PnP-DETR: Towards Efficient Visual Analysis with Transformers

Tao Wang[1,3*] Li Yuan[4*] Yunpeng Chen[2] Jiashi Feng[4] Shuicheng Yan[4]

[1] Institute of Data Science, National University of Singapore [2] Yitu Technology

[3] Integrative Science and Engineering Programme, NUS Graduate School, National University of Singapore

[4] Department of Electrical and Computer Engineering, National University of Singapore

twangnh@gmail.com ylustcnus@gmail.com yunpeng.chen@yitu-inc.com

jshfeng@gmail.com shuicheng.yan@gmail.com

Abstract

Recently, DETR [3] pioneered the solution of vision tasks with transformers, it directly translates the image feature map into the object detection result. Though effective, translating the full feature map can be costly due to redundant computation on some area like the background. In this work, we encapsulate the idea of reducing spatial redundancy into a novel poll and pool (PnP) sampling module, with which we build an end-to-end PnP-DETR architecture that adaptively allocates its computation spatially to be more efficient. Concretely, the PnP module abstracts the image feature map into fine foreground object feature vectors and a small number of coarse background contextual feature vectors. The transformer models information interaction within the fine-coarse feature space and translates the features into the detection result. Moreover, the PnP-augmented model can instantly achieve various desired trade-offs between performance and computation with a single model by varying the sampled feature length, without requiring to train multiple models as existing methods. Thus it offers greater flexibility for deployment in diverse scenarios with varying computation constraint. We further validate the generalizability of the PnP module on ***panoptic segmentation*** *and the recent transformer-based image recognition model* ***ViT*** *[7] and show consistent efficiency gain. We believe our method makes a step for efficient visual analysis with transformers, wherein spatial redundancy is commonly observed. Code and models will be available.*

1. Introduction

Object detection is a fundamental computer vision task aiming to recognize object instances in the image and localize them with precise bounding boxes. Modern detectors address this set prediction task mainly with proxy learning objectives, *i.e.*, regressing offset from pre-defined anchor boxes [23, 18] or boundaries from grid locations [27, 34, 9]. Those heuristic designs not only complicate the model design but also require hand-crafted post-processing for duplicate removal. A recent method DETR [3] eliminates those hand-crafted designs and achieves end-to-end object detection. It builds an effective set prediction framework on top of convolution feature maps with transformers [28] and shows competitive performance to the two-stage Faster R-CNN [23] detector. The image feature map is flattened in the spatial dimension into one-dimensional feature vectors. The transformer then processes them with its strong attention mechanism to generate the final detection list.

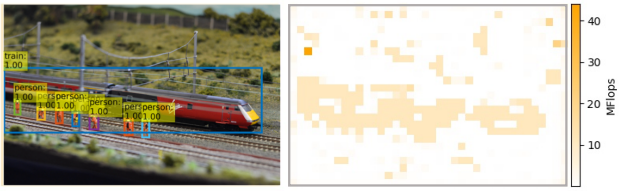

Figure 1. Left: Detection result. Right: Transformer computation density map. Proposed method allows the model to adaptively allocate computation spatially and avoid computation expenditure on less informative background area.

Albeit simple and effective, applying the transformer networks to a image feature map can be computationally costly, mainly due to the attention operation [28] over the long flattened feature vectors. These features may be redundant: natural images often contain enormous background areas apart from the interested objects, which may occupy large part in the corresponding feature representation; also, some discriminative feature vectors may already suffice for detecting the objects. Existing works improving the transformer efficiency mainly focus on accelerating the attention operation [16, 15, 29, 5], and few of them consider the spatial redundancy discussed above.

To address the above limitation, we develop a learnable

*Work done during an internship at Yitu Tech.

poll and pool (PnP) sampling module. It aims to compress an image feature map into an abstracted feature set composed of fine feature vectors and a small number of coarse feature vectors. The fine feature vectors are deterministically sampled from the input feature map to capture the fine foreground information, which thus are crucial for detecting the objects. The coarse feature vectors aggregate information from the background locations, and the resulting contextual information helps better recognize and localize the objects. A transformer then models the information interaction within the fine-coarse feature space and obtains the final result. As the abstracted set is much shorter than the directly flattened image feature map, the transformer computation is reduced significantly and mainly distributed over the foreground locations. Our approach is orthogonal to the approaches improving the transformer efficiency [16, 15, 29, 5] and can be further combined with them to obtain more efficient models.

Concretely, the PnP module is composed of two core sub-modules: a *poll sampler* and a subsequent *pool sampler*. The *poll sampler* incorporates a content-aware meta-scoring network that learns to predict the infromativeness score of the feature vector at each spatial location. The feature vectors are then ranked spatially with the informativeness scores and a subset of most informative feature vectors are selected. The subsequent *pool sampler* dynamically predicts attention weights on the non-sampled feature vectors and aggregates them into a small number of feature vectors that summarize the background information. Similar to the region proposal networks [23], the PnP module also aims to extract object-relevant information, but is end-to-end learned without explicit objective like object bounding box regression. We build a PnP-DETR with the PnP module, which operates on the fine-coarse feature space and adaptively allocates its transformer computation in the spatial domain. Fig. 1 is an example detection with computation density map (refer to Sec. 4.2 for details of the map construction). Existing methods of improving model efficiency still need train multiple models of different complexities for achieving various trade-offs of computation and performance. Compared with them, the proposed PnP sampling allows the transformer to work with a variable number of input feature vectors and achieve instant computation and performance trade-off.

