Category Contrast for Unsupervised Domain Adaptation in Visual Tasks
Appendix
A. Theoretical insights of CaCo

A.1. Proof of Proposition 1

**Proposition 1** The category contrastive learning can be modeled as a maximum likelihood (ML) problem optimized via Expectation Maximization (EM).

**Proof:**

Maximum likelihood (ML) was initially proposed to model clustering tasks, and can be optimized by expectation maximization (EM). For our proposed category contrastive learning, the objective is to find the encoder weights $\theta_{f_k}$ that maximizes the log-likelihood function of both labeled data $X_s$ and unlabeled data $X_t$:

$$
\theta_{f_k}^* = \arg \max_{\theta_{f_k}} \sum_{x_s \in X_s} \log p(x_s; \theta_{f_k}) + \sum_{x_t \in X_t} \log p(x_t; \theta_{f_k}).
$$

(1)

As the labeled data are with annotations, the first term of the right-hand side (RHS) in Eq. 1 can be maximized by the supervised learning that minimizes a cross-entropy loss between the predictions of $X_s$ and their annotations $Y_s$:

$$
\arg \min_{\theta_{G}} L_{sup} = \arg \min_{\theta_{f_k}, \theta_h} \sum_{x_s \in X_s, y_s \in Y_s} -y_s \log(h(f_{q}(x_s))),
$$

(2)

where $h$ is the category classifier and the combination of $h$ and $f_q$ forms the visual task model $G = h(f_q(\cdot))$.

Please refer to [8,12,34] for the detailed proofs that minimizing the cross-entropy loss leads to the likelihood maximization.

As the unlabeled data are without annotations, CaCo maximizes the second term by the proposed category contrastive learning. Below please find the proof.

We assume that the unlabeled samples $X_t$ are related to latent variable $\{k_c\}_{c=1}^C$ which denotes the categorical keys of the data. $C$ stands for the number of categories. In this way, we can re-write the second term of the RHS in Eq. 1 as follows:

$$
\theta_{f_k}^* = \arg \max_{\theta_{f_k}} \sum_{x_t \in X_t} \log \sum_{c=1}^C p(x_t, k_c; \theta_{f_k})
$$

(3)

As it is difficult to optimize Eq.3 directly, we utilize a surrogate function to lower-bound the log-likelihood function:

$$
\sum_{x_t \in X_t} \log \sum_{c=1}^C p(x_t, k_c; \theta_{f_k}) = \sum_{x_t \in X_t} \log \sum_{c=1}^C \mathcal{D}(k_c) \frac{p(x_t, k_c; \theta_{f_k})}{\mathcal{D}(k_c)} \\
\geq \sum_{x_t \in X_t} \sum_{c=1}^C \mathcal{D}(k_c) \log \frac{p(x_t, k_c; \theta_{f_k})}{\mathcal{D}(k_c)},
$$

(4)

where $\mathcal{D}(k_c)$ denotes some distribution over $k$’s ($\sum_{c=1}^C \mathcal{D}(k_c) = 1$), and the last step of derivation utilizes Jensen’s inequality [10, 19, 26]. This equality holds if $\frac{p(x_t, k_c; \theta_{f_k})}{\mathcal{D}(k_c)} = \text{Constant}$. Thus, we can get:

$$
\mathcal{D}(k_c) = \frac{p(x_t, k_c; \theta_{f_k})}{\sum_{c=1}^C p(x_t, k_c; \theta_{f_k})} = \frac{p(x_t, k_c; \theta_{f_k})}{p(x_t; \theta_{f_k})} = p(k_c; x_t, \theta_{f_k})
$$

(5)

By ignoring the constant $-\sum_{x_t \in X_t} \sum_{c=1}^C \mathcal{D}(k_c) \log \mathcal{D}(k_c)$ in Eq.4, we are supposed to maximize:

$$
\sum_{x_t \in X_t} \sum_{c=1}^C \mathcal{D}(k_c) \log p(x_t, k_c; \theta_{f_k})
$$

