# Open Compound Domain Adaptation

Ziwei Liu[1*] Zhongqi Miao[2*] Xingang Pan[1] Xiaohang Zhan[1]
Dahua Lin[1] Stella X. Yu[2] Boqing Gong[3]
[1] The Chinese University of Hong Kong [2] UC Berkeley / ICSI [3] Google Inc.
https://liuziwei7.github.io/projects/CompoundDomain.html

## Abstract

*A typical domain adaptation approach is to adapt models trained on the annotated data in a source domain (e.g., sunny weather) for achieving high performance on the test data in a target domain (e.g., rainy weather). Whether the target contains a single homogeneous domain or multiple heterogeneous domains, existing works always assume that there exist clear distinctions between the domains, which is often not true in practice (e.g., changes in weather). We study an open compound domain adaptation (OCDA) problem, in which the target is a compound of multiple homogeneous domains without domain labels, reflecting realistic data collection from mixed and novel situations. We propose a new approach based on two technical insights into OCDA: 1) a curriculum domain adaptation strategy to bootstrap generalization across domains in a data-driven self-organizing fashion and 2) a memory module to increase the model's agility towards novel domains. Our experiments on digit classification, facial expression recognition, semantic segmentation, and reinforcement learning demonstrate the effectiveness of our approach.*

## 1. Introduction

Supervised learning can achieve competitive performance for a visual task when the test data is drawn from the same underlying distribution as the training data. This assumption, unfortunately, often does not hold in reality, *e.g.*, the test data may contain the same class of objects as the training data but different backgrounds, poses, and appearances [41, 46].

The goal of domain adaptation is to adapt the model learned on the training data to the test data of a different distribution [41, 34, 12]. Such a distributional gap is often formulated as a shift between discrete concepts of well defined data domains, *e.g.*, images collected in sunny weather versus those in rainy weather. Though domain generalization [24, 22] and latent domain adaptation [16, 11] have attempted to tackle complex target domains, most existing works usually assume that there is a known clear distinction between domains [12, 8, 48, 30, 42].

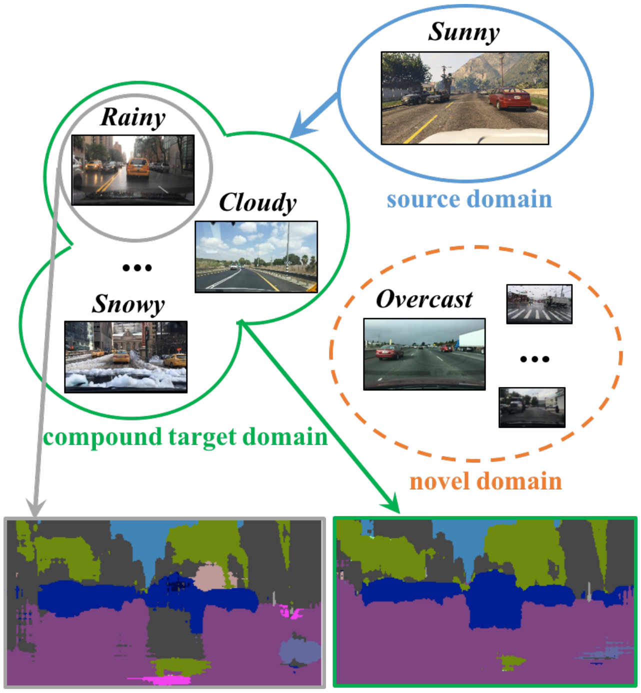


Figure 1: **Open compound domain adaptation.** Unlike existing domain adaptation which assumes clear distinctions between discrete domains (cf. the examples in gray frames), our compound target domain is a combination of multiple traditionally homogeneous domains without any domain labels. We also allow novel domains to show up at the inference time.

Such a known and clear distinction between domains is hard to define in practice, *e.g.*, test images could be collected in mixed, continually varying, and sometimes never seen weather conditions. With numerous factors jointly contributing to data variance, it becomes implausible to separate data into discrete domains.

We propose to study *open compound domain adaptation* (OCDA), a continuous and more realistic setting for domain

[*] Equal contribution.

Table 1: **Comparison of domain adaptation settings.** Domain Labels tell to which domain each instance belongs. Open Classes refer to novel classes showing up during testing but not training. Open Domains are the domains of which no instances are seen during training.

| Domain Adaptation Setting | # Target Domains | Domain Labels | Open Classes | Open Domains |
|---|---|---|---|---|
| Unsupervised Domain Adaptation | single | known | × | × |
| Multi-Target Domain Adaptation | multiple | known | × | × |
| Open/Partial Set Domain Adaptation | single | known | ✓ | × |
| **Open Compound Domain Adaptation** | **multiple** | **unknown** | × | ✓ |

adaptation (cf. Figure 1 and Table 1). The task is to learn a model from labeled *source domain* data and adapt it to unlabeled *compound target domain* data which could differ from the source domain on various factors. Our target domain can be regarded as a combination of multiple traditionally homogeneous domains where each is distinctive on one or two major factors, and yet none of the domain labels are given. For example, the five well-known datasets on digits recognition (SVHN [33], MNIST [21], MNIST-M [7], USPS [19], and SynNum [7]) mainly differ from each other by the backgrounds and text fonts. It is not necessarily the best practice, and not feasible under some scenarios, to consider them as distinct domains. Instead, our compound target domain pools them together. Furthermore, at the inference stage, OCDA tests the model not only in the compound target domain but also in open domains that have previously unseen during training.

In our OCDA setting, the target domain no longer has a predominantly uni-modal distribution, posing challenges to existing domain adaptation methods. We propose a novel approach based on two technical insights into OCDA: 1) a curriculum domain adaptation strategy to bootstrap generalization across domain distinction in a data-driven self-organizing fashion and 2) a memory module to increase the model's agility towards novel domains.

Unlike existing curriculum adaptation methods [56, 6, 29, 25, 58, 57] that rely on some holistic measure of instance difficulty, we schedule the learning of unlabeled instances in the compound target domain according to their *individual gaps* to the labeled source domain, so that we solve an incrementally harder domain adaptation problem till we cover the entire target domain.

Specifically, we first train a neural network to 1) discriminate between classes in the labeled source domain and to 2) capture domain invariance from the easy target instances which differ the least from labeled source domain data. Once the network can no longer differentiate between the source domain and the easy target domain data, we feed the network harder target instances, which are further away from the source domain. The network learns to remain discriminative to the classification task and yet grow more robust to the entire compound target domain.

Technically, we must address the challenge of characterizing each instance's gap to the source domain. We first extract domain-specific feature representations from the data and then rank the target instances according to their distances to the source domain in that feature space, assuming that such features do not contribute to and even distract the network from learning discriminative features for classification. We use a class-confusion loss to distill the domain-specific factors and formulate it as a conventional cross-entropy loss with a randomized class label twist.

Our second technical insight is to prepare our model for open domains during inference with a memory module that effectively augments the representations of an input for classification. Intuitively, if the input is close enough to the source domain, the feature extracted from itself can most likely already result in accurate classification. Otherwise, the input-activated memory features can step in and play a more important role. Consequently, this memory-augmented network is more agile at handling open domains than its vanilla counterpart.

To summarize, we make the following contributions. **1)** We extend the traditional discrete domain adaptation to OCDA, a more realistic continuous domain adaptation setting. **2)** We develop an OCDA solution with two key technical insights: instance-specific curriculum domain adaptation for handling the target of mixed domains and memory augmented features for handling open domains. **3)** We design several benchmarks on classification, recognition, segmentation, and reinforcement learning, and conduct comprehensive experiments to evaluate our approach under the OCDA setting.

## 2. Related Works

We review literature according to Table 1.

**Unsupervised Domain Adaptation.** The goal is to retain recognition accuracies in new domains without ground truth annotations [41, 46, 49, 38]. Representative techniques include latent distribution alignment [12], backpropagation [7], gradient reversal [8], adversarial discrimination [48], joint maximum mean discrepancy [30], cycle consistency [17] and maximum classifier discrepancy [42]. While their results are promising, this traditional domain adaptation setting focuses on "one source domain, one target domain", and cannot deal with more complicated scenarios where multiple target domains are present.

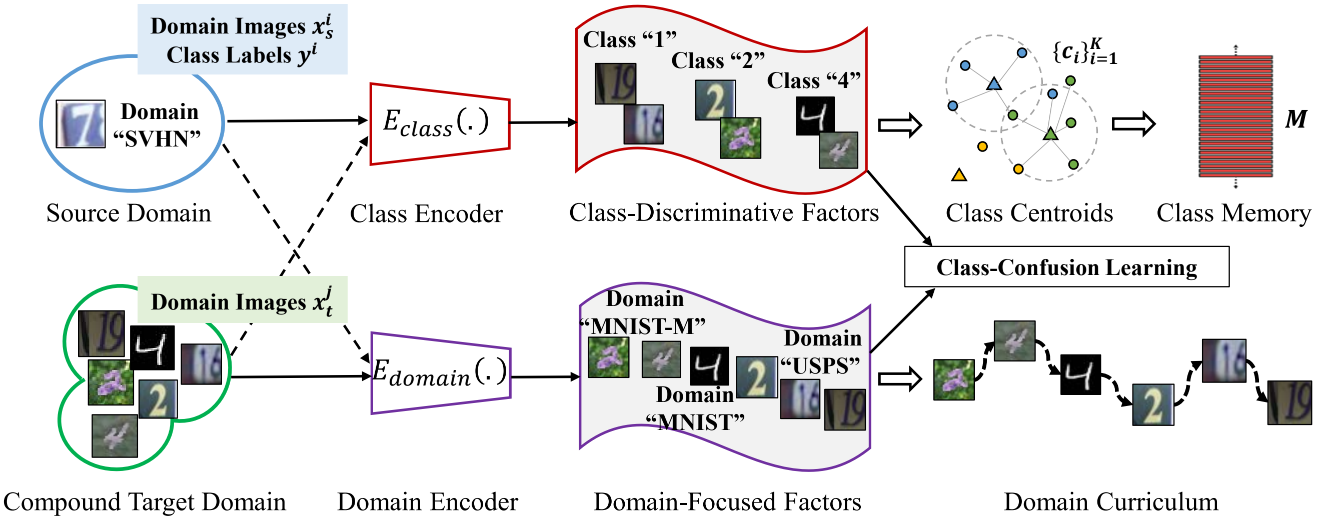


Figure 2: **Overview of disentangling domain characteristics and curriculum domain adaptation.** We separate characteristics specific to domains from those discriminative between classes. It is achieved by a class-confusion algorithm in an unsupervised manner. The teased out domain feature is used to construct a curriculum for domain-robust learning.

**Latent & Multi-Target Domain Adaptation.** The goal is to extend unsupervised domain adaptation to latent [16, 51, 32] or multiple [11, 9, 54] or continuous [2, 13, 31, 50] target domains, when only the source domain has class labels. These methods usually assume clear domain distinction or require domain labels (*e.g.* test instance $i$ belongs to the target domain $j$), but this assumption rarely holds in the real-world scenario. Here we take one step further towards compound domain adaptation, where both category labels and domain labels in the test set are unavailable.

**Open/Partial Set Domain Adaptation.** Another route of research aims to tackle the category sharing/unsharing issues between source and target domain, namely open set [37, 43] and partial set [55, 3] domain adaptation. They assume that the target domain contains either 1) new categories that don't appear in source domain; or 2) only a subset of categories that appear in source domain. Both settings concern the "openness" of categories. Instead, here we investigate the "openness" of domains, *i.e.* there are novel domains existing that are absent in the training phase.

**Domain Generalized/Agnostic Learning.** Domain generalization [52, 23, 22] and domain agnostic learning [39, 5] aim to learn universal representations that can be applied in a domain-invariant manner. Since these methods focus on learning semantic representations that are invariant to the domain shift, they largely neglect the latent structures inside the target domains. In this work, we explicitly model the latent structures inside the compound target domain by leveraging the learned domain-focused factors for curriculum scheduling and dynamic adaptation.

## 3. Our Approach to OCDA

Figures 2 and 3 present our overall workflows. There are three major components: 1) disentangling domain characteristics with only class labels in the source domain, 2) scheduling data for curriculum domain adaptation, and 3) a memory module for handling new domains.

