A. More Details of DETR

A.1. Multi-head Attention

**Single-head Attention** (SHA). We start with the attention mechanism with single head. Given the key-value sequence $x_{kv}$, query sequence $x_q$, and linear projection of the attention head $f_q$, $f_k$, $f_v$, we can compute so-called query $q$, key $k$, and value $v$ embeddings:

$$
q = f_q(x_q + \phi_q);
\quad k = f_k(x_{kv} + \phi_p);
\quad v = f_v(x_{kv}),
$$

where $\phi_p$ is the positional embedding for the key-value sequences, and $\phi_q$ is the positional embedding for the query sequences. And the attention outputs $\hat{q}$ are computed by the aggregation of weighted values:

$$
\hat{q} = \text{SHA}(q, k, v) = \sum_j \alpha_{i,j} v_j,
$$

where the attention weights is based on softmax of scaled dot products between $i$-th query and $j$-th key:

$$
\alpha_{i,j} = \text{Softmax}(\frac{q_i k_j^T}{\sqrt{d_k}}),
$$

where $d_k$ is a scaling factor.

**Multi-head Attention** (MHA). Through concatenating $N$ single-head attentions followed by a projection $f_{\text{MHA}}$, we can compute the multi-head attention:

$$
\hat{q} = \text{MHA}(q, k, v) = f_{\text{MHA}}\left(\text{concat}[\text{SHA}_0(q, k, v), \ldots, \text{SHA}_N(q, k, v)]\right).
$$

Note that the output $\hat{q}$ is the same size as the input query sequences $x_q$.

A.2. Bipartite Matching

Following [1, 2], we apply the Hungarian algorithm [3] to match the $N$ predictions $\hat{y}$ with the ground truth $y$. The matching loss $H$ is defined as:

$$
H(y, \hat{y}) = \sum_{i=1}^{N} \left[ -\eta_0 \log \tilde{k}_{\sigma(i)} + 1_{(k_i = 1)} L_{\text{box}}(b_i, \tilde{b}_{\sigma(i)}) \right],
$$

where $k$ is the binary classification indicating whether each query is matched ($k_i = 1$) or not ($k_i = 0$), $L_{\text{box}}$ is a combination of generalized IoU loss [6] and $\ell_1$ loss, and $\tilde{\sigma}(i)$ is the index of prediction that matching with $i$-th ground truth optimally. The coefficients of binary classification $\eta_0$, generalized IoU loss $\eta_1$, and $\ell_1$ loss $\eta_2$ in Equation 5 are set to 1, 2, 5 following [1], respectively.

B. More Ablations

**Hyper-parameters.** We follow [1] to set loss weight of $L_{\text{loc}}$ ($\lambda_2$) to 1.0 in all setups and further ablate the $\lambda_0$ and $\lambda_1$ using Conditional DETR on COCO. Figure 1 illustrates the sensitivity of $\lambda_0$ and $\lambda_1$. It suggests that the transfer performance is robust to $\lambda_0$ and $\lambda_1$ variation. To yield the best performance, we set $\lambda_0$, $\lambda_1$ to 3, 10 on ImageNet, and $\lambda_0$, $\lambda_1$ to 0.3, 3 on COCO. For other novel datasets, a simple selection (e.g., $\lambda_1 = 1.0, \lambda_2 = 1.0$) will be okay.

**Data Efficiency.** Self-supervised pretrained models not only achieve high performance when transferring to downstream tasks, but provide a better initialization when using limited data. To verify the data efficiency of Siamese DETR, we consider the transfer performance on the limited amount of downstream datasets. Specifically, we pretrain Conditional DETR on ImageNet and finetune it on 10%/30%/50%/70% PASCAL VOC datasets. All these

---

**Figure 1. Sensitivity of Hyper-parameters in Siamese DETR.**

![Graph showing sensitivity of hyper-parameters in Siamese DETR with λ₀ and λ₁.](image-url)
splits are selected randomly. As shown in Figure 2, Siamese DETR can achieve a similar (49.3 AP) performance with from scratch model (49.9) using only 30% of datasets. Moreover, Siamese DETR outperforms UP-DETR by a large margin in all splits.

