset. For this Faster-RCNN baseline we use the same data augmentation techniques as in the DETR and use a training schedule of with $3x$ iterations (around 40 epochs). We use a batch size of two images per GPU and a learning rate of 0.02 with cosine annealing [78]. The rest of the settings are those set by default in the Detectron2 model zoo [69].

3) POSITIONAL ENCODING

The positional encoding is used to represent the association between encoder activations and their corresponding image features. The one used in DETR-FP adopts a generalization

of the encoding in the original Transformer [12], but adapted to the 2D case [40]. Specifically, for a feature map $f \in \mathbb{R}^{d \times H \times W}$, $\frac{d}{2}$ sine and cosine functions with different frequencies are used for both spatial coordinates of each embedding independently. These embeddings are then concatenated to get the final $d \times H \times W$ channel positional encoding which is added to the input features.

4) OBJECT QUERIES

The object queries are learned embeddings tasked with *looking* at different regions of the input feature map and *querying* it for boxes of different sizes and shapes. One can think of them as learned anchors, that can generalize to objects of different shapes and sizes. As a result, each query specializes on certain areas and box sizes.

V. RESULTS

We show qualitative and quantitative results of DETR in both MS-COCO [13] and OpenLogo [14] along with a comparison against a Faster-RCNN baseline. We first provide a deeper analysis into the results obtained for small objects using the original DETR architecture, and subsequently by decomposing the performance of each model into different types of errors using TIDE [79]. This will allows us to get a deeper insight into DETR performance based on the self-attention feature map for the encoder and the decoder around points of interest. In addition, these results will justify the use of Feature Pyramids, as a proper, coherent extension of DETR.

A. DETR VS. FASTER-RCNN

There are several differences between the training setting of DETR and the one used in Faster-RCNN. Transformers are usually trained with very long training schedules and with Adam or Adagrad optimizers. In the case of DETR the models are trained for 500 epochs with AdamW optimizer [76]. Faster-RCNN, however has a much shorter training schedule and is trained with SGD.

Despite these differences we use a Faster-RCNN baseline for comparison. Localizing small objects is a challenge, but it can be alleviated by replacing the 2×2 stride in the last group of the ResNet backbone with dilated convolution, resulting in a larger feature map. This is a standard practice in object detection. Table 1 shows how DETR matches the

TABLE 1. Comparison with Faster-RCNN using a ResNet-50 on MS-COCO validation set. The + sign indicates an extended training schedule (~ 108 epochs) and DC5 indicate a dilated backbone. The dilated DETR model outperforms the rest, specially on large objects, however its performance on small objects is not as good as a Faster-RCNN model. (numerical results taken from [11]).

Model	GFLOPS	#params	AP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L
Faster RCNN-DC5	320	166M	39.0	60.5	42.3	21.4	43.5	52.5
Faster RCNN-FPN	180	42M	40.2	61.0	43.8	24.2	43.5	52.0
Faster RCNN-DC5+	320	166M	41.1	61.4	44.3	22.9	45.9	55.0
Faster RCNN-FPN+	180	42M	42.0	62.1	45.5	26.6	45.4	53.4
DETR	86	41M	42.0	62.4	44.2	20.5	45.8	61.1
DETR-DC5	187	41M	43.3	63.1	45.9	22.5	47.3	61.1



(a) DETR encoder self-attention feature map, visualized around the center of four random ground truth annotations.

(b) DETR decoder attention map, for the four detections with highest score.

FIGURE 3. Visualization of DETR encoder and decoder attention in a random instance of the MS-COCO validation set. The decoder looks at corners in the objects in order to localize them properly while the encoder isolates each instance of an object individually, even when the bounding boxes of these objects are overlapped. When an object is very small in an image the DETR encoder fails to properly isolate it. This is the case for the car in the top left corner of the image.

performance of the baseline in MS-COCO, with less operations (GFLOPS) and less parameters. Furthermore, using a transformer architecture allows us to visualize the attention maps and gain an insight into what each part of the model is trying to do. In the case of DETR the encoder seems to encode each instance of every object in an image in order to separate them, while the decoder focuses on localizing each of these

instances in order to identify them. This behaviour is clearly illustrated in Figure 3.

Logo detection is a rather difficult problem for DETR. Logos are usually small, and this is precisely the main weakness of this approach. Dilation can help alleviate this problem as evidenced above, however it is very expensive to do in DETR, computationally and memory-wise, due to

TABLE 2. The Faster-RCNN model without dilation or FPN falls clearly behind of DETR with the same conditions. The DC5 model has a comparable performance with DETR, while the FPN model outperforms the rest, except for large objects, where DETR excels.

