



Transformer-Based Context Condensation for Boosting Feature Pyramids in Object Detection

Zhe Chen¹ · Jing Zhang¹ · Yufei Xu¹ · Dacheng Tao¹

Received: 14 July 2022 / Accepted: 22 May 2023 / Published online: 24 June 2023
© Crown 2023

Abstract

Current object detectors typically have a feature pyramid (FP) module for multi-level feature fusion (MFF) which aims to mitigate the gap between features from different levels and form a comprehensive object representation to achieve better detection performance. However, they usually require heavy cross-level connections or iterative refinement to obtain better MFF result, making them complicated in structure and inefficient in computation. To address these issues, we propose a novel and efficient context modeling mechanism that can help existing FPs deliver better MFF results while reducing the computational costs effectively. In particular, we introduce a novel insight that comprehensive contexts can be decomposed and condensed into two types of representations for higher efficiency. The two representations include a locally concentrated representation and a globally summarized representation, where the former focuses on extracting context cues from nearby areas while the latter extracts general contextual representations of the whole image scene as global context cues. By collecting the condensed contexts, we employ a Transformer decoder to investigate the relations between them and each local feature from the FP and then refine the MFF results accordingly. As a result, we obtain a simple and light-weight Transformer-based Context Condensation (TCC) module, which can boost various FPs and lower their computational costs simultaneously. Extensive experimental results on the challenging MS COCO dataset show that TCC is compatible to four representative FPs and consistently improves their detection accuracy by up to 7.8% in terms of average precision and reduce their complexities by up to around 20% in terms of GFLOPs, helping them achieve state-of-the-art performance more efficiently. Code will be released at <https://github.com/zhechen/TCC>.

Keywords Object detection · Feature pyramid · Context modeling

Mathematics Subject Classification 35A01 · 65L10 · 65L12 · 65L20 · 65L70

1 Introduction

Object detection aims to detect and localize objects with various sizes on images. To detect objects of diversified sizes, researchers have found that a feature pyramid (FP), which collects features from different levels of the backbone network and performs detection in a pyramidal hierarchy, are important for high-quality detection. Using a deep convolutional neural network (DCNN) like ResNet (He et al., 2016) which provides multiple levels of DCNN features as the backbone network, the feature pyramid, such as the FPN (Lin et al., 2017a), detects smaller objects at shallower levels and larger objects at deeper levels. The shallower-level features have higher resolutions and thus are better at describing objects with smaller sizes, but these features are worse at extracting abstract and robust representation. On the other

This work was supported in part by Australian Research Council Projects IH-180100002 and FL-170100117.

✉ Dacheng Tao
dacheng.tao@gmail.com
Zhe Chen
zhe.chen1@sydney.edu.au
Jing Zhang
jing.zhang1@sydney.edu.au
Yufei Xu
yuxu7116@uni.sydney.edu.au

¹ Faculty of Engineering, School of Computer Science, The University of Sydney, Darlingtown, NSW 2008, Australia

hand, the deeper-level features have much smaller scales and are better for objects with larger sizes, while they lose too many local details for detection. Although feature pyramids can achieve more accurate detection by enabling the detection of smaller objects on shallower-level features and larger objects on deeper-level features, using either lower-level or higher-level features alone still cannot satisfy high-quality detection demand. In the related research, it shows that cooperating the advantages of features from different levels by performing multi-level feature fusion (MFF) is very helpful to obtain more descent representation of objects and achieve better detection performance.

To achieve effective MFF, gaps between features of different levels are supposed to be mitigated for better fusion. However, we found that current popular mechanisms usually require excessive computational complexity to ensure more promising MFF results. For example, in the typical feature pyramid as introduced in FPN (Lin et al., 2017a), the MFF is fulfilled by a normal 3 by 3 kernel based convolution operation after adding each shallower-level feature with the upsampled adjacent deeper-level feature. This design is simple but very primitive. It can be quite difficult for only using a 3 by 3 convolution to effectively bridge the gap between different feature representations to refine the final detection results promisingly. Nevertheless, the 3 by 3 convolution itself requires plenty of extra complexity, especially for shallower levels with high resolutions. Derived from FPN, researchers further developed more complicated and also more costly architectures to help achieve better MFF. For example, the authors of NasFPN (Ghiasi et al., 2019) applied an auto-network searching technology (Zoph & Le, 2017; Zoph et al., 2018; Baker et al., 2016) to make the machine automatically search for a better feature pyramid structure and better MFF strategy. The searched feature pyramid is then iterated for multiple times in a successive order to further refine the MFF results, which still brings considerable computational complexity. Later, the FPG (Chen et al., 2020) and RFP (Qiao et al., 2021) explored even more comprehensive and complicated MFF strategies in pyramid architectures. Although these methods achieved state-of-the-art performance, they continues to introduce multiple iterations of the same processing structure to ensure more effective MFF in feature pyramids, making them still very computational costly. Instead of repeated and exhaustive feature modeling, we find that contexts could be of great benefit for refining the fused representation after MFF. More specifically, by considering extra visual cues from a larger surrounding area as contexts, it could become easier for shallower-level features to resist visual noises and for upsampled deeper-level features to improve local detailed descriptions. As a result, the gap between features from different levels could be alleviated effectively with contexts. Some studies (Zhao et al., 2017; Chen et al., 2021) have demonstrated the effective-

ness of including contexts for improving detection and other visual understanding tasks. However, a more comprehensive context modeling is usually still accompanied with a higher computational complexity, which would limit its advantages for boosting feature pyramids efficiently.

Although comprehensive context modeling may still be costly, we further find that the rich contexts can be decomposed and simplified to avoid the need for heavy computation. We are motivated by the studies (Uz Kent & Ermon, 2020; Zhang et al., 2014; Chen et al., 2019b) that successfully summarized complicated visual appearances using a specific local representation and some sparse general contextual representations. In those studies, the specific local representation is used to represent local visual patterns in images, and the sparse general contextual representations summarize overall visual patterns into a few contextual features. The specific local representation encodes more detailed visual information that is closely related to specific objects, describing visual patterns from surrounding areas of each object. For example, the information from a round face, two arms, and two legs can form a specific representation of a human body. Meanwhile, the sparse general contextual representation describes the representative contextual feature of a large image area. It can provide general information and strong hints about objects, but it may not directly relate to a specific object. For example, only a few crosswalk lines can provide strong hints about the presence of pedestrians or vehicles, thus these crosswalk lines can provide representative contextual cues for the entire image scene. Then, by making sparse general contextual representations tend to complement specific local representation based on the relations between both types of representations, jointly using both types of representations can avoid exhaustive feature processing without sacrificing much visual understanding performance. Accordingly, we propose that the rich contexts can also be decomposed into a specific local representation and some general contextual representations to achieve efficient modeling. We tend to term such decomposition of contexts as context condensation.

To condense and utilize contexts for boosting MFF and feature pyramids efficiently, in this study, we propose a novel Transformer-based context condensation (TCC) module for object detection. After adding features from different levels as fusion results, the TCC first condenses rich contexts on the fused representations and then use the condensed contexts to refine the feature of each entry on the fused feature map with the help of a Transformer decoder. More specifically, in a TCC module, we attempt to first condense contexts by decomposing comprehensive context information into a locally concentrated representation and a globally summarized representation. We make the locally concentrated representation provide specific local representation of local contexts, and we attempt to summarize global contexts into a

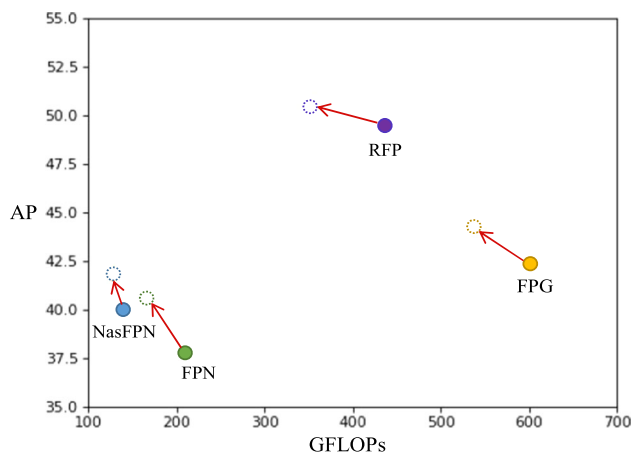


Fig. 1 The proposed Transformer-based Context Condenser (TCC) (dotted circles) can effectively improve the multi-level feature fusion in different baseline feature pyramids (solid circles) and the final detection performance by utilising rich context information. In the meantime, the TCC also reduces the required computational complexity promisingly by condensing the contexts into two simplified representations, including a locally concentrated representation and a globally summarized representation

few general contextual features whose locations are predicted by an extra small neural network. Then, we collect the feature of locally concentrated representation and the features of globally summarized contexts as the condensed context representation. With the condensed contexts, we apply a Transformer decoder to estimate the relations between the condensed contexts and each feature to be refined, so that more related and useful context information can be identified and highlighted from condensed contexts. Based on the estimated relations, the Transformer decoder translates the obtained condensed context features into a more appropriate contextual representation to achieve better MFF refinement. Therefore, using the condensed contexts, the exhaustive modeling of comprehensive contexts on the feature map can be reduced to the modeling of only a few context features. The TCC can thus achieve promising MFF refinement performance without introducing too large complexities. Figure 1 shows the performance improvement and complexity reduction of applying our proposed TCC on MFF of different typical feature pyramids. Despite being motivated by Uzcent and Ermon (2020); Zhang et al. (2014); Chen et al. (2019b), we would like to clarify that our TCC is very different from these studies. These studies mainly aim to augment DCNN features for image recognition tasks by extracting complementary information from the decomposed representations of images. In these studies, the complementary information is usually rough and primitive, thus requiring carefully designed mechanisms to cooperate with the normal DCNN features. Besides, they lack explicit context modeling mechanisms for reasoning high-level relations between each local feature and its surrounding visual cues. This makes

them quite inappropriate for object detection which requires descent local representations for dense predictions. Different from these studies, our method devises a novel simple context collection mechanism that can learn to find information that is beneficial for improving each local feature accurately and efficiently. Using a high-level attention-based context fusion strategy, our method delivers effective and explicit context modeling for object detection.

