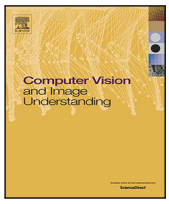
Computer Vision and Image Understanding 248 (2024) 104106

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu



DHS-DETR: Efficient DETRs with dynamic head switching

Hang Chen [a,b,c], Chufeng Tang [a,b,c], Xiaolin Hu [a,b,c,d,*]

[a] Department of Computer Science and Technology, THU-Bosch JCML Center, Tsinghua University, Beijing, 100084, China
[b] Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing, 100084, China
[c] Institute for Artificial Intelligence, Tsinghua University, Beijing, 100084, China
[d] Chinese Institute for Brain Research (CIBR), Beijing, 100010, China

ARTICLE INFO

Communicated by Nicu Sebe

Keywords:
Object detection
Dynamic neural networks
Knowledge distillation

ABSTRACT

Detection Transformer (DETR) and its variants have emerged a new paradigm to object detection, but their high computational cost hinders practical applications. By investigating their essential components, we found that the transformer-based head usually occupies a significant amount of computation. Through further comparing heavy and light transformer heads, we observed that both heads produced satisfactory results for easy images while showing a noticeable difference for hard images. Inspired by these findings, we propose a dynamic head switching (DHS) strategy to dynamically select the proper head for each image at inference for a better balance of efficiency and accuracy. Specifically, our DETR model incorporates multiple heads with different computational complexity and a lightweight module which selects proper heads for given images. This module is optimized to maximize detection accuracy while adhering to the overall computational budget limitations. To minimize the potential accuracy drop when executing the lighter heads, we propose online head distillation (OHD) to improve the accuracy of the lighter heads with the help of the heavier head. Extensive experiments on the MS COCO dataset validated the effectiveness of the proposed method, which demonstrated a better accuracy–efficiency trade-off compared to the baseline using static heads.

1. Introduction

Existing object detection methods could be roughly classified into two basic categories: two-stage methods (Girshick, 2015; Ren et al., 2015; He et al., 2017; Cai and Vasconcelos, 2021) and one-stage methods (Lin et al., 2017; Tian et al., 2019; Zhang et al., 2020; Li et al., 2020). Recently, Detection Transformer (DETR) (Carion et al., 2020) and its variants (Meng et al., 2021; Zhu et al., 2021; Liu et al., 2022; Li et al., 2022b; Chen et al., 2022b) (collectively referred to as DETRs in this paper) have grown into a new paradigm for object detection. DETRs utilize transformer (Vaswani et al., 2017) as the detection head to directly predict the set of objects through the interaction between object queries and visual features, free from multiple hand-designed components such as anchors and Non-Maximum Suppression (NMS). However, existing DETRs usually suffer from high computational overhead, which limits their practical applications. By investigating the essential components of DETRs, we found that the transformer-based head usually occupies a significant amount of computation (*e.g.*, comparable to a ResNet-50 backbone network). This observation inspired us to explore methods to reduce the computational cost of the heavy transformer head for a better balance between accuracy and efficiency.

Previous works on improving the efficiency of DETRs primarily focused on developing more compact networks (Chang et al., 2022; Chen et al., 2022a; Roh et al., 2022; Wang et al., 2021), while overlooking the inherent differences in difficulty levels among different images. To address this, we conducted an investigation to determine *whether it is necessary to employ a heavy transformer head for each image*. By comparing the predictions of heavy and light transformer heads, we discovered that for easy images (*e.g.*, images with only a few distinct objects), the light head was sufficient in producing good results. As depicted in Fig. 1, the two heads generated similar predictions for the easy image, but exhibited a larger discrepancy for the hard image (*e.g.*, images with more objects and occlusion). This observation motivated us to design a model that dynamically process images of varying difficulties using transformer heads with different computational complexities. Our proposed model assigns easy images to lighter heads, while leaving the hard ones to heavier heads, which is expected to achieve a more favorable balance between accuracy and efficiency.

To realize this idea, we construct a DETR model that incorporates multiple heads with varying computational complexity and propose a *dynamic head switching (DHS)* strategy to select the most proper head for each image during inference. The lighter heads are constructed

* Corresponding author at: Department of Computer Science and Technology, THU-Bosch JCML Center, Tsinghua University, Beijing, 100084, China.
E-mail address: xlhu@mail.tsinghua.edu.cn (X. Hu).

https://doi.org/10.1016/j.cviu.2024.104106
Received 19 January 2024; Received in revised form 29 July 2024; Accepted 1 August 2024
Available online 6 August 2024

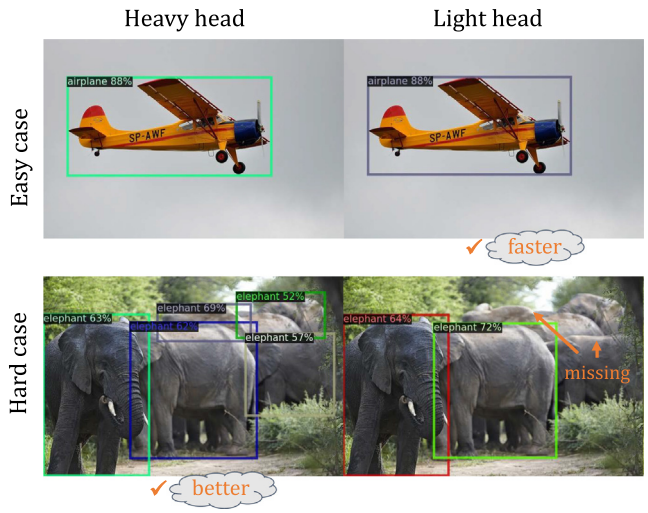


Fig. 1. Illustration of our motivation. For easy images (1st row), the light transformer head is sufficient to yield good predictions and is faster. For hard images (2nd row), the heavy transformer head predicts much better than the light head. Assigning images of different difficulties to different transformer heads might allow for a better accuracy–efficiency trade-off.

via reducing the number of transformer layers in the original DETR head. These lighter heads are then added in parallel to the original head in the backbone network, and jointly trained using ground-truth annotations (*i.e.*, all these heads are trained on each training image). To enable the dynamic switching between heads, we incorporate a lightweight module, such as a 3-layer MLP, into the backbone. This module is finetuned after the training of the DETR model with multiple heads. During finetuning, the module is optimized using the Gumbel-Softmax technique (Jang et al., 2017) for end-to-end optimization. During inference, the most appropriate head is selected for each image based on its prediction. By adjusting the budget restriction of the dynamic switching module, we are able to control the overall average computational complexity. This offers an additional advantage of flexibility, enabling our model to work with different computational budgets without requiring re-training of the entire network.

To address the potential decrease in accuracy when executing the lighter heads, it is important to carefully consider the design and training of suitable lighter heads. There are two main aspects to consider: (1) Since the transformer head typically consists of an encoder and a decoder, we search for good recipes (*i.e.*, number of encoder and decoder layers, respectively) that improve efficiency while maintaining high accuracy. (2) To further benefit from the joint training of the multiple heads, we propose *online head distillation (OHD)*. This approach involves guiding the learning of the lighter heads during training by leveraging the knowledge from the original heavy head. OHD is applied to both the encoder and decoder of the transformer head.

To summarize, our method enables the detector to process images of different difficulty using heads with different computational complexity, shifting the focus from solely pursuing accuracy to achieving a balance between accuracy and efficiency that better aligns with real-world scenarios. Extensive experiments conducted on the MS COCO dataset validated the effectiveness of the proposed method, which achieved competitive results while offering remarkable flexibility for scenarios with different computational budgets. Our contributions are as follows:

- We analyzed the computational overhead and predictions of the DETR head and proposed a dynamic head switching (DHS) strategy for a better balance between accuracy and efficiency during inference.
- We proposed online head distillation (OHD) which improves the accuracy of lighter heads by leveraging the knowledge from heavier heads during training.
- Extensive experiments on the MS COCO dataset demonstrated that the proposed methods achieved competitive results with promising flexibility under various computational budgets.

