ABSTRACT

Self-supervised learning (SSL) has recently become the favorite among feature learning methodologies. It is therefore appealing for domain adaptation approaches to consider incorporating SSL. The intuition is to enforce instance-level feature consistency such that the predictions of a model are invariant across domains. However, most existing SSL methods in the regime of domain adaptation usually are treated as standalone auxiliary components, leaving the signatures of domain adaptation unattended. Actually, the optimal region where the domain gap vanishes and the instance level constraint that SSL pursues may not coincide at all. From this point, we present a particular paradigm of self-supervised learning tailored for domain adaptation, i.e., Transferrable Contrastive Learning (TCL), which links the SSL and the desired cross-domain transferability congruently. We find contrastive learning intrinsically a suitable candidate for domain adaptation, as its instance invariance assumption can be conveniently promoted to cross-domain class-level invariance favored by domain adaptation tasks. Based on particular memory bank constructions and pseudo label strategies, TCL then penalizes cross-domain intra-class domain discrepancy between source and target through a clean and novel contrastive loss. The free lunch is, thanks to the incorporation of contrastive learning, TCL relies on a moving-averaged key encoder that naturally achieves a temporally assembled version of pseudo labels for target data, which avoids pseudo label error propagation at no extra cost. TCL therefore efficiently reduces cross-domain gaps. Through extensive experiments on benchmarks (Office-Home, VisDA-2017, Digits-five, PACS and DomainNet) for both single-source and multi-source domain adaptation tasks, TCL has demonstrated state-of-the-art performances.

CCS CONCEPTS
• Computing methodologies → Transfer learning; • Theory of computation → Semi-supervised learning.

KEYWORDS
Domain adaptation; Contrastive learning; Self-supervised learning

1 INTRODUCTION

Deep Neural Networks (DNN) [23, 33] have shown powerful capability of feature learning when trained on large-scale datasets. Conventional supervised learning using DNN frameworks requires enormous annotated data via expensive and time-consuming manual labeling. This has constrained the scalability of DNN vision models, especially when the annotation is unavailable or only limited compared to a large number of parameters constructing the network. Unsupervised Domain Adaptation (UDA) aims to alleviate this problem by making the most out of the available labels and the learned knowledge obtained from a rich-resource domain (i.e., source domain with labeled instances), such that this information can be recycled and applied to understand the scarce-resource domain (i.e., target domain without label annotations). Due to the domain distributional shift between the source domain and target domain, much effort has been made to explicitly reduce domain shift by aligning source and target distributions. A typical example
is [36], where the domain gap is reduced by minimizing the Maximum Mean Discrepancy (MMD) metric between representations of source and target instances (Figure 1(a)).

Inspired by semi-supervised learning methods [51, 56], Self-Supervised Learning (SSL) has become an alternative that pioneers the way of UDA approaches [7, 15, 54, 61]. The basic idea behind these SSL works is to reformulate the UDA task as a semi-supervised learning problem: to perform self-supervised pretext tasks (e.g., rotation prediction) using unlabeled target and labeled source data (Figure 1(b)), and to perform supervised learning on labeled source data. Take for instance, [54, 61] apply 2d rotation to training data, and the encoder has to learn useful semantic features to accomplish the rotation prediction task. Although these seminal SSL based UDA approaches have achieved state-of-the-art performances in comparison to conventional UDA frameworks in Figure 1(a), existing SSL paradigms, unfortunately, tend to ignore the rich information and inherent distributional assumptions hidden behind the domain gaps. Since the domain gap assumption does not necessarily respect the various instance-level hypothesis that conventional SSL desires, there is still plenty of room to improve SSL frameworks when it is associated with UDA tasks.

The above considerations motivate us to tailor brand new SSL feature extraction recipes exclusively for UDA. Accordingly, we present Transferrable Contrastive Learning (TCL), a novel UDA algorithm that unifies and fortifies the strength of both SSL and UDA congruently. Our launching point is to refurbish conventional Contrastive Learning – a recently prevailing SSL framework, such that our novel UDA distributional assumptions, useful prior knowledge in the UDA field, and long-existing old training tricks such as pseudo labeling can all be snugly plugged in with suitable modifications. To this end, we build domain-specific memories: source memory and target memory. The delicate design of these memories are of pivotal importance that validates TCL, and bridges SSL with domain adaptation tasks from there. We define source memory to track source domain instances’ features along with their class information. We also have target memory that records target domain feature along with their “pseudo-labels”. These particular slicing strategies of memories according to their domain and labels (or pseudo-labels) are essential for contrastive learning to rejuvenate in the context of UDA. The constructed domain-specific memories are aimed to offer a bed for later jointly modeling among heterogeneous pretext tasks (e.g., pseudo-labeling and class-level instance discrimination) while the domain gap is simultaneously reduced.

We summarize our contribution succinctly here: we propose a novel self-supervised learning paradigm called TCL. The TCL is a brand new UDA recipe to achieve both inter-class discriminativeness and cross-domain transferability of features through contrastive feature learning, TCL provides a tailored solution that focuses on the effective integration of contrastive learning and UDA via its associated pseudo labeling strategy and memory slicing policies. The TCL framework contrasts with any existing UDA algorithms that combine instance-level invariance rigidly as merely a standalone “icing on the cake” for UDA problems, and TCL discusses how the conventional contrastive learning could be recast exclusively to serve the UDA goals. We demonstrate that TCL achieves state-of-the-art empirical performances on five benchmarks, in regard to both single and multi-source domain adaptation tasks.

2 RELATED WORK

Unsupervised Domain Adaptation methods can be generally categorized into four groups. The first group is the domain discrepancy based approach [28, 36, 38, 39, 65, 67]. Motivated by the theoretical bound proposed in [2, 3], this direction aims at minimizing the discrepancy between the source and target domains. The adopted discrepancy loss varies across different statistic distances metrics (e.g., Maximum Mean Discrepancy (MMD) [36], Jensen-Shannon (JS) Divergence [17] and Wasserstein distance [52]). The second group usually resorts to adversarial learning, which aims to leverage adversarial learning to align two domains on either feature-level [14, 16, 31, 32, 37, 57] or pixel-level [4, 12, 18, 62]. The third group is the pseudo label based approach [5, 9, 15, 19, 27, 45]. Take for example, [9, 45] utilize pseudo labels to estimate class centers in the target domain, which are then used to match each class center across the source domain and target domain. The fourth direction is the recently emerging self-supervised learning based approaches [7, 15, 54, 61]. In Jigsaw [7], the permutation of images crops from each instance are predicted, so that the network improves its semantic learning capability for the UDA task. In addition to single source domain UDA, works also consider the problem of multi-source domain adaptation task [25, 43, 46, 59, 63, 66, 72], which is a more generalized and practical scenario when annotated training data from multiple source domains are accessible. For example, MDAN [72] utilizes adversarial adaptation to induce invariant features for all pairs of source-target domains. P3SDA [46] matches the statistical moments of all source domains and target domain to reduce the domain gap.