Model	Weights	Schedule	AP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L
Faster RCNN	ImageNet	40	31.8	53.1	33.2	15.9	32.27	44.9
Faster RCNN	MS-COCO	40	32	49.7	36.3	14.9	32.2	45.6
Faster RCNN-DC5	ImageNet	40	38.5	57.1	43.9	22.4	38.9	51.6
Faster RCNN-DC5	MS-COCO	40	40.0	62.0	45.2	24.4	41.8	51.6
Faster RCNN-FPN	ImageNet	40	39.9	58.7	45.5	24.1	41.0	51.0
Faster RCNN-FPN	MS-COCO	40	40.5	62.0	46.0	25.9	42.0	51.1
DETR	ImageNet	300	13.5	24.2	13.5	6.0	13.3	21.87
DETR	MS-COCO	300	38.6	57.6	43.3	24.7	39.2	52.0



(a) DETR encoder self-attention feature map, visualized around the center of four random ground truth annotations.



(b) DETR decoder attention map, for the four detections with highest score.

FIGURE 4. Visualization of DETR encoder and decoder attention in a random instance of the OpenLogo validation set. The decoder looks at corners in the objects in order to localize them properly while the encoder isolates each instance of an object individually. When the objects are small, the encoder fails to isolate them and the self-attention is blurred and unfocused. Even when this happens, most of the time, the decoder manages to localize the object and the model detects it correctly.

the increase in complexity (see section III-C). This is not possible to do with our hardware, so we do not dilate the backbone in our experiments. Table 2 shows the results of both Faster-RCNN in DETR in the OpenLogo benchmark.

It is clear that without a dilated backbone Faster-RCNN struggles considerably in comparison with DETR. However, with the addition of an FPN or by dilating the backbone, the performance is very similar. Also notice that DETR needs MS-COCO weights in order to perform well due to the absence of priors and the complex relations that the transformer has to learn, it becomes impossible to train the model

in a reasonable amount of time with a small dataset such as OpenLogo. We visualize the attention maps for the encoder and decoder for some instances in the OpenLogo dataset (see Figure 4). Notice that in spite of the encoder struggling to separate instances of small objects, the decoder localizes them correctly.

B. HYPERPARAMETER SEARCH

There are a fair number of parameters to set in DETR, the weight for each different criteria, the background coefficient, learning rates, to name a few. In an attempt to find

