Reducing Divergence in Batch Normalization for Domain Adaptation

Ellen Yi-Ge[1], Mingjing Wu[2*], Zhenghan Chen[3*]
[1]Carnegie Mellon University
[2]Nanyang Technological University
[3]Microsoft (China) Co., Ltd
yige@andrew.cmu.edu, mwu014@e.ntu.edu.sg, pandaarych@gmail.com

Abstract

The widespread adoption of Batch Normalization (BN) in contemporary deep neural architectures has demonstrated significant efficacy, particularly in the domain of Unsupervised Domain Adaptation (UDA) for cross-domain applications. Notwithstanding its success, extant BN variants often conflate source and target domain information within identical channels, potentially compromising transferability due to inter-domain feature misalignment. To address this limitation, we introduce Refined Batch Normalization (RBN), a novel normalization paradigm that leverages estimated shift to quantify discrepancies between estimated population statistics and their expected values. Our pivotal observation reveals that estimated shift can accumulate through BN stacking within the network, potentially degrading target domain performance. We elucidate how RBN mitigates this accumulation, thereby enhancing overall system efficacy. The practical implementation of this technique is realized through the RBNBlock, which supplants conventional BN with RBN in the bottleneck architecture of residual networks. Extensive empirical evaluation across diverse cross-domain benchmarks corroborates the superiority of RBN in augmenting inter-domain transferability. This perspective transcends immediate performance metrics, offering a foundational lens through which subsequent research can more deeply understand and refine the interplay between normalization strategies and domain adaptation.

Code — https://github.com/EllenYiGe/RBN

Introduction

In the pursuit of enhanced feature transferability and domain-specific knowledge acquisition, researchers have expanded their focus beyond traditional feature alignment and pixel-level image translation techniques to investigate the optimization of feature normalization modules within deep neural networks (DNNs).

Batch Normalization (BN) (Ioffe and Szegedy 2015), while instrumental in mitigating internal covariate shift within DNNs, has been identified as potentially detrimental to domain-specific information preservation in UDA contexts. This limitation arises from the indiscriminate sharing of mean and variance statistics across domains, an approach that fails to account for domain-specific nuances. To address this shortcoming, several innovative methodologies have emerged, each aiming to retain crucial domain-specific knowledge.

AdaBN (Li et al. 2017a) pioneered the use of distinct domain statistics for source and target domains. However, its exclusive reliance on target statistics during inference phases risks the loss of valuable source domain information. AutoDIAL (Maria Carlucci et al. 2017) offers a more nuanced approach, employing a shared weight parameter to merge domain statistics on a channel-by-channel basis. In contrast, InterBN (Wang et al. 2021) leverages scaling factors derived from individual BN channels to orchestrate a self-adjusting mechanism for channel importance, thereby facilitating the preservation of domain-specific information.

These advancements underscore the critical role of feature normalization in UDA, highlighting the ongoing efforts to refine and optimize knowledge transfer across domains while maintaining the integrity of domain-specific characteristics.

Up until now, certain methods have made progress via unsupervised domain adaptation (UDA) approaches through the discovery and utilization of domain-specific knowledge contained in the BN channels, yet their generalizability is currently limited under complicated scenarios. An important drawback of BN is its dependency on the mini-batch size, which according to (Huang et al. 2022), the error rate of BN changes in inverse relation to batch size resulting in a higher error rate as the batch size is decreased. This is a new challenge, step we have no remedy in the domain adaptation literature.

Inspired by the batch-free normalization (BFN) paradigm (Huang et al. 2022), we investigate a different UDA BN configuration during best practice: When we do not have to use specific statistics of individual batches, we suggest to replace them with expected statistics over a population through required 3D histograms, computed on the source data split. This approach normalizes these factors by taking into consideration the inherent dynamic changes of the mean activation distribution in the training stage which causes unreliable population statistics. We argue that the difference of the estimated population statistics of BN and their "expected" values becomes the "estimation shift" of BN – a notion we introduce to clarify this issue.

Our research reveals a pivotal phenomenon: the cumulative effect of batch normalization (BN) estimation shift within

*Equal contribution as corresponding authors.

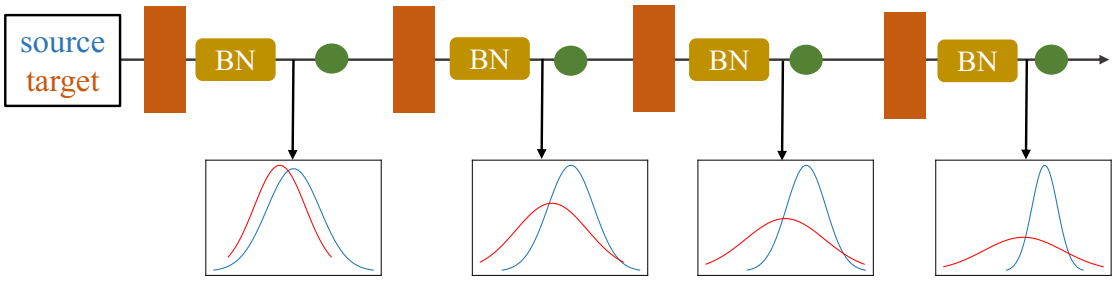

Figure 1: Key findings visualization: orange rectangles denote linear transformations; green circles represent non-linear operations.

neural networks during domain adaptation (as illustrated in Figure 1). This discovery elucidates the substantial performance deterioration observed in BN-equipped networks under conventional normalization techniques, particularly when confronted with distribution shifts in test data. Furthermore, it underscores the necessity of adapting BN population statistics in such scenarios.