To sum up, the major contributions of this work can be summarized as follows:

- A novel Transformer-based Context Condensation (TCC) module is presented to boost MFF in feature pyramids for detecting object effectively and efficiently. It mainly helps MFF achieve better detection performance without requiring great computational complexities.
- The proposed TCC module is easy-to-inject and can boost MFF in many different popular feature pyramid methods developed for object detection. It is also compatible with different backbone networks and is not sensitive to specific DCNN architectures. It can be trained with different feature pyramids in an end-to-end manner.
- Comprehensive empirical studies on MS COCO detection dataset (Lin et al., 2014) show that our proposed TCC module can reduce the typical FPN method by more than 30 GFlops while achieving around 7.8% relative improvement. By applying the TCC module to state-of-the-art feature pyramids like NasFPN (Ghiasi et al., 2019) and RFP (Qiao et al., 2021), our method consistently reduces complexity and improves detection performance, accessing compelling performance on the COCO dataset. Codes shall be released upon acceptance of this paper.

2 Related Work

2.1 Feature Pyramids for Object Detection

In computer vision, building pyramidal representations which describe visual appearances at different levels or resolutions have shown to be effective for tackling scale variation problem (Chen et al., 2020). In modern deep learning-based object detectors, feature pyramids have become a core component. Early works like SSD (Liu et al., 2016) and MS-CNN (Cai et al., 2016) extract and utilize multi-level feature maps to perform detection, but they do not consider feature fusion or aggregation across different levels, which makes them sub-optimal for detection. Then, the most popular feature pyramid for detection is introduced in the FPN (Lin et al., 2017a). It presents a simple multi-level feature extraction and fusion architecture, which consists of several top-down pathways and lateral connections, to help distribute the detection of objects with different sizes on features from different

levels of the backbone DCNN. Similar to the design logic of FPN, many methods (Kong et al., 2017; Huang et al., 2017; Liu et al., 2018; Sun et al., 2019; Yu et al., 2018; Zhao et al., 2019) have been proposed to achieve more effective pyramidal feature extraction by developing more comprehensive and more complicated multi-level feature extraction and fusion structures. In addition to these hand-crafted feature pyramids, the work of NasFPN (Ghiasi et al., 2019) follows the NAS algorithms (Zoph & Le, 2017; Zoph et al., 2018; Baker et al., 2016) which enables automatic searching for more efficient and more effective network architectures to facilitate the automatic searching for a better feature pyramid for object detection. The NasFPN achieves promising improvements over the FPN, but it requires several stacks of the same feature pyramid structure to refine the multi-level features. Later, Auto-FPN (Xu et al., 2019) has introduced a more comprehensive searching policy to improve detection, but its searched architecture has different structures in different stacks, making it less scalable. More recently, researchers have developed methods like FPG (Chen et al., 2020) and RFP (Qiao et al., 2021) to achieve more comprehensive feature fusion in different levels of several stacks of feature pyramids. Although state-of-the-art performance can be accessed by these methods, they still introduce much extra computation complexity for object detection.

2.2 Context Modeling for Object Detection

Visual contexts have been demonstrated to be important for visual understanding in both computer vision and cognitive science (Torralba, 2003; Biederman, 2017; Rumelhart & McClelland, 1982). In particular, computer vision researchers have introduced additional visual cues as useful contextual information to facilitate the description of visual appearance at each particular location. In object detection, contexts have been used for improving the representation of convolutional features extracted at different levels of the backbone DCNN. For example, many methods (Gidaris & Komodakis, 2015; Komodakis & Gidaris, 2016) sampled extra windows with different scales around each region of interest (RoI) to extract surrounding visual features that can improve the later classification and regression operations in detectors. In addition to context windows, some other studies (Zeng et al., 2017; Chen & Gupta, 2017) successfully improved the features for describing objects of different sizes by introducing recurrent neural networks to encode contextual information in larger areas. Authors of Chen et al. (2021) further proposed a comprehensive and dynamic context modeling framework to improve multi-level feature and boost object detection performance effectively. Besides, the non-local operation (Wang et al., 2018) shows great benefits for improving detection by modeling all the sub-sampled features extracted from input images. However,

current cutting-edge context modeling methods would usually introduce excessive computational complexity to model contexts more comprehensively, making it difficult to apply current context modeling methods to augment multi-level feature fusion in feature pyramids for object detection.

2.3 Decomposition of Feature Representation

Researchers have found that complicated visual patterns can be decomposed into different types of simplified representations. For example, Zhang et al. (2014) introduced part-based representations to complement the holistic representation for fine-grained classification. It relied on randomly sampled object proposals and mainly extracts information from raw image patches without explicitly considering relationships between patches. However, the random proposals may not be quite accurate, and the information extraction mechanism in this study is complicated. Compared to this study, our method uses an attention-based structure to help the network learn by itself to find precise locations of summarized global contexts with a more sensible and efficient mechanism to explore the relationships between different types of context features. The authors of Uzket and Ermon (2020) used part-global representation to learn when and where to zoom images to facilitate recognition. It investigated how to complement low-resolution representation with high-resolution image content for better image recognition and remote sensing. To utilize high-resolution images with feasible complexities, this study attempted to search for better local areas to focus on. However, since their method is based on global information from multi-resolution which is difficult to guide the focus, they applied a reinforcement learning mechanism to fulfill the search of focus areas, which is complicated and costly for training. Alternative to this study, our method devises an effective method that can also learn to search for meaningful locations to summarize global contexts into a few features but with a novel, simple and efficient mechanism. Another related study (Chen et al., 2019b) attempted to improve the common convolution operations by using low-frequency information to complement the high-frequency features and thus avoid heavy computation costs. The authors factorized the mixed feature maps by their frequencies and represented image features with information from low and high spatial frequency domains. This method is effective for image and video recognition tasks, but it extracts and fuses low-frequency features in a primitive way, without modeling larger surrounding areas as contexts. This method also lacks the modeling of high-level relationships between features at different locations. Instead, our proposed method explicitly explores contexts from a large area and introduces an advanced attention modeling mechanism to achieve effective feature fusion. In general, we found in these studies that the correlation between the general and specific features lacks

sufficient study. We argue that the correlation between different types of simplified features is important to achieve promising performance. In particular, the level of correlation between different features can indicate their relevance and correlation reveal whether involved features share similar and complementary information that is of interest. Although researchers have developed cosine similarity-based (Wojke & Bewley, 2018), Euclidean distance-based (Qi et al., 2017), Intersect-over-Union (IoU) based (Chen et al., 2018), and also learning-based (Hu et al., 2018; Chen & Gupta, 2017) correlation estimation methods for various vision tasks, these methods either cannot be directly applied on the general and specific features or could be difficult to describe complicated relations. Instead, inspired by the Transformers for vision (Dosovitskiy et al., 2020; Liu et al., 2021; Xu et al., 2021) that show to be extremely powerful at modeling complicated relations between features, we propose a Transformer-based context encoder to model specific features and learn to better complement the general features.

2.4 Transformer for Object Detection

Recently, the Transformer has achieved great success in computer vision (Dosovitskiy et al., 2021; Touvron et al., 2021; Wang et al., 2021a). Researchers also developed a Transformer-based object detector, named DETR (Carion et al., 2020). It effectively takes advantage of the powerful relation modeling in Transformer for high-quality detection. However, the pure Transformer-based detector generally suffers from excessively long training periods. To address this problem, many researchers found that locality modeling is important for accelerating the training of DETR (Tay et al., 2020; Xu et al., 2021; Zhang et al., 2022). For example, the Deformable DETR (Zhu et al., 2020) applied deformable operations (Dai et al., 2017) to better focus on a few local areas in the Transformer. The method SMCA (Gao et al., 2021) introduced multi-scale co-attention to improve DETR with refined local representations. In addition, other studies like Conditional DETR (Meng et al., 2021) and Anchor DETR (Wang et al., 2021b) tend to improve the spatial embedding in Transformer to help accelerate training. These two methods enhance the locality modeling of Transformer by making attention focus on potentially valuable areas on the image learned with positional embeddings. The authors of Chen et al. (2022) introduces recurrent refinement to improve the locality modeling of DETR. Despite progress, the Transformer-based methods require a long training period to achieve promising performance, while our method aligns with normal DCNN-based object detectors and requires shorter training periods to converge.

Different from DETR methods that focus on the detection task, there is another group of Transformer methods that tend to extract tend to summarize complicated features into a uni-

fied latent space, including Perceiver (Jaegle et al., 2021b), Perceiver IO (Jaegle et al., 2021a), Flamingo (Alayrac et al., 2022), and so on. The Perceiver (Jaegle et al., 2021b) is a Transformer-based method that is designed for scaling to arbitrary inputs. Using an asymmetric attention mechanism, it can process inputs into a tight latent bottleneck representation. By extracting information from arbitrary inputs, it can achieve effective image classification. Rather than the Perceiver that only focuses on classification, Perceiver IO (Jaegle et al., 2021a) introduces a unified visual language model that can produce arbitrary outputs in addition to varied inputs. The Perceiver IO can produce language, optical flow, and multimodal videos with audio. Using the same structure as the Perceiver, the Perceiver IO is linear in the input and output size. Besides the Perceiver IO, the Flamingo (Alayrac et al., 2022) is another Perceiver-based method that focuses on visual language modeling for visual understanding tasks like visual question answering. It extracts information from arbitrarily interleaved text and image information and introduces a Perceiver resampler from varying-size large feature maps to a few visual tokens, so that complicated visual inputs can be encoded and decoded to a few desired outputs. It performs especially well on few-shot learning tasks.

These great works have introduced impressive performance in visual-language understanding. Their idea of extracting only a few important information from an arbitrarily shaped input is similar to our globally summarized contextual representations. However, there are a few major differences. For example, we have different information extraction methods - the Perceiver-based methods mainly use a few queries and cross-attention to scale to any input, while our global summarized representation collects general contextual features based on max pooling and importance scores. Our method can be more efficient at extracting informative global contexts by ignoring the massive areas with low importance scores. In addition, the Perceiver-based methods rely on multiple iterative attention layers to help encode complicated input information into a unified latent space, while our method only needs to extract general contextual features only once, making our method more light-weighted for processing massive input information.

3 Method

As mentioned previously, we introduce Transformer contexts condensation (TCC) modules to refine MFF and boost different feature pyramids for object detection without introducing too much computational overhead. In the TCC, condensed contexts are first collected and then a Transformer decoder is employed to translate the collected contexts into a better representation for refining features. Figure 2 shows an overview of the proposed method.