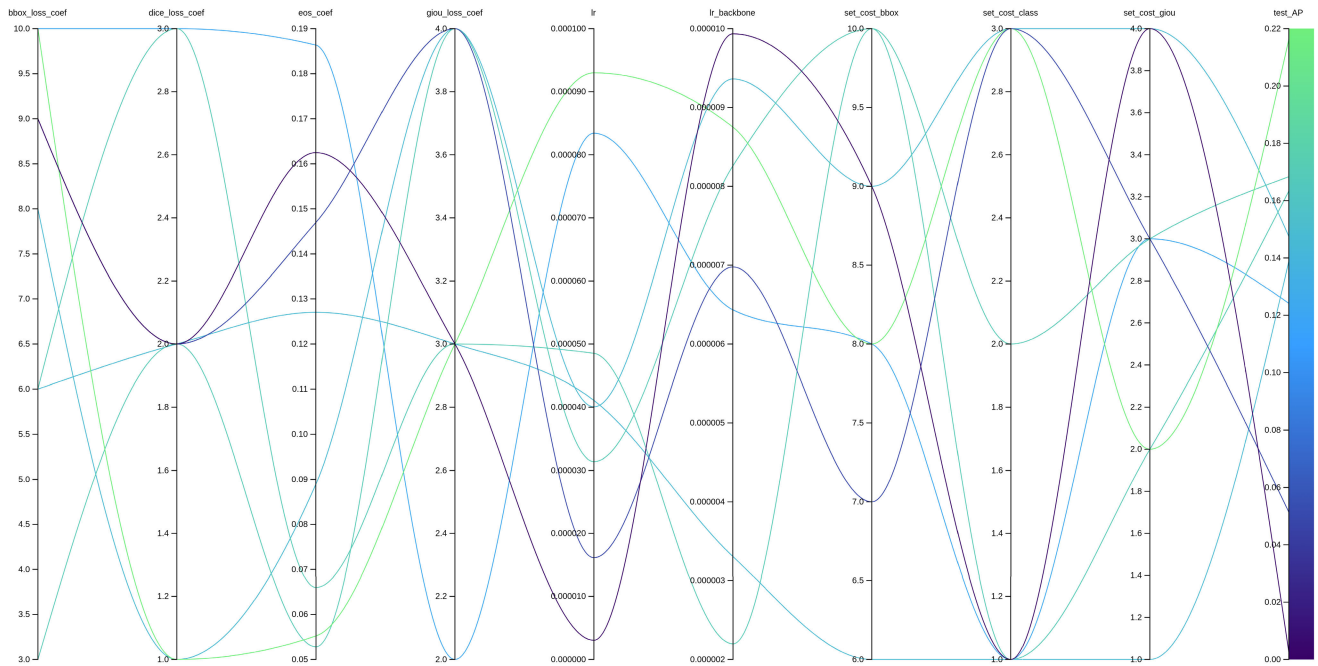


FIGURE 5. Parallel plot indicating the values selected by random search for each experiment and the resulting mAP. Each line represents one run. The greener the line, the higher the mAP, the opposite is true for blue lines. It seems that a high learning rate and a low weight for the no object class (*eof_loss_coef*) influence the mAP in a positive way. Keep in mind that in order to find optimal hyperparameters, more experiments over a greater range of values need to be run.

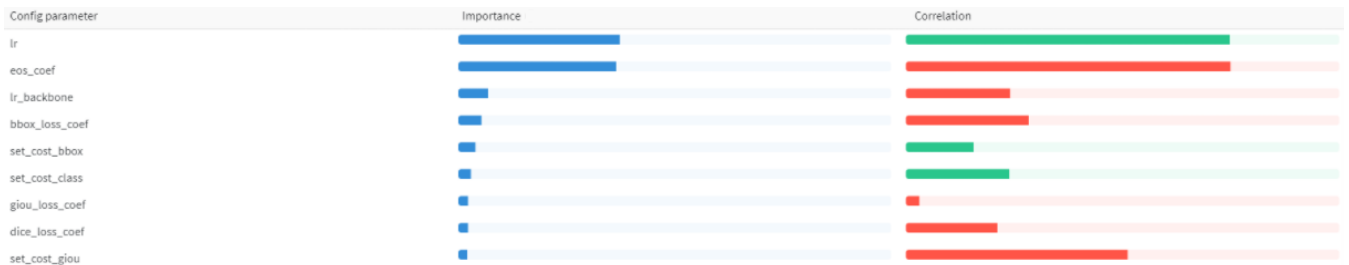


FIGURE 6. Importance of each parameter and its correlation with the mAP. Red and green indicate negative and positive correlation respectively. The importance and correlations values confirm that the learning rate and the relative classification weight of the no-object class (*eof_loss_coef*) are the hyperparameters that influence the performance of the model the most.

the optimal values for these parameters for the OpenLogo benchmark. We conducted random search over all of them, during 50 epochs (for time purposes). The results can be best visualized using a parallel plot (see Figure 5). Clearly the values of the parameters vary the results significantly, but it is hard to tell which one is more important.

In order to establish the importance of each parameter one can use correlation between each hyperparameter and the metric we are trying to maximize. However correlation cannot capture second order interactions between inputs and it can behave poorly when comparing inputs with wildly different ranges. This importance can be obtained by training a random forest classifier with the hyperparameters as inputs and the metric one is trying to maximize as target and report the feature importance values for

the random forest classifier [80]. The importance of each parameter and their correlation with the mAP, is shown in Figure 6.

C. DECOMPOSING THE PERFORMANCE

The standard metric used today for object detection is mean average precision (mAP). A complicated term that involves integrating over precision-recall curve and averaging over several criteria. There are many sources of errors that affect mAP, and yet, all we have to analyze the performance of our model is this number. Therefore it is hard to analyze which sources of error (e.g, classification, duplicate detections, localization, misses) is contributing the most to the errors our model is making. TIDE [79] breaks down the missing mAP into six types of errors, that fully explain

