COLLABORATIVE NORMALIZATION FOR UNSUPERVISED DOMAIN ADAPTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Batch Normalization (BN) as an important component assists Deep Neural Networks achieving promising performance for extensive learning tasks by scaling distribution of feature representations within mini-batches. However, the application of BN suffers from performance degradation under the scenario of Unsupervised Domain Adaptation (UDA), since the estimated statistics fail to concurrently describe two different domains. In this paper, we develop a novel normalization technique, named Collaborative Normalization (CoN), for eliminating domain discrepancy and accelerating the model training of neural networks for UDA. Unlike typical strategies only exploiting domain-specific statistics during normalization, our CoN excavates cross-domain knowledge and simultaneously scales features from various domains by mimicking the merits of collaborative representation. Our CoN can be easily plugged into popular neural network backbones for cross-domain learning. One the one hand, theoretical analysis guarantees that models with CoN promote discriminability of feature representations and accelerate convergence rate; on the other hand, empirical study verifies that replacing BN with CoN in popular network backbones effectively improves classification accuracy with 4% in most learning tasks across three cross-domain visual benchmarks.

1 INTRODUCTION

The hierarchical structure of Deep Neural Networks (DNN) facilitates itself to achieve appealing performances with prolific semantic representations in most learning tasks (Saito et al. (2019); Xu et al. (2019)). As an indispensable component in DNN, batch normalization (BN) aims to scale internal features to promote modeling ability of DNN (Ioffe & Szegedy (2015)). Concretely, typical BN preserves the scale of distribution invariant among various network layers by normalizing features, avoiding gradient vanishing and accelerating model convergence (Li et al. (2018)). To accurately estimate property of distribution, the training of DNN thus requires abundant well-labeled instances, which is unsuitable for real-life scenarios.

Unsupervised Domain Adaptation (UDA) casts a light on such a barren condition and explores external source domain with sufficient annotation to build a model generalized to unlabeled target domain. The primary challenge for UDA is to overcome domain shift that multi-domains belong to various distributions (Zhang et al. (2019a); Ma et al. (2019)). Existing mainstream solutions attempt to eliminate cross-domain discrepancy by learning domain-invariant representations with DNN (Zhang et al. (2019b); Long et al. (2017)). Along this line, one successful strategy adopts adversarial mechanism between feature extractor and domain discriminator to perform domain confusion (Liu et al. (2019); Zhang et al. (2019b)). Other efforts (Long et al. (2017); Kang et al. (2019)) focusing on the alignment of various distributions expect both domains to share the identical statistics (e.g., mean value and co-variance). For the convenient implementation, these works generally bind the corresponding constraints with full-connection features following convolutional operations. However, the effect of objective function flowing in stacked network architecture gradually becomes too weak to align source and target features specially on shallow layers during back propagation.

To overcome such a problem, the variants of BN linking adjacent layers attract massive attentions on domain adaptation. Traditional methods typically adopt domain-specific BNs to separately scale source and target features (Kang et al. (2019); Long et al. (2018)), however, their major drawback lies in the difficulty of capturing cross-domain association. From statistical perspective, adaptive
batch normalization (AdaBN) (Li et al. (2018)) thus exploits the same BN module across various
domains by training normalization component on source domain and fixing parameters to identify
target samples for inference stage. However, significant domain discrepancy has negative influence
on the direct application of source statistics on target domain. To mitigate collision, the automatic
domain alignment layer (Cariucci et al. (2017)) considers the linear combination of source and
target statistics as indicator of BN. The primary challenge is accurately to select the combination
coefficient as the key to succeed. Another perspective delves into the learning of transferable feature
representations. Specifically, TransNorm (TN) claims that the similar convolutional channels across
both domains tend to record similar patterns intensified in BN operation to promote the transferability
of features (Chen et al. (2019)). For visual signals, these attributions, however, are corresponding to
the same concepts such as blue sky, green grass et al. instead of our interested objects. Therefore, the
enhancement of them brings a little benefit for classification of target domain.

Different from their viewpoints, we explore cross-domain feature alignment during forward propa-
gation from manifold distribution perspective (Luo et al. (2020); Fernando et al. (2013)). Although
convolutional representations distribute in high-dimensional feature space, instances from the iden-
tical category lie in the same cluster within each domain. However, domain divergence results in
constituting various subspaces of source and target features with the same annotation. In this paper,
we have alternatively transformed domain adaptation task into subspace fusion problem. Thus, we
propose a novel collaborative normalization (CoN) to answer how to carry out cross-domain subspace
alignment in forward propagation of features. First, CoN module exploits domain-specific statistics
to normalize features to avoid destroying original data distribution. Second, CoN investigates cross-
domain structural information through the global pooling of convolutional features. Finally, CoN
attempts to estimate the location of source (target) features in target (source) subspace and gradually
align samples from its own subspace to the other. The main contributions of our work are summarized
in three folds:

• We advance traditional BN with a novel feature adjustment mechanism in forward propaga-
tion and easily plug our CoN module into convolutional layers without additional parameters.
• Our theoretical analysis further illustrates why our CoN effectively achieves domain align-
ment via translation between source and target samples and accelerates convergence speed.
• Experimental evaluations on several visual domain adaptation benchmarks demonstrate that
our CoN facilitates convolutional neural network to learn better domain-invariant feature
representations than other normalization techniques as traditional BN.

