DA-DETR: Domain Adaptive Detection Transformer with Information Fusion

Jingyi Zhang\textsuperscript{1,*}, Jiaxing Huang\textsuperscript{1,*}, Zhipeng Luo\textsuperscript{1,2}, Gongjie Zhang\textsuperscript{1,3}, Xiaojin Zhang\textsuperscript{4}, Shijian Lu\textsuperscript{1,†}
\textsuperscript{1} S-lab, Nanyang Technological University \textsuperscript{2} SenseTime Research \textsuperscript{3} Black Sesame Technologies \textsuperscript{4} Wenzhou University

Abstract

The recent detection transformer (DETR) simplifies the object detection pipeline by removing hand-crafted designs and hyperparameters as employed in conventional two-stage object detectors. However, how to leverage the simple yet effective DETR architecture in domain adaptive object detection is largely neglected. Inspired by the unique DETR attention mechanisms, we design DA-DETR, a domain adaptive object detection transformer that introduces information fusion for effective transfer from a labeled source domain to an unlabeled target domain. DA-DETR introduces a novel CNN-Transformer Blender (CTBlender) that fuses the CNN features and Transformer features ingeniously for effective feature alignment and knowledge transfer across domains. Specifically, CTBlender employs the Transformer features to modulate the CNN features across multiple scales where the high-level semantic information and the low-level spatial information are fused for accurate object identification and localization. Extensive experiments show that DA-DETR achieves superior detection performance consistently across multiple widely adopted domain adaptation benchmarks.

1. Introduction

Object detection aims to predict a bounding box and a class label for interested objects in images and it has been a longstanding challenge in the computer vision research. Most existing work adopts a two-stage detection pipeline that involves heuristic anchor designs, complicated post-processing such as non-maximum suppression (NMS), etc. The recent detection transformer (DETR) \cite{carion2020end} has attracted increasing attention which greatly simplifies the two-stage detection pipeline by removing hand-crafted anchors \cite{huang2017speed, lin2017focal, ren2015faster} and NMS \cite{huang2017speed, lin2017focal, ren2015faster}. Despite its great detection performance under a fully supervised setup, how to leverage the simple yet effective DETR architecture in domain adaptive object detection is largely neglected.

*Equal contribution, \{jingyi.zhang, jiaxing.huang\} @ntu.edu.sg.
†Corresponding author, shijian.lu@ntu.edu.sg.

Different from the conventional CNN-based detection architectures such as Faster RCNN \cite{ren2015faster}, DETR has a CNN backbone followed by a transformer head consisting of an encoder-decoder structure. The CNN backbone and the transformer head learn different types of features \cite{feichtenhofer2019open, liu2019path, zhang2020scaling} - the former largely captures low-level features \cite[e.g.,][]{feichtenhofer2019open, liu2019path, zhang2020scaling} of objects such as edges and lines around object boundaries while the latter largely captures high-level semantic features. At the other end, many prior studies show that fusing different types of features often is often helpful in various visual recognition tasks \cite{feichtenhofer2019open, liu2019path}. Hence, it is very meaningful to investigate how to fuse the two types of DETR features to address the domain adaptive object detection challenge effectively.

We design DA-DETR, a simple yet effective Domain Adaptive DETR that introduces information fusion into the DETR architecture for effective domain adaptive object detection. The core design is a CNN-Transformer Blender (CTBlender) that employs the high-level semantic
features in the Transformer head to conditionally modulate the low-level localization features in the CNN backbone. CTBlender consists of two sequential fusion components, including split-merge fusion (SMF) that fuses CNN and Transformer features within an image and scale aggregation fusion (SAF) that fuses the SMF features across multiple feature scales. Different from the existing weight-and-sum fusion [9, 11], SMF first splits CNN features into multiple groups with different semantic information as captured by the Transformer head and then merges them with channel shuffling for effective information communication among different groups. The SMF features of each scale are then aggregated by SAF for fusing both semantic and localization information across multiple feature scales. Hence, CTBlender captures both semantic and localization features ingeniously which enables comprehensive and effective inter-domain feature alignment with a single discriminator.