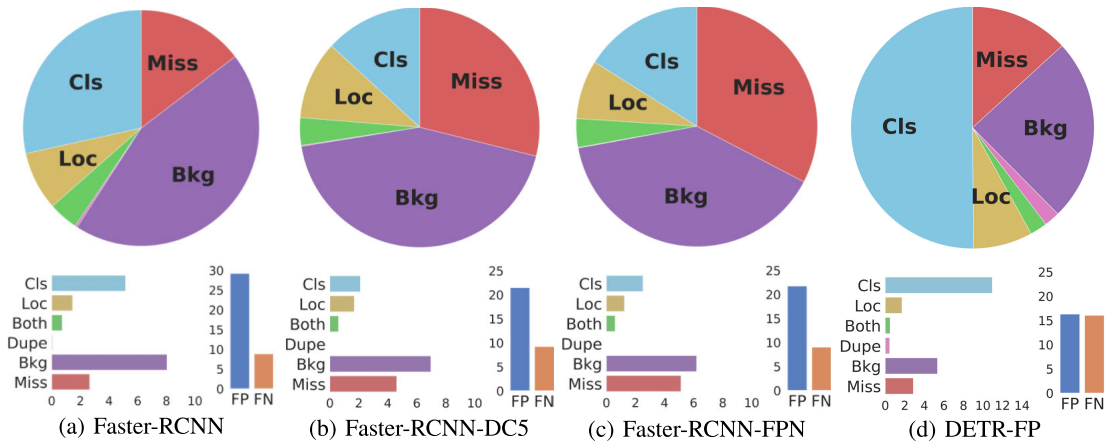
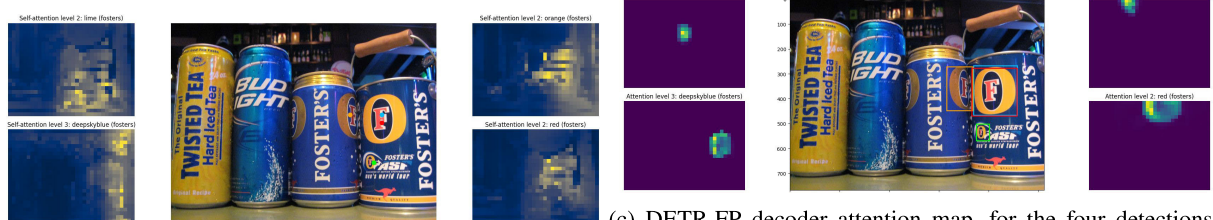


FIGURE 7. Performance of different models with MS-COCO weights, fine tuned on the OpenLogo dataset. As one can see the Faster-RCNN model makes far more classification mistakes than the rest, due to the lack of dilation or an FPN. Furthermore all of the Faster-RCNN models suffer from a high background and miss detection errors, while DETR-FP suffer mostly from classification error.



(a) DETR-FP predictions and confidence (dashed lines) and GT (solid lines).



(b) DETR-FP encoder self-attention feature map, visualized around the center of four random ground truth annotations. (c) DETR-FP decoder attention map, for the four detections with highest score. Keep in mind that the level 2 predictions are done on the 4 crops (top-left, top-right, bottom-left, bottom-right) of the second level of the pyramid.

FIGURE 8. DETR-FP Results for one image of the OpenLogo validation set. Notice how the queries have learned to look for small objects on the different crops of the second level of the pyramid and how despite the localization in the encoder does not seem useful or accurate, the decoder is capable of finding the object. However, we can observe some duplicate predictions due to the fact that the same object can be found in different levels of the feature pyramid.

where the model is losing performance. Let us denote IoU_{max} to denote the maximum IoU overlap of a false positive with a ground truth of the given category, t_f as the foreground IoU threshold and t_b as the background IoU threshold. Then, the six error types are defined as follows:

- 1) **Classification error:** $IoU_{max} \geq t_f$ for GT of the *incorrect* class (i.e., localized correctly, but classified incorrectly)
- 2) **Localization error:** $t_b \leq IoU_{max} \leq t_f$ for GT of the *correct* class (i.e., classified correctly, but localized incorrectly).

TABLE 3. Quantitative results of each model in the OpenLogo validation set. It is clear that without dilation or low level feature information models make a lot of classification mistakes. *Cls, Loc, Dupe Bkg, Miss* stand for the six types of errors described previously, while *FP, FN* stand for false positive and false negative.

Model	Cls	Loc	Both	Dupe	Bkg	Miss	FP	FN
Faster-RCNN	5.15	1.45	0.74	0.06	8.05	2.65	29.17	8.83
Faster-RCNN-DC5	2.11	1.69	0.59	0.02	6.97	4.62	21.46	9.18
Faster-RCNN-FP	2.53	1.24	0.61	0.01	6.21	5.14	21.71	9.00
DETR-FP (Our approach)	10.96	1.72	0.50	0.45	5.34	2.88	16.29	16.03



FIGURE 9. Qualitative results of DETR-FP applied to logo detection. Note that logos are correctly detected on quite small regions.