(6)

**Expectation step.** We estimate the posterior probability $p(k_c; x_t, \theta_{f_k})$. For this purpose, we conduct $C$-category pseudo labeling on the key embeddings $k = f_k(x_k)$ ($x_k \in X_s \cup X_t$) that are encoded by the momentum encoder to obtain
category-level keys \( \{k_c\}_{c=1}^C \). The categorical key \( k_c \) is defined as the key \( k \) that belongs to the \( c \)-th semantic category \( (c = \arg \max_i \hat{y}_i^{(c)}) \) and the predicted category label \( \hat{y}_k \) of \( k = f_k(x_k) \) is derived by:

\[
\arg \max_{\hat{y}_k} \sum_{c=1}^C \hat{y}_k^{(c)} \log p(c; k, \theta_k), \text{ s.t. } \hat{y} \in \Delta^C, \forall k
\]  

(7)

where \( h \) is the category classifier that predicts \( C \)-category probabilities for each embedding (e.g., \( k \)), and \( \hat{y} = (\hat{y}^{(1)}, \hat{y}^{(2)}, \ldots, \hat{y}^{(C)}) \) is the predicted category label. To get the pseudo label \( \hat{y}_q \) of the query embedding \( q = f_q(x_t) \) (\( x_t \in X_t \)) encoded by current encoder, we simply repeat above steps by replacing all the notation “\( k \)” with “\( q \)”.

Next, we calculate \( p(k_c; x_t, \theta_{f_q}) = \hat{y}_q \times \hat{y}_{k_c} \), where \( \hat{y}_q \times \hat{y}_{k_c} = 1 \) if both refer to the same category; otherwise, \( \hat{y}_q \times \hat{y}_{k_c} = 0 \).

Maximization step. Now, we are ready to maximize the lower-bound in Eq.6.

\[
\sum_{x_t \in X_t} \sum_{c=1}^C D(k_c) \log p(x_t, k_c; \theta_{f_q}) = \sum_{x_t \in X_t} \sum_{c=1}^C p(k_c; x_t, \theta_{f_q}) \log p(x_t, k_c; \theta_{f_q})
= \sum_{x_t \in X_t} \sum_{c=1}^C (\hat{y}_q \times \hat{y}_{k_c}) \log p(x_t, k_c; \theta_{f_q})
\]  

(8)

We assume a uniform prior over categorical keys. Then, we get:

\[
p(x_t, k_c; \theta_{f_q}) = p(x_t; k_c, \theta_{f_q})p(k_c; \theta_{f_q}) = \frac{1}{C} \cdot p(x_t; k_c, \theta_{f_q}),
\]  

(9)

where we let the prior probability \( p(k_c; \theta_{f_q}) \) for each \( k_c \) as \( 1/C \) as no data is provided.

Under the assumption that the embedding distribution around each categorical key \( k_c \) is an isotropic Gaussian \([3]\), we get:

\[
p(x_t; k_c, \theta_{f_q}) = \exp \left( \frac{-(q - k_+)^2}{2\sigma_+^2} \right) \sum_{c=1}^C \exp \left( \frac{-(q - k_c)^2}{2\sigma_c^2} \right),
\]  

(10)

where \( q = f_q(x_t) \), and \( k_+ \) is defined as the key \( k_c \) that belongs to the same category as \( q \) (i.e., \( \hat{y}_q \times \hat{y}_{k_+} = 1 \)). By applying \( \ell_2 \)-normalization to \( q \) and \( k_+ \), we get \( (q - k)^2 = 2 - 2q \cdot k \). Combining this equation with Eqs.3, 4, 6, 8, 9, 10, we formulate the likelihood maximization as:

\[
\theta_{f_q} = \arg \min_{\theta_{f_q}} \sum_{x_t \in X_t} -\log \frac{\exp(q \cdot k_+/\tau_+)}{\sum_{c=1}^C \exp(q \cdot k_c/\tau_c)},
\]  

(11)

where \( \tau \propto \sigma^2 \) represents the density level of the embedding distribution around a categorical key (e.g., \( k_c \)).