### 3.1. Disentangling Domain Characteristics

We separate characteristics specific to domains from those discriminative between classes. They allow us to construct a curriculum for increment domain adaptation.

We first train a neural network classifier using the labeled source domain data $\{x^i, y^i\}_i$. Let $E_{class}(\cdot)$ denote the encoder up to the second-to-the-last layer and $\Phi(E_{class}(\cdot))$ the classifier. The encoder captures primarily the class-discriminative representation of the data.

We assume that all the factors not covered by this class-discriminative encoder reflect domain characteristics. They can be extracted by another encoder $E_{domain}(\cdot)$ that satisfies two properties: **1)** Completeness: $Decoder(E_{class}(x), E_{domain}(x)) \approx x$, i.e., the outputs of the two encoders shall provide sufficient information for a decoder to reconstruct the input, and **2)** Orthogonality: the domain encoder $E_{domain}(x)$ shall have little mutual information with the class encoder $E_{class}(x)$. We leave the algorithmic details for meeting the first property to the appendices as they are not our novelty.

For the orthogonality between $E_{domain}(x)$ and $E_{class}(x)$, we propose a **class-confusion algorithm**, which

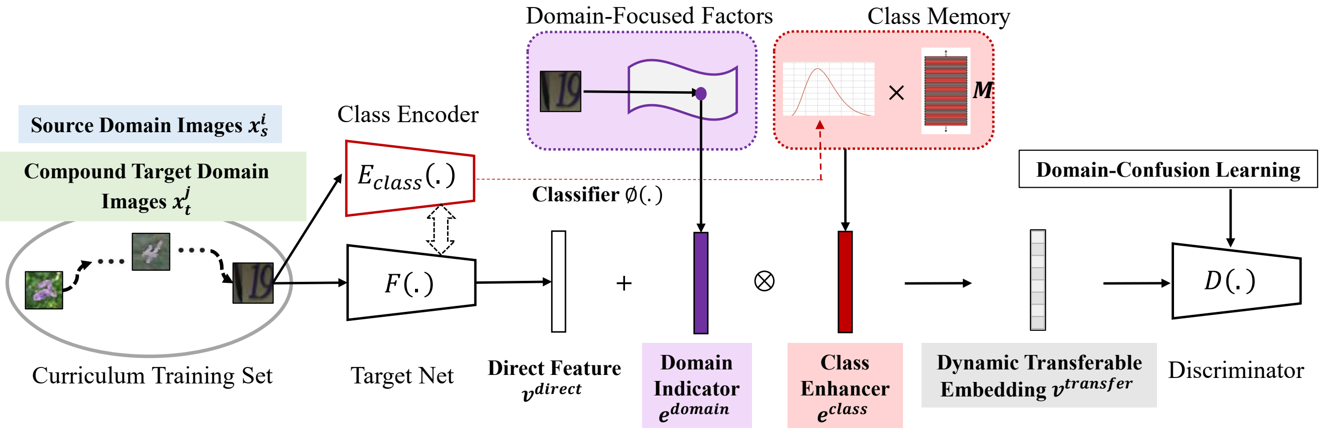

Domain-Focused Factors
Class Memory
M
Source Domain Images $x_s^i$
Class Encoder
$E_{class}(.)$
Compound Target Domain Images $x_t^j$
Classifier ∅(.)
Domain-Confusion Learning
$F(.)$
+
⊗
$D(.)$
Curriculum Training Set
Target Net
Direct Feature $v^{direct}$
Domain Indicator $e^{domain}$
Class Enhancer $e^{class}$
Dynamic Transferable Embedding $v^{transfer}$
Discriminator

Figure 3: **Overview of the memory-enhanced deep neural network.** We enhance our network with a memory module that facilitates knowledge transfer from the source domain to target domain instances, so that the network can dynamically balance the input information and the memory-transferred knowledge for more agility towards previously unseen domains.

alternates between the two sub-problems below:

$$\min_{E_{domain}} \quad -\sum_i z^i_{random} \log D(E_{domain}(x^i)), \qquad (1)$$

$$\min_{D} \quad -\sum_i y^i \log D(E_{domain}(x^i)), \qquad (2)$$

where superscript $i$ is the instance index, and $D(\cdot)$ is a discriminator the domain-encoder $E_{domain}(\cdot)$ tries to confuse. We first train the discriminator $D(\cdot)$ with the labeled data in the source domain. For the data in the target domain, we assign them pseudo-labels by the classifier $\Phi(E_{class}(\cdot))$ we have trained earlier. The learned domain encoder $E_{domain}(\cdot)$ is class-confusing due to $z^i_{random}$, a random label uniformly chosen in the label space. As the classifier $D(\cdot)$ is trained, the first sub-problem essentially learns the domain-encoder such that it classifies the input $x^i$ into a random class $z^i_{random}$. Algorithm 2 details our domain disentanglement process.

Figure 4 (a) and (b) visualize the examples embedded by the class encoder $E_{class}(\cdot)$ and domain encoder $E_{domain}(\cdot)$, respectively. The class encoder places instances in the same class in a cluster, while the domain encoder places instances according to their common appearances, regardless of their classes.

## 3.2. Curriculum Domain Adaptation

We rank all the instances in the compound target domain according to their distances to the source domain, to be used for curriculum domain adaptation [56]. We compute the *domain gap* between a target instance $x_t$ and the source domain $\{x_s^m\}$ as their mean distance in the domain feature space: $\mathtt{mean}_m(\|E_{domain}(x_t) - E_{domain}(x_s^m)\|_2)$.

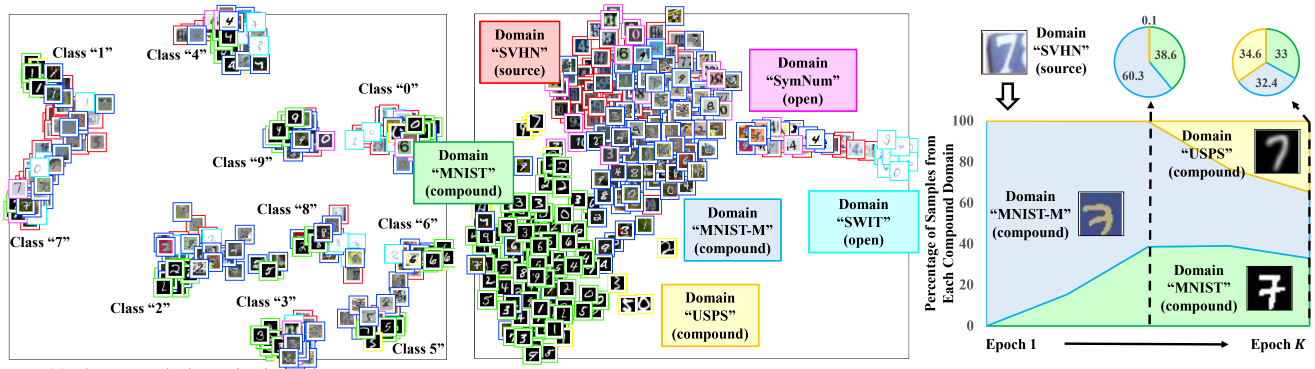
We train the network in stages, a few epochs at a time, gradually recruiting more instances that are increasingly far from the source domain. At each stage of the curriculum learning, we minimize two losses: One is the cross-entropy loss defined over the labeled source domain, and the other is the domain-confusion loss [48] computed between the source domain and the currently covered target instances. Figure 4 (c) illustrates a curriculum in our experiments.

**Algorithm 1** Domain Disentanglement.

```
Input: The class encoder $E_{class}(\cdot)$ and classifier $\Phi$ have been trained using source-domain data, $Deccoder(\cdot)$: the decoder, $C$: the number of classes, $\gamma$: a constant.
for k iterations do
    Sample mini-batch $\{x^i\}$.
    Compute pseudo labels $y^i_{pseudo} \leftarrow \Phi\left(E_{class}\left(x^i\right)\right)$.
    Update the discriminator $D$.
    Prepare random labels $z^i_{random} \sim uniform\{0, 1, ..., C-1\}$.
    Compute adversarial loss: $L_{adv} \leftarrow \sum_i -z^i_{random} \log\left(D\left(E_{domain}(x^i)\right)\right)$.
    Compute reconstruction loss: $L_{rec} \leftarrow \sum_i \|Decoder\left(E_{class}\left(x^i\right), E_{domain}\left(x^i\right)\right) - x^i\|_2$.
    Update the domain encoder $E_{domain}$ with: $\nabla_{\theta_{E_{domain}}}\left(L_{adv} + \gamma L_{rec}\right)$.
end for
```

## 3.3. Memory Module for Open Domains

Existing domain adaptation methods often use the features $v_{direct}$ extracted directly from the input for adaptation. When the input comes from a new domain that significantly differs from the seen domains during training, this representation becomes inadequate and could fool the classifier. We propose a memory module to enhance our model; It allows knowledge transfer from the source domain so that the network can dynamically balance the input-conveyed information and the memory-transferred knowledge for more classification agility towards previously unseen domains.



(a) The Learned Class-Discriminative Factors

(b) The Learned Domain-Focused Factors

(c) The Learned Curriculum along Epoches

Figure 4: **t-SNE Visualization** of our (a) class-discriminative features, (b) domain features, and (c) curriculum. Our framework disentangles the mixed-domain data into class-discriminative factors and domain-focused factors. We use the domain-focused factors to construct a learning curriculum for domain adaptation.

**Class Memory $M$.** We design a memory module $M$ to store the class information from the source domain. Inspired by [45, 36, 28] on prototype analysis, we also use class centroids $\{c_k\}_{k=1}^{K}$ to construct our memory $M$, where $K$ is the number of object classes.

**Enhancer $v_{enhance}$.** For each input instance, we build an enhancer to augment its direct representation $v_{direct}$ with knowledge in the memory about the source domain: $v_{enhance} = (\Psi(v_{direct}))^T M = \sum_{k=1}^{K} \psi_k c_k$, where $\Psi(\cdot)$ is a softmax function. We add this enhancer to the direct representation $v_{direct}$, weighted by a domain indicator.

**Domain Indicator $e_{domain}$.** With open domains, the network must dynamically calibrate how much knowledge to transfer from the source domain and how much to rely on the direct representation $v_{direct}$ of the input. Intuitively, the larger domain gap between an input $x$ and the source domain, the more weight on the memory feature. We design a domain indicator for such domain awareness: $e_{domain} = T(E_{domain}(x))$, where $T(\cdot)$ is a lightweight network with the `tanh` activation functions and $E_{domain}(\cdot)$ is the domain encoder we have learned earlier.

**Source-Enhanced Representation $v_{transfer}$.** Our final representation of the input is a dynamically balanced version between the direct image feature and the memory enhanced feature:

$$v_{transfer} = v_{direct} + e_{domain} \otimes v_{enhance}, \tag{3}$$

which transfers class-discriminative knowledge from the labeled source domain to the input in a domain-aware manner. Operator $\otimes$ is element-wise multiplication. Adopting cosine classifiers [27, 10], we $\ell_2$-normalize this representation before sending it to the softmax classification layer. All of these choices help cope with domain mismatch when the input is significantly different from the source domain.

## 4. Experiments

**Datasets.** To facilitate a comprehensive evaluation on various tasks (*i.e*., classification, segmentation, and navigation), we carefully design four open compound domain adaptation (OCDA) benchmarks: C-Digits, C-Faces, C-Driving, and C-Mazes, respectively.

1. *C-Digits*: This benchmark aims to evaluate the classification adaptation ability under different appearances and backgrounds. It is built upon five classic digits datasets (SVHN [33], MNIST [21], MNIST-M [7], USPS [19] and SynNum [7]), where SVHN is used as the source domain, MNIST, MNIST-M, and USPS are mixed as the compound target domain, and SynNum is the open domain. We employ SWIT [1] as an additional open domain for further analysis.

2. *C-Faces*: This benchmark aims to evaluate the classification adaptation ability under different camera poses. It is built upon the Multi-PIE dataset [14], where C05 (frontal view) is used as source domain, C08-C14 (left side view) are combined as the compound target domain, and C19 (right side view) is kept out as the open domain.