We conduct extensive experiments on the COCO benchmark, and the results show PnP-DETR effectively reduces the cost and achieves dynamic computation and performance trade-off. For example, without bells and whistels, a single PnP-DETR-DC5 obtains a 42.7 AP with 72% reduction of transformer computation compared to the 43.3 AP baseline and competitive 43.1 AP with 56% reduction. We further validate the efficiency gain with panoptic segmentation and the recent vision transformer model (ViT [7]). For example, PnP-ViT achieves near half of FLOPs reduction with only 0.3 drop of accuracy. To summarize, the contributions are:

- We identify the spatial redundancy issue of the image feature map, which causes excessive computation of the transformer network in a DETR model. We therefore propose to abstract the feature map, so as to significantly reduce the model computation.
- To realize the feature abstraction, we design a novel two-step poll-and-pool sampling module. It first employs a poll sampler to extract the foreground fine feature vectors, and then utilizes a pool sampler to obtain the contextual coarse feature vectors.
- We then build PnP-DETR, wherein the transformer operates on the abstract fine-coarse feature space and adaptively distributes the computation in the spatial domain. PnP-DETR is more efficient and achieves instant computation and performance trade-off with a single model, by varying length of the fine feature set.
- The PnP sampling module is general and end-to-end learned without explicit supervision like the region proposal networks [23]. We further validate it on panoptic segmentation and recent ViT model [7] and show consistent efficiency gain. We believe our method provides useful insights for future research on efficient solutions of vision tasks with transformers.

2. Related Work

Object Detection In recent years performance of object detection has been substantially improved [14, 13, 23, 20, 18, 27] over traditional approaches [26, 10]. Those modern methods mainly address the task with a relaxed learning objective, *i.e.*, learning on a set of matched positive anchor box samples and predicting with post-processing (NMS) to suppress duplicates. The handcraft designs Recently, [3] proposed an end-to-end DETR framework that learns an explicit set based objective with transformers [28], showing decent performance compared to previous two-stage methods [23]. Our work aims at improving efficiency of end-to-end objectors by reducing spatial redundancy. Compared to most recent *deformable* DETR [35] that improves the attention efficiency, we aim to directly compress the feature map, which is from different perspective and could be potentially combined together. For example, by implementing bilinear interpolation kernel in the irregular sampled space [24, 25] to enable the learning of deformable offset prediction.

Sparse Execution and Sampling Lots of works explored sparse execution in convolution layers [12, 22, 2, 30, 11,

6, 11], saving computation by avoiding convolution operations on some less informative spatial locations. In this work, we are partially inspired by the sparse convolution and explore sparse execution of transformers [28] by developing a dynamic image feature sampling method for efficient subsequent processing. Our work is also related to literature on learning a sampling policy for point cloud understanding tasks [8, 17, 21]. Different from these works where sampling is achieved by new data point generation, we directly address discrete sampling by using a novel sampling as ranking strategy.

3. Method

We first revisit the DETR [3]. Then we elaborate the proposed feature abstraction scheme, followed by detailed design of the PnP Sampling that realizes the abstraction. Finally we illustrate the PnP-augmented models and their advantages. We denote constants, scalars, vectors, tensors and sets as upper-case, lower-case, bold lower-case, bold upper-case and blackboard-bold upper-case letters, respectively, *e.g.*, N, i, $\mathbf{f}$, $\mathbf{F}$, $\mathbb{F}$.

3.1. Preliminaries

Without loss of generality, DETR [3] first utilizes a backbone convolution network $\mathcal{C}$ with parameters θ_c to extract the image feature map $\mathbf{F}$:

$$\mathbf{F} = \mathcal{C}(\mathbf{I}, \theta_c) \tag{1}$$

$\mathbf{F}$ can be viewed as a grid structured feature vector set $\mathbb{F}$:

$$\mathbb{F} = \{\mathbf{f}_{ij} \in \mathbb{R}^C | i = 1, \dots, H, j = 1, \dots, W\} \tag{2}$$

Here $\mathbf{f}_{ij}$ is the feature vector at location (i, j), C is the number of feature channels, H, W are the height and width of the extracted image feature map. The grid-structured feature set $\mathbb{F}$ is then viewed as a set of high-level visual tokens with strong semantic information and translated into the detection result with a transformer $\mathcal{T}$ parametrized with θ_t:

$$\{(cls_k, box_k) | k = 1, \dots, D\} = \mathcal{T}(\mathbb{F}, \theta_t) \tag{3}$$

(cls_k, box_k) denotes one detected object with category and bounding box, the number of detections is fixed to D.

An intrinsic limitation of the grid structured visual token representation $\mathbb{F}$ is that it spans uniformly over the spatial locations and covers a large amount of background. Although the transformer can attend to different areas with its strong attention capability, the computation does not benefit from this advantage and is uniformly distributed over the spatial domain. This deviates from our expectation that the processing power can be dynamically assigned to more relevant area like foreground locations while focusing less on area like background of a visual scene.