2 RELATED WORK

In this section, we mainly review unsupervised domain adaptation problem and batch normalization
strategies, and highlight the difference between our proposed method.

Unsupervised Domain Adaptation (UDA) aims to train a source-supervised model with high
generalization on target domain. The primary challenges for UDA are to learn transferable feature
and achieve the alignment of distribution. To overcome such issues, domain-adversarial manner is
adopted to train a neural network with generator and discriminator and learn domain-invariant features
(Chen et al. (2019); Zhang et al. (2019b)). Another solution claims that statistics of data reflect
the situation of distribution and forces source and target domains to share the identical indicators
such as MMD and its variants (Long et al. (2015; 2017); Kang et al. (2019)). Both schemes apply
back propagation to delivery the corresponding constraints by using objective function on top layers.
However, gradient vanishing as the increasing number of network layers gradually decrease the
effect of condition on bottom layers. Unlike them, this paper attempts to eliminate cross-domain
discrepancy during forward propagation by proposing a novel network normalization component.

Batch Normalization (BN) as an important component has been widely studied to demonstrate that
it effectively promotes the performance of DNN by scaling internal representations across network
layers (He et al. (2016)). There exist many variants of BN to satisfy specific requirement for other
applications (Cooijmans et al. (2016); Wang et al. (2018); Nam & Kim (2018)). To fight off domain
mismatch, a few works thus explore novel techniques based on domain-specific BN to address issue
of domain adaptation (Li et al. (2018); Wang et al. (2019b); Roy et al. (2019)). AutoDIAL attempts
to construct new statistics through the linear combination of indicators derived from source and
target domains to concurrently normalize all features (Cariucci et al. (2017)). However, the optimal parameters of combination are yet not simply accessible. DSBN Chang et al. (2019) adopts domain-specific BN to scale source and target features to preserve more domain-specific knowledge, which difficulty captures the cross-domain association to achieve distribution alignment. Furthermore, TN points out several channels of convolutional features are more likely to record similar content for both domains and intensifies representation of these channels to promote the transferability of features (Wang et al. (2019a)). The sense of TN is that these similar contents are task-relevant patterns such as objects instead of background, which is difficult to guarantee. Differently, the proposed method considers the translation of sample from one domain to another and carries such motivation into normalization. Such a strategy not only overcomes domain shift, but also promotes the discriminability of features due to advantage of collaborative representation.

3 THE PROPOSED APPROACH

3.1 PRELIMINARIES AND MOTIVATION

Denote a well-labeled source domain \( D_s = \{(X^s_i, Y^s_i)\}_{i=1}^{n_s} \) and a target domain \( D_t = \{X^t_i\}_{i=1}^{n_t} \) without any annotation, where \( X^s_i \in \mathbb{R}^{C \times 1} \) is the label for \( X^s_i \), where \( C \) is the number of category. Unsupervised Domain Adaptation (UDA) aims to borrow knowledge from \( D_s \) to annotate the unlabeled target instances. Benefiting from prolific semantic knowledge learned by hierarchical network architecture, existing explorations for UDA apply DNN to generate domain-invariant features (Zhang et al. (2019b); Tang & Jia (2020)). Without loss of generality, the convolutional features of the \( k \)-th hidden layer are defined as \( F^{s/t}_i \in \mathbb{R}^{L \times W \times H} \) for each mini-batch with \( m \) samples, where \( L, W \) and \( H \) mean the length, width and channel number of feature tensor, respectively.

Domain-specific batch normalizations following convolutional operation are explored to scale representations \( F^s \) and \( F^t \) as Figure 1 (a) in green background with the estimated mean value and co-variance of \( F^{s/t}_j \) for the \( j \)-th channel (\( j \in \{1, 2, \cdots, H\} \)):

\[
\mu^{s/t}_j = \frac{1}{mLW} \sum_{i=1}^{m} \sum_{a=1}^{L} \sum_{b=1}^{W} F^{s/t}_{i,a,b(j)}, \quad \sigma^{s/t}_j = \sqrt{\frac{1}{mLW} \sum_{i=1}^{m} \sum_{a=1}^{L} \sum_{b=1}^{W} \left( F^{s/t}_{i,a,b(j)} - \mu^{s/t}_j \right)^2}. \tag{1}
\]

To improve the modeling capacity of neural network, the transformed representation \( \hat{F}^{s/t}_{i,j} \) from \( F^{s/t}_i \) is further scaled and shifted into the following formulation:

\[
\hat{F}^{s/t}_{i,j} = \frac{F^{s/t}_{i,j} - \mu^{s/t}_j}{\sigma^{s/t}_j}, \quad Y^{s/t}_{i,j} = \gamma_{(j)} \hat{F}^{s/t}_{i,j} + \beta_{(j)}, \tag{2}
\]

where \( \gamma_{(j)} \) and \( \beta_{(j)} \) are learnable parameters. Actually, this operation for each domain effectively scales the feature representations across various network layers to stabilize the model training and accelerate the convergence rate. However, such a normalization strategy using different statistics to scale source and target samples suffers from the difficulty of eliminating domain discrepancy. To handle this bottleneck, TN (Wang et al. (2019a)) concerns on learning transferable features and accelerates typical BN with dashed lines in Figure 1 (a) by sharing information of channel across both domains during forward propagation stage. Concretely, for each channel, TN module computes cross-domain difference \( d_{(j)} \) and evaluates the channel transferability via parameter \( \alpha_{(j)} \):

\[
d_{(j)} = \left| \frac{\mu^{s}_j - \mu^{t}_j}{\sigma^{s}_j} \right|, \quad \alpha_{(j)} = \frac{H_k(1 + d_{(j)})^{-1}}{\sum_{i=1}^{H_k} (1 + d_{(j)})^{-1}}. \tag{3}
\]