3. *C-Driving*: This benchmark aims to evaluate the segmentation adaptation ability from simulation to different real driving scenarios. The GTA-5 [40] dataset is adopted as the source domain, while the BDD100K dataset [53] (with different scenarios including “rainy”, “snowy”, “cloudy”, and “overcast”) is taken for the compound and open domains.

4. *C-Mazes*: This benchmark aims to evaluate the navigation adaptation ability under different environmental appearances. It is built upon the GridWorld environment [18], where mazes with different colors are used

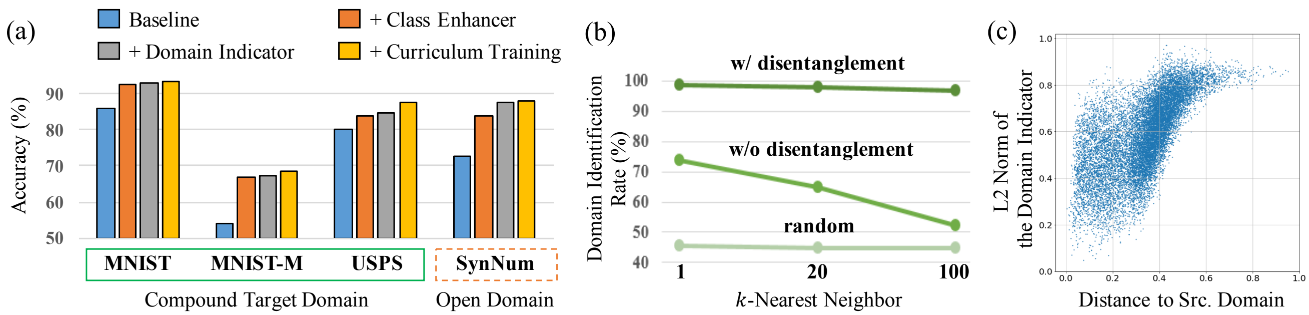
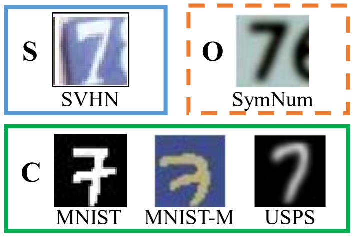


Figure 5: **Results of ablation studies about** (a) the memory-enhanced embeddings and curriculum domain adaptation, (b) the domain-focused factors disentanglement, and (c) the memory-induced domain indicator vs. gaps to the source.

Table 2: **Performance on the C-Digits benchmark**. The methods in gray are especially designed for multi-target domain adaptation. †MTDA uses domain labels, while ‡BTDA and DADA use the open domain images during training.



| Src. Domain<br>SVHN → | Compound Domains (C)<br>**MNIST** | <br>**MNIST-M** | <br>**USPS** | Open (O)<br>**SynNum** | Avg.<br>**C** | <br>**C+O** |
|---|---|---|---|---|---|---|
| ADDA [48] | 80.1±0.4 | 56.8±0.7 | 64.8±0.3 | 72.5±1.2 | 67.2±0.5 | 68.6±0.7 |
| JAN [30] | 65.1±0.1 | 43.0±0.1 | 63.5±0.2 | 85.6±0.0 | 57.2±0.1 | 64.3±0.1 |
| MCD [42] | 69.6±1.4 | 48.6±0.5 | 70.6±0.2 | **89.8±2.9** | 62.9±1.0 | 69.9±1.3 |
| MTDA† [9] | 84.6±0.3 | 65.3±0.2 | 70.0±0.2 | - | 73.3±0.2 | - |
| BTDA‡ [5] | 85.2±1.6 | **65.7±1.3** | 74.3±0.9 | 84.4±2.2 | 75.1±1.3 | 77.4±1.5 |
| DADA‡ [39] | - | - | - | - | - | 80.1±0.4 |
| Ours | **90.9±0.2** | **65.7±0.5** | **83.4±0.3** | 88.2±0.8 | **80.0±0.3** | **82.1±0.5** |

as the source and open domains. Since reinforcement learning often assumes no prior access to the environments, there are no compound target domains here.

**Network Architectures.** To make a fair comparison with previous works [48, 9, 39], the modified LeNet-5 [21] and ResNet-18 [15] are used as the backbone networks for C-Digits and C-Faces, respectively. Following [47, 58, 35], a pre-trained VGG-16 [44] is the backbone network for C-Driving. We additionally test our approach on reinforcement learning using ResNet-18 following [18].

**Evaluation Metrics.** The C-digits performance is measured by the digit classification accuracy, and the C-Faces performance is measured by the facial expression classification accuracy. The C-Driving performance is measured by the standard mIOU, and the C-Mazes performance is measured by the average successful rate in 300 steps. We evaluate the performance of each method with five runs and report both the mean and standard deviation. Moreover, we report both results of individual domains and the averaged results for a comprehensive analysis.

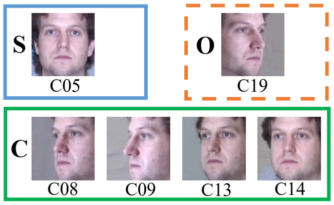
**Comparison Methods.** For classification tasks, we choose for comparison state-of-the-art methods in both conventional unsupervised domain adaptation (ADDA [48], JAN [30], MCD [42]) and the recent multi-target domain adaptation methods (MTDA [9], BTDA [5], DADA [39]). Since MTDA [9], BTDA [5] and DADA [39] are the most related to our work, we directly contrast our results to the numbers reported in their papers. For the segmentation task, we compare with three state-of-the-art methods, AdaptSeg [47], CBST [58], IBN-Net [35] and PyCDA [26]. For the reinforcement learning task, we benchmark with MTL, MLP [18] and SynPo [18], a representative work for adaptation across environments. We apply these methods to the same backbone networks as ours for a fair comparison.

### 4.1. Ablation Study

**Effectiveness of the Domain-Focused Factors Disentanglement.** Here we verify that the domain-focused factors disentanglement helps discover the latent structures in the compound target domain. It is probed by the domain identification rate within the $k$-nearest neighbors found by different encodings. Figure 5 (b) shows that features produced by our disentanglement have a much higher identification rate ($\sim$95%) than the counterparts without disentanglement ($\sim$65%).

**Effectiveness of the Curriculum Domain Adaptation.** Figure 5 (a) also reveals that, in the compound domain, the curriculum training contributes to the performance on USPS more than MNIST and MNITS-M. On the other hand, we can observe from Figure 4 and Table 2 that USPS is the furthest target domain from the source domain SVHN. It implies that curriculum domain adaptation makes it easy to adapt to the distant target domains through an easy-to-hard adaptation schedule.

Table 3: **Performance on the C-Faces benchmark**. The methods in gray are especially designed for multi-target domain adaptation. $^\dagger$MTDA uses domain labels during training.



| Src. Domain | Compound Domains (C) | | | | Open (O) | Avg. | |
|---|---|---|---|---|---|---|---|
| **C05 →** | **C08** | **C09** | **C13** | **C14** | **C19** | **C** | **C+O** |
| ADDA [48] | 46.9±0.2 | 36.4±0.5 | 39.1±0.3 | 65.4±0.4 | 71.8±0.8 | 47.0±0.4 | 51.9±0.4 |
| JAN [30] | 63.5±0.3 | 40.6±1.0 | 83.5±0.4 | **92.0±0.8** | 52.5±1.5 | 69.7±0.6 | 66.2±0.8 |
| MCD [42] | 50.4±0.5 | 45.8±0.2 | 77.8±0.1 | 88.0±0.1 | 60.4±0.9 | 65.7±0.2 | 64.6±0.4 |
| MTDA$^\dagger$ [9] | 49.0±0.2 | 48.2±0.1 | 53.1±0.2 | 84.3±0.1 | - | 58.7±0.2 | - |
| Ours | **73.3±0.2** | **55.1±0.4** | **84.1±0.1** | 88.9±0.3 | **72.7±0.6** | **75.4±0.3** | **74.8±0.3** |

Table 4: **Performance on the C-Driving (left) and C-Mazes benchmarks (right)**. “SynPo+Aug.” indicates that we equip SynPo with proper color augmentation/randomization during training. Visual illustrations of both datasets are in Figure 6.

| Source | Compound (C) | | | Open (O) | Avg. | |
|---|---|---|---|---|---|---|
| **GTA-5 →** | **Rainy** | **Snowy** | **Cloudy** | **Overcast** | **C** | **C+O** |
| Source Only | 16.2 | 18.0 | 20.9 | 21.2 | 18.9 | 19.1 |
| AdaptSeg [47] | 20.2 | 21.2 | 23.8 | 25.1 | 22.1 | 22.5 |
| CBST [58] | 21.3 | 20.6 | 23.9 | 24.7 | 22.2 | 22.6 |
| IBN-Net [35] | 20.6 | 21.9 | 26.1 | 25.5 | 22.8 | 23.5 |
| PyCDA [26] | 21.7 | 22.3 | 25.9 | 25.4 | 23.3 | 23.8 |
| Ours | **22.0** | **22.9** | **27.0** | **27.9** | **24.5** | **25.0** |

| Source | Open(O) | | | | Avg. |
|---|---|---|---|---|---|
| **M0 →** | **M1** | **M2** | **M3** | **M4** | **O** |
| Source Only | 0±0 | 0±0 | 0±0 | 0±0 | 0±0 |
| MTL | 0±0 | 30±5 | 75±0 | 65±5 | 42.5±2.5 |
| MLP [18] | 5±5 | 45±10 | 75±5 | 80±10 | 51.2±7.5 |
| SynPo [18] | 5±5 | 30±20 | 80±5 | 30±5 | 36.3±8.8 |
| SynPo+Aug. | 0±5 | 40±10 | 95±5 | 45±5 | 45.0±6.3 |
| Ours | **80±2.5** | **75±10** | **85±5** | **90±5** | **82.5±5.6** |

**Effectiveness of Memory-Enhanced Representations.** Recall that the memory-enhanced representations consist of two main components: the enhancer coming from the memory and the domain indicator. From Figure 5 (a), we observe that the class enhancer leads to large improvements on all target domains. It is because the enhancer from the memory transfers useful semantic concepts to the input of any domain. Another observation is that the domain indicator is the most effective on the open domain (“SynNum”), because it helps dynamically calibrate the representations by leveraging domain relations (Figure 5 (c)).

## 4.2. Comparison Results

**C-Digits.** Table 2 shows the comparison performances of different methods. We have the following observations. Firstly, ADDA [48] and JAN [30] boost the performance on the compound domain by enforcing global distribution alignment. However, they also sacrifice the performance on the open domain since there is no built-in mechanism for handling any new domains, “overfitting” the model to the seen domains. Secondly, MCD [42] improves the results on the open domain, but its accuracy degrades on the compound target domain. Maximizing the classifier discrepancy increases the robustness to the open domain; however, it also fails to capture the fine-grained latent structure in the compound target domain. Lastly, compared to other multi-target domain adaptation methods (MTDA [9] and DADA [39]), our approach discovers domain structures and performs domain-aware knowledge transfer, achieving substantial advantages on all the test domains.