Notably, our investigations demonstrate that the implementation of batch-free normalization (BFN) (Huang et al. 2022)—a technique that normalizes samples independently, disregarding batch dimensionality—effectively mitigates the accumulation of estimation shift in unsupervised domain adaptation (UDA). This approach significantly ameliorates network performance degradation in the presence of distribution shifts.

These insights culminated in the development of the RBN-Block, a novel architectural element that strategically replaces a single BN layer with Refined Batch Normalization (RBN) at the network's bottleneck. Our research contributions can be delineated as follows:

- We propose a novel approach to UDA based on Refined Batch Normalization (RBN), exploiting the capability of batch normalization layers to adapt with general architectural changes. The main benefit of RBN is that it avoids adding extra modules, allowing it to be easily implemented and at lower computing cost since it is located on the backbone network.
- RBN is surprisingly flexible, easily applicable to various UDA approaches to enhance their performance holistically.
- RBN's potentiality to achieve consistent and significant performance improvement is conclusively validated through extensive empirical evaluation over multiple cross-domain benchmarks, including Office-31, ImageCLED-DA, Office-Home and VisDA-2017.

Related Work

Unsupervised Domain Adaptation (UDA)

For the unsupervised domain adaptation (UDA) (Fang et al. 2024), the loss functions are generally formulated under two-tab paradigm. The first type of paradigm attempts to minimize distributional differences across domains by matching statistical properties. Typical implementations of this strategy are the Deep Domain Confusion (DDC) (Tzeng et al. 2014) and Deep Adaptation Networks (DAN) (Long et al. 2015a), both of which leverage Maximum Mean Discrepancy (MMD) (Gretton et al. 2007) for measuring and then minimizing the discrepancy across domains. JAN: Joint Adversarial Adaptation and Alignment (Long et al. 2017a) further improves the adversarial distillation model by concatenating adversarial learning with Maximum Mean Discrepancy (MMD) via the Joint Maximum Mean Discrepancy. More recently, newly developed domain discrepancy metrics have led to further refinements in discrepancy measurement methodology, including Sliced Wasserstein Distance (SWD) (Lee et al. 2019) and Contrastive Adaptation Network (CAN) (Kang et al. 2019).

Normalization Techniques

In adaptation research, new normalization architectures (Li et al. 2024) have been introduced to cater to the challenges imposed by the domain. There have been some previous works that have called for different statistics for source and target domains like AdaBN (Li et al. 2017b) or an integration of statistics on a per-channel basis, AutoDIAL (Cariucci et al. 2017), or separating the normalization statistics of source and target such as Domain-Specific Batch Normalization (DSBN) (Chang et al. 2019), or Transferable Normalization (TN) (Wang et al. 2019a) to establish cross-domain statistical alignment using channel attention mechanisms, or ConvNorm (Li and Vasconcelos 2019) to implement applied domain adaptation in a separate adaptation layer, or Domain Whitening Transform (DWT) (Roy et al. 2019) to structure a domain-specific whitening of the feature maps using dual covariance matrices.

Methodology

Batch Normalization: Principles and Challenges

Let $\vec{x} \in \mathbb{R}^d$ denote a d-dimensional input to the MLP, i.e. for a particular layer. In the training phase, the batch normalization (Ioffe and Szegedy 2015) is used to normalize each

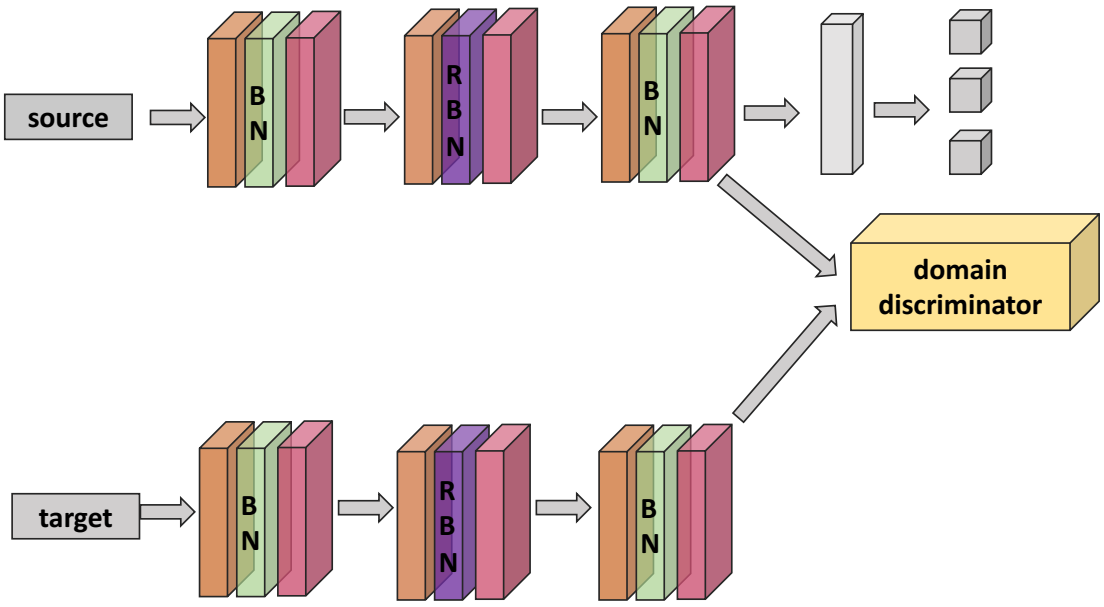

Figure 2: UDA method framework with source and target domain inputs.