According to the above definition, \( \alpha_{(j)} \) with large value indicates the \( j \)-th channels corresponding to two domains contain similar pattern. Promoting the importance of such channels with \( \bar{Y}^{s/t}_i = (1 + \alpha) \odot Y^{s/t}_i \) to some extent reduces domain shift (\( \odot \) denotes element-wise multiplication), where \( \alpha \) is a vector with a concatenation of the values \( \{\alpha_{(j)}\}_{j=1,2,\cdots,H} \). TN assumes that these enhanced features include knowledge of our interested object. Unfortunately, the current version fails to
We have to post two important questions: where is the location of samples in another subspace and how to move it into the position. Before the specific implementation, we firstly reshape hidden feature of target samples as the location of the given source instance in target subspace. For the clarity of generality, we take source-to-target translation as an example and consider the linear combination theory that each sample can be represented by the linear combination of others. Without losing the solution to the first challenge, how to move it into the position.

Different from them, we focus on the manifold distribution of features in forward propagation. For each domain, similar feature representations with the same annotation lie in the identical subspace. However, source and target features from the same category distribute in different subspaces due to considerable domain shift. Alternatively, the reduction of domain discrepancy also means the subspace alignment. Motivated by such a consideration, we expect to estimate the location of source (target) features in target (source) domain and gradually align instances from its own subspace to the other. Therefore, this paper proposes a novel collaborative normalization (CoN) strategy to implement our purpose.

3.2 Collaborative Normalization (CoN)

The CoN module mainly involves three operations: domain-specific normalization, collaborative translation and excavation of cross-domain structural knowledge.

**Domain Specific Normalization.** Domain shift means that source and target instances come from two completely-different distributions. Thus, the normalization of features across both domain with the same statistic easily undermines the original distribution information. To avoid such problem, we follow the usual solution (Ganin et al. (2017)) to scale features with domain-specific statistics and shift them with the identical parameters \((\gamma_{(j)}, \beta_{(j)})\) into \(Y_{s/t}^{i(j)}\).

**Collaborative Translation.** The core of CoN is to achieve the alignment of source and target subspaces by moving instances from its own subspace to the other. Before the specific implementation, we have to post two important questions: where is the location of samples in another subspace and how to move it into the position. **The solution to the first challenge** is motivated by manifold theory that each sample can be represented by the linear combination of others. Without losing generality, we take source-to-target translation as an example and consider the linear combination of target samples as the location of the given source instance in target subspace. For the clarity of illustration, we firstly reshape hidden feature \(Y_{s/t}^{i(j)}\) corresponding to each sample into vector form \(y_{s/t}^{i(j)} \in \mathbb{R}^{1 \times d}\) without \((j)\), where \(d\) is the dimension of feature. Given source sample \(y_{s}^{i}\), target samples \(Z^{t} = (y_{1}^{t})^{T}, (y_{2}^{t})^{T}, \ldots, (y_{m_{t}}^{t})^{T}\) \((y_{i}^{t} \in \mathbb{R}^{1 \times d})\) are regarded as a set of basis vectors to represent it, i.e., \(y_{s}^{i} \approx m_{i}^{t}Z^{t}\), where \(m_{i}^{t} \in \mathbb{R}^{1 \times m_{t}}\) denotes coefficient of linear combination for \(i\)-th source instance and \(m_{t}\) denotes batch size. Specifically, suppose that \(y_{s}^{i}\) and \(y_{j}^{t}\) belong to the same...
category, we should emphasize the contribution of $y^d_j$ for linear representation, which means that the $j$-th element in $m^i_d$ tends to be larger value than others. Similarly, when these two domains change their roles in linear combination, the formulation of collaborative translation will be maintained, i.e., $y^d_i \approx m^i_d Z^s$, where $m^i_d$ has the same meaning with $m^i_s$.

With respect to the second challenge, we are motivated by the explanation that $m^i_d Z^t$ $(m^i_s Z^s)$ serves as projection of $y^d_i$ $(y^s_i)$ on subspace spanned by $Z^t$ $(Z^s)$ and attempt to adjust source feature to the approximation to achieve subspace alignment. However, we further concern about another question about the reliability of adjustment. Alternatively, when there exists small difference between them, performing the corresponding adjustment tends to be confident, vice versa. Thus, we define adjustment coefficient to evaluate the difference between $y^d_i$ and $m^i_d Z^t$ to gradually conduct adjustment:

$$
\eta^d_i = \frac{d}{||y^d_i - m^i_d Z^t||_2}, \quad \eta^s_i = \frac{d}{||y^s_i - m^s_i Z^s||_2},
$$

where $\eta^d_i$ $(\eta^s_i)$ with larger value means the higher credibility of this adjustment. Therefore, the collaborative translations is formulated as: $Y^s = Y^s + \eta^s \circ (A^{st} Z^t - Y^s)$ and $Y^t = Y^t + \eta^t \circ (A^{ts} Z^s - Y^t)$, where $\eta^{s/t} = [\eta^{s/t}_1, \eta^{s/t}_2, \ldots, \eta^{s/t}_{m^i_d}]^T$, $A^{st} = [(m^i_1)^T, (m^i_2)^T, \ldots, (m^i_{m^i_d})]^T$ and $A^{ts} = [(m^s_1)^T, (m^s_2)^T, \ldots, (m^s_{m^s_s})]^T$. The final step is to reshape $Y^{s/t}$ into a 2-D feature map with the same size with $Y^{s/t}_{s/t}$.