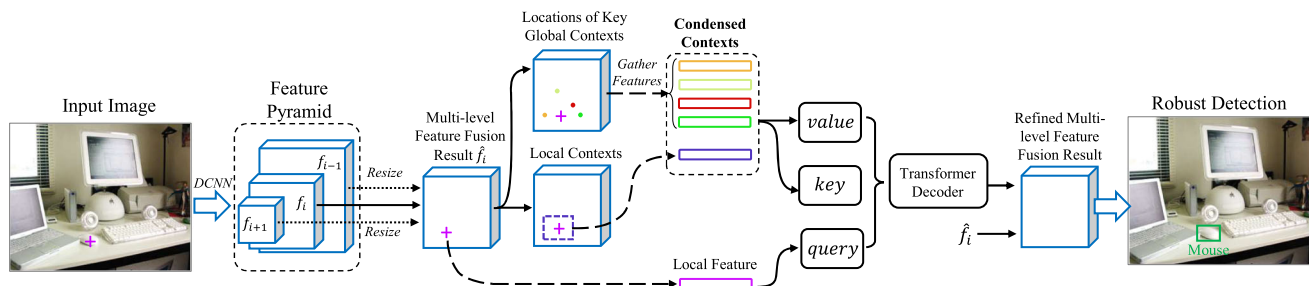


Fig. 2 Processing pipeline of the proposed Transformer-based Context Condensation (TCC) module for boosting multi-level feature fusion. In the figure, the purple cross indicates a local entry on the fusion results at level i , the dots with different colors indicate predicted locations of

globally summarized context features, and dashed deep purple rectangular represents locally concentrated context feature. Solid arrows, dotted arrows, dashed arrows, and light grey arrows represent convolutions, resizing operations, and feature gathering operations, respectively

In the following sections, we will subsequently describe in details about our formulation of MFF operation in a feature pyramid, how we collect condensed contexts to improve MFF in the feature pyramids, and how we use Transformer to decode the collected contexts.

3.1 MFF in a Feature Pyramid

In general, feature pyramids used in object detectors usually have the following processing steps, i.e. collecting multi-level features from the backbone DCNN, re-sampling features from another level to match the scale of features at the current level, and the MFF that fuses and refines the collected multi-level features to deliver descent feature representation for detection at the current level.

Mathematically, we use the symbol f_i to represent the i -th level feature extracted from the backbone DCNN and use the symbol $\mathcal{S}_{j \rightarrow i}[f_j]$ represents a re-sampling function that resizes the scale of the j -th level feature f_j into the scale of f_i . Then, we can summarize the MFF of the i -th level in the feature pyramids as the following operation:

$$\hat{f}_i = g_r \left(f_i + \sum_{j \in \mathcal{N}(i)} \mathcal{S}_{j \rightarrow i}[f_j] \right), \quad (1)$$

where $\mathcal{N}(i)$ refers to the other levels considered for fusion at level i , g_r refers to the feature refinement function used in different methods, and $\hat{f}_i$ is the obtained MFF result for level i in the feature pyramid. For example, in the typical FPN (Lin et al., 2017a), $\mathcal{N}(i)$ is defined as its adjacent deeper level, i.e. $\mathcal{N}(i) = i + 1$. Meanwhile, the FPN employs a simple 3 by 3 convolution to implement g_r for refining fused features. Some methods like FPG (Chen et al., 2020) and RFP (Qiao et al., 2021) have introduced more complicated mechanisms to improve g_r , while they usually requires several repeats of the same feature processing procedure. In practice, these methods could introduce considerably large extra computational complexity for object detectors.

3.2 TCC

Alternative to existing MFF mechanisms in feature pyramids, we introduce a Transformer-based context condensation (TCC) module to implement g_r in Eq. 1 for improving the feature representation obtained after fusion through addition. As mentioned previously, the TCC first collect condensed contexts and then decode them for better refinement. As a result, the general mathematical description about TCC re-writes the Eq. 1 as follows:

$$\begin{cases} \hat{f}_i = \text{Trans}(q = \tilde{f}_i; k/v = g_{cc}(\tilde{f}_i)), \\ \tilde{f}_i = f_i + \sum_{j \in \mathcal{N}(i)} \mathcal{S}_{j \rightarrow i}[f_j], \end{cases} \quad (2)$$

where Trans refers to a Transformer decoder module, the q, k, v represent the query, key, and value of the Transformer, and g_{cc} refers to the collection of condensed contexts.

3.2.1 Condensed Context Collection

To take advantage of rich contexts without introducing excessive computational complexity, we propose to condense contexts by decomposing them into a locally concentrated representation and a globally summarized representation that consists of a few general contextual features. As mentioned previously, we make the locally concentrated representation describe a local specific description about nearby contexts and make the globally summarized representation delivers simplified descriptions about informative cues in the whole scene. Accordingly, we can formulate the condensed context collection as:

$$g_{cc}(\tilde{f}_i) = [\tilde{f}_i^{lr}, \tilde{f}_i^{gr}], \quad (3)$$

where $[\cdot]$ represents a feature concatenation operation, $\tilde{f}_i^{lr}$ and $\tilde{f}_i^{gr}$ refer to the locally concentrated representation and the globally summarized representation, respectively.

To extract the locally concentrated representation as local contexts, we can simply use a dilated convolution operation. However, different from common dilated convolutions, here we only use a very small channel number and a small dilation rate to describe local appearances efficiently. Considering that the $\tilde{f}_i^{lr}$ is the specific local representation for i -th level, we have the following formulation:

$$\tilde{f}_i^{lr} = \text{DilatedConv}(\tilde{f}_i), \quad (4)$$

where *DilatedConv* means a dilated convolution operation. To avoid excessive computational costs, the used channel number is calculated by 8×2^i and the dilation rate is empirically set as 2.

To extract the globally summarized representation, we attempt to use a few general contextual features to summarize and represent useful information from the whole scene. We hypothesize that the objects in the same visual scene on an input image usually share information on similar global contextual information. For example, when detecting a person and sheep in the same image, the sky or the meadow can facilitate the recognition of both types of objects in this image. That is, there can be strong statistical dependence between objects and global scene environments, making $P(\text{person}|\text{sky}, \text{meadow})$ and $P(\text{sheep}|\text{sky}, \text{meadow})$ relatively high in a natural image. Our experiments can validate that these shared general contextual features can improve the overall detection performance effectively.

To identify the general contextual features from the scene, we tend to make networks automatically learn to discover which contextual features are more useful. In particular, we mainly perform a two-step extraction procedure: we first employ a small network to predict the locations of these general contextual features, and then we gather these features according to the predicted locations as the summarized global contexts. Recall that $\tilde{f}_i^{gr}$ represents the desired globally summarized representation for the i -th level. Given a location set $P_i = \{(x, y)_{i,1}, \dots, (x, y)_{i,n}\}$ describing the predicted locations of general contextual features, we then have the summarized global contexts according to:

$$\tilde{f}_i^{gr} = \Phi(\tilde{f}_i, P_i) \quad (5)$$

where $\Phi(\tilde{f}_i, P_i)$ is the function that gathers features from input feature map $\tilde{f}_i$ at the locations defined in the set P_i . The function Φ is further defined as:

$$\Phi(\tilde{f}_i, P_i) = [\phi(\tilde{f}_i, (x, y)_{i,1}), \dots, \phi(\tilde{f}_i, (x, y)_{i,k})], \quad (6)$$

where $\phi(\tilde{f}_i, (x, y)_{i,k})$ is to collect detailed visual feature at the location of $(x, y)_{i,k}$.

To obtain an appropriate location set P_i for collecting general contextual features, we make networks learn to predict

the location set automatically. To achieve this, we introduce importance scores for locations on the feature map and build the location set by collecting locations with the maximum importance scores. Suppose we collect n general contextual features, then we make the importance scores have n dimensions (or n elements for each location on the feature map). Further suppose importance scores are denoted by $C_i = \{c_{i,k} \mid k = 1, 2, \dots, n\}$ where $c_{i,k}$ represents the k -th importance score map. In a TCC module, we compute C_i using convolutions according to:

$$C_i = \text{Split}(\text{Conv}(\tilde{f}_i), n), \quad (7)$$

where *Conv* represents a 1x1 convolution operation with a channel number of n , and *Split*($\cdot$) means the operation that splits the obtained feature map into n different feature maps along channel dimension. Each split feature map $c_{i,k}$ has 1 feature channel and can be considered as the importance score map for indicating the location of the desired k -th general contextual feature.

With the calculated and split importance score maps, we collect P_i whose corresponding importance scores have the highest values in its related score maps. For example, when collecting the location of k -th general contextual feature, we perform global max-pooling operation on the feature map $c_{i,k}$ to help identify the location $(x, y)_{i,k}$ of interest and the corresponding maximum importance score $s_{i,k}$:

$$s_{i,k}, (x, y)_{i,k} = \text{MaxPool}(c_{i,k}), \quad (8)$$

where *MaxPool* refers to the global max pooling operation that returns the pooled maximum value $s_{i,k}$ and its location $(x, y)_{i,k}$.

To properly supervise the training of importance scores C_i , we multiply the sigmoid-squashed maximum scores with the corresponding features collected as general contextual features. That is, in addition to original feature, we further formulate the collection operation $\phi(\tilde{f}_i, (x, y)_{i,k})$ as:

$$\phi(\tilde{f}_i, (x, y)_{i,k}) = \tilde{f}_i|_{(x,y)_{i,k}} \cdot \sigma(s_{i,k}), \quad (9)$$

where $\tilde{f}_i|_{(x,y)_{i,k}}$ means the specific feature on the feature map of $\tilde{f}_i$ located at $(x, y)_{i,k}$, and σ represents the sigmoid function.

3.2.2 Transformer-Based Condensed Context Decoder

With the collected condensed contexts, it is then important to incorporate the contextual information into the MFF results for refinement. Since the Transformer is powerful at modeling complicated relationships between different features and fusing features according to the estimated relationships, we apply a Transformer decoder to correlate each local feature

to be refined and its corresponding condensed contexts and then translate the collected condensed contexts into a better representation for refining the fusion results of MFF. It is worth mentioning that we do not apply the encoder in a typical Transformer structure because we found in the experiments that this module delivers marginal improvement but extra costs in our proposed TCC module.

In the typical Transformer decoder architecture (Vaswani et al., 2017; Carion et al., 2020), a cross-attention structure is introduced for modeling complicated relationships between different visual features. The cross-attention estimate correlation between the input *query* and *key* and then aggregate *value* features according to the estimated correlations. In the proposed TCC modeling module, we formulate each local feature as *query* and then describe the collected specific representations as both *key* and *value*.

