Towards Deployable Multi-Domain Learning for Inductive-Transductive Transfer

Jogendra Nath Kundu, Abhinav Agarwalla, Suvaansh Bhambri, Akshay Kulkarni,
Varun Jampani, R. Venkatesh Babu, *Senior Member, IEEE*

Abstract—There is a strong motivation to develop versatile learning paradigm for improved deployability to domain and task shifts. To this end, we propose a novel deployable multi-domain paradigm where a vendor prepares a foresighted multi-domain network (MDN) by training on proprietary multi-domain datasets. In the absence of data-exchange, only the network is shipped to a client. The client intends to deploy the model for a variety of inductive-transductive tasks. Here, the generalizability-focused vendor develops an effective learning technique to enable maximal retention of the rich transductive-inductive knowledge. Whereas, the extensibility-focused client aims to leverage the full capability of MDN for a given task in hand. We formally define the paradigm with theoretical insights towards developing an efficient MDN. It turns out that an MDN that advocates to retain both domain-generic and domain-specific knowledge is a better alternative over a domain-invariant MDN. Next, we develop client-side learning strategies that are tailored to leverage the most out of the vendor-provided MDN. Experiments reveal our state-of-the-art performance on standard benchmarks for transfer learning, domain adaptation and domain generalization problems. Our baseline comparisons demonstrate that beyond simple addition of multi-domain data, an effective MDN framework plays significant role in improving the final deployment performance.

Index Terms—Multi-Domain Learning, Unsupervised Domain Adaptation, Transfer Learning

1 Introduction

There has been an increase in interest to develop general-purpose frameworks that can be easily extended to solve a variety of image understanding problems [1], [2], [3]. This is beyond the general trend to learn independent, non-extendable models which are meant to solve specific challenges encountered during model deployment. Benchmarking such frameworks involves analyzing the model's effectiveness along two prominent aspects, *i.e.*, *generalizability* and *extensibility* (see Fig. 1A). Here, better generalizability implies learning of better feature representations to support future extensions and extensibility refers to the effectiveness of the transfer learning algorithm in leveraging the full potential of the generic model for downstream tasks.

Broadly, *generalizability* is of two types, transductive-generalizability and inductive-generalizability. Transductive (or domain) generalizability refers to the extent of input distributional coverage within which the model's predictions are considered to be robust. Input samples beyond this coverage are termed as out-of-distribution data. Similarly, inductive (or task) generalizability refers to the extent of support exhibited by the learned inductive bias for diverse task-shift scenarios. Thus, generalizability can be improved by exposing the model to a wide range of tasks and data-domains, with an intent to cover the unknown deployment scenarios. On the other hand, *extensibility* refers to effectiveness of the learning algorithm to leverage the capability of the generalized model to work well on novel deployment scenarios (change in output task or input domain). Similar to generalizability, extensibility can be broadly discussed under, **a)** extensibility to task-shifts and **b)** extensibility to domain-shifts.

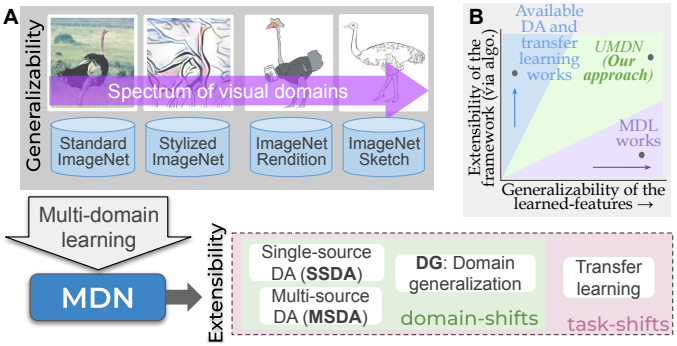


Fig. 1. **A.** In the proposed deployable paradigm, the vendor incorporates *generalizability* by utilizing multiple ImageNet variants. The client develops learning techniques with greater *extensibility* to deploy it for a variety of inductive-transductive tasks. **B.** In literature, multi-domain learning (MDL) works focus on generalizability while DA and transfer learning works focus on extensibility. Our unified framework supports both traits, yielding superior performance across task and domain shifts.

In literature, we observe two lines of research which separately cater to generalizability and extensibility. This is conceptually illustrated in Fig. 1B along with characteristic comparisons in Table 1. The works under the umbrella of Multi-Domain Learning (MDL) [4], [5], [6], [7] aim to improve the generalizability (Table 1D, X-axis of Fig. 1B) by realizing a model with minimal risk on a group of datasets (meta-training sets) drawn from distinct distributions. The MDL evaluation is done on a meta-test dataset which includes labeled support set and unlabeled query set. However, both support and query samples are from the same domain. On the other hand, an equivalent scenario in domain adaptation (DA) involves a labeled source (equivalent to the support set) and an unlabeled target (equivalent to the query set). Note that, support-query domain-shift is the key difference between DA and MDL evaluation (Table 1B). Since MDL works test on few-shot settings,

Varun Jampani is with Google Research, Cambridge, USA.
The rest of the authors are with the Video Analytics Lab, Dept. of Computational and Data Sciences, Indian Institute of Science, Bangalore, India.
E-mail: jogendrak@iisc.ac.in, agarwallaabhinav@gmail.com, suvaanshbhambri@gmail.com, akshayk.vnit@gmail.com, varunjampani@google.com and venky@iisc.ac.in

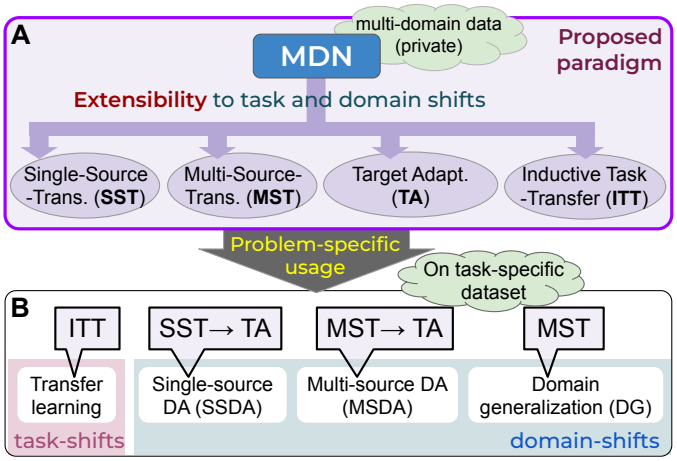


Fig. 2. MDN extends to various domain and task shift problems enabled via the proposed techniques; ITT, SST, MST and TA. Here, ITT extends MDN to solve TL [9]. SST followed by TA extends MDN to solve SSDA [10] problems. Similarly, MST followed by TA extends it to solve MSDA [11]. Just MST also solves a DG [12] problem.

there has been limited progress in developing effective extensible DA algorithms that build on top of the generalizable MDL setups (Table 1E).

On the other side, the works under the umbrella of Domain Adaptation (DA) and Transfer Learning (TL) exclusively focus on improving extensibility (Table 1E, Y-axis of Fig. 1B) via effective learning algorithms. However, they usually build on a vanilla ImageNet-pretrained base-model that exhibit low generalizability compared to an MDL base-model (Table 1D). We observe that a trivial extension of such extensible algorithms on top of MDL works is unable to leverage the full potential of the generalizability, leading to sub-optimal performance. To the best of our knowledge, no prior method collectively addresses both generalizability and extensibility. We believe a suitable marriage between these two broad directions can lead to a unified framework supporting both valuable traits.

Towards a unified framework, we ask ourselves, *what could be a practical setup for extensible MDL?* To this end, we propose a novel deployable multi-domain paradigm under a vendor-client setup [8]. Here, the vendor has access to large-scale proprietary data and thus can prepare a multi-domain network (MDN) by employing an effective MDL algorithm. In a privacy preserving paradigm, only the network is shipped to the client without the rich and memory-intensive multi-domain data. The client intends to deploy the network for a variety of inductive-transductive problems such as transfer learning, single-source domain adaptation, multi-source domain adaptation, *etc*. In a cooperative setup, both vendor and client are equally responsible to improve the network's performance on the final deployment-related task. Here, the vendor's objective would be to configure a setup that can improve the network's generalizability in the absence of any knowledge of deployment-side requirements. Similarly, the client's objective would be to develop learning techniques, with improved extensibility, that are well tailored to leverage the most out of the vendor-provided MDN.

Next, we intuitively discuss the learning strategies in the proposed deployable multi-domain paradigm.

a) Vendor-side generalizability. Here, the primary requirement is to get hold of a labeled, large-scale, multi-domain dataset. In the context of object recognition, the multi-domain data must cover a wide range of variations in visual stimulus that one may encounter in everyday life. To capture such domain diversity, we resort to large-scale ImageNet-variants (termed prototype-domains) as shown in Fig. 1. Next, we focus on developing an efficient multi-domain representation learning procedure while maintaining a suitable balance between domain-generic and domain-specific representations. We call the resulting model as Multi-Domain Network (MDN).

TABLE 1
Characteristic comparisons of prior multi-domain learning (MDL), domain adaptation (DA) and transfer learning (TL) works.

		MDL	DA	TL	Ours
Pre-training protocol	**A.** Task and domain shift b/w Prototype domain and Source domain	✓	✓	✓	✓
Evaluation protocol	**B.** Support-query (Source-target) domain-shift	✗	✓	✗	✓
	C. Support-query (Source-target) task-shift	✗	✗	✓	✓
D. Generalizability focused		✓	✗	✗	✓
E. Extensibility focused		✗	✓	✓	✓

b) Client-side extensibility. Here, the primary goal is to demonstrate wide applicability of MDN for deployment-related problems such as a) transfer learning (TL), b) single-source domain adaptation (SSDA), c) domain generalization (DG), and d) multi-source domain adaptation (MSDA). To this end, we propose novel learning techniques that can effectively utilize the multi-head nature of MDN. Broadly, we introduce the following learning strategies (see Fig. 2), a) inductive task transfer (ITT) b) single-source transfer (SST), c) multi-source transfer (MST), d) target adaptation (TA). We pay special attention to leverage the head-specificity of MDN thereby facilitating seamless extensibility, involving minimal training and parameter update.

Our prime contributions are as follows:

- To the best of our knowledge, we are the first to collectively address both generalizability and extensibility in a unified framework. To this end, we formally define a deployable multi-domain paradigm with theoretical insights towards developing an efficient multi-domain network, MDN.
- Alongside an effective vendor-side strategy, we develop client-side strategies that are well tailored to leverage the most out of the vendor-provided MDN. For extending to multi-source scenarios (*i.e*. MST) we formulate a novel instance-to-head affinity which respects the head-specificity of MDN while disregarding the domain-label of the task-specific multi-source dataset. We also utilize the head-specificity to guide the self-training for target adaptation (*i.e*. TA) through a novel multi-head-unanimity strategy.
- We thoroughly evaluate our approach against prior works and demonstrate that an effective MDN framework achieves significant improvements over competitive baselines even while utilizing the same multi-domain data. We outperform comparable prior arts on standard benchmarks under domain-shift, *i.e*. SSDA, MSDA, DG, and under task-shift, *i.e*. TL.

2 Related Work

We summarize and compare the characteristics of related works in multi-domain learning (MDL), domain adaptation (DA) and

transfer learning (TL) with our proposed MDN in Table 1.

Inductive transfer learning. ImageNet trained models have had remarkable success in transferring to downstream tasks [13], [14], [15], [16]. Multiple studies [17], [18], [19], [20] attempt to decode the underlying factors that lead to better extensibility. It is widely accepted that transfer learning performance decreases with the increase in domain [20] and task disparity [19]. Recent works show the effectiveness of removing the source-bias by learning adversarially-robust [9] models for transfer learning.