- 3) **Classification and Localization error:** $t_b \leq IoU_{max} \leq t_f$ for GT of the *incorrect* class. (i.e., classified and localized incorrectly).
- 4) **Duplicate detection error:** $IoU_{max} \geq t_f$ for GT of the *correct* class, after a higher-scoring detection already matched that GT.
- 5) **Background error:** $IoU_{max} \leq t_b$ for all GT (i.e., detected background as foreground)

- 6) **Missed GT error:** All undetected GT (false negatives) not already covered by classification and localization error.

Using these metrics we can decompose the error of our models, and get an insight into what can be improved in order to boost their performance. As seen in Figure 7, the Faster-RCNN model struggles with the classification of objects and detects a lot of background as foreground. The

former is alleviated by the addition of dilation in the backbone or an FP, but the latter remains present across all of the Faster-RCNN models, along with a high number of miss detections. In contrast the DETR-FP improves over DETR with the addition of the feature pyramid (FP).

We also provide the exact numbers for the missing mAP decomposition, shown in Figure 7, these can be found in Table 3. Looking at the table it becomes clear that the performance of the model with dilation and FP are very similar, but the Faster-RCNN model falls clearly behind.

D. DETR-FP

In this section we provide some extra qualitative results on the OpenLogo validation set, by visualizing the detections for DETR-FP along with the attention of the transformer. Figure 8 shows DETR-FP predictions for one image of the OpenLogo validation set. The figure shows how the queries in DETR-FP learn to *query* different crops of the lower levels of the feature pyramid for objects that are small or in the background. It is worth noticing that the model learns in which level to *look* for small, medium or large objects without any extra supervision.

Finally, we show in Figure 9 examples of small logo detections of DETR-FP, thanks to the Feature Pyramid.

VI. CONCLUSION

In this paper we extend DETR, a new fully differentiable end-to-end solution for object detection that requires no geometric priors and no post processing, for small object detection. We improved its performance by feeding multi level information using a Feature Pyramid (FP) and compared its results with a strong Faster-RCNN baseline in the MS-COCO and OpenLogo benchmarks where we obtain up to a 30% relative improvement. There is, however much room for improvement in the DETR-FP approach. It's considerable computational cost prevents us from using the lower levels of the feature pyramid. Furthermore the introduction of different feature maps for the object queries to analyse, increases the amount of duplicate detections, since the same object can be detected in different levels of the pyramid. The direction of our future work moves towards addressing these issues in order to make DETR-FP performance a total improvement over the strong baselines that make use of several geometric priors.

REFERENCES

- [1] 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.
- [2] 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.*, Jun. 2016, pp. 779–788.
- [3] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 7263–7271.
- [4] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: <http://arxiv.org/abs/1804.02767>
- [5] X. Zhou, D. Wang, and P. Kráhenbühl, "Objects as points," 2019, *arXiv:1904.07850*. [Online]. Available: <http://arxiv.org/abs/1904.07850>
- [6] 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.* Berlin, Germany: Springer, 2016, pp. 21–37.
- [7] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.
- [8] T. Kong, F. Sun, H. Liu, Y. Jiang, L. Li, and J. Shi, "FoveaBox: Beyond anchor-based object detector," 2019, *arXiv:1904.03797*. [Online]. Available: <http://arxiv.org/abs/1904.03797>
- [9] M. Bonnaerens, M. Freiberger, and J. Dambre, "Anchor pruning for object detection," 2021, *arXiv:2104.00432*. [Online]. Available: <http://arxiv.org/abs/2104.00432>
- [10] 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.
- [11] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," 2020, *arXiv:2005.12872*. [Online]. Available: <http://arxiv.org/abs/2005.12872>
- [12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
- [13] 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.* Berlin, Germany: Springer, 2014, pp. 740–755.
- [14] H. Su, X. Zhu, and S. Gong, "Open logo detection challenge," in *Proc. Brit. Mach. Vis. Conf.*, 2018, pp. 1–14.
- [15] R. Girshick, "Fast R-CNN," 2015, *arXiv:1504.08083*. [Online]. Available: <http://arxiv.org/abs/1504.08083>
- [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] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," 2017, *arXiv:1703.06870*. [Online]. Available: <http://arxiv.org/abs/1703.06870>
- [18] J. Wang, K. Chen, S. Yang, C. C. Loy, and D. Lin, "Region proposal by guided anchoring," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 2965–2974.
- [19] T. Yang, X. Zhang, Z. Li, W. Zhang, and J. Sun, "MetaAnchor: Learning to detect objects with customized anchors," 2018, *arXiv:1807.00980*. [Online]. Available: <http://arxiv.org/abs/1807.00980>
- [20] C. Cao, B. Wang, W. Zhang, X. Zeng, X. Yan, Z. Feng, Y. Liu, and Z. Wu, "An improved faster R-CNN for small object detection," *IEEE Access*, vol. 7, pp. 106838–106846, 2019.
- [21] 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.
- [22] S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Z. Li, "Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 9759–9768.
- [23] R. Stewart, M. Andriluka, and A. Y. Ng, "End-to-end people detection in crowded scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2325–2333.
- [24] B. Romera-Paredes and P. H. S. Torr, "Recurrent instance segmentation," in *Proc. Eur. Conf. Comput. Vis.* Berlin, Germany: Springer, 2016, pp. 312–329.
- [25] A. Salvador, M. Bellver, V. Campos, M. Baradad, F. Marques, J. Torres, and X. Giro-i-Nieto, "Recurrent neural networks for semantic instance segmentation," 2017, *arXiv:1712.00617*. [Online]. Available: <http://arxiv.org/abs/1712.00617>
- [26] M. Ren and R. S. Zemel, "End-to-end instance segmentation with recurrent attention," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6656–6664.
- [27] E. Park and A. C. Berg, "Learning to decompose for object detection and instance segmentation," 2015, *arXiv:1511.06449*. [Online]. Available: <http://arxiv.org/abs/1511.06449>
- [28] 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.
- [29] J. Hosang, R. Benenson, and B. Schiele, "Learning non-maximum suppression," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 4507–4515.