neuron or channel based on m mini-batch data, formulated as:

$$\hat{x}_j = BN\left(x_j\right) = \frac{x_j - \mu_j}{\sqrt{\sigma_j^2 + \epsilon}}, j = 1, 2, \ldots, d \tag{1}$$

Here, μ_j denotes the mini-batch mean, calculated as the average of m samples of x_j, while σ_j^2 represents the variance of m samples of $(x_j - \mu_j)^2$. A small constant ϵ is introduced to ensure numerical stability.

During the inference or testing phase, the population mean $\tilde{\mu}$ and variance $\tilde{\sigma}^2$ of the layer input are requisite for deterministic predictions:

$$\hat{x}_j = BN_{inf}\left(x_j\right) = \frac{x_j - \bar{\mu}_j}{\sqrt{\sigma_j^2}}, j = 1, 2, \ldots, d \tag{2}$$

Given that direct computation of population statistics $\{\tilde{\mu}, \tilde{\sigma}^2\}$ is infeasible, we approximate them using $\{\hat{\mu}, \hat{\sigma}^2\}$, which are derived from running averages of mini-batch statistics across training iterations t. These estimates are updated using a factor β:

$$\begin{cases} \hat{\mu}^t = (1-\beta)\hat{\mu}^{t-1} + \beta\mu^{t-1} \\ \left(\hat{\sigma}^t\right)^2 = (1-\beta)\left(\hat{\sigma}^{t-1}\right)^2 + \beta\left(\sigma^{t-1}\right)^2. \end{cases} \tag{3}$$

The discrepancy between BN's behavior during training and inference poses challenges, particularly in recurrent neural networks and scenarios involving small batch sizes, where population statistics estimation may be inaccurate (Huang et al. 2022).

To circumvent the necessity of population statistics estimation, Batch-Free Normalization (BFN) (Huang et al. 2022) has been proposed. This approach eschews normalization along the batch dimension, ensuring consistency between training and inference operations. Layer Normalization (LN) (Ba, Kiros, and Hinton 2016) exemplifies this methodology, standardizing the layer input within neurons for each training sample:

$$\hat{x}_j = LN\left(x_j\right) = \frac{x_j - \mu}{\sqrt{\sigma^2 + \epsilon}}, \quad j = 1, 2, \ldots, d \tag{4}$$

This method could help mitigate some of the issues faced with normalizing over batches in specific neural networks and training situations.

In the case of Layer Norm (LN) (Ba, Kiros, and Hinton 2016), $\mu = \frac{1}{d}\sum_{i=1}^{d} xj$, $\sigma^2 = \frac{1}{d}\sum i = 1^d (x_j - \mu)^2$ refer to the sample-specific mean and sample-specific variance, respectively. LN generalizes this idea by normalizing the layer input over neurons and normalizing neurons independently within some defined groups. This method leads to more flexibility than traditionally normalization approaches such that they achieve better performance on visual tasks, especially with small batch sizes during training.

The field was further advanced by Batch-Free Normalization (BFN) (Huang et al. 2022), which solves the accumulation of estimation drift in BFN. What sets the BFN apart is its sample-independent normalization technique based on batch-independent processing. The proposed technique is efficient

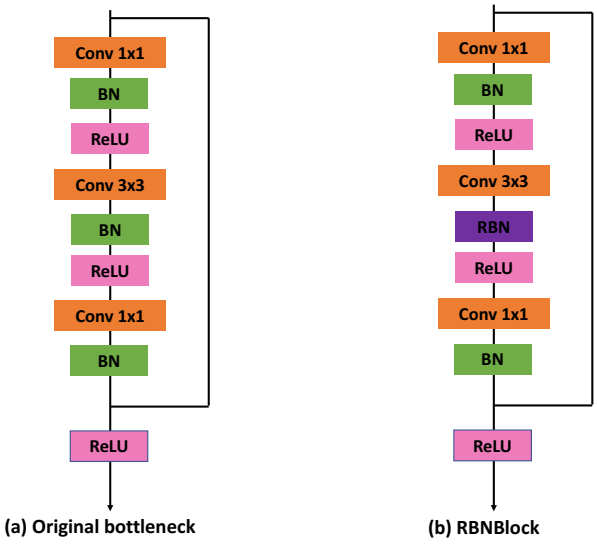

Figure 3: Key findings visualized.

