Grand Unified Domain Adaptation

Ziyun Cai*[1]
Ziyun.Cai@ieee.org
Tengfei Zhang[1]
tfzhang@126.com
Xiao-yuan Jing[2]
jingxy_2000@126.com
Ling Shao[3]
ling.shao@ieee.org

[1] College of Automation, Nanjing University of Posts and Telecommunications, Nanjing 210023, China
[2] School of Computer, Wuhan University, Wuhan 430072, China
[3] Inception Institute of Artificial Intelligence, Abu Dhabi, UAE

Abstract

Existing domain adaptation (DA) methods try to handle various DA scenarios subject to imbalanced label sets or multiple source/target domains, *e.g.*, Closed set, Open set, Multi-Source, Partial and Multi-Target DA. Though Universal Domain Adaptation (UniDA) and Versatile Domain Adaptation (VDA) have been proposed to address these scenarios simultaneously, the related proposed methods still suffer from two issues: i) UniDA and VDA can hardly cover all of existing DA scenarios, *e.g.*, UniDA cannot handle Multi-Source and Multi-Target DA scenarios, and VDA does not include Open set DA. ii) The proposed UniDA and VDA methods mainly focus on the versatility, and they ignore the essential DA problem, *i.e.*, domain mismatch. This paper introduces Grand Unified Domain Adaptation (GUDA) scenario, which needs no prior knowledge about the number of source/target domains or the overlap. GUDA can cover more existing DA scenarios. Towards tackling GUDA, we formulate a grand unified adaptation network called **G**raph **C**ontrastive **A**daptation **N**etwork (GCAN), which can handle above mentioned DA scenarios and further reduces the domain mismatch without any modification. GCAN includes a graph contrastive adaptation objective at the node level, and a transferability rule to gain the common category identification loss. The results illustrate that GCAN works stably on different GUDA settings and shows comparable performance against recent DA methods on five benchmarks.

1 Introduction

Deep learning methods could improve the recognition performance when a large number of labelled data are used for training, but always drops significantly on another new domain, which can be called domain mismatch problem [3, 4, 31]. In practice, domain mismatch often exists in the real-world scenario, where the training and test data are typically acquired from different sensors or deployed from different environments. A short distance between the networkąŕs training target domains will lead to wrong predictions and affect the performance significantly. Moreover, abundant training data for a new target domain require large

© 2021. The copyright of this document resides with its authors.
It may be distributed unchanged freely in print or electronic forms.

labor work on gaining labeled data, which limits the applications of deep learning methods. Thus, domain adaptation (DA) task is proposed to circumvent this issue, which can transfer the knowledge trained from the source domain to the target domain, while removing the domain shift.

Most of the existing methods [2, 9, 27] assume that the label sets across source and target domains are identical, which is termed as Closed Set Domain Adaptation (CSDA). Though this impractical assumption helps gain many insights for DA methods, it is not true in the real-world scenario, where the category number in the target domain is often different from that in source domain. In addition, in many practical settings, source and target domains may contain many different domains, *e.g.*, in multi-camera applications. Aiming to handle these, some broad adaptation settings: Partial Domain Adaptation (PDA) [5, 6, 33], Open Set Domain Adaptation (OSDA) [14] [25] and Multi-Source/Target Domain Adaptation (MSDA/MTDA) [19, 21, 24, 30] are proposed. However, these settings must acquire the prior knowledge about label sets and domain configurations before training, which limits the applications for the practical scenarios. Therefore, researchers may struggle to choose a proper method when encountering a new unknown DA scenario.

For this purpose, two generalized DA scenarios Universal Domain Adaptation (UniDA) [8, 26, 32] and Versatile Domain Adaptation (VDA) [12] are proposed, which try to handle several different DA scenarios, simultaneously. However, these two scenarios still suffer from two issues: i) UniDA and VDA cannot cover all of existing DA scenarios. In UniDA scenario, it can only address CSDA, PDA and OSDA issues, where MSDA and MTDA scenarios cannot be handled. On the other side, VDA can hardly handle OSDA scenario. ii) Proposed UniDA and VDA methods mainly focus on the versatility, and they ignore the essential DA problem, *i.e.*, domain mismatch. From [12], we can observe that when encountering traditional Unsupervised Domain Adaptation (UDA) issue, where the source and target domains have identical label set, [12] performs worse than the state-of-the-art UDA approaches, by a large margin (5%). Therefore, the motivation of this work aims to propose a model, which works well on all existing DA scenarios including OSDA, MSDA and MTDA, simultaneously, while still removing domain mismatch effectively.

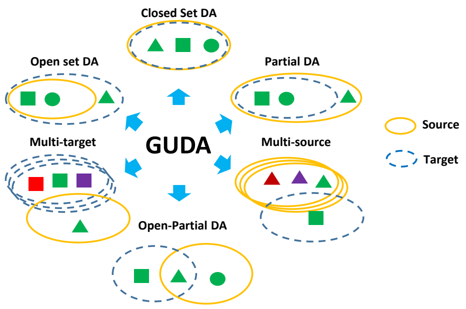


Figure 1: Grand Unified Domain Adaptation (GUDA) and existing domain adaptation settings with respect to label sets of source and target domains, and multi-target/source domains, where different shapes mean different classes, and same shape with different colors mean different domains.

In this paper, we propose a generalized scenario, termed as Grand Unified Domain Adaptation (GUDA) (Fig. 1). In GUDA, the label set constraints are removed. GUDA needs no prior knowledge about the number of the source/target domains and the label set. When a source domain is obtained, for a new target, we need to find the outlier categories and mark these outliers as "unknown". Then we try to classify the samples correctly. GUDA should tackle most of the existing scenarios without any modification.