Contrastive Learning. Among the recent state-of-the-art self-supervised learning algorithms [1, 8, 10, 11, 22, 24, 42, 68], contrastive learning [6, 21, 34] has offered impressively superior performance, especially when used for pre-training tasks. The key idea of contrastive learning is to achieve instance-level discrimination and invariance, by pushing semantically distinct points away in the feature space, while pulling semantically nearby points closer. Inspired by NCE [20], CPC [42] considers a softmax classifier to distinguish between individual instance classes. As NCE theoretically justified, a large number of instances help to alleviate the learning problem. MoCo [22] thus further pushes the limit of contrastive training via the construction of a momentum updated memory bank, to store past old representations as more as possible. SimCLR [10], SwAV [8] and Moco-v3 [11] provide alternative training techniques that efficiently improve contrastive learning.

In this work, we seek to establish alternative novel symbiosis to combine contrastive learning with UDA task in a more congruent way. Our proposed TCL loss is inspired by contrastive learning but further goes beyond the instance level discrimination in typical contrastive learning. In comparison to previous UDA methods that also utilize SSL, TCL aims to leverage and fortify the functional components of both class-level instance discrimination and domain alignment across different domains, and to achieve “the whole is greater than the sum of its parts” advantage out of the two.

3 APPROACH

In this work, we tailor self-supervised learning (e.g., contrastive learning) for UDA, by coupling SSL pretext tasks and the objective
of domain alignment via a single framework in a contrastive fashion. An overview of our Transferrable Contrastive Learning (TCL) architecture is depicted in Figure 2.

3.1 Preliminaries

Contrastive Learning. In the context of contrastive learning (e.g., MoCo [22]), each image \( x_i \) is considered as an instance, specified by index \( i \). A popular paradigm for contrastive learning is to first produce two randomly generated transformations of \( x_i \) to query image \( x_i^q \) and key image \( x_i^k \), and then consider these two augmentations are coming from the same distribution, also specified by the instance \( i \). The motivation of contrastive learning is to find some instance-invariant encoders so that \( q_i = h_q(x_i^q) \) and \( k_i^+ = h_k(x_i^k) \) are as close as possible. In the meanwhile, the constructed \( h_q \) and \( h_k \) need to keep features \( q_i \) and \( k_i^+ \) discriminative against instances generated from other noise distribution. In this way, the contrastive loss aims to penalize dissimilarity within each positive pair \((q_i, k_i^+)\).

Given features \( k_{i,j}^- \), \( j \in [1, K] \) generated from other distributions, the encoders \( h_q \) and \( h_k \) are also expected to encourage discriminativeness within each negative pair \((q_i, k_{i,j}^-)\). A popular formulation to achieve this goal is to cast the contrastive learning into a classification problem, e.g., as in InfoNCE [42], where the loss classifies positive pairs from \( K \) negative pairs via a softmax function:

\[
L_{NCE}(q_i, k_i^+) = -\log \frac{\exp(q_i, k_i^+)/\tau}{\sum_j \exp(q_i, k_{i,j}^-)/\tau}
\]  

(1)

Here \( \tau \) is the temperature parameter, and \( (q_i, k_i^+) \) denotes the inner product between \( q_i \) and \( k_i^+ \). Different approaches also employ efficient policies for effectively generating negative pairs. For example, in MoCo, the key features are sequentially queued into a “memory bank”. While \( h_q \) is updated via backpropagation, the \( h_k \) is only momentum updated according to the changes of \( h_q \). In this way, the contrastive loss is able to read a large number of negative keys from the slowly evolving memory bank, so that the number of negative pairs are no longer constrained by the batch size.

Problem Formulation of UDA. Consider a source domain having \( N_s \) labeled samples \( S = \{(x_i^s, y_i^s)\}_{i=1}^{N_s} \) distributed across \( C \) classes, where \( y_i^s \in \{1, \ldots, C\} \) is the label of the \( i \)th image \( x_i^s \). In the meanwhile, assume there is also a target domain with \( N_t \) number of unlabeled samples \( T = \{x_i^t\}_{i=1}^{N_t} \) ranging over the same \( C \) as in the source domain, without any annotations though. The goal of UDA is to train a network by exploiting labeled data from source domain \( S \) and unlabeled data from \( T \), so that the obtained model may also be well adapted to the target domain on prediction tasks, even if there is evident and non-negligible domain gap present between the distribution of \( S \) and \( T \).

Notations. We firstly generate two augmentations of each source input \( x_i^s \), i.e., respectively into \( x_i^s,q \) and \( x_i^s,k \). Analogically, we also generate two augmentations of each target data \( x_i^t \), denoted as \( x_i^t,q \) and \( x_i^t,k \). According to the standard paradigms in contrastive learning, we implement a query feature extractor \( z_i^q = f_q(x_i^s,q, \theta_q) \), followed by an additional query projection \( q_i^q = f_q(z_i^q, \beta_q) \in \mathbb{R}^d \), where \( q_i^q \) is the source query vector of the \( i \)th instance. Similarly, we deploy another key feature extractor \( z_i^k = f_k(x_i^s,k, \theta_k) \), followed by a key projection \( k_i^k = f_k(z_i^k, \beta_k) \in \mathbb{R}^d \), where \( k_i^k \) is the source key vector of the \( i \)th instance. In parallel, we also require target query and target key features are generated via exactly the same encoders: i.e., \( q_i^t = f_q(x_i^t,q, \theta_q) \in \mathbb{R}^d \), where \( z_i^t,q = f_q(x_i^t,q, \theta_q) \); and \( k_i^t = f_k(x_i^t,k, \theta_k) \in \mathbb{R}^d \), where \( z_i^t,k = f_k(x_i^t,k, \theta_k) \). We also respectively define query classifier \( g_q(\cdot, \theta_q, c) \) and key classifier \( g_k(\cdot, \theta_k, c) \), where the symbol “-” corresponds to the relevant input depending on the context. Accordingly, the group of query/key feature extractor, projection and classifier can be naturally treated as the query/key encoder. During training, the parameters of query

![Figure 2: An overview of TCL. In addition to conventional supervised learning over labeled source query sample, pseudo-labeling is performed to supervise the prediction of target query sample via Cross-Entropy (CE) loss. TCL loss then minimizes the cross-domain discrepancy between query and positive keys belonging to the same class, while the assumed negative keys from different classes are stored in domain-specific memories. TCL loss is essentially a cross-domain class-level objective that naturally unifies both class-level instance discrimination and domain alignment objectives.](image-url)
encoder \((\Theta_q = (\theta_q, \beta_q, \psi_q))\) are updated via backpropagation according to the loss, whereas the parameters of key encoder \((\Theta_k = (\theta_k, \beta_k, \psi_k))\) are only momentum updated. We defer the parameter update details to the later Section 3.2.