We follow the typical Transformer decoder but make some minor modifications to save computation costs. As described in Eq. 2, we suppose that each local entry of the MFF result $\tilde{f}_i$ is the query q with E as its feature dimension. We unify the key and value and consider condensed contexts $g_{cc}(\tilde{f}_i)$ as keys and values. Each feature of condensed contexts also has E feature dimensions. We stack all the condensed contexts, include locally concentrated representation $\tilde{f}_i^{lc}$ and the globally summarized representation $\tilde{f}_i^{gc}$, into a unified tensor F_i^{cc} : $F_i^{cc} = g_{cc}(\tilde{f}_i)$. With $\tilde{f}_i$ and F_i^{cc} , we implement the Transformer decoder, abbreviated as “ $\mathcal{D}$ ”, according to:

$$\mathcal{D}(\tilde{f}_i, F_i^{cc}) = W_{\mathcal{A}}(\tilde{f}_i + \mathcal{A}^i F_i^{cc}), \quad (10)$$

where $W_{\mathcal{A}}$ is a trainable linear projection matrix and $\mathcal{A}^i$ represents the attentional weight:

$$\mathcal{A}^i = \text{Softmax} \left(\frac{\tilde{f}_i (F_i^{cc})^T}{\sqrt{E}} \right). \quad (11)$$

3.2.3 Discussions

Although our introduction of the condensed contexts is motivated by the studies (Uzkent & Ermon, 2020; Chen et al., 2019b) which show that rich visual details can be significantly simplified with the help of a specific local representation and a few sparse general contextual representations, we would like to clarify that our approach is significantly different from these methods. For example, existing methods do not rely on importance score-based general contextual location prediction to summarize global contexts. Moreover, existing methods generally require additional supervision for training, while our method learns automatically and maintains the end-to-end processing structure with different feature pyramids and object detectors. Lastly, current methods do not investigate the relations between condensed

contexts and the local representation for identifying potentially useful information with the help of Transformer.

3.3 Other Implementation Details

The TCC is easy-to-plug-in for MFF in different feature pyramids. Besides, the TCC can also be added to the features both after and before fusion. Adding TCC before fusion can bring 0.4 increase in AP point for MS COCO *val* dataset comparing to only adding TCC after the fusion for MFF. In this paper, we mainly adopt the implementation of adding TCC for both before and after fusion in the experiments. In addition, we also apply the other implementation details as follows. Firstly, remind that we reduce the input channel dimension computed by 8×2^i , which also means that we use 64 times smaller channel numbers than the input features from backbone network when using the backbone like ResNet50 (He et al., 2016), and the level i considered here ranges from 0 to 3. Secondly, we apply a residual connection between the TCC result $\tilde{f}_i$ and the MFF result $\tilde{f}_i$ after Eq. 2 to perform the MFF refinement. It is also worth mentioning that we discard the multi-head structure for higher efficiency. We found that this does not sacrifice much performance. Lastly, when improving MFF before or after fusion, we stack 2 TCC modules together to achieve better refinement performance. Then, in each TCC module, we predict 4 general contextual locations to extract globally summarized context features. The ablation study of these hyper-parameters can be found at the experiment section.

Regarding the Transformer implementation of our method, we have the following implementation details. In general, we follow the typical Transformer architectures (Carion et al., 2020; Dosovitskiy et al., 2021) to implement our decoder as described in Eq. 11. However, there are a few major differences, including the following aspects: (1) In the typical Transformer, there are 6 encoding layers and 6 decoding layers. Instead, we only use 1 decoding layer. We adopt this implementation because we found that it is a waste to apply too many layers to process only a few context features as general contextual and value with only one local feature as query in the Transformer. Our ablation study validates that using more layers in our TCC will introduce excessive computation complexity and blow up GPU memory. (2) We discard the multi-layer perception in the Transformer decoder, so that we can avoid the introduction of massive parameters. We can show in our ablation study that removing the multi-layer perception does not compromise the performance too much. (3) We use only 1 head in the decoding layer rather than 8 heads in the original Transformer. This is because we only have very small channel numbers for the decoding layer, making it not very meaningful to apply many heads.

According to the implementation as described above, the computational complexity of our TCC module would not

be very large by performing feature dimension reduction. In general, the complexity of the TCC is mainly from the extraction of condensed contexts and the correlation of two different condensed contexts for refinement. For condensed context extraction, the locally concentrated representation is light-weighted because of the reduced feature dimensions, and the globally summarized representation would also not consume many computational resources because we only use a few (e.g. 4) general contextual points. We can show in the experiments that adding more general contextual points can quickly reach saturation in improvement. Regarding correlating both types of condensed context for refining MFF, the Transformer structure we used would still not bring great computational burdens due to the following designs. For example, we only employ decoders without involving encoders which we found in the experiments only have trivial impacts on the performance. Besides, since the considered feature dimension and the number of general contextual features are both small, it is quite efficient for obtaining the attentional weights with Transformer.

4 Experiments

4.1 Setup

For evaluation, we mainly use the popular MS COCO (Lin et al., 2014) dataset to help reveal the detection performance of different models. It has 118k training images, 5k validation images, and around 21k *test-dev* images. We follow the MS COCO protocol and report the performance using the evaluation metrics of average precision (AP), AP at 0.5, AP at 0.75, and AP for small, medium, and large objects. The validation set is mainly used for ablation study, and the *test-dev* set is used for main comparison. To present the analysis on computational complexities, we use the GFLOPs¹ as the main indicator.

To validate the effectiveness of the proposed TCC, we apply the TCC on MFF for different popular and representative feature pyramids, such as FPN (Lin et al., 2017a), NasFPN (Ghiasi et al., 2019), FPG (Chen et al., 2020), and RFP (Qiao et al., 2021). The details about these feature pyramids are listed as follows:

- *FPN* (Lin et al., 2017a) is the initial and the most typical feature pyramid used in object detection. Its MFF is implemented after the fusion of higher-level feature and lower-level feature followed by a 3 by 3 convolution as refinement. To apply TCC in FPN, we simply replace the 3 by 3 convolution with the proposed TCC module.

- *NasFPN* (Ghiasi et al., 2019) is derived from the FPN and introduces neural architecture search technology to obtain a novel feature pyramid structure. The MFF in the searched NasFPN structure is also based on operations like concatenation, sum, and convolution. The detailed implementation is discovered by the neural network by itself. The NasFPN is repeated several times to further refine the MFF results. In NasFPN, we apply the TCC after final MFF of each level and reduce one repetition of processing to save costs.
- *FPG* (Chen et al., 2020) refers to the feature pyramid grid which introduces a deep multi-pathway feature pyramid by formulating the features from different levels as a regular grid of parallel bottom-up pathways. The MFF is performed by applying multi-directional lateral connections. It is not based on searching, but its structure is also repeated several times for refinement. Similar with NasFPN, we apply the TCC after final MFF of each level and also reduce one repetition.
- *RFP* (Qiao et al., 2021) means recursive feature pyramid, which incorporates additional feedback connections from FPN into the bottom-up backbone layers. It cooperates with a switchable atrous convolution for MFF. It is also similar with NasFPN and FPG and applies multiple RFP modules to achieve advanced performance. In RFP, the original FPN structure is also applied. We thus can directly apply the TCC in FPN. In addition, we also apply the TCC after the switchable atrous convolution for MFF.

It is worth mentioning that these feature pyramids are originally implemented in different detection pipelines such as Faster RCNN (Ren et al., 2015), RetinaNet (Lin et al., 2017b), and HTC (Chen et al., 2019a). For example, for FPN and FPG, we use the Faster RCNN (Ren et al., 2015) as the detection pipeline for comparison. For NasFPN, we use the RetinaNet (Lin et al., 2017b) as the detection pipeline. For the RFP, we use the HTC (Chen et al., 2019a) as the detection pipeline. For a fair comparison, we do not alter the pipeline in the original implementation of each included FPN method and only modify the MFF part for experiments. For training different feature pyramids, we follow the protocols reported in the related papers and reproduce all the baseline results using the publicly released code.²

4.2 Detection Performance

4.2.1 Performance on Different Feature Pyramids

Firstly, we study the improvement brought by TCC on different feature pyramids regarding efficiency and accuracy. We also use different backbone networks to prove that our

¹ GFLOPs: Giga floating point operations.

² <https://github.com/open-mmlab/mmdetection>.

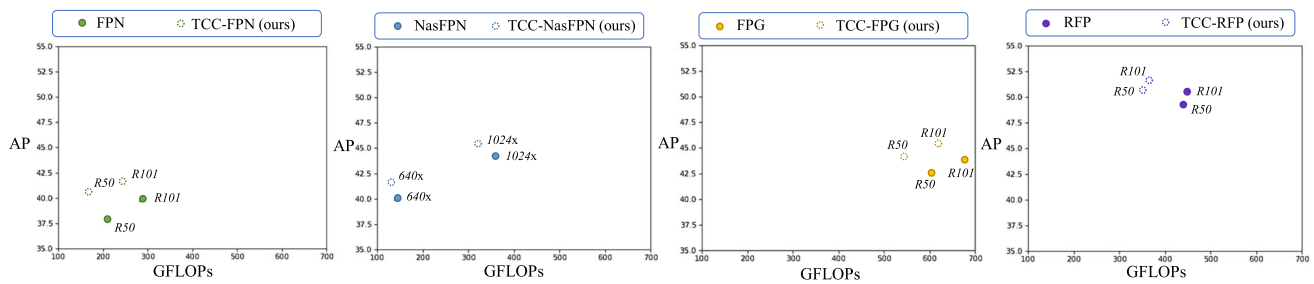


Fig. 3 Accuracy in terms of AP against computational complexity in terms of GFLOPs of whether using the proposed TCC on multi-level fusion of different feature pyramids. The experiments are performed on the *val* set of MS COCO (Lin et al., 2014) and the compared feature

pyramids include FPN (Lin et al., 2017a), NasFPN (Ghiasi et al., 2019), FPG (Chen et al., 2020), and RFP (Qiao et al., 2021). The details about these methods can be found in Sect. 4.1 and Sect. 4.2.1