To this end, we formulate a grand unified adaptation network called **G**raph **C**ontrastive **A**daptation **N**etwork (GCAN), which can be considered as a pilot tool when researchers encounter a new unknown DA task. GCAN includes a graph contrastive adaptation objective at the node level, and a transferability rule to calculate the common category identification loss. In addition, aiming to address Multi-Source/Target domain adaptation scenarios, Multi-Source/Target Alignment (MSTA) network is introduced to predict the weights for source and target samples at each iteration, which can align the target and source distributions. Through GCAN, the examples which are from the common label set or multi-domains are successfully recognized and matched, and the outlier target examples are marked as "unknown" class. To summarize, major contributions are as follows:

1. We propose a more practical setting, Grand Unified Domain Adaptation (GUDA), which needs no prior knowledge about the label set and the source/target domains number. Different from previous UniDA and VDA, GUDA can simultaneously tackle more existing DA scenarios including OSDA, MSDA and MTDA without modification.

2. A GUDA method is proposed, **G**raph **C**ontrastive **A**daptation **N**etwork (GCAN), which can tackle GUDA problem in an end-to-end fashion.

3. Aiming to handle the unsatisfactory performance of UniDA and VDA methods on UDA problem, we introduce a graph contrastive adaptation objective at the node level to further reduce the mismatch between source and target domains.

4. Experimental results on five benchmarks show that GCAN works stably across different GUDA settings and performs favorably against recent DA methods, including label set imbalanced tasks and multi-domain tasks. Deeper analyses illustrate that when the tasks are changed to special cases of GUDA, *i.e.*, OSDA, CSDA or PDA , the results of GCAN is still comparable.

2 Related Work

Partial Domain Adaptation. PDA aims to handle the domain mismatch issue when the source domain categories are larger than target domain categories. In PDA scenario, target samples may wrongly be recognized to the outlier source classes, resulting negative transfer issue [17]. To solve PDA issue, some approaches perform importance-weighting on source data to find the similar data in the target [5, 6, 33].

Open Set Domain Adaptation. OSDA assumes that there are some samples in the target, which do not belong to the source classes. The classes private to source and target domains are "unknown" categories [14] [25]. Existing OSDA methods assume that the target domain must contain unknown categories, which leads to the limitations of the applications for PDA and CSDA scenarios.

Multi-Source/Target Domain Adaptation. Compared to above single-domain DA, MSDA and MTDA assume realistic scenarios that source domain or target domain can be collected from multiple domains. Aiming to handle this, some representative deep models based domain adaptation methods are proposed [19, 21, 24, 30].

Universal/Versatile Domain Adaptation. Traditional domain adaptations are limited by the label set constraints. UniDA is a general setting of domain adaptation, which requires no prior knowledge of label set relationship, and includes some of the previous adaptation settings [8, 26, 32]. However, UniDA cannot handle MSDA and MTDA scenarios. VDA is first proposed in [12], which aims to handle several different DA scenarios by one method without any modification. Aiming to handle VDA scenario, [12] proposes a loss function called Minimum Class Confusion (MCC). Though extensive results show that MCC can outperform state of the art scenario-specific DA methods, it can hardly handle OSDA scenario.

3 Proposed Method

Problem Setup. Our task is Grand Unified Domain Adaptation (GUDA), which can tackle almost all domain adaptation scenarios without any modification. Given a labeled source domain $\mathcal{D}_s$ with the corresponding label set $\mathbf{Y}_s = [\mathbf{y}_{s_1}, \cdots, \mathbf{y}_{s_{N_s}}]$, and an unlabeled target domain $\mathcal{D}_t$ with the corresponding label set $\mathbf{Y}_t = [\mathbf{y}_{t_1}, \cdots, \mathbf{y}_{t_{N_t}}]$. N_t and N_s are the number of samples in the target and source domains, respectively. Note that $\mathbf{Y}_t$ cannot be accessed when training, and it is only applied to define the domain adaptation task. The marginal and conditional distributions between domains are considered as $P_{\mathcal{D}_s} \neq P_{\mathcal{D}_t}$ and $P_{\mathbf{Y}_s|\mathcal{D}_s} \neq P_{\mathbf{Y}_t|\mathcal{D}_t}$, respectively. GUDA aims to predict $\mathbf{Y}_t$, while eliminating the domain mismatch problem between $\mathcal{D}_s$ and $\mathcal{D}_t$. $\mathcal{C}_s$ is the number of source classes, and $\mathcal{C}_t$ is the number of target classes.

In GUDA scenario, the target domain may contain part of the source classes and some unknown classes. In addition, no prior knowledge about the overlap or the number of the source or target domains can be acquired. GUDA removes all restrictions and handles all the mentioned adaptation scenarios. The shared classes are denoted as $\mathcal{C} = \mathcal{C}_s \cap \mathcal{C}_t$. The merged source and target domains are also denoted as $\mathcal{D}_s$ and $\mathcal{D}_t$. The category sets private to the source and target domains are denoted as $\overline{\mathcal{C}}_s = \mathcal{C}_s \backslash \mathcal{C}$ and $\overline{\mathcal{C}}_t = \mathcal{C}_t \backslash \mathcal{C}$, respectively.

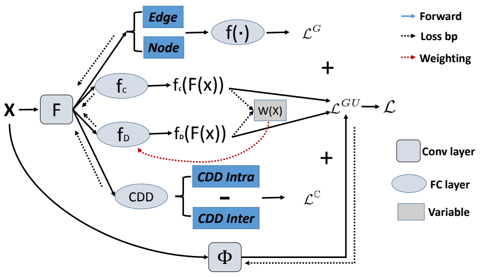


Figure 2: Architecture of Graph Contrastive Adaptation Network (GCAN). $\mathcal{L}$ is the the final objective. See text for detail.