2. Related work

Object Detection. Existing object detection methods can be roughly divided into two-stage (Girshick, 2015; Ren et al., 2015; He et al., 2017; Cai and Vasconcelos, 2021) and one-stage methods (Lin et al., 2017; Tian et al., 2019; Zhang et al., 2020; Li et al., 2020). Two-stage methods typically generate dense proposals via RPN, and then extract RoI features to refine the prediction results. One-stage methods directly predict the bounding boxes and classification probabilities with dense anchor boxes (Lin et al., 2017) or anchor points (Tian et al., 2019; Li et al., 2020; Tian et al., 2019). Duplicate predictions are removed through NMS (Girshick et al., 2014) in these methods. Recently proposed Detection Transformer (DETR) (Carion et al., 2020) and its variants (Meng et al., 2021; Zhu et al., 2021; Liu et al., 2022; Li et al., 2022b) are gradually becoming a new paradigm for object detection. DETR regards object detection as a set prediction problem, directly predicting bounding boxes and classes of objects. Being free of redundant designs such as anchor and NMS makes DETR much more concise. However, the existing DETR-like methods still suffer from the heavy backbone and transformer heads. In this work, we focus on a better trade-off of the accuracy and efficiency of transformer head in DETRs with dynamic head switching and online head distillation. Our method applies to any DETR variants.

Knowledge Distillation. Knowledge distillation (Hinton et al., 2015) was first proposed for building compact classification models. By mimicking soft labels, learned knowledge can be transferred from the heavier teacher network to the lightweight student network (Liang et al., 2023; Wang et al., 2023). In recent years, several methods (Li et al., 2017; Chen et al., 2017; Wang et al., 2019; Zheng et al., 2022; Yang et al., 2022a; Nguyen et al., 2022; Li et al., 2022a) have been proposed to extend knowledge distillation to object detection. Early works (Li et al., 2017; Dai et al., 2021; Chen et al., 2017) focus on two-stage detectors, while other works (Wang et al., 2019; Zheng et al., 2022; Du et al., 2021; Yang et al., 2022a; Li et al., 2022a) are concerned more with one-stage detectors (*i.e.*, dense object detectors). However, directly applying these methods to DETRs might be sub-optimal due to their specialized designs. A challenge is how to align the outputs of teacher and student, as DETRs directly predict objects with a transformer head, rather than based on fixed anchor points or anchor boxes. Several more recent works (Chang et al., 2022; Chen et al., 2022a; Wang et al., 2022) have proposed distillation strategies dedicated to DETRs, *e.g.*, feeding the teacher's object queries into the student network to align their outputs. These works focus on compressing the backbone networks and require well-trained teacher networks in advance. Instead, we employ online knowledge distillation to improve the accuracy of the light head to incorporate dynamic head switching to realize a better accuracy–efficiency trade-off.

Dynamic Neural Networks. Dynamic neural networks include dynamic parameter and dynamic architecture methods, which enable advantages in terms of accuracy, efficiency, and adaptiveness by adjusting their parameters and structures for different inputs, respectively (Han et al., 2022). In this work, we focus on dynamic architecture approaches. One set of these methods reduces the computation on simple inputs by early exiting or skipping (Wang et al., 2018; Huang et al., 2018; Bolukbasi et al., 2017; Liu et al., 2017). Another line of work adjusts the computational graph for different inputs by dynamic routing (Liu and Deng, 2018; Hehn et al., 2020; Tanno et al., 2019). However, these methods are limited to classification tasks. PnP

DETR (Wang et al., 2021) and Sparse DETR (Roh et al., 2022) improve the computational efficiency of DETR's encoder by dynamically skipping background tokens. But these methods still employ detection heads with same computational complexity for images of varying difficulty. Complementary to them, we propose a more general method that adaptively switches the head for images of different difficulties.

Efficient Detection Transformers. In addition to the dynamic neural network method discussed earlier, recent studies have focused on enhancing the efficiency of DETR through various approaches (Li et al., 2023; Zhao et al., 2024; Yao et al., 2021). Lite DETR (Li et al., 2023) improves efficiency by decoupling the multi-scale self-attention within the encoder. RT-DETR (Zhao et al., 2024), on the other hand, leverages a hybrid encoder and high-quality initial queries to reduce computational cost while preserving accuracy. Efficient DETR (Yao et al., 2021) utilizes a dense detection head to predict the initial object query, thereby lessening reliance on the number of decoder layers. Our proposed method, in comparison to these approaches, is more versatile and can be applied to a range of DETR variations, including the ones mentioned above.

3. Methods

We first provide a succinct review of DETR, followed by an analysis of the computational cost associated with its essential components. This analysis motivated us to consider how to reduce the computational cost of the transformer head. We subsequently present an overview of our proposed method and provide a detailed description of each module.

3.1. Preliminaries

The architecture of DETR (Carion et al., 2020) consists of three components: a backbone network, a transformer encoder, and a transformer decoder. Given an input image $I \in \mathbb{R}^{3\times H_0\times W_0}$, the backbone network (*e.g.*, ResNet 50) is used to generate the feature map $F_b \in \mathbb{R}^{C\times H\times W}$, which is further processed by the encoder containing multiple transformer layers (usually 6 layers) to generate an enhanced feature map $F_e \in \mathbb{R}^{D\times H\times W}$. F_e is then fed into the decoder (typically with 6 transformer layers) to perform cross-attention with N learnable object queries, obtaining the final predictions $\hat{y} = \{\hat{y}_i\}_{i=1}^N$. During training, the optimal bipartite matching (denoted as a permutation σ) between the prediction $\hat{y}$ and the ground-truth y (padded with *no object*) is first calculated:

$$\hat{\sigma} = \arg\min_{\sigma} \sum_{i=1}^{N} \mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)}), \tag{1}$$

where $\mathcal{L}_{\text{match}}$ is a matching cost function. The detection loss $\mathcal{L}_{\text{detr}}$ is then computed over pairs of the matched prediction and the ground-truth.

Transformer Decoder. Given a set of object queries $Q = \{Q_i\}_{i=1}^N \in \mathbb{R}^{N\times D}$, the decoder aggregates the i−th query's feature $Z_i \in \mathbb{R}^D$ from the encoder feature map with cross-attention[1]:

$$\begin{aligned} Z_i^{(m)} &= \text{Softmax}(Q_i^{(m)} K_i^{(m)\top}/\sqrt{D}) V_i^{(m)\top} \\ &= A_i^{(m)} V_i^{(m)\top}, \end{aligned} \tag{2}$$

where m indicates the index of attention head (in multi-head attention) and $Q_i^{(m)}$, $K_i^{(m)}$, and $V_i^{(m)}$ are obtained by linear projections of Q_i, F_e and F_e respectively. This operation can be interpreted as weighting image features from the encoder with attention maps $A_i^{(m)} \in (0,1)^{H\times W}$. After that, the object features Z_i is obtained by concatenating features from different heads and applying a linear transformation matrix W^o:

$$Z_i = \text{Concat}(Z_i^{(1)}, \dots, Z_i^{(M)}) W^{o\top}, \tag{3}$$

where M stands for the total number of heads. Finally, two MLPs are applied to Z_i for predicting object class and bounding box, respectively.

[1] We write it without flattening for the convenience of later derivation.

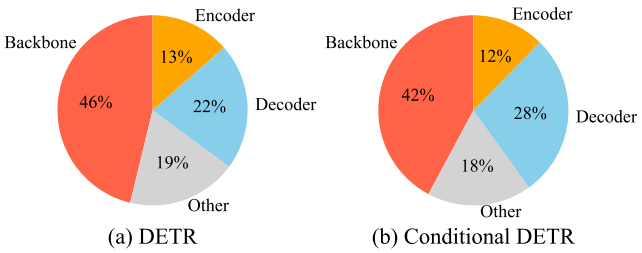


Fig. 2. Inference time of each essential component of DETRs. 'Other' indicates the time cost of pre-processing and post-processing. ResNet-50 backbone was used here.