In practice, Eq. 11 can be achieved by minimizing a category contrastive loss:

\[
\arg \min_{\theta_{f_q}} \mathcal{L}_{\text{CatNCE}} = \arg \min_{\theta_{f_q}} \sum_{x_t \in X_t} - \left( \frac{1}{M} \sum_{m=1}^M \log \frac{\sum_{c=1}^C \exp(q \cdot k_m^c/\tau_m^c)(\hat{y}_q \times \hat{y}_{k_m^c})}{\sum_{c=1}^C \exp(q \cdot k_m^c/\tau_m^c)} \right).
\]  

(12)

Please note that Eq. 12 is an instance of Eq. 11. They look different because: 1) Eq. 11 uses \( k_+ \) to denote the positive key instead of using a complex expression to identify the positive key (i.e., \( \sum_{c=1}^C \exp(q \cdot k_m^c/\tau_m^c)(\hat{y}_q \times \hat{y}_{k_m^c}) \)), for the simplicity of theoretic proof; 2) Eq. 11 only shows one group of categorical keys instead of \( M \)-group categorical keys, for the simplicity of theoretic proof.

**A.2. Proof of Proposition 2**

**Proposition 2** The categorical contrastive learning is convergent under certain conditions.

**Proof:**

For the supervised learning on *labeled data*, please refer to \([8, 12, 34]\) for the detailed proofs of the fact that the likelihood maximization by minimizing the cross-entropy loss is convergent under certain conditions.

For the unsupervised learning on *unlabeled data*, please find the convergence proof below.

We let...
It has been illustrated in Section A.1 that the inequality in Eq. 13 holds with equality if \( D \). Cityscapes show that existing unsupervised representation learning methods over the UDA task GTA \( \rightarrow \) 28, 41, 45\], patch ordering \[9, 27\], rotation prediction \[11\], and denoising/context/colorization auto-encoders \[31, 38, 47, 48\].

We compared CaCo with unsupervised representation learning methods over the UDA task. Most existing methods achieve unsupervised representation learning through certain pretext tasks, such as instance contrastive learning \[1, 6, 7, 13, 14, 16, 17, 28, 41, 45\], patch ordering \[9, 27\], rotation prediction \[11\], and denoising/context/colorization auto-encoders \[31, 38, 47, 48\].

The experiments (shown in Table 2) over the UDA task GTA \( \rightarrow \) Cityscapes show that existing unsupervised representation learning does not perform well in the UDA task. The major reason is that these methods were designed to learn instance-discriminative representations without considering semantic priors and domain gaps. CaCo also performs unsupervised learning but works for UDA effectively, largely because it learns category-discriminative yet domain-invariant representations which is essential to various visual UDA tasks.

**B. Discussion**

**B.1. Conceptual comparisons**

We provided conceptual comparisons of different UDA methods in Table 1.