**Structural Knowledge Excavation:** The next discussion is about the design of transfer coefficient $A^{st}$ and $A^{ts}$ in Figure 1 (b). Due to the accessibility of source and target features, we learn the closed-form solution of coefficient via the optimization of the ordinary least square between source and the combined features. However, the strategy with optimization operation postpones the forward propagation of features and hardly captures sample-to-sample relationship without sufficient training instances in each mini-batch. To light off these drawbacks, we alternatively turn to the application of cross-domain structural knowledge derived from $F^s$ and $F^t$. Concretely, when channel number of feature $H \geq 2$, the global pooling operation calculates the average of all elements in each 2-D tensor $F^{s/t}_{s/t}$, and then compresses feature of each sample $F^{s/t}_{s/t}$ into $G^{s/t}_{s/t} \in \mathbb{R}^{1 \times H}$. Thus, we formulate the element of $A^{st}$ as $A^{st}_{s/t} = \frac{G^s_i (G^t_i)^T}{||G^s_i||_2 ||G^t_i||_2}$ and $A^{ts} = (A^{st})^T$. For full-connection (FC) layers, the cross-domain graph is directly computed from the cosine distance of source and target features. To this end, we easily plug our CoN layer into any network layers without additional parameters.

### 3.3 Why can CoN work for UDA?

The subspace spanned by target samples $Z^t$ is formulated as $\Phi$ geometrically shown as a plane in Figure 2. Since any source sample $y^s$ can be linearly represented by $Z^t$, we formulate the approximated error between them as $\epsilon = y^s - \tilde{y}^s$, where $\tilde{y}^s = A^{st} Z^t$, $A^{st} \in \mathbb{R}^{1 \times m^i_t}$. Meanwhile, we notice that target samples of the $i$-th category construct a subspace $\Phi_i \in \Phi$ while others form other subspace $\Phi_j \in \Phi$. In this way, the estimated vector $\tilde{y}^s$ can be decomposed into two components: $\chi_i \in \Phi_i$ and $\chi_j \in \Phi_j$. Akin to such decomposition, one component of the approximated error is formulated as vector $\epsilon_i$. In the following, we firstly provide theoretical analysis about our CoN and then illustrate how it does work for UDA.

**Theorem 3.1.** Given target samples divided into $C$ categories and any source sample $y^s$, we assume each class has only one target domain. For $i,j \in \{1, 2, ..., C\}$, where $i \neq j$, the following conclusion holds: \(\exists \epsilon_j\) such that \(||\epsilon_j||_2 \leq ||\epsilon_i||_2\), while $y^s$ belongs to the $i$-th category.

**Proof.** According to the design of CoN, it is straightforward to obtain that $\chi_i = \cos \theta \cdot \chi^i$ and $||\chi_i||_2 = \cos \theta ||y^s_i||_2$. In addition, we have to point out that $\epsilon \perp \Phi$. Due to the auxiliary line $\xi \perp \chi_i$,
we have:

\[ \|\nu\|_2 = \|y^*\|_2 \cdot \cos \theta = \|y^*\|_2 \cdot \cos \rho, \quad \cos \rho = \frac{\|y^*\|_2 \cos \theta}{\|y^*\|_2}. \] (5)

From Figure 2, the formulation \( \|y^*\|_2 \cdot \sin \rho = \|\chi_i\|_2 \cdot \sin \omega \) holds under the law of sines. In terms of these discussions, we further have the following formulation:

\[ \|\chi_i\|_2 = \sqrt{\|y^*\|_2 + \|y^s_i\|_2 \cdot \cos^2 \theta - 2\|y^s_i\|_2 \cdot \|y^s_i\|_2 \cdot \cos \theta \cdot \cos \rho} = \sqrt{\|y^*\|_2 - \cos^2 \theta}. \] (6)

To achieve the final result, we simultaneously shrink vector \( y^s \) and \( y^s_i \) into the same scale, i.e., \( \|y^s\|_2 = \|y^s_i\|_2 = 1 \). From another perspective, we achieve \( \|\epsilon_i\|_2^2 = \|\epsilon\|_2^2 + \|\chi_i\|_2^2 \) and rewrite it as:

\[ \|\epsilon_i\|_2^2 = \|\epsilon\|_2^2 + \|y^s\|_2^2 - \cos^2 \theta. \] (7)