### 3.2 TCL framework

In short, TCL capitalizes on particular class-level invariance assumption and memory bank slicing policy to tackle the UDA problems. These hypotheses are expected to leverage the benefits from both domain adaptation techniques and self-supervised feature learning in a coupled manner. The intuition here is that, we still are able to construct positive and negative pairs across domains based on true/tentative label predictions over source/target data, so that intra-class similarity and domain alignment are learned in a reliable contrastive way. However, since the conventional contrastive learning assumptions purely aim at instance-level invariance, a rigid combination of the two (i.e., UDA and contrastive learning) does not necessarily reduce the degree of freedom of the problem when searching the parameter regions that minimizes the domain gaps. We fix this issue by introducing TCL.

**Pseudo-labeling in Target.** To start with, we define our specific pseudo-labeling method for the unlabeled target domain. The motivation behind this is to promote instance-level invariance intrinsically favored by contrastive learning further to the cross-domain class-level invariance desired by UDA. Let the probability \(p(c|z_i^{s}; q)\) denote the predicted classification distribution across \(c \in [1, \ldots, C]\) classes, given the source query feature \(z_i^{s}; q\). We minimize the standard cross-entropy loss between the prediction and the ground truth label distribution for each source sample:

\[
L_{src} = - \sum_{(s_i, q_i)} y_i^{s} \log p(y_i^{s} = c|Z_i^{s}; q),
\]

where the class prediction is obtained via classifier \(p(y_i^{s} = c|z_i^{s}; q) = g_q(z_i^{s}; q), c\) and \(\psi_q\) is the classifier parameter. Conventionally, the obtained classifier \(g_q(\cdot, \psi_q, c)\) should be able to classify unseen test data from the same distribution as \(S\). In the context of UDA though, it is necessary that \(g_q(\cdot, \psi_q, c)\) is also exposed to target data \(T\), so that the classifier becomes invariant and robust against domain distributional discrepancy.

According to the discussion in the seminal work [2], the domain discrepancy between \(S\) and \(T\) must satisfy the ideal joint hypothesis, so that the domain adaptation problem itself is applicable. If this hypothesis performs poorly, we cannot expect to learn any good target classifier by minimizing source error. Based on this principle, the classification error from directly implementation the classifier on target data is rigorously upper bounded.

The ideal joint hypothesis therefore reassures us that provided \(g_q(\cdot, \psi_q, c)\) is accurate on source data \(S\), the error from implementing \(g_q(\cdot, \psi_q, c)\) directly on target data likely remains acceptable to some extent. Otherwise, the target domain probably is too distinct to adapt for, and any adaptation approaches will be irrelevant. Based on this hypothesis, we label each target data tentatively with the predication \(y_i^{t}\) through the key classifier \(g_k(\cdot, \psi_k, c)\):

\[
y_i^{t} = \arg \max_c g_k(z_i^{t}; k, \psi_k, c),
\]

where \(z_i^{t; k} = \tilde{F}_k(z_i^{t; k}, \theta_k)\). During each iteration of training, the classifier \(g_q(\cdot, \psi_q, c)\) is updated via backpropagation, while \(g_k(\cdot, \psi_k, c)\) is only momentum updated.

Note the unique training mechanism of classifier \(g_k(\cdot, \psi_k, c)\) is critical for our TCL approach to succeed. Mathematically speaking, the momentum updated \(g_k(\cdot, \psi_k, c)\) is equivalently a temporally ensembled version of \(g_q(\cdot, \psi_q, c)\). This implicit averaging strategy effectively reduces the pseudo label error propagation owing to the presence of the domain gap, and efficiently stabilizes the classification on target data. Accordingly, we assign the predicted \(y_i^{t}\) value to be the class "pseudo label" of sample \(x_i^{t; q}\). The standard cross entropy loss \(L_{tar}\) on target data is then computed as the target classification error, i.e.,

\[
L_{tar} = - \sum_{x_i^{t}} \sum_{c} \Psi(y_i^{t} = c|z_i^{t}; q) \log p(y_i^{t} = c|z_i^{t}; q).
\]

Here, the indicator function \(\Psi(x) = 1\) if condition \(x\) holds, otherwise \(\Psi(x) = 0\). Please pay attention to Eq. (4), where the pseudo label prediction \(p(y_i^{t} = c|z_i^{t}; q)\) is computed given the target key features \(z_i^{t; k}\), but used to supervise the target query features update via prediction \(p(y_i^{t} = c|z_i^{t}; q)\). This echoes our motivation that classifier \(g_k(\cdot, \psi_k, c)\) is a temporally ensembled version of \(g_q(\cdot, \psi_q, c)\), whose parameter evolution is more reliable and stable than the conventional pseudo labeling strategy.

**Key Encoder and Domain-specific Memories.** Our memory bank definition is intentionally tailored to serve UDA task. Recall that during each forward pass, we have obtained key features \(k_i^{t} = \tilde{F}_k(z_i^{t; k}, \beta_k)\). We enqueue each \((k_i^{t}, y_i^{t})\) pair sequentially into each of the domain specific source memory bank \(D\) number of banks if multiple sources). Following [22], the spherical K-means is also adopted here to perform the clustering on target samples, which helps further refine the pseudo label prediction \(\hat{y}_i^{t}\). We then sequentially enqueue each batch of target key feature and the corresponding pseudo label \((k_i^{t}, \hat{y}_i^{t})\) into the target memory bank. In regard of memory bank updates, TCL shares similar spirit of [22]: As the training proceeds, the oldest batch of keys in each target/source memory bank are removed, and the current batch of keys are enqueued. But in comparison to conventional contrastive learning, the main goal of TCL is rather to efficiently log both the feature simultaneously with their label/psudo label information in a pairwise manner, which eases our implementation of TCL loss as follows.

**TCL Loss: from Instance-level to Cross-domain Class-level Invariance.** Having all of the notations clarified in the previous sections, we eventually arrive at the transferable contrastive learning loss: TCL loss, our essential adaptation mechanism. The introduction of TCL loss is expected to naturally leverage the heterogeneous self-supervised learning tasks, i.e., instance-level invariance into cross-domain class-level invariance that favored by both contrastive learning [22] and UDA problem itself.