The main contributions of this work can be summarized in three aspects. First, we propose DA-DETR, a simple yet effective domain adaptive detection transformer that introduces information fusion for effective domain adaptive object detection. To the best of our knowledge, this is the first work that explores information fusion for domain adaptive object detection. Second, we design a CNN-Transformer Blender that fuses the CNN features and Transformer features ingeniously for effective feature alignment and knowledge transfer across domains. Third, extensive experiments show that DA-DETR achieves superior object detection over multiple widely studied domain adaptation benchmarks as compared with the state-of-the-art as shown in Fig. 1.

2. Related Work

Transformers [56] have achieved great success in various neural language processing (NLP) tasks [3, 14, 40, 46, 47] due to their computational efficiency and scalability. Inspired by the success of transformers in NLP, several studies [5, 8, 12, 15, 16, 62, 70, 71, 77, 78, 81] attempt to adapt transformers to computer vision tasks. For example, ViT [16] adopts transformer for image classification, which splits each image into patches and treats them as input tokens to transformers. SETR [78] extends ViT to semantic segmentation by introducing multiple decoders designed for pixel-wise segmentation. For object detection, different from CNN-based detector, DETR [5] treats detection as a set prediction task [50], which eliminates the dependence on various heuristic and hand-crafted designs such as anchor generation, ROI pooling and non-maximum suppression. Existing vision transformers achieve very promising performance in supervised learning. However, how to adapt and generalize them to unsupervised domain adaptation tasks has been largely neglected. In this work, we investigate domain adaptive detection transformers in an unsupervised manner.

Unsupervised Domain Adaptation (UDA) has been studied extensively in recent years, largely for alleviating data annotation constraint in deep network training in various visual recognition tasks [7, 19, 23, 28, 30–32, 41, 42, 44, 52, 58, 63, 66, 73, 74, 82, 83]. For object detection, the target of UDA is to mitigate the domain gap between a source domain and a target domain, so that the source data can be employed to train better detectors for target data. Most existing domain adaptive detectors [6, 7, 36, 51, 55, 57, 64, 76, 79, 80] adopt CNN-based detector (i.e., Faster R-CNN) and achieve UDA via adversarial learning [7, 26, 36, 51, 55, 64, 79], image translation [1, 1, 29, 33, 35, 38, 54, 67] and self-training [30, 67]. However, little research [59, 68] is conducted on how to adopt DETR in domain adaptive detection tasks, e.g., SFA [59] tackles the domain adaptive object detection via query-based feature alignment and token-wise feature alignment. Differently, we introduce the information fusion idea into the DETR architecture for effective domain adaptive object detection.

Feature Fusion is often helpful in various visual recognition tasks. For example, ResNet [25] fuses features of different network layers with skip connections which achieves obvious performance gains along with increased network depth. Feature Pyramid Network (FPN) [39] fuses features of different scales to build high-level semantic feature maps, and it usually improves the object detection with clear margins in various detection tasks. Some work [9, 11] instead achieves feature fusion with channel-wise attention [27, 45] and spatial-wise attention [4, 37, 60, 61]. For example, AFF [11] presents a multi-scale channel attention module for better fusion features from different layers and scales that capture different types of semantics. LS-DeconvNet [9] introduces a gated fusion layer to fuse RGB and depth features effectively. We explore information fusion for domain adaptive detection transformer, and design a CNN-Transformer Blender for fusing CNN features and Transformer features for effective domain adaptive object detection. Different from [9, 11] that adopts a weight-and-sum strategy, our CNN-Transformer Blender employs the Transformer features to modulate the CNN features for more effective adversarial feature alignment.

3. Preliminaries of Detection Transformer

DETR [5] consists of a CNN backbone [25] to extract features, an encoder-decoder transformer and a simple feed forward network (FFN) to make final detection prediction. Given an image $x$, the CNN backbone $G$ first generates feature $f$ and then reshapes $f$ to a vector. The encoder-decoder in DETR follows the standard architecture of the transformer [56], which consists of multiple multi-head self-
The work focuses on the problem of unsupervised domain adaptation (UDA) in object detection. It involves a source domain $D_s$ and a target domain $D_t$, where $D_s = \{(x_s^i, y_s^i)\}_{i=1}^{N_s}$ is fully labeled, and $y_s^i$ represents the labels of the sample image $x_s^i$. The goal is to train a detection transformer that well performs on unlabeled target-domain data $x_t^i$. The baseline model is trained with the labeled source data (i.e., $D_s$) only:

$$\mathcal{L}_{det} = l(T(G(x_s)), y_s),$$

where $G$ denotes backbone, $T$ denotes transformer encoder-decoder, and $l(\cdot)$ denotes the supervised detection loss that consists of a matching cost and a Hungarian loss [5, 81] for object category and object box predictions.