DG and MSDA. Recent DG methods [21], [22], [23] employ well explored distribution alignment techniques to learn useful domain-invariant features. Certain DG methods borrow techniques used in self-supervised approaches, such as data-augmentation [24], [25] and other training strategies [12]. Certain approaches focus on decomposing domain generalization into domain-specific and domain-invariant components [26], [27], [28]. Meta-learning techniques have also been applied as a solution for DG [26], [29], [30], [31]. Unlike in DG, MSDA considers access to target domain samples for a simultaneous training on both source and target. Several MSDA works [11], [32], [33] draw motivations from the alignment process employed in single-source DA and extend it for all possible source-target pairs.

Multi-domain learning. Multi-domain learning refers to learning universal representations [3] that can support consistent performance across a range of visual domains. [34] and [35] learn independent models for each domain and then learn to retrieve or mix relevant models for a new task in order to learn a universal representation. Partially-shared architectures have also shown to suit well for this task. [3], [36], [37] and [38] propose to train a single network to learn classification task in presence of multiple distinct domains. Along with a shared CNN backbone with universal parameters, these networks contain domain-specific modules to store domain-specific information using feature-wise linear units [37], light-weight residual adapters [36], [38] and normalization layers [3]. Alternatively, [39] uses conditional batch-normalization instead of learning domain-specific network parameters. Different from these, we encourage explicit specificity by incorporating *OOD* detection objective for each domain-specific head training.

3 Approach

Our prime objective is to realize a general purpose multi-domain feature learning framework followed by developing learning strategies to demonstrate its extensibility.

3.1 Background and motivation

3.1.1 Notations and problem formulation

We introduce notations to describe the proposed learning setup.

a) Vendor-side prototype domains. We assume access to a set of large-scale prototype domains denoted by $\{\mathcal{B}_m\}_{m=1}^{M}$. Each domain $\mathcal{B}_m$ is associated with probability distribution μ_m which is a measure over $\mathcal{X}\times\mathcal{Y}$. Here, $\mathcal{X}$ and $\mathcal{Y}$ denote the input and output space respectively. **b) Client-side data domains.** Let $\mathcal{S}$ and $\mathcal{T}$ be the source and target domains characterized by the distributions p and q respectively. Here, p and q are defined over $\mathcal{X}\times\mathcal{Y}$.

Assumptions. We assume that the marginals $\mu_m(y|x) \neq p(y|x)$. Implying, there exist an unknown output task-shift (non-matching labels) between the vendor and client side data. Further, $\mu_m(x) \neq p(x) \neq q(x)$, implying existence of domain-shift between the input domains $\mathcal{B}_m$, $\mathcal{S}$, and $\mathcal{T}$. However, we consider $p(y|x) = q(y|x)$ implying the client-side source and target follow the closed-set domain adaptation setting. In case of single-source adaptation, we consider M number of labeled prototype-domain datasets with a single labeled source and an unlabeled target.

3.1.2 Multi-domain learning paradigm

Here, the objective is to utilize the knowledge from all the domains (*i.e.*, via concurrent access to $\{\mathcal{B}_m\}_{m=1}^{M}$, $\mathcal{S}$, and $\mathcal{T}$) in order to realize a hypothesis h with a small target error, *i.e.*, $h^* = \arg\min_{h\in\mathcal{H}} \epsilon_q(h)$. Here,

$$\epsilon_q(h) = \mathop{\mathbb{E}}_{q(x,y)}[\ell(h(x), y)] \text{ where } h \in \mathcal{A}^{\alpha^*}(\mathcal{B}^{\alpha^*}, \mathcal{S}, \mathcal{T}) \subset \mathcal{H} \quad (1)$$

Here, ℓ is the loss and $\mathcal{H}$ denotes the hypothesis space. And, $\mathcal{B}^{\alpha^*}$ is defined by the distribution $\mu^{\alpha^*} = \sum_{m=1}^{M} \alpha^*_{[m]}\mu_m$ with $\alpha^* \in \Delta = \{(\alpha^*_{[m]})_{m=1}^{M} : \alpha^*_{[m]} \in [0,1] \text{ and } \sum_{m=1}^{M} \alpha^*_{[m]} = 1\}$. In other words, μ^{α^*} is a convex combination of the prototype domain distributions $\{\mu_m\}_{m=1}^{M}$. Here, $\mathcal{A}^{\alpha^*}(\mathcal{B}^{\alpha^*}, \mathcal{S}, \mathcal{T})$ (or in short, $\mathcal{A}^{\alpha^*}$) $\subset\mathcal{H}$ can be interpreted as a hypothesis subspace characterized by the best α^* with concurrent access to $\{\mathcal{B}_m\}_{m=1}^{M}$, $\mathcal{S}$ and $\mathcal{T}$, *i.e.*,

$$\alpha^* \in \Delta \text{ and } \alpha^* = \arg\min_{\alpha}(\arg\min_{h\in\mathcal{A}^{\alpha}} \epsilon_q(h)) \quad (2)$$

Objective. For the l^{th} source-target pair $(\mathcal{S}_l, \mathcal{T}_l)$ there exists a particular α_l^* such that $\epsilon_q(h \in \mathcal{A}^{\alpha_l^*}) \leq \epsilon_q(h \in \mathcal{A}^{\alpha}),\ \forall\ \alpha \in \Delta$. However, it is not practical to perform an optimization over the larger hypothesis space $\mathcal{A}^{\alpha}(\mathcal{B}^{\alpha}, \mathcal{S}_l, \mathcal{T}_l),\ \forall\alpha \in \Delta$ to find the optimal α_l^* for every encounter of a new source-target pair. Thus, we propose the deployable multi-domain learning paradigm.

3.2 Deployable multi-domain learning paradigm

The prime objective of this work is to realize a deployment friendly paradigm that is naturally viable for a privacy preserving setting *i.e.*, in the absence of data sharing between the vendor and client. To this end, we formalize the following paradigm.

Definition 1. ***(Deployable multi-domain paradigm)*** *Consider a vendor with access to labeled datasets from M prototype domains $\{\mathcal{B}_m\}_{m=1}^{M}$ and a client with an adaptation problem while having access to a dataset pair $(\mathcal{S}_l, \mathcal{T}_l)$ i.e., a labeled source $\mathcal{S}_l$ and an unlabeled target $\mathcal{T}_l$. In the deployable paradigm, the vendor prepares a foresighted model with a union of limited (K number of) hypothesis supports $\tilde{\mathcal{H}} = \cup_{k=1}^{K} \mathcal{A}^{\alpha_k}(\mathcal{B}^{\alpha_k}, ., .);\ \alpha_k \in \Delta$ without any knowledge about $(\mathcal{S}_l, \mathcal{T}_l)$. This model is later shipped to the client for adaptation in the absence of any data sharing.*

Note that, the target error in a deployable paradigm is always lower bounded by the non-deployable counterpart (obtaining $\mathcal{A}^{\alpha_l^*}$ with concurrent access to $\{\mathcal{B}_m\}_{m=1}^{M}$, $\mathcal{S}$ and $\mathcal{T}$), *i.e.*,

$$\epsilon_q(h \in \mathcal{A}^{\alpha_l^*}) \leq \epsilon_q(h \in \tilde{\mathcal{H}}) \quad (3)$$

The proposed paradigm has two important traits. Firstly, it restricts data-exchange between the two parties ensuring copyright and privacy compliance requirements. Secondly, the paradigm allows the vendor to prepare a single model to be shared across multiple clients to be deployed for various tasks (*i.e.*, SSDA, MSDA, DG, TL, *etc.*).

Definition 2. ***(Multi-domain extensibility criterion)*** *The foresighted vendor model with a limited hypothesis support, $\tilde{\mathcal{H}} = \cup_{k=1}^{K} \mathcal{A}^{\alpha_k};\ \alpha_k \in \Delta$, is termed extensible for a given client-side*

source-target pair, $(\mathcal{S}_l, \mathcal{T}_l)$ if with at least $(1-\delta)$ probability $\epsilon_q(h \in \tilde{\mathcal{H}})$ does not exceed $\epsilon_q(h \in \mathcal{A}^{\alpha_l^*})$ by more than ζ, i.e.,

$$\mathbb{P}[\epsilon_q(h \in \tilde{\mathcal{H}}) \leq \epsilon_q(h \in \mathcal{A}^{\alpha_l^*}) + \zeta] \geq 1 - \delta \tag{4}$$

Here, the probability $\mathbb{P}$ is computed over a given client-side source-target pair $(\mathcal{S}_l, \mathcal{T}_l)$. According to the above definition, the vendor's prime objective is to construct a model with the best possible $\tilde{\mathcal{H}}$ which can support a wide variety of unknown client-side scenarios within the generalizability bounds. With this intent, we introduce the following configurations that are later analyzed to build a reasonably good vendor-side model.

3.2.1 Empirical risk minimization (ERM) baseline

A straightforward approach would be to prepare a vendor-model with $\tilde{\mathcal{H}}_g = \mathcal{A}_g^{\alpha_g}$ where $\alpha_{g,[m]} = 1/M$ *i.e.*, with a single hypothesis support head $\mathcal{A}_g^{\alpha_g}$, implying $K = 1$. We denote this by ERM [40] as it aims to learn a domain-generic representation for all the prototype-domains. The inequality in Eq. 3 can be restated as,

$$\epsilon_q(h \in \mathcal{A}^{\alpha_l^*}) \leq \epsilon_q(h \in \tilde{\mathcal{H}}_g) \;\text{ where }\; \tilde{\mathcal{H}}_g = \mathcal{A}_g^{\alpha_g} \tag{5}$$

For a conceptual illustration, in Fig. 3, the ERM support is the most suitable for the client-side source-target pair of Scenario-1, as it is the closest among other supports.

3.2.2 Prototype-specific (PS) baseline

Consider a scenario where input distribution of $\mathcal{T}_l$ matches with one of the prototype domains $\mathcal{B}_{m'}$ instead of being entirely unique *i.e.*, $q(x) \approx \mu_{m'}(x)$. Here, a vendor model specific to the prototype m' would perform better than the ERM-baseline discussed above. However, we can not generalize it for any client-side $(\mathcal{S}_l, \mathcal{T}_l)$. Motivated by this, the hypothesis support for domain-specific baseline is formalized as $\tilde{\mathcal{H}}_{PS} = \cup_{k=1}^{K} \mathcal{A}_{PS}^{\alpha_k}$ where $\alpha_{k,[m]} = 1$ *if* $m = k$, 0 *otherwise*. Implying, $K = M$ number of support heads. The inequality in Eq. 3 can be expressed as,

$$\epsilon_q(h \in \mathcal{A}^{\alpha_l^*}) \leq \epsilon_q(h \in \tilde{\mathcal{H}}_{PS}) \;\text{ where }\; \tilde{\mathcal{H}}_{PS} = \cup_{k=1}^{K} \mathcal{A}_{PS}^{\alpha_k} \tag{6}$$

For a conceptual illustration, in Fig. 3, one of the *PS* supports $(\mathcal{A}_{PS}^{\alpha_{k_1}})$ is the most suitable for the client-side source-target pair of Scenario-2, as it is the closest among other supports.