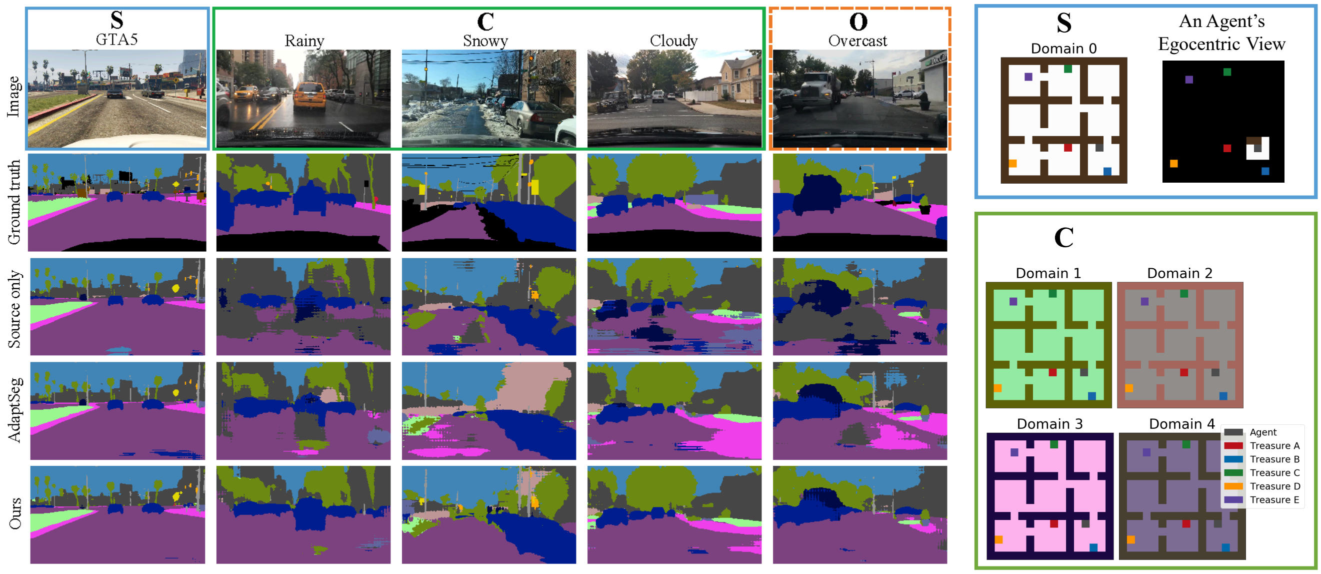
**C-Faces.** Similar observations can be made on the C-Faces benchmark as shown in Table 3. Since face representations are inherently hierarchical, JAN [30] demonstrates competitive results on C14 due to its layer-wise transferring strategy. Under the domain shift with different camera poses, our approach still consistently outperforms other alternatives for both the compound and open domains.

**C-Driving.** We compare with the state-of-the-art semantic segmentation adaptation methods such as AdaptSeg [47], CBST [58], and IBN-Net [35]. All methods are tested under real-world driving scenarios in the BDD100K dataset [53]. We can see that our approach has clear advantages on both the compound domain (1.1% gains) and the open domain (2.4% gains) as shown in Table 4 (left). We show detailed per-class accuracies in the appendices. The qualitative comparisions are shown in Figure 6 (a).

**C-Mazes.** To directly compare with SynPo [18], we also evaluate on the GridWorld environments they provided. The task in this benchmark is to learn navigation policies that can successfully collect all the treasures in the given mazes. Existing reinforcement learning methods suffer from environmental changes, which we simulate as the appearances of the mazes here. The final results are listed in Table 4 (right). Our approach transfers visual knowledge among navigation experiences and achieves more than 30% improvements over the prior arts.

## 4.3. Further Analysis

**Robustness to the Complexity of the Compound Target Domain.** We control the complexity of the compound target domain by varying the number of traditional target domains / datasets in it. Here we gradually increase constituting domains from a single target domain (*i.e.*, MNIST) to two, and eventually three (*i.e.*, MNIST + MNIST-M + USPS). From Figure 7 (a), we observe that as the number of datasets increase, our approach only undergoes a moderate



(a) C-Driving Benchmark

(b) C-Mazes Benchmark

Figure 6: (a) **Qualitative results comparison** of semantic segmentation on the source domain (**S**), the compound target domain (**C**), and the open domain (**O**). (b) **Illustrations** of the 5 different domains in the C-Mazes benchmark. Our approach consistently outperforms existing domain adaptation methods across all compound and open target domains.

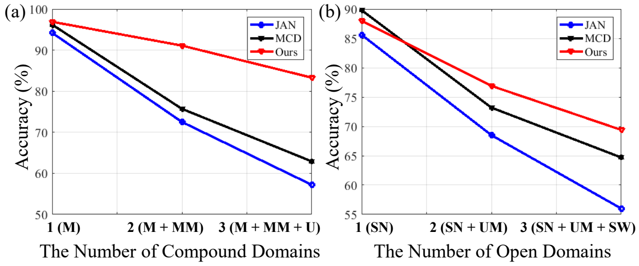


Figure 7: **Further analysis** on the (a) robustness to the complexity of the compound target domain and (b) robustness to the number of open domains. "M", "MM" and "U" stand for MNIST, MNIST-M, and USPS, respectively, while "SN", "UM" and "SW" stand for SynNum, USPS-M, and SWIT, respectively.

performance drop. The learned curriculum enables gradual knowledge transfer that is capable of coping with complex structures in the compound target domain.

**Robustness to the Number of Open Domains.** The performance change w.r.t. the number of open domains is demonstrated in Figure 7 (b). Here we include two new digits datasets, USPS-M (crafted in a similar way as MNIST-M) and SWIT [1], as the additional open domains. Compared to JAN [30] and MCD [42], our approach is more resilient to the various numbers of open domains. The domain indicator module in our framework helps dynamically calibrate the embedding, thus enhancing the robustness to open domains. Figure 8 presents the t-SNE visualization comparison between the obtained embeddings of JAN [30], MCD [42], and our approach.

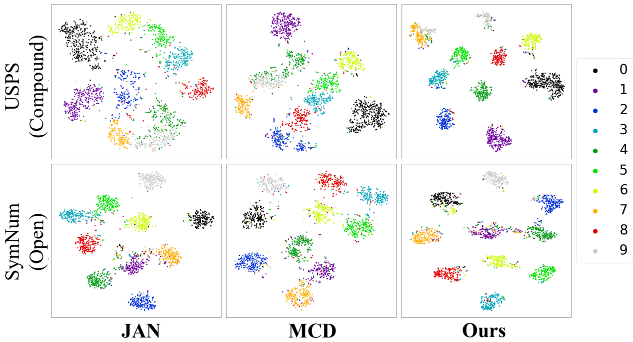


Figure 8: **t-SNE visualization** of the obtained embeddings. Compared to other methods, our approach is capable of producing class-discriminative features on both compound and open target domains.

## 5. Summary

We formalize a more realistic topic called open compound domain adaptation for domain-robust learning. We propose a novel model which includes a self-organizing curriculum domain adaptation to bootstrap generalization and a memory enhanced feature representation to build agility towards open domains. We develop several benchmarks on classification, recognition, segmentation, and reinforcement learning and demonstrate the effectiveness of our model.

**Acknowledgements.** This research was supported, in part, by the General Research Fund (GRF) of Hong Kong (No. 14236516 & No. 14203518), Berkeley Deep Drive, DARPA, NSF 1835539, and US Government fund through Etegent Technologies on Low-Shot Detection in Remote Sensing Imagery.

## References

[1] Switzerland handwritten digits dataset. https://github.com/kensanata/numbers. Accessed: 2019-03-15. 5, 8

[2] Andreea Bobu, Eric Tzeng, Judy Hoffman, and Trevor Darrell. Adapting to continuously shifting domains. *ICLR Workshop*, 2018. 3

[3] Zhangjie Cao, Lijia Ma, Mingsheng Long, and Jianmin Wang. Partial adversarial domain adaptation. In *ECCV*, 2018. 3, 11

[4] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *ICLR*, 2015. 13

[5] Ziliang Chen, Jingyu Zhuang, Xiaodan Liang, and Liang Lin. Blending-target domain adaptation by adversarial meta-adaptation networks. In *CVPR*, 2019. 3, 6, 11, 12

[6] Dengxin Dai, Christos Sakaridis, Simon Hecker, and Luc Van Gool. Model adaptation with synthetic and real data for semantic dense foggy scene understanding. *IJCV*, 2019. 2

[7] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, 2015. 2, 5, 12

[8] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 2016. 1, 2

[9] Behnam Gholami, Pritish Sahu, Ognjen Rudovic, Konstantinos Bousmalis, and Vladimir Pavlovic. Unsupervised multi-target domain adaptation: An information theoretic approach. *arXiv preprint arXiv:1810.11547*, 2018. 3, 6, 7, 11, 12

[10] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *CVPR*, 2018. 5

[11] Boqing Gong, Kristen Grauman, and Fei Sha. Reshaping visual datasets for domain adaptation. In *NIPS*, 2013. 1, 3

[12] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, 2012. 1, 2

[13] Rui Gong, Wen Li, Yuhua Chen, and Luc Van Gool. Dlow: Domain flow for adaptation and generalization. In *CVPR*, 2019. 3

[14] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker. Multi-pie. In *2008 8th IEEE International Conference on Automatic Face Gesture Recognition*, 2008. 5, 12

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6, 11, 13

[16] Judy Hoffman, Brian Kulis, Trevor Darrell, and Kate Saenko. Discovering latent domains for multisource domain adaptation. In *ECCV*, 2012. 1, 3

[17] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *ICML*, 2018. 2, 12, 13

[18] Hexiang Hu, Liyu Chen, Boqing Gong, and Fei Sha. Synthesized policies for transfer and adaptation across tasks and environments. In *NIPS*, 2018. 5, 6, 7, 12, 13, 19

[19] J. J. Hull. A database for handwritten text recognition research. *TPAMI*, 1994. 2, 5, 12

[20] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015. 13

[21] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998. 2, 5, 6, 12, 13

[22] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Learning to generalize: Meta-learning for domain generalization. In *AAAI*, 2018. 1, 3

[23] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In *CVPR*, 2018. 3

[24] Wen Li, Zheng Xu, Dong Xu, Dengxin Dai, and Luc Van Gool. Domain generalization and adaptation using low rank exemplar svms. *TPAMI*, 2017. 1

[25] Xiaoxiao Li, Ziwei Liu, Ping Luo, Chen Change Loy, and Xiaoou Tang. Not all pixels are equal: Difficulty-aware semantic segmentation via deep layer cascade. In *CVPR*, 2017. 2

[26] Qing Lian, Fengmao Lv, Lixin Duan, and Boqing Gong. Constructing self-motivated pyramid curriculums for cross-domain semantic segmentation: A non-adversarial approach. In *ICCV*, 2019. 6, 7

[27] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*, 2017. 5

[28] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *CVPR*, 2019. 5

[29] Ziwei Liu, Sijie Yan, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Fashion landmark detection in the wild. In *ECCV*, 2016. 2

[30] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *ICML*, 2017. 1, 2, 6, 7, 8, 14, 17, 18, 20, 21

[31] Massimiliano Mancini, Samuel Rota Bulo, Barbara Caputo, and Elisa Ricci. Adagraph: Unifying predictive and continuous domain adaptation through graphs. In *CVPR*, 2019. 3

[32] Massimiliano Mancini, Lorenzo Porzi, Samuel Rota Bulo, Barbara Caputo, and Elisa Ricci. Inferring latent domains for unsupervised deep domain adaptation. *TPAMI*, 2019. 3

[33] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. *NIPS*, 2011. 2, 5, 12

[34] Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 2010. 1

[35] Xingang Pan, Ping Luo, Jianping Shi, and Xiaoou Tang. Two at once: Enhancing learning and generalization capacities via ibn-net. In *ECCV*, 2018. 6, 7, 13, 15

[36] Yingwei Pan, Ting Yao, Yehao Li, Yu Wang, Chong-Wah Ngo, and Tao Mei. Transferrable prototypical networks for unsupervised domain adaptation. In *CVPR*, 2019. 5

[37] Pau Panareda Busto and Juergen Gall. Open set domain adaptation. In *ICCV*, 2017. 3, 11

[38] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *ICCV*, 2019. 2

[39] Xingchao Peng, Zijun Huang, Ximeng Sun, and Kate Saenko. Domain agnostic learning with disentangled representations. In *ICML*, 2019. 3, 6, 7, 11, 12

[40] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016. 5, 12

[41] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, 2010. 1, 2

[42] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*, 2018. 1, 2, 6, 7, 8, 14, 17, 18, 20, 21

[43] Kuniaki Saito, Shohei Yamamoto, Yoshitaka Ushiku, and Tatsuya Harada. Open set domain adaptation by backpropagation. In *ECCV*, 2018. 3, 11

[44] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 6, 11

[45] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NIPS*, 2017. 5

[46] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *CVPR*, 2011. 1, 2

[47] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *CVPR*, 2018. 6, 7, 13, 14, 15

[48] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *CVPR*, 2017. 1, 2, 4, 6, 7, 13