- [30] 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.
- [31] H. Hu, J. Gu, Z. Zhang, J. Dai, and Y. Wei, "Relation networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3588–3597.
- [32] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 658–666.
- [33] S. H. Rezatofighi, A. Milan, E. Abbasnejad, A. Dick, and I. Reid, "DeepSetNet: Predicting sets with deep neural networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5257–5266.
- [34] O. Vinyals, S. Bengio, and M. Kudlur, "Order matters: Sequence to sequence for sets," 2015, *arXiv:1511.06391*. [Online]. Available: <http://arxiv.org/abs/1511.06391>
- [35] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Res. Logistics Quart.*, vol. 2, nos. 1–2, pp. 83–97, Mar. 1955.
- [36] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*. [Online]. Available: <http://arxiv.org/abs/1409.0473>
- [37] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*. [Online]. Available: <http://arxiv.org/abs/1810.04805>
- [38] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.
- [39] C. Lüscher, E. Beck, K. Irie, M. Kitzka, W. Michel, A. Zeyer, R. Schlüter, and H. Ney, "RWTH ASR systems for LibriSpeech: Hybrid vs attention—W/O data augmentation," 2019, *arXiv:1905.03072*. [Online]. Available: <http://arxiv.org/abs/1905.03072>
- [40] N. Parmar, A. Vaswani, J. Uszkoreit, A. Kaiser, N. Shazeer, A. Ku, and D. Tran, "Image transformer," 2018, *arXiv:1802.05751*. [Online]. Available: <http://arxiv.org/abs/1802.05751>
- [41] G. Synnaeve, Q. Xu, J. Kahn, T. Likhomanenko, E. Grave, V. Pratap, A. Sriram, V. Liptchinsky, and R. Collobert, "End-to-end ASR: From supervised to semi-supervised learning with modern architectures," 2019, *arXiv:1911.08460*. [Online]. Available: <http://arxiv.org/abs/1911.08460>
- [42] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.
- [43] A. Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. Driessche, E. Lockhart, and L. Cobo, "Parallel WaveNet: Fast high-fidelity speech synthesis," 2017, *arXiv:1711.10433*. [Online]. Available: <http://arxiv.org/abs/1711.10433>
- [44] J. Gu, J. Bradbury, C. Xiong, V. O. K. Li, and R. Socher, "Non-autoregressive neural machine translation," 2017, *arXiv:1711.02281*. [Online]. Available: <http://arxiv.org/abs/1711.02281>
- [45] M. Ghazvininejad, O. Levy, Y. Liu, and L. Zettlemoyer, "Mask-predict: Parallel decoding of conditional masked language models," 2019, *arXiv:1904.09324*. [Online]. Available: <http://arxiv.org/abs/1904.09324>
- [46] W. Chan, C. Saharia, G. Hinton, M. Norouzi, and N. Jaitly, "Imputer: Sequence modelling via imputation and dynamic programming," 2020, *arXiv:2002.08926*. [Online]. Available: <http://arxiv.org/abs/2002.08926>
- [47] A. Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, and G. Driessche, "Parallel wavenet: Fast high-fidelity speech synthesis," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 3918–3926.
- [48] D. S. Doermann, E. Rivlin, and I. Weiss, "Logo recognition using geometric invariants," in *Proc. 2nd Int. Conf. Document Anal. Recognit.*, 1993, pp. 894–897.
- [49] F. Cesarini, E. Francesconi, M. Gori, S. Marinai, J. Q. Sheng, and G. Soda, "A neural-based architecture for spot-noisy logo recognition," in *Proc. 4th Int. Conf. Document Anal. Recognit.*, vol. 1, Aug. 1997, pp. 175–179.
- [50] E. Francesconi, P. Frasconi, M. Gori, S. Marinai, J. Sheng, G. Soda, and A. Sperduti, "Logo recognition by recursive neural networks," in *Proc. Int. Workshop Graph. Recognit.* Berlin, Germany: Springer, 1997, pp. 104–117.
- [51] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," in *Eur. Conf. Comput. Vis.* Berlin, Germany: Springer, 2006, pp. 404–417.
- [52] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.
- [53] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2564–2571.