**B.2. Comparisons with existing unsupervised representation learning methods**

The experiments (shown in Table 3) over the UDA segmentation task GTA \( \rightarrow \) Cityscapes show that \( M \) does not affect UDA clearly while it changes from 50 to 150.
Table 1. Conceptual comparisons of different UDA methods. Mec. denotes Mechanisms. AT, IT, ST, and IC denote adversarial training, image translation, self-training, and instance contrast, respectively.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Mec.</th>
<th>Cross-domain adaptation</th>
<th>Intra-domain adaptation</th>
<th>Category aware</th>
<th>Task generalizable</th>
<th>Setup generalizable</th>
<th>Main assumption</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdaptSeg [36]</td>
<td>AT</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>Domain-invariant representations can be learnt via adversarial training (AT) in output space (AdaptSeg).</td>
</tr>
<tr>
<td>CLAN [24]</td>
<td>AT</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>or class joint AT (CLAN and SIM), multi-level AT (SWDA), regularized AT (CRDA), or intra-domain AT (IDA).</td>
</tr>
<tr>
<td>AdvEnt [39]</td>
<td>AT</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>PatAlign [37]</td>
<td>AT</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>IDA [29]</td>
<td>AT</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>CrCDA [18]</td>
<td>AT</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>SIM [40]</td>
<td>AT</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>SWDA [32]</td>
<td>AT</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>CRDA [42]</td>
<td>AT</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>TIR [21]</td>
<td>IT</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>Domain-invariant representations can be learnt via image translation (TIR), spectrum swapping (FDA).</td>
</tr>
<tr>
<td>FDA [44]</td>
<td>IT</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>CBST [51]</td>
<td>ST</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
<td>Category-discriminative representation can be learnt via self-training (CBST), regularized ST (CRST).</td>
</tr>
<tr>
<td>CRST [50]</td>
<td>ST</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>CaCo (ours)</td>
<td>IC</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Category-discriminative yet domain-invariant representation can be learnt via instance contrast (IC).</td>
</tr>
</tbody>
</table>

Table 2. Comparisons with existing unsupervised representation learning methods: For the semantic segmentation over GTA → Cityscapes adaptation, CaCo performs the best consistently by large margins.

<table>
<thead>
<tr>
<th>Method</th>
<th>mIoU gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline [15]</td>
<td>36.6</td>
</tr>
<tr>
<td>Jigsaw [27]</td>
<td>38.5 (+1.9)</td>
</tr>
<tr>
<td>Rotation [11]</td>
<td>37.0 (+0.4)</td>
</tr>
<tr>
<td>Colorization [47]</td>
<td>38.7 (+2.1)</td>
</tr>
<tr>
<td>SimCLR [6]</td>
<td>38.4 (+1.8)</td>
</tr>
<tr>
<td>InstDisc [41]</td>
<td>38.0 (+1.4)</td>
</tr>
<tr>
<td>MoCo [14]</td>
<td>38.9 (+2.3)</td>
</tr>
<tr>
<td><strong>CaCo</strong></td>
<td><strong>49.2 (+12.6)</strong></td>
</tr>
</tbody>
</table>

Table 3. The length of categorical dictionary (parameter $M$) affects unsupervised domain adaptation (evaluated over semantic segmentation on GTA → Cityscapes adaptation).

<table>
<thead>
<tr>
<th>Method</th>
<th>M (the length of categorical dictionary)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50</td>
</tr>
<tr>
<td>CaCo</td>
<td>48.9</td>
</tr>
</tbody>
</table>

B.4. Generalization across different learning setups

We studied the scalability of the proposed CaCo from the view of learning setups. Specifically, we evaluated CaCo over a variety of tasks that involve unlabeled data learning and certain semantic priors such as unsupervised model adaptation, partial-set domain adaptation and open-set domain adaptation. Experiments (in Tables 4-6) show that CaCo achieve competitive performance consistently across all the tasks.
Unsupervised model adaptation | mIoU gain
--- | ---
Baseline [15] | 36.6 N.A.
UR [35] | 45.1 +8.5
SFDA [23] | 45.8 +9.2
CaCo | 47.6 +11.0

Table 4. Comparison on unsupervised model adaptation (UMA) over GTA5 → Cityscapes adaptation: For semantic segmentation, CaCo achieves competitive performance as compared with state-of-the-art UMA methods. (Compared with UDA, UMA dose not use labeled source data during adaptation.)