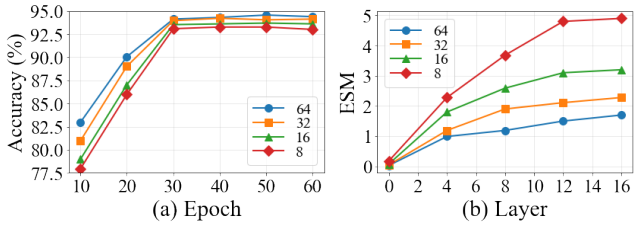

Figure 4: (a) Larger batch size improves network performance. (b) ESM increases with more layers.

in combating the growing estimation drift by Batch Normalization, which reduces performance loss of the network in the presence of distributional variations.

Inspired by the BFN paradigm, we abord the multisource Unsupervised Domain Adaptation (UDA) task from a perspective of arguing refined batch normalization. In this post, we will present a transformative perspective that opens new doors to tackle some of the most persistent problems facing domain adaptation, particularly in cases where traditional normalization approaches have not yielded satisfactory results.

By leveraging the strengths of BFN and applying them to the UDA context, we posit that refined batch normalization can provide a more robust framework for handling the inherent distributional discrepancies between source and target domains. This not only overcomes the shortcoming of traditional batch normalization under UDA scenarios but also provides a new avenue for better adapting and generalizing neural networks to heterogeneous domains.

Refine Batch Normalization (RBN)

At the same time, recent literature in UDA has proposed multiple approaches to overcome Batch Normalization (BN) limitations. The key methods include AdaBN (Li et al. 2017b), AutoDIAL (Cariucci et al. 2017), Domain-specific Whitening Transform (DWT) (Roy et al. 2019), Domain-Specific Batch Normalization (DSBN) (Chang et al. 2019) and Transferable Normalization (TN) (Wang et al. 2019a). The above three techniques of UDA normalization are depicted in comparison with our proposed Robust Batch Normalization (RBN) in Figure 3.

A common thread among these methodologies is the implementation of separate normalization strategies to circumvent the sharing of identical mean and variance parameters. However, this approach is not without its drawbacks, particularly in terms of estimation bias. This phenomenon occurs when the estimated population statistics derived from BN fail to accurately reflect the expected statistics, potentially compromising the efficacy of the normalization process.

Given the critical role of batch normalization in deep learning architectures, it is imperative to conduct a thorough investigation into the ramifications of estimation bias on network performance within the context of UDA. To this end, our research endeavors to quantitatively assess the discrepancy between estimated and expected population statistics in UDA scenarios.

We introduce $\tilde{\mu}$ (expected pop mean of BN) and $\tilde{\sigma}^2$ (expected pop variance of BN), and we use $\hat{\mu}$ and $\hat{\sigma}^2$ to denote their estimated quantities.

In addition, we perform an experimental analysis in this work to shed light on the effect of batch normalization estimation bias on UDA network performance. In addition, through this empirical analysis, we seek to quantify the effects of estimation bias, as well as offer possible solutions to mitigate the bias and decrease its negative effect on UDA tasks.

Evaluation

In this section we conduct a thorough empirical study of the effect of BN estimation shift on the performance of batch normalized neural networks, and present potential remedies.

We demonstrate, based on our theoretical analysis over our Batch Fission Normalization (BFN) framework that the statistical pass of BN is not solely responsible for training testing error gap, but specifically; inaccurate estimate of BN statistics is the root cause. Novelties behind this estimation uncertainty are cumulative, as networks deepen with BN layers added.

In addition, the authors of BFN explain that the variations in input distribution that may exist between training and test datasets, can lead to even more estimation bias in estimation (bias), which will further limit the generalization of the network as well as the performance during testing. An interesting observation in our experiments is that the standard deviation of Estimation Shift Magnitude (ESM_σ) for deep BN layers may increase towards end of training.

In essence, our experimental results suggest that the estimation shift in BN can potentially accumulate in networks with stacked BN layers, likely resulting in detrimental effects on the network's test performance, particularly when distribution shifts occur between domains.

Methods	A→W	D→W	W→D	A→D	D→A	W→A	Avg
Source Only (He et al. 2016)	68.4	96.7	99.3	68.9	62.5	60.7	76.1
DDC (Tzeng et al. 2014)	75.6	96.0	98.2	76.5	62.2	61.5	78.3
DAN (Long et al. 2015b)	80.5	97.1	99.6	78.6	63.6	82.8	80.4
RTN (Long et al. 2016)	84.5	96.8	99.4	77.5	66.2	64.8	81.6
DANN (Ganin et al. 2016)	82.0	96.9	99.1	79.7	68.2	67.4	82.2
ADDA (Tzeng et al. 2017)	86.2	96.2	98.4	77.8	69.5	68.9	82.9
JAN (Long et al. 2017b)	85.4	97.4	99.8	84.7	68.6	70.0	84.3
MADA (Pei et al. 2018)	90.0	97.4	99.6	87.8	70.3	66.4	85.2
MCD (Saito et al. 2018)	88.6	98.5	**100.0**	92.2	69.5	69.7	86.5
DWL (Xiao and Zhang 2021)	89.2	99.2	**100.0**	91.2	73.1	69.8	87.1
TADA (Wang et al. 2019b)	94.3	98.7	99.8	91.6	72.9	73.0	88.4
SHOT (Liang et al. 2022)	90.1	98.7	99.9	93.9	75.3	75.0	88.8
SymNet (Zhang et al. 2019)	95.2	98.8	**100.0**	93.9	74.6	72.5	88.4
SAR (Wang and Zhang 2020)	95.2	98.6	**100.0**	91.7	74.5	73.7	89.0
CDAN (Long et al. 2018)	94.1	98.6	**100.0**	92.9	71.0	69.3	87.7
CDAN+RBN	**95.9**	**99.1**	**100.0**	**95.7**	**76.1**	**74.5**	**90.2**

Table 1: Accuracy (%) on Office-31 with ResNet-50.