3.2. Feature Abstraction

We propose a feature abstraction scheme to address the above limitation. It obtains two sets of feature vectors for compact feature representation:

$$\mathbb{F}_f = \{\mathbf{f}_n \in \mathbb{R}^C | n = 1, \dots, N\} \tag{4}$$

$$\mathbb{F}_c = \{\mathbf{f}_m \in \mathbb{R}^C | m = 1, \dots, M\} \tag{5}$$

The fine feature set $\mathbb{F}_f$ is discretely sampled from the full set $\mathbb{F}$, containing fine information that is essential for recognizing and detecting the objects. The coarse feature set $\mathbb{F}_c$ is obtained by aggregating information from multiple spatial locations and encodes background contextual information. Together, they form an abstraction set $\mathbb{F}^*$:

$$\mathbb{F}^* = \mathbb{F}_f \cup \mathbb{F}_c \tag{6}$$

$\mathbb{F}^*$ encodes all necessary high-level information for detecting the objects within an image and is passed to a transformer for generating the object detection result. Refer to supplementary for a theoretical analysis on the computation saving. The feature abstraction scheme can also be viewed as a tokenization formulation that suits well for solving vision tasks with transformers.

3.3. Poll and Pool (PnP) Sampling

The above abstraction scheme need address two challenges. **1)** The fine set requires deterministic binary sampling, which is non-differentiable. A handcrafted sampler can be learnt with some intermediate objectives, *e.g.*, the region proposal networks [23] or point proposal networks [34, 9], which is however incompatible with end-to-end learning, and the handcraft sampling rules may not be optimal. **2)** To extract a compact, coarse feature set only focusing on background contextual information is difficult. We divide the abstraction scheme into two steps and develop a poll sampler and a pool sampler to realize it. The poll sampler first samples some feature vectors from the full set $\mathbb{F}$; the pool sampler then dynamically aggregates the remaining non-sampled feature vectors into a small number of coarse feature vectors. Fig. 2 is an overview of the proposed method. The samplers are deterministic and end-to-end learned with negligible computation cost.

Poll Sampler The poll sampler aims to obtain a fine feature set $\mathbb{F}_f$. Since explicitly learning a binary sampler is infeasible, we develop a sample as ranking strategy. We use a small meta-scoring network to predict the informativeness score for each spatial feature location (i, j):

$$s_{ij} = \text{ScoringNet}(\mathbf{f}_{ij}, \theta_s) \tag{7}$$

The larger the score is, the more informative the feature vector $\mathbf{f}_{ij}$ is. We then sort all the scores $\{s_{ij}\}$ as

$$[s_l, | l = 1, \dots, L], \aleph = \text{Sort}(\{s_{ij}\}) \tag{8}$$

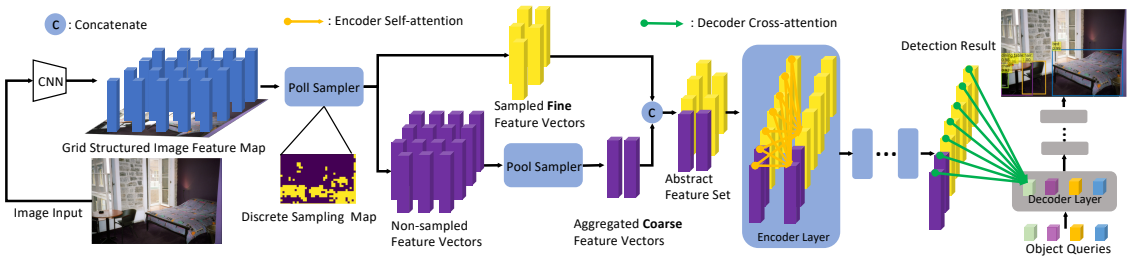

Figure 2. Illustration of the proposed PnP-DETR. The grid structured image feature map is first discretely sampled to obtain the fine feature vector set by a poll sampler, and the remaining non-sampled feature vectors are then aggregated into a small number of coarse feature vectors that summarize the contextual background information. The transformer encoder and decoder then operate on the fine-coarse feature space to model the information interaction and obtain the detection result.

where $\aleph$ is the sorting order and $L = HW$. With $\aleph$, we then take the top N scoring vectors to form the fine feature set:

$$\mathbb{F}_f = [\mathbf{f}_l, |l = 1, \ldots, N] \tag{9}$$

To enable the learning of ScoringNet with back-propagation, we take the predicted informativeness score as a modulating factor to the sampled fine feature set:

$$\mathbb{F}_f = [\mathbf{f}_l * s_l, |l = 1, \ldots, N] \tag{10}$$

We find that normalizing the feature vectors before modulating can stabilize the learning of ScoringNet:

$$\mathbb{F}_f = [LayerNorm(\mathbf{f}_l) * s_l, |l = 1, \ldots, N] \tag{11}$$

We use layer normalization [1] and turn off the affine parameters. Ideally, N may vary with the image content, but we observe that fixed amount sampling already generates good performance, *i.e.*, $N = \alpha L$ where α is a constant fractional value, which we name as the poll ratio. This design also enables an extension to single model computation and performance trade-off discussed in Sec. 3.4.