<table>
<thead>
<tr>
<th>Partial-set DA</th>
<th>A→C</th>
<th>A→P</th>
<th>A→R</th>
<th>C→A</th>
<th>C→P</th>
<th>C→R</th>
<th>P→A</th>
<th>P→C</th>
<th>P→R</th>
<th>R→A</th>
<th>R→C</th>
<th>R→P</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet-50 [15]</td>
<td>46.3</td>
<td>67.5</td>
<td>75.9</td>
<td>59.1</td>
<td>59.9</td>
<td>62.7</td>
<td>58.2</td>
<td>41.8</td>
<td>74.9</td>
<td>67.4</td>
<td>48.2</td>
<td>74.2</td>
<td>61.3</td>
</tr>
<tr>
<td>IWAN [46]</td>
<td>53.9</td>
<td>54.5</td>
<td>78.1</td>
<td>61.3</td>
<td>48.0</td>
<td>63.3</td>
<td>54.2</td>
<td>52.0</td>
<td>81.3</td>
<td>76.5</td>
<td>56.8</td>
<td>82.9</td>
<td>63.6</td>
</tr>
<tr>
<td>SAN [4]</td>
<td>44.4</td>
<td>68.7</td>
<td>74.6</td>
<td>67.5</td>
<td>65.0</td>
<td>77.8</td>
<td>59.8</td>
<td>44.7</td>
<td>80.1</td>
<td>72.2</td>
<td>50.2</td>
<td>78.7</td>
<td>65.3</td>
</tr>
<tr>
<td>ETN [5]</td>
<td>59.2</td>
<td>77.0</td>
<td>79.5</td>
<td>62.9</td>
<td>65.7</td>
<td>75.0</td>
<td>68.3</td>
<td>55.4</td>
<td>84.4</td>
<td>75.7</td>
<td>57.7</td>
<td>84.5</td>
<td>70.5</td>
</tr>
<tr>
<td>SAFN [43]</td>
<td>58.9</td>
<td>76.3</td>
<td>81.4</td>
<td>70.4</td>
<td>73.0</td>
<td>77.8</td>
<td>72.4</td>
<td>55.3</td>
<td>80.4</td>
<td>75.8</td>
<td>60.4</td>
<td>79.9</td>
<td>71.8</td>
</tr>
<tr>
<td>CaCo</td>
<td>61.2</td>
<td>83.7</td>
<td>90.5</td>
<td>73.9</td>
<td>75.4</td>
<td>81.5</td>
<td>76.7</td>
<td>61.3</td>
<td>89.4</td>
<td>80.5</td>
<td>66.1</td>
<td>86.9</td>
<td>77.3</td>
</tr>
</tbody>
</table>

Table 5. Comparison on partial-set UDA (PS-UDA) over Office-Home: For image classification, CaCo achieves competitive performance as compared with state-of-the-art PS-UDA methods. (In PS-UDA setting, source and target domains do not share a completely same label space.)