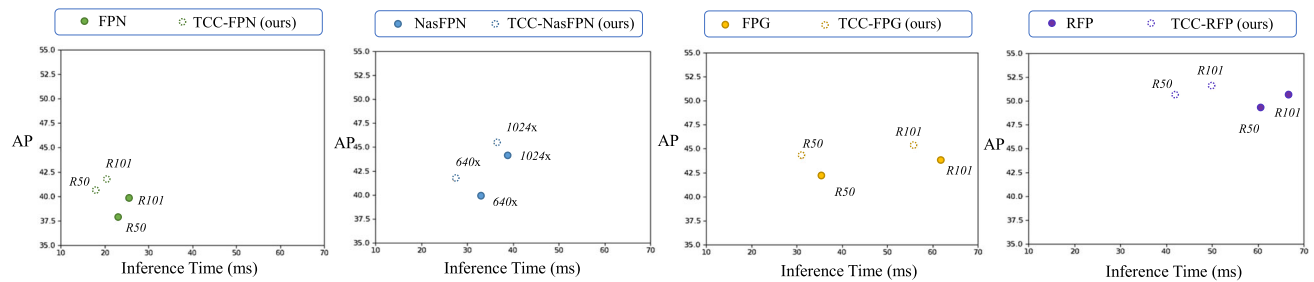


Fig. 4 Accuracy in terms of AP against inference speed in terms of inference speed (ms) of whether using the proposed TCC on multi-level fusion of different feature pyramids. The experiments are performed on the *val* set of MS COCO (Lin et al., 2014) and the compared feature

pyramids include FPN (Lin et al., 2017a), NasFPN (Ghiasi et al., 2019), FPG (Chen et al., 2020), and RFP (Qiao et al., 2021). The details about these methods can be found in Sect. 4.1 and Sect. 4.2.1

method is not sensitive to different implementation choices. In particular, we use the aforementioned feature pyramid methods, including FPN, NasFPN, FPG, and RFP, as our baseline MFF methods. We mainly follow the released code to obtain their implementation as mentioned in the previous section. In this section, we use the backbone networks, *i.e.*, ResNet50 and ResNet101, for FPN, FPG, and RFP to perform performance analysis. For the NasFPN, we mainly use the ResNet50 as the backbone, but varying the input size from 640 by 640 (640 $\times$) to 1024 by 1024 (1024 $\times$).

Using the *test-dev* set of MS COCO for evaluation. Figure 3 shows the AP results against GFLOPs. From the presented results, we can observe that our TCC is able to consistently improve the accuracy for different baseline feature pyramids and different backbone networks using less computational complexities. In particular, for a typical FPN (Lin et al., 2017a), the baseline method with ResNet50 (He et al., 2016) as backbone achieves an AP of 37.3 at a cost of 208 GFLOPs, while using our proposed TCC can help the FPN with ResNet50 achieve 40.6 at a cost of 163 GFLOPs, which means that the TCC improves AP with 3.3 points with 45 less GFLOPs. With a deeper backbone, ResNet101, the TCC also improves AP and reduces GFLOPs promisingly. For the NasFPN (Chen et al., 2020), the baseline uses different input

image sizes, *e.g.* 640 by 640 or 1024 by 1024. Using TCC, we improve the AP of NasFPN from 39.9 and 44.1 to 41.6 and 45.4 for 640 and 1024 input image sizes, respectively. Correspondingly, the GFLOPs are reduced from 139 and 355 to 128 and 326, respectively. For the FPG (Chen et al., 2020), there are similar improvements of using TCC, *i.e.* increasing AP from 42.4 and 43.8 to 44.1 and 45.2, and decreasing GFLOPs from 602 and 681 to 539 and 518 for using ResNet50 and ResNet101 as backbone networks, respectively. For the RFP (Qiao et al., 2021) that uses HTC (Chen et al., 2019a) as detection pipeline, we can observe that our method still helps deliver improved performance with decreased GFLOPs and inference speed. For example, our method improves from 50.7 in AP to 51.5 in AP with a GFLOPs from 449 to 365.

In addition to the GFLOPs, we also present the AP results against inference speeds of different methods in Fig. 4. All the speeds are obtained using the same hardware environment. Similar to the performance of AP against GFLOPs, we can observe that our method improves the AP and reduces the processing time, demonstrating the effectiveness of the proposed TCC for MFF in different feature pyramids.

Comparison with DETR methods We also performed an experiment to test our method against the DETR (Carion et

Table 1 Effects of our proposed TCC on the DETR and Deformable DETR (DDETR) (Zhu et al., 2020) comparing to the original implementation

Method	AP	AP ₅₀	AP ₇₅	GFLOPs	Params (M)
DETR Carion et al. (2020); Meng et al. (2021)	39.3	60.3	41.4	88	44
TCC-DETR (ours)	39.6	57.9	42.8	77	42
DDETR Zhu et al. (2020)	44.5	63.6	48.7	173	40
TCC-DDETR (ours)	45.0	64.3	49.3	163	39

The DETR (Carion et al., 2020) here is a modified version following the (Meng et al., 2021)

al., 2020; Meng et al., 2021) and Deformable DETR (Zhu et al., 2020). The results are shown in Table 1.

In general, the DETR (Carion et al., 2020) is a Transformer-based end-to-end object detector which directly transforms the last DCNN feature map into desired detection outputs. The Deformable DETR is a popular DETR-based method that applies deformable operations to fuse features from different levels. This fusion mechanism adaptively samples features from different locations on adjacent levels to obtain the fused feature, which is of great importance in reducing the training difficulty of DETR methods (Carion et al., 2020) and accelerating their convergence speed.

Rather than the MFF in one-stage and two-stage detectors, the DETR-based methods flatten DCNN features during encoding, making it difficult to explicitly extract contextual information. To study the effect of our proposed TCC module on the DETR-based methods, we attempt to restore the shapes of encoded features to match the shape of un-flattened DCNN features for applying our method. In the decoding stage, we then continue to flatten the DCNN features to maintain the unified detection pipeline. It is worth mentioning that the original implementation of DETR suffers from extremely long training periods. We attempt to alleviate this by incorporating the convergence acceleration tricks as introduced by the study (Meng et al., 2021). In addition, the Deformable DETR has many tricks to further improve performance. Here, we simply follow the basic setting of the Deformable DETR mainly to demonstrate the effectiveness of our method on boosting performance. Besides, the parameters of the TCC modules we applied here are the same as the TCC on FPN in our paper. Similar to other feature pyramids, we attempt to reduce the complexity of the DETR-based methods to avoid excessive computational complexity and mainly reduce 1 encoding layer in the original method when applying our TCC.

From the presented results, we can find that our method can reduce around 10 GFLOPs and can achieve around 0.3 point and 0.5 point performance gain in AP for DETR and Deformable DETR (“DDETR” in the table) on the MS COCO val dataset. Although the improvements on these DETR-based methods are not as significant as on FPN methods, we analyze that this may be because the DETR-based methods already model both short-range and long-range visual patterns within its self-attention operations during the

encoding stage. Such ability could lower the benefits of contexts as modeled in our method. Especially, the DETR only has one feature map level, which further limits the advantages of TCC in improving MFF. For instance, our method shows a slightly inferior performance compared to the DETR method on the AP₅₀ metric. This can be attributed to a reduced recall rate for larger objects. Although our method boasts a 0.12 point higher average recall rate for all objects, its average recall rate for large objects lags by 0.4 point when compared to the DETR method. We deduce that this occurs because the DETR method employs a single feature map level, causing the detection of both small and large objects to rely on the same feature map. As a result, our method may tend to prioritize context learning for improving the detection of small objects over large ones. This results in a deeper impact on the AP₅₀ performance, as it emphasizes the detection of larger objects. In contrast, our method outperforms across all metrics when using the DDETR, which incorporates multi-level feature maps, thereby validating the effectiveness of our approach.

Nevertheless, our method still shows potential in reducing complexity and improving performance promisingly. We hypothesize that, since the DETR-based methods tend to distribute almost the same attention weights on every location of the feature maps, it could be quite difficult for these methods to identify useful information from trivial information efficiently, and they need multiple encoding and decoding layers to gradually obtain better feature representation. Alternatively, our method by design has the potential of complementing these DETR-based methods by quickly focusing on locally or globally useful information, ignoring other trivial information to save efficiency and avoid the impacts of irrelevant information. The obtained results can validate the effectiveness of our TCC method. The study (Wang et al., 2022) can partially support our hypothesis by discovering that sparse feature sampling from images is key to efficient learning for DETR methods.

4.2.2 Results with High-Performance Detectors

In addition to the ResNet50 and ResNet101 based object detectors, we further introduce high-performance implementations to study the effects of TCC comparing to different state-of-the-art object detection algorithms. Table 2 shows

Table 2 Effects of TCC on different feature pyramids using more complicated backbone networks on the MS COCO *text-dev* split, comparing to other state-of-the-art object detectors

Feature pyramid	Detector	Backbone	AP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L	GFLOPs	Params (M)
FPN Lin et al. (2017a)	FRCNN Ren et al. (2015)	R101 He et al. (2016)	36.2	59.1	39.0	18.2	39.0	48.2	283	61
FPN Lin et al. (2017a)	RetinaNet Lin et al. (2017b)	R101 He et al. (2016)	39.1	59.1	42.3	21.8	42.7	50.2	315	57
FPN Lin et al. (2017a)	FCOS Tian et al. (2019)	X101 Xie et al. (2017)	44.7	64.1	48.4	27.6	47.5	55.6	439	90
FPN Lin et al. (2017a)	HTC Chen et al. (2019a)	X101 Xie et al. (2017)	47.1	63.9	44.7	22.8	43.9	54.6	521	99
Hourglass Newell et al. (2016)	CornerNet Law and Deng (2018)	Hourglass-104 Newell et al. (2016)	40.5	56.5	43.1	19.4	42.7	53.9	–	–
PANet Liu et al. (2018)	MaskRCNN He et al. (2017)	X101 Xie et al. (2017)	46.6	65.1	50.6	29.3	50.5	60.1	627	135
SMN Chen and Gupta (2017)	FRCNN Ren et al. (2015)	VGG Simonyan and Zisserman (2014)	31.6	52.2	33.2	14.4	35.7	45.8	–	–
OctaveConv Chen et al. (2019b) †		R101 He et al. (2016)	41.0	61.9	44.5	22.9	43.8	52.5	202	62
GC-NonLocal Wang et al. (2018); Cao et al. (2019) †		R101 He et al. (2016)	41.9	63.4	46.5	24.5	45.1	52.5	283	79
ReCoR Chen et al. (2021) †		R101 He et al. (2016)	42.3	64.2	45.6	25.0	45.8	54.4	303	63
FPN Lin et al. (2017a)	FRCNN Ren et al. (2015)	R101 He et al. (2016)	40.5	61.2	44.1	22.3	43.7	51.6	287	60
TCC-FPN (ours)			42.4	63.1	46.6	24.0	45.8	54.2	238	59
FPN Lin et al. (2017a)	FRCNN Ren et al. (2015)	X101 Xie et al. (2017)	41.3	62.5	45.1	24.2	44.7	51.7	284	61
TCC-FPN (ours)			43.0	64.0	47.0	24.2	45.9	55.3	237	60
NasFPN Ghiasi et al. (2019)	RetinaNet Lin et al. (2017b)	R50* He et al. (2016)	44.6	62.6	47.8	26.2	47.9	57.8	555	60
TCC-NasFPN (ours)			46.2	65.0	49.7	29.0	49.7	57.6	510	56
FPG Chen et al. (2020)	FRCNN Ren et al. (2015)	X101 Xie et al. (2017)	44.9	64.3	48.3	24.6	48.7	58.9	686	100
TCC-FPG (ours)			46.3	66.9	50.8	28.7	49.4	56.5	626	100
RFP Qiao et al. (2021)	HTC Chen et al. (2019a)	ViTAEv2-S Zhang et al. (2022)	53.1	72.2	57.7	32.8	56.2	68.3	548	115
TCC-RFP (ours)			53.5	72.6	58.2	33.3	56.7	68.6	455	114