Pool Sampler The above poll sampler extracts the fine feature set. The remaining feature vectors mainly correspond to the background area. To compress them into a small feature set that summarizes the contextual information, we design a pool sampler that performs a weighted pooling of the remaining feature vectors to obtain a fixed number of M background contextural feature vectors. This is partially inspired by the bilinear pooling [19] and double attention [4] operation where global descriptors are generated for capturing the second-order statistics of the feature map. Formally, the remaining feature vector set is

$$\mathbb{F}_r = \mathbb{F} \setminus \mathbb{F}_f = \{\mathbf{f}_r, |r = 1, \ldots, L - N\} \tag{12}$$

We project the feature vectors with a learnable weight $\mathbf{W}^a \in \mathbb{R}^{C \times M}$ to obtain the aggregation weight $\mathbf{a}_r \in \mathbb{R}^M$:

$$\mathbf{a}_r = \mathbf{f}_r \mathbf{W}^a \tag{13}$$

and project the feature vectors with a learnable weight $\mathbf{W}^v \in \mathbb{R}^{C \times C}$ to obtain the projected feature:

$$\mathbf{f}_r^{'} = \mathbf{f}_r \mathbf{W}^v \tag{14}$$

We then normalize the aggregation weight over all the remaining non-sampled locations with softmax:

$$a_{rm} = \frac{e^{a_{rm}}}{\sum_{r'=1}^{N-L} e^{a_{r'm}}} \tag{15}$$

With the normalized aggregation weight, the projected feature vectors are aggregated to obtain a new feature vector that summarizes the information of non-sampled locations:

$$\mathbf{f}_m = \sum_{r=1}^{L-N} \mathbf{f}_r^{'} * a_{rm} \tag{16}$$

By aggregating with all M aggregation weights, we obtain the summarized coarse background contextual feature set:

$$\mathbb{F}_c = \{\mathbf{f}_m, |r = 1, \ldots, M\} \tag{17}$$

It has been shown in [32] that the context information is crucial for recognizing the objects and is better aggregated by pyramid features of different scales. Our pool sampler is able to freely obtain context information of different scales, by dynamically generating the aggregation weights. That is, some feature vectors may capture local context while others may encode global context. We empirically show such an ability of the pool sampler by visualizing the aggregation weights. Together with the fine set $\mathbb{F}_f$ from the poll sampler, the desired abstraction set $\mathbb{F}^*$ is obtained. Note that instead of convolution feature map, the PnP module can also be applied after a transformer layer.

Reverse Projection for Dense Prediction Tasks The PnP module reduces the image feature map from 2D coordinate space to an abstracted space, which cannot be used for dense prediction tasks like image segmentation. To address

the limitation, we propose to project the encoder output feature vectors back to the 2D coordinate space. Specifically, the fine feature vectors are scattered back to the sampled locations; the coarse feature vectors are first diffused back to original 2D space with the aggregation weight:

$$\hat{\mathbf{f}}_r = \sum_{m=1}^{M} \hat{\mathbf{f}_m} * a_{rm} \tag{18}$$

and then scattered back the non-sampled locations of the poll sampler. $\hat{\mathbf{f}_m}$ denotes output coarse feature vector from the encoder and $\hat{\mathbf{f}}_r$ means the projected feature vector. The obtained 2D feature map is then used for dense prediction.

3.4. PnP-augmented Models

The PnP module is general and straightforward. It can be plugged into existing models to enable them to operate on the fine-coarse feature space for better efficiency. We here describe the models we build to evaluate the PnP module and our proposed random poll ratio scheme to enable instant computation and performance trade-off with a single model.

PnP-DETR and PnP-ViT Recently [7] introduced a transformer-based image recognition model named Vision Transformer (ViT). We evaluate the generalizability of our method on the ViT model. We build the PnP-DETR and PnP-ViT by plugging the PnP module before the transformer network. The resulting models are end-to-end learned and other settings are the same with original models. We use the hybrid ViT architecture [7]. Unlike original DETR and ViT wherein the transformer directly operates over the full image feature space, the PnP augmented transformer models the information interaction on the fine-coarse feature space and adaptively allocates its computation in the spatial domain to achieve better efficiency.

Instant Computation and Performance Trade-off To achieve different computation and performance trade-offs, existing methods improving transformer efficiency generally train multiple models with different complexities controlling hyperparameters, *e.g.*, number of hashes in Reformer [16] and projected feature dimension in Linformer [29]. Unlike them, a model equipped with a PnP module can achieve instant single model computation and performance trade-off. This is enabled by controlling the poll ratio α to determine the amount of fine information preserved. With a larger α, more fine feature vectors are obtained, and the overall performance is expected to be higher; with a smaller α, the performance may be lower but more computation is saved. However, we find inference with a different α to training severely degrades the performance. We propose to generate a random poll ratio during training:

$$\alpha = uniform(\alpha_{low}, \alpha_{high}) \tag{19}$$

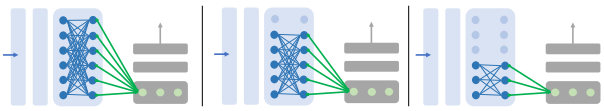

Figure 3. Instant computation-performance trade-off, by executing at different length. Blue: encoder layers Gray: decoder layers.

Where α_{low} and α_{high} defines the value range. α is updated in each iteration. In this way, the transformer learns to work with variable length of input feature vectors, and thus achieves the desired single model computation and performance trade-off by inferring with different poll sample ratios (Fig. 3). The model only needs to be trained once.

4. Experiments

4.1. Implementation Details

For training PnP-DETR, we use 4 images per GPU on 8-GPU machine, with a total batch size of 32. For training PnP-ViT, we use 32 images per GPU, with a total batch size of 256. The meta-scoring network is instantiated with a 2-layer MLP. Unless otherwise stated, the pool sample number M is set to 60 and 240 for R50 and R50-DC5 models, respectively. Other settings including hyper-parameters, network architecture and loss functions follow the baselines for fair comparison. Due to space limit, we defer more details like position embeddings to supplementary.