[49] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, 2017. 2, 14

[50] Zuxuan Wu, Xin Wang, Joseph E Gonzalez, Tom Goldstein, and Larry S Davis. Ace: Adapting to changing environments for semantic segmentation. In *ICCV*, 2019. 3

[51] Caiming Xiong, Scott McCloskey, Shao-Hang Hsieh, and Jason J Corso. Latent domains modeling for visual domain adaptation. In *AAAI*, 2014. 3

[52] Zheng Xu, Wen Li, Li Niu, and Dong Xu. Exploiting low-rank structure from latent domains for domain generalization. In *ECCV*, 2014. 3

[53] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2018. 5, 7, 12

[54] Huanhuan Yu, Menglei Hu, and Songcan Chen. Multi-target unsupervised domain adaptation without exactly shared categories. *arXiv preprint arXiv:1809.00852*, 2018. 3

[55] Jing Zhang, Zewei Ding, Wanqing Li, and Philip Ogunbona. Importance weighted adversarial nets for partial domain adaptation. In *CVPR*, 2018. 3, 11

[56] Yang Zhang, Philip David, Hassan Foroosh, and Boqing Gong. A curriculum domain adaptation approach to the semantic segmentation of urban scenes. *TPAMI*, 2019. 2, 4

[57] Yang Zou, Zhiding Yu, Xiaofeng Liu, BVK Kumar, and Jinsong Wang. Confidence regularized self-training. In *ICCV*, 2019. 2

[58] Yang Zou, Zhiding Yu, BVK Vijaya Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *ECCV*, 2018. 2, 6, 7, 13, 15

# Appendices

In this supplementary material, we provide details omitted in the main text including:

- Section A: relation to other DA problems (Sec. 2 "Related Works" of the main paper.)
- Section B: more methodology details (Sec. 3 "Our Approach" of the main paper.)
- Section C: detailed experimental setup (Sec. 4 "Experiments" of the main paper.)
- Section D: additional comparison results (Sec. 4.2 "Comparison Results" of the main paper.)
- Section E: additional visualization of our approach (Sec. 4.3 "Further Analysis" of the main paper.)

## A. Relation to Other DA Problems

Human vision system shows remarkable generalization ability to see clear across many different domains, such as in foggy and rainy days. Computer vision system, on the other hand, has long been haunted by this domain shift issue. Several sub-fields in domain adaptation have been studies to mitigate this challenge.

**Open/Partial Set Domain Adaptation.** Another route of research aims to tackle the category sharing/unsharing issues between source and target domain, namely open set [37, 43] and partial set [55, 3] domain adaptation. They assume that the target domain contains either (1) new categories that don't appear in source domain; or (2) only a subset of categories that appear in source domain. Both settings concern the "openness" of categories. Instead, in this work we investigate the "openness" of domains, *i.e.* we assume there are unknown domains existing that are absent in the training phase.

## B. More Methodology Details

**Notation Summary.** We summarize the notations used in the paper in Table 5.

**Details of Class and Domain Manifold Disentanglement.** The detailed class and domain manifold disentanglement algorithm is shown in Algorithm 2.

**Time Complexity.** Our approach introduces negligible computational overhead (1.3%) to the standard deep networks, such as VGG [44] and ResNet [15], since only a lightweight memory module is inserted during inference.

**Methodology Highlight.** Our main methodology contribution is the entire neural architecture that can address the complexity of compound domains during training and handle the unseen domains during testing, as depicted in Figure 9.

Table 5: **Summary** of notations.

| Notation | Meaning |
|---|---|
| $x$ | input image |
| $y$ | category label |
| $z_{random}$ | random category label |
| $E_{class}(\cdot)$ | class encoder |
| $\Phi(\cdot)$ | class classifier |
| $E_{domain}(\cdot)$ | domain encoder |
| $Decoder(\cdot)$ | class decoder |
| $D(\cdot)$ | class classifier after domain encoder |
| $v_{direct}$ | direct feature |
| $M$ | visual memory |
| $v_{enhance}$ | class enhancer |
| $e_{domain}$ | domain indicator |
| $T(\cdot)$ | network that generates $e_{domain}$ |
| $v_{transfer}$ | source-enhanced representation |

**Algorithm 2** Disentangling training. $E_{class}(\cdot)$ and $\Phi$ has been trained using source-domain data, $Deccoder(\cdot)$: the decoder, $C$: number of classes, $\gamma$: a constant.

```
Input: $E_{class}(\cdot)$, $E_{domain}(\cdot)$, $\Phi(\cdot)$, $D(\cdot)$, $Decoder(\cdot)$, $C$, $\gamma$
  for k iterations do
      Sample mini-batch $x$.
      Compute pseudo label $y_{pseudo} \leftarrow \Phi\left(E_{class}\left(x\right)\right)$.
      Update the discriminator $D$ with: $\nabla_{\theta_D} \sum_j y^j_{pseudo} \log\left(D\left(E_{domain}\left(x^j\right)\right)\right)$.
      Prepare random label $z_{random} \sim uniform\{0, 1, ..., C - 1\}$, and convert it to one-hot vector $y_{random}$.
      Compute adversarial loss: $L_{adv} \leftarrow \sum_j y^j_{random} \log\left(D\left(E_{domain}(x^j)\right)\right)$.
      Compute reconstruction loss: $L_{rec} \leftarrow \sum_j |Decoder\left(E_{class}\left(x^j\right), E_{domain}\left(x^j\right)\right) - x^j|$.
      Update the domain encoder $E_{domain}$ with: $\nabla_{\theta_{E_{domain}}}\left(L_{adv} + \gamma L_{rec}\right)$.
  end for
```

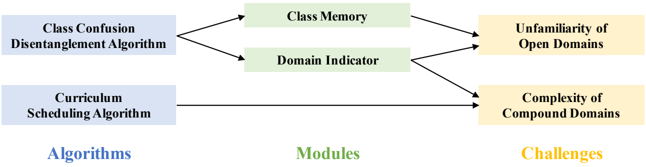

Class Confusion Disentanglement Algorithm
Curriculum Scheduling Algorithm
Class Memory
Domain Indicator
Unfamiliarity of Open Domains
Complexity of Compound Domains
Algorithms
Modules
Challenges

Figure 9: **Methodology highlight** of our entire neural architecture.

**Methodology Comparisons.** Table 6 summarizes the key methodological differences between MTDA [9], BTDA [5], DADA [39] and our OCDA.

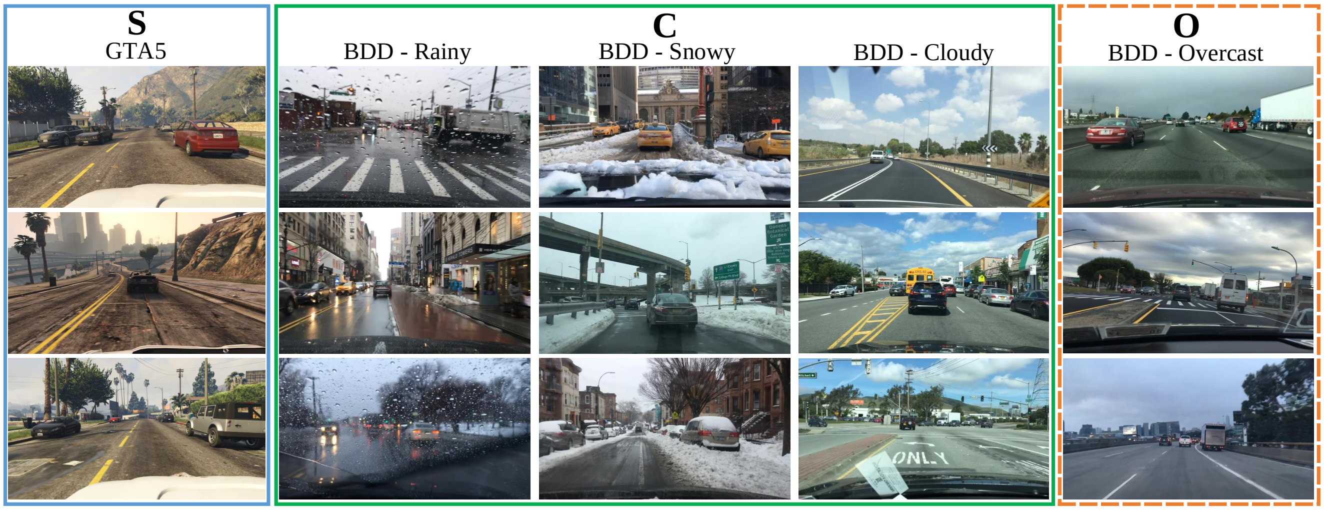


Figure 10: **Examples of the C-Driving benchmark** (GTA5 and BDD100K datasets).

Table 6: **Methodology comparisons** between MTDA, BTDA, DADA and our OCDA.

| | **MTDA** [9] | **BTDA** [5] | **DADA** [39] | **OCDA** |
|---|---|---|---|---|
| Feature Disentangle | entropy minimization | × | mutual information | class confusion |
| Domain Invariance | adversarial | adversarial | adversarial | memory+ adversarial |
| Latent Domain Discovery | × | clustering | × | domain indicator |
| Latent Domain Ranking | × | × | × | curriculum |

## C. Experimental Setup

### C.1. OCDA Datasets and Benchmarks

**C-Digits.** The C-Digits dataset is consist of 5 digit datasets: SVHN [33], MNIST [21], MNIST-M [7], USPS [19], and SynNum [7]. SVHN is a dataset of street view housing numbers. MNIST and USPS are two datasets of handwritten numbers. MNIST-M and SynNum are two datasets of synthetic numbers. We choose SVHN as the source domain, MNIST, MNIST-M, and USPS as target domains that can be accessed during training, and SynNum as the open domain, which is only accessible during testing. Images from all domains are scaled to 32 × 32 and converted to RGB format. We follow the origin training/testing split of each dataset. The SVHN dataset is balanced across the classes, following [17]. Besides, no further pre-processing is applied to the data.

**C-Faces.** We choose images of 6 different view angles from the We choose images of 6 different view angles from the Multi-PIE dataset [14]: C051, C080, C090, C130, C140, and C190. 6 facial expressions are treated as classification categories: neutral, smile, surprise, squint, disgust, and scream. We use images of view angle C051 as the source domain in our experiment, C080, C090, C130, and C140 as target domains that can be accessed during training, and C190 as the open domain. All the images are aligned according to the face landmarks and cropped around the facial areas into 224 × 224 images.

**C-Driving.** For semantic segmentation on driving scenarios, we adopt the **GTA5** [40] dataset as the source domain, and the **BDD100K** [53] dataset as the Compound and open domains. Examples of these datasets are illustrated in Figure 10.

GTA5 is a virtual street view dataset generated from Grand Theft Auto V (GTA5). It has 24966 images of resolution 1914 × 1052. The domain categories and statistics of the BDD100K dataset are provided in Table 7. We use the *Rainy*, *Snowy*, *Cloudy*, and *Overcast* domains in the training set of BDD100K in our experiments. The *Overcast* domain is used as the open domain while the rest are used as the Compound domains. Other domains of the original BDD100K dataset like *Clear* and *Foggy* are not used. Among these data, a small fraction with annotations is used as the validation set, while the rest are used as the training set for adaptation. The results in our experiments are reported on the validation set.

**C-Mazes.** This is a reinforcement learning dataset which is consist of mazes with different colors. The mazes are generated following [18], where agent is asked to collect treasure spots from the mazes by different orders. For simplicity, we only use one scene (*i.e.* maze with one topology) and two tasks (*i.e.* two different treasure collection orders) for the experiments. We randomly generated 5 combinations of colors to construct the domains (which are illustrated in Figure 11). The colors of agent and five treasures are the same across the domain. We use **Domain 0** as the source domain, and the rest of the 4 domains are target domains.