Since the original DETR (Carion et al., 2020) suffer from slow convergence, various works (Meng et al., 2021; Liu et al., 2022; Li et al., 2022b) have been proposed to accelerate it in different ways.

3.2. Analysis on computational cost

To identify the key factors influencing DETR's efficiency, we analyzed the computational cost during inference (measured as inference time) of the essential components by taking DETR and Conditional DETR as example. As shown in Fig. 2, the inference cost of the transformer head (including both the encoder and decoder) is usually as large as the backbone (*i.e.*, ResNet-50). To improve model efficiency, most existing studies focused on designing or learning efficient backbones (Howard et al., 2017; Zhang et al., 2018; Chang et al., 2022; Wang et al., 2022; Chen et al., 2022a). In this work, we explored a complementary way by learning efficient heads for DETRs. This analysis suggests that it is promising to greatly enhance the efficiency of DETRs by learning efficient heads.

3.3. Overview of the proposed method

Fig. 3 illustrates the overview of the proposed method. We attach multiple transformer heads in parallel to the backbone network and train them jointly with the supervision of the ground-truth (*i.e.*, all these heads are trained on each image). The switching module is then fine-tuned to assign different heads to input images with different difficulties. During inference, for a given image, one of the heads is selected according to the prediction of the switching module. To minimize the accuracy drop when executing the light head, we further propose an online head distillation to transfer the knowledge from heavier head to lighter heads during training.

3.4. Head recipes

We first investigate how to construct these light heads to maximize efficiency while preserving accuracy. The lighter heads are built by reducing the number of transformer layers based on the original head. As the head typically includes the encoder and decoder, it is worth exploring how much each should be scaled down. To select suitable head recipes, following Yang et al. (2022b), we propose Accuracy Cost Ratio (ACR) as a quantitative metric. ACR is negatively correlated with the accuracy drop while positively correlated with the efficiency gain, where accuracy and efficiency are measured by mAP and FPS, respectively. Our ACR is defined as

$$\text{ACR} = 0.5 \times (1 - \frac{\Delta\text{AP}}{\text{AP}_\text{o}}) + 0.5 \times \frac{\Delta\text{FPS}}{\text{FPS}_\text{o}}, \tag{4}$$

where AP_o and FPS_o denote the mAP and FPS of the original head, respectively, and ΔAP and ΔFPS represent the decrease in mAP and the increase in FPS after reducing the number of layers, respectively. ACR is normalized into $[0, 1]$. We favor heads with higher ACR, which offer better accuracy–efficiency trade-off.

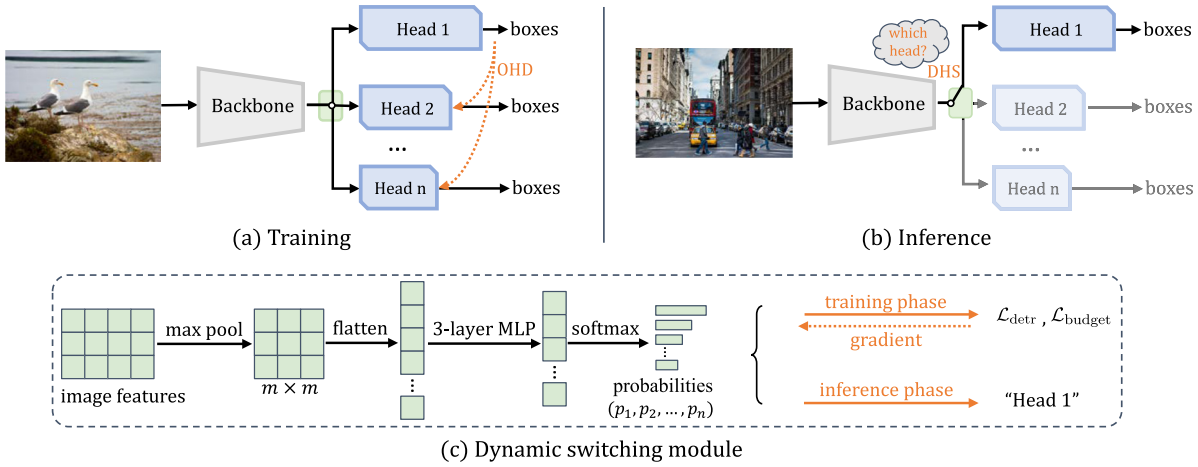


Fig. 3. Overview of the proposed method. (a) During training, we train multiple heads concurrently and distill the capacity of the heavier head to the lighter heads via online head distillation (OHD). (b) During inference, given an image, the dynamic switching module adaptively selects an suitable detection head. (c) Details of the dynamic switching module. The module is lightweight and mainly consists of a 3-layer MLP.

3.5. Dynamic head switching

The dynamic head switching strategy is designed to dynamically assign proper heads to images of different difficulty via a lightweight module (denoted as ϕ_d). The module is implemented as a 3-layer MLP attached to the backbone network (Fig. 3(c)). We first present how to implement head selection and then explain how to optimize this module.

The head selection is determined by an onehot vector $\boldsymbol{z} \in \{0,1\}^n$, where n is the number of heads, such that the ith head is selected if and only if $\boldsymbol{z}_i = 1$. Specifically, given image features F_b, we let the module ϕ_d predict the probabilities of selecting each head $\boldsymbol{p} = \phi_d(F_b) \in (0,1)^n$. During inference, the head with the largest probability (denoted as h_{i^*}) is executed:

$$\hat{y}_{\text{inference}} = h_{i^*}(F_b), \quad i^* = \arg\max_{1 \le i \le n} \boldsymbol{p}_i, \tag{5}$$

where $\hat{y}_{\text{inference}}$ is the output of the object detector.

The module is expected to achieve as high accuracy as possible for a given computational budget (measured by the average FLOPs during inference). We formalize the optimization of the module as a constrained optimization problem, which minimizes the detection loss $\mathcal{L}_{\text{detr}}$ under given computational budget constraint $\mathcal{L}_{\text{budget}}$. We solve for ϕ_d by gradient descent via the following equation:

$$\min_{\phi_d} \ \mathcal{L}_{\text{detr}} + \lambda_{\text{budget}} \cdot \mathcal{L}_{\text{budget}}, \tag{6}$$

where λ_{budget} is a hyperparameter.

The backward propagation of the head selection poses a discrete optimization problem since *argmax* in Eq. (5) is not differentiable. To address this problem, during training, we employ the Gumbel-Softmax technique (Jang et al., 2017), which calculates the onehot vector $\boldsymbol{z}$ as

$$\boldsymbol{z} = \text{onehot}\left(\arg\max_{i \le n}[g_i + \log p_i]\right), \tag{7}$$

where the noise g_i is *i.i.d* sampled from the Gumbel$(0,1)$ distribution. Eq. (7) is differentiated with the Straight-Through Gumbel Estimator (Jang et al., 2017). The output $\hat{y}_{\text{train}}$ is then determined by $\boldsymbol{z}$ for calculating the detection loss $\mathcal{L}_{\text{detr}}$,

$$\hat{y}_{\text{train}} = \sum_{i=1}^{n} \boldsymbol{z}_i \cdot h_i(F_b). \tag{8}$$

To determine the computational budget loss $\mathcal{L}_{\text{budget}}$ as shown in Eq. (6), we first estimate the average computational cost $\bar{C}$ by taking the mean of the FLOPs of the selected transformer heads within the mini-batch:

$$\bar{C} = \text{mean}\left(\sum_{i=1}^{n} \boldsymbol{z}_i \cdot C_i\right), \tag{9}$$

where C_i is the FLOPs of ith head, and the `mean` operation is carried out across the mini-batch. Subsequently, the computational budget loss is evaluated as:

$$\mathcal{L}_{\text{budget}} = \max\{\bar{C}/T - 1, 0\}. \tag{10}$$

When $\bar{C}$ exceeds the target computational budget (denoted as T), the loss is positively correlated with $\bar{C}$; Otherwise, it is 0.

To enable the detector to work well with different budgets, we fine-tune a set of modules ϕ_d using Eq. (6) with different budget T in parallel. Specifically, during fine-tuning, the backbone network and the transformer heads are both frozen and only the dynamic switching modules are updated. The cost of fine-tuning is marginal since ϕ_d is very lightweight.