In terms of Eq. (7), the approximated error \( \|\epsilon_i\|_2 \) depends on three terms \( \epsilon, \ y^s \) and \( \cos \theta \). From manifold perspective, features from the same category lie in a very compact subspace. Thus, the cosine similarity between \( y^s \) and \( y^s_i \) both from the \( i \)-th category becomes higher than the distance between \( y^s \) and \( y^s_i \) from various classes. With the aid of manifold theory, the conclusion \( /2\epsilon_j \), \( \|\epsilon_j\|_2 \leq \|\epsilon_i\|_2 \) holds. Next, we utilize Eq. (7) to illustrate why our CoN can achieve domain alignment. According to CoN module \( \hat{y}^s = y^s + \eta' (\hat{y}^s - y^s) \), \( y^s \) is transformed into \( \hat{y}^s \) and the component of approximated error \( \epsilon_i \) tends to be \( \hat{\epsilon}_i \), where \( \|\epsilon_i\|_2 = \|\epsilon\|_2 + \|\chi_i\|_2 \) and \( \delta = \eta (\hat{y}^s - y^s) \). Therefore, we have the following inequality over \( \epsilon_i \) and \( \hat{\epsilon}_i \):

\[ \|\epsilon_i\|_2^2 = \|\epsilon\|_2^2 + \|\delta\|_2^2 - 2\|\epsilon\|_2 \|\delta\|_2 \leq \|\epsilon_i\|_2^2, \quad \|\delta\|_2 \leq \|\epsilon\|_2. \] (8)

The above formulation denotes the adjusted features \( \hat{y}^s \) is closer to its corresponding target category when compared with the original feature \( y^s \). Therefore, our method utilizes such a reliable adjustment to gradually achieve distribution alignment and improve the discriminative ability of features.

4 EXPERIMENTS

To verify the effectiveness of Collaborative Normalization (CoN), we apply the proposed method into two well-known deep transfer learning backbones CDAN (Long et al. (2018)) and DANN (Ganin et al. (2017)) and evaluate their performance on three popular benchmark datasets.

4.1 EXPERIMENTAL SETTING

Datasets: 1) Image-CLEF collects visual signals from three subsets: Caltech-256 (C), ImageNet ILSVRC 2012 (I) and Pascal VOC 2012 (P) with the same number of samples. Concretely, arbitrary subset includes 600 images evenly distributed in 12 categories. 2) Office-31 (Saenko et al. (2010)) as a benchmark dataset of domain adaptation involves 4,652 images from 31 categories. These instances are divided into three subsets: Amazon (A, 2,817 images), DSLR (D, 498 images) and Webcam (W, 795 images). 3) Office-Home (Venkateswara et al. (2017)) consists of four subsets: Artistic images (Ar), Clip Art (Cl), Product images (Pr) and Real-World images (Rw). Four subsets with 15,500 images share the identical label space of 65 categories.

Competitive baselines: We not only explore the state-of-the-art domain adaptation methods including DAN (Long et al. (2015)), JAN (Long et al. (2017)), MADA (Pei et al. (2018)), DSR (Cai et al. (2019)), SymNets (Zhang et al. (2019b)), TADA (Wang et al. (2019b)), SAFN (Xu et al. (2019)), DRMEA (Luo et al. (2020)), DADA (Tang et al. (2020)) but also combine network backbones (CDAN and DANN) with multi-norm strategies: BN (Long et al. (2018)) and TN (Wang et al. (2019a)) as baselines. We follow the standard protocols operated with CDAN and DANN to evaluate the effectiveness of our proposed normalization technique. To make fair comparisons, results of all above methods are directly copied from the corresponding literature under the exactly same protocols.

Experimental results on Image-CLEF, Office-31 and Office-Home datasets are summarized in Tables 1, 2, and 3, respectively. From these results, we achieve three main conclusions. First, the integration of CDAN and CoN surpasses all comparisons in most domain adaptation tasks.
while the highest accuracy of three normalization tools is in Red type. And BN, TN and CoN are plugged into the same backbone CDAN.

Table 1: Classification Accuracy (%) on Image-CLEF dataset (ResNet-50). The best results among all methods are shown with **underline** while the highest accuracy of three normalization tools is in Red type. And BN, TN and CoN are plugged into the same backbone CDAN.

<table>
<thead>
<tr>
<th>Method</th>
<th>Res-Net</th>
<th>JAN</th>
<th>DAN</th>
<th>MADA</th>
<th>SAFN</th>
<th>DRMEA</th>
<th>BN</th>
<th>TN</th>
<th>CoN</th>
</tr>
</thead>
<tbody>
<tr>
<td>I→P</td>
<td>74.8±0.3</td>
<td>76.8±0.4</td>
<td>74.5±0.4</td>
<td>75.0±0.3</td>
<td>79.3±0.1</td>
<td>80.7</td>
<td>77.7</td>
<td>78.3</td>
<td><strong>80.2</strong></td>
</tr>
<tr>
<td>P→I</td>
<td>83.9±0.1</td>
<td>88.0±0.2</td>
<td>82.2±0.2</td>
<td>87.9±0.2</td>
<td>93.3±0.4</td>
<td>92.5</td>
<td>90.7</td>
<td>90.8</td>
<td><strong>93.3</strong></td>
</tr>
<tr>
<td>I→C</td>
<td>91.5±0.3</td>
<td>94.7±0.2</td>
<td>92.8±0.2</td>
<td>96.0±0.3</td>
<td>96.3±0.4</td>
<td>97.2</td>
<td><strong>97.7</strong></td>
<td>96.7</td>
<td>97.5</td>
</tr>
<tr>
<td>C→I</td>
<td>78.0±0.2</td>
<td>89.5±0.3</td>
<td>86.3±0.4</td>
<td>88.8±0.3</td>
<td>91.7±0.0</td>
<td>90.5</td>
<td>91.3</td>
<td>92.3</td>
<td><strong>94.3</strong></td>
</tr>
<tr>
<td>C→P</td>
<td>65.5±0.3</td>
<td>74.2±0.3</td>
<td>69.2±0.4</td>
<td>75.2±0.2</td>
<td>77.6±0.1</td>
<td>77.7</td>
<td>74.2</td>
<td>78.0</td>
<td><strong>80.4</strong></td>
</tr>
<tr>
<td>P→C</td>
<td>91.2±0.3</td>
<td>91.7±0.3</td>
<td>89.8±0.4</td>
<td>92.2±0.3</td>
<td>95.3±0.1</td>
<td>96.1</td>
<td>94.3</td>
<td>94.3</td>
<td><strong>96.2</strong></td>
</tr>
<tr>
<td>Avg</td>
<td><strong>80.7</strong></td>
<td><strong>85.8</strong></td>
<td><strong>82.5</strong></td>
<td><strong>85.8</strong></td>
<td><strong>88.9</strong></td>
<td><strong>89.1</strong></td>
<td><strong>87.7</strong></td>
<td><strong>88.5</strong></td>
<td><strong>90.3</strong></td>
</tr>
</tbody>
</table>