Table 7: **Statistics of BDD100K dataset** in our Open Compound Domain Adaptation (OCDA) setting.

|  | Compound (C) |  |  | Open |  |
|---|---|---|---|---|---|
| **Domains** | **Rainy** | **Snowy** | **Cloudy** | **Overcast** | **Total** |
| training set w/o label | 4855 | 5307 | 4535 | 8143 | 22840 |
| validation set w/ label | 215 | 242 | 346 | 627 | 1430 |

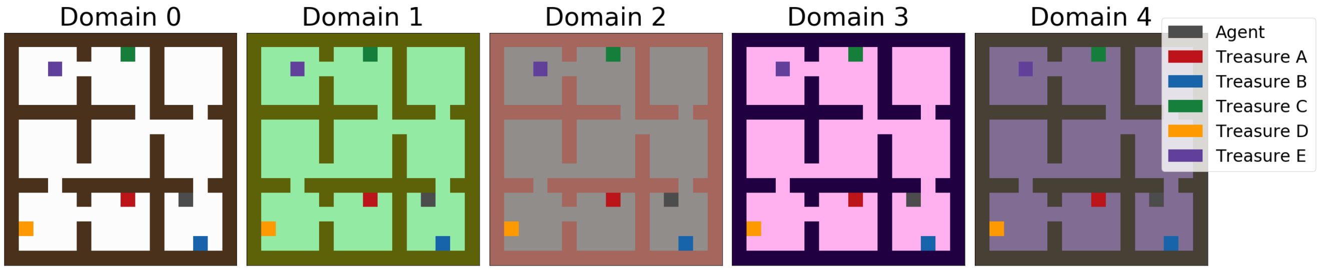


Figure 11: **Illustrations** of the five different domains in the C-Mazes benchmark.

## C.2. Training Details

**C-Digits.** The backbone model for this experiment is a LeNet-5 [21], following the setups in [17]. There are two stages in the training process. (1) In the first stage, we train the network with discriminative centroid loss for 100 epochs as a warm start for the memory-enhanced deep neural network. (2) In the second stage, we further fine-tune the networks with curriculum sampling and memory modules on top of the backbone network, where a domain adversarial loss [48] is incorporated and the weights learned in the first stage are copied to two identical but independent networks (source network and target network). Only weights of the target network are updated during this stage. Centroids of each class (*i.e.* constituting elements in the class memory) are calculated in the beginning of this stage, and the classifiers are reinitialized. The model is trained without the discriminative centroid loss in stage 2. Some major hyper-parameters can be found in Table 8.

**C-Faces.** Experiments on the C-Faces dataset are similar to experiments on the C-Digits dataset. However, the backbone model is ResNet18 [15] with random initialization, instead of a LeNet-5. Some major hyper-parameters can also be found in Table 8.

Table 8: **The major hyper-parameters** used in our experiments. "LR." stands for learning rate.

| Dataset | Initial LR. | Epoch | betas for ADAM |
|---|---|---|---|
| C-Digits (stage 1) | 1e-4 | 100 | (0.9, 0.999) |
| C-Digits (stage 2) | 1e-5 | 200 | (0.9, 0.999) |
| C-Faces (stage 1) | 1e-4 | 100 | (0.9, 0.999) |
| C-Faces (stage 2) | 1e-5 | 200 | (0.9, 0.999) |

**C-Driving.** Our implementation mainly follows [47]. We use DeepLab-VGG16 [4] model with synchronized batch normalization and the batch size is set to 8. The initial learning rate is 0.01 and is decreased using the ”poly” policy with 0.9 power. The maximum iteration number is 40k and we apply early stop at 5k iteration to reduce overfitting. The GTA5 and BDD100K images are resized to $1280 \times 720$ and $960 \times 540$ for training respectively. For the IBN-Net [35] baseline, we replace the batch normalization layers after the $\{2, 4, 7\}$-*th* convolution layers with instance normalization layers. For the AdaptSeg [47] baseline, we use the Adam optimizer [20] and 0.005 initial learning rate for the discriminator.

In our method, we use dynamic transferable embedding and curriculum training in addition to adversarial adaptation [47]. The visual memory here also comprises of a set of class centroids, which is an aggregation of local features belonging to the same category. Inspired by [58], the curriculum learning procedure is further designed to include the averaged probability confidence of each image as a guidance. Specifically, the samples that are easier, *i.e.*, have higher confidence, are firstly fed into the model for adaptation. Since domain encoder is not accessible here, we only use class enhancer for dynamic transferable embedding.

**C-Mazes.** The experiments for C-Mazes are also conducted in two-stages. In the first stage, We follow the setups in [18] to train a randomly initialized ResNet-18 policy network. We only use one single topology and two different tasks for the experiments. The total episodes is set to 16000. The initial learning rate is set to 0.001. Then in the second stage, we use the pre-trained model to calculate state feature centers for each actions and use memory module to fine-tune the model.

## D. Additional Results

**C-Digits.** We have experimented with mnist, mnist-m or usps as the source domain. On average, the performance gain of our approach is 8.1% over the baseline method JAN [30] and 8.9% over MCD [42]; Likewise, the average performance gain with Multi-PIE as the source domain is 36.5% and 14.4% over the baselines.

**C-Driving.** Figure 13 shows example results on the GTA5 and BDD100K dataset. Our method produces more accurate segmentation results on the compound domains and the open domain compared to 'source only' and AdaptSeg [47]. Per-category results are provided in Table 9, and ablation study of dynamic transferable embedding and curriculum training are shown in Figure 12.

**Office-Home [49].** We had some preliminary results on Office-Home [49], where our approach outperforms baseline methods (JAN [30] and MCD [42]) by 15.3% and 7.9%, respectively.

## E. More Visualizations

**t-SNE Visualization.** Here we show t-SNE visualizations of the learned dynamic transferable embedding on the C-Digits, C-Faces, and C-Mazes testing data (Figure 14 - 16), among various methods. The dynamic transferable embedding on the C-Mazes benchmark are state features of each actions. Our approach generally learns a more discriminative feature space thanks to the proposed disentanglement and memory modules.

**Confusion Matrices.** Here we show visualizations of class confusion matrices on the C-Digits and C-Faces testing data in Figure 17 and Figure 18. Compared to MCD [42] and JAN [30], our approach performs better on the discriminative accuracies of each class.

Table 9: **Per-category IoU(%) results on the C-Driving Benchmark.** (BDD100K dataset is used as the real-world target domain data.) The ’train’ and ’bicycle’ categories are not listed because their results are close to zero.

| Domain | Method | road | sidewalk | building | wall | fence | pole | light | sign | veg | terrain | sky | person | rider | car | truck | bus | mcycle | mIoU |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Rainy | Source only | 48.3 | 3.4 | 39.7 | 0.6 | 12.2 | 10.1 | 5.6 | 5.1 | 44.3 | 17.4 | 65.4 | 12.1 | 0.4 | 34.5 | 7.2 | 0.1 | 0.5 | 16.2 |
| | AdaptSeg [47] | 58.6 | 17.8 | 46.4 | 2.1 | 19.6 | 15.6 | 5.0 | 7.7 | **55.6** | **20.7** | 65.9 | 17.3 | 0.0 | 41.3 | 7.4 | 3.1 | 0.0 | 20.2 |
| | CBST [58] | 59.4 | 13.2 | 47.2 | 2.4 | 12.1 | 14.1 | 3.5 | 8.6 | 53.8 | 13.1 | **80.3** | 13.7 | **17.2** | **49.9** | 8.9 | 0.0 | **6.6** | 21.3 |
| | IBN-Net [35] | 58.1 | **19.5** | 51.0 | **4.3** | **16.9** | **18.8** | 4.6 | **9.2** | 44.5 | 11.0 | 69.9 | **20.0** | 0.0 | 39.9 | 8.4 | 15.3 | 0.0 | 20.6 |
| | Ours | **63.0** | 15.4 | **54.2** | 2.5 | 16.1 | 16.0 | **5.6** | 5.2 | 54.1 | 14.9 | 75.2 | 18.5 | 0.0 | 43.2 | **9.4** | **24.6** | 0.0 | **22.0** |
| Snowy | Source only | 50.8 | 4.7 | 45.1 | **5.9** | **24.0** | 8.5 | 10.8 | 8.7 | 35.9 | 9.4 | 60.5 | 17.3 | 0.0 | 47.7 | 9.7 | 3.2 | **0.7** | 18.0 |
| | AdaptSeg [47] | 59.9 | 13.3 | 52.7 | 3.4 | 15.9 | 14.2 | 12.2 | 7.2 | **51.0** | **10.8** | 72.3 | 21.9 | 0.0 | 55.0 | 11.3 | 1.7 | 0.0 | 21.2 |
| | CBST [58] | 59.6 | 11.8 | 57.2 | 2.5 | 19.3 | 13.3 | 7.0 | **9.6** | 41.9 | 7.3 | 70.5 | 18.5 | 0.0 | **61.7** | 8.7 | 1.8 | 0.2 | 20.6 |
| | IBN-Net [35] | 61.3 | **13.5** | 57.6 | 3.3 | 14.8 | **17.7** | 10.9 | 6.8 | 39.0 | 6.9 | 71.6 | **22.6** | 0.0 | 56.1 | **13.8** | **20.4** | 0.0 | 21.9 |
| | Ours | **68.0** | 10.9 | **61.0** | 2.3 | 23.4 | 15.8 | **12.3** | 6.9 | 48.1 | 9.9 | **74.3** | 19.5 | 0.0 | 58.7 | 10.0 | 13.8 | 0.1 | **22.9** |
| Cloudy | Source only | 47.0 | 8.8 | 33.6 | 4.5 | 20.6 | 11.4 | **13.5** | 8.8 | 55.4 | 25.2 | 78.9 | 20.3 | 0.0 | 53.3 | 10.7 | 4.6 | 0.0 | 20.9 |
| | AdaptSeg [47] | 51.8 | 15.7 | 46.0 | 5.4 | **25.8** | 18.0 | 12.0 | 6.4 | **64.4** | 26.4 | 82.9 | **24.9** | 0.0 | 58.4 | 10.5 | 4.4 | 0.0 | 23.8 |
| | CBST [58] | 56.8 | **21.5** | 45.9 | 5.7 | 19.5 | 17.2 | 10.3 | 8.6 | 62.2 | 24.3 | **89.4** | 20.0 | 0.0 | 58.0 | **14.6** | 0.1 | 0.1 | 23.9 |
| | IBN-Net [35] | 60.8 | 18.1 | 50.5 | **8.2** | 25.6 | **20.4** | 12.0 | **11.3** | 59.3 | 24.7 | 84.8 | 24.1 | **12.1** | 59.3 | 13.7 | 9.0 | 1.2 | 26.1 |
| | Ours | **69.3** | 20.1 | **55.3** | 7.3 | 24.2 | 18.3 | 12.0 | 7.9 | 64.2 | **27.4** | 88.2 | 24.7 | 0.0 | **62.8** | 13.6 | **18.2** | 0.0 | **27.0** |
| Overcast | Source only | 46.6 | 9.5 | 38.5 | 2.7 | 19.8 | 12.9 | **9.2** | 17.5 | 52.7 | 19.9 | 76.8 | 20.9 | 1.4 | 53.8 | 10.8 | 8.4 | **1.8** | 21.2 |
| | AdaptSeg [47] | 59.5 | 24.0 | 49.4 | 6.3 | 23.3 | 19.8 | 8.0 | 14.4 | **61.5** | 22.9 | 74.8 | 29.9 | 0.3 | 59.8 | 12.8 | 9.7 | 0.0 | 25.1 |
| | CBST [58] | 58.9 | **26.8** | 51.6 | 6.5 | 17.8 | 17.9 | 5.9 | **17.9** | 60.9 | 21.7 | **87.9** | 22.9 | 0.0 | 59.9 | 11.0 | 2.1 | 0.2 | 24.7 |
| | IBN-Net [35] | 62.9 | 25.3 | 55.5 | 6.5 | 21.2 | **22.3** | 7.2 | 15.3 | 53.3 | 16.5 | 81.6 | **31.1** | **2.4** | 59.1 | 10.3 | **14.2** | 0.0 | 25.5 |
| | Ours | **73.5** | 26.5 | **62.5** | **8.6** | **24.2** | 20.2 | 8.5 | 15.2 | 61.2 | **23.0** | 86.3 | 27.3 | 0.0 | **64.4** | **14.3** | 13.3 | 0.0 | **27.9** |

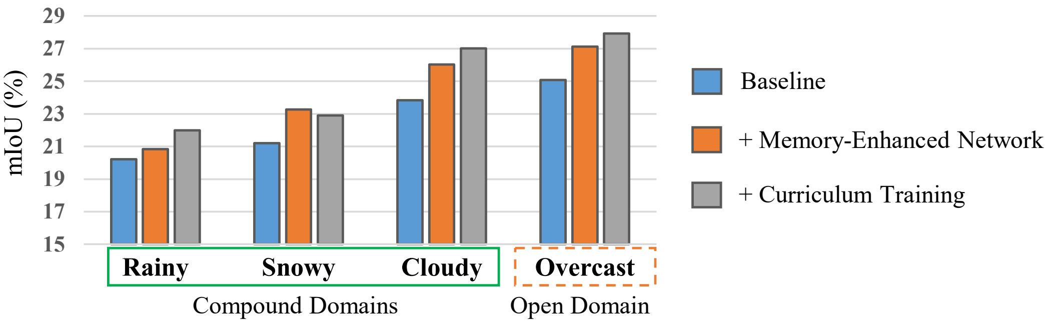


Figure 12: **Ablation study** of memory-enhanced neural network and curriculum training on the C-Driving benchmark (GTA5 and BDD100K datasets).