3.2.3 Specific+generic (SG) MDL

According to the above discussion, the client-side task would perform well on one of the above two baselines. Thus, the best multi-domain network (MDN) should encapsulate both domain-generic and domain-specific prototype representations. Motivated by this, the hypothesis support for specific+generic MDN is formalized as $\tilde{\mathcal{H}}_{SG} = (\cup_{m=1}^{M} \mathcal{A}_{PS}^{\alpha_m}) \cup \mathcal{A}_g^{\alpha_g} = \tilde{\mathcal{H}}_{PS} \cup \tilde{\mathcal{H}}_g$ where $K = M + 1$ implies $M + 1$ number of support heads.

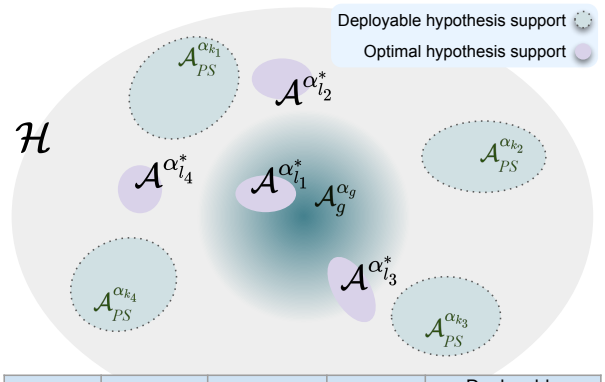
For a conceptual illustration, in Fig. 3, one of the *SG* supports $(\mathcal{A}_{PS}^{\alpha_{k_3}})$ is the most suitable for the client-side source-target pair of Scenario-3, as it is the closest among other supports.

We summarize the analysis in the following result.

Result 1. *For any client-side source-target pair $(\mathcal{S}_l, \mathcal{T}_l)$ and the above defined vendor-side hypothesis support spaces (*i.e.*, $\tilde{\mathcal{H}}_g$, $\tilde{\mathcal{H}}_{PS}$, and $\tilde{\mathcal{H}}_{SG}$) the target loss is upper bounded as follows,*

$$\epsilon_q(h \in \mathcal{A}^{\alpha_l^*}) \leq \epsilon_q(h \in \tilde{\mathcal{H}}_{SG}) = \min(\epsilon_q(h \in \tilde{\mathcal{H}}_{PS}),\ \epsilon_q(h \in \tilde{\mathcal{H}}_g)) \tag{7}$$

From the above analysis, we infer that *SG* is equipped to reasonably support a wide range of target scenarios. The architecture (Sec. 3.3.2) and learning strategy (Sec. 3.3.3) of the proposed multi-domain network follows the above discussed proposition.



Varied client-side scenarios	Vendor-side prototype domains	Client-side source-target domain pairs	Optimal hypothesis support	Deployable hypothesis support		
				ERM	PS	SG
Scenario-1	$\{\mathcal{B}_m\}_{m=1}^{M}$	$(\mathcal{S}_{l_1}, \mathcal{T}_{l_1})$	$\mathcal{A}^{\alpha_{l_1}^*}$	✓	✗	✓
Scenario-2		$(\mathcal{S}_{l_2}, \mathcal{T}_{l_2})$	$\mathcal{A}^{\alpha_{l_2}^*}$	✗	✓	✓
Scenario-3		$(\mathcal{S}_{l_3}, \mathcal{T}_{l_3})$	$\mathcal{A}^{\alpha_{l_3}^*}$	✗	✓	✓
Scenario-4		$(\mathcal{S}_{l_4}, \mathcal{T}_{l_4})$	$\mathcal{A}^{\alpha_{l_4}^*}$	✗	✓	✓

Fig. 3. An illustration of the hypothesis subspaces. The tick and cross marks for different scenarios denote suitability of the corresponding vendor-side configuration (*i.e.*, *ERM*, *PS*, and *SG*). Note that, *SG* is equipped to reasonably support for a wide range of deployable scenarios while strictly ensuring the copyright and privacy compliance.

3.3 Preparing Prototypical Multi-Domain Network

We introduce a Multi-Domain Network (MDN) to facilitate learning of general purpose representations which would be suitable for a wide range of deployment scenarios. To this end, we pay special attention to the following three aspects; a) multi-domain dataset b) multi-domain network architecture, and c) training strategy.

3.3.1 Prototypical Multi-domain dataset

While forming the multi-domain dataset, we look for the following two traits, a) input diversity coverage, and b) extent of categorization (large number of fine-grained classes). The selected domains must cover a wide range of variations in visual stimulus with an intention to subsume the unknown client-side domains. Further, generalizability to task-shifts can be stretched by training under large-scale categorization (*e.g.*, 1000-class ImageNet). We explore the following two ways to get hold of such prototypical dataset.

a) Publicly available ImageNet variants (`ImNet-P`). We look for publicly available large-scale datasets as used in domain-specific studies, such as ImageNet-Sketch [41] (*ImNet-Sk*) for sketch recognition. With original ImageNet [42] (`ImNet-O`) and *ImNet-Sk* at the poles (see Fig. 1A), we use two intermediate domains *i.e.* ImageNet-Rendition [43] (*ImNet-R*) and Stylized-ImageNet [44] (*ImNet-Sty*). *ImNet-R* includes naturally occurring changes in image style such as art, paintings, *etc.* And, *ImNet-Sty* contains stylized [45] versions of the original `ImNet-O` samples.

b) Domain-varying augmentations of ImageNet (`ImNet-A`). Certain prototype datasets, such as *ImNet-Sk*, *ImNet-R* selected under `ImNet-P`, require explicit data collection effort. Thus, a relaxed version of `ImNet-P` would be to prepare the prototype variants via diverse domain varying augmentations of the original ImageNet (`ImNet-O`) samples *i.e.*, without relying on any explicit data collection effort. To this end, we introduce `ImNet-A` which includes; **a)** original ImageNet (`ImNet-O`), **b)** ImageNet-Edge (*ImNet-E*), **c)** ImageNet-Cartoon (*ImNet-C*), and **d)** ImageNet-Stylized (*ImNet-Sty*). Here, *ImNet-E* (equivalent to

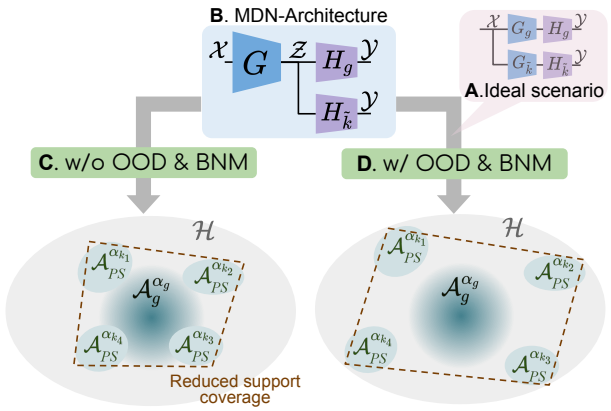


Fig. 4. **A.** In an ideal scenario, one should use fully unshared multi-domain network architecture to fully leverage the prototype-specific (PS) attributes. **B.** However, owing to computation complexity we resort to a partially-shared MDN-architecture. We observe that, **C.** without encouraging explicit specificity the *PS* supports tend to be closer to the *ERM* support. In order to recover from this detrimental effect, we instilling out-of-domain detection capability to individual heads via *OOD & BNM*.

ImNet-Sk) is obtained by passing ImNet-O samples via a contour detection model [46]. Similarly, *ImNet-C* (equivalent to *ImNet-R*) is obtain by applying cartoonize augmentation [47] of the original ImNet-O samples. Note that, *ImNet-Sty* is also a part of ImNet-P.

How to select the most influential domain prototypes?

It is crucial to formalize a principled prototype selection criteria to select a minimal set of prototype domains. Let $\mathbb{B}_c = \{\mathcal{B}_{m'}\}_{m'=1}^{M'}$ be the candidate set of prototype domains with M' number of candidate members. And, we aim to select a subset $\mathbb{B} = \{\mathcal{B}_m\}_{m=1}^{M}$ with $M < M'$. The number of domain-prototypes chosen, as well as the domain-prototypes themselves, are crucial to the effectiveness of vendor-side training. **1)** It is critical to select a diverse set of domain-prototypes to aid the proposed MDN in learning both domain-invariant and domain-specific properties. **2)** A higher number of domain-prototypes incurs increased computational costs in vendor-side training. As a result, determining a low enough M with a considerable performance gain is important.

We start with selecting the most preferred prototype domain, *i.e.*, the ImNet-O dataset as the only member of $\mathbb{B}^{(1)}$ at the first iteration. Following this, $\mathcal{B}_{m'} \in \mathbb{B}_c$ (associated with distribution $\mu_{m'}$) is chosen as the next member of $\mathbb{B}^{(i+1)}$ if it satisfies the following condition,

$$m' = \underset{\hat{m} \in \mathbb{B}_c \setminus \mathbb{B}^{(i)}}{\arg\max}\ \epsilon_{\mu_{\hat{m}}}(h); \quad h = \underset{\hat{h} \in \mathcal{A}_g^{\alpha_g}(\mathbb{B}^{(i)})}{\arg\min}\ \epsilon_{\mu^{\alpha_g^{(i)}}}(\hat{h}) \tag{8}$$

Here, $\alpha_g^{(i)}$ is associated with the i^{th} iteration of the ERM baseline, which a measure over $\mathbb{B}^{(i)}$ with a set cardinality i. In other words, at each iteration i, we aim to select the prototype domain that incurs the highest generalization error for the ERM baseline. This procedure is described in Fig. 6A for ImNet-A.

Notations. The final multi-domain dataset is denoted by $\mathbb{B} = \cup_{m=1}^{M} \mathcal{B}_m$. $\mathbb{B}$ is a collection of paired samples from all the domains, *i.e.*, $(x, y) \in \mathbb{B}$ with M being the total number of prototype domains. Let J denote the number of categories in the prototype domains.

Algorithm 1 Training the MDN base model.

require: Multi-domain dataset $\mathbb{B}$, Initialize the parameters $\theta_G, \theta_{H_g}, \theta_{H_m}$ (of G, H_g, H_m respectively) from an ImageNet trained model. Weight for *OOD* loss, $w_{\text{ood}} = 0.1$
while the training has not converged **do**
  $(x, y) \leftarrow \mathbb{B}$ batch (equal no. of samples from $\mathcal{B}_m$)
  Comp. $\mathcal{O} : \sum_{(x,y)} \{\text{CE}(y, h_g^b)\}$ using H_g head
  for each m' in $1, 2, .., M$ **do**
    Perform batch-norm-masking (BNM) i.e. update the batch-norm statistics of $H_{m'}$ only for samples from $\mathcal{B}_{m'}$ though other samples are also inferred via $H_{m'}$ for the OOD objective.
    Comp. $O_{m'} : \sum_{(x,y) \in \mathcal{B}_{m'}} \{\text{CE}(y, h_{m'}^b)\}$ via $H_{m'}$
    Comp. $O_{m',\text{ood}} : \sum_{(x,y) \in \mathbb{B} \setminus \mathcal{B}_{m'}} \{-log(h_{m',\text{ood}}^b)\}$ via $H_{m'}$
  end for
  update $\theta_G, \theta_{H_g}, \theta_{H_m}$ by minimizing the following objective: $\mathcal{O} + \sum_{m'=1}^{M} (O_{m'} + w_{\text{ood}} O_{m',ood})$
  update affine parameters of batch-norm for each head H_m using only corresponding in-domain samples from $\mathcal{B}_m$
end while