### 4.2. Framework Overview

As shown in Fig. 2, the proposed DA-DETR consists of a base detector (including a backbone $G$ and a transformer encoder-decoder $T$), a discriminator $C_d$ and a CNN-Transformer Blender (CTBlender). We adopt the deformable-DETR [81] as the base detector, where $G$ extracts features from the input images and $T$ predicts a set of bounding boxes and pre-defined semantic categories according to the extracted features. CTBlender consists of two sub-modules including a split-merge fusion (SMF) and a scale aggregation fusion (SAF) as in Fig. 3. Taking the

1x1 convolution. The lowest resolution feature map, i.e., $f^4$, is extracted by 3x3 stride 2 convolution on the output feature maps of Block C5. Such multi-scale design enables attention to capture relationships among different-scale features effectively.

### 4.1. Task Definition

The focus is on the problem of unsupervised domain adaptation (UDA) in object detection. It involves a source domain $D_s$ and a target domain $D_t$, where $D_s = \{(x_s^i, y_s^i)\}_{i=1}^{N_s}$ is fully labeled, and $y_s^i$ represents the labels of the sample image $x_s^i$. The goal is to train a detection transformer that well performs on unlabeled target-domain data $x_t^i$. The baseline model is trained with the labeled source data (i.e., $D_s$) only:

$$\mathcal{L}_{det} = l(T(G(x_s)), y_s),$$

where $G$ denotes backbone, $T$ denotes transformer encoder-decoder, and $l(\cdot)$ denotes the supervised detection loss that consists of a matching cost and a Hungarian loss [5, 81] for object category and object box predictions.
CNN features from $G$ and the Transformer features from the encoder $E$ as inputs, CTBlender fuses the positional and semantic information in Transformer features and the localization information in CNN features for comprehensive and effective feature alignment across domains.

Given an input image from either source or target domain, the backbone $G$ will first produce multi-scale features $f^l (l = 1, 2, 3, 4)$ and then feeds them to the transformer encoder to obtain Transformer features $p^l (l = 1, 2, 3, 4)$. For supervised learning, the Transformer features generated by source images $x_s \in D_s$ are further fed to decoder to predict a set of bounding boxes and pre-defined semantic categories, which will be used to calculate a detection loss $L_{det}$ under the supervision of the corresponding ground-truth label $y_s \in D_s$. For unsupervised learning, CTBlender takes $f^l (l = 1, 2, 3, 4)$ and $p^l (l = 1, 2, 3, 4)$ generated by both source and target images (i.e., $x_s \in D_s$ and $x_t \in D_t$) as inputs. Finally, the output of CTBlender is fed to the discriminator $C_d$ to compute an adversarial loss $L_{adv}$ for inter-domain feature alignment. The overall network is optimized by the adversarial loss $L_{adv}$ and the detection loss $L_{det}$.

4.3. CNN-Transformer Blender

One key component in DA-DETR is a CNN-Transformer Blender (CTBlender) that fuses different features for effective domain alignment. CTBlender takes multi-scale CNN features and the corresponding multi-scale Transformer features as the input, where the semantic and positional information in the Transformer features are fused with the localization information in the CNN features via split-merge fusion (SMF). The SMF-fused features are then aggregated across multiple scales via scale aggregation fusion (SAF).

**Split-Merge Fusion.** In SMF, the rich semantic and positional information in the multi-scale Transformer features $p^l = \{p^l_j\}_{j=1}^{L_l}$ are exploited to fuse with multi-scale CNN features $f^l = \{f^l_i\}_{i=1}^{L_l}$ for adversarial feature alignment across domains. As SMF operations at each feature scale are the same, we take the first scale $l = 1$ as an example to illustrate how we perform split-merge fusion.