All results of feature pyramids are reproduced using released codes. Best results are in **bold**

*Input images are resized to 1280 by 1280. †Re-implemented methods

all the results. We compared the methods like Cascade Mask RCNN (He et al., 2017), HTC (Chen et al., 2019a), FCOS (Tian et al., 2019), and so on. More specifically, for FPN and FPG that use Faster RCNN detection pipeline, we use the ResNeXt101 (Xie et al., 2017) as the backbone network to improve detection accuracy. For the NasFPN that use RetinaNet, we use input image sizes of 1280 by 1280 to achieve high-performance detection. Then, for HTC-based RFP, we apply the most recently introduced ViTAEv2 (Zhang et al., 2022) as backbone network for comparison.

Moreover, to fairly compare with other cutting-edge context modeling methods, we also report the performance of OctaveConv (Chen et al., 2019b), GCNet (Cao et al., 2019; Wang et al., 2018), and ReCoR (Chen et al., 2021) on the FRCNN (Ren et al., 2015) detection architecture using R101 (He et al., 2016) as the backbone. The OctaveConv (Chen et al., 2019b) enhances image representation by incorporating both high-frequency and low-frequency modeling processes in each convolutional block of a ResNet. GCNet (Cao et al., 2019; Wang et al., 2018) is an improved version of non-local operations that integrates global information into detection. ReCoR (Chen et al., 2021) is a method that aims to achieve comprehensive context modeling through a decomposed and recursive context modeling procedure. In addition to these methods, we also attempted to adapt other related methods like (UzKent & Ermon, 2020; Zhang et al., 2014) for generic object detection but found it to be significantly challenging. For example, the method (UzKent & Ermon, 2020) was designed exclusively for image recognition and remote sensing and did not provide a straightforward implementation for object detection. The method (Zhang et al., 2014) improved fine-grained recognition but required part annotations that are not available in object detection datasets like MS COCO. Implementing the SMN (Chen & Gupta, 2017) that uses a memory mechanism to reason object relations in an image is also very difficult. Here, we only include its performance as a reference. For the compared methods (Chen et al., 2019b; Cao et al., 2019; Chen et al., 2021), although they did not report performance for FRCNN and R101 in object detection, we were able to re-implement them based on their papers and released codes. We report the re-implemented results for comparison with our method.

According to the presented results, we can find that our method continues improving AP and reducing required GFLOPs at the same time, even on the much better implementations of detectors that may reach saturated performance. For example, the TCC improves FPN by 1.7 AP points and reduces around 40 GFLOPs. Similar improvements can also be observed for NasFPN and FPG. When using HTC-based RFP, we achieve 0.5 points' increase in AP and around 90 GFLOPs decrease in complexity compared to the HTC-based RFP using ViTAEv2-S (Zhang et al., 2022) as the backbone. We would like to mention that the ViTAEv2 is a strong back-

bone network that can make the detector easily saturate for MS COCO. Nevertheless, we still improve the AP with TCC and our TCC can outperform other compared state-of-the-art detectors when using the HTC-based RFP. When comparing our method with other context modeling methods under the same setting, we observe that our method attains favorable performance with significantly lower complexity. For instance, when applying our TCC to the FPN, our method achieves comparable performance with ReCoR (Chen et al., 2021) using only 238 GFLOPs which is 65 GFLOPs fewer than the ReCoR. This highlights the effectiveness of our method, as ReCoR was specifically designed to capture comprehensive context information as extensively as possible. These results demonstrate that our method is effective in improving detection performance while reducing complexities using contexts, emphasizing the efficiency and benefits of our method.

4.3 Ablation Study

In this section, we study the effects of different components in the TCC module. We also present ablation studies on the numbers of predicted general contextual locations.

Table 3 shows the performance of different MFF refinement methods and different components in the TCC. FPN with ResNet50 is used as the baseline, and *val* set of MS COCO is used for evaluation. According to the results, we can find that the feature pyramid achieves degraded performance without refining MFF results. We then test the performance of using only local context modeling implemented by the dilated convolution, the performance of using the condensed contexts but without Transformer attention $\mathcal{A}^i$ in Eq. 10, and the performance of using the complete TCC module. According to the results, we can observe progressive improvement by gradually introducing each component in the TCC. This illustrates that the contextual information is helpful for improving MFF in the feature pyramid and that the proposed condensed contexts are effective for collecting rich contextual information. Then, by further including the Transformer, our proposed TCC module achieves the highest accuracy and introduces smaller computational complexity than the baseline method.

In Table 4, we present the study of the effects of using different numbers of summarized general contextual features. We have studied the numbers ranging from 1 to 8, and the related detection accuracy and computational complexities are listed for comparison. According to the presented results, we can find that using 4 general contextual features achieves the best speed/accuracy trade-off. Using 1 or 2 general contextual features are beneficial but can be improved while using 8 general contextual features makes the detector saturates in accuracy. These results suggest that the global contexts are helpful for improving MFF in feature pyra-

Table 3 Ablation study of different multi-level feature fusion methods in FPN on MS COCO *val* set

MMF refinement	AP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L	GFLOPs	Params(M)
No refinement	36.9	58.1	40.0	21.5	40.7	47.6	156	40
Refine with original FPN (Convolution 3×3)	37.4	58.1	40.4	21.2	41.0	48.1	207	42
Refine with local contexts only (Dilated convolution)	38.9	60.1	42.3	22.9	42.6	51.2	159	40
Refine with condensed contexts only (without using $\mathcal{A}^i$ in Eq. 10)	39.3	60.2	42.9	23	24.7	51.1	161	41
Refine with TCC	40.3	61.8	43.5	23.7	43.9	52.6	162	41

Table 4 Effects of predicting and using different numbers of general contextual features when condensing contexts in TCC

Number of general contextual features	AP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L
1	39.8	60.9	43.2	22.9	43.4	51.7
2	40.2	61.3	43.9	23.7	43.5	52.5
4	40.3	61.8	43.5	23.7	43.9	52.6
8	40.2	61.7	43.5	24.5	43.9	52.8

Table 5 Comparison of different Transformer architecture implementation choices in TCC

	NE	ND	NC	NH	AP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L	GFLOPs	Params (M)
Baseline FPN	–	–	–	–	36.9	58.1	40.0	21.5	40.7	47.6	207	42
TCC-FPN (ours)	0	1	–	1	40.3	61.8	43.5	23.7	43.9	52.6	162	41
	0	1	64	1	40.0	61.5	43.4	23.1	43.7	53.5	163	42
	0	1	512	1	40.1	61.6	42.8	22.8	43.8	54.2	164	44
TCC-FPN with other	0	1	512	4	40.6	61.8	44.4	23.8	44.1	53.9	178	44
Transformer architectures	0	1	512	8	40.6	61.4	44.7	25.4	45.2	53.4	195	49
	1	1	512	4	40.9	61.4	44.8	23.3	44.2	53.9	218	45
	2	2	512	4	40.8	63.0	44.1	24.0	44.4	54.2	265	47

NE number of encoding layers; *ND* number of decoding layers; *NC* number of feature channels for multi-layer perception; *NH* number of heads

mids and they can be condensed into features from only a few general contextual locations. It is worth mentioning that varying the number of considered general contextual features does not change the computational complexity too much, e.g. using 8 general contextual features would introduce less than 1 GFLOPs and less than 1 million (M) extra parameters.

In Table 5, we present the study of different choices for implementing the Transformer architecture in our proposed TCC. As described previously, our proposed TCC mainly follow the original implementation of Transformer (Carion et al., 2020; Dosovitskiy et al., 2021) to implement our decoder structure. Different from original Transformer, we simplified its architecture greatly to avoid excessive computational costs. Please refer to Sect. 3.3 for more details. In the presented results, we can find that introducing more complicated Transformer architectures can consistently boost detection performance, demonstrating that the condensed contexts in TCC are truly helpful for detection. However, more promising performance is usually accompanied with much greater computational complexity. For example, using 1 encoding layer, 1 decoding layers, 512 feature channels for multi-layer perception, and 4 heads, we can achieve the highest average

precision scores, but we need around additional 54 GFLOPs and 6 millions of parameters than our proposed TCC module. Since this can hinder the efficiency of our proposed module without introducing too much performance gain, we do not intend to use highly complicated Transformer in our the TCC. In addition, although using 1 decoding layer, 512 feature channels, and 1 head can bring promising improvement without introducing too much complexity, we found in the experiment that this architecture requires longer processing time, increasing from around 18 ms (our TCC) to around 27 ms. This could attribute to the reshaping operations and multi-layer perception operations, making them slower than the FPN baseline method. Furthermore, we would also like to mention that using more than 2 encoding layers and 2 decoding layers at the same time can easily blow up our GPU memory, making it infeasible for our experiment.