3.3.2 *Multi-domain network (MDN) architecture*

The multi-domain network architecture is devised by drawing motivation from the Specific+Generic (*SG*) MDL configuration discussed in Sec. 3.2.3. To this end, we employ a multi-head architecture consisting of a shared CNN backbone, G followed by multiple prototype support heads, *i.e.*, $\{H_k\}_{k=1}^{K}$ (see Fig. 5A). Here, $K = M + 1$. Thus, the support heads include M number of prototype-specific (*PS*) heads $\{H_m\}_{m=1}^{M}$ alongside a single domain-generic (or *ERM*) head H_g. This is inline with the Specific+Generic (*SG*) MDL configuration, as the classifier heads $\{H_m\}_{m=1}^{M}$ and H_g subscribe to the hypothesis supports $\{\mathcal{A}_{PS}^{\alpha_m}\}_{m=1}^{M}$ and $\mathcal{A}_g^{\alpha_g}$ respectively (see Fig. 3). Accordingly, the H_g head is trained on samples from all the prototype domains, $\mathbb{B}$ (inline with *ERM* configuration) whereas individual H_m heads are trained only on the respective prototype domain samples, $\mathcal{B}_m$ (inline with *PS* configuration).

a) Requirement of explicit specificity. While training the MDN architecture, our prime objective is to leverage the advantages of the Specific+generic (*SG*) MDL discussed in Sec. 3.2.3. In an ideal scenario, one should learn fully un-shared networks as shown in Fig. 4A. However, owing to higher computation complexity sharing the backbone network G seems to be a better alternative. In an end-to-end training of MDN (with shared G), the features extracted at the output of G must retain a mixture of both domain-generic and domain-specific attributes. While H_g would attend to the domain-generic factors, domain-specific heads H_m should explicitly attend to the domain-specific features. To this end, we hypothesize that instilling out-of-domain *(OOD)* detection capability would guide the prototype specific heads to explicitly attend prototype distinctive attributes, implying better specificity. In the absence of the additional *OOD* objective the prototype-specific hypothesis supports tends to be closer to the *ERM*-support (refer Fig. 4C), thereby limiting the overall support coverage which is detrimental to client-side extensibility.

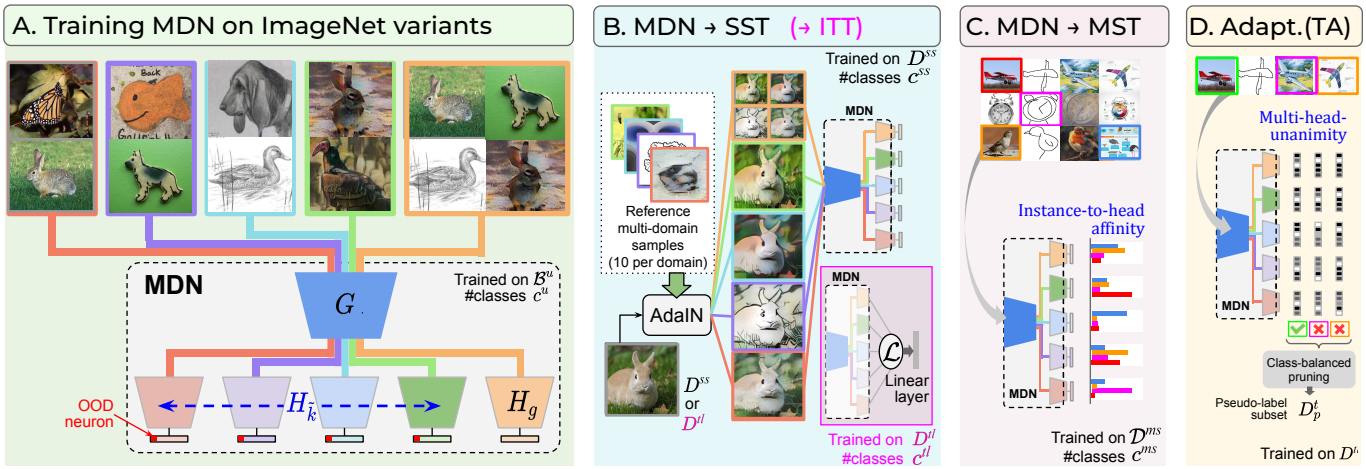


Fig. 5. **A.** MDN is trained on ImageNet-variant to yield domain-generic and domain-specific representations. **B.** MDN extends to single-source tasks (transfer learning and single-source domain adaptation) through head-specific input stylization. **C.** MDN extends to multi-source tasks through instance-to-head affinity. **D.** Unanimity based pseudo-labeling criterion is utilized for target adaptation.

Algorithm 2 Training for ITT

1: **require:** Transfer learning dataset D^{tl}. Initialize $\theta_G, \theta_{\tilde{H}_g}, \theta_{\tilde{H}_m}$ from the MDN while C^{tl} is initialized randomly. Here, $\tilde{H}$ denotes MDN head w/o the last layer. Stylization probability β is set as 0.25
2: **while** the training has not converged **do**
3: $(x, y) \leftarrow$ minibatch sampled from D^{tl}
4: **for** m in $\{1, 2, ...M\}$ **do**
5: $x_m^b \leftarrow$ reference style images sampled from stored set
6: $\tilde{h}_m = \tilde{H}_m \circ G \begin{cases} x & Prob.\ 1-\beta \\ \texttt{AdaIN}(x, x_m^b) & Prob.\ \beta \end{cases}$
7: **end for**
8: $\tilde{h}_g = \tilde{H}_g \circ G(x)$, *i.e.* no stylization for $\tilde{H}_g$.
9: $\mathcal{F} = \mathcal{L}(\tilde{h}_g, \{\tilde{h}_m\}_{m=1}^M)$, here $\mathcal{L}$ denotes the aggregation operation ($\mathcal{L}$:add or $\mathcal{L}$:cat denotes addition or concatenation based aggregation choices).
10: $\hat{y}^{tl} = C^{tl} \circ \mathcal{F}$,
11: **update** $\theta_{C^{tl}}$ by minimizing $\mathcal{O} : \sum_{(x,y)}\{\text{CE}(y, \hat{y}^{tl})\}$
12: **end while**

3.3.3 *Proposed multi-domain learning (MDL) strategy*

Here, we discuss the training objectives for MDL. For the generic-head, H_g, the final *softmax* output is denoted by $h_g^b = H_g \circ G(x)$. Here, $\circ$ represents the functional composition. For the generic-head we employ the following training objective; $\mathcal{O} : \mathbb{E}_{(x,y)\in\mathbb{B}}\{\text{CE}(y, h_g^b)\}$.

Here, CE represents the cross-entropy loss. However, for each domain specific head $H_{m'}$, we first extend the dimension of last layer output to $(J+1)$ by introducing a single *OOD* neuron. The output of *softmax* applied over the $(J+1)$ logits is denoted by $h_{m'}^b = H_{m'} \circ G(x)$. Note that, the last neuron $h_{m',ood}^b$ must activate only for out-of-domain samples, *i.e.* for $(x, y) \in \mathbb{B} \setminus \mathcal{B}_{m'}$. Whereas, for in-domain samples, *i.e.* for $(x, y) \in \mathcal{B}_{m'}$, the model should activate the neuron corresponding to the true class-id. Thus, for $H_{m'}$, we devise the objectives $O_{m'}$ and $O_{m',ood}$ defined in line 7 and 8 of Algo. 1 respectively. The final objective for an end-to-end training of MDN is given by line 10 in Algo. 1.

a) Enforcing explicit specificity via *OOD* & *BNM*. Note that the training objectives for the domain-specific heads put equal emphasis on both in-domain and out-of-domain samples. However, we aim to realize a greater specificity by explicitly prioritizing the learning on in-domain samples. We achieve this by proposing a modified batch normalization (BN) strategy namely, batch-norm-masking *(BNM)*. In contrast to the usual BN [39], we compute batch-statistics using only in-domain samples and update the affine BN parameters only for the in-domain samples. *BNM* favours *OOD* detection, which allows the model to focus on solving the in-domain classification task over the *OOD* detection task.

3.4 Extending MDN at client-side

Once the vendor shares the multi-domain network with the client, the client can extend it to a downstream task of their choice. The downstream task could be inductive (task-shift) or tranductive (domain-shift). Different downstream tasks are shown in Fig. 2B.

3.4.1 *Extending for Inductive-Task-Transfer (ITT)*

ITT refers to the widely used transfer-learning setting where we intend to evaluate the extensibility of learned representations to a diverse set of recognition tasks.

Problem setting. We are given access to a new dataset D^{tl} consisting of labelled images. In relation to the multi-domain data $\mathbb{B}$, D^{tl} exhibits a varied shift in task, in the absence of domain shift. Following linear evaluation protocol, we introduce a single linear layer C^{tl} to perform a classification over c^{tl} classes as per the categorization in D^{tl}.

Learning strategy. D^{tl} may or may not be similar to the domains used in $\mathbb{B}$. Thus, we leverage the domain-specificity knowledge of the MDN heads by generating stylized versions of D^{tl} to construct a virtual multi-domain data $\mathcal{D}^{tl} = \cup_{m=1}^M D_m^{tl}$ (see Fig. 5B). Here, samples for each virtual domain, D_m^{tl} is obtained by stylizing the D^{tl} samples via AdaIN [45] using a small set of reference style images (10 images per prototype domain) stored from $\mathcal{B}_m$. No stylization is applied for H_g. This makes $\mathcal{D}^{tl}$ to resemble samples from the multi-domain data $\mathbb{B}$. Next, we prepare the general purpose representation for each D^{tl} instance, $\mathcal{F}$ by aggregating penultimate domain-specific features from all the MDN heads. These are obtained by forwarding the corresponding head-specific stylizations of the same instance. The aggregation could be simply a concatenation or an addition operation. The training objective is simply cross-entropy as expressed in line 11 of Algo. 2.

TABLE 2
Notation table for vendor-side and client-side approach (Sec. 3.3, 3.4)

	Sym.	Description
Network components	G	Shared backbone (*ResNet*) of MDN
	H_m	Domain specific head of MDN
	H_g	The generic head of MDN
	$\tilde{H}$	Represents MDN head w/o the classifier layer
	C^{tl}	Last linear layer for ITT
	C^{ss}_m	SST classifier post $\tilde{H}_{\tilde{k}}$
	C^{ss}_g	SST classifier post $\tilde{H}_g$
	C^{ms}_m	MST classifier post $\tilde{H}_{\tilde{k}}$
	C^{ms}_g	MST classifier post $\tilde{H}_g$
Misc. sizes	M	Number of prototype domains; $m \in \{1, \ldots, M\}$
	J	Number of classes in prototype domains; $j \in \{1, \ldots J\}$
	K	Number of heads in MDN; $k \in \{1, 2, \ldots, K\}$
	I	Number of classes in client-side datasets. I^{ss}, I^{ms}, I^{tl} denotes the same for single-source, multi-source and transfer learning datasets. $i \in \{1, \ldots, I\}$

3.4.2 Extending for Single-Source-Transfer (SST)

We propose single-source transfer as a source-only initialization step before performing single-source DA (SSDA).

Problem Setting. We are given access to a single source domain dataset, D^{ss} consisting of images with paired category annotations. In relation to the multi-domain data $\mathbb{B}$, D^{ss} exhibits a diverse shift across both domain and task.

Learning strategy. We follow a similar training strategy proposed for ITT while allowing fine-tuning. Instead of aggregating penultimate features, we enforce the cross-entropy loss on all heads independently (see Fig. 5B). The learning retains head-specificity characteristics of the base-MDN as each head is trained on correspondingly stylized samples.