Inspired by the split-fuse-merge in [72], SMF first splits CNN features into multiple groups and then fuses them with the Transformer features. After that, the fused features are merged with channel shuffling for effective information communication among different groups. Given a Transformer feature $p^1 \in \mathbb{R}^{C \times H \times W}$ and a backbone CNN feature $f^1 \in \mathbb{R}^{C \times H \times W}$ ($C, H, W$ indicate the number of channel of feature map, and the height and the width of feature map, respectively), $p^1$ is first split into $K$ groups evenly along channels, i.e., $\{p^1_k\}_{k=1}^{K} \in \mathbb{R}^{(C/K) \times H \times W}$, where
each group captures different semantic information of the input image. The fusion in each group is then achieved via spatial-wise fusion and channel-wise fusion, respectively.

For the spatial-wise fusion, the split Transformer features are firstly fed into a normalization layer and then re-weighted by a learnable weight map and a learnable bias map:

\[
\hat{p}^1_{ks} = f_s \left( w_s \cdot GN \left( p^1_k \right) + b_s \right),
\]

where \( f_s(\cdot) \) is an activation function that limits the input in the range of \([0, 1]\).

For the channel-wise fusion, the split Transformer feature is firstly compacted by the Global Average Pooling (GAP) layer. The shuffled features are concatenated to obtain the fused feature map

\[
K
\]

\[
\hat{p}^1_{kc} = f_s \left( w_c \cdot GAP(p^1_k) + b_c \right),
\]

where \( f_s(\cdot) \) is an activation function that limits the input to the range of \([0, 1]\).

Similar to the operation for Transformer feature \( p^1 \), the CNN feature \( f^1 \) is also divided into \( K \) groups along the channels, \( \{f^1_k\}_{k=1}^K \in \mathbb{R}^{(C/K) \times H \times W} \).

We further adopt shuffle operation to enable information communication across channels \([43, 75]\). Specifically, we first re-weight the split CNN feature by the corresponding re-weighted Transformer feature (i.e., \( \hat{p}^1_{ks} \) and \( \hat{p}^1_{kc} \)) to generate re-weighted split CNN feature \( \hat{f}^1_k \).

Then we shuffle \( \hat{f}^1_k \) along channels to enable information flow across channels for better feature fusion. Lastly, we conduct the above operations \( K \) times to generate \( K \) shuffled features for each group, \( \{\hat{f}^1_k\}_{k=1}^K \in \mathbb{R}^{(C/K) \times H \times W} \). The shuffled features are concatenated to obtain the fused feature map

\[
\hat{f}^1 = f_c \left( \hat{f}^1_1, \ldots, \hat{f}^1_k, \ldots, \hat{f}^1_K \right),
\]

where similar operations are conducted to get the results of all levels \( \hat{f} = \{\hat{f}^1\}_{l=1}^L \).

**Scale Aggregation Fusion.** To explicitly perform feature fusion of different scales, we design a scale aggregation fusion (SAF) to aggregate features \( \hat{f} \) with different scale weights as illustrated in the bottom part of Fig. 3.

Specifically, we compact each scale of feature \( \hat{f} = \{\hat{f}^1\}_{l=1}^L \) into a channel-wise vector \( u = \{u^l\}_{l=1}^L \in \mathbb{R}^{C \times 1 \times 1} \) via a Global Average Pooling (GAP) layer. The scale weights \( \alpha_l \) are obtained from channel-wise vectors \( u^l \). Firstly, the channel-wise vectors are merged together to obtain merged vector \( u_m \) by an element-wise addition.

Then, a fully connected layer separates \( u_m \) to \( L \) scale-weight vectors \( \alpha^l \in \mathbb{R}^{C \times 1 \times 1} \). Finally, \( V^a \) is obtained by

\[
V^a = \sum_{l=1}^L \hat{f}^l \cdot \alpha^l,
\]

where \( V^a \) is a highly embedded feature that captures rich semantic information and localization information.

### 4.4. Network Training

The network is trained with two losses, i.e., a supervised object detection loss \( L_{det} \) as defined in Eq. 4 and an adversarial alignment loss \( L_{adv} \) that is defined as follow:

\[
L_{adv} = \mathbb{E}_{(f,p) \in \mathcal{D}_s} \log C_d \left( \mathcal{H} \left( f \right), p \right)) + \mathbb{E}_{(f,p) \in \mathcal{D}_t} \log \left( 1 - C_d \left( \mathcal{H} \left( f \right), p \right) \right),
\]

where \( f = G \left( x \right) \) and \( p = E \left( G \left( x \right) \right) \). \( G \) denotes backbone; \( E \) denotes transformer encoder; \( \mathcal{H} \) denotes CNN-Transformer Blender (CTBlender) and \( C_d \) denotes the discriminator. Both source images \( x_s \) and target images \( x_t \) are utilized to compute adversarial loss.