3.6. Online head distillation

To minimize accuracy drop when executing the light heads and benefit more from joint training of multiple heads, we propose online head distillation to transfer knowledge in the heavier head to the lighter heads during training, including both the encoder distillation and decoder distillation (Fig. 3(a)).

Encoder Distillation. We perform feature distillation for the encoder of the heads. A naive solution is to calculate the mean squared error (MSE) between two encoder feature maps directly. However, this solution might be sub-optimal since many locations on the feature map that are not critical for prediction may dominate the gradient, as discussed in previous studies (Wang et al., 2019; Kang et al., 2021; Du et al., 2021; Li et al., 2022a). To emphasize valuable encoder features, we reuse the attention map corresponding to positive object queries (*i.e.*, the ones matched the ground-truth objects). Intuitively, regions with larger attention weights may have a stronger influence on the final prediction and are thus more important. We therefore derive encoder distillation with MSE loss weighted by multi-head attention maps of the positive object queries:

$$\mathcal{L}_{\text{encoder}} = \lambda_{\text{encoder}} \cdot \bar{A}^h \odot \|F_e^h - F_e^l\|_2, \tag{11}$$

where $F_e^h, F_e^l \in \mathbb{R}^{D \times H \times W}$ denote the encoder feature maps of the heavy head and light head, respectively. λ_{encoder} is the loss weight. The l^2-norm $\|\cdot\|_2$ is calculated on the channel dimension. Operator $\odot$ stands for element-wise product and summation. $\bar{A}^h \in (0,1)^{H \times W}$ represents the average attention map of multi-head attention (*i.e.*, $A_i^{(m)}$ in Section 3.1) of positive object queries in the heavy head.

Decoder Distillation. For decoder distillation, a critical problem is how to align object queries of the teacher and the student (Chen et al., 2022a; Chang et al., 2022; Wang et al., 2022). Similar to Eq. (1),

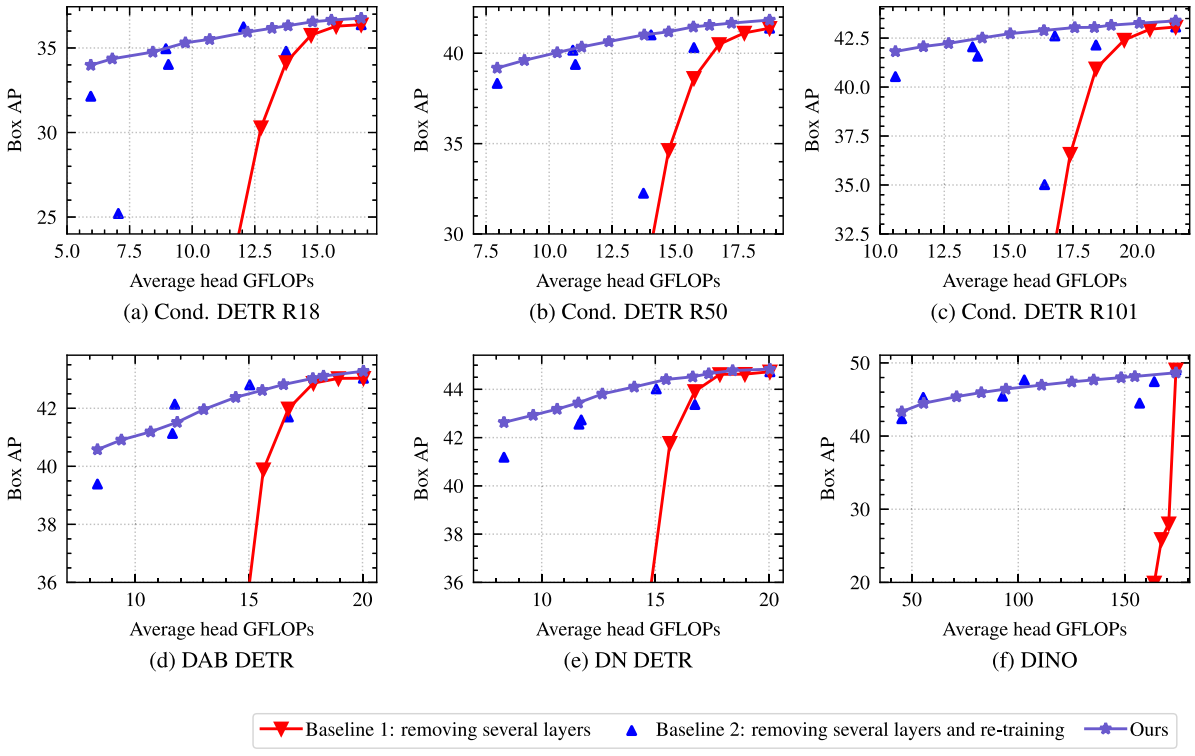


Fig. 4. Overall results. Each curve or individual point represents a individually trained model. (a)–(c) presents comparisons under various backbone networks, whereas the detectors were fixed to Conditional DETR. (d)–(f) presents comparisons under various detector frameworks, whereas the backbone networks were fixed to ResNet-50. The meaning of the marks is indicated by the legend in the lower right corner.

we first perform bipartite matching between the predictions from the heavy head and the light head (denoted as $\hat{y}^h$ and $\hat{y}^l$, respectively):

$$\hat{\sigma}' = \arg\min_{\sigma'} \sum_{i=1}^{N} \mathcal{L}_{\text{match}}(\hat{y}_i^h, \hat{y}_{\sigma'(i)}^l). \tag{12}$$

Then, we compute the knowledge distillation loss for outputs and attention maps of the matched object queries:

$$\mathcal{L}_{\text{decoder}} = \sum_{i=1}^{N} \left[\mathcal{L}_{\text{detr}}(\hat{y}_i^h, \hat{y}_{\hat{\sigma}'(i)}^l) + \lambda_{\text{att}} \cdot \text{MSE}(A_i^h, A_{\hat{\sigma}'(i)}^l) \right], \tag{13}$$

where $\mathcal{L}_{\text{detr}}$ is the same function as detection loss in DETR (Carion et al., 2020), and λ_{att} is a balancing factor. A minor adjustment is made to $\mathcal{L}_{\text{detr}}$ by substituting the focal loss with binary cross entropy loss in order to deal with the float-type class probabilities of $\hat{y}_i^h$. $A_i^h, A_{\hat{\sigma}'(i)}^l$ are the attention maps corresponding to the ith and $\hat{\sigma}'(i)$-th object queries in the heavy and light heads, respectively.

The total knowledge distillation loss is the sum of both encoder and decoder distillation losses,

$$\mathcal{L}_{\text{kd}} = \mathcal{L}_{\text{encoder}} + \mathcal{L}_{\text{decoder}}. \tag{14}$$

The combination distills the knowledge in both the encoder and decoder to achieve better accuracy.

4. Experiments

4.1. Experimental setting

Our experiments were mainly conducted on the challenging MS COCO (Lin et al., 2014) dataset. The box AP on the validation set was reported. Our experiments covered various representative variants of DETR,[2] including Conditional DETR, DAB DETR, DN DETR and DINO, as well as various backbone networks. We default to two heads containing 1-layer encoder, 3-layer decoder and 6-layer encoder, 6-layer decoder, respectively. The model was initialized from ImageNet pre-training and trained for 50 epochs, except for DINO which was trained for 12 epochs. The switching module was fine-tuned for 4 epochs while keeping all other parameters frozen. Our code is based on `detectron2` (Wu et al., 2019) and `detrex` (Ren et al., 2022). All other hyper-parameters were kept consistent with the original setting. Most of our experiments were conducted on 8 NVIDIA 3090 GPUs. More implementation details are available in Appendix A.

4.2. Main results

We present the overall results in Fig. 4. As our motivation is to enable the single model to perform well under different computational budgets, we constructed the following baselines for comparison, which adapt to different computational budgets by modifying the head recipes (*i.e.*, number of encoder and decoder layers). These models are static and share the same computational complexity for different images.