Methods	I→P	P→I	I→C	C→I	C→P	P→C	Avg
Source Only (He et al. 2016)	74.8	83.9	91.5	78.0	65.5	91.2	80.7
DAN (Long et al. 2015b)	74.5	82.2	92.8	86.3	69.2	89.8	82.5
RTN (Long et al. 2016)	75.6	86.8	95.3	86.9	72.7	92.2	84.9
DANN (Ganin et al. 2016)	75.0	86.0	96.2	87.0	74.3	91.5	85.0
JAN (Long et al. 2017b)	76.8	88.0	94.7	89.5	74.2	91.7	85.8
MADA (Pei et al. 2018)	75.0	87.9	96.0	88.8	75.2	92.2	85.8
SAFN (Xu et al. 2019)	78.0	91.7	96.2	91.1	77.0	94.7	88.1
SAR (Wang and Zhang 2020)	78.3	91.3	96.7	90.5	78.1	96.2	88.5
CDAN+RN (Huang et al. 2021)	78.6	92.7	97.2	92.8	79.1	94.8	89.2
CDAN (Long et al. 2018)	77.7	90.7	97.7	91.3	74.2	94.3	87.7
CDAN+RBN	**81.5**	**93.6**	**98.2**	**94.5**	**81.3**	**96.4**	**90.9**

Table 2: Accuracy (%) on ImageCLEF-DA with ResNet-50.

Methods	A→C	A→P	A→R	C→A	C→P	C→R	P→A	P→C	P→R	R→A	R→C	R→P	Avg
Source Only (He et al. 2016)	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
SymNet (Zhang et al. 2019)	47.7	72.9	78.5	64.2	71.3	74.2	64.2	48.8	79.5	74.5	52.6	82.7	67.6
ATM (Li et al. 2020)	52.4	72.6	78.0	61.1	<u>72.0</u>	72.6	59.5	52.0	79.1	73.3	58.9	83.4	67.9
DAN (Long et al. 2015b)	43.6	57.0	67.9	45.8	56.5	60.4	44.0	43.6	57.7	63.1	51.5	74.3	56.3
TADA (Wang et al. 2019b)	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
DANN (Ganin et al. 2016)	45.6	59.3	70.1	47.0	58.5	60.9	46.1	43.7	68.5	63.2	51.8	76.8	57.6
JAN (Long et al. 2017b)	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
CDAN (Long et al. 2018)	50.7	70.6	76.0	57.6	70.0	70.0	57.4	50.9	77.3	70.9	56.7	81.6	65.8
CDAN+RBN	**53.8**	**73.5**	**79.4**	**63.2**	**72.9**	**75.7**	**66.3**	**54.2**	**81.3**	**74.5**	**62.9**	**84.8**	**70.2**

Table 3: Accuracy (%) on Office-Home with ResNet-50.

To further validate these observations, we conducted additional experiments using Conditional Domain Adver-

Methods	aero	truck	train	skate	person	plant	motor	knife	horse	car	bus	bicycle	Avg
Source Only (Long et al. 2018)	55.1	8.5	73.5	26.5	31.2	81.0	79.7	17.9	80.6	59.1	61.9	53.3	52.4
DAN (Long et al. 2015b)	87.1	20.7	85.8	36.3	53.1	49.7	63.0	42.9	90.3	42.0	76.5	63.0	59.2
DANN (Ganin et al. 2016)	81.9	7.8	82.8	54.6	65.1	51.9	65.1	29.5	81.2	44.3	82.8	77.7	60.4
MCD (Saito et al. 2018)	87.0	25.8	83.0	40.3	76.9	88.6	84.7	79.6	88.9	64.0	83.7	60.9	72.0
BSP+DANN (Chen et al. 2019)	92.2	37.1	84.5	66.9	72.4	80.6	86.8	54.0	87.0	47.5	83.8	72.5	72.1
BSP+CDAN (Chen et al. 2019)	92.4	38.4	82.1	77.9	77.0	84.2	90.1	80.6	89.0	57.5	81.0	61.0	75.9
DSAN (Zhu et al. 2020)	90.9	39.4	89.1	67.6	75.1	92.8	93.7	77.0	88.9	62.4	75.7	66.9	75.1
DWL (Xiao and Zhang 2021)	90.1	28.7	85.6	57.1	78.0	90.6	86.8	81.5	92.4	67.6	86.1	80.2	77.1
CDAN (Long et al. 2018)	85.2	38.0	81.9	76.0	74.5	83.4	88.1	74.9	84.2	50.8	83.0	66.9	74.0
CDAN+RBN	**95.9**	**46.7**	**81.3**	**79.8**	**80.1**	**93.7**	**94.8**	**84.2**	**96.7**	**73.8**	**87.7**	**76.3**	**82.6**

Table 4: Accuracy (%) on VisDA-2017 with ResNet-101.