In summary, the overall optimization objective of DA-DETR is formulated by

\[
\max_{C_d} \min_{G,T,\mathcal{H}} L_{det}(G,T) - \lambda L_{adv}(\mathcal{H},C_d),
\]

where \( T \) denotes the transformer in DETR, \( \lambda \) is the weight factor that balances the influences of \( L_{det} \) and \( L_{adv} \) in training. Note that we adopt a gradient reverse layer (GRL) \([19]\) to enable the gradient of \( L_{adv} \) to be reversed before back-propagating to \( \mathcal{H} \) from \( C_d \).

### 5. Experiments

This section presents experimentation including experiment setups, implementation details, ablation studies, comparisons with the state-of-the-art and discussion. More details are to be described in the ensuing subsections.

#### 5.1. Experimental Setup

**Datasets.** Following \([7, 26, 33, 35, 51, 64]\), we evaluate DA-DETR under four widely adopted domain adaptation scenarios with eight datasets as listed: 1) Normal Weather to Foggy Weather (Cityscapes [10] → Foggy
### Cityscapes → Foggy Cityscapes

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>person</th>
<th>rider</th>
<th>car</th>
<th>truck</th>
<th>bus</th>
<th>train</th>
<th>mcycle</th>
<th>bicycle</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deformable-DETR [81]</td>
<td>ResNet-50</td>
<td>37.7</td>
<td>39.1</td>
<td>44.2</td>
<td>17.2</td>
<td>26.8</td>
<td>5.8</td>
<td>21.6</td>
<td>35.5</td>
<td>28.5</td>
</tr>
<tr>
<td>DAF [7]</td>
<td>ResNet-50</td>
<td>48.2</td>
<td>48.8</td>
<td>61.5</td>
<td>22.6</td>
<td>43.1</td>
<td>20.2</td>
<td>30.3</td>
<td>42.1</td>
<td>39.6</td>
</tr>
<tr>
<td>SWDA [51]</td>
<td>ResNet-50</td>
<td>49.0</td>
<td>49.0</td>
<td>61.4</td>
<td>23.9</td>
<td>43.1</td>
<td>22.9</td>
<td>31.0</td>
<td>45.2</td>
<td>40.7</td>
</tr>
<tr>
<td>SCL [55]</td>
<td>ResNet-50</td>
<td>49.4</td>
<td>48.6</td>
<td>61.2</td>
<td>27.2</td>
<td>41.1</td>
<td>34.8</td>
<td>28.5</td>
<td>42.5</td>
<td>41.7</td>
</tr>
<tr>
<td>GPA [65]</td>
<td>ResNet-50</td>
<td>49.5</td>
<td>46.7</td>
<td>58.6</td>
<td>26.4</td>
<td>42.2</td>
<td>32.3</td>
<td>29.1</td>
<td>41.8</td>
<td>40.8</td>
</tr>
<tr>
<td>CRDA [64]</td>
<td>ResNet-50</td>
<td>49.8</td>
<td>48.4</td>
<td>61.9</td>
<td>22.3</td>
<td>40.7</td>
<td>30.0</td>
<td>29.9</td>
<td>45.4</td>
<td>41.1</td>
</tr>
<tr>
<td>CF [79]</td>
<td>ResNet-50</td>
<td>49.6</td>
<td>49.7</td>
<td>62.6</td>
<td>23.3</td>
<td>43.4</td>
<td>27.4</td>
<td>30.2</td>
<td>44.8</td>
<td>41.4</td>
</tr>
<tr>
<td>SAP [36]</td>
<td>ResNet-50</td>
<td>49.3</td>
<td>49.9</td>
<td>62.5</td>
<td>23.0</td>
<td>44.1</td>
<td>29.4</td>
<td>31.3</td>
<td>45.8</td>
<td>41.9</td>
</tr>
<tr>
<td>SFA [59]</td>
<td>ResNet-50</td>
<td>46.5</td>
<td>48.6</td>
<td>62.6</td>
<td>25.1</td>
<td>46.2</td>
<td>29.4</td>
<td>28.3</td>
<td>44.0</td>
<td>41.3</td>
</tr>
<tr>
<td>MTTrans [68]</td>
<td>ResNet-50</td>
<td>47.7</td>
<td>49.9</td>
<td>65.2</td>
<td>25.8</td>
<td>45.9</td>
<td>33.8</td>
<td>32.6</td>
<td>46.5</td>
<td>43.4</td>
</tr>
<tr>
<td>DA-DETR</td>
<td>ResNet-50</td>
<td>49.9</td>
<td>50.0</td>
<td>63.1</td>
<td>24.0</td>
<td>45.8</td>
<td>37.5</td>
<td>31.6</td>
<td>46.3</td>
<td>43.5</td>
</tr>
</tbody>
</table>