Formally, in a training batch, we define all source query \(q_i^{s}\) that belong to some specific class \(c\) as positive queries to class \(c\). We also consider all the target samples \(k_i^{t}\) currently enqueued in the target memory bank with pseudo label \(\hat{y}_i^{t} = c\) as positive keys of class \(c\). All of the remaining keys in the target memory bank are treated as
negative keys to this specific class $c$. Mathematically, we yield:

$$L(s, t) = - \sum_{c, t, j} \sum_{i} y_i^s \cdot c \cdot \log \frac{\exp(q_i^s, k_j^t) / r}{\exp(q_i^s, k_j^t) / r + \sum_{c, j} \sum_{i} \exp(q_i^s, k_j^t) / r},$$

where class category $c \in [1, C]$ ranges over all existing classes in the current batch. Analogically, we also construct a $L(t, s)$ that is symmetric to the definition of $L(s, t)$:

$$L(t, s) = - \sum_{c, t, j} \sum_{i} y_i^t \cdot c \cdot \log \frac{\exp(q_i^t, k_j^s) / r}{\exp(q_i^t, k_j^s) / r + \sum_{c, j} \sum_{i} \exp(q_i^t, k_j^s) / r}.$$  

The TCL loss boils down to:

$$L_{tcl} = L(t, s) + L(s, t).$$ \hspace{1cm} (7)

It is transparent now that TCL is a cross-domain class-level contrastive loss, an effective extension of the instance-invariance assumption that snugly bridges contrastive learning with UDA. In comparison to conventional instance level contrastive learning as in Eq. (1), the motivation behind Eq. (7) is quite straightforward: contrastive loss penalizes incompatibility of each cross-domain positive pairs $(q_i^s, k_j^t)$ and $(q_i^t, k_j^s)$ that most likely fall into the same category, given the predicted pseudo label on target samples. Eq. (7) also effectively pushes away cross-domain negative pairs if these samples are believed to be coming from distinct class categories. Accordingly, such class-level contrastive learning naturally erases the intra-class feature variance between source and target, while inter-class feature discrimination is further improved.

Extension to Multi-source Domain Adaptation. An immediate extension of our proposal is to jointly apply Eq. (7) across multiple domains. Consider the scenario where we have a dataset composed of $M$ distinct labeled source domains: $S_1, S_2, ..., S_m, ..., S_M$ distributed across classes $c \in [1, C]$. Our goal becomes to best exploit all of these diverse sources and the associated annotations simultaneously so that we can improve the prediction on unlabeled target data $T$. One extra bonus out of the proposed TCL loss is that, Eq. (7) can be conveniently plugged into this multi-source scenario with least modifications:

$$L_{tcl-M} = \sum_{m, n, m, n} L(s_m, s_n) + \sum_{m} L(s_m, t) + \sum_{m} L(t, s_m).$$ \hspace{1cm} (8)

Here, we introduce an extra term $L(s_m, s_n)$ that takes into account the cross source-source domain feature correlations. Loss $L(s_m, s_n)$ basically retains the same form of $L(s, t)$ as in Eq. (5), and only differs in that the positive/negative pair construction for $L(s_m, s_n)$ is completely based on ground truth annotation of cross-source-source domain data, instead of the utility of pseudo labels in $L(s, t)$. Correspondingly, we leverage the cross-domain class-level correlation for each specific class $c$ by capitalizing on all of the annotations available from multiple sources. This potentially imposes stronger cross-domain feature invariance for each class, so that the pseudo label predication on $T$ becomes more reliable.

Overall Objective. To summarize, for multi-source adaptation, we minimize the overall loss $L_{total-M}$ for each training batch:

$$L_{total-M} = L_{src} + L_{tar} + \lambda L_{tcl-M}.$$ \hspace{1cm} (9)

The hyperparameter $\lambda$ trades-off the impact of $L_{tcl-M}$ against the classification loss $L_{src}$ and $L_{tar}$. Note the $L_{tcl-M}$ loss reduces to the plain form of $L_{tcl}$ if and only if there is only a single labeled source domain of data considered, i.e., when $M = 1$. This leads to our overall loss $L_{total-S}$ for single-source adaptation problem:

$$L_{total-S} = L_{src} + L_{tar} + \lambda L_{tcl}.$$ \hspace{1cm} (10)

During the training, the parameters in query and key encoders are updated in order to optimize Eq. (9) or Eq. (10) depending on the actual task. Specifically, the parameters of query encoder $\Theta_q = \{\theta_q, \beta_q, \psi_q\}$ are updated via traditional backpropagation, whereas the parameters of key encoder $\Theta_k = \{\theta_k, \beta_k, \psi_k\}$ is momentum updated as:

$$\Theta_k(it) = \alpha \cdot \Theta_k(it - 1) + (1 - \alpha) \cdot \Theta_q(it - 1),$$ \hspace{1cm} (11)

where $it$ indicates the $it^{th}$ training iteration and $\alpha \in [0, 1)$ is a momentum coefficient.