3.4.3 Extending for Multi-Source-Transfer (MST)

We propose MST as a source-only preparation of MDN (see Fig. 5C) which is useful for both domain generalization (DG) and multi-source domain adaptation (MSDA).

Problem setting. We are given access to a multi-source dataset $\mathcal{D}^{ms} = \cup_{l=1}^{L^{ms}} D^{ms}_l$ which is a collection of L^{ms} source domains each consisting of images with paired category annotations (with I^{ms} number of classes). In relation to $\mathbb{B}$, $\mathcal{D}^{ms}$ exhibits a diverse shift across both domain and task.

Learning strategy. Unlike D^{tl} and D^{ss}, $\mathcal{D}^{\text{ms}}$ is a mixture of samples from multiple sources. We find that a particular MDN head or a combination is best suited only for a subset of multi-source samples. It turns out that such association is not present at the domain-level but at an instance level (see Section 4.2, Observation 3). We leverage these instance-level associations to utilize the instilled head-specificity knowledge of MDN. For each instance in $\mathcal{D}^{\text{ms}}$, we introduce an *instance-to-head* affinity score denoted as λ_k where k is the index over all the MDN heads (including the generic-head). The prime question that arises is how to determine the affinity values λ_k.

a) Determining instance-level affinities. Let, $\{y^{\text{ms}}_i\}_{i=1}^{I^{ms}}$ denote the class-labels for $\mathcal{D}^{ms}$ and $h^b_{k,j}$ denotes the confidence of j^{th} class computed for each MDN head $\tilde{\mathcal{H}}_k$, $k = 1, 2, \ldots, K$. The instance-level soft-affinities are determined by extending a transferability measure for the proposed multi-head MDN. Intuitively,

Algorithm 3 Training for MST

1: **require:** Multi-source dataset $\mathcal{D}^{ms}$. Initialize θ_G, $\theta_{\tilde{H}_g}$, $\theta_{\tilde{H}_m}$ from MDN. Here, $\tilde{H}$ denotes MDN head w/o the last layer. Random initialization of C^{ms}_m, C^{ms}_g.

Step 1: *Initializing instance-level affinities*

2: **for** i in $\{1, 2, \ldots, I^{ms}\}$ **do** (*i.e.* over classes in $\mathcal{D}_{ms}$)
3: Obtain $\mathcal{D}^{ms}_i = \{(x, y) : (x, y) \in \mathcal{D}^{ms} \text{ and } y = i\}$
4: Precompute $e_{k,j,i} = \frac{1}{|\mathcal{D}^{ms}_i|} \sum_{(x,y) \in \mathcal{D}^{ms}_i} h^b_{k,j}$ where $h^b_{k,j}$ denotes confidence of j^{th} prototype-class from k^{th} head of MDN.
5: **end for**
6: Initialize instance-level affinity, $\lambda^{(x)}_k = \sum_j h^b_{k,j} \frac{e_{k,j,y}}{\sum_i e_{k,j,i}}$ $\forall (x, y) \in \mathcal{D}^{ms}$ and $\forall\ k$ (using the precomputed $e_{k,j,i}$).

Step 2: *Training process*

7: **while** the training has not converged **do**
8: $(x, y) \leftarrow$ minibatch sampled from $\mathcal{D}^{ms}$
9: Compute $h^{ms}_k = \textit{softmax}(\lambda^{(x)}_k C^{ms}_k \circ \tilde{H}_k \circ G(x))$
10: Compute $\mathcal{O} : \sum_{(x,y)} \{\text{CE}(y, h^{ms}_k)\}$
11: **update** $\theta_G, \theta_{\tilde{H}_m}, \theta_{\tilde{H}_g}, \theta_{C^{ms}_m}, \theta_{C^{ms}_g}, \lambda$ by minimizing $\mathcal{O}$. Note that λ is updated by backprop during training
12: **end while**

affinity value $\lambda^{(x)}_k$ for an instance x (see line 6 of Algo. 3) can be seen as the expected confidence of a classifier-head in the I^{ms} space, considering J as a latent space. We use normalized $\lambda^{(x)}_k$ to initialize the affinity score separately for each source instance x.

b) Training objective. After initializing the affinities, we fine-tune the base MDN by modifying the last layer of each MDN head k to perform classification over c^{ms} classes (*i.e.* $C^{ms}_k \circ \tilde{H}_k$). The affinities are then updated by back-propagation. The training objective is as expressed in line 10 in Algo. 3. Unlike in SST, the generic head is treated as one of the candidate heads similar to other domain-specific ones. This allows samples from novel sources to have a higher affinity towards the generic-head. For such samples, domain-generic knowledge is more suitable than domain-specific.

3.4.4 Extending for Target Adaptation (TA)

TA refers to the adaptation phase (see Fig. 5D) as required for both single source DA and multi source DA.

Problem setting. After extending MDN for SST and MST, TA is performed to adapt the extended models to new unlabeled target data, D^t. D^t exhibits a diverse input domain-shift w.r.t. D^{ss} or $\mathcal{D}^{ms}$, but no shift in the output task, though entirely unlabeled.

Learning strategy. We draw motivation from the pseudo-label based self-training approaches [48]. Let D^t_p denote the pseudo-label subset. In literature, several approaches rely on *"confidence-thresholding"* [49], [50] as the selection-criteria. Here, a target sample is selected if its maximum class confidence exceeds a certain threshold value. Such approaches are shown to be suffering from information redundancy and label noise [33], [51].

a) Multi-head-unanimity (mhu). Acknowledging the above, we devise a novel selection criteria by leveraging the multi-head predictions of MDN. In order to remove label-noise, we essentially need easy samples which we argue should be correctly predicted irrespective of the domain. To this end, we propose a criterion where a target sample is selected only if its predictions across all

Algorithm 4 Training for TA

require: Single-source (or multi-source) data D^{ss} (or D^{ms}). Unlabeled target data D^t. Initialize θ_G, $\theta_{\tilde{H}_m}$, $\theta_{\tilde{H}_g}$, $\theta_{C_g^{ss}}$, $\theta_{C_m^{ss}}$ from SST (or MST) trained model. Here, $\tilde{H}$ denotes MDN head w/o the last layer. Hyperparameters γ.

Step 1: *Pseudo-labeling process*

for k in $\{1, 2, ..., K\}$ **do** (*i.e.* over all MDN heads)

  Compute $h_k^t \leftarrow C_k^{ss} \circ \tilde{H}_k \circ G(x), \forall x \in D^t$

end for

Multi-head-unanimity (*mhu*) for each instance $x \in D^t$, $u^t(x) \leftarrow \prod_{k \neq k'} \mathbb{1}(\underset{i \in \{1,...,I^{ss}\}}{\text{argmax}}\ h_{k,i}^t = \underset{i \in \{1,...,I^{ss}\}}{\text{argmax}}\ h_{k',i}^t)$

where $h_{k,i}^t$ denotes confidence of k^{th} head for i^{th} class of D^{ss}

Prepare pseudo-labeled subset with samples having *mhu* 1, $D_p^t \leftarrow \{(x, y_p) : x \in D^t, u^t(x) = 1, y_p = \underset{i \in \{1,...,I^{ss}\}}{\text{argmax}}\ h_{k,i}^t\ \forall k\}$

Prune D_p^t by selecting samples w/ confidence greater than γ, $D_p^t \leftarrow \{(x, y_p) \in D_p^t : h_k^t > \gamma\ \forall k\}$

Step 2: *Adaptation process*

while training has not converged **do**

  $(x, y) \leftarrow$ minibatch sampled from D^{ss}.

  $(x_p, y_p) \leftarrow$ minibatch sampled from D_p^t.

  $h_m^{ss} \leftarrow C_m^{ss} \circ \tilde{H}_m \circ G(x)$

  $h_g^{ss} \leftarrow C_g^{ss} \circ \tilde{H}_g \circ G(x)$

  $h_m^t \leftarrow C_m^{ss} \circ \tilde{H}_m \circ G(x_p)$

  $h_g^t \leftarrow C_g^{ss} \circ \tilde{H}_g \circ G(x_p)$

  Comp. $\mathcal{O}_m : \sum\{\text{CE}(y, h_m^{ss}) + \text{CE}(y_p, h_m^t)\}$

  Comp. $\mathcal{O}_g : \sum\{\text{CE}(y, h_g^{ss}) + \text{CE}(y_p, h_g^t)\}$

  update θ_G, $\theta_{\tilde{H}_m}$, $\theta_{\tilde{H}_g}$, $\theta_{C_g^{ss}}$, $\theta_{C_m^{ss}}$ by minimizing $\mathcal{O}_g + \sum_m \mathcal{O}_m$

end while

(For multi-source dataset $\mathcal{D}^{ms}$, the same process is followed by replacing h^{ss} with h^{ms} and similarly for other terms)

the heads fire maximally for the same class id (*i.e. "multi-head-unanimity" (mhu)*) (see lines 6-7 in Algo. 4).

b) How do instance-affinities help? While addressing MSDA, the usage of soft-affinities in MST allows the MDN heads to retain head-specificity without respecting the source-specificity of the multi-source data $\mathcal{D}^{ms}$. Essentially, the domain-label information of $\mathcal{D}^{ms}$ remains unused. This allows the samples of a particular source domain to exhibit varied instance-level affinity. Thus, post MST, the MDN heads can be perceived as mix-source ensembles. Therefore, an agreement among such mix-source ensembles reduces the generalization error thereby facilitating selection of reliable and informative target pseudo-labels.

4 Experiments

At the vendor side, we evaluate the impact of Specific+generic (*SG*) MDL to enable extensibility at the client side. At the client side, we evaluate the extensibility of MDN under various task and domain shift benchmarks such as transfer learning (TL), single-source (SSDA) and multi-source domain adaptation (MSDA) and domain generalization (DG).

Datasets. We summarize the datasets used for vendor-side and client-side experiments in Table 3. For MDN training, we utilize ImageNet-Original [42], ImageNet-Rendition [43], ImageNet-Sketch [41] and Stylized-ImageNet [44]. The combined set is referred as ImNet-P. For TL tasks, we utilize the downstream image classification datasets as examined in [9], [17]. For evaluating performance on SSDA, MSDA and DG, we utilize **DomainNet** [32], **PACS** [53], Office-31 [52] and Office-Home [66] datasets.

TABLE 3
Dataset summary for vendor-side and client-side experiments

Vendor-side (K=5)	
ImNet-P (K∼=4)	**ImNet-A (K∼=4)**
ImNet-O (Original) [42]	ImNet-O (Original)
ImNet-Sk (Sketch) [41]	*ImNet-E* (Edge)
ImNet-R (Rendition) [43]	*ImNet-C* (Cartoon)
ImNet-Sty (Stylized) [44]	*ImNet-Sty* (Stylized)

Client Side				
SSDA	**L**	**#c**	**TL**	**#c**
DomainNet [32]	6	345	FGVC Aircraft [55]	100
Office-31 [52]	3	31	Stanford Cars [56]	196
			CIFAR-100 [57]	100
DG / MSDA	**L**	**#c**	CIFAR-10 [57]	10
DomainNet [32]	6	345	Caltech-101 [58]	101
PACS [53]	4	7	Caltech-256 [59]	257
Office-Home [54]	6	65	DTD [60]	47
			Oxford 102 Flowers [61]	102
			Oxford-IIIT Pets [62]	37

TABLE 4
Detailed architecture of MDN and its extensions.