- **Baseline 1: removing several layers.** Since most existing models cannot change their computational cost once trained, we provide a preliminary baseline by shrinking the decoder of a trained model at inference, *i.e.*, using the output of intermediate decoder layers instead of the last decoder layer. As shown in Fig. 4 (red solid line), the baseline can only work under a few fixed budgets,

[2] As the original DETR is known to converge very slowly, we did not directly perform experiments on it.

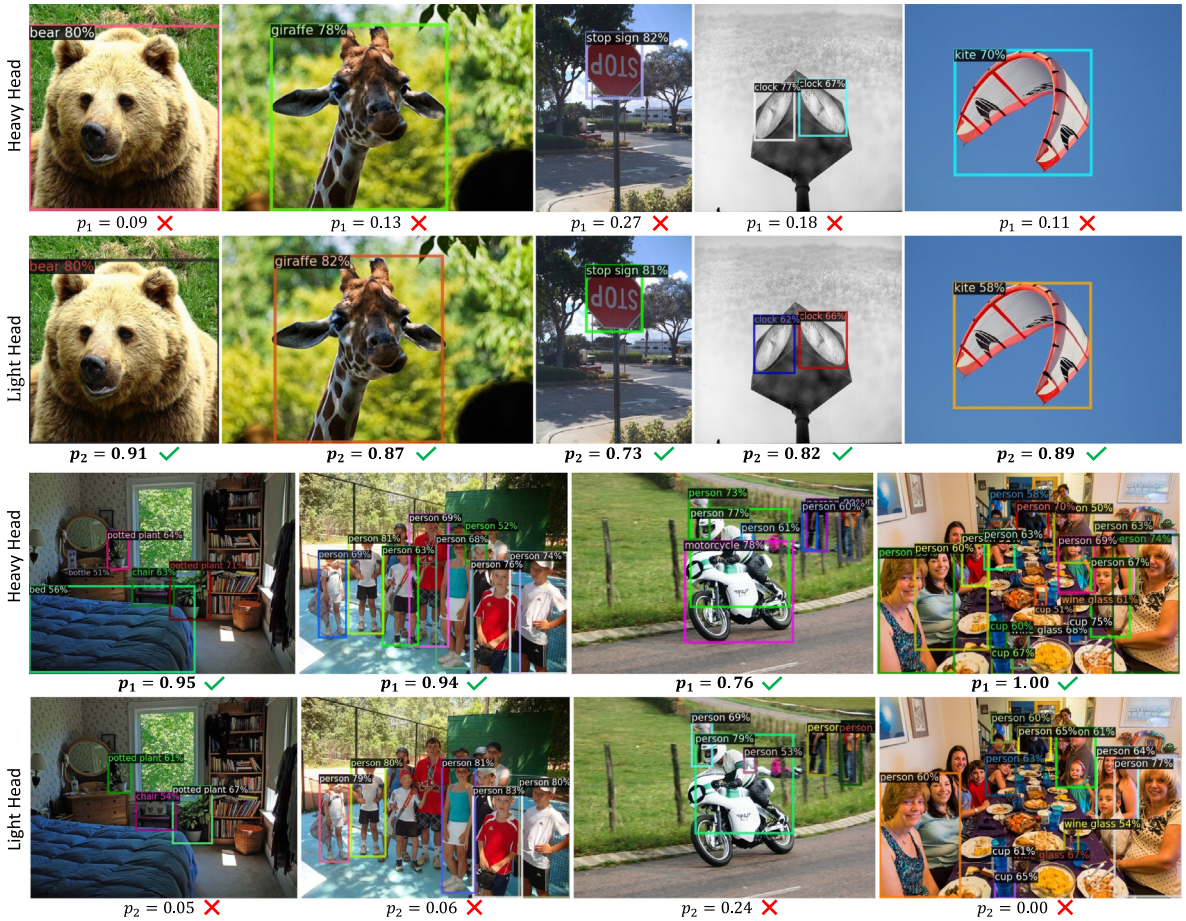


Fig. 5. Visualization of the predictions and switching probabilities. The 1st-2nd rows present some easy images while 3rd-4th rows present some hard images. The selected and not selected heads are denoted by ✓ and ×, respectively.

making it less flexible. Besides, the baseline performs poorly when the budget is limited.

- **Baseline 2: removing several layers and re-training.** Another alternative is to train a set of individual models with different computational overheads by using different head recipes during training. We present the results of these individual models as blue triangles in Fig. 4 (*i.e.*, each corresponds to a specified numbers of encoder and decoder layers). The performance of these models varies according to the head recipes. While some of these models achieved considerable results at specific computational budgets, they lacked flexibility at inference and were costly to train them all.

For the proposed model, different sampling points (purple stars) corresponded to different computational budgets. Specifically, we took one of the switching modules trained with different computational budgets to (Section 3.5) and performed the inference with dynamic head switching of that module. The advantages of the proposed method include two aspects: (1) *flexibility*: once trained, the proposed model can be switched to different computational budgets at inference, which baseline 2 is unable to achieve; (2) *better trade-off between accuracy and efficiency*: under different computational budgets, the proposed method achieved higher accuracy compared to baseline 1 and competitive results with the optimal head recipes of baseline 2. Note that the module typically brings only 0.01 GFLOPs of computational overhead, which is negligible compared to the detection heads. Furthermore, our method consistently demonstrated advantages across different detection heads and backbone networks, highlighting its generalizability.

4.3. Qualitative analysis

To qualitatively show the effect of DHS, we visualized the predicted probabilities of the dynamic switching module and the predictions of the two transformer heads in Fig. 5. The predicted probabilities for the heads are denoted as p_1, p_2 respectively. For simple images containing only a few distinct objects, the module prefers the light head. For example, the images in the first two rows only contain one or two objects and these objects are clearly distinguished from the background and each other. The module confidently selected the lighter head, as its predictions were as accurate as those of the heavier head. In contrast, the last two rows of images depict scenes with multiple objects and occlusions. In these cases, the lighter head performed noticeably worse than the heavier head, for example, resulting in missing objects. The switching module correctly selected the heavy head for these images. These results demonstrate that the proposed DHS is capable of selecting the appropriate head based on the difficulty level of the image.

4.4. Ablation study

We conducted ablation experiments to verify the effectiveness of the proposed method. The experiments were conducted on Conditional DETR with ResNet-18 as the backbone network, trained for 24 epochs

Table 1
Accuracy cost ratio of different heads. FPS was measured on the same V100 GPU with input size (1333, 800).

Enc.	Dec.	FPS	GFLOPs	#Params.	AP	ACR
6	6	31.6	47.9	30.7M	32.9	0.50
6	3	41.5	44.9	25.2M	31.2	0.59
3	6	35.1	43.2	26.8M	32.2	0.54
1	6	37.6	40.1	24.2M	31.3	0.55
3	3	47.4	40.2	21.3M	30.0	**0.62**
3	1	63.0	38.2	17.6M	19.6	0.41
1	3	52.5	37.1	18.6M	28.2	**0.62**
0	3	54.4	35.5	17.3M	25.9	0.57

Table 2
Ablation study on the joint training. `R18-x-y` indicates model (with ResNet-18 as backbone) contains `x` layers of encoder and `y` layers of decoder.

	R18-1-3			R18-6-6		
	AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}
Individually	28.2	46.8	29.0	32.7	52.5	33.6
Jointly	28.4	46.8	29.4	33.1	52.9	34.4
w/ OHD	**30.0**	**49.0**	**30.9**	**33.4**	**53.2**	**34.8**

Table 3
Ablation study on the number of heads. `R18-x-y` indicates model (with ResNet-18) contains `x` layers of encoder and `y` layers of decoder.

	R18-1-3		R18-3-3		R18-6-6	
	AP	AP_{50}	AP	AP_{50}	AP	AP_{50}
Individually	28.2	46.8	**30.0**	48.9	32.7	52.5
Jointly (2 heads)	**28.4**	46.8	–	–	**33.1**	52.9
Jointly (3 heads)	27.3	45.3	29.3	47.8	31.3	50.1

(to save training time). By default, we used two heads containing 1-layer encoder, 3-layer decoder and 6-layer encoder, 6-layer decoder, respectively. We started by exploring the proper head recipes, and then investigated the effect of the proposed method and alternatives.