4.2. Experiments on Object Detection

Fixed Poll Ratio Training Tab. 1 shows the results of the fixed poll ratio training on the COCO benchmark. For the DETR-R50 model, with an α = 0.33, PnP-DETR achieves 41.1 AP and 60% reduction of transformer computation cost. Further increasing α to 0.5, the performance reaches a similar level as the DETR baseline (AP of 41.8 vs. 42.0), with 45% reduction of the computation. For DETR-R50-DC5 model, a similar trend is observed but more computation is saved. We also evaluate the setting of mismatched training and test poll ratio. The model trained with α = 0.33 gets nearly 5 AP drop when evaluating with α = 0.5. This observation shows the necessity of applying random poll ratio training for the model to work with variable poll ratio. We also compare to the deformable DETR [35], as we did not incorporate multi-scale features, which is not the focus of this work, we compare to single scale deformable DETR for fair comparison. Our method performs better than deformable DETR with less FLOPs, especially for large objects, *e.g.*, AP_l of 60.0 vs. 57.8 for the ResNet-50 backbone.

Dynamic Poll Ratio Training As shown in Fig. 4, by training with the random poll ratio with a value range of $(0.15, 0.8)$, the obtained model can achieve dynamic computation and performance trade-off by evaluating with variable poll ratio. The AP for certain poll ratio is similar to

Model	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l	F-encoder	F-decoder	F-sampler	F-total
DETR-R50 [3]	42.0	62.4	44.2	20.5	45.8	61.1	9.6G	1.9G	-	11.5G
Deformable-DETR [35]	40.4	60.5	43.4	21.3	44.6	57.8	-	-	-	5.5G (-52%)
PnP-DETR-R50-α-0.33	41.1	61.5	43.7	20.8	44.6	60.0	3.2G	1.3G	0.1G	4.6G (**-60%**)
Inference-α-0.5	36.1	59.8	36.1	13.9	38.7	57.7	-	-	-	-
PnP-DETR-R50-α-0.5	41.8	62.1	44.4	21.2	45.3	60.8	4.8G	1.5G	0.1G	6.4G (-45%)
DETR-R50-DC5 [3]	43.3	63.1	45.9	22.5	47.3	61.1	69.2G	4.8G	-	74.0G
ACT+MTKD(L=32) [33]	43.1	-	-	22.2	47.1	61.4	-	-	-	58.2 (-21%)
ACT+MTKD(L=24) [33]	42.3	-	-	21.3	46.4	61.0	-	-	-	53.1 (-28%)
Deformable-DETR-DC5 [35]	42.1	62.3	45.6	24.3	45.6	57.3	-	-	-	26.4G (-64%)
PnP-DETR-R50-DC5-α-0.33	42.7	62.8	45.1	22.4	46.2	60.0	17.8G	2.5G	0.4G	20.7G (**-72%**)
PnP-DETR-R50-DC5-α-0.5	43.1	63.4	45.3	22.7	46.5	61.1	29.1G	3.1G	0.7G	32.9G (-56%)

Table 1. Results with fixed poll ratio training on COCO `val` set. F-encoder, F-decoder, F-sampler, F-total denote the FLOPs of the encoder, decoder, PnP sampler and the full transformer, respectively. The FLOPs is obtained by averaging over the first 100 images of `val` set. The backbone FLOPs is omitted as we focus on the transformer efficiency. Inference-α-0.5 means inference with a mismatched poll ratio of 0.5 for PnP-DETR-R50-α-0.33 model. *Note we report single scale deformable DETR [35] with 500 epochs training for fair comparison*, the result is obtained with the official implementation. Refer to Sec. 2 for the relation between our method and deformable DETR.

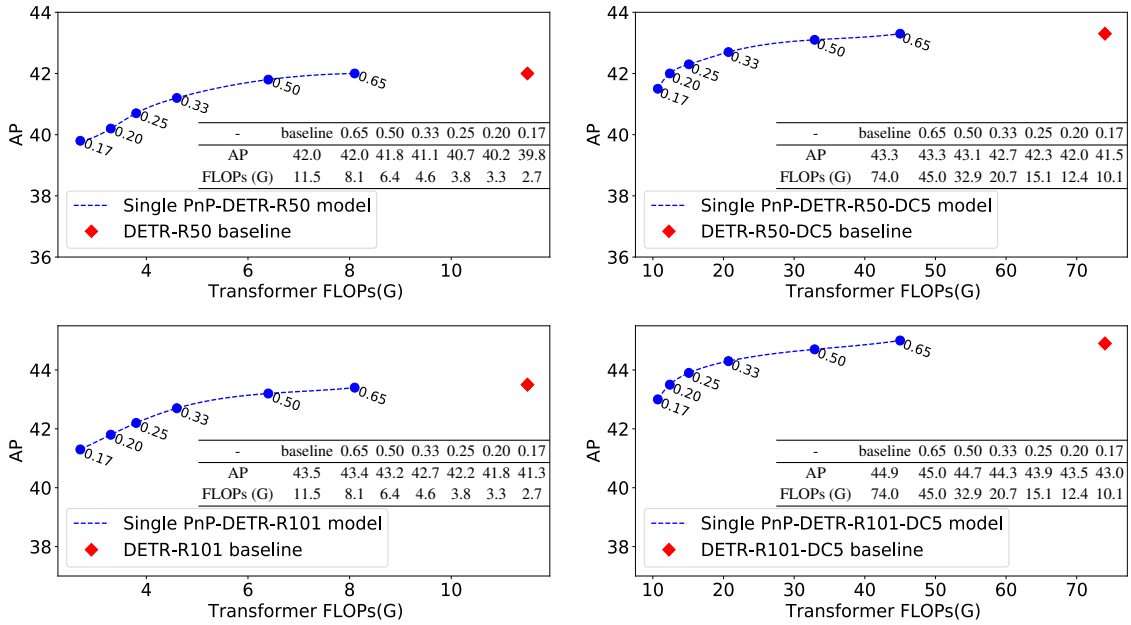

Figure 4. Dynamic AP and FLOPs trade-off curve with single model trained with our method. The curve is obtained by evaluating with different poll ratios (α) as denoted on the curve. The chosen α values roughly equals the fractions of $\frac{1}{6}$, $\frac{1}{5}$, $\frac{1}{4}$, $\frac{1}{3}$, $\frac{1}{2}$ and $\frac{1}{1.5}$.