**Downstream initialization.** To investigate the downstream initialization of Siamese DETR, we pretrain the Vanilla DETR on ImageNet and only finetune the box prediction and classifier head on PASCAL VOC while keeping the parameters of pretrained CNN backbone, encoder and decoder fixed. We report average recall with detecting top $K$ objects, denoted as AR@$K$.

**Table 1.** Ablations on downstream initialization. We initialize the Vanilla DETR using models pretrained by Siamese DETR, DETReg and UP-DETR without finetuning. We report average recall with detecting top $K$ objects, denoted as AR@$K$.

<table>
<thead>
<tr>
<th>Method</th>
<th>AR@1</th>
<th>AR@10</th>
<th>AR@100</th>
</tr>
</thead>
<tbody>
<tr>
<td>random</td>
<td>0.0</td>
<td>0.1</td>
<td>0.5</td>
</tr>
<tr>
<td>UP-DETR</td>
<td>9.5</td>
<td>17.9</td>
<td>24.0</td>
</tr>
<tr>
<td>DETReg</td>
<td>11.2</td>
<td>20.5</td>
<td>26.5</td>
</tr>
<tr>
<td>ours</td>
<td>12.4</td>
<td>23.0</td>
<td>30.7</td>
</tr>
</tbody>
</table>

More DETR-like architecture. We provide transfer results of Siamese DETR with more advanced DETR-like architecture, *i.e.*, DAB-Deformable-DETR with 300 queries [5] and DN-DAB-Deformable-DETR with 300 queries [4]. We follow the default setup in their origin paper. The results are shown in Table 2. Both DAB-DETR and DN-DETR can benefit from the initialization of Siamese DETR, verifying the generalization of Siamese DETR. Furthermore, Siamese DETR also leads a significant margin when using limited downstream datasets.

**GPU Memory Cost and Iteration Time.** Firstly, it is emphasised that none of three pre-training methods increase extra cost in GPU memory and time during downstream finetuning. We provide a quantitative comparison on GPU memory cost and iteration time during pretraining in Table 3. All three methods follow the same setup, *i.e.*, pretraining Conditional DETR (100 queries) on COCO using the same GPU. The batch size is set to 4 on each GPU. The input images are processed with the same augmentation.

<table>
<thead>
<tr>
<th>Method</th>
<th>GPU Mem.</th>
<th>Iteration time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Siamese DETR</td>
<td>7528 MB</td>
<td>0.4036 s/it</td>
</tr>
<tr>
<td>UP-DETR</td>
<td>7377 MB</td>
<td>0.3118 s/it</td>
</tr>
<tr>
<td>DETReg</td>
<td>8885 MB</td>
<td>0.3528 s/it</td>
</tr>
</tbody>
</table>

**More DETR-like architecture.** We pretrain the model using Siamese DETR on COCO for the 40/60 schedule, then finetune on full/10% PASCAL VOC dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>VOC</th>
<th>VOC 10%</th>
</tr>
</thead>
<tbody>
<tr>
<td>from scratch</td>
<td>57.9</td>
<td>32.2</td>
</tr>
<tr>
<td>Siamese DETR</td>
<td>62.2 (+4.3)</td>
<td>41.8 (+9.6)</td>
</tr>
<tr>
<td>UP-DETR</td>
<td>63.4 (+4.5)</td>
<td>43.6 (+10.7)</td>
</tr>
</tbody>
</table>

**Table 3.** More DETR-like architecture. We pretrain the model using Siamese DETR on COCO for the 40/60 schedule, then finetune on full/10% PASCAL VOC dataset.

**C. More Visualization**

**C.1. Convergence**

Figure 3 and Figure 4 illustrate the convergence curves of models finetuned on COCO and PASCAL VOC, respectively. The model initialized by Siamese DETR converges faster and outperforms its counterparts by significant margins in all setups.

**C.2. More Qualitative Results**

We also provide more qualitative results of box predictions and corresponding attention maps when initializing the downstream model using Siamese DETR, UP-DETR, and DETReg without finetuning in Figure 5. The visualization results verify better transferability of Siamese DETR against its counterpart.
Figure 3. Illustration of convergence curves when finetuned on COCO.

Figure 4. Illustration of convergence curves when finetuned on PASCAL VOC.

References


Figure 5. More visualization on box predictions and attention maps when initializing the downstream models using Siamese DETR, UP-DETR and DETReg without finetuning.