<table>
<thead>
<tr>
<th>Open-set DA</th>
<th>A→C</th>
<th>A→P</th>
<th>A→R</th>
<th>C→A</th>
<th>C→P</th>
<th>C→R</th>
<th>P→A</th>
<th>P→C</th>
<th>P→R</th>
<th>R→A</th>
<th>R→C</th>
<th>R→P</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet [15]</td>
<td>36.3</td>
<td>54.8</td>
<td>69.1</td>
<td>33.8</td>
<td>44.4</td>
<td>49.2</td>
<td>36.8</td>
<td>29.2</td>
<td>56.8</td>
<td>51.4</td>
<td>35.1</td>
<td>62.3</td>
<td>46.6</td>
</tr>
<tr>
<td>ATI-λ [30]</td>
<td>55.2</td>
<td>52.6</td>
<td>53.5</td>
<td>69.1</td>
<td>63.5</td>
<td>74.1</td>
<td>61.7</td>
<td>64.5</td>
<td>70.7</td>
<td>79.2</td>
<td>72.9</td>
<td>75.8</td>
<td>66.1</td>
</tr>
<tr>
<td>OSBP [33]</td>
<td>56.7</td>
<td>51.5</td>
<td>49.2</td>
<td>67.5</td>
<td>65.5</td>
<td>74.0</td>
<td>62.5</td>
<td>64.8</td>
<td>69.3</td>
<td>80.6</td>
<td>74.7</td>
<td>71.5</td>
<td>65.7</td>
</tr>
<tr>
<td>OpenMax [2]</td>
<td>56.5</td>
<td>52.9</td>
<td>53.7</td>
<td>69.1</td>
<td>64.8</td>
<td>74.5</td>
<td>64.1</td>
<td>64.0</td>
<td>71.2</td>
<td>80.3</td>
<td>73.0</td>
<td>76.9</td>
<td>66.7</td>
</tr>
<tr>
<td>STA [22]</td>
<td>58.1</td>
<td>53.1</td>
<td>54.4</td>
<td>71.6</td>
<td>69.3</td>
<td>81.9</td>
<td>63.4</td>
<td>65.2</td>
<td>74.9</td>
<td>85.0</td>
<td>75.8</td>
<td>80.8</td>
<td>69.5</td>
</tr>
<tr>
<td>CaCo</td>
<td>63.5</td>
<td>78.7</td>
<td>83.8</td>
<td>61.1</td>
<td>74.0</td>
<td>79.6</td>
<td>64.2</td>
<td>58.2</td>
<td>82.3</td>
<td>68.8</td>
<td>62.9</td>
<td>81.7</td>
<td>71.6</td>
</tr>
</tbody>
</table>

Table 6. Comparison on open-set UDA (OS-UDA) over Office-Home: For image classification, CaCo achieves competitive performance as compared with state-of-the-art OS-UDA methods. (In OS-UDA setting, source and target domains do not share a completely same label space.)

B.5. Category-aware dictionary

What about assigning all keys with the same temperature? In Eq.5 in the submitted manuscript, we assigned different temperatures to different keys as their predicted labels have different uncertainties (labeled source samples also have corresponding prediction uncertainties even if they are labeled). In this section, we conduct experiments to show how this adaptive temperature design affects the performance. Table 7 shows that CaCo (with adaptive temperature) outperforms its uncertainty-independent version (with fixed temperature). The reason is that CoCo (with adaptive temperature) alleviates the negative effects from the wrongly pseudo-labeled keys, i.e., suppressing the effect of keys with high uncertainty (about the category pseudo labeling). In another word, CaCo encourages to employ well-learnt embeddings (instead of using under-learnt embeddings) as keys in representation learning.

What about using two individual dictionaries (for source and target data) instead of a single domain-mixed dictionary? In the description of categorical dictionary in the submitted manuscript, the dictionary keys are domain-mixed, i.e., evenly sampled from source and target domains. In this section, we conduct experiments to show how domain-mixing design affects the performance. Table 8 shows that CaCo (with domain-mixed dictionary) outperforms its vanilla version (with two individual dictionaries) clearly. The reason is that CaCo (with domain-mixed dictionary) enables information communication across domains (like in Shufflenet [49]) that helps mitigate inter-domain discrepancy. For instance, with a group
### Table 7. Fixed or adaptive temperature in the proposed Category Contrast: CaCo with adaptive temperature clearly outperforms its uncertainty-independent version with fixed temperature, evaluated over UDA-based semantic segmentation task GTA → Cityscapes.

<table>
<thead>
<tr>
<th>Fixed or Adaptive temperature</th>
<th>mIoU</th>
<th>gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline [15]</td>
<td>36.6</td>
<td>N.A.</td>
</tr>
<tr>
<td>CaCo (with fixed temperature)</td>
<td>47.9</td>
<td>+11.3</td>
</tr>
<tr>
<td>CaCo (with adaptive temperature)</td>
<td>49.2</td>
<td>+12.6</td>
</tr>
</tbody>
</table>

of domain-mixed keys that contains a “car” key encoded from target domain and $C - 1$ keys (the rest categories) encoded from source domain, a target “car” query could be pulled closer to its positive target-domain key and pushed away from its negative source-domain keys, which makes the learning process more efficient and effective.