the fixed poll ratio trained counterpart. For example, a PnP-DETR-R50 model gets 41.1 AP with fixed poll ratio 0.33 training and 41.2 AP with random poll ratio training. The performance is the same to the baseline with a poll ratio of 0.65. We observe when the poll ratio is large, *e.g.*, 0.5, increasing the poll ratio brings diminished gain in AP. This is likely because the fine feature set already covers the essential spatial locations for detecting the objects, and thus more fine information only brings limited gain. Similar observations are made with the ResNet-101 backbone. Tab. 2 shows the inference time compared to baseline model, the inference time is significantly reduced.

Visualization of Computation Density Map Fig. 5 shows some example detection results and associated computation density maps, with poll ratio of 0.33. The objects are well detected while the computation is dynamically allocated to the spatial domain in a content-aware manner. To compute the density map, we assign a weight to each spatial location. For poll sampled locations, the weight is 1. For each of other locations, the weight is the cumulative value of all pool sample aggregation weights at this location. Then the transformer cost is distributed with the normalized weights to obtain the computation density map.

Methods	Encoder	Decoder	PnP-sampler
DETR (baseline)	72.4	11.1	-
PnP-DETR-α-0.5	**28.4**	10.5	2.1
PnP-DETR-α-0.33	**17.4**	10.3	2.0

Table 2. Inference time (ms) measured on TITAN RTX GPU, with ResNet-50-DC5 backbone.

4.3. Experiments on Other Tasks

Panoptic Segmentation Following [3], we evaluate our method on the panoptic segmentation task. To perform dense per-pixel segmentation as DETR, we project the encoder output feature back to the original 2D coordinate space. As shown in Tab. 3, the model saves computation and achieves instant performance and computation trade-off by varying the poll ratio α, *e.g.*, achieving Panoptic Quality (PQ) of 43.2 compared to 43.4 of a baseline DETR model, with 5G less FLOPs (*i.e.*, 6.6G vs. 11.6G).

Image Recognition We also apply the PnP sampling to the recent transformer-based image classification model of ViT [7]. We use the hybrid architecture with ResNet50-stage4 feature map (14x14) and train the model on the ImageNet-1k dataset from scratch. We set the pool sample number to 10. We train the PnP-ViT with random poll ratio in the value range of $[0.2, 0.8]$. As shown in Tab. 4, the PnP-ViT achieves dynamic computation and performance trade-off as observed with the DETR model. The results show the generalizability of PnP sampling design.

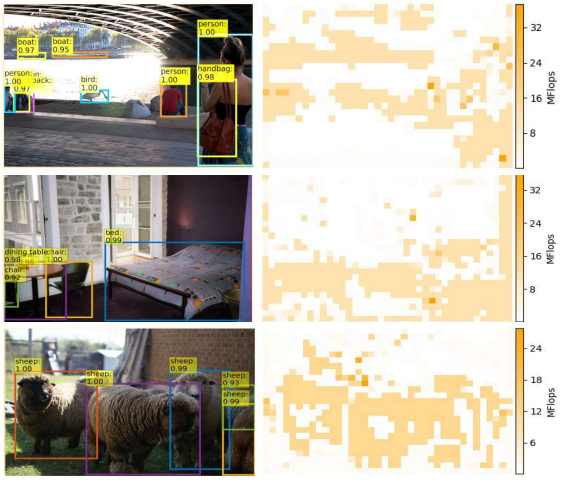

Figure 5. Example detection results and computation density maps with PnP-DETR-R50 model at poll ratio 0.33.

4.4. Model Analysis

We then provide several experimental analysis to better understand the proposed method. To save experiment time, we sample the COCO benchmark to obtain a smaller dataset and conduct all experiments on the **sampled COCO dataset**. We design a class-incremental sampling that helps preserve the data distribution. Due to space limit, we defer sampling details and more experiments to supplementary.

-	DETR	α-0.65	α-0.5	α-0.33	α-0.25	α-0.2
PQ	43.4	43.5	43.2	42.8	42.4	41.8
SQ	79.3	79.2	79.1	78.9	78.7	78.4
RQ	53.8	53.8	53.4	53.0	52.4	51.7
FLOPs (G)	11.6	8.3	6.6	4.8	4.0	3.5

Table 3. Results on panoptic segmentation. ResNet-50 backbone is used. α-* means inferring with a variable poll ratio.

-	ViT	α-0.7	α-0.5	α-0.33	α-0.25	α-0.2
Top1-Acc	82.2	82.1	81.9	81.6	81.4	81.2
FLOPs (G)	10.0	7.3	5.5	3.9	3.2	2.8

Table 4. Results on ViT model with hybrid architecture based on ResNet-50. α-* means a single PnP-ViT model with a variable poll ratio for inference.