3.1 The model

As shown in Fig. 2, our model includes feature extractor F, label classifier f_C, domain classifier f_D, Graph Neural Networks (GNN) encoder $f(\cdot)$, a Contrastive Domain Discrepancy (CDD) component and an alignment network $\Phi(\cdot)$. Data point x from $\mathcal{D}_s$ or $\mathcal{D}_t$ is put into the feature extractor F, denoting as $F(x)$. f_C outputs the label prediction of categories from source domain $f_C(F(x))$. f_D outputs the probability of the sample being from source domain $f_D(F(x))$. $f_D(F(x))$ can be considered as a domain factor, which can estimate the probability of samples being in the source domain, $\in [0,1]$.

Graph Contrastive Domain Mismatch. We minimize the domain mismatch of the last fully connected (FC) layers and fine-tune the layers by back-propagation. The proposed graph contrastive domain mismatch can be incorporated into the objective as an adaptation module over the activations of FC layers. Motivated by recent contrastive learning models, we can maximize the agreement between two modalities of the same graph via a contrastive loss to perform the pre-training. Graph representation learning through GNN has become a useful tool to explore graph structured data. Some methods have combined the contrastive strategy [34], which can learn discriminative representations through contrasting positive and negative examples. Following the trend, we introduce graph contrastive concept into domain adaptation to eliminate the mismatch between the source and target domains for a better performance. Let $\mathcal{G} = (\mathcal{V}; \mathcal{E})$ denotes a undirected graph, where $\mathcal{V}$ represents the node set and $\mathcal{E}$ is the edge set. Let $\mathbf{X}$ denotes the set of all possible data points. The feature matrix is denoted as $\mathbf{X} \in \mathbb{R}^{\mathcal{V} \times N}$, where $x_i \in \mathbb{R}^N$ is the N-dimensional attribute vector of the node $\mathbf{v}_i \in \mathcal{V}$. The aim is to obtain a M-layer GNN encoder $f(\cdot) \in \mathbb{R}^{\mathcal{V} \times N^*}$, which can receive graph features as input, and output node embeddings, *i.e.*, $N^* < N$. $f(\cdot)$ can transform nodes to low-dimensional embeddings for preservation of the graph structural features. In GCAN, we develop contrastive learning for GNN pre-training. Multi-layer perceptron (MLP) is used for extracting graph-level representation vectors:

$$\text{MLP}(f(\mathcal{G})) = \text{MLP}(\mathcal{R} : \{\mathbf{h}_i^m : \mathbf{v}_i \in \mathcal{V}\}), \tag{1}$$

where m is m-th layer and $\mathbf{h}_i$ is the embedding of node $\mathbf{v}_i$. $\mathcal{R}$ is the Readout function, which can summarize the obtained patch representations into graph-level representations [34] [28].

During the pre-training, graphs are randomly processed and sampled by contrastive learning. At each iteration, two correlated graph views are generated as positive (similar) pair of examples, denoted as $f(\mathcal{G}_i)$ and $f(\mathcal{G}_j)$. We optimize the corresponding contrastive loss:

$$\mathcal{L}^G = \frac{1}{N}\Big(\sum_{n=1}^{N}\big(\log(\sum_{n^*=1}^{N}\exp(\frac{\mathcal{S}(f(\mathcal{G}_{n^*,i}), f(\mathcal{G}_{n^*,j}))}{\nu})) - \frac{\mathcal{S}(f(\mathcal{G}_{n,i}), f(\mathcal{G}_{n,j}))}{\nu}\big)\Big), \tag{2}$$

where $n^* \neq n$, ν is the temperature parameter and $\mathcal{S}(\cdot)$ denotes the cosine similarity function. The contrastive loss is the mutual information maximization between two correlated graph views. Thus Eq. (2) is rewritten as the expectation form, while removing the subscript n/n^*:

$$\mathcal{L}^G = \mathbb{E}_{P_{\mathcal{G}_i}}\big(\log(\mathbb{E}_{P_{\mathcal{G}_j}})e^{\mathcal{P}(f(\mathcal{G}_i), f(\mathcal{G}_j))} - \mathbb{E}_{P_{\mathcal{G}_j|\mathcal{G}_i}}\mathcal{P}(f(\mathcal{G}_i), f(\mathcal{G}_j))\big) - \log N, \tag{3}$$

where $P_{\mathcal{G}_j|\mathcal{G}_i}$ and $P_{\mathcal{G}_i}$ are the conditional and marginal distribution of the graph views, respectively. $\mathcal{P}(\cdot)$ parametrizes the cosine similarity function $\mathcal{S}(\cdot)$ and ν. Through minimizing Eq. (3), the lower bound of mutual information between $f(\mathcal{G}_i)$ and $f(\mathcal{G}_j)$ is maximized.

We introduce Contrastive Domain Discrepancy (CDD) [13] to further eliminate the domain difference across the two domains. CDD maximizes the inter-class margin and minimizes the intra-class difference, simultaneously:

$$\mathcal{L}^{\mathbb{C}} = \sum_{l=1}^{n}\left(\frac{1}{\mathcal{C}}\sum_{c=1}^{\mathcal{C}}\mathcal{L}^{cc}(\mathbf{y}_{t_1:t_{N_t}}, \phi) - \frac{1}{\mathcal{C}(\mathcal{C}-1)}\sum_{c=1}^{\mathcal{C}}\sum_{c'=1}^{\mathcal{C}}\mathcal{L}^{cc'}(\mathbf{y}_{t_1:t_{N_t}}, \phi)\right), \tag{4}$$

where $\mathcal{L}^{cc}$ and $\mathcal{L}^{cc'}$ are the kernel mean embedding estimation for the two classes c and c'. ϕ is a mapping from the input to a specific layer, which can be defined through the deep learning model. l is the number of the layers. Eq. (4) optimizes a new metric which can model the inter-class and intra-class domain discrepancies jointly to improve the adaptation performance by end-to-end mini-batch training. The inter-class margin is maximized across domains to push the representations of each other further away. Therefore, the graph contrastive domain objective can be expressed as: $\mathcal{L}^{G\&\mathbb{C}} = \mathcal{L}^G + \mathcal{L}^{\mathbb{C}}$.