Task	Component	Layers
	G	*ResNet* till *conv4_x*
	$\tilde{H}_1, \tilde{H}_2, \tilde{H}_3, \tilde{H}_4, \tilde{H}_g$	*ResNet conv5_x* → GAP → L2-Norm
ITT	Aggregation layer	$\mathcal{L}(\tilde{H}_1, \tilde{H}_2, \tilde{H}_3, \tilde{H}_4, \tilde{H}_g)$
	$C^{tl}, \mathcal{L} : cat$	FC(5*2048×$\lvert c^{tl} \rvert$)
	$C^{tl}, \mathcal{L} : add$	FC(2048×$\lvert c^{tl} \rvert$)
SST	C_m^{ss}, C_g^{ss}	FC(2048×1024) → ELU → BN → Dropout(0.1) → FC(1024×$\lvert c^{ss} \rvert$)
MST	C_m^{ms}, C_g^{ms}	FC(2048×1024) → ELU → BN → Dropout(0.1) → FC(1024×$\lvert c^{ms} \rvert$)

Implementation Details. We adopt ResNet [67] architecture for MDN. ResNet is split into the shared backbone G and the classifier-heads H_k. All the layers upto *conv4_x* constitute the shared backbone G. *conv5_x* layer is duplicated for each classifier-head H_k. The batch-norm [39] layers in the classifier-heads are replaced with batch-norm-masking *(BNM)* layers. Table 4 contains all architectural details. All the models are trained using Adam [68] optimizer starting with a learning rate of 5e^{-5} and a scheduler reducing the learning rate by half every 5k steps, for a total of maximum 500k steps. We randomly store 10 reference images per domain from $\mathbb{B}$ for stylization in ITT. For DG, SSDA and MSDA, classifier-head predictions are simply averaged at inference time.

4.1 Evaluating our vendor-side strategy

We analyze the different components proposed for our vendor-side strategy as well as compare with state-of-the-art MDL works.

Observation 1. *In the absence of natural multi-domain data (ImNet-P), datasets formed via domain-varying augmentations (ImNet-A) yield competitive performance with MDN.*

Remarks. To this end, we train an MDN network solely on ImNet-A *i.e.* augmented version of a single-domain dataset. We report the obtained results along with those obtained with

TABLE 5
Ablation Table. We apply FACT [63] to all heads and final result is an ensemble of all heads, and we do the same for DRT [64]

	Vendor-data Dataset ablations	Training Strategy MDL Ablations		#r	Client-side: TL A. Finetune	DG on PACS: B. FACT [63]	DG on PACS: C. Ours-DG	MSDA on DomainNet: D. DRT [64]	MSDA on DomainNet: E. Ours-MSDA
Vendor-side	ImNet-O	*ERM* baseline		1	72.32	84.52	84.05	51.3	49.91
	ImNet-A	*ERM* baseline		2	72.57	84.54	84.75	51.34	51.08
		Ours PS	*w/o BNM + OOD*	3	73.15	84.49	85.00	51.42	51.58
			w/ BNM + OOD	4	**75.25**	**84.93**	**86.46**	**51.87**	**52.21**
		Ours SG	*w/o BNM + OOD*	5	74.24	84.74	85.69	51.71	52.25
			w/ BNM + OOD	6	**76.29**	**85.21**	**87.23**	**52.05**	**53.18**
		MDL Prior-arts	SUR [34] ECCV'20	7	73.11	84.57	85.09	51.35	51.14
			URL [65] ICCV'21	8	73.24	84.68	85.39	51.45	51.50
	ImNet-P	*ERM* baseline		9	73.16	84.62	85.25	51.31	51.65
		Ours PS	*w/o BNM + OOD*	10	74.6	85.24	85.03	51.45	51.42
			w/ BNM + OOD	11	**77.21**	**85.82**	**86.31**	**51.85**	**53.02**
		Ours SG	*w/o BNM + OOD*	12	75.58	85.43	85.86	51.65	52.06
			w/ BNM + OOD	13	**78.34**	**86.22**	**87.84**	**52.15**	**53.91**
		MDL Prior-arts	SUR [34] ECCV'20	14	74.01	84.71	85.13	51.39	51.46
			URL [65] ICCV'21	15	74.27	85.37	86.07	51.47	51.71

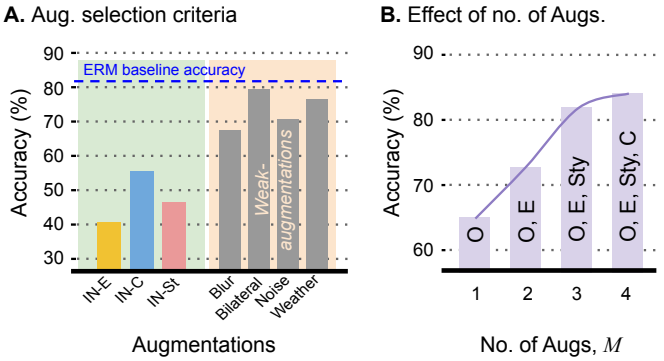


Fig. 6. **A.** For ImNet-A, only augmentations of vendor-side data resulting in significant accuracy drop w.r.t. ERM baseline on original vendor-side data are selected. **B.** Performance of vendor-side trained models on target data when varying number of augmentations M. Performance saturates as M reaches 4. Here, O, E, St, C indicate use of ImNet-O, *ImNet-E*, *ImNet-Sty* and *ImNet-C* respectively (see Sec 4.1.2).

ImNet-P for multiple downstream tasks and datasets (Tables 6-9). We observe that simply using a single-domain dataset with domain-varying augmentations can yield significant improvements w.r.t. competing methods. This underlines the practical utility of the MDL paradigm to learn features that can generalize well to diverse domains and tasks with just domain-varying augmentations.

Observation 2. *Batch-norm-masking (BNM) for out-of-domain (OOD) samples improves the extensibility of MDN and instills domain-specificity in its learned representations.*

Remarks. We evaluate the utility of *OOD* and *BNM* for multiple client-side tasks and vendor-side training strategies, and report the results in Table 5. Table 5 has 15 rows (numbered 1 to 15) and 5 columns (named A to E). Henceforth, we refer to specific cells in Table 5 by their column name and row number (*e.g.* A3, E5, *etc*). We observe consistent improvements, comparing *w/o* and *w/ BNM+OOD*, across different client-side learning strategies, downstream tasks, and MDL strategies (*e.g.* A3 vs. A4, A10 vs. A11, *etc*). In Fig. 7D, we visualize the domain specificity of the representations learned by MDN using t-SNE [69] plots. We pass ImNet-P images through H_1, the branch specific to ImNet-O, and t-SNE is performed on features obtained at the penultimate layer. We observe that in-domain and out-of-domain samples are well separated after incorporating *BNM* and *OOD*.

TABLE 6
Domain generalization (MST) accuracy on DomainNet with ResNet-50. IN-O, IN-A, IN-P indicate ImNet-O, ImNet-A, ImNet-P.

Method	$\mathbb{B}$	→C	→I	→P	→Q	→R	→S	Avg.
MetaReg [29]	IN-O	59.7	25.5	50.2	11.5	64.5	50.1	43.6
DMG [70]	IN-O	65.2	22.1	50.0	15.7	59.6	49.0	43.6
STEAM [71]	IN-O	64.6	**27.0**	54.0	15.8	65.6	52.2	46.5
DMG [70]	IN-A	66.1	21.3	51.9	16.2	60.0	49.3	44.1
Ours (w/o aff)	IN-A	66.2	22.0	52.6	17.2	64.1	52.4	45.8
(MST) (w/ aff)	IN-A	67.6	22.5	55.8	19.3	66.0	57.6	48.1
DMG [70]	IN-P	66.4	21.9	52.4	16.4	60.3	49.8	44.5
Ours (w/o aff)	IN-P	66.7	22.1	53.1	17.4	64.8	52.9	46.1
(MST) (w/ aff)	IN-P	**68.2**	22.9	**56.0**	**19.7**	**66.3**	**58.1**	**48.5**

4.1.1 *Disentangling the gains of ImNet-P over ImNet-O*

Question. *Can the performance gains be attributed simply to the additional data in ImNet-A or ImNet-P against ImNet-O?*

Remarks. Towards this, we evaluate the *ERM* baseline with all three ImageNet variants *i.e.* ImNet-O, ImNet-A and ImNet-P in Table 5. For the challenging MSDA task on DomainNet, we observe that using ImNet-A and ImNet-P improves over ImNet-O by 1.1% and 1.7% respectively (E9, E2 vs. E1, Table 5). Further, our proposed approach improves over the ImNet-A and ImNet-P ERM baselines by 2.1% and 2.2% respectively (E6 vs. E2 and E13 vs. E9, Table 5). We observe a similar trend across all tasks reported in Table 5. Thus, while some gains are observed from additional ImNet-A or ImNet-P data, our vendor-side and client-side strategies effectively leverage the multi-domain knowledge, yielding further significant improvements.

4.1.2 *Effect of number of domain prototypes*

In Fig. 6B, we investigate the influence of number of domain-prototypes on performance. We focus on client-side models (DG on PACS) with varying numbers of domain prototypes utilized during vendor-side training. Because there are several candidate domain prototype combinations to pick from, we prioritize the most diversified ones to obtain the best possible performance at a

TABLE 7
Domain Generalization (MST) on Office-Home with ResNet-18. IN-O, IN-A, IN-P indicate ImNet-O, ImNet-A, ImNet-P respectively.

Method	𝔹	→Ar	→Cl	→Pr	→Rw	Avg.
Jigen [12]	IN-O	53.0	47.5	71.5	72.8	61.2
RSC [72]	IN-O	58.4	47.9	71.6	74.5	63.1
L2A-OT [73]	IN-O	60.6	50.1	74.8	77.0	65.6
DDAIG [74]	IN-O	59.2	52.3	74.6	76.0	65.5
DSON [75]	IN-O	59.4	45.7	71.8	74.7	62.9
ATSRL [76]	IN-O	60.7	52.9	75.8	77.2	66.7
MixStyle [77]	IN-O	58.7	53.4	74.2	75.9	65.5
COPA [78]	IN-O	59.4	55.1	74.8	75.0	66.1
STEAM [71]	IN-O	62.1	52.3	75.4	77.5	66.8
FACT [63]	IN-O	60.3	54.8	74.5	76.5	66.5
FACT [63]	IN-A	60.8	55.4	75.1	77.0	67.1
Ours (w/o aff)	IN-A	59.5	52.5	74.8	77.3	66.1
(MST) (w/ aff)	IN-A	63.5	57.4	76.7	78.5	69.0
FACT [63]	IN-P	61.7	55.7	75.6	77.5	67.6
Ours (w/o aff)	IN-P	60.5	53.4	74.5	77.8	66.5
(MST) (w/ aff)	IN-P	**64.2**	**57.8**	**77.0**	**79.1**	**69.5**

TABLE 8
Single-Source DA (SST → TA) on Office-31 with ResNet-50. IN-O, IN-A, IN-P indicate ImNet-O, ImNet-A, ImNet-P respectively.