4 EXPERIMENTS

Datasets. We evaluate the domain adaptation approaches on the following 5 datasets: A) Office-Home [58] is often considered as a challenging dataset for visual domain adaptation tasks, which consists of 15,500 images distributed across $C = 65$ classes. The Office-Home dataset is a composition dataset including four distinct domains: Artistic (Ar), Clip Art (Cl), Product (Pr) and Real-World (Rw). Following [15, 17, 39], we consider single source adaptation on this data. The adaptation directions range over all possible permutations of 2 out of 4 domains. For example, direction $Ar \rightarrow Cl$ means domain Ar is labeled source domain and Cl is the unlabeled target domain. B) VisDA-2017 [47] is a large scale dataset containing 152, 409 synthetic images (Syn) of 12 classes in the source training set and 55,400 real images (Real) from MS-COCO as the target domain validation set. We only consider the single adaption direction Syn $\rightarrow$ Real, following the state-of-the-art approaches. C) Digits-five [63] is a popular benchmark for multi-source domain adaptation (MSDA), which consists of 10 classes of digit images respectively sampled from five different datasets, including MNIST (mt) [29], MNIST-M (mm) [16], SVHN (sv) [41], USPS (up) [26] and Synthetic Digits (sy) [16]. These five datasets essentially represent five distinct domains. Following [46, 63], we construct training/test dataset for MNIST, MNIST-M, SVHN and Synthetic Digits by randomly sampling 25,000 training images/9,000 test images from each of the corresponding domain. For USPS, we used their complete training and test datasets. The adaptation direction on Digits-five is defined as follows. Each individual domain rotates its role as the unlabeled target domain, while the remaining 4 domains are regarded as multiple labeled source domains. Take for instance $\rightarrow sv$ means dataset sv is the target dataset, while all of the remaining domains are considered as the labeled source domains. Therefore, there are in total 5 adaptation directions under this Digits-five dataset. D) PACS [30] is another popular benchmark for MSDA, which is composed of four domains (Art, Cartoon, Photo and Sketch). Each domain includes samples from 7 different categories, including a total of 9,991 samples. E) DomainNet [46] is a more challenging benchmark frequently used for evaluations on MSDA. The DomainNet dataset contains samples from 6 domains: Clipart (clp), Infograph (inf), Painting (pnt), Quickdraw (qdr), Sketch (skt), and Real (rel). Each domain has 345 classes, and the dataset has 596,010 images in total. We use one domain as target and the remaining as sources, as the same setting of [46, 66].
advantages over the existing state-of-the-art methods across all the corresponding to Office-Home Table 1 reports the classification performances of various models on VisDA-2017. As the two domains turn out to be substantially different. This might be a good justification that the integration of SSL pretext tasks and domain alignment target in a contrastive genre is a legitimate solution in the context of domain adaptation. Actually, by merely aligning the data distributions between source and target domains, DAN and JAN demonstrate relatively better performance than source only baselines. In comparison, TCL further stresses the impact of cross-domain inner-class invariance while the inter-class discrimination is effectively preserved, and therefore demonstrates stronger resilience against adaptation directions. Please note that RSDA also benefits from pseudo labeling. Nevertheless, RSDA lacks effective mechanism in preserving the discriminativeness across classes and thus performs inferior to TCL. We highlight the method SSDA, which uses instance level rotation prediction as unsupervised learning in addition to the supervised task. TCL performs drastically worse than our TCL, validating our assumption that instance level self-supervised learning does not necessarily coincide with the optimal parameter region of the supervised task objective.

### Implementation Details
For fair comparison with SOTA approaches across different datasets, we utilize the commonly adopted ImageNet pre-trained ResNet-50/101/18/101 as feature extractor (i.e., $F_{q}/F_{r}$ for Office-Home, VisDA-2017, PACS and DomainNet, respectively. For Digits-five, the feature extractor is composed of three $conv$ layers and two $fc$ layers, which is also exploited in [46]. We adopt task-specific $fc$ layer to parameterize the classifier $g_{q}/g_{r}$. The projection $f_{q}/f_{r}$ is implemented by a $fc$ layer with an output dimension of 256. We finetune from the pre-trained layers and train the newly added layer, where the learning rate of the latter is 10 times that of the former. We use mini-batch SGD with momentum of 0.9 to train the network for all experiments. The initial learning rate is that of the former. We use mini-batch SGD with momentum of $\frac{1}{256}$ for the convolutional layers and $\frac{1}{L}$ for the projection $f_{q}$ and $f_{r}$. The momenta coefficient $L_{tq}$ and $L_{tr}$ in Eq. (5) and Eq. (6) is fixed to 0.05, and momentum coefficient $\alpha$ in Eq. (11) is 0.99.

### 4.1 Performance Comparison

#### Single-Source UDA on Office-Home
Table 1 reports the classification accuracy comparisons on Office-Home dataset. Generally, our TCL exhibits clear advantages over the existing state-of-the-art methods across all the adaptation directions. Please note that TCL is especially effective and commanding on harder transfer tasks, e.g., Pr $\rightarrow$ Cl and Ar $\rightarrow$ Cl, as the two domains turn out to be substantially different. This might be a good justification that the integration of SSL pretext tasks and domain alignment target in a contrastive genre is a legitimate solution in the context of domain adaptation. Actually, by merely aligning the data distributions between source and target domains, DAN and JAN demonstrate relatively better performance than source only baselines. In comparison, TCL further stresses the impact of cross-domain inner-class invariance while the inter-class discrimination is effectively preserved, and therefore demonstrates stronger resilience against adaptation directions. Please note that RSDA also benefits from pseudo labeling. Nevertheless, RSDA lacks effective mechanism in preserving the discriminativeness across classes and thus performs inferior to TCL. We highlight the method SSDA, which uses instance level rotation prediction as unsupervised learning in addition to the supervised task. TCL performs drastically worse than our TCL, validating our assumption that instance level self-supervised learning does not necessarily coincide with the optimal parameter region of the supervised task objective.

#### Single-Source UDA on VisDA-2017
Table 2 displays the performances of various models on VisDA-2017. Similar to the observations on Office-Home, TCL again demonstrates strong superiority over its competitors. Particularly, TCL offers significant performance boost on the classes of “bicycle”, “person” and “truck” in adaptability over its competitors. Particularly, TCL offers significant performance boost on the classes of “bicycle”, “person” and “truck” in

### Implementation Details
For fair comparison with SOTA approaches across different datasets, we utilize the commonly adopted ImageNet pre-trained ResNet-50/101/18/101 as feature extractor (i.e., $F_{q}/F_{r}$ for Office-Home, VisDA-2017, PACS and DomainNet, respectively. For Digits-five, the feature extractor is composed of three $conv$ layers and two $fc$ layers, which is also exploited in [46]. We adopt task-specific $fc$ layer to parameterize the classifier $g_{q}/g_{r}$. The projection $f_{q}/f_{r}$ is implemented by a $fc$ layer with an output dimension of 256. We finetune from the pre-trained layers and train the newly added layer, where the learning rate of the latter is 10 times that of the former. We use mini-batch SGD with momentum of 0.9 to train the network for all experiments. The initial learning rate is that of the former. We use mini-batch SGD with momentum of $\frac{1}{256}$ for the convolutional layers and $\frac{1}{L}$ for the projection $f_{q}$ and $f_{r}$. The momenta coefficient $L_{tq}$ and $L_{tr}$ in Eq. (5) and Eq. (6) is fixed to 0.05, and momentum coefficient $\alpha$ in Eq. (11) is 0.99.

### 4.1 Performance Comparison

#### Single-Source UDA on Office-Home
Table 1 reports the classification accuracy comparisons on Office-Home dataset. Generally, our TCL exhibits clear advantages over the existing state-of-the-art methods across all the
In addition, TCL also consistently demonstrates significant improvements over the existing best state-of-the-art approach. 5 significantly surpasses other baselines by a large margin (around 8%). Here we compare with several methods on multi-source domain adaptation tasks: DANN [16], WBN [40], MCD [50], MDAN [72], JiGen [7], CMSS [66], and M3DA [46]. We directly extract the results of the above methods from their publications or from the reports in [59, 66].