4.4 Performance on Pascal VOC

In addition to MS COCO, we further validate the effectiveness of our TCC on the PascalVOC (Everingham et al., 2010) dataset. The results are shown in Table 6. This dataset con-

Table 6 Performance of TCC (ours) for different feature pyramids on the Pascal VOC 07 (Everingham et al., 2010) test set

Method	AP	Plane	Bicycle	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Table	Dog	Horse	Motor	Person	Plant	Sheep	Sofa	Train	TV	GFLOPs	Params (M)
CC Zeng et al. (2017)	81.1	80.9	84.8	83.0	75.9	72.3	88.9	88.4	90.3	66.2	87.6	74.0	89.5	89.3	83.6	79.6	55.2	83.4	81.0	87.8	80.7	–	–
RFCN Dai et al. (2016)	80.5	79.9	87.2	81.5	72.0	69.8	86.8	88.5	89.8	67.0	88.1	74.5	89.8	90.6	79.9	81.2	53.7	81.8	81.5	85.9	79.9	–	–
CR Chen et al. (2018)	82.2	84.7	88.2	83.1	76.2	71.1	87.9	88.7	89.5	68.7	88.6	78.2	89.5	88.7	84.8	86.2	55.4	84.7	82.0	86.0	81.7	–	–
FPN	80.6	84.6	87.8	84.1	71.4	70.1	85.1	88.2	89.1	65.5	85.3	75.4	88.1	87.1	85.0	85.9	54.4	82.4	76.0	85.3	80.4	131	41
TCC-FPN	81.4	85.7	87.6	84.2	72.6	71.0	86.3	87.9	89.6	68.0	87.4	73.5	88.5	88.0	83.6	86.4	58.5	84.0	79.1	86.4	79.4	103	40
NasFPN	80.8	83.9	87.5	85.6	73.7	58.7	87.8	87.6	89.8	62.3	87.5	75.5	88.7	89.2	85.2	83.2	57.0	84.3	79.0	87.4	82.0	82	58
TCC-NasFPN	81.2	85.5	86.5	86.2	75.3	60.8	87.9	88.1	89.5	63.8	88.7	73.9	88.4	88.9	85.5	83.0	59.4	84.4	79.8	87.3	81.7	73	52
FPG	81.1	86.3	87.8	84.0	69.3	67.3	86.3	88.4	89.9	66.2	87.4	77.5	88.9	87.4	87.0	86.1	55.8	84.3	77.3	84.7	79.4	360	79
TCC-FPG	81.7	88.2	87.4	83.9	72.7	70.8	86.5	88.6	89.2	68.5	84.3	77.8	87.9	87.2	85.7	85.8	59.2	85.1	78.3	86.5	81.2	296	75
RFP	82.0	87.4	88.1	80.6	74.5	73.1	88.1	89.0	90.0	68.6	88.7	74.0	89.4	88.9	87.2	86.2	57.7	86.3	73.9	86.9	78.8	156	124
TCC-RFP	82.4	88.2	88.0	85.8	75.0	74.0	87.0	89.0	90.0	68.5	88.4	75.1	89.5	89.2	85.3	86.4	57.8	86.7	77.7	86.8	83.2	105	123

We use ResNet50 as the backbone for all the tested feature pyramid methods. Please refer to Sect. 4.4 for more details. Best results are in **bold**

tains 39k images for training and 5k images for testing. It has labels for 20 object categories. The appeared objects in this dataset usually have medium or large sizes. We follow the popular protocol and use its 07 and 12 dataset for training and use 07 dataset for evaluation. We tested the performance of TCC on FPN (Lin et al., 2017a), NasFPN (Ghiasi et al., 2019), FPG (Chen et al., 2020), and RFP (Qiao et al., 2021). We mainly use ResNet50 as our backbone network. We also change the image resolution for the evaluated methods. For FPN and FPG, we use the image size of 1000 by 600 (the shorter size is 600, but the longer size does not exceed 1000). For NasFPN, we use 512 by 512 as the image size. We train networks with 12 epochs. We would also like to mention that we are not able to directly apply RFP (Qiao et al., 2021) on the HTC (Chen et al., 2019a) method in the original implementation because the HTC requires instance segmentation annotation that the Pascal VOC dataset does not provide. We instead use the Cascade RCNN (Cai & Vasconcelos, 2018) rather than HTC to evaluate the RFP on the Pascal VOC dataset.

In Table 6, we present the average precision (AP) performance at the IoU threshold of 0.5. From the results, we can find that our TCC module continues to improve different baseline methods as well as reduce their complexities. For example, the TCC improves the FPN baseline by 0.8 point in AP and reduce the complexity by around 30 GFLOPs. By boosting the FPG with TCC, we can achieve the highest AP among the compared methods while reducing the complexity by nearly 60 GFLOPs at the same time. These results demonstrate that the proposed TCC can be effective for different datasets. We would like to further mention that the PascalVOC contains a lot of large objects and thus is easier than MS COCO, which enables all the compared methods to achieve promising AP scores. Nevertheless, even for easier objects, our proposed TCC still shows advantages in further improving the accuracy with much more efficient processing complexity for different feature pyramids. In particular, compared with other context modeling methods like (Zeng et al., 2017), our method still achieves the highest AP scores.

4.5 Qualitative Analysis

In addition to quantitative analysis, we also attempt to visualize the forms of contextual information condensed by the proposed TCC and compare our method with the baseline method visually. The FPN with ResNet50 is selected as the baseline in this section.

Firstly, we present the spatial arrangements of learned locally concentrated and globally summarised contexts *w.r.t.* the feature from a specific local entry. Figure 5 shows the visualization results. Note that the results of the last TCC on the deepest-level feature is selected for visualization. In this figure, we mainly present the locations of the specific local



Fig. 5 Visualization of the spatial arrangements of condensed contexts and their relations to an entity from a specific local entry

feature (pink cross), its locally concentrated contexts (purple dotted rectangular), and the summarized general contextual features (coloured dots), respectively. In addition to the locations, we also indicate the relations of different condensed contexts *w.r.t.* the specific local feature. The relation orders (illustrated as coloured numbers: 1 for the most related; 5 for the least related) are also presented along side the corresponding contexts. From the figure, we can observe that the summarized general contextual features are generally located at meaningful instances that can represent the theme of the scene presented in the image. For example, in the top middle figure, the learned general contextual features are distributed on vehicles, buildings, and poles, which are reasonable to describe the urban scene of the image. Besides, the locations of summarized general contextual features, the relations between condensed contexts and the local feature learned by the Transformer in TCC also look natural. As an example, in the top-left figure, it shows that the TCC searches for human feet, paddles, and local surrounding visual information as the more useful cues for accurate detection of the surfboard, while the sea and the upper-body of the human are less important, which does not violate the common sense. These results have clearly demonstrated that the TCC can effectively model the relations between condensed contexts and local features and then help achieve effective detection based on the contexts and the modeled relations.

Besides the contexts, we further present the comparison of detection results in Fig. 6. According to the presented examples, we can find that our proposed TCC can help the

baseline FPN improve the recognition quality promisingly. More specifically, on the top-left figure, the FPN with TCC can produce more accurate recognition of the train, avoiding many false positive detection results from the baseline FPN. In addition, we can observe the 2-nd row left figure and the top right figure, which show that our method helps the FPN provide more accurate detected bounding boxes to cover the objects more compactly. In other presented examples, we can find that the TCC continues to improve the FPN by enhancing the recognition quality and reducing the false alarms. These results demonstrate that our method is effective to help the detector understand the scene more robustly by introducing contexts.

5 Conclusions

In this study, we propose a Transformer-based context condensation module to achieve improved MFF results in different feature pyramids with a reduced complexity. By decomposing rich context information into a locally concentrated representation and a globally summarised representation, we obtain condensed contexts that can avoid exhausted context modeling to achieve promising refinement of fusion results. With the help of a Transformer decoder, our proposed TCC module can help boost detection accuracy with a smaller computational cost. In practice, we inject the TCC into 4 different feature pyramids, including FPN, NasFPN, FPG, and RFP, we can observe consistent improve-

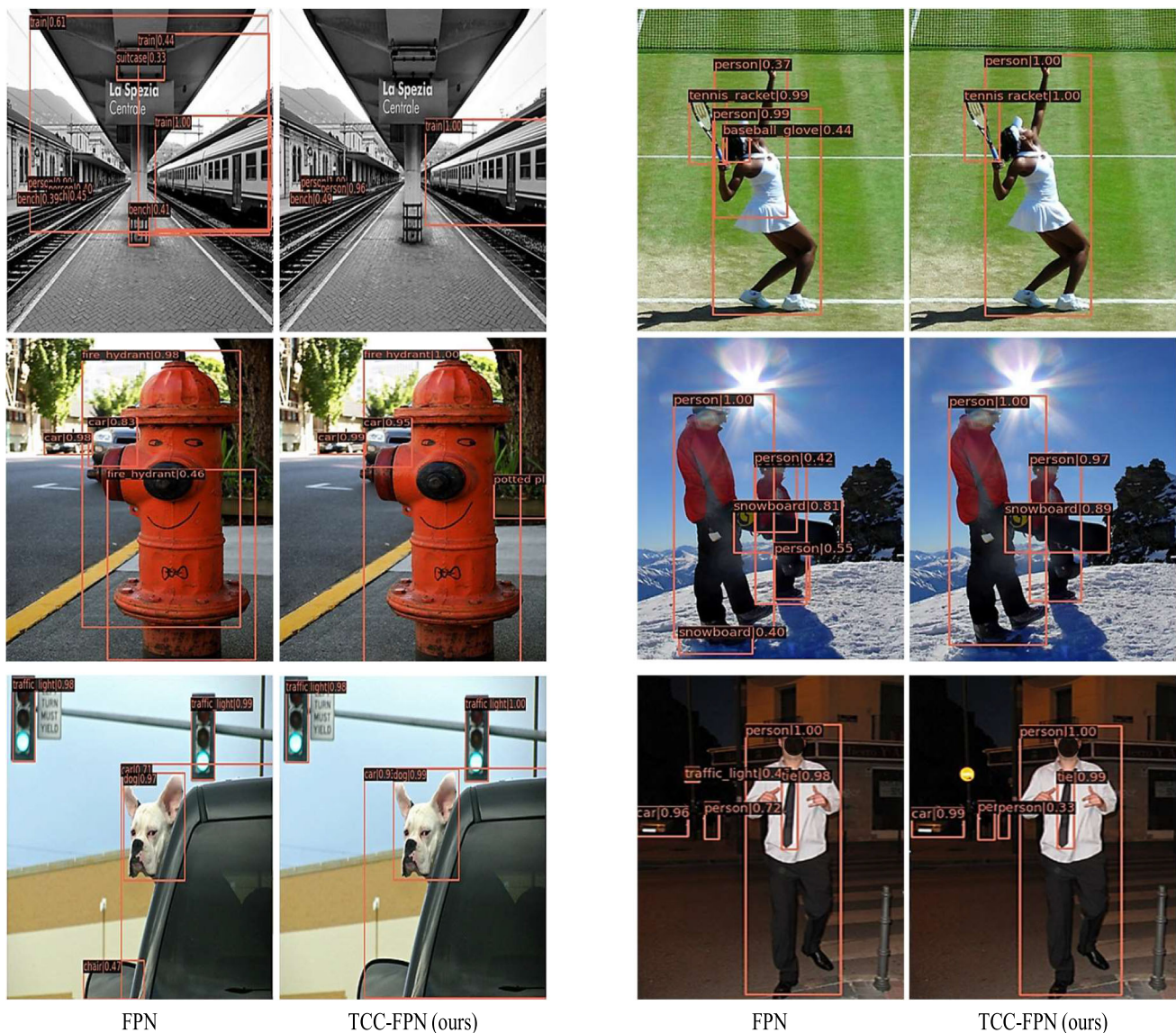


Fig. 6 Visual examples of the detection results of TCC compared to the baseline FPN method

ments, which validate the effectiveness of our method. We hope that our introduction of condensed contexts can open up a novel view of using contexts to improve visual understanding performance.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions.