### Table 8. Domain-mixed or individual dictionary: CaCo with domain-mixed dictionary clearly outperforms its vanilla version with two individual dictionaries, evaluated over the UDA-based semantic segmentation task GTA → Cityscapes.

<table>
<thead>
<tr>
<th>Domain-mixed or individual dictionary</th>
<th>mIoU</th>
<th>gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline [15]</td>
<td>36.6</td>
<td>N.A.</td>
</tr>
<tr>
<td>CaCo-S</td>
<td>46.8</td>
<td>+10.2</td>
</tr>
<tr>
<td>CaCo-T</td>
<td>48.3</td>
<td>+11.7</td>
</tr>
<tr>
<td>CaCo (with two individual dictionaries)</td>
<td>48.5</td>
<td>+11.9</td>
</tr>
<tr>
<td>CaCo (with domain-mixed dictionary)</td>
<td>49.2</td>
<td>+12.6</td>
</tr>
</tbody>
</table>

### What about sampling queries from source-domain (i.e., training networks with an extra supervised source-domain contrastive loss)?

As described in Fig. 1 and Eq. 5 in the submitted manuscript, CaCo samples queries from target domain only in contrastive learning of its unlabelled samples. This strategy is intuitive and reasonable as the annotated source-domain data can be well learnt with supervised losses without bothering with an extra contrastive loss. Nevertheless, [20] shows that incorporating an extra supervised contrastive loss could further improve the supervised learning of source-domain data. We thus conduct new experiments to explore whether training networks with an extra source-domain contrastive loss could further improve domain adaptation. As Table 9 shows, the baseline with a supervised source-domain contrast marginally outperforms its vanilla version, which indicates that the supervised contrastive learning could improve the generalization of the baseline model. We conjecture that further including an extra supervised source-domain contrastive loss may distract the network from learning target-domain data which leads to the slight degradation.

### Table 9. Sampling queries from source-domain? As described in Fig. 1 and Eq. 5 in the submitted manuscript, CaCo samples queries from target domain only in contrastive learning of its unlabelled samples. This strategy is intuitive and reasonable as the annotated source-domain data can be well learnt with supervised losses without bothering with an extra contrastive loss. Nevertheless, [20] shows that incorporating an extra supervised contrastive loss could further improve the supervised learning of source-domain data. We thus conduct new experiments to explore whether training networks with an extra source-domain contrastive loss could further improve domain adaptation. As Table 9 shows, the baseline with a supervised source-domain contrast marginally outperforms its vanilla version, which indicates that the supervised contrastive learning could improve the generalization of the baseline model. However, incorporating supervised source-domain contrast into CaCo slightly degrades the domain adaptation. We conjecture that further including an extra source-domain contrastive loss may distract the network from learning target-domain data which leads to the slight degradation.

<table>
<thead>
<tr>
<th>Sampling queries from source-domain?</th>
<th>mIoU</th>
<th>gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline [15]</td>
<td>36.6</td>
<td>N.A.</td>
</tr>
<tr>
<td>+supervised source-domain contrast</td>
<td>38.2</td>
<td>+1.6</td>
</tr>
<tr>
<td>CaCo</td>
<td>49.2</td>
<td>+12.6</td>
</tr>
<tr>
<td>+supervised source-domain contrast</td>
<td>49.0</td>
<td>+12.4</td>
</tr>
</tbody>
</table>

### What about updating the dictionary by memory bank [41] or current mini-batch [6]?

In this paper, we use a momentum encoder to encode the keys and update the dictionary. In this section, we conduct experiments to show how dictionary-update strategy affects the performance. Table 10 shows that the performance is not sensitive to dictionary-update...
strategies. The reason is that CaCo improves domain adaptation largely by reducing domain gaps and enhancing category discrimination whereas dictionary-update strategy has little effect on these two factors.