Head Recipes. We first studied how to select suitable detection head recipes with favorable trade-offs between accuracy and efficiency. We compared the models with encoder and decoder differing in the number of transformer layers in Table 1. We found that several recipes achieved better accuracy cost ratio (ACR, as described in Section 3.4). For example, the model with a 1-layer encoder, 3-layer decoder head improves FPS by 66% over the original model with a 6-layer encoder, 6-layer decoder head, while only reducing AP by 14%. We consequently chose these heads as the default light heads. The recipe allows the DHS to work well with the lowest budget.

Joint Training. We investigated the effects of joint training (*i.e.*, training two heads on a shared backbone) in Table 2. We compared this training method to the individual training of two standalone networks. We found that our model achieved a slight improvement (0.2–0.4% AP) even without online head distillation. This could be attributed to that the back propagation of both heads allows more sufficient supervision of the backbone network. These results ensure that our final model works well with both the highest and lowest computational budgets.

Number of Heads. We compared the results of training 2 and 3 heads simultaneously in Table 3. We found that when more heads were used, the performance of each head was degraded. It could be that too many heads over-supervised the backbone network, with different heads having conflicting requests for backbone features, causing each head suboptimal. Given that the performance of each heads directly affects the final result, and that more heads impose additional training cost, we use two heads by default.

Online Head Distillation. We compared the results under different computational budgets for the model with and without OHD in Fig. 7. The overall performance of the model was significantly improved with OHD. Additionally, we note that OHD also improved the heavy head slightly, which could be attributed to the backbone sharing of the heads. To further validate the generalizability of OHD, we conducted experiments with different head recipes. We varied the number of encoder and decoder layers for the lighter head, keeping the heavier head constant (*i.e.*, 6-layer encoder and 6-layer decoder). As shown in Table 4, OHD achieved consistent improvements over these lighter heads. These experiments validated the effectiveness and generalizability of the proposed online head distillation method, mitigating accuracy drop when executing the lighter heads. More ablation studies on OHD are provided in Appendix B.

Table 4
Results of online head distillation on different head recipes.

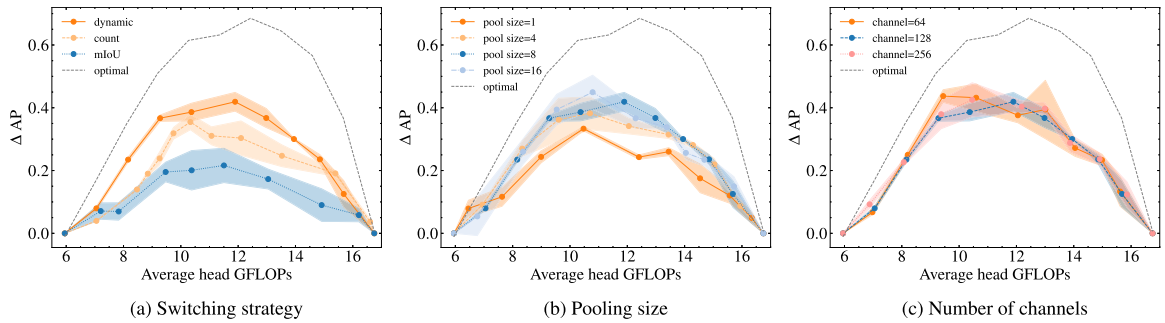
Enc.	Dec.	OHD	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
1	3		28.2	46.8	29.0	11.9	30.5	41.8
		✓	**30.0**	49.0	30.9	12.6	32.6	44.6
3	3		30.0	48.9	30.7	12.5	32.0	45.0
		✓	**31.8**	50.9	33.3	13.9	33.9	48.2
3	6		32.2	52.0	33.4	13.7	34.7	48.3
		✓	**33.1**	52.9	34.5	14.1	35.6	49.8

Switching Strategies. To investigate the impact of dynamic switching strategies, we set two heuristic baselines: (1) Switch according to the number of objects (termed as *count*), *i.e.*, estimating the number of objects in the image and assigning a heavy head if the number of objects exceeds a threshold. (2) Switch according to the degree of crowding (termed as *mIoU*), *i.e.*, estimating the mean box IoU between all object pairs in the image and assigning a heavy head if the mean IoU exceeds a threshold. Models under different computational budgets could be obtained by adjusting the thresholds. Furthermore, we estimated an optimal switching strategy that, within a given budget, prioritizes assigning the heavier head to the image that exhibited a larger accuracy difference between the two heads (measured by the mean IoU of predictions and their matched GT). The results are shown in Fig. 6(a), where ΔAP represents the improvement relative to randomly executing two heads according to the computational budget. Details about these heuristic baselines and the definition of ΔAP are available in Appendices C and D, respectively. Our dynamic switching strategy yielded best results, suggesting that judging the difficulty of an image solely based on the number of objects or crowding is sub-optimal.

Switching Module. We compared the effect of pooling size and number of channels on the final result in Fig. 6(b) and (c). Larger pooling sizes preserved more spatial information and therefore led to better results, with saturation when the pooling size was larger than 8. Our module was not sensitive to the number of channels, yet too few channels (*i.e.*, 64) yield more variance. Accordingly, we defaulted to 8 and 128 as the pooling size and the number of channels, respectively.

5. Conclusions

In this work, we investigated how to improve the accuracy of DETR-like models under different computational budgets to pursue a better trade-off between accuracy and efficiency. We first analyzed the essential components of DETR and found that the transformer head of DETR typically incurs significant computational overhead. Motivated by the observation that heavy and light heads predict similar results on easy images, we proposed dynamic head switching to adaptively select the proper head according to the difficulty of the image. To alleviate accuracy degradation, we further proposed online head distillation to improve the accuracy of the light heads with the guidance of the heavy head. The proposed method allows the model to adaptively process the images, enabling a better trade-off between accuracy and efficiency. Extensive experiments on the MS COCO dataset verified the effectiveness of the proposed method. We hope our work could inspire future studies toward dynamic and efficient DETRs.



(a) Switching strategy (b) Pooling size (c) Number of channels

Fig. 6. Ablation studies of dynamic head switching module. ΔAP represents the improvement relative to randomly executing two heads according to the computational budget. (a) Different switching strategy. `count` and `mIoU` denote the heuristic switching strategies for comparison. (b) Different pooling size. (c) Different number of channels. Each experiment was conducted three times, with solid lines indicating the mean and shaded areas indicating the standard deviation. The black dashed line indicates the estimated optimal switching strategy. Best viewed in color.

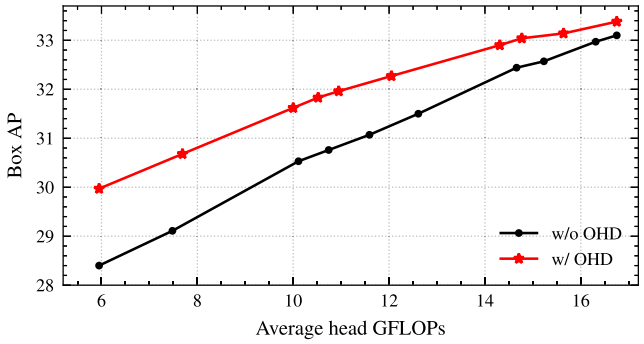


Fig. 7. Results under different computational budget w/ and w/o OHD.

Table B.5
Comparison of different distillation strategies. '+' means the method is performed on the basis of encoder distillation.

	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
baseline	28.2	46.8	29.0	11.9	30.5	41.8
FitNet-style	29.0	47.7	30.1	12.4	31.4	43.8
Ours (Encoder)	**29.5**	48.3	30.3	12.7	31.9	43.8
+ D^3DETR-style	29.0	47.4	29.9	12.6	31.3	43.4
+ Ours (Decoder)	**30.0**	49.0	30.9	12.6	32.6	44.6

CRediT authorship contribution statement

Hang Chen: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Chufeng Tang:** Writing – review & editing, Supervision, Methodology. **Xiaolin Hu:** Writing – review & editing, Supervision, Resources, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

We thank Gang Zhang for the valuable feedback. This work was supported by the National Natural Science Foundation of China (No. U2341228) and THU-Bosch JCML center.