Grand Unified Approach to Domain Adaptation. The main motivation of this paper is to design a grand unified method for almost all existing domain adaptation scenarios. To estimate the confidence whether data point x is from $\mathcal{C}$, and select the samples to align the source and target feature distributions, we introduce $w(x)$. At the beginning, grand unified approach provides the high transferability samples to align target and source. When the alignment has been completed, approach will deal with the next set of samples with priority. When $w(x)$ is low, x is less likely from $\mathcal{C}$. $w(x)$ is considered as a threshold to decide whether we can label the test data as $\overline{\mathcal{C}}_t$. Normally, when the source data are similar with the target data, the samples have high probability to be in $\mathcal{C}$. Thus, the transferability rule is:

$$w(x) = f_D(F(x)) + \max f_C(F(x)) + \frac{f_C(F(x))}{\log|\mathcal{C}_s| + 1}, \tag{5}$$

where $f_D(F(x)) \in [0,1]$, which illustrates that higher values can be associated with the source samples. $\max f_C(F(x))$ and $\log|\mathcal{C}_s| \in [0,1]$, which obtains $w(x) \in [0,2.5]$. The transferability rule holds the assumption that if points in the source domain seem to be similar to the target samples, they have a higher possibility to in the shared categories. The maximum of $f_C(F(x))$ is a measure to classify target samples which can be recognized as the shared classes $\mathcal{C}$. Moreover, in GUDA setting, since recognize every sample in the target usually results in adverse results, only the data which are more likely to be from the shared categories are classified. In general, when source samples are more similar to target samples, they are more likely to be in $\mathcal{C}$. Therefore,

$$\mathop{\mathbb{E}}_{(x,\mathbf{Y})\in P_{\mathcal{D}_s}, P_{\mathbf{Y}_s}} \max f_C(F(x)) > \begin{cases} \mathop{\mathbb{E}}_{(x,\mathbf{Y})\in P_{\mathcal{D}_t}, P_{\mathbf{Y}_t} | \mathbf{Y}_t \in \mathcal{C}} \max f_C(F(x)) \\ \mathop{\mathbb{E}}_{(x,\mathbf{Y})\in P_{\mathcal{D}_t}, P_{\mathbf{Y}_t} | \mathbf{Y}_t \in \overline{\mathcal{C}}_t} \max f_C(F(x)), \end{cases} \tag{6}$$

where $\mathbf{Y}$ denotes the corresponding label of x.

In addition, we introduce Multi-Source/Target Alignment (MSTA) network $\Phi(\cdot)$ with parameters ϕ. MSTA predicts the weights for source and target samples at each iteration, which aligns the target and source distributions. MSTA estimates the preference of source and target samples to keep up with the improving domain classifier. Common category identification part is obtained from the transferability rule. The source and target alignment part automatically learns which samples are best suited to align to the target. The intuition is that the network $\Phi(\cdot)$ constantly re-measures the transferability of latent domains over time to reduce the distance between source and target domains. At the beginning, the network prefers examples which have higher transferability to align with the target. When these examples are aligned, the network prioritizes the next set of source examples for alignment.

Given some samples, $[x_s^1, \cdots, x_s^{N_s}] \in \mathcal{D}_s$, we put the samples into $\Phi(\cdot)$ for the scores to obtain the weight vectors. With the source sample weights, the loss for the identification of the common categories across source and target domains, and align the source and target feature distributions can be written as:

$$\begin{aligned} \mathcal{L}^{GU} = & \overbrace{\mathop{\mathbb{E}}_{(x,\mathbf{Y})\in P_{\mathcal{D}_s}, P_{\mathbf{Y}_s}} [\mathcal{L}_{CE}(\mathbf{Y}, f_C(F(x)))] + \mathop{\mathbb{E}}_{(x,\mathbf{Y})\in P_{\mathcal{D}_s}, P_{\mathbf{Y}_s}} [\mathbf{1}_{w(x)>\varepsilon} \cdot \mathcal{L}_{CE}((\mathrm{argmax} f_C(F(x)), f_C(F(x)))]}^{\textbf{\textit{Common Category Identification}}} \\ & \overbrace{+ \frac{1}{N_s}\sum_{i=1}^{N_s}\Big(\Phi(x_s^i)\log(f_D(F(x_s^i))) + \log(1 - f_D(F(x_s^i)))\Big) + \frac{1}{N_t}\sum_{i=1}^{N_t}\Big(\Phi(x_t^i)\log(f_D(F(x_t^i))) + \log(1 - f_D(F(x_t^i)))\Big)}^{\textbf{\textit{Source and Target Alignment}}}, \end{aligned} \tag{7}$$

where $[x_t^1, \cdots, x_t^{N_t}] \in \mathcal{D}_t$. $\mathcal{L}_{CE}$ is the standard cross-entropy loss. ε is the validated threshold, which is used to avoid negative transfer. Initially, the threshold is high. When the network is trained till it can better classify samples, the threshold is lower further.

Thus, the final objective can be expressed as: $\mathcal{L} = \mathcal{L}^{G\&\mathbb{C}} + \lambda \mathcal{L}^{GU}$, and λ is trade-off hyper-parameter.

4 Experiments

The experiments are run on a 20-GPU cluster and implemented in PyTorch platform.

Table 1: Accuracies (%) based on ResNet-50 with distance: β= 0.32, 0.15, 0.50, 0.07 and 0.43 (best performance is highlighted in bold, and second performance is underlined).