Method	𝔹	A(→W	→D)	D(→W	→A)	W(→D	→A)	Avg.
Single Source Transfer								
CAN(SO) [79]	IN-O	68.4	68.9	96.7	62.5	99.3	60.7	76.1
CAN(SO) [79]	IN-A	68.9	69.5	97.1	62.7	99.3	61.2	76.4
Ours (w/o Sty)	IN-A	69.2	70.5	97.3	63.1	99.3	62.3	76.9
(SST) (w/ Sty)	IN-A	69.6	72.8	97.9	64.4	99.4	62.7	77.8
CAN(SO) [79]	IN-P	69.3	69.7	97.4	62.8	99.4	61.8	76.7
Ours (w/o Sty)	IN-P	69.7	71.5	97.4	63.8	99.4	63.1	77.5
(SST) (w/ Sty)	IN-P	70.2	73.5	98.0	64.7	99.4	63.4	78.2
Single Source Domain Adaptation								
DMRL [80]	IN-O	90.8	93.4	99.0	73.0	100	71.2	87.9
GSDA [81]	IN-O	95.7	94.8	99.1	73.5	100	74.9	89.7
SFIT [82]	IN-O	91.8	89.9	98.7	73.9	99.9	72.0	87.7
CAN [79]	IN-O	94.5	95.0	99.1	78.0	99.8	77.0	90.6
d-SNE* [83]	IN-O	96.6	94.6	99.1	75.5	100	74.2	90.0
SRDC [84]	IN-O	95.7	95.8	99.2	76.7	100	77.1	90.8
RSDA-MSTN [85]	IN-O	96.1	95.8	99.3	77.4	100	78.9	91.1
SHOT++ [86]	IN-O	90.4	94.3	98.7	76.2	99.9	75.8	89.2
SHOT [87]	IN-O	90.9	93.1	98.8	74.5	99.9	74.8	88.7
SFIT [82]	IN-O	91.8	89.9	98.7	73.9	99.9	72.0	87.7
FAA [88]	IN-O	92.3	94.4	99.2	80.5	99.7	78.7	90.8
RADA [89]	IN-O	96.2	**96.1**	99.3	77.5	100	77.4	91.1
RFA [90]	IN-O	92.8	93.0	99.1	78.0	100	77.7	90.2
FixBi [91]	IN-O	96.1	95.0	99.3	78.7	100	79.4	91.4
FixBi [91]	IN-A	96.3	95.4	99.5	79.4	99.9	79.7	91.7
Ours (conf)	IN-A	91.9	92.1	97.7	78.8	99.8	78.1	89.7
(SST→TA) (mhu)	IN-A	96.4	95.6	99.4	80.6	100	80.1	92.0
FixBi [91]	IN-P	96.8	96.0	99.7	80.0	99.9	80.4	92.1
Ours (conf)	IN-P	92.2	93.1	98.1	79.1	100	79.4	90.3
(SST→TA) (mhu)	IN-P	**97.1**	95.8	**99.7**	**82.2**	**100**	**81.8**	**92.8**

lower M. In other words, the prototype with the highest error for the ERM baseline model is picked first. The sequence is shown in Fig. 6B as *ImNet-E*, *ImNet-St*, and *ImNet-C*. When we use this order to select prototype domains for $M = \{1, 2, ..., 4\}$, we observe that performance saturates as M approaches 4. As a consequence, we conclude that adding additional domain prototypes would not result in a significant performance gain. A similar process is used to choose prototype domains for ImNet-P.

4.1.3 Comparisons with prior MDL works

We evaluate the generalizability of our proposed vendor-side specific+generic (*SG*) framework with existing Multi-Domain Learning (MDL) works [34], [65]. These prior MDL works report only few-shot multi-domain transfer learning results where vendor-side is similar to our setup *i.e.* a multi-domain dataset is used for training. However, in client-side, they use a support set and a query set (source and target in our setup) but there is no support-query domain shift. Thus, we report results by training with their MDL strategies on both ImNet-A and ImNet-P, along with different downstream task algorithms. We also evaluate the standard *ERM* baseline, where all domain data is used without considering the domain labels.

TABLE 9
Multi-Source DA (MST → TA) on Office-Home with ResNet-50. IN-O, IN-A, IN-P indicate ImNet-O, ImNet-A, ImNet-P respectively.

Method	𝔹	→Ar	→Cl	→Pr	→Rw	Avg.
MIMFTL [92]	IN-O	72.6	64.3	81.9	83.1	75.5
DECISION [93]	IN-O	74.5	59.4	84.4	83.3	75.4
WAMDA [94]	IN-O	71.9	61.4	84.1	82.3	74.9
CAiDA [95]	IN-O	**75.2**	60.5	84.7	**84.2**	76.2
DARN [96]	IN-O	$70.0_{\pm 0.4}$	$68.4_{\pm 0.1}$	$82.7_{\pm 0.2}$	$83.9_{\pm 0.2}$	76.2
SImpAI [97]	IN-O	$70.8_{\pm 0.2}$	$56.3_{\pm 0.2}$	$80.2_{\pm 0.3}$	$81.5_{\pm 0.3}$	72.2
MIAN [98]	IN-O	$69.9_{\pm 0.3}$	$64.2_{\pm 0.7}$	$80.9_{\pm 0.4}$	$81.5_{\pm 0.2}$	74.1
CMSDA [99]	IN-O	$71.5_{\pm 0.3}$	$67.7_{\pm 0.2}$	$84.2_{\pm 0.3}$	$83.0_{\pm 0.4}$	76.6
CMSDA [99]	IN-A	$71.9_{\pm 0.2}$	$69.2_{\pm 0.4}$	$85.2_{\pm 0.3}$	$83.5_{\pm 0.2}$	77.45
Ours (conf)	IN-A	$71.8_{\pm 0.1}$	$72.9_{\pm 0.3}$	$83.3_{\pm 0.4}$	$82.1_{\pm 0.2}$	77.5
(MST→TA) (mhu)	IN-A	$73.2_{\pm 0.1}$	$76.2_{\pm 0.2}$	$85.3_{\pm 0.3}$	$82.9_{\pm 0.1}$	79.4
CMSDA [99]	IN-P	$72.0_{\pm 0.3}$	$69.5_{\pm 0.3}$	$85.6_{\pm 0.4}$	$83.4_{\pm 0.2}$	77.6
Ours (conf)	IN-P	$72.0_{\pm 0.1}$	$70.2_{\pm 0.4}$	$85.9_{\pm 0.3}$	$83.5_{\pm 0.1}$	77.9
(MST→TA) (mhu)	IN-P	$73.8_{\pm 0.2}$	$\mathbf{76.5}_{\pm 0.3}$	$\mathbf{85.5}_{\pm 0.2}$	$83.3_{\pm 0.1}$	**79.8**

Transfer Learning (TL). We report the average accuracy over downstream transfer learning to 9 datasets (Table 3, ITT). With the ImNet-A vendor-side dataset, prior MDL works SUR [34] and URL [65] achieve marginal improvements of ∼0.6% over the ERM baseline (A7, A8 vs. A2 in Table 5). Whereas, our proposed prototype-specific (PS) baseline and specific+generic (SG) MDL yield gains of ∼2% and ∼3% over URL (A4, A6 vs. A8 in Table 5). With the more diverse ImNet-P vendor-side dataset, our PS baseline and SG MDL achieve higher gains of ∼3% and ∼4% over the equivalent URL results (A11, A13 vs. A15 in Table 5).

Domain Generalization (DG). Following standard DG evaluation methodology, we report the average target accuracy considering each domain as target and remaining domains as multi-source domains. With ImNet-P vendor-side data, SUR underperforms and URL improves by ∼0.8% w.r.t. the ERM baseline respectively (C14, C15 vs. C9, Table 5) while our final SG variant yields a gain of 1.8% over URL (C13 vs. C15, Table 5).

Multi-Source DA (MSDA). We follow the same evaluation protocol as in DG. With ImNet-P vendor-side data, SUR again underperforms w.r.t. ERM baseline while URL improves marginally by 0.06% (E14, E15 vs. E9, Table 5). Our final SG variant improves by 2.2% over URL (E13 vs. E15, Table 5).

4.2 Evaluating our client-side strategy

We analyze the different components proposed for client-side algorithms and compare with state-of-the-art DG, SSDA, MSDA and TL works on a wide range of standard benchmarks.

Observation 3. *Instance-level affinity better suits MST and subsequent TA than domain-level affinity, and results in improved performance over any single branch of MDN.*

Remarks. We compare the affinities at domain-level and at instance-level for Art-Painting (→Ar) domain from PACS in

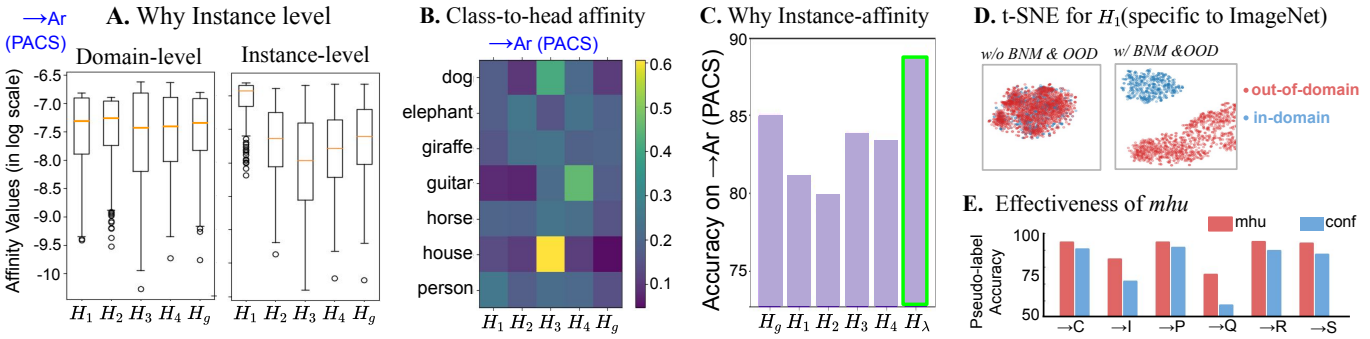


Fig. 7. **A.** Comparison of domain-level (left) and instance-level (right) affinity for a single domain using box plots (see Observ. 3, Sec. 4.2). **B.** Some classes have higher affinities towards certain heads, while others are distributed. Together with **A**, this demonstrates the necessity of instance-level affinity. **C.** Combining based on affinity (shown as H_λ) works better than any single branch. **D.** t-SNE plots for representations learned w/ and w/o *OOD & BNM* (see Observ. 2, Sec. 4.1). **E.** *mhu* reduces label noise compared to confidence thresholding (see Observ. 4, Sec. 4.2).

TABLE 10
Single-best DA (SST → TA) on DomainNet w/ ResNet-50. SO, IN-O, IN-A, IN-P indicate source only, ImNet-O, ImNet-A, ImNet-P.

Method	𝔹	R(→C	→I	→P)	C(→Q	→S)	P→R	Avg.
Single Source Transfer								
M3SDA (SO) [32]	IN-O	48.4	22.2	49.4	11.1	41.0	54.5	37.8
M3SDA (SO) [32]	IN-A	48.5	22.2	49.6	11.5	41.1	54.4	37.9
Ours (w/o Sty)	IN-A	49.4	17.1	48.5	14.2	45.8	53.5	38.1
(SST) (w/ Sty)	IN-A	51.6	18.5	49.6	16.0	47.0	52.3	39.1
M3SDA (SO) [32]	IN-P	48.8	22.6	49.5	11.7	41.3	54.7	38.1
Ours (w/o Sty)	IN-P	49.9	17.4	48.8	14.3	46.0	53.8	38.4
(SST) (w/ Sty)	IN-P	51.7	18.8	50.1	18.4	47.2	52.6	39.8
Single Source Domain Adaptation								
CGDM [100]	IN-O	50.8	38.5	54.5	17.8	54.9	49.7	44.3
MIMTFL [92]	IN-O	51.7	19.0	47.6	12.3	43.1	55.4	38.2
DRT [64]	IN-O	56.2	26.6	53.4	12.2	55.5	44.8	41.5
DRT [64]	IN-A	56.8	26.3	54.8	15.3	56.9	45.4	42.6
Ours (conf)	IN-A	54.1	20.8	51.4	21.1	47.2	56.5	41.8
(SST→TA) (mhu)	IN-A	58.0	23.2	53.1	24.9	49.4	57.5	44.3
DRT [64]	IN-P	56.7	26.7	55.3	15.5	57.2	45.9	42.9
Ours (conf)	IN-P	54.4	20.2	51.5	21.5	47.9	57.1	42.1
(SST→TA) (mhu)	IN-P	**58.3**	**23.5**	**53.8**	**25.6**	**49.6**	**58.2**	**44.8**

TABLE 11
Domain generalization (MST) accuracy on PACS with ResNet-18. IN-O, IN-A, IN-P indicate ImNet-O, ImNet-A, ImNet-P respectively.