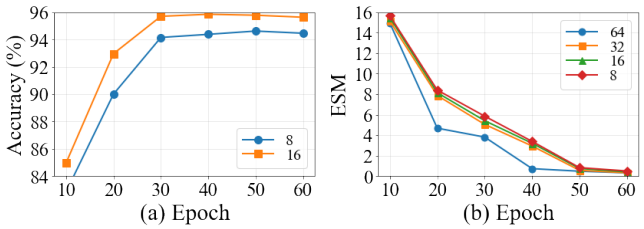

Figure 5: (a) Larger batch size improves performance. (b) Replacing BN with GN reduces ESM.

sarial Networks (CDAN) as our baseline model (illustrated in Figure 2). We systematically varied the batch size ($\{8, 16, 32, 64\}$) and the number of training epochs ($\{10, 20, 30, 40, 50, 60\}$).

As depicted in Figure 4 (a), our results demonstrate that batch size exerts a notable influence on model performance. Concurrently, Figure 4 (b) reveals that as the number of layers increases, the model exhibits stable performance characteristics.

These empirical observations further corroborate our hypothesis that the distribution shift between source and target domains in adaptation scenarios can induce estimation bias in BN, negatively impacting domain adaptation performance. Moreover, our findings suggest that in deeper BN architectures, Estimation Shift Magnitudes (ESMs) have the potential to attain higher values towards the conclusion of the training process.

These insights underscore the critical need for robust normalization techniques that can effectively mitigate estimation bias, particularly in the context of deep neural networks and domain adaptation scenarios. Our subsequent analyses will focus on developing novel approaches to enhance the reliability and effectiveness of batch normalization in diverse visual recognition tasks across domains.

In our novel approach, we introduce a hybrid architecture termed RBN, wherein we substitute the deeper Batch Normalization (BN) layers with Group Normalization (GN) layers. As illustrated in Figure 5, this strategic modification effectively mitigates the cumulative estimation errors inherent in BN when confronted with distribution shifts, thereby enhancing the overall robustness and performance of the network. The granular details pertaining to the implementation of RBN within the context of Conditional Domain Adversarial Networks (CDAN) are elucidated in the Supplementary Materials, providing a comprehensive overview of our methodological framework.

Experiments

Datasets

Office-31 (Saenko et al. 2010) is a seminal benchmark in the domain adaptation field.
ImageCLEF-DA [1] dataset, derived from the ImageCLEF 2014 challenge, serves as another crucial benchmark for assessing domain adaptation methodologies.
Office-Home (Li et al. 2019) represents a more expansive benchmark, comprising four diverse domains: Art (**A**), Clipart (**C**), Product (**P**), and Real World (**R**).
VisDA-2017 (Peng et al. 2017) presents a particularly challenging simulation-to-real scenario. It features two highly disparate domains: synthetic renderings of 3D models captured under various angles and lighting conditions, and real-world natural images. The dataset is structured around 12 classes, distributed across training, validation, and test domains, offering a rigorous evaluation of domain adaptation algorithms in bridging the gap between simulated and real-world data.

Implementation Details

To evaluate the efficacy of our proposed Refined Batch Normalization (RBN), we selected Conditional Domain Adversarial Networks (CDAN) (Long et al. 2018) as our baseline, designating our enhanced model as CDAN+RBN. Our implementation leverages the Pytorch framework, employing mini-batch stochastic gradient descent (SGD) for optimization. We set the weight decay to 5×10^{-4}, momentum to 0.9, and learning rate to 10^{-3}. For feature extraction, we

[1] http://imageclef.org/2014/adaptation

Methods	A→W	D→W	W→D	A→D	D→A	W→A	Avg
BN (He et al. 2016)	82.0	96.9	99.1	79.7	68.2	67.4	82.2
AdaBN (Li et al. 2016)	82.4	97.7	99.8	81.0	67.2	68.2	82.7
AutoDIAL (Maria Carlucci et al. 2017)	84.8	97.7	100.0	85.7	63.9	68.7	83.5
TransNorm (Wang et al. 2019b)	91.8	97.7	100.0	88.0	68.2	70.4	86.0
CDAN+RBN	**95.9**	**99.1**	**100.0**	**95.7**	**76.1**	**74.5**	**90.2**

Table 5: Accuracy on Office-31: BN, AdaBN, AutoDIAL, CDAN+RBN.

Methods	A→W	D→W	W→D	A→D	D→A	W→A	Avg
DANN (Ganin et al. 2016)	82.0	96.9	99.1	79.7	68.2	67.4	82.2
DANN (Ganin et al. 2016)+RBN	**83.7**	**97.5**	**99.1**	**81.2**	**69.5**	**68.9**	**83.3**
Source Only (He et al. 2016)	68.4	96.7	99.3	68.9	62.5	60.7	76.1
Source Only (He et al. 2016)	**69.7**	**98.1**	**99.1**	**69.5**	**63.7**	**61.8**	**77.0**
BSP (Ganin et al. 2016)	93.3	98.2	100.0	93.0	73.6	72.6	88.5
BSP (Ganin et al. 2016)+RBN	**95.4**	**99.2**	**100.0**	**94.8**	**75.1**	**73.2**	**89.6**

Table 6: Results with ResNet-50 and baselines.

utilize ResNet-50 as the backbone architecture for Office-31, ImageClEF-DA, and Office-Home datasets, while opting for a pre-trained ResNet-101 for the VisDA-2017 dataset. Our RBN implementation involves substituting BN layers with Group Normalization (GN), incorporating RBNBlocks throughout the network architecture. Specifically, we replace a predetermined number of deeper layers. Our experimental protocol involves utilizing all labeled source data and all unlabeled target data, reporting the mean classification accuracy across five randomized experiments for each transfer task. All other training parameters remain consistent with the baseline setup.