	Model	Office-31				Office-Home							VisDA	Im-Ca	DomainNet			
		A→W	D→W	A→D	Avg	A→C	A→P	A→R	C→A	C→P	C→R	Avg	Acc	I→C	P→R	P→S	R→S	Avg
	ResNet [11]	75.94	89.60	80.45	82.00	59.37	76.58	87.48	69.86	71.11	81.66	74.34	52.80	70.28	58.23	49.65	52.68	53.52
i)	DAN [15]	76.28	83.14	82.35	80.59	53.27	77.35	74.28	65.43	70.94	78.26	69.92	50.21	70.21	54.45	47.81	53.16	51.81
CSDA	DANN [9]	80.65	80.94	88.07	83.22	56.17	81.72	86.87	68.67	73.38	83.76	75.10	52.94	71.37	55.31	49.19	54.38	52.96
	ADDA [27]	75.24	73.81	78.58	75.88	52.63	78.32	77.87	70.23	72.45	84.21	72.62	53.28	72.81	53.21	42.27	49.67	48.38
ii)	STA [14]	77.13	80.76	75.54	77.81	67.24	63.62	77.86	66.80	63.26	75.83	69.10	37.69	65.82	54.12	47.63	50.79	50.85
OSDA	OSBP [25]	66.13	73.57	72.92	70.87	47.75	60.90	76.78	59.23	61.58	74.33	63.43	30.26	62.08	53.15	45.82	47.64	48.87
iii)	SAN [5]	82.64	84.82	81.58	83.01	52.23	75.44	77.73	66.21	67.83	82.63	70.35	56.24	70.04	59.37	51.85	52.64	54.62
PDA	IWAN [33]	85.25	90.09	84.27	86.54	52.55	81.40	86.51	70.58	70.99	85.29	74.55	58.72	72.19	58.26	52.97	56.47	55.90
	ETN [6]	87.94	97.62	88.47	91.34	54.29	78.82	79.93	73.07	72.54	85.45	74.02	59.08	73.24	59.97	53.64	57.28	56.96
iv)	DANCE [26]	88.65	97.53	89.46	91.88	66.75	84.16	87.20	75.26	74.94	87.79	79.35	62.31	76.92	62.56	57.79	59.86	60.07
UniDA/	UAN [32]	85.62	94.77	86.50	88.96	63.00	82.83	87.85	76.88	78.70	85.36	79.10	60.83	75.28	60.28	56.14	58.64	58.35
VDA	CMU [8]	86.86	95.72	89.11	90.56	63.52	83.81	**88.94**	77.72	79.37	86.85	80.04	61.42	76.45	65.76	60.35	62.45	62.85
	MCC [12]	81.37	96.29	87.82	88.49	61.74	81.97	84.82	74.84	77.29	85.68	77.72	**65.76**	**79.21**	60.21	54.48	59.30	58.00
Ours	**GCAN**	**91.24**	**97.73**	**93.16**	**94.04**	**69.98**	**85.27**	88.69	**83.24**	**81.59**	**89.73**	**83.08**	64.97	79.13	**66.84**	**61.64**	**64.88**	**64.45**

4.1 Experimental Setup

Datasets. We use five datasets, which are widely benchmarked in recent DA models: i) Office-31 [23], which has 3 domains, *e.g.*, Amazon (**A**), DSLR (**D**), Webcam (**W**)) including 31 classes. ii) Office-Home [29], which has 4 domains, *e.g.*, Art (**A**), Clip Art (**C**), Product (**P**) and Real World (**R**) containing 65 classes. iii) VisDA [18], contains 2 domains with 12 classes, focusing on a special DA setting (synthetic to real). iv) ImageNet-Caltech [22] [10], which includes a large number of classes with 1000 and 256, respectively. v) DomainNet dataset [20] is the largest DA dataset up to now, which has 6 domains, *e.g.*, Quickdraw (**Q**), Real (**R**), Painting (**P**), Infograph (**I**), Sketch (**S**) and Clipart (**C**) with 345 classes.

For the scenarios on CSDA, OSDA and PDA, we follow the same experimental settings as [32] [8]. In order to show the distance between the label sets, we introduce commonness value between source and target domains: $\beta = |\mathcal{C}_s \cap \mathcal{C}_t| / |\mathcal{C}_s \cup \mathcal{C}_t|$. CSDA is the special case when $\beta = 1$. For the scenarios on other DA scenarios, we follow the same setup as [12].

Compared Methods. GCAN is compared with seven kinds of DA methods: i) CSDA: Deep Adaptation Network (DAN) [15], Domain-Adversarial Neural Networks (DANN) [9], Adversarial Discriminative Domain Adaptation (ADDA) [27], ii) OSDA: Open Set Back-Propagation (OSBP) [25], Separate to Adapt (STA) [14], iii) PDA: Selective Adversarial Network (SAN) [5], Importance Weighted Adversarial Network (IWAN) [33], Example Transfer Network (ETN) [6]. iv) UniDA/VDA: Domain Adaptative Neighborhood Clustering via Entropy optimization (DANCE) [26], Universal Adaptation Network (UAN) [32], Calibrated Multiple Uncertainties (CMU) [8] and Minimum Class Confusion (MCC) [12]. v) MTDA: Maximum Classifier Discrepancy (MCD) [24] and Deep Adversarial Disentangled Autoencoder (DADA) [21]. vi) MSDA: Deep Cocktail Network (DCTN) [30] and Moment Matching for Multi-Source Domain Adaptation (M^3SDA) [19]. vii) MSPDA/MTPDA: Partial Adversarial Domain Adaptation (PADA) [5] and Adaptive Feature Norm (AFN) [31].

Moreover, ResNet [11] results are provided, which shows the lower bound without label domain adaptation and disalignment. ResNet is used as the network backbone.

Implementation Details. We employ ResNet pre-trained on ImageNet as the feature extractor. We utilize Adam optimizer with a fixed learning rate of 10^{-6} for the pre-trained and 10^{-5} for the final two randomly initialized layers. We set λ as 10^{-1} for our method, and provide parameter sensitivity analysis. The networks are trained for 20000 iterations. The training uses mini-batches composed of 256 samples. To enable more diverse classifiers in deep ensemble, we use affine scheme for data augmentation. The validated threshold ε has the following form: $\varepsilon = 1 - p/(2P)$, where p is the current iteration and P is the number of total iterations. We run the model for 10 runs and report the averaged performances.