- [54] R. Boia, A. Bandrabur, and C. Florea, "Local description using multi-scale complete rank transform for improved logo recognition," in *Proc. 10th Int. Conf. Commun. (COMM)*, May 2014, pp. 1–4.
- [55] A. Joly and O. Buisson, "Logo retrieval with a contrario visual query expansion," in *Proc. 17th ACM Int. Conf. Multimedia*, 2009, pp. 581–584.
- [56] A. P. Psyllos, C.-N. E. Anagnostopoulos, and E. Kayafas, "Vehicle logo recognition using a SIFT-based enhanced matching scheme," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 2, pp. 322–328, Jun. 2010.
- [57] S. Romberg, L. G. Pueyo, R. Lienhart, and R. van Zwol, "Scalable logo recognition in real-world images," in *Proc. 1st ACM Int. Conf. Multimedia Retr. (ICMR)*, 2011, pp. 1–8.
- [58] G. Zhu and D. Doermann, "Automatic document logo detection," in *Proc. 9th Int. Conf. Document Anal. Recognit. (ICDAR)*, vol. 2, Sep. 2007, pp. 864–868.
- [59] J. Chen, M. K. Leung, and Y. Gao, "Noisy logo recognition using line segment Hausdorff distance," *Pattern Recognit.*, vol. 36, no. 4, pp. 943–955, Apr. 2003.
- [60] D. Nistér and H. Stewénius, "Scalable recognition with a vocabulary tree," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, vol. 2, Jun. 2006, pp. 2161–2168.
- [61] S. Romberg and R. Lienhart, "Bundle min-hashing for logo recognition," in *Proc. 3rd ACM Conf. Int. Conf. multimedia Retr. (ICMR)*, 2013, pp. 113–120.
- [62] Z. Wu, Q. Ke, M. Isard, and J. Sun, "Bundling features for large scale partial-duplicate web image search," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 25–32.
- [63] S. Bianco, M. Buzzelli, D. Mazzini, and R. Schettini, "Logo recognition using cnn features," in *Proc. Int. Conf. Image Anal. Process.* Berlin, Germany: Springer, 2015, pp. 438–448.
- [64] S. Bianco, M. Buzzelli, D. Mazzini, and R. Schettini, "Deep learning for logo recognition," *Neurocomputing*, vol. 245, pp. 23–30, Jul. 2017.
- [65] S. C. H. Hoi, X. Wu, H. Liu, Y. Wu, H. Wang, H. Xue, and Q. Wu, "LOGO-net: Large-scale deep logo detection and brand recognition with deep region-based convolutional networks," 2015, *arXiv:1511.02462*. [Online]. Available: <http://arxiv.org/abs/1511.02462>
- [66] F. N. Iandola, A. Shen, P. Gao, and K. Keutzer, "DeepLogo: Hitting logo recognition with the deep neural network hammer," 2015, *arXiv:1510.02131*. [Online]. Available: <http://arxiv.org/abs/1510.02131>
- [67] G. Oliveira, X. Frazao, A. Pimentel, and B. Ribeiro, "Automatic graphic logo detection via fast region-based convolutional networks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 985–991.
- [68] A. Tüzkö, C. Herrmann, D. Manger, and J. Beyerer, "Open set logo detection and retrieval," 2017, *arXiv:1710.10891*. [Online]. Available: <http://arxiv.org/abs/1710.10891>
- [69] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick. (2019). *Detectron2*. [Online]. Available: <https://github.com/facebookresearch/detectron2>
- [70] I. Bello, B. Zoph, A. Vaswani, J. Shlens, and Q. V. Le, "Attention augmented convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2019, pp. 3286–3295.
- [71] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2117–2125.
- [72] H. Su, X. Zhu, and S. Gong, "Deep learning logo detection with data expansion by synthesising context," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2017, pp. 530–539.
- [73] Y. Kalantidis, L. G. Pueyo, M. Trevisiol, R. van Zwol, and Y. Avrithis, "Scalable triangulation-based logo recognition," in *Proc. 1st ACM Int. Conf. Multimedia Retr. (ICMR)*, Trento, Italy, Apr. 2011, pp. 1–7.
- [74] Y. Liao, X. Lu, C. Zhang, Y. Wang, and Z. Tang, "Mutual enhancement for detection of multiple logos in sports videos," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 4856–4865.
- [75] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, and A. Kopf, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019, pp. 8024–8035. [Online]. Available: <http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf>