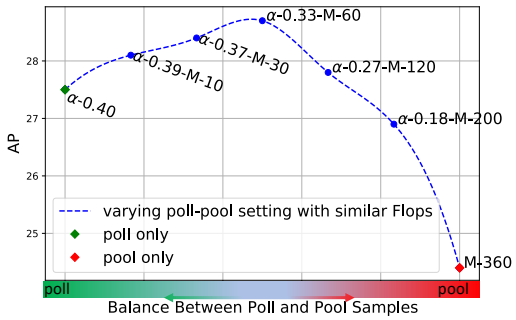

Figure 6. Varying the poll ratio (α) and pool sample number (M) with the same amount of computation, with ResNet-50 backbone.

The Balance Between Poll and Pool Samplers As shown in Fig. 6, we vary the the poll sample ratio and the pool sample number to obtain the performance curve with the same amount of computation cost. We observe that **1)** with only poll sampling (α-0.4), the performance is suboptimal; incorporating pool feature vector samples can significantly improve AP with the complementary background information from non-sampled locations, *e.g.*, α-0.39-M-10 model achieving about 0.7 AP higher than the α-0.4 model. **2)** with only pool sampling, the performance drops by a large margin. We assume it is difficult for the pool sampling to preserve accurate fine information, as it is designed to aggregate feature vectors spatially from different locations. **3)** the optimal setting is $1/3$ poll ratio with 60 pool samples, indicating that a compact feature set should be mainly composed of fine feature vectors for accurate object detection. We further individually examine the effects of pool sample number M and poll sample ratio α: **1)** We vary M by fixing α. **2)** We vary α by fixing M. Due to space limit, we defer the experiment results and analysis to the supplementary.

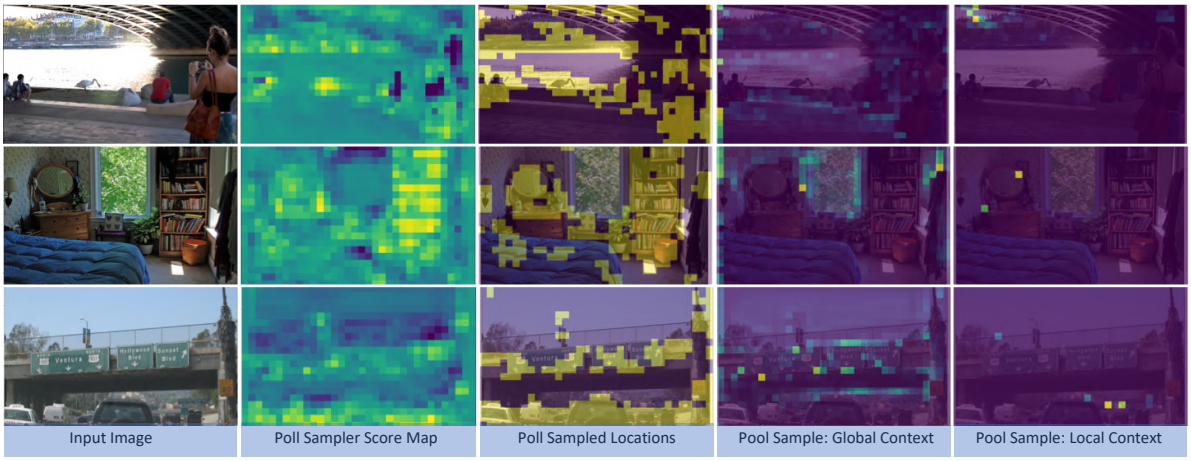

Figure 7. Visualization of poll sample locations and example aggregation weight map from the pool sampler, with PnP-DETR-R50. 1st col: input images; 2nd/3rd cols: score maps of poll sampler and its sample maps correspondingly; last two columns: the example aggregation weight maps from the pool sampler, in which the former aggregates global context, while the latter aggregates local context.

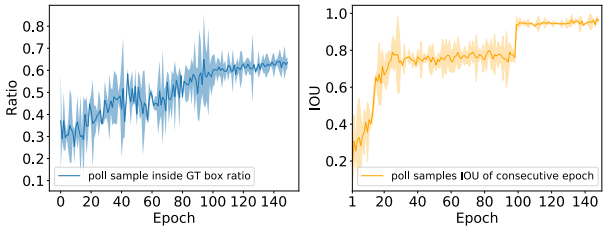

Figure 8. The learning dynamics of the poll sampler, with PnP-DETR-R50. The model is trained for 150 epochs with learning rate decay at 100 epochs. The left figure shows the proportion of sampled locations that lie within the GT bounding box areas. The right figure depicts the pixel IOU of sampled locations with previous epoch. The statistics are obtained on the `val` set.

Visualizing Poll and Pool Sampling As shown in Fig. 7, we visualize the poll sampler's scoring map, its sampled locations, and example aggregation weight map of the pool sampler. To summarize, **1)** the poll sampler learns to sample the locations within and surrounding objects; **2)** the pool sampler obtains different scales of context. For example, on the first row, the first pool sample attends to a wide range of spatial locations and encodes global context information; the second sample attends to a small area around the sky, and thus captures local context. We also have some other intriguing observations on the poll sampler: **1)** It learns to sample object alike area beyond the object categories used for training. For example, for the last row in Fig. 7, locations around the traffic signs and the tree-like object are sampled. The behavior is similar to a learned region proposal network (**RPN**) [23], but learned without explicit supervision. **2)** It tends to sample coarsely for some large and 'easy' objects but finely for small ones. For example, fewer points are sampled for the woman in the first row and the bed in the second row; the books in the second image and the cars in the last image are smaller and more difficult to detect, so the poll sampler finely samples feature vectors for those objects and surrounding areas.