Appendix A. More implementation details

Dynamic switching module. Given image features, the dynamic switching module first downsamples them to a fixed size of $m \times m$ using max pooling (m is set to 8 by default). The pooled features are then flattened and passed through two fully-connected layers. The number of hidden layer channels is set to 128, and the activation function used is GELU. Finally, the features pass through one more fully-connected layer with output channels n, which corresponds to the number of heads. The module was fine-tuned after training the DETR with multiple heads, keeping the rest of parameters frozen. We used the AdamW optimizer with a learning rate of 10^{-5} and fine-tuned for 4 epochs, using a batch size of 16. In our experiments, we set λ_{budget} in Eq. (6) to 10.

Online head distillation. The loss weights of encoder and decoder distillation (λ_{encoder} and λ_{decoder}) were set to 10 and 1, respectively. For DINO, which utilizes deformable attention instead of the standard cross attention, we followed to Eq. (11) and applied the same sampling points as query for the L2 loss in encoder distillation. The decoder distillation was not performed in this case.

Appendix B. More details about OHD

We compared different implementations of online head distillation in Table B.5. For encoder distillation, our method outperforms FitNet-style distillation, which assigns the same weights to all locations. This improvement is reflected in a 0.5% increase in AP. This result suggests that regions where the model paid more attention to are more critical in distillation. For decoder distillation, we compared different methods based on encoder distillation. Some previous works (Chen et al., 2022a; Chang et al., 2022; Wang et al., 2022) also feed the teacher's object queries into the student's decoder to align the predictions for knowledge distillation. We empirically found that this practice does not apply to online knowledge distillation since the teacher object queries are variable during training. We observed that directly applying D^3DETR-style method led to a decline in accuracy. The reason could be that feeding the variable object queries into the student head misled the bipartite matching process and led to unstable training. Instead, our decoder distillation method achieved a 0.5% improvement in AP over encoder distillation, demonstrating its effectiveness.

Appendix C. More details about the heuristic baselines

The heuristic switching strategies were used as baselines for DHS. To determine the difficulty of an image, we employed heuristic rules that assign heads based on the number of objects or the degree of crowding. To achieve that, we used two 3-layer MLP ϕ_{obj} and ϕ_{IoU}

Table E.6
Quantitative comparison between easy and hard samples. For images with different numbers of objects, we computed the mean switching probabilities for the heavier head (head 1) and the lighter head (head 2) respectively, as well as the mean IoU of their predictions with respect to the matched GT.

#Objects	<2	[2, 4)	[4, 8)	[8, 16)	≥16
mean p_1	0.175	0.279	0.645	0.881	0.985
mean p_2	0.825	0.721	0.355	0.119	0.015
mean IoU_1	0.845	0.867	0.792	0.703	0.626
mean IoU_2	0.841	0.860	0.782	0.693	0.614

(shared the same structure as ϕ_d in Section 3.5) to estimate the number of objects and the mean IoU of boxes, respectively. The mean IoU is calculated by considering all pairs of ground-truth boxes in the image. The MLP has a hidden layer with 128 channels and the GELU activation function is used. No activation function is applied to the last layer. To train these modules, we use the mean squared error (MSE) loss with a loss weight of λ_h ($\lambda_h = 10$ in our experiments). After training the DETR model with multiple heads, we incorporated these two modules and fine-tuned them for 4 epochs each, while keeping all other parameters frozen. The setups were consistent with Appendix A. During inference, we adjust a threshold and execute the heavy head if the output of the module is above the threshold; otherwise, we execute the light head. Different thresholds lead to models with different overall computational complexity.

Appendix D. More details about ΔAP

The purpose of the switching module is to choose heads with varying computational complexity based on the image difficulty. To measure the effectiveness of this selection, we introduce a metric called ΔAP, which represents the improvement in average precision (AP) compared to randomly selecting heads within a given budget. For a given computational budget, we determine the number of executions of heavy and light heads that would result in an average FLOPs equal to the budget. The detection accuracy of this execution strategy is denoted as $\text{AP}_{\text{random}}$. The detection accuracy of the switching strategy under evaluation is denoted as $\text{AP}_{\text{strategy}}$. We then calculate ΔAP as their difference:

$$\Delta\text{AP} = \text{AP}_{\text{strategy}} - \text{AP}_{\text{random}}. \tag{D.1}$$

Note that when the computational budget is equal to the FLOPs of the heaviest or lightest head, all strategies reduce to selecting either the heaviest or lightest head exclusively. Therefore, at both ends of Fig. 6, ΔAP is equal to zero.

Appendix E. Quantitative analysis

This section aims to present a quantitative analysis of the behavior of the switching module in relation to image difficulty. The experimental setup was consistent with Section 4.3. Note that it is difficult to quantitatively define easy and hard samples, because the difficulty of an image depends on various factors. In this experiment, we gauged difficulty based on the number of objects within the image, as images with a higher object count were considered more challenging. As shown in Table E.6, for more difficult images (containing more number of objects), the module preferred to call a heavier head and the IoU gap between the two heads was larger. This observation aligned with the findings discussed in Section 4.3, supporting the assertion that the proposed method effectively adapted head selection based on image difficulty.

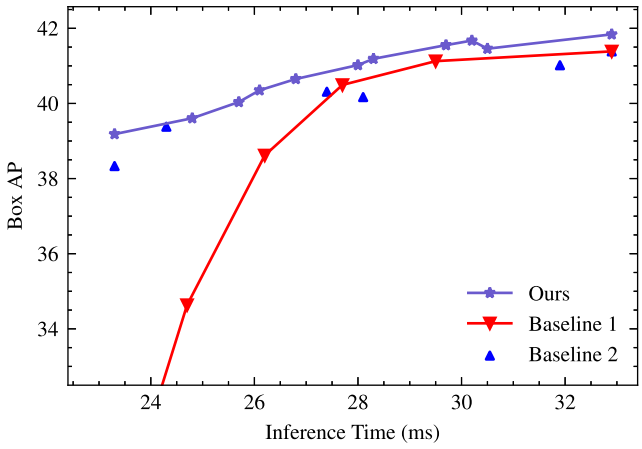


Fig. F.8. Comparison of inference time and AP. The models were based on Conditional DETR with ResNet-50 backbone. The inference times were measured on an NVIDIA 3090 GPU.

Appendix F. Inference time

To validate the inference speed on real devices, we compared the actual inference times and AP between the proposed model and the baselines (see Section 4.2). The results are shown in Fig. F.8. Inference time denotes the duration taken by the model to process a single image (calculated as an average across 5000 images in the COCO validation set). The switching module took about 0.3 ms, which is about 1% of the total inference time. Our findings aligned with those presented in Section 4.2, demonstrating that the proposed method delivered competitive AP across various inference times while offering the flexibility to adjust its budget after training.