Table 2. Experimental results (%) of the scenario Normal weather to Foggy weather: Cityscapes → Foggy Cityscapes.

### SIM 10k → Cityscapes

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>mAP on Car</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deformable-DETR [81]</td>
<td>ResNet-50</td>
<td>47.4</td>
</tr>
<tr>
<td>DAF [7]</td>
<td>ResNet-50</td>
<td>49.8</td>
</tr>
<tr>
<td>SWDA [51]</td>
<td>ResNet-50</td>
<td>50.5</td>
</tr>
<tr>
<td>SCL [55]</td>
<td>ResNet-50</td>
<td>51.6</td>
</tr>
<tr>
<td>GPA [65]</td>
<td>ResNet-50</td>
<td>51.3</td>
</tr>
<tr>
<td>CRDA [64]</td>
<td>ResNet-50</td>
<td>52.1</td>
</tr>
<tr>
<td>CF [79]</td>
<td>ResNet-50</td>
<td>52.5</td>
</tr>
<tr>
<td>SAP [36]</td>
<td>ResNet-50</td>
<td>52.3</td>
</tr>
<tr>
<td>SFA [59]</td>
<td>ResNet-50</td>
<td>52.6</td>
</tr>
<tr>
<td>MTTrans [68]</td>
<td>ResNet-50</td>
<td>57.9</td>
</tr>
<tr>
<td>DA-DETR</td>
<td>ResNet-50</td>
<td>54.7</td>
</tr>
</tbody>
</table>

Table 3. Experimental results (%) of the scenario Synthetic scene to Real scene: SIM 10k → Cityscapes.

### KITTI → Cityscapes

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>mAP on Car</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deformable-DETR [81]</td>
<td>ResNet-50</td>
<td>39.5</td>
</tr>
<tr>
<td>DAF [7]</td>
<td>ResNet-50</td>
<td>43.6</td>
</tr>
<tr>
<td>SWDA [51]</td>
<td>ResNet-50</td>
<td>44.3</td>
</tr>
<tr>
<td>SCL [55]</td>
<td>ResNet-50</td>
<td>44.5</td>
</tr>
<tr>
<td>GPA [65]</td>
<td>ResNet-50</td>
<td>43.2</td>
</tr>
<tr>
<td>CRDA [64]</td>
<td>ResNet-50</td>
<td>44.8</td>
</tr>
<tr>
<td>CF [79]</td>
<td>ResNet-50</td>
<td>45.2</td>
</tr>
<tr>
<td>SAP [36]</td>
<td>ResNet-50</td>
<td>46.5</td>
</tr>
<tr>
<td>SFA [59]</td>
<td>ResNet-50</td>
<td>46.7</td>
</tr>
<tr>
<td>DA-DETR</td>
<td>ResNet-50</td>
<td>48.9</td>
</tr>
</tbody>
</table>

Table 4. Experimental results (%) of the scenario Cross-camera Adaptation: KITTI → Cityscapes.

5.2. Implementation Details

In all experiments, we adopt deformable-DETR [81] as the base detector. Since there is only few prior study [59] on transformer-based domain adaptive detection, we modify existing object detectors using Faster R-CNN [7, 24, 36, 51, 64, 79] to the transformer-based domain adaptive detection for fair comparisons. The modification is accomplished by keeping domain adaptation modules unchanged but replacing post-processing modules in Faster R-CNN (e.g., region proposal network, proposal classification module, etc.) by the encoder-decoder module of deformable DETR. In addition, we adopt ResNet-50 [25] (pre-trained on ImageNet [13]) as backbone, and use SGD optimizer [2] with a momentum 0.9 and a weight decay $1e^{-4}$ in all experiments with deformable-DETR.