Results

Analysis Tables 1 through 4 detail the performance of our CDAN+RBN approach. Our method consistently outperforms the CDAN baseline, with average accuracy improvements of 2.5%, 3.2%, and 4.4% on the Office-31, ImageCLEF-DA, and Office-Home datasets, respectively. Particularly, our approach excels in challenging tasks within the Office-31 dataset, boosting accuracy from 71.0% to 76.1% for the **D**→ **A** task, and from 69.3% to 74.5% for the **W**→ **A** task. On the VisDA-2017 dataset, our method delivers an 8.6% improvement over the CDAN baseline. Overall, our approach achieves superior average classification performance compared to baseline methods across all four datasets. Notably, since our method and CDAN differ only by the replacement of BN with RBN, the observed performance gains can be directly attributed to RBN.

Comparative Analysis of Normalization Modules. RBN, designed as an end-to-end trainable layer, enhances generalization capabilities. To isolate RBN's impact, we conducted a comparative analysis against other normalization methods, including vanilla BN, AdaBN, AutoDIAL, and TransNorm, by substituting RBN with these alternatives while keeping other network components constant. As shown in Table 5, CDAN + RBN consistently outperforms the comparative methods on the Office-31 dataset.

Efficacy of Our Proposed Method. To further validate our method, we applied it to three additional baselines: DANN (Ganin et al. 2016), Source Only (He et al. 2016), and BSP (Ganin et al. 2016). Table 6 shows that DANN+RBN, Source Only + RBN, and BSP + RBN yield improvements of 1.1%, 0.9%, and 1.1%, respectively, over their baselines. These results underscore the effectiveness and versatility of our method as a plug-and-play solution.

Conclusion

Our findings indicate that BN's estimation shifts can accumulate within a network, leading to potential performance drops during inference under distribution shifts. We propose that refining batch normalization can alleviate this issue, minimizing performance degradation. Our approach seamlessly integrates into various network architectures by replacing the BN layer with the RBN module during training. By shifting focus from raw accuracy improvements to the underlying principles of normalization and domain alignment, the insights gained from RBN inspire a richer theoretical comprehension of cross-domain challenges. This conceptual groundwork equips future investigators with a robust platform for innovating and advancing domain adaptation methodologies, paving the way for more nuanced and principled exploration. Looking ahead, we expect our method to have a meaningful impact on real-world applications and large-scale model tasks.

References

Ba, L. J.; Kiros, J. R.; and Hinton, G. E. 2016. Layer Normalization. *CoRR*, abs/1607.06450.

Cariucci, F. M.; Porzi, L.; Caputo, B.; Ricci, E.; and Bulò, S. R. 2017. AutoDIAL: Automatic Domain Alignment Layers. In *Proceedings of the IEEE International Conference on Computer Vision*.

Chang, W.-G.; You, T.; Seo, S.; Kwak, S.; and Han, B. 2019. Domain-specific batch normalization for unsupervised domain adaptation. In *CVPR*, 7354–7362.

Chen, X.; Wang, S.; Long, M.; and Wang, J. 2019. Transferability vs. discriminability: Batch spectral penalization for adversarial domain adaptation. In *ICML*, 1081–1090.

Fang, Y.; Yap, P.-T.; Lin, W.; Zhu, H.; and Liu, M. 2024. Source-free unsupervised domain adaptation: A survey. *Neural Networks*, 106230.

Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; and Lempitsky, V. 2016. Domain-adversarial training of neural networks. *JMLR*, 2096–2030.

Gretton, A.; Borgwardt, K.; Rasch, M.; Schölkopf, B.; and Smola, A. J. 2007. A Kernel Method for the Two-Sample-Problem. In *Advances in Neural Information Processing Systems*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*, 770–778.

Huang, L.; Zhou, Y.; Wang, T.; Luo, J.; and Liu, X. 2022. Delving into the estimation shift of batch normalization in a network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 763–772.

Huang, Z.; Sheng, K.; Li, K.; Liang, J.; Yao, T.; Dong, W.; Zhou, D.; and Sun, X. 2021. Reciprocal Normalization for Domain Adaptation. *CoRR*.

Ioffe, S.; and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, 448–456. pmlr.

Kang, G.; Jiang, L.; Yang, Y.; and Hauptmann, A. G. 2019. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Lee, C.-Y.; Batra, T.; Baig, M. H.; and Ulbricht, D. 2019. Sliced wasserstein discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Li, A.; Qiu, C.; Kloft, M.; Smyth, P.; Rudolph, M.; and Mandt, S. 2024. Zero-shot anomaly detection via batch normalization. *Advances in Neural Information Processing Systems*, 36.