References

Bolukbasi, T., Wang, J., Dekel, O., Saligrama, V., 2017. Adaptive neural networks for efficient inference. In: International Conference on Machine Learning. pp. 527–536.
Cai, Z., Vasconcelos, N., 2021. Cascade R-CNN: high quality object detection and instance segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 1483–1498.
Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S., 2020. End-to-end object detection with transformers. In: European Conference on Computer Vision. pp. 213–229.
Chang, J., Wang, S., Xu, G., Chen, Z., Yang, C., Zhao, F., 2022. DETRDistill: A universal knowledge distillation framework for DETR-families. arXiv preprint arXiv:2211.10156.
Chen, X., Chen, J., Liu, Y., Zeng, G., 2022a. D^{3}ETR: Decoder distillation for detection transformer. arXiv preprint arXiv:2211.09768.
Chen, Q., Chen, X., Wang, J., Feng, H., Han, J., Ding, E., Zeng, G., Wang, J., 2022b. Group DETR: Fast DETR training with group-wise one-to-many assignment. arXiv preprint arXiv:2207.13085.
Chen, G., Choi, W., Yu, X., Han, T.X., Chandraker, M., 2017. Learning efficient object detection models with knowledge distillation. In: Advances in Neural Information Processing Systems. pp. 742–751.
Dai, X., Jiang, Z., Wu, Z., Bao, Y., Wang, Z., Liu, S., Zhou, E., 2021. General instance distillation for object detection. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 7842–7851.
Du, Z., Zhang, R., Chang, M., Zhang, X., Liu, S., Chen, T., Chen, Y., 2021. Distilling object detectors with feature richness. In: Advances in Neural Information Processing Systems. pp. 5213–5224.
Girshick, R.B., 2015. Fast R-CNN. In: IEEE International Conference on Computer Vision. pp. 1440–1448.
Girshick, R.B., Donahue, J., Darrell, T., Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 580–587.
Han, Y., Huang, G., Song, S., Yang, L., Wang, H., Wang, Y., 2022. Dynamic neural networks: A survey. IEEE Trans. Pattern Anal. Mach. Intell. 44 (11), 7436–7456.
He, K., Gkioxari, G., Dollár, P., Girshick, R.B., 2017. Mask R-CNN. In: IEEE International Conference on Computer Vision. pp. 2980–2988.

Hehn, T.M., Kooij, J.F.P., Hamprecht, F.A., 2020. End-to-end learning of decision trees and forests. Int. J. Comput. Vis. 128 (4), 997–1011.

Hinton, G.E., Vinyals, O., Dean, J., 2015. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531.

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., 2017. MobileNets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.

Huang, G., Chen, D., Li, T., Wu, F., van der Maaten, L., Weinberger, K.Q., 2018. Multi-scale dense networks for resource efficient image classification. In: International Conference on Learning Representations.

Jang, E., Gu, S., Poole, B., 2017. Categorical reparameterization with gumbel-softmax. In: International Conference on Learning Representations.

Kang, Z., Zhang, P., Zhang, X., Sun, J., Zheng, N., 2021. Instance-conditional knowledge distillation for object detection. In: Advances in Neural Information Processing Systems. pp. 16468–16480.

Li, Q., Jin, S., Yan, J., 2017. Mimicking very efficient network for object detection. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 7341–7349.

Li, G., Li, X., Wang, Y., Zhang, S., Wu, Y., Liang, D., 2022a. Knowledge distillation for object detection via rank mimicking and prediction-guided feature imitation. In: AAAI Conference on Artificial Intelligence. pp. 1306–1313.

Li, X., Wang, W., Wu, L., Chen, S., Hu, X., Li, J., Tang, J., Yang, J., 2020. Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. In: Advances in Neural Information Processing Systems.

Li, F., Zeng, A., Liu, S., Zhang, H., Li, H., Zhang, L., Ni, L.M., 2023. Lite detr: An interleaved multi-scale encoder for efficient detr. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 18558–18567.

Li, F., Zhang, H., Liu, S., Guo, J., Ni, L.M., Zhang, L., 2022b. DN-DETR: accelerate DETR training by introducing query DeNoising. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 13609–13617.

Liang, D., Liu, A., Wu, L., Li, C., Qian, R., Chen, X., 2023. Privacy-preserving multi-source semi-supervised domain adaptation for seizure prediction. Cogn. Neurodyn. 1–14.

Lin, T., Goyal, P., Girshick, R.B., He, K., Dollár, P., 2017. Focal loss for dense object detection. In: IEEE International Conference on Computer Vision. pp. 2999–3007.

Lin, T., Maire, M., Belongie, S.J., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft COCO: common objects in context. In: European Conference on Computer Vision. pp. 740–755.

Liu, L., Deng, J., 2018. Dynamic deep neural networks: Optimizing accuracy-efficiency trade-offs by selective execution. In: AAAI Conference on Artificial Intelligence. pp. 3675–3682.

Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., Zhang, C., 2017. Learning efficient convolutional networks through network slimming. In: IEEE International Conference on Computer Vision. pp. 2755–2763.

Liu, S., Li, F., Zhang, H., Yang, X., Qi, X., Su, H., Zhu, J., Zhang, L., 2022. DAB-DETR: dynamic anchor boxes are better queries for DETR. In: International Conference on Learning Representations.

Meng, D., Chen, X., Fan, Z., Zeng, G., Li, H., Yuan, Y., Sun, L., Wang, J., 2021. Conditional DETR for fast training convergence. In: IEEE International Conference on Computer Vision. pp. 3631–3640.

Nguyen, C.H., Nguyen, T.C., Tang, T.N., Phan, N.L.H., 2022. Improving object detection by label assignment distillation. In: IEEE Winter Conference on Applications of Computer Vision. pp. 1322–1331.

Ren, S., He, K., Girshick, R.B., Sun, J., 2015. Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems. pp. 91–99.

Ren, T., Liu, S., Zhang, H., Li, F., Liao, X., Zhang, L., 2022. detrex. https://github.com/IDEA-Research/detrex.

Roh, B., Shin, J., Shin, W., Kim, S., 2022. Sparse DETR: efficient end-to-end object detection with learnable sparsity. In: International Conference on Learning Representations.

Tanno, R., Arulkumaran, K., Alexander, D.C., Criminisi, A., Nori, A.V., 2019. Adaptive neural trees. In: International Conference on Machine Learning. pp. 6166–6175.

Tian, Z., Shen, C., Chen, H., He, T., 2019. FCOS: fully convolutional one-stage object detection. In: IEEE International Conference on Computer Vision. pp. 9626–9635.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. In: Advances in Neural Information Processing Systems. pp. 5998–6008.

Wang, X., Yu, F., Dou, Z., Darrell, T., Gonzalez, J.E., 2018. SkipNet: Learning dynamic routing in convolutional networks. In: European Conference on Computer Vision. pp. 420–436.

Wang, T., Yuan, L., Chen, Y., Feng, J., Yan, S., 2021. PnP-DETR: Towards efficient visual analysis with transformers. In: IEEE International Conference on Computer Vision. pp. 4641–4650.

Wang, T., Yuan, L., Zhang, X., Feng, J., 2019. Distilling object detectors with fine-grained feature imitation. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 4933–4942.

Wang, W., Zhang, Y., Zhu, L., 2023. DRF-DRC: dynamic receptive field and dense residual connections for model compression. Cogn. Neurodyn. 17 (6), 1561–1573.

Wang, Y., et al., 2022. Knowledge distillation for detection transformer with consistent distillation points sampling. arXiv preprint arXiv:2211.08071.

Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., Girshick, R., 2019. Detectron2. https://github.com/facebookresearch/detectron2.

Yang, C., Ochal, M., Storkey, A.J., Crowley, E.J., 2022a. Prediction-guided distillation for dense object detection. arXiv preprint arXiv:2203.05469.

Yang, J., Shi, S., Ding, R., Wang, Z., Qi, X., 2022b. Towards efficient 3D object detection with knowledge distillation. In: Advances in Neural Information Processing Systems.

Yao, Z., Ai, J., Li, B., Zhang, C., 2021. Efficient detr: improving end-to-end object detector with dense prior. arXiv preprint arXiv:2104.01318.

Zhang, S., Chi, C., Yao, Y., Lei, Z., Li, S.Z., 2020. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 9756–9765.

Zhang, X., Zhou, X., Lin, M., Sun, J., 2018. ShuffleNet: An extremely efficient convolutional neural network for mobile devices. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 6848–6856.

Zhao, Y., Lv, W., Xu, S., Wei, J., Wang, G., Dang, Q., Liu, Y., Chen, J., 2024. Detrs beat yolos on real-time object detection. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 16965–16974.

Zheng, Z., Ye, R., Wang, P., Ren, D., Zuo, W., Hou, Q., Cheng, M., 2022. Localization distillation for dense object detection. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 9397–9406.

Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J., 2021. Deformable DETR: deformable transformers for end-to-end object detection. In: International Conference on Learning Representations.