Tracking Poll Sampler Learning To better understand the learning process and dynamics of the poll sampler, we record two statistics during training: **(1)** the proportion of sampled locations that are within the GT bounding boxes; **(2)** the pixel IOU of the sampled locations between consecutive epochs. As shown in Fig. 8, we make following observations. **1)** The poll sampler gradually learns to sample more feature vectors that lie within the ground truth area but finally remains steady at about 60%, indicating that it also attends some background and contextual locations that are crucial for recognizing and detecting the objects. **2)** The poll sampler initially has a large variation on its sampled locations, and thus the sampled areas of consecutive epochs have small IOU (*i.e.*, about 0.2). During training, the IOU quickly converges to about 0.7 with around 30 epochs and remains steady at about 0.75, indicating that the sampler quickly learns to sample crucial feature vectors and the sampled locations does not change much. After learning rate decay at 100 epoch, the IOU of the consecutive epoch is close to 1.0, meaning the poll sampler converges.

5. Conclusion

In this paper, we encapsulate the idea of reducing spatial redundancy into a learnable PnP module. It is composed of a ranking based poll sampler that discretely samples fine feature information and a subsequent adaptive pool sampler that summarizes the background contextual information. The PnP module is general and can be incorporated into existing model for efficient processing while maintaining the performance, which is verified on object detection, panoptic segmentation and image recognition. We believe the proposed method offers insights for future research into efficient visiual analysis with transformers.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 4

[2] Shijie Cao, Lingxiao Ma, Wencong Xiao, Chen Zhang, Yunxin Liu, Lintao Zhang, Lanshun Nie, and Zhi Yang. Seernet: Predicting convolutional neural network feature-map sparsity through low-bit quantization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11216–11225, 2019. 3

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *arXiv preprint arXiv:2005.12872*, 2020. 1, 2, 3, 6, 7

[4] Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. Aˆ 2-nets: Double attention networks. In *Advances in neural information processing systems*, pages 352–361, 2018. 4

[5] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020. 1, 2

[6] Xuanyi Dong, Junshi Huang, Yi Yang, and Shuicheng Yan. More is less: A more complicated network with less inference complexity. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5840–5848, 2017. 3

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1, 2, 5, 7

[8] Oren Dovrat, Itai Lang, and Shai Avidan. Learning to sample. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2760–2769, 2019. 3

[9] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6569–6578, 2019. 1, 3

[10] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2009. 2

[11] Michael Figurnov, Maxwell D Collins, Yukun Zhu, Li Zhang, Jonathan Huang, Dmitry Vetrov, and Ruslan Salakhutdinov. Spatially adaptive computation time for residual networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1039–1048, 2017. 3

[12] Mikhail Figurnov, Aizhan Ibraimova, Dmitry P Vetrov, and Pushmeet Kohli. Perforatedcnns: Acceleration through elimination of redundant convolutions. In *Advances in Neural Information Processing Systems*, pages 947–955, 2016. 3

[13] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 2

[14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014. 2

[15] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. *arXiv preprint arXiv:2006.16236*, 2020. 1, 2

[16] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020. 1, 2, 5

[17] Itai Lang, Asaf Manor, and Shai Avidan. Samplenet: Differentiable point cloud sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7578–7588, 2020. 3

[18] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 1, 2

[19] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear cnn models for fine-grained visual recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 1449–1457, 2015. 4

[20] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016. 2

[21] Ehsan Nezhadarya, Ehsan Taghavi, Ryan Razani, Bingbing Liu, and Jun Luo. Adaptive hierarchical down-sampling for point cloud classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12956–12964, 2020. 3

[22] Mengye Ren, Andrei Pokrovsky, Bin Yang, and Raquel Urtasun. Sbnet: Sparse blocks network for fast inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8711–8720, 2018. 3

[23] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 1, 2, 3, 8

[24] Ivan Skorokhodov. Cuda implementation of non-uniform interpolation. https://github.com/universome/non-uniform-interpolation, 2020. 2

[25] Ivan Skorokhodov. Interpolating points on a non-uniform grid using a mixture of gaussians. *arXiv preprint arXiv:2012.13257*, 2020. 2

[26] K-K Sung and Tomaso Poggio. Example-based learning for view-based human face detection. *IEEE Transactions on pattern analysis and machine intelligence*, 20(1):39–51, 1998. 2

[27] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 9627–9636, 2019. 1, 2

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 1, 2, 3

[29] Sinong Wang, Belinda Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020. 1, 2, 5

[30] Zhenda Xie, Zheng Zhang, Xizhou Zhu, Gao Huang, and Stephen Lin. Spatially adaptive inference with stochastic feature sampling and interpolation. *arXiv preprint arXiv:2003.08866*, 2020. 3

[31] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zihang Jiang, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *arXiv preprint arXiv:2101.11986*, 2021.

[32] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017. 4

[33] Minghang Zheng, Peng Gao, Xiaogang Wang, Hongsheng Li, and Hao Dong. End-to-end object detection with adaptive clustering transformer. *arXiv preprint arXiv:2011.09315*. 6

[34] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 1, 3

[35] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 2, 5, 6