In all experiments, the weight factor $\lambda$ in Eq. 10 is fixed at 0.1 and the number of split groups $K$ in SMF is fixed at 32. All the experiments are implemented in Pytorch. For evaluation metrics, we report average precision (AP) for each object category and mean average precision (mAP) of all object categories with a threshold of intersection over union (IoU) at 0.5 as in [7, 51, 64].

5.3. Ablation Studies

The proposed CTBlender consists of split-merge fusion (SMF) and scale aggregation fusion (SAF). We first study the two fusion modules to examine how they contribute to the overall unsupervised domain adaptive detection performance. Table 1 shows experimental results over the validation data of Foggy Cityscapes under the adaptation scenario ‘normal weather to foggy weather’.

As Table 1 shows, the Baseline [81] trained using the labeled source data only does not perform well due to domain shifts. The model Direct-align aligns CNN features directly via adversarial learning which improves the Baseline from 28.5% to 38.4% in mAP. The proposed SMF is evaluated under three settings including with Splitting op-
operation, with Shuffling operation and with both. It can be observed that SMF under all three settings outperforms the Direct-align consistently, while the SMF with both Splitting and Shuffling performs the best. In addition, including SAF alone over the Direct-align improves mAP by 3.3%. The incorporation of SMF and SAF achieves the best mAP at 43.5%, demonstrating that SMF and SAF are complementary to each other.

### 5.4. Comparisons with the State-of-the-Art

We evaluate DA-DETR under four domain-shift scenarios: 1) Normal weather to Foggy weather; 2) Synthetic scene to Real scene; 3) Cross-camera Adaptations and 4) Real-world images to Artistic images. In each domain-shift scenario, we compare DA-DETR with a number of state-of-the-art unsupervised domain adaptive methods.

**Normal Weather to Foggy Weather:** We first study the adaptation from normal weather to foggy weather on the task Cityscapes → Foggy cityscapes. As Table 2 shows, DA-DETR outperforms the baseline deformable DETR [81] greatly. It also outperforms the state-of-the-art [59] by 2.2% in mAP. For certain categories such as ‘train’ that can not be well detected by existing methods, DA-DETR achieves the best AP of 37.5. Such experimental results verify that the proposed CTBlender helps to identify both the semantic information and the localization information effectively.

**Synthetic Scene to Real Scene:** Table 3 shows experiments of adaptation from synthetic to real scenes on the task SIM 10k → Cityscapes. We can observe that DA-DETR achieves the best accuracy with a mAP 54.7%, showing that DA-DETR is powerful when there is only one object category ‘car’ in cross-domain detection task.

**Cross-camera Adaptation:** Table 4 shows experiments of cross-camera adaptation over the task KITTI → Cityscapes. We can observe that DA-DETR outperforms the state-of-the-art and improves the baseline model [81] from 39.5% to 48.9% in mAP. These experiments further show that the proposed DA-DETR can well generalize to different domain adaptation tasks.

**Real-world Images to Artistic Images:** We evaluate the adaptation from real-world images to clipart-style images on the task PASCAL VOC → Clipart1k. Table 5 shows experimental results, where DA-DETR achieves the best mAP of 41.3%. In addition, DA-DETR improves the baseline by large margins for certain categories that are not well detected by the baseline model [81] such as bird and sofa. This experiment shows that DA-DETR can handle domain adaptation with multiple categories effectively.

To demonstrate the generalization capability of DA-DETR, we also evaluate it over the tasks PASCAL VOC → Watercolor2k and PASCAL VOC → Comic2k, respectively. As Table 6 shows, DA-DETR outperforms the baseline [81] by large margins, and it also outperforms all state-of-the-art methods over two tasks consistently.

### 5.5. Discussion

**Effectiveness of Split Fusion in CTBlender.** As described in Section 4, the split operation in SMF splits input feature into K groups which helps to encode different semantic information into the fused feature. We examine the effectiveness of the split operation over domain adaptation task Cityscapes → Foggy cityscapes by visualizing the weight generated by each group. We sampled 4 groups from the total 32 groups for each image and highlighted the produced weight over the sample images as shown in Fig. 4. We can observe that SMF captures different foreground regions over different groups, demonstrating that the split operation
Original image  k=8  k=16  k=24  k=32

Figure 4. Visualization of generated weight in SMF for each group. We take two sample images from validation set of Cityscapes, which are shown in the first column. For each image, we sample 4 groups from the total 32 groups and highlight the generated attention over each sample image as shown in columns 2 to 5, respectively. We can observe that the generated weight in different groups detect different foreground regions effectively. $k$ denotes the $k^{th}$ group defined in Section 4.