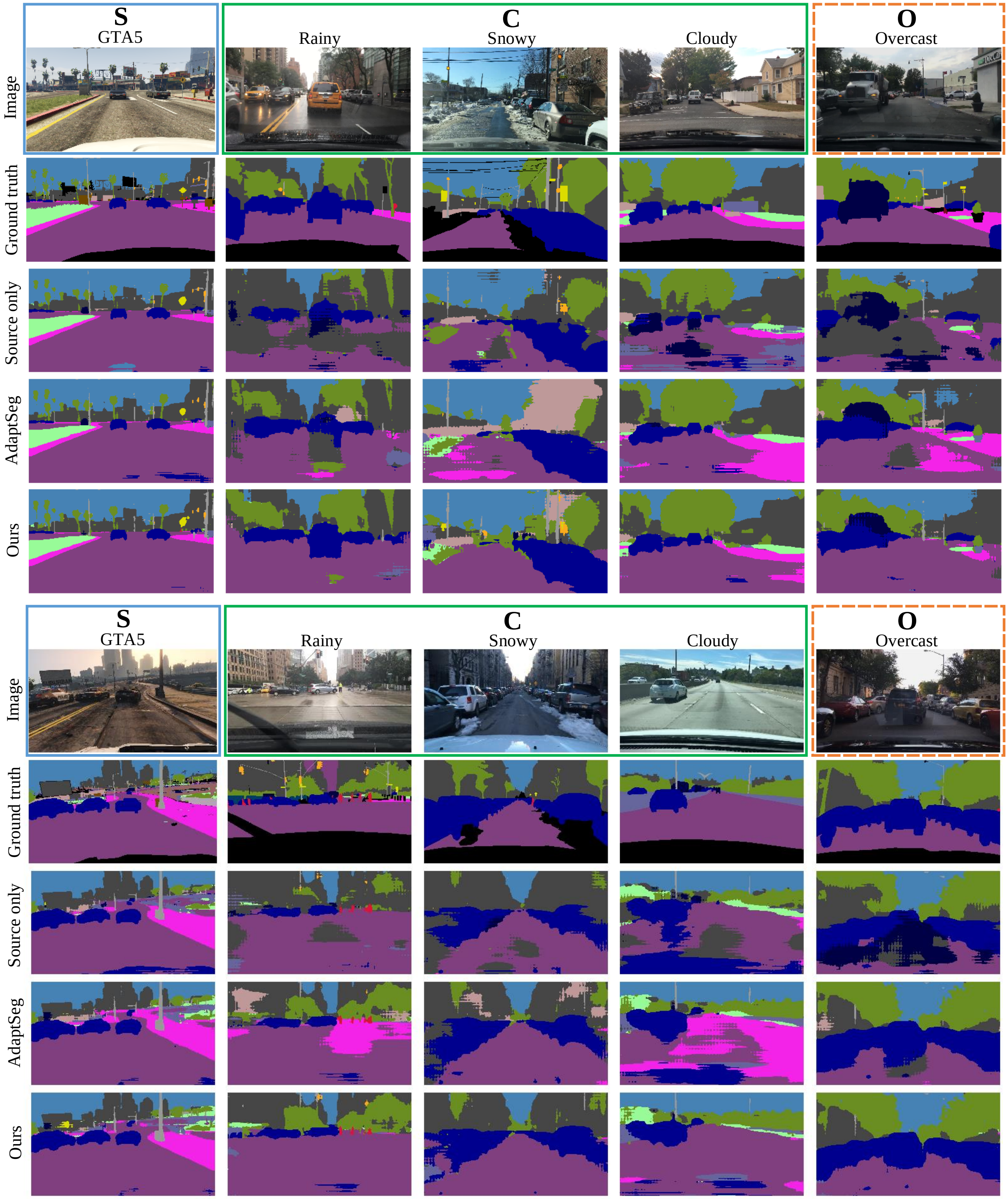


Figure 13: **Qualitative results comparison of semantic segmentation** on the source domain (**S**), the compound domains (**C**), and the open domain (**O**).

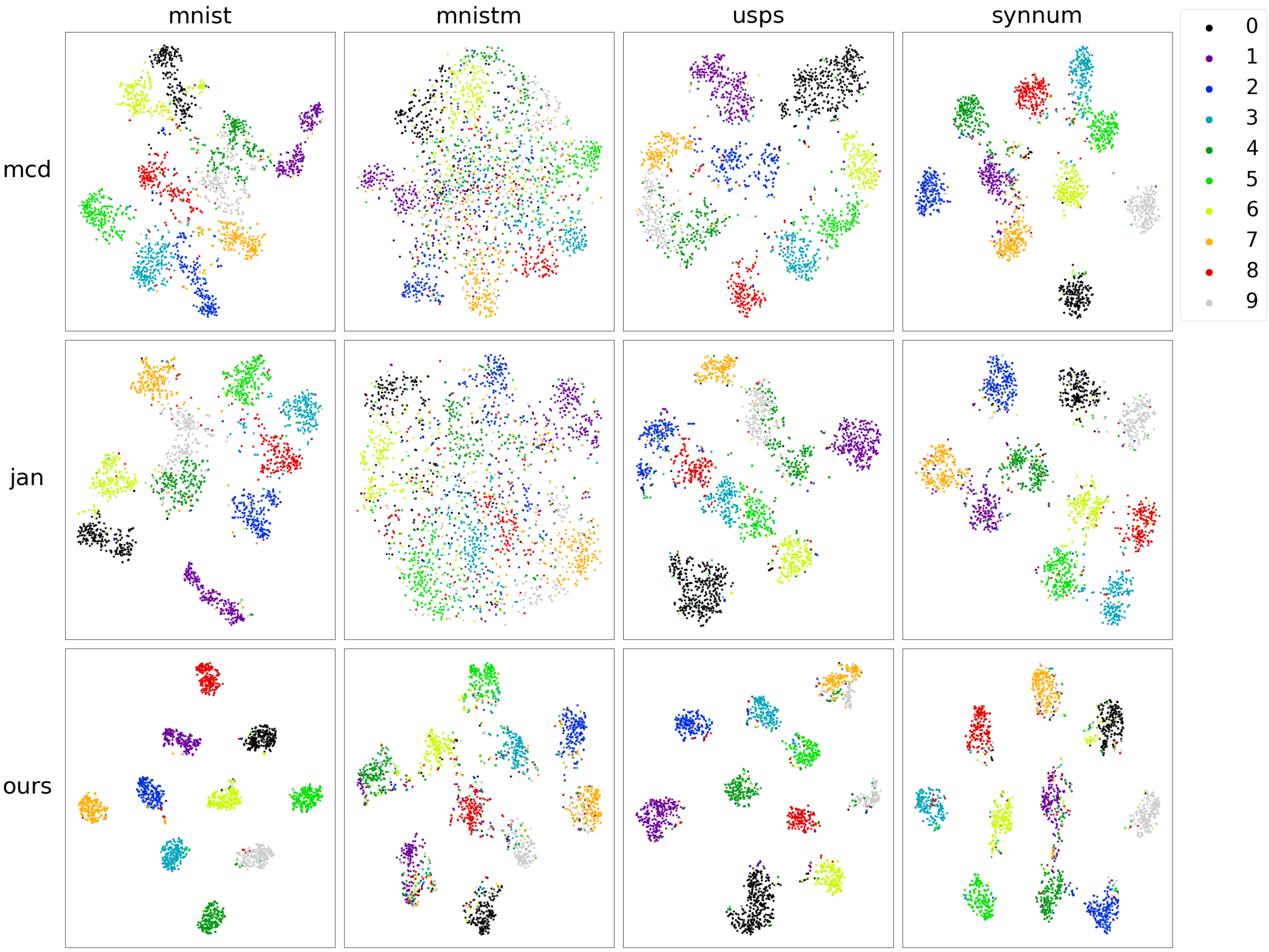


Figure 14: **t-SNE of the C-Digits features** of MCD [42], JAN [30], and our approach.

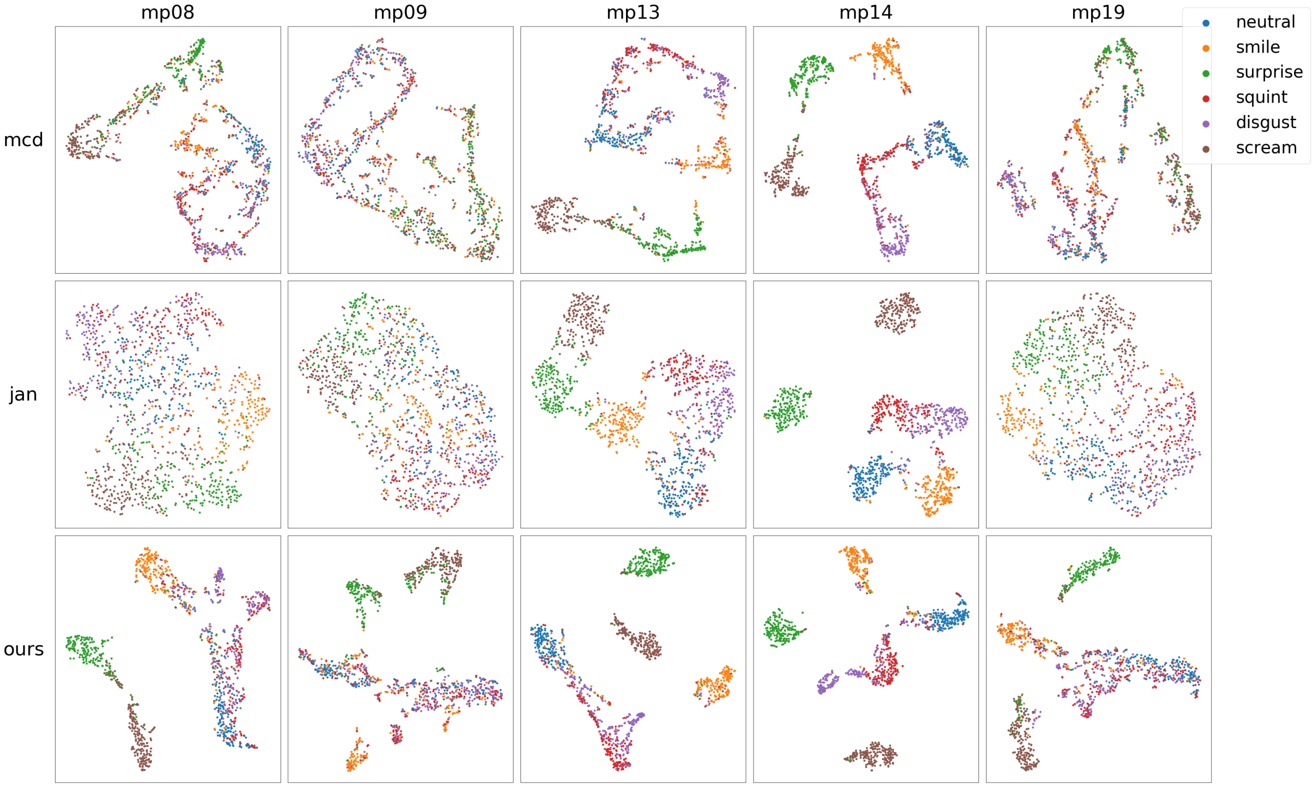


Figure 15: **t-SNE of the C-Faces expression features** of MCD [42], JAN [30], and our approach.