Table 2: Accuracy (%) on DomainNet for MTDA and MSDA, and accuracy (%) on Office-Home for MSPDA and MTPDA. "m:" means that m domain is considered as the source domain, and other domains are merged into the target domain and vice versa.

Model	MTDA						MSDA						MSPDA				MTPDA			
	c:	i:	p:	q:	r:	s:	:c	:i	:p	:q	:r	:s	:A	:C	:P	:R	A:	C:	P:	R:
ResNet [illegible]	25.6	16.8	25.8	9.2	20.6	22.3	47.6	13.0	38.1	13.3	51.9	33.7	-	-	-	-	-	-	-	-
MCD [illegible]	25.1	19.1	27.0	10.4	20.2	22.5	54.3	22.1	45.7	7.6	58.4	43.5	-	-	-	-	-	-	-	-
DADA [illegible]	26.1	20.0	26.5	12.9	20.7	22.8	-	-	-	-	-	-	-	-	-	-	65.1	63.0	60.4	63.0
DCTN [illegible]	-	-	-	-	-	-	48.6	23.5	48.8	7.2	53.5	47.3	-	-	-	-	-	-	-	-
M^3SDA [illegible]	-	-	-	-	-	-	58.6	26.0	52.3	6.3	62.7	49.5	67.4	55.3	72.2	80.4	-	-	-	-
PADA [illegible]	-	-	-	-	-	-	-	-	-	-	-	-	62.8	51.8	71.7	79.2	59.9	53.7	51.1	61.4
AFN [illegible]	-	-	-	-	-	-	-	-	-	-	-	-	77.1	61.2	79.3	82.5	68.7	65.6	63.4	67.5
DANCE [illegible]	27.2	23.1	26.4	11.8	24.3	24.2	61.8	25.7	54.1	11.3	63.5	51.9	77.2	60.3	78.5	83.2	65.4	70.6	65.7	66.8
CMU [illegible]	26.4	22.3	27.7	12.4	21.9	27.5	62.3	24.7	52.4	13.5	61.2	48.7	76.4	62.7	77.9	82.8	66.6	71.3	64.1	65.9
MCC [illegible]	33.6	30.0	32.4	13.5	28.0	35.3	65.5	26.0	56.6	16.5	68.0	52.7	79.6	67.5	80.6	85.1	73.1	72.1	69.4	68.3
GCAN	**34.2**	**32.5**	**35.1**	**17.4**	**30.8**	**38.2**	**70.7**	**29.4**	**60.4**	**20.8**	**71.3**	**55.6**	**81.2**	**71.8**	**82.3**	**87.8**	**75.7**	**74.2**	**71.0**	**75.9**

4.2 Results & Analysis

Compared to CSDA, OSDA, PDA and UniDA/VDA. The classification results based on ResNet-50 are shown in Table 1. Some results of the baseline methods are from [8]. We can make following observations: 1) GCAN works stably on different GUDA settings and shows comparable performance against recent DA methods, which shows that GCAN can perform well on different kinds of datasets. 2) Since the violation of label set assumption, *i.e.*, OSDA methods are used to address PDA scenario, some DA methods perform even worse than ResNet. 3) MCC, which is proposed for VDA scenario, outperforms GCAN on VisDA and ImageNet-Caltech datasets. It may because that MCC is designed to tackle CSDA and PDA scenarios, and source domain contains more classes than target domain in the two datasets.

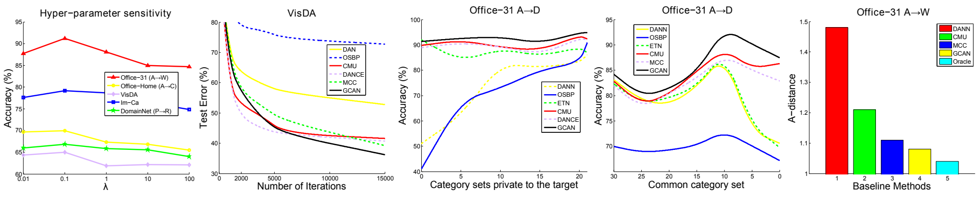
Compared to MTDA, MSDA and MSPDA/MTPDA. We evaluate the MTDA, MSDA, MSPDA and MTPDA tasks following the protocol of MCC [12]. Some results of the baseline methods are also from [12]. For MTDA and MSDA scenarios, we validate our method on DomainNet. For MSPDA and MTPDA scenarios, the performances are evaluated on Office-Home. The multiple source domains or target domains are merged into one source domain or one target domain, respectively. Table 2 shows classification accuracy based on ResNet-101 on these datasets. It illustrates that GCAN can achieve state-of-the-art results over other methods *w.r.t* performance accuracy generally, which illustrates that GCAN can address multi-domain issue effectively. The average classification accuracy on each dataset is improved with a big margin (4.0%).

Compared to UDA. One contribution of our work is that the proposed GCAN can better address domain mismatch problem than UniDA (DANCE and CMU) and VDA (MCC) methods. We evaluate GCAN for the most common UDA scenario on Office-31 dataset, where only the 10 common categories are used for the source and target domains. As reported in Table 3, GCAN outperforms other baseline methods.

Table 3: UDA scenario on Office-31 dataset, where only 10 common categories are used.