Li, J.; Chen, E.; Ding, Z.; Zhu, L.; Lu, K.; and Huang, Z. 2019. Cycle-consistent Conditional Adversarial Transfer Networks. In *Proceedings of the 27th ACM International Conference on Multimedia, MM 2019, Nice, France, October 21-25, 2019*, 747–755. ACM.

Li, J.; Chen, E.; Ding, Z.; Zhu, L.; Lu, K.; and Shen, H. T. 2020. Maximum Density Divergence for Domain Adaptation. *TPAMI*.

Li, Y.; and Vasconcelos, N. 2019. Efficient multi-domain learning by covariance normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5424–5433.

Li, Y.; Wang, N.; Shi, J.; Liu, J.; and Hou, X. 2016. Revisiting batch normalization for practical domain adaptation. *ICLR*.

Li, Y.; Wang, N.; Shi, J.; Liu, J.; and Hou, X. 2017a. Revisiting Batch Normalization For Practical Domain Adaptation. In *5th International Conference on Learning Representations*. OpenReview.net.

Li, Y.; Wang, N.; Shi, J.; Liu, J.; and Hou, X. 2017b. Revisiting batch normalization for practical domain adaptation. In *International Conference on Learning Representations*.

Liang, J.; Hu, D.; Wang, Y.; He, R.; and Feng, J. 2022. Source Data-Absent Unsupervised Domain Adaptation Through Hypothesis Transfer and Labeling Transfer. *TPAMI*, 8602–8617.

Long, M.; Cao, Y.; Wang, J.; and Jordan, M. 2015a. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning*.

Long, M.; Cao, Y.; Wang, J.; and Jordan, M. 2015b. Learning transferable features with deep adaptation networks. In *ICML*, 97–105.

Long, M.; Cao, Z.; Wang, J.; and Jordan, M. I. 2018. Conditional adversarial domain adaptation. In *NeurIPS*, 1640–1650.

Long, M.; Zhu, H.; Wang, J.; and Jordan, M. I. 2016. Unsupervised domain adaptation with residual transfer networks. *NeurIPS*.

Long, M.; Zhu, H.; Wang, J.; and Jordan, M. I. 2017a. Deep transfer learning with joint adaptation networks. In *International Conference on Machine Learning*.

Long, M.; Zhu, H.; Wang, J.; and Jordan, M. I. 2017b. Deep transfer learning with joint adaptation networks. In *ICML*, 2208–2217.

Maria Carlucci, F.; Porzi, L.; Caputo, B.; Ricci, E.; and Rota Bulo, S. 2017. Autodial: Automatic domain alignment layers. In *Proceedings of the IEEE international conference on computer vision*, 5067–5075.

Pei, Z.; Cao, Z.; Long, M.; and Wang, J. 2018. Multi-adversarial domain adaptation. In *AAAI*.

Peng, X.; Usman, B.; Kaushik, N.; Hoffman, J.; Wang, D.; and Saenko, K. 2017. VisDA: The Visual Domain Adaptation Challenge. *CoRR*, abs/1710.06924.

Roy, S.; Siarohin, A.; Sangineto, E.; Bulo, S. R.; Sebe, N.; and Ricci, E. 2019. Unsupervised domain adaptation using feature-whitening and consensus loss. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Saenko, K.; Kulis, B.; Fritz, M.; and Darrell, T. 2010. Adapting visual category models to new domains. In *ECCV*, 213–226.

Saito, K.; Watanabe, K.; Ushiku, Y.; and Harada, T. 2018. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*, 3723–3732.

Tzeng, E.; Hoffman, J.; Saenko, K.; and Darrell, T. 2017. Adversarial discriminative domain adaptation. In *CVPR*, 7167–7176.

Tzeng, E.; Hoffman, J.; Zhang, N.; Saenko, K.; and Darrell, T. 2014. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*.

Wang, M.; Wang, W.; Li, B.; Zhang, X.; Lan, L.; Tan, H.; Liang, T.; Yu, W.; and Luo, Z. 2021. Interbn: Channel fusion for adversarial unsupervised domain adaptation. In *Proceedings of the 29th ACM international conference on multimedia*, 3691–3700.

Wang, S.; and Zhang, L. 2020. Self-adaptive Re-weighted Adversarial Domain Adaptation. *IJCAI*.

Wang, X.; Jin, Y.; Long, M.; Wang, J.; and Jordan, M. I. 2019a. Transferable Normalization: Towards Improving Transferability of Deep Neural Networks. In *Advances in Neural Information Processing Systems*.

Wang, X.; Jin, Y.; Long, M.; Wang, J.; and Jordan, M. I. 2019b. Transferable normalization: Towards improving transferability of deep neural networks. In *NeurIPS*, 1953–1963.

Xiao, N.; and Zhang, L. 2021. Dynamic Weighted Learning for Unsupervised Domain Adaptation. In *CVPR*, 15242–15251.

Xu, R.; Li, G.; Yang, J.; and Lin, L. 2019. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *ICCV*.

Zhang, Y.; Tang, H.; Jia, K.; and Tan, M. 2019. Domain-symmetric networks for adversarial domain adaptation. In *CVPR*.

Zhu, Y.; Zhuang, F.; Wang, J.; Ke, G.; Chen, J.; Bian, J.; Xiong, H.; and He, Q. 2020. Deep Subdomain Adaptation Network for Image Classification. *TNNLS*.