- [76] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” 2017, *arXiv:1711.05101*. [Online]. Available: <http://arxiv.org/abs/1711.05101>
- [77] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.
- [78] I. Loshchilov and F. Hutter, “SGDR: Stochastic gradient descent with warm restarts,” 2016, *arXiv:1608.03983*. [Online]. Available: <http://arxiv.org/abs/1608.03983>
- [79] D. Bolya, S. Foley, J. Hays, and J. Hoffman, “Tide: A general toolbox for identifying object detection errors,” in *Proc. ECCV*, 2020, pp. 558–573.
- [80] *Feature Importance, Tree Interpreter*. Accessed: Sep. 13, 2020. [Online]. Available: <https://course18.fast.ai/lessonsml1/lesson4.html>



F. XAVIER ROCA received the Ph.D. degree in computer science from the Universitat Autònoma de Barcelona (UAB), Cerdanyola del Vallès, Spain, in 1990. He is currently an Associate Professor and the Director of the Department of Computer Science, UAB. He is also a Research Fellow with the Computer Vision Center. He has been the Principal Researcher at several projects (public and private funds). He is working in technological transfer computer vision. The topics of his research are active vision, biometrics, and tracking.



DIEGO A. VELAZQUEZ is currently pursuing the Ph.D. degree in computer science with the Universitat Autònoma de Barcelona (UAB). He is also working at Visual Tagging Services. His research interests include deep learning, artificial intelligence, and computer vision.



SEIICHI OZAWA (Senior Member, IEEE) received the Dr.Eng. degree in computer science from Kobe University. He is currently the Deputy Director of the Center for Mathematical and Data Sciences and a Full Professor with the Department of Electrical and Electronic Engineering, Graduate School of Engineering, Kobe University, Japan. He has published more than 160 journal articles and conference papers, and book chapters/monographs. His current research interests include deep learning, machine learning, pattern recognition, incremental learning, big data analytics, cybersecurity, text mining, computer vision, and privacy preserving data mining. He is a member of the Neural Networks TC and the Smart World TC of the IEEE CI Society. He is the Vice President for Membership of the International Neural Network Society, Finance of the Asia Pacific Neural Network Society, and the Board of Governor of Japan Neural Network Society. He is an Associate Editor of IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON CYBERNETICS, and two international journals.



JOSEP M. GONFAUS received the Ph.D. degree in computer engineering from the Universitat Autònoma de Barcelona (UAB), in 2012. He participated in various Pascal challenges. He cofounded a spin-off based on their research by applying computer vision and deep learning techniques for analyzing social media data (Visual Tagging). His main research interest includes deep learning techniques to mimic human cognitive capabilities.



PAU RODRÍGUEZ received the Ph.D. degree in computer science from the Universitat Autònoma de Barcelona, in 2019. He is currently a Research Intern at Element AI, Montreal. His research interests include machine learning, pattern recognition, and computer vision.



JORDI GONZÁLEZ received the Ph.D. degree in computer engineering from the Universitat Autònoma de Barcelona (UAB), in 2004. He is currently an Associate Professor in computer science with the Department of Computer Science, UAB. He is also a Research Fellow with Computer Vision Center, where he has cofounded three spin-offs, namely Cloud Size Services, Visual Tagging, and Care Respite, and the Image Sequence Evaluation (ISE Lab) Research Group. His research interest includes machine learning techniques for the computational interpretation of social images, or visual hermeneutics.

• • •