Model	MTDA						
	A→W	D→W	W→D	A→D	D→A	W→A	Avg
DAN [illegible]	80.5	97.1	99.6	78.6	63.6	62.8	80.4
DANN [illegible]	82.0	96.9	99.1	79.7	68.2	67.4	82.2
CDAN [illegible]	94.1	98.6	100.0	92.9	71.0	69.3	87.7
AFN [illegible]	88.8	98.4	99.8	87.7	69.8	69.7	85.7
DANCE [illegible]	93.7	98.1	98.4	94.6	72.3	73.2	88.4
CMU [illegible]	94.5	97.9	99.3	89.8	71.9	75.4	88.1
MCC [illegible]	95.5	98.6	100.0	94.4	72.9	74.9	89.4
GCAN	**97.6**	**99.3**	**100.0**	**96.2**	**78.3**	**80.2**	**91.9**

Hyper-parameter sensitivity. Though we have selected fixed hyper-parameters for learning rate and mini-batch size, another hyper-parameter λ still needs to be decided before training. We conduct experiments with values of λ from $\{10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}\}$ in Fig. 3 (a). It can be found that GCAN is less sensitive to the variety of λ. In addition, GCAN can achieve the best performances when $\lambda = 10^{1}$, generally. In the range of the selected λ, GCAN still



(a) Hyper-parameter sensitivity (b) Convergence Speed (c) Target Private Label Set (d) Common Label Set (e) A-distance

Figure 3: (a): Hyper-parameter sensitivity test. (b): Convergence Issue Study. (c): Accuracy *w.r.t* $\overline{\mathcal{C}}_t$. (d): Accuracy *w.r.t* $\mathcal{C}$. (e): A-Distance of the last fc-layer features.

shows comparable performance against other baseline methods.

Convergence Speed. Fig. 3 (b) shows the training curves of GCAN on VisDA dataset. It shows that the performance of GCAN increases stably with more iterations, and converges after 2000 iterations. The baselines (DAN, CDAN, CMU, DANCE and MCC) also converge fast but at this point they show unsatisfactory performance.

Varying $\overline{\mathcal{C}}_s$ and $\overline{\mathcal{C}}_t$. Following [32], we vary $\overline{\mathcal{C}}_s$ and $\overline{\mathcal{C}}_t$ on Office-31 dataset **A**→**D**, while fixing $\mathcal{C}_s \cap \mathcal{C}_t$ and β. Fig. 3 (c) shows that GCAN performs better than the selected methods (DANN, OSBP, ETN, CMU, DANCE and MCC), and is robust to the unknown classes in the source and target domains. In addtion, $\overline{\mathcal{C}}_t = 21$ is a special case of OSDA problem with $\mathcal{C}_s \in \mathcal{C}_t$. The results of GCAN is comparable to the results of OSBP. $\overline{\mathcal{C}}_t = 0$ is the PDA problem when $\mathcal{C}_t \in \mathcal{C}_s$. GCAN still performs comparable to ETN.

Varying Common Label $\mathcal{C}$. We analyze the behavior under the different number of common classes on Office-31 dataset **A**→**D**. Following [32], we fix $\overline{\mathcal{C}}_t = \overline{\mathcal{C}}_s + 1$ and vary $\mathcal{C}$ from 0 to 31. As showing in Fig. 3 (d), GCAN outperforms the selected methods on all of $\mathcal{C}$. Note that when $\mathcal{C} = 31$, it is a CSDA problem. GCAN is still comparable to the results of DANN.

Theoretical Insight. [1] derives the expected error $\mathcal{E}_{\mathcal{D}_t}(h)$ for a hypothesis h on the target domain $\mathcal{E}_{\mathcal{D}_t}(h) \leq \mathcal{E}_{\mathcal{D}_s}(h) + \frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_s, \mathcal{D}_t) + \varepsilon$ by the A-distance $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_s, \mathcal{D}_t)$, which is a measure of domain discrepancy. From Fig. 3 (e), we can observe that MCC has lower A-distance than other selected DA methods on Office-31 dataset **A**→**W**. The oracle one is the supervised learning on source and target domains.

Feature Visualization. We plot the last-layer learned target features for ResNet, DANN, MCC and GCAN on Office-31 dataset **A**→**W** task with t-Distributed Stochastic Neighbor Embedding (t-SNE) [7] in Fig. 4. We use ResNet-50 as the pretrained backbone. As we discussed in the method section, our method aims to minimize domain-divergence, and identifies the common classes. We can find that common target features (black plots) are well clustered by GCAN. Moreover, most of the other features (other colors) become far from common features, which illustrates that the features extracted from GCAN are domain-invariant. Similar samples about category sets private to the target $\overline{\mathcal{C}}_t$ are clustered together.

Ablation Study. Ablation study is shown through decomposing some variants of GCAN. Note that for the modified models, we directly use the optimal hyper-parameters, *i.e.*, $\lambda = 0.1$, the fixed learning rates 10^{-6} and 10^{-5}.

i) GCAN (DA alleviation), which excludes $f_D(F(x))$ in Eq. (5).

ii) GCAN (w/o $\max f_C(F(x))$), which excludes the classification component $\max f_C(F(x))$ in Eq. (5).

iii) GCAN (w/o Graph Contrastive Domain (GCD) mismatch), which excludes $\mathcal{L}^G$, yielding an incomplete loss function: $\mathcal{L} = \mathcal{L}^{\mathbb{C}} + \lambda \mathcal{L}^{GU}$.

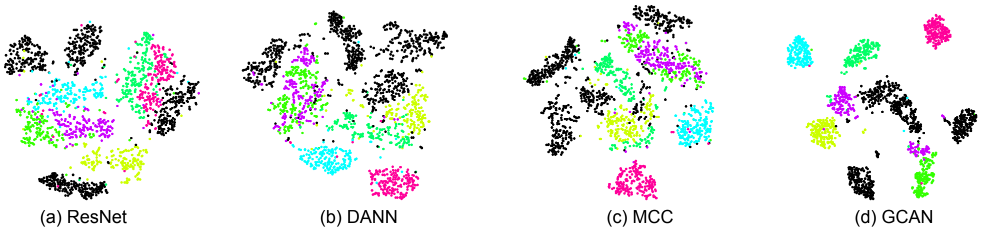


Figure 4: Feature visualization with t-SNE. Different colors are different classes.

iv) GCAN (w/o Multi-Source/Target Alignment (MSTA) network), where the loss $\mathcal{L}^{GU}$ in Eq. (7) ignores the source and target alignment part.