Table 3: Classification accuracy (mean ± std %) on Digits-five and DomainNet datasets.

| Method       | Source Only | MDAN [72] | DCTN [63] | M3DA [46] | MDMA [73] | CMSS [66] | LIC-MSDA [59] | TCL |
|--------------|-------------|------------|------------|-----------|-----------|-----------|--------------|-----|
|              | mm          | mt          | up          | sv         | sy         | clp        | inf          | pnt | qdr | rel | skt | Avg |
|              | 63.4±0.7    | 69.5±0.3    | 70.5±1.2    | 72.8±1.1   | 78.6±0.6  | 95.6±0.8  | 85.6±0.8    |     |     |     |     | 88.3 |
|              | 90.5±0.8    | 98.0±0.9    | 92.8±0.8    | 98.4±0.7   | 99.0±0.4  | 93.0±0.5  | 93.0±0.5    | 96.6 | 99.4 | 93.3 | 91.3 | 97.8 |
|              | 87.7±1.0    | 69.2±0.6    | 77.6±0.4    | 81.3±0.9   | 79.3±0.8  | 83.2±0.6  | 90.0±0.8    | 91.7 | 97.1 | 93.8 | 98.3 | 97.5 |
|              | 84.2±0.6    | 62.4±0.3    | 88.5±0.8    | 89.6±0.6   | 87.9±0.7  | 93.0±0.5  | 87.5±0.8    | 92.8 | 96.6 | 91.6 | 99.2 | 94.6 |
|              | 77.7±0.9    | 74.9±1.1    | 79.1±0.3    | 89.9±0.2   | 88.7±1.0  | 93.9±0.4  | 93.7±0.2    | 97.9 | 90.4 | 89.3 | 89.3 | 90.5 |
|              | 82.4±0.6    | 75.7±1.2    | 76.0±0.7    | 89.7±0.5   | 97.4±0.8  | 93.4±0.4  | 93.7±0.2    | 98.3 | 93.2 | 89.3 | 89.3 | 93.2 |
|              | 89.4±1.2    | 91.8±1.2    | 91.4±0.8    | 97.0±1.5   | 96.4±0.4  | 93.9±3.9  | 97.3±0.3    | 92.7 | 95.9 | 91.6 | 98.5 | 94.6 |
|              | 74.7±1.5    | 74.4±1.0    | 72.0±0.8    | 80.8±1.5   | 83.8±0.5  | 79.1±0.8  | 88.3±2.8    | 89.3 | 89.3 | 88.3 | 89.3 | 89.3 |
|              | 76.6±0.8    | 81.5±1.0    | 79.6±0.8    | 83.8±0.5   | 87.0±0.7  | 89.3±0.4  | 90.5±0.5    | 89.5 | 89.5 | 89.5 | 89.5 | 89.5 |
|              | 76.6±0.8    | 81.5±1.0    | 79.6±0.8    | 83.8±0.5   | 87.0±0.7  | 89.3±0.4  | 90.5±0.5    | 89.5 | 89.5 | 89.5 | 89.5 | 89.5 |

Table 4: Classification accuracy (mean ± std %) on PACS dataset based on ResNet-18.

| Method       | Art         | Cartoon     | Photo       | Sketch     | Avg       |
|--------------|-------------|-------------|-------------|------------|-----------|
| Source Only  | 74.9±0.8    | 72.1±0.7    | 94.5±0.5    | 64.7±1.5   | 76.6      |
| DANN [16]    | 81.9±1.1    | 77.5±1.2    | 91.8±1.2    | 74.6±1.0   | 81.5      |
| MDAN [72]    | 79.1±0.3    | 76.0±0.7    | 91.4±0.8    | 72.0±0.8   | 79.6      |
| WBN [40]     | 89.9±0.2    | 89.7±0.5    | 97.4±0.8    | 58.0±1.5   | 83.8      |
| MCD [50]     | 88.7±1.0    | 88.9±1.5    | 96.4±0.4    | 73.9±3.9   | 87.0      |
| M3DA [46]    | 89.3±0.4    | 89.9±1.0    | 97.3±0.3    | 76.7±2.8   | 88.3      |
| JiGen [7]    | 84.9±2.8    | 81.1±0.1    | 98.0±0.0    | 79.1±0.8   | 85.7      |
| CMSS [66]    | 88.8±0.3    | 90.4±0.8    | 96.9±0.2    | 82.0±0.5   | 89.5      |
| TCL          | 93.6±0.2    | 91.7±0.3    | 98.5±0.1    | 87.5±0.8   | 92.8      |

empirically justifies the advantage of $L_{tcl} - M$ loss defined in Eq. (8), where the joint contribution of multiple source domains has been effectively incorporated into the framework of contrastive learning. As shown in Table 4, our TCL achieves state-of-the-art average accuracy of 92.8% on PACS. On the most challenging → Sketch task, we obtain 87.5%, clearly outperforming other baselines. Note that JiGen is a typical UDA algorithm that utilizes instance level self-supervised learning to improve feature learning. The superiority of TCL over JiGen again justifies advantages of TCL by promoting the instance level assumptions to usefulUDA hypothesis.

Table 3 also displays the adaptation performance of various algorithms on DomainNet. TCL again outperforms the existing works on most of the adaptation tasks, and TCL achieves the SOTA average accuracy of 51.9%. The DomainNet is particularly challenging from two perspectives: Firstly, the domain gap in each adaption category is significant. Secondly, a relatively large number of categories (i.e., 345) has made learning discriminative features much more challenging. Nevertheless, TCL successfully relieves these issues, owing to its unique mechanism in the cross-domain class alignment, and TCL correspondingly demonstrates its outstanding ability to find transferable features on this challenging dataset.