<table>
<thead>
<tr>
<th>Dictionary updating strategies</th>
<th>mIoU</th>
<th>gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline [15]</td>
<td>36.6</td>
<td>N.A.</td>
</tr>
<tr>
<td>CaCo (end-to-end updating [6])</td>
<td>49.0</td>
<td>+12.4</td>
</tr>
<tr>
<td>CaCo (memory bank updating [41])</td>
<td>48.9</td>
<td>+12.3</td>
</tr>
<tr>
<td>CaCo (momentum updating [14])</td>
<td>49.2</td>
<td>+12.6</td>
</tr>
</tbody>
</table>

Table 10. Dictionary updating strategies: Different dictionary updating strategies (i.e. end-to-end updating [6], memory bank updating [41], and momentum updating [14]) have little effect on CaCo’s performance, evaluated over the UDA-based semantic segmentation task GTA → Cityscapes.

Figure 1. The t-SNE [25] visualization of feature distribution for target images on task GTA → Cityscapes: Each colour represents one semantic category of image pixels with a digit showing the category center. \(a\), \(b\) and \(c\) on the top of each graph are intra-category variance, inter-category distance and cross-domain distance of the corresponding feature distribution. The proposed CaCo greatly outperforms “Baseline”, “ADVENT” (adversarial training based), “CRST” (self-training based) and “FDA” (image translation based) qualitatively and quantitatively. Please note that we did not include the source feature distribution for simplicity and clarity.

B.6. Feature distribution analysis

Feature distribution visualization. We provide the t-SNE [25] visualization of feature distribution for target images on the GTA → Cityscapes task. To illustrate the unique features of the proposed “CaCo”, we compare it with the “Baseline” and three typical UDA approaches, \(i.e., \) “ADVENT” [39] (adversarial training based), “CRST” [50] (self-training based) and “FDA” [44] (image translation based) as shown in Fig.1. In addition, we evaluate the learnt features by using three metrics including intra-category variance, inter-category distance, and cross-domain distance which are labelled by \(a\), \(b\), and
It can be observed that the feature distribution of the Baseline model is messy and not category-discriminative (i.e., $a = 544.69$ and $b = 43.27$) due to the large distribution gaps across domains (i.e., $c = 20.12$). ADVENT generates features with less cross-domain discrepancy (i.e., $c = 15.65$) as it employs adversarial training to reduce domain gaps. However, adversarial training is category-unaware, which leads to sub-optimal category discrimination (i.e., $a = 376.69$ and $b = 43.65$) for visual recognition which requires category-discriminative features. FDA adopts image translation and generates features (i.e., $a = 374.09$, $b = 45.70$ and $c = 14.76$) in a similar manner to ADVENT. In addition, CRST generates category-discriminative features (i.e., $a = 347.21$ and $b = 48.23$) as it employs category-wise self-training. On the other hand, self-training is less effective on cross-domain gaps reduction, leading to sub-optimal cross-domain distance ($c = 16.92$) for domain adaptation. The proposed CaCo learns category-discriminative yet domain-invariant features (i.e., $a = 254.78$, $b = 50.79$ and $c = 12.86$) as it employs a category-aware and domain-mixed dictionary for categorical contrastive learning. The visualization verifies the first and second claims as mentioned in the fifth paragraph of the Introduction.

**Category imbalance mitigation.** As shown in Fig. 1, it can be also observed that the feature distribution generated by the proposed CaCo is more category-balanced. For example, the features of both dominant and less-dominant categories are well learnt and separated (e.g., category 0-2), but the features of less-dominant categories are poorly learnt and separated (e.g., category 5&7, 8&10 in “Baseline”, category 2&5 in “ADVENT”, category 5&7&12, 2&11 in “FDA” and category 5&11&12 in “CRST”).

**References**