<table>
<thead>
<tr>
<th>Method</th>
<th>Aligned Features</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deformable-DETR [81]</td>
<td>N.A.</td>
<td>28.5</td>
</tr>
<tr>
<td>Direct-Align [19]</td>
<td>CNN features</td>
<td>38.4</td>
</tr>
<tr>
<td>Direct-Align [19]</td>
<td>Transformer features</td>
<td>38.9</td>
</tr>
<tr>
<td>Direct-Align [19]</td>
<td>CNN features</td>
<td>40.2</td>
</tr>
<tr>
<td>Addition [25]</td>
<td>and Transformer features</td>
<td>42.1</td>
</tr>
<tr>
<td>Multiplication [61]</td>
<td></td>
<td>41.9</td>
</tr>
<tr>
<td>Convolution [72]</td>
<td></td>
<td>41.8</td>
</tr>
<tr>
<td>AFF [11]</td>
<td></td>
<td>42.4</td>
</tr>
<tr>
<td>LS-DeconvNet [9]</td>
<td></td>
<td>42.6</td>
</tr>
<tr>
<td>CTBlender (ours)</td>
<td></td>
<td>43.5</td>
</tr>
</tbody>
</table>

Table 7. Comparing the proposed CTBlender with conventional fusion mechanisms in cross-domain alignment (on the domain adaptive object detection task Cityscapes $\rightarrow$ Foggy Cityscapes).

helps to learn different semantic features effectively.

Analysis of CNN Features and Transformer Features. We study how CNN features and Transformer features affect unsupervised domain adaptation by examining the adaptation performance of the direct alignment of CNN features $f$, the direct alignment of Transformer features $p$ and both, respectively. As shown in Rows 1-4 of Table 7, aligning CNN features and Transformer features simultaneously brings clear further performance improvement over either CNN feature alignment or Transformer feature alignment. This shows that either the localization information in CNN features or the semantic information in Transformer features can facilitate domain adaptation in some degree while these two types of information are complementary for cross-domain alignment.

Comparison with Conventional Fusion Mechanisms. We study how different feature fusion strategies affect the domain adaptation performance by comparing our CTBlender with existing feature fusion strategies [9,11,25,61,72], e.g., fusing CNN features and Transformer features via 1) addition [25], multiplication [61] and convolution [72], and 2) attention-based fusion [9,11]. As shown in the bottom part of Table 7, all fusion strategies improve the Direct-Align baseline clearly, demonstrating the effectiveness of aligning the fused features in UDA. In addition, we can observe that our CTBlender performs the best clearly, largely attributed to its split-merge fusion and scale aggregation fusion designs that fuses CNN features and Transformer features ingeniously. Specifically, the split-merge fusion in CTBlender splits the CNN/Transformer feature which enables to fuse them along spatial and channel dimensions respectively, leading to comprehensive information fusion along both feature dimensions. Besides, the scale aggregation fusion in CTBlender aggregates rich information across multiple image scales, leading to effective cross-domain feature alignment for different scales that facilitates cross-domain detection against large scale variance.

6. Conclusion

This paper presents DA-DETR, an unsupervised domain adaptive detection transformer that introduces information fusion into the DETR framework for effective knowledge transfer from a labeled source domain to an unlabeled target domain. We design a novel CNN-Transformer Blender that fuses the CNN features and Transformer features ingeniously for effective feature alignment and domain adaptation across domains. Extensive experiments over multiple domain adaptation scenarios show that DA-DETR achieves superior performance in unsupervised domain adaptive object detection. Moving forwards, we plan to continue to investigate innovative cross-domain alignment strategies for better domain adaptive transformer detection.

Acknowledgement. This study is supported under the RIE2020 Industry Alignment Fund – Industry Collaboration Projects (IAF-ICP) Funding Initiative, as well as cash and in-kind contribution from the industry partner(s).
References


[76] Yixin Zhang, Zilei Wang, and Yushi Mao. Rpn prototype alignment for domain adaptive object detector. In Proceedings...