### 4.2 Experimental Analysis

#### Ablation Study

In this section, we investigate the influence of each component in the overall objective $L_{total-S}$ defined in Eq. (10). Table 5 examines the impact of two key components of TCL: the pseudo-labeling loss on target ($L_{tar}$) and transferrable contrastive loss ($L_{tcl}$) in our TCL. The mean accuracy over 12 tasks on Office-Home and the mean accuracy over 12 classes on VisDA-2017 validation set are reported.

| Dataset       | w/o. $L_{tar}$ | w/o. $L_{tcl}$ | TCL   |
|---------------|----------------|----------------|-------|
| Office-Home   | 72.0           | 71.4           | 73.4  |
| VisDA-2017    | 88.7           | 85.5           | 89.6  |

comparison to the existing state of the art methods. When taking a closer inspection, one might observe that the source only model performs extremely poor on these 3 classes. This well verifies the significant distribution gap between source and target domains on these classes, highlighting the efficiency of our TCL. Moreover, compared to CAN that also exploits pseudo-labeling for domain adaptation, our TCL achieves better performances. The reason might be that CAN updates pseudo labels based on iterative learning during each epoch. Instead, TCL stabilizes pseudo-labeling via a temporally encoded encoder, and it enforces inner-class invariance among cross-batch instances. Although the source only baseline seemingly offers the best adaptation result on the “car class”, we conjecture that the model has possibly overfitted to the car class coincidentally in the presence of domain gaps, by sacrificing the bicycle class, person class and etc. From this point, the average score possibly is a more balanced indicator to disambiguate the effectiveness of all algorithms.

### Multi-Source UDA on Digits-five, PACS and DomainNet

Here we compare with several methods on multi-source domain adaptation task: DANN [16], WBN [40], MCD [50], MDAN [72], DCTN [63], M3DA [46], MDMA [73], JiGen [7], CMSS [66], and LIC-MSDA [59]. We directly extract the results of the above methods from their publications or from the reports in [59, 66].

Table 5 illustrates the performance comparisons on the five multi-source adaption directions of Digits-five. Notably, TCL achieves an impressive 96.9% averaged accuracy across 5 directions, significantly surpasses other baselines by a large margin (around 5.1% improvements over the existing best state-of-the-art approach). In addition, TCL also consistently demonstrates distinct advantages in each individual transfer direction, especially on the most challenging direction $\rightarrow$ mm (96.6%), and $\rightarrow$ sv (91.3%) task. This
advantage of our unique strategy of momentum pseudo labeling according to Eq. (4). In addition, if we remove the loss $L_{\text{tar}}$ from Eq. (10), the performance of TCL also degrades by 1.4% and 0.9%. This further supports our assumption that reliable pseudo-label information and class information are critical for the class-level contrastive learning to succeed when embedded in UDA tasks. Without effective learning from the target data, the contrastive loss would be misled by wrong positive pairs, and worsens performance.

**Sensitivity of Trade-off Parameter $\lambda$.** We examine the impact of $\lambda (\lambda \in [0, 1])$ in Eq. (10), a hyperparameter that trades off between the classification loss ($L_{\text{src}}, L_{\text{tar}}$) and TCL loss $L_{\text{tcl}}$. Figure 3 shows that TCL model is relatively robust against the change of $\lambda$, although we observe an evident drop when $\lambda > 0.6$. This probably is attributed to the weaker source domain supervision owing to the large $\lambda$, under which the pseudo label becomes less reliable. The pseudo labels therefore become more noisy and tend to offer wrong class information to the $L_{\text{tar}}$ and $L_{\text{tcl}}$, which further propagate the error. In contrast, if $\lambda$ is too small (e.g., $\lambda < 0.1$), the influence of $L_{\text{tcl}}$ is diluted and TCL loses its unique capability on encouraging cross-domain alignment and inter-class discriminativeness. We observe similar curve pattern on the multi-source adaptation task, and we do not include the result owing to limited space.

**Evaluations of Variants of Contrastive Learning Losses.**

Recall that our proposed TCL loss is a cross-domain class-level contrastive learning objective. To verify the advantage of TCL framework over the conventional instance invariance assumption in the context of UDA, we include two contrastive learning loss variants for comparison. (1) **Instance-level Discrimination Loss (IDL):** We replace class-level invariance loss $L_{\text{tcl}}$ in Eq. (7) directly by conventional instance-level contrastive loss on unlabeled target data, i.e., the IDL framework composes of a conventional instance level InfoNCE [42] in addition to $L_{\text{src}}$ and $L_{\text{tar}}$ in Eq. (10). (2) **Intra-domain Class-level Discrimination Loss (ICDL):** An alternative to perform class-specific contrastive learning is to replace $L_{\text{tcl}}$ by instead penalizing class-level contrastive loss separately for source domain and target domain. We call this framework ICDL. ICDL only considers either intra-source or intra-target class-level discrimination without any cross-domain intra-class constraints. Detailed loss functions of these variants are in supplementary material.

As Table 6 shows, if we replace $L_{\text{tcl}}$ by the IDL loss, the averaged accuracy on Office-Home and VisDA-2017 respectively drops by 2.8% and by 4.5% in comparison to TCL framework. The drop well corroborates our hypothesis in the paper that rigid combination of instance level invariance with UDA task is sub-optimal, as the two assumptions does not necessarily coincide. Note IDL loss only emphasizes instance-level discrimination, and thus introduces noise into the memory bank as the negative keys can even be from the same class of each query instance. IDL loss thus inevitably deviates from the motivation of UDA, which aims to erase inter-domain gap.

Table 6 also shows the performances of our TCL by replacing ICDL objective for domain adaptation. As expected, none of these modifications has improved over original TCL. The main reason is that ICDL objective does not impose any inter-domain invariance within each class, and therefore cannot effectively erase any domain gap as TCL does. Please note that our TCL loss goes beyond merely intra-domain class-level instance discrimination as loss ICDL does, and effectively unifies both class-level instance discrimination and inter domain alignment across different domains.

**Extending TCL loss to MSDA.** As presented in Eq. (8), we extend TCL framework to deal with MSDA tasks. Eq. (8) takes into account the cross source-source domain intra-class feature correlations. In comparison, an alternative is to simply combine multiple source domains as a single source domain and directly perform single source TCL loss in Eq. (7), we call it “TCL-SourceCombine”. As shown in Figure (4), “TCL-SourceCombine” is worse than our TCL, justifying that exploring class-level domain alignment across all domains even in the source domain offers further advantage.

### 5 CONCLUSIONS

In this work, we present Transferrable Contrastive Learning (TCL), which explores domain adaptation in a self-supervised manner. Particularly, we aim to establish a brand new symbiosis that unifies and fortifies self-supervised pretext tasks (pseudo-labeling and class-level instance discrimination) and domain discrepancy minimization in a single framework. By promoting instance level invariance to inter-domain class-level invariance, TCL elegantly couples the contrastive learning strategy with the goal of UDA. Our proposed TCL loss is devised to integrate both class-level instance discrimination and domain discrepancy minimization via contrastive loss, which best screens the semantic compatibility between query-key pairs across different domains. Our TCL can yield state-of-the-art results on five benchmarks for both single-source and multi-source domain adaptation tasks, especially when compared with the conventional instance level contrastive learning baselines.
Acknowledgments. This work was supported in part by NSFC No. 61872329 and the Fundamental Research Funds for the Central Universities under contract WK3490000005.