Table 2: Classification Accuracy (%) on Office-31 dataset (ResNet-50). The best results among all methods are shown with **underline** while the highest accuracy of three normalization tools is in Red type. And BN, TN and CoN are plugged into the same backbone CDAN.

<table>
<thead>
<tr>
<th>Method</th>
<th>Res-Net</th>
<th>JAN</th>
<th>DADA</th>
<th>SymNets</th>
<th>TADA</th>
<th>SAFN</th>
<th>BN</th>
<th>TN</th>
<th>CoN</th>
</tr>
</thead>
<tbody>
<tr>
<td>A→W</td>
<td>68.4±0.2</td>
<td>85.4±0.3</td>
<td>92.3±0.1</td>
<td>90.8±0.1</td>
<td>94.3±0.3</td>
<td>90.3</td>
<td>94.1</td>
<td>95.7</td>
<td><strong>96.4</strong></td>
</tr>
<tr>
<td>D→W</td>
<td>96.7±0.1</td>
<td>97.4±0.2</td>
<td>99.2±0.1</td>
<td>98.8±0.3</td>
<td>98.7±0.1</td>
<td>98.7</td>
<td>98.6</td>
<td>98.7</td>
<td><strong>98.4</strong></td>
</tr>
<tr>
<td>W→D</td>
<td>99.3±0.1</td>
<td>99.8±0.2</td>
<td><strong>100±0.0</strong></td>
<td>100.0±0.0</td>
<td>99.8±0.2</td>
<td>100.0</td>
<td><strong>100.0</strong></td>
<td><strong>100.0</strong></td>
<td><strong>100.0</strong></td>
</tr>
<tr>
<td>A→D</td>
<td>68.9±0.2</td>
<td>84.7±0.3</td>
<td>93.9±0.2</td>
<td>93.9±0.5</td>
<td>91.6±0.3</td>
<td>90.7</td>
<td><strong>92.9</strong></td>
<td>94.0</td>
<td><strong>96.0</strong></td>
</tr>
<tr>
<td>D→A</td>
<td>62.5±0.3</td>
<td>68.6±0.3</td>
<td>74.4±0.1</td>
<td>74.6±0.6</td>
<td>72.9±0.2</td>
<td>73.4</td>
<td>71.0</td>
<td>73.4</td>
<td><strong>77.3</strong></td>
</tr>
<tr>
<td>W→A</td>
<td>60.7±0.3</td>
<td>70.0±0.4</td>
<td>74.2±0.1</td>
<td>72.5±0.5</td>
<td>73.0±0.3</td>
<td>71.2</td>
<td>69.3</td>
<td>74.2</td>
<td><strong>75.7</strong></td>
</tr>
<tr>
<td>Avg</td>
<td><strong>76.1</strong></td>
<td><strong>84.3</strong></td>
<td><strong>89.0</strong></td>
<td><strong>88.4</strong></td>
<td><strong>88.4</strong></td>
<td><strong>87.6</strong></td>
<td><strong>87.7</strong></td>
<td><strong>89.3</strong></td>
<td><strong>90.6</strong></td>
</tr>
</tbody>
</table>

Specifically, with respect to tasks $D \rightarrow A$ and $W \rightarrow A$ in Office-31, our proposed strategy separately exceeds the second highest classification accuracy by 2.9% and 1.5%. It demonstrates that collaborative normalization effectively promotes the model generalization ability on target domain. Second, compared with other normalization techniques (BN and TN), CoN successfully scales latent features across various domains to achieve the alignment of different distributions. For example, due to the assistance of CoN, CDAN learns more transferable features which dramatically eliminates domain shift and improves accuracy by 6% and 4.3% on tasks $(C_I \rightarrow A_r)$ and $(C_I \rightarrow P_r)$ when making comparison with BN. Finally, network backbones associated with CoN become more robust for several challenging situations where there exists huge discrepancy about the number of sample within source and target domains. Although DSLR or Webcam domain has less samples than Amazon domain in Office-31, the proposed method still obtains comparable performance, which verifies that the application of collaborative transfer in mini-batch tends to capture more cross-domain information and learn better domain-invariant features. Moreover, CoN layers are also plugged into DANN architecture, which still achieves promising performances in Figure 3. That means it is simple yet effective to deploy our proposed normalization strategy to any frameworks used for domain adaptation.