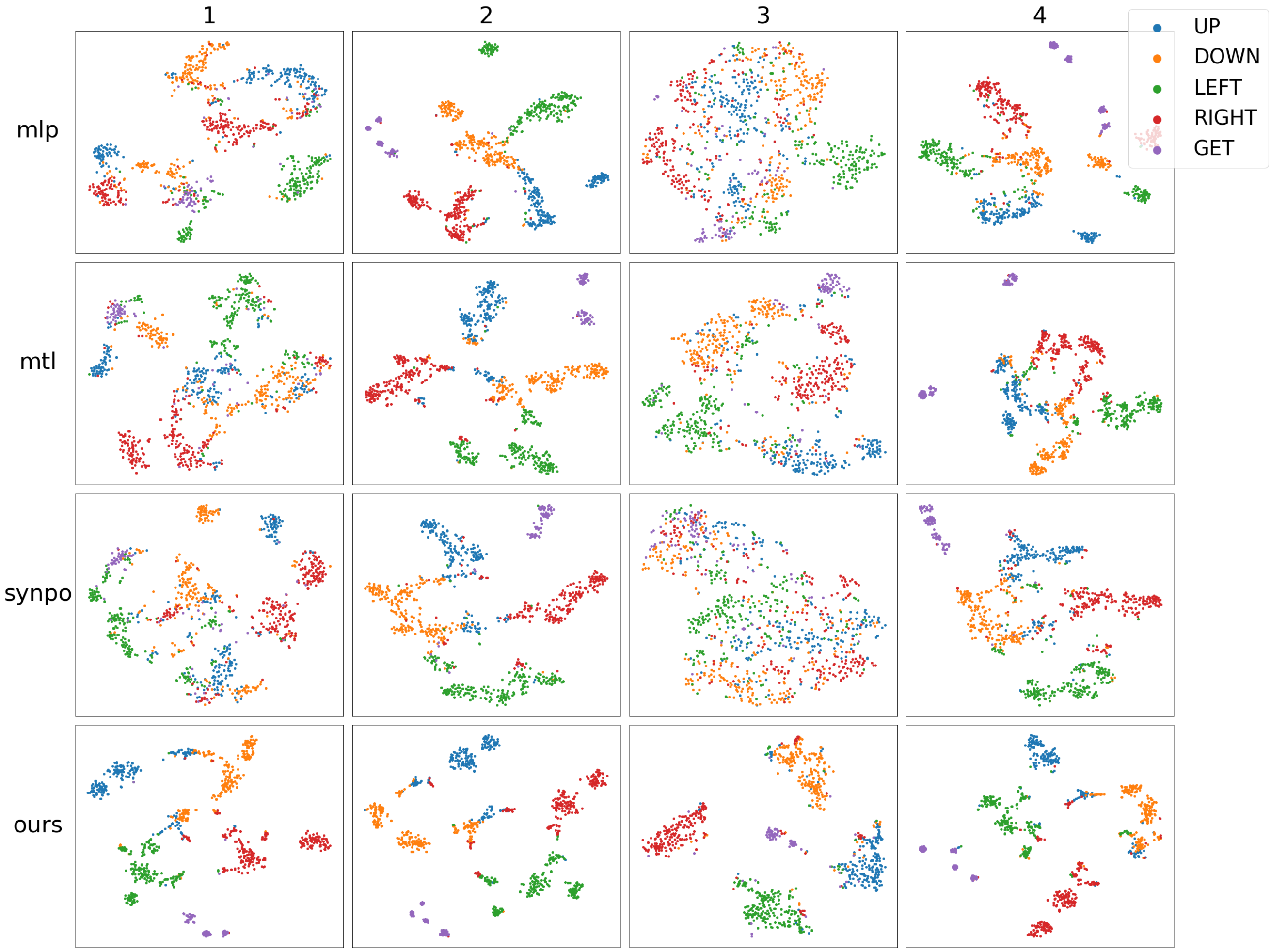


Figure 16: **t-SNE of the C-Maze action features** of MLP [18], MTL, SynPo [18], and our approach.

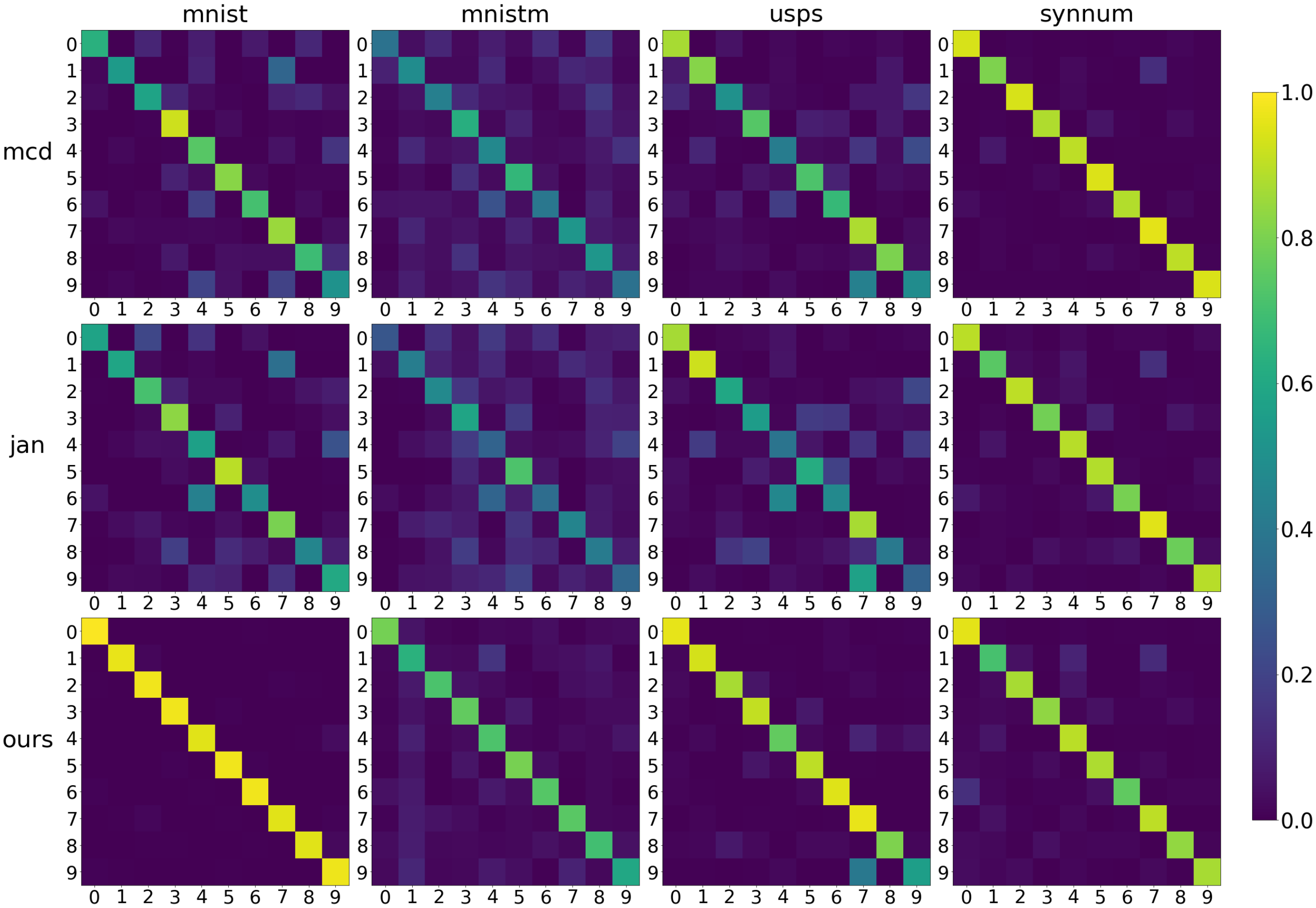


Figure 17: **Confusion matrices visualizations on the C-Digits benchmark** for MCD [42], JAN [30], and our approach.

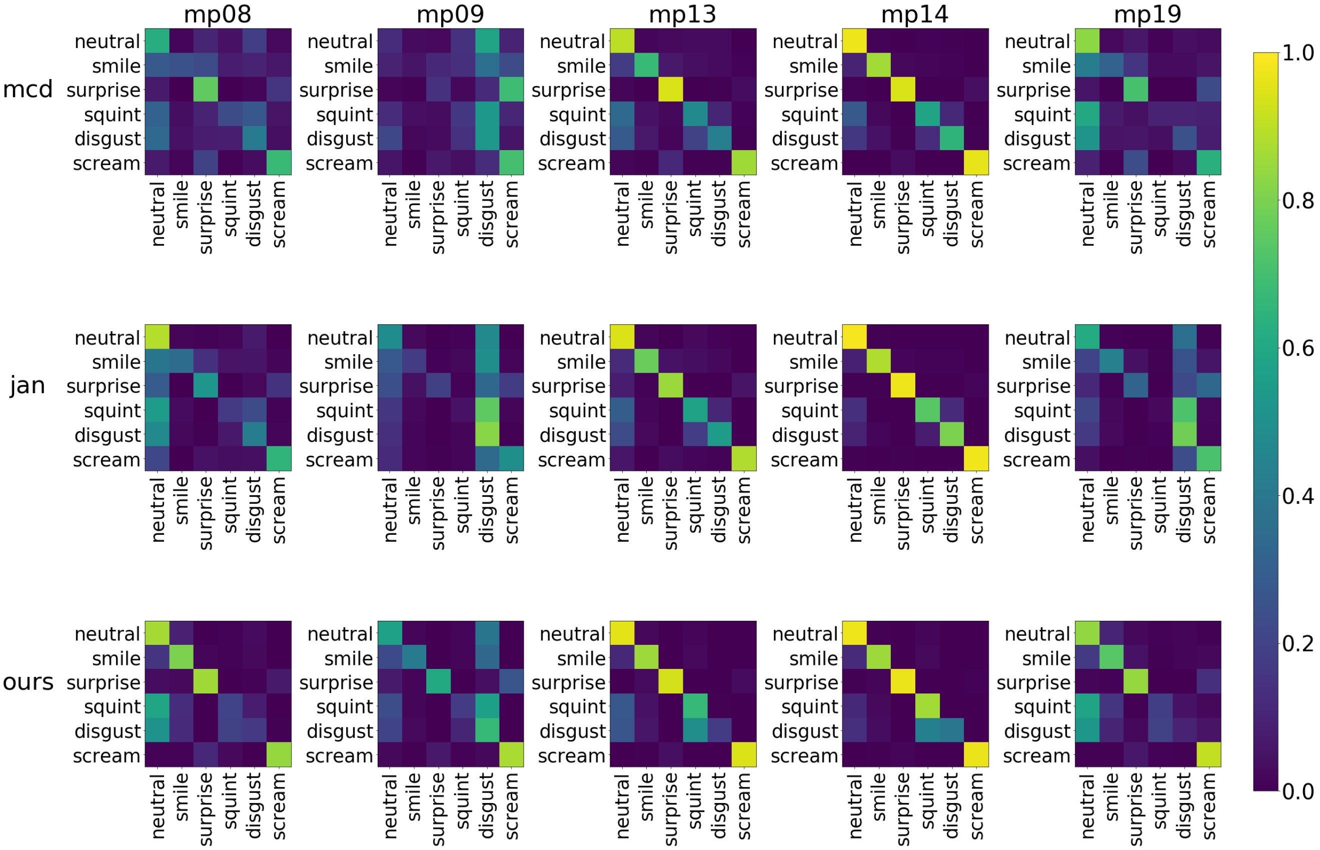


Figure 18: **Confusion matrices visualizations on the C-Faces benchmark** for MCD [42], JAN [30], and our approach.