REFERENCES

[1] Philip Bachman, R Devon Hjelm, and William Buchwalter. 2019. Learning representations by maximizing mutual information across views. In NeurIPS.

[2] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. Machine learning (2010).

[3] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. 2006. Analysis of representations for domain adaptation. In NeurIPS.

[4] Konstantinos Boumalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. 2017. Unsupervised pixel-level domain adaptation with generative adversarial networks. In CVPR.

[5] Qi Cai, Yangwei Pan, Chong-Wah Ngo, Ximnez Tian, Lingyu Duan, and Ting Yao. 2019. Exploring object relation in mean teacher for cross-domain detection. In CVPR.

[6] Qi Cai, Yu Wang, Yangwei Pan, Ting Yao, and Tao Mei. 2020. Joint contrastive learning with infinite possibilities. In NeurIPS.

[7] Fabio M Cariucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. 2019. Domain generalization by solving jigsaw puzzles. In CVPR.

[8] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Samuel Rota Bulo. 2021. Unsupervised learning of visual features by contrasting cluster assignments. In NeurIPS.

[9] Chaog Chen, Weiping Xie, Wenbing Huang, Yu Rong, Xinghao Ding, Yue Huang, Tingyang Xu, and Junzhou Huang. 2019. Progressive feature alignment for unsupervised domain adaptation. In CVPR.

[10] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In ICML.

[11] Xinlei Chen, Saining Xie, and Kaiming He. 2021. An Empirical Study of Training Self-Supervised Visual Transformers. arXiv preprint arXiv:2104.02573 (2021).

[12] Yin Cui, Yingwei Pan, Yehao Li, Qi Cai, Yang Chen, and Ting Yao. 2019. Multi-Source Domain Adaptation and Semi-Supervised Domain Adaptation with Focus on Visual Domain Adaptation Challenge 2019. arXiv preprint arXiv:1910.03548 (2019).

[13] Yingwei Pan, Ting Yao, Yehao Li, Chong-Wah Ngo, and Tao Mei. 2020. Exploring category-agnostic clusters for open-set domain adaptation. In CVPR.

[14] Yang Chen, Yingwei Pan, Ting Yao, Ximnez Tian, and Tao Mei. 2019. Mocycle-gan: Unpaired video-to-video translation. In ACM MM.

[15] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. 2020. Randaugment: A simple data augmentation method that improves both robustness and accuracy. In NeurIPS Workshop.

[16] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748 (2018).

[17] Xinlei Chen, Saining Xie, and Kaiming He. 2021. An Empirical Study of Training Self-Supervised Visual Transformers. arXiv preprint arXiv:2104.02573 (2021).

[18] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748 (2018).

[19] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In ICML.

[20] Xinlei Chen, Saining Xie, and Kaiming He. 2021. An Empirical Study of Training Self-Supervised Visual Transformers. arXiv preprint arXiv:2104.02573 (2021).

[21] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In ICML.

[22] Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. 2019. Domain generalization by solving jigsaw puzzles. In CVPR.

[23] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. Machine learning (2010).

[24] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In ICML.

[25] Xinlei Chen, Saining Xie, and Kaiming He. 2021. An Empirical Study of Training Self-Supervised Visual Transformers. arXiv preprint arXiv:2104.02573 (2021).

[26] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In ICML.

[27] Yingwei Pan, Ting Yao, Yehao Li, Chong-Wah Ngo, and Tao Mei. 2020. Exploring category-agnostic clusters for open-set domain adaptation. In CVPR.

[28] Yingwei Pan, Ting Yao, Yehao Li, Chong-Wah Ngo, and Tao Mei. 2020. Exploring category-agnostic clusters for open-set domain adaptation. In CVPR.

[29] Yingwei Pan, Ting Yao, Yehao Li, Chong-Wah Ngo, and Tao Mei. 2020. Exploring category-agnostic clusters for open-set domain adaptation. In CVPR.

[30] Yingwei Pan, Ting Yao, Yehao Li, Chong-Wah Ngo, and Tao Mei. 2020. Exploring category-agnostic clusters for open-set domain adaptation. In CVPR.

[31] Yingwei Pan, Ting Yao, Yehao Li, Chong-Wah Ngo, and Tao Mei. 2020. Exploring category-agnostic clusters for open-set domain adaptation. In CVPR.
[63] Ruijia Xu, Ziliang Chen, Wangmeng Zuo, Junjie Yan, and Liang Lin. 2018. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In CVPR.

[64] Ruijia Xu, Guanbin Li, Jihan Yang, and Liang Lin. 2019. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In ICCV.

[65] Hongliang Yan, Yukang Ding, Pehua Li, Qihong Wang, Yong Xu, and Wangmeng Zuo. 2017. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In CVPR.

[66] Luyu Yang, Yogesh Balaji, Ser-Nam Lim, and Abhinav Shrivastava. 2020. Curriculum Manager for Source Selection in Multi-Source Domain Adaptation. In ECCV.

[67] Ting Yao, Yingwei Pan, Chong-Wah Ngo, Houqiang Li, and Tao Mei. 2015. Semi-supervised domain adaptation with subspace learning for visual recognition. In CVPR.

[68] Ting Yao, Yiheng Zhang, Zhaofan Qiu, Yingwei Pan, and Tao Mei. 2021. Seco: Exploring sequence supervision for unsupervised representation learning. In AAAI.

[69] Yabin Zhang, Bin Deng, Kui Jia, and Lei Zhang. 2020. Label Propagation with Augmented Anchors: A Simple Semi-Supervised Learning baseline for Unsupervised Domain Adaptation. In ECCV.

[70] Yuchen Zhang, Tianle Liu, Mingleng Long, and Michael I Jordan. 2019. Bridging theory and algorithm for domain adaptation. arXiv preprint arXiv:1904.05801 (2019).

[71] Yabin Zhang, Hui Tang, Kui Jia, and Mingkui Tan. 2019. Domain-symmetric networks for adversarial domain adaptation. In CVPR.

[72] Han Zhao, Shanghang Zhang, Guanhang Wu, José MF Moura, João P Costeira, and Geoffrey J Gordon. 2018. Adversarial multiple source domain adaptation. In NeurIPS.

[73] Sicheng Zhao, Guangzhi Wang, Shanghang Zhang, Yang Gu, Yaxian Li, Zhichao Song, Pengfei Xu, Runbo Hu, Hua Chai, and Kurt Keutzer. 2020. Multi-source Distilling Domain Adaptation. In AAAI.