4.2 Empirical Analysis

Convergence Speed: According to the aforementioned working mechanism, the proposed collaborative normalization adaptively adjusts the direction of feature representations within a mini-batch. Such operation explores cross-domain knowledge to gradually eliminate the discrepancy between source and target domains. Importantly, our strategy effectively accelerates the convergence speed. To clearly illustrate this point, the iterative procedures of CDAN with CoN on tasks $P \rightarrow I$ and $C \rightarrow P$ are reported in Figure 5 (a). We easily find that compared to BN and TransNorm, our method with CDAN rapidly achieves the optimal solution. Concretely, take task $P \rightarrow I$ as an example, collaborative transfer strategy only costs 2,000 iterations to reach the highest classification accuracy,
Table 3: Classification Accuracy (%) on Office-Home dataset (ResNet-50). The best results among all methods are shown with **underline** while the highest accuracy of three normalization tools is in **Red** type.

<table>
<thead>
<tr>
<th>Method</th>
<th>Ar:Cl</th>
<th>Ar:Pr</th>
<th>Ar:Rw</th>
<th>Cl:Ar</th>
<th>Cl:Pr</th>
<th>Cl:Rw</th>
<th>Pr:Ar</th>
<th>Pr:Cl</th>
<th>Pr:Rw</th>
<th>Rw:Ar</th>
<th>Rw:Cl</th>
<th>Rw:Pr</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Res-Net</td>
<td>34.9</td>
<td>50.0</td>
<td>58.0</td>
<td>37.4</td>
<td>41.9</td>
<td>46.2</td>
<td>38.5</td>
<td>31.2</td>
<td>60.4</td>
<td>53.9</td>
<td>41.2</td>
<td>59.9</td>
<td>46.1</td>
</tr>
<tr>
<td>JAN</td>
<td>45.9</td>
<td>61.2</td>
<td>68.9</td>
<td>50.4</td>
<td>59.7</td>
<td>61.0</td>
<td>45.8</td>
<td>43.4</td>
<td>70.3</td>
<td>63.9</td>
<td>52.4</td>
<td>76.8</td>
<td>58.3</td>
</tr>
<tr>
<td>DSR</td>
<td>53.4</td>
<td>71.6</td>
<td>77.4</td>
<td>57.1</td>
<td>66.8</td>
<td>69.3</td>
<td>56.7</td>
<td>49.2</td>
<td>75.7</td>
<td>68.0</td>
<td>54.0</td>
<td>79.5</td>
<td>64.9</td>
</tr>
<tr>
<td>SymNets</td>
<td>53.1</td>
<td>72.3</td>
<td>77.2</td>
<td>59.1</td>
<td>71.2</td>
<td>72.1</td>
<td>59.7</td>
<td>53.1</td>
<td>78.4</td>
<td>72.4</td>
<td>60.0</td>
<td>82.9</td>
<td>67.6</td>
</tr>
<tr>
<td>TADA</td>
<td>52.0</td>
<td>71.7</td>
<td>76.3</td>
<td>57.6</td>
<td>70.0</td>
<td>70.0</td>
<td>57.4</td>
<td>50.9</td>
<td>77.3</td>
<td>70.9</td>
<td>57.1</td>
<td>81.5</td>
<td>67.3</td>
</tr>
<tr>
<td>SAFN</td>
<td>50.7</td>
<td>70.6</td>
<td>76.0</td>
<td>57.6</td>
<td>70.0</td>
<td>70.0</td>
<td>57.4</td>
<td>50.9</td>
<td>77.3</td>
<td>70.9</td>
<td>57.1</td>
<td>81.5</td>
<td>67.3</td>
</tr>
<tr>
<td>CDAN+BN</td>
<td>50.2</td>
<td>71.4</td>
<td>77.4</td>
<td>59.3</td>
<td>72.7</td>
<td>73.1</td>
<td>61.0</td>
<td>53.1</td>
<td>79.5</td>
<td>71.9</td>
<td>59.0</td>
<td>82.9</td>
<td>67.6</td>
</tr>
<tr>
<td>CDAN+TN</td>
<td>51.2</td>
<td>72.2</td>
<td>77.5</td>
<td>63.6</td>
<td>74.3</td>
<td>72.0</td>
<td>61.9</td>
<td>56.5</td>
<td>79.8</td>
<td>75.4</td>
<td>55.6</td>
<td>84.2</td>
<td>68.7</td>
</tr>
<tr>
<td>CDAN+CoN</td>
<td>52.0</td>
<td>71.7</td>
<td>76.3</td>
<td>64.2</td>
<td>71.2</td>
<td>72.1</td>
<td>59.7</td>
<td>53.1</td>
<td>78.4</td>
<td>72.4</td>
<td>60.0</td>
<td>82.9</td>
<td>67.6</td>
</tr>
</tbody>
</table>