Table 4 illustrates the superiority of GCAN over the variants. When some parts in GCAN are modified or deleted, the performance decreases, sometimes with a large margin. Therefore, all components are necessary to achieve the best performance for GUDA scenario.

Table 4: Accuracies (%) by decomposing variants of GCAN.

	Office-31	**Office-Home**			**VisDA**
	A→W	**A→C**	**c:**	**:c**	Acc
GCAN (DA alleviation)	87.64	63.24	-	-	59.31
GCAN (w/o $\max f_C(F(x))$)	82.13	60.31	-	-	58.24
GCAN (w/o GCD)	84.24	61.53	-	-	57.69
GCAN (w/o MSTA)	88.71	65.82	28.4	58.3	62.86
GCAN	**91.24**	**69.98**	**34.2**	**70.7**	**64.97**

5 Conclusion

In this paper, we have introduced a novel Grand Unified Domain Adaptation (GUDA) setting, which needs no prior knowledge about the overlap or the number of source/target domains. GUDA covers most of existing DA scenarios. We propose Graph Contrastive Adaptation Network (GCAN), which can handle GUDA scenario and further reduces the domain mismatch without any modification. Extensive experiments on five datasets illustrate that GCAN shows comparable performance against recent DA methods. GCAN can be considered as a pilot tool when researchers encounter an unknown DA task.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 62006127, Grant 62073173, and Grant 61833011; in part by the National Science Foundation of Jiangsu Province, China, under Grant BK20191376 and Grant BK20190728; in part by the Jiangsu Shuang-Chuang Project under Grant CZ005SC19019; in part by Postdoctoral Science Foundation of Jiangsu Grant 2021K290B; and in part by the Leading Technology Foundation Research Project of Jiangsu Province under Grant BK20202011.

References

[1] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine Learning*, 79(1-2):151–175, 2010.

[2] Ziyun Cai, Yang Long, and Ling Shao. Adaptive rgb image recognition by visual-depth embedding. *IEEE Transactions on Image Processing*, 27(5):2471–2483, 2018.

[3] Ziyun Cai, Xiao-Yuan Jing, and Ling Shao. Visual-depth matching network: Deep rgb-d domain adaptation with unequal categories. *IEEE Transactions on Cybernetics*, 2020.

[4] Ziyun Cai, Jie Song, Tengfei Zhang, Xiao-Yuan Jing, and Ling Shao. Dual contrastive universal adaptation network. In *IEEE International Conference on Multimedia and Expo*, pages 1–6, 2021.

[5] Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Michael I Jordan. Partial transfer learning with selective adversarial networks. In *IEEE Conference on CVPR*, pages 2724–2732, 2018.

[6] Zhangjie Cao, Kaichao You, Mingsheng Long, Jianmin Wang, and Qiang Yang. Learning to transfer examples for partial domain adaptation. In *IEEE Conference on CVPR*, pages 2985–2994, 2019.

[7] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International Conference on Machine Learning*, pages 647–655, 2014.

[8] Bo Fu, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Learning to detect open classes for universal domain adaptation. In *European Conference on Computer Vision*, 2020.

[9] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 17(1):2096–2030, 2016.

[10] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on CVPR*, pages 770–778, 2016.

[12] Ying Jin, Ximei Wang, Mingsheng Long, and Jianmin Wang. Minimum class confusion for versatile domain adaptation. In *European Conference on Computer Vision*, 2020.

[13] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4893–4902, 2019.

[14] Hong Liu, Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Qiang Yang. Separate to adapt: Open set domain adaptation via progressive separation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2927–2936, 2019.

[15] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I Jordan. Learning transferable features with deep adaptation networks. *arXiv preprint arXiv:1502.02791*, 2015.

[16] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I. Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, pages 1647–1657, 2018.

[17] S. J. Pan and Y. Qiang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[18] Xingchao Peng, Ben Usman, Neela Kaushik, Dequan Wang, Judy Hoffman, and Kate Saenko. Visda: A synthetic-to-real benchmark for visual domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 2021–2026, 2018.

[19] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *IEEE International Conference on Computer Vision*, pages 1406–1415, 2019.

[20] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *IEEE International Conference on Computer Vision*, pages 1406–1415, 2019.

[21] Xingchao Peng, Zijun Huang, Ximeng Sun, and Kate Saenko. Domain agnostic learning with disentangled representations. In *International Conference on Machine Learning*, volume 97, pages 5102–5112, 2019.

[22] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115 (3):211–252, 2015.

[23] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, pages 213–226, 2010.

[24] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3723–3732, 2018.

[25] Kuniaki Saito, Shohei Yamamoto, Yoshitaka Ushiku, and Tatsuya Harada. Open set domain adaptation by backpropagation. In *ECCV*, pages 153–168, 2018.

[26] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, and Kate Saenko. Universal domain adaptation through self supervision. *arXiv preprint arXiv:2002.07953*, 2020.

[27] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *IEEE Conference on CVPR*, pages 7167–7176, 2017.

[28] Petar Velickovic, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R. Devon Hjelm. Deep graph infomax. In *International Conference on Learning Representations*, 2019.

[29] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *IEEE Conference on Conference on Computer Vision and Pattern Recognition*, pages 5018–5027, 2017.

[30] Ruijia Xu, Ziliang Chen, Wangmeng Zuo, Junjie Yan, and Liang Lin. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3964–3973, 2018.

[31] Ruijia Xu, Guanbin Li, Jihan Yang, and Liang Lin. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1426–1435, 2019.

[32] Kaichao You, Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Universal domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2720–2729, 2019.

[33] Jing Zhang, Zewei Ding, Wanqing Li, and Philip Ogunbona. Importance weighted adversarial nets for partial domain adaptation. In *IEEE Conference on CVPR*, pages 8156–8164, 2018.

[34] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep Graph Contrastive Representation Learning. In *International Conference on Machine Learning Workshop*, 2020.