Data Availability The datasets generated during and/or analysed during the current study are available in the MS COCO repository, <https://cocodataset.org/>.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material

in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Alayrac, J.B., Donahue, J., & Luc, P., et al. (2022). Flamingo: A visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*.
- Baker, B., Gupta, O., & Naik, N., et al. (2016). Designing neural network architectures using reinforcement learning. In *ICLR*.
- Biederman, I. (2017). On the semantics of a glance at a scene. *Perceptual organization* (pp. 213–253). Routledge.

- Cai, Z., & Vasconcelos, N. (2018). Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 6154–6162).
- Cai, Z., Fan, Q., & Feris, R.S., et al. (2016). A unified multi-scale deep convolutional neural network for fast object detection. In *European Conference on Computer Vision* (pp. 354–370). Springer.
- Cao, Y., Xu, J., & Lin, S., et al. (2019). Gcnet: Non-local networks meet squeeze-excitation networks and beyond. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*.
- Carion, N., Massa, F., & Synnaeve, G., et al. (2020). End-to-end object detection with transformers. In *European Conference on Computer Vision* (pp. 213–229). Springer.
- Chen, K., Pang, J., & Wang, J., et al. (2019a). Hybrid task cascade for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 4974–4983).
- Chen, K., Cao, Y., & Loy, C.C., et al. (2020). Feature pyramid grids. arXiv preprint [arXiv:2004.03580](https://arxiv.org/abs/2004.03580).
- Chen, X., & Gupta, A. (2017). Spatial memory for context reasoning in object detection. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 4086–4096).
- Chen, Y., Fan, H., & Xu, B., et al. (2019b). Drop an octave: Reducing spatial redundancy in convolutional neural networks with octave convolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 3435–3444).
- Chen, Z., Huang, S., & Tao, D. (2018). Context refinement for object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)* (pp. 71–86).
- Chen, Z., Zhang, J., & Tao, D. (2021). Recursive context routing for object detection. *International Journal of Computer Vision*, 129(1), 142–160.
- Chen, Z., Zhang, J., & Tao, D. (2022). Recurrent glimpse-based decoder for detection with transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 5260–5269).
- Dai, J., Li, Y., & He, K., et al. (2016). R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, 29.
- Dai, J., Qi, H., & Xiong, Y., et al. (2017). Deformable convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 764–773).
- Dosovitskiy, A., Beyer, L., & Kolesnikov, A., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.
- Dosovitskiy, A., Beyer, L., & Kolesnikov, A., et al. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*.
- Everingham, M., Van Gool, L., Williams, C. K., et al. (2010). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2), 303–338.
- Gao, P., Zheng, M., & Wang, X., et al. (2021). Fast convergence of detr with spatially modulated co-attention. arXiv preprint [arXiv:2101.07448](https://arxiv.org/abs/2101.07448).
- Ghiasi, G., Lin, T.Y., & Le, Q.V. (2019). Nas-fpn: Learning scalable feature pyramid architecture for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 7036–7045).
- Gidaris, S., & Komodakis, N. (2015). Object detection via a multi-region and semantic segmentation-aware CNN model. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 1134–1142).
- He, K., Zhang, X., & Ren, S., et al. (2016). Deep residual learning for image recognition. In *CVPR* (pp 770–778).
- He, K., Gkioxari, G., & Dollár, P., et al. (2017). Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 2961–2969).
- Hu, H., Gu, J., & Zhang, Z., et al. (2018). Relation networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 3588–3597).
- Huang, G., Chen, D., & Li, T., et al. (2017). Multi-scale dense convolutional networks for efficient prediction. arXiv preprint [arXiv:1703.09844](https://arxiv.org/abs/1703.09844).
- Jaegle, A., Borgeaud, S., & Alayrac, J.B., et al. (2021a). Perceiver io: A general architecture for structured inputs & outputs. arXiv preprint [arXiv:2107.14795](https://arxiv.org/abs/2107.14795).
- Jaegle, A., Gimeno, F., & Brock, A., et al. (2021b). Perceiver: General perception with iterative attention. In *International conference on machine learning*, PMLR, (pp. 4651–4664).
- Komodakis, N., & Gidaris, S. (2016). Attend refine repeat: Active box proposal generation via in-out localization. In *BMVC*.
- Kong, T., Sun, F., & Yao, A., et al. (2017). Ron: Reverse connection with objectness prior networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 5936–5944).
- Law, H., & Deng, J. (2018). Cornernet: Detecting objects as paired key-points. In *Proceedings of the European Conference on Computer Vision (ECCV)*, (pp. 734–750).
- Lin, T.Y., Maire, M., & Belongie, S., et al. (2014). Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, (pp. 740–755). Springer.
- Lin, T.Y., Dollár, P., & Girshick, R., et al. (2017a). Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, (pp 2117–2125).
- Lin, T.Y., Goyal, P., & Girshick, R., et al. (2017b). Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, (pp. 2980–2988).
- Liu, S., Qi, L., & Qin, H., et al. (2018). Path aggregation network for instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, (pp. 8759–8768).
- Liu, W., Anguelov, D., & Erhan, D., et al. (2016). Ssd: Single shot multibox detector. In *European Conference on Computer Vision* (pp. 21–37). Springer.
- Liu, Z., Lin, Y., & Cao, Y., et al. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 10012–10022).
- Meng, D., Chen, X., & Fan, Z., et al. (2021). Conditional detr for fast training convergence. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 3651–3660).
- Newell, A., Yang, K., & Deng, J. (2016). Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision* (pp. 483–499). Springer.
- Qi, C.R., Su, H., & Mo, K., et al. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp 652–660).
- Qiao, S., Chen, L.C., & Yuille, A. (2021). Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 10213–10224).
- Ren, S., He, K., & Girshick, R., et al. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems* 28.
- Rumelhart, D. E., & McClelland, J. L. (1982). An interactive activation model of context effects in letter perception: Ii. the contextual enhancement effect and some tests and extensions of the model. *Psychological Review*, 89(1), 60.
- Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint [arXiv:1409.1556](https://arxiv.org/abs/1409.1556).
- Sun, K., Xiao, B., & Liu, D., et al. (2019). Deep high-resolution representation learning for human pose estimation. In *Proceedings of*

- the *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 5693–5703).
- Tay, Y., Dehghani, M., & Bahri, D., et al. (2020). Efficient transformers: A survey. arXiv preprint [arXiv:2009.06732](https://arxiv.org/abs/2009.06732).
- Tian, Z., Shen, C., & Chen, H., et al. (2019). Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 9627–9636).
- Torralba, A. (2003). Contextual priming for object detection. *International Journal of Computer Vision*, 53, 169–191.
- Touvron, H., Cord, M., & Douze, M., et al. (2021). Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, PMLR, (pp. 10347–10357).
- Uzkent, B., & Ermon, S. (2020). Learning when and where to zoom with deep reinforcement learning. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 12345–12354).
- Vaswani, A., Shazeer, N., & Parmar, N., et al. (2017). Attention is all you need. In *NIPS*.
- Wang, W., Cao, Y., & Zhang, J., et al. (2021a). Fp-detr: Detection transformer advanced by fully pre-training. In: *International Conference on Learning Representations*.
- Wang, W., Zhang, J., & Cao, Y., et al. (2022). Towards data-efficient detection transformers. In *European Conference on Computer Vision* (pp 88–105). Springer.
- Wang, X., Girshick, R., & Gupta, A., et al. (2018). Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 7794–7803).
- Wang, Y., Zhang, X., & Yang, T., et al. (2021b). Anchor detr: Query design for transformer-based detector. arXiv preprint [arXiv:2109.07107](https://arxiv.org/abs/2109.07107).
- Wojke, N., & Bewley, A. (2018). Deep cosine metric learning for person re-identification. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)* (pp. 748–756). IEEE.
- Xie, S., Girshick, R., & Dollár, P., et al. (2017). Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1492–1500).
- Xu, H., Yao, L., & Zhang, W., et al. (2019). Auto-fpn: Automatic network architecture adaptation for object detection beyond classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 6649–6658).
- Xu, Y., Zhang, Q., & Zhang, J., et al. (2021). Vitae: Vision transformer advanced by exploring intrinsic inductive bias. In *Advances in Neural Information Processing Systems* 34.
- Yu, F., Wang, D., & Shelhamer, E., et al. (2018). Deep layer aggregation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2403–2412).
- Zeng, X., Ouyang, W., Yan, J., et al. (2017). Crafting gbd-net for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(9), 2109–2123.
- Zhang, N., Donahue, J., & Girshick, R., et al. (2014). Part-based r-cnns for fine-grained category detection. In: *European conference on computer vision* (pp. 834–849). Springer.
- Zhang, Q., Xu, Y., & Zhang, J., et al. (2022). Vitaev2: Vision transformer advanced by exploring inductive bias for image recognition and beyond. arXiv preprint [arXiv:2202.10108](https://arxiv.org/abs/2202.10108).
- Zhao, H., Shi, J., & Qi, X., et al. (2017). Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2881–2890).
- Zhao, Q., Sheng, T., & Wang, Y., et al. (2019). M2det: A single-shot object detector based on multi-level feature pyramid network. In *Proceedings of the AAAI Conference on Artificial Intelligence* (pp 9259–9266).
- Zhu, X., Su, W., & Lu, L., et al. (2020). Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint [arXiv:2010.04159](https://arxiv.org/abs/2010.04159).
- Zoph, B., & Le, Q.V. (2017). Neural architecture search with reinforcement learning. In *ICLR*.
- Zoph, B., Vasudevan, V., & Shlens, J., et al. (2018). Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 8697–8710).

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.