Table 3: Classification Accuracy (%) on Office-Home dataset (ResNet-50). The best results among all methods are shown with **underline** while the highest accuracy of three normalization tools is in **Red** type.

```
Method | Ar:Cl | Ar:Pr | Ar:Rw | Cl:Ar | Cl:Pr | Cl:Rw | Pr:Ar | Pr:Cl | Pr:Rw | Rw:Ar | Rw:Cl | Rw:Pr | Avg  |
-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|------|
Res-Net | 34.9  | 50.0  | 58.0  | 37.4  | 41.9  | 46.2  | 38.5  | 31.2  | 60.4  | 53.9  | 41.2  | 59.9  | 46.1 |
JAN     | 45.9  | 61.2  | 68.9  | 50.4  | 59.7  | 61.0  | 45.8  | 43.4  | 70.3  | 63.9  | 52.4  | 76.8  | 58.3 |
DSR     | 53.4  | 71.6  | 77.4  | 57.1  | 66.8  | 69.3  | 56.7  | 49.2  | 75.7  | 68.0  | 54.0  | 79.5  | 64.9 |
SymNets | 53.1  | 72.3  | 77.2  | 59.1  | 71.2  | 72.1  | 59.7  | 53.1  | 78.4  | 72.4  | 60.0  | 82.9  | 67.6 |
TADA    | 52.0  | 71.7  | 76.3  | 57.6  | 70.0  | 70.0  | 57.4  | 50.9  | 77.3  | 70.9  | 57.1  | 81.5  | 67.3 |
SAFN    | 50.7  | 70.6  | 76.0  | 57.6  | 70.0  | 70.0  | 57.4  | 50.9  | 77.3  | 70.9  | 57.1  | 81.5  | 67.3 |
CDAN+BN | 50.2  | 71.4  | 77.4  | 59.3  | 72.7  | 73.1  | 61.0  | 53.1  | 79.5  | 71.9  | 59.0  | 82.9  | 67.6 |
CDAN+TN | 51.2  | 72.2  | 77.5  | 63.6  | 74.3  | 72.0  | 61.9  | 56.5  | 79.8  | 75.4  | 55.6  | 84.2  | 68.7 |
CDAN+CoN| 52.0  | 71.7  | 76.3  | 64.2  | 71.2  | 72.1  | 59.7  | 53.1  | 78.4  | 72.4  | 60.0  | 82.9  | 67.6 |
```

Figure 4: Visualization of features on task $C \rightarrow P$ by using CDAN with multi-normalization tools.

while TN and BN require 3,500 and 4,500 iterations, respectively. The main reason results from the application of cross-domain knowledge which guides features to quickly shrink to the correct direction and makes features more discriminative.

**Feature Visualization & Singular Values:** To explicitly understand the situation of distribution in abstract semantic space, the t-SNE technique is exploited to visualize feature representations in 2D-panel. The comparative experiments among BN, TN and CoN with CDAN are performed on task $C \rightarrow P$ of Image-CLEF. Different from BN and TN, there exists tangible boundary among various categories generated by CoN (Figure 4 (c)). And it is difficult to distinguish source samples from target instances. These experimental performances demonstrate that replacing traditional BN with CoN effectively scales feature representations and dramatically mitigates the influence of domain shift. Moreover, we also explore SVD tool to obtain singular values from the learned features on task $C \rightarrow P$. As shown in Figure 5 (b), CDAN+CoN has smaller difference between the largest and the smallest singular values compared with CDAN+BN. According to the theory of BSP (Chen et al. (2019), we achieve the conclusion that CoN can learn more discriminative features, which is consistent with our theoretical analysis in Sec. 3.3.

**Generalization Analysis:** Under the adversarial scheme, domain discrepancy is approximated by $\mathcal{A}$-divergence (Ben-David et al. (2010)) with the formulation as:

$$d_{\mathcal{H} \triangle \mathcal{H}}(S, T) = 2 \left(1 - \frac{1}{n_s + n_t} \left( \sum_{x : D(x) = 0} I(x \in D_s) + \sum_{x : D(x) = 1} I(x \in D_t) \right) \right),$$

(9)

where $D(\cdot)$ means the domain classifier distinguishing source domain from target domain ($D(x \in D_s) = 1$ and $D(x \in D_t) = 0$). This indicator reflects the alignment of distribution in the latent space. In addition, in terms of the learning bound theory in (Ben-David et al. (2010)) ($\epsilon_T(h) \leq$ $$-

Figure 5: Visualization of convergence speed, eigen-values, $\mathcal{A}$-distance and $\lambda$-value.
Under review as a conference paper at ICLR 2021

$\epsilon_S(h) + \frac{1}{2} d_{H\Delta H}(S, T) + \lambda$, the expected error of hypothesis space $h$ on target domain $\epsilon_T(h)$ is also determined by the error $\lambda$ of the ideal joint hypothesis $h^*$ on two domains. We apply domain classifier in CDAN with various normalization tools on task $C \rightarrow P$ to evaluate $\mathcal{A}$-divergence and $\lambda$ in Figure 5 (c) and (d), where CoN with CDAN obtains lower values than BN and TN. It indicates that scaling features with CoN easily learns domain-invariant features and achieve distribution alignment.

5 Conclusion

Unsupervised domain adaptation (UDA) aims to learn model with high generalization ability by achieving domain adaptation. In this paper, we rethink UDA from manifold distribution perspective and propose a novel collaborative normalization strategy suitable for the forward propagation of features to achieve domain alignment. Theoretical and experimental studies fully illustrate that the application of CoN in convolutional layers effectively improves classification performance and accelerates model training convergence on solving UDA issue.

References