Method	𝔹	→Ar	→Ca	→Ph	→Sk	Avg.
DSON [75]	IN-O	84.7	77.7	95.9	82.2	85.1
ASR [101]	IN-O	84.8	81.8	96.1	82.6	86.3
COPA [78]	IN-O	83.3	79.8	94.6	82.5	85.1
STEAM [71]	IN-O	85.5	80.6	**97.5**	82.9	86.6
L2D [102]	IN-O	81.4	79.6	95.5	80.6	84.3
MixStyle [77]	IN-O	$84.1_{\pm 0.4}$	$78.8_{\pm 0.4}$	$96.1_{\pm 0.3}$	$75.9_{\pm 0.9}$	83.7
DIRT-GAN [103]	IN-O	$82.6_{\pm 0.4}$	$76.4_{\pm 0.3}$	$95.6_{\pm 0.5}$	$79.9_{\pm 0.2}$	83.6
MBDG [104]	IN-O	$80.6_{\pm 1.1}$	$79.3_{\pm 0.2}$	$97.0_{\pm 0.4}$	$85.2_{\pm 0.2}$	85.6
RSC [72]	IN-O	$83.4_{\pm 0.8}$	$80.3_{\pm 0.4}$	$96.0_{\pm 0.0}$	$80.9_{\pm 1.7}$	85.2
ATSRL [76]	IN-O	$85.8_{\pm 0.6}$	$80.7_{\pm 0.5}$	$97.3_{\pm 0.3}$	$77.3_{\pm 0.5}$	85.3
SelfReg [105]	IN-O	$82.3_{\pm 0.5}$	$78.4_{\pm 0.7}$	$96.2_{\pm 0.3}$	$77.5_{\pm 0.8}$	83.6
FACT [63]	IN-O	$85.4_{\pm 0.3}$	$78.4_{\pm 0.3}$	$95.1_{\pm 0.3}$	$79.1_{\pm 0.7}$	84.5
FACT [63]	IN-A	$86.0_{\pm 0.3}$	$79.6_{\pm 0.3}$	$95.7_{\pm 0.4}$	$79.7_{\pm 0.6}$	85.2
Ours (w/o aff)	IN-A	$83.4_{\pm 0.4}$	$81.3_{\pm 0.4}$	$93.9_{\pm 0.5}$	$80.2_{\pm 0.2}$	84.7
(MST) (w/ aff)	IN-A	$86.6_{\pm 0.3}$	$81.5_{\pm 0.3}$	$96.4_{\pm 0.4}$	$84.2_{\pm 0.2}$	87.2
FACT [63]	IN-P	$86.4_{\pm 0.3}$	$80.4_{\pm 0.2}$	$96.5_{\pm 0.4}$	$81.6_{\pm 0.3}$	86.2
Ours (w/o aff)	IN-P	$83.8_{\pm 0.4}$	$81.8_{\pm 0.2}$	$94.8_{\pm 0.5}$	$80.5_{\pm 0.2}$	85.2
(MST) (w/ aff)	IN-P	$\mathbf{87.2}_{\pm 0.3}$	$\mathbf{82.0}_{\pm 0.2}$	$97.1_{\pm 0.3}$	$\mathbf{84.7}_{\pm 0.2}$	**87.8**

Fig. 7A, 7B. For domain-level, we observe very high within-head variability with no observed affinity for any head. For instance-level, we observe that the affinity for the current head is significantly higher than other heads, with reduced within-head variance. Additionally, we observe that certain classes have preferences towards certain branches, while other classes do not have any particular preference (Fig. 7B). Instance-level affinities are effective for these classes. Fig. 7C demonstrates how using affinity λ_k to combine the predictions performs better than any other single branch. We also evaluate the effectiveness of initializing affinities (refer Section 3.4.3) by testing it on →Ar (PACS). On random initialization, the performance drops from 91.22 to 90.49 highlighting the importance of affinity initialization.

Observation 4. *Multi-head unanimity (mhu) is well-tailored to aid TA in MDN, especially in challenging settings.*

Remarks. The use of mhu reduces the pseudo-label noise by exploiting both domain-specific and domain-generic heads in MDN, as depicted in Fig. 7E. Note that, while label noise reduction may not translate to a proportional performance improvements in all cases, it will yield significant gains in difficult settings as they tend to have higher label noise. *e.g.* →Q in DomainNet is highly challenging but mhu achieves +7.8% over conf (Table 13).

4.2.1 *Extensibility of prior DA works with MDN*

Observation 5. *Our proposed client-side algorithms are better equipped, than existing DG and DA algorithms, to properly leverage the capabilities of the learned MDN representations.*

Remarks. We train the SOTA DG work, FACT [63], and the SOTA MSDA work, DRT [64], initialized with different variants of MDN trained on ImNet-A and ImNet-P (see Table 5). We observe that Ours-DG consistently outperforms FACT (column C2-15 vs. B2-15) and Ours-MSDA consistently outperforms DRT (column E2-15 vs. D2-15). Thus, our proposed client-side algorithms better leverage the MDN knowledge compared to existing DG and DA works.

4.2.2 *Comparisons with prior TL and DA works*

a) Extensibility to task-shifts (TL). Table 12 reports the accuracy using the linear evaluation protocol [9] for transfer learning. MDN outperforms the standard ImageNet trained model on 7 out of 9 datasets. Note that [9] performs robust training with 3-step adversaries *i.e.* 3 backward steps for adversary generation and 1 backward step for model training. Thus, MDN achieves similar performance to [9] with ∼4 times lower computation (*i.e.* faster training). This demonstrates the extensibility of MDN to task-shifts across a diverse set of downstream datasets with varying image

TABLE 12
Transfer accuracy of MDN on downstream tasks (ITT) using linear evaluation with different stylization and aggregation settings ($\mathcal{L}$). See Sec 4.2.2a.

Method		Datasets									
		Aircraft	Cars	CIFAR-10	CIFAR-100	Caltech-101	Caltech-256	DTD	Flowers	Pets	Avg.
Robust ImageNet [9]		44.14	50.67	95.53	81.08	92.76	85.08	70.37	91.84	92.05	78.08
Standard ImageNet		38.69	44.63	81.31	60.14	90.12	82.78	70.09	**91.90**	**91.83**	72.32
MDN + ITT	$\mathcal{L}$:add	46.08	57.19	87.67	70.11	91.78	82.06	69.52	89.43	85.93	75.53
	$\mathcal{L}$:cat	47.40	56.74	87.80	70.87	92.61	82.84	**72.71**	90.8	86.52	76.48
	+Sty	**52.30**	**61.10**	**90.51**	**73.16**	**93.28**	**83.43**	71.86	91.07	88.38	**78.34**

TABLE 13
Multi-source DA (MST→TA) on DomainNet with ResNet-101. IN-O, IN-A, IN-P indicate ImNet-O, ImNet-A, ImNet-P respectively.

Method	$\mathbb{B}$	→C	→I	→P	→Q	→R	→S	Avg.
MDAN [11]	IN-O	52.4	21.3	46.9	8.6	54.9	46.5	38.4
DCTN [33]	IN-O	48.6	23.5	48.8	7.2	53.5	47.3	38.2
M^{3}SDA [32]	IN-O	58.6	26.0	52.3	6.3	62.7	49.5	42.6
MDDA [106]	IN-O	59.4	23.8	53.2	12.5	61.8	48.6	43.2
MIMFTL [92]	IN-O	67.2	25.0	54.4	13.4	67.0	54.1	46.8
WAMDA [94]	IN-O	59.3	21.8	52.1	9.5	65.0	47.7	42.6
LtC-MSDA [107]	IN-O	63.1	28.7	56.1	16.3	66.1	53.8	47.4
CMSS [108]	IN-O	64.2	28.0	53.6	16.0	63.4	53.8	46.5
SImpAl [97]	IN-O	66.4	26.5	56.6	18.9	68.0	55.5	48.6
PFSA [109]	IN-O	64.5	29.2	57.6	17.2	67.2	55.1	48.5
KD3A [110]	IN-O	**72.5**	23.4	60.9	16.4	72.7	60.6	51.1
DECISION [93]	IN-O	61.5	21.6	54.6	18.9	67.5	51.0	45.9
T-SVDNet [111]	IN-O	66.1	25.0	54.3	16.5	65.4	54.6	47.0
STEM [112]	IN-O	72.0	28.2	61.5	25.7	72.6	60.2	53.4
RADA [89]	IN-O	66.9	26.1	54.6	18.9	63.9	54.6	47.5
CMSDA [99]	IN-O	70.9	26.6	57.6	21.3	68.1	59.5	50.5
NEL [113]	IN-O	68.3	22.1	54.7	**22.8**	67.3	57.1	48.7
MUST [114]	IN-O	60.8	20.5	48.2	12.2	65.1	49.8	42.8
DRT [64]	IN-O	71.0	31.6	61.0	12.3	71.4	60.7	51.3
DRT [64]	IN-A	71.1	30.6	61.3	12.6	72.1	61.1	52.0
Ours (conf)	IN-A	67.3	22.4	55.3	12.4	66.6	57.8	47.0
(MST→TA) (mhu)	IN-A	71.2	32.4	60.5	20.1	73.4	61.5	53.2
DRT [64]	IN-P	71.3	32.6	61.3	12.9	71.5	61.0	52.2
Ours (conf)	IN-P	67.7	22.6	55.9	14.3	68.6	58.0	47.9
(MST→TA) (mhu)	IN-P	71.5	**33.1**	**61.7**	22.1	**73.0**	**62.1**	**53.9**

diversity, class granularity, size, *etc*. Different transfer strategies are characterized by the aggregation operator $\mathcal{L}$ (addition (add) or concatenation (cat)) and input stylization (w/oSty vs. w/Sty). ITT(w/oSty+$\mathcal{L}$:add) delivers similar performance to ITT(w/oSty+$\mathcal{L}$:cat) despite being computationally much cheaper. With the slightly higher computational cost of stylization, ITT(w/Sty+$\mathcal{L}$:cat) outperforms the other variants affirming the utility of domain-specific heads.

b) Extensibility to domain-shifts (DG). We report the DG results for DomainNet, PACS and Office-Home in Tables 6, 11 and 7 respectively. The evaluation scores for competing methods are taken from [70], [74]. The model trained with and without instance-level affinities are denoted as MST(w/ aff) and MST(w/o aff) respectively. MDN generalizes on new domains significantly better than the competing methods. The effects of learning good representations at vendor-side are visible on target domains like sketch, painting and clipart which benefit from MDN initialization from the broad spectrum of ImageNet variants. Table 11 shows our low variance across 3 random seeds, highlighting the statistical significance of the improvements.

c) Extensibility to domain-shifts (SSDA). Following [32], we report the *single best* accuracy in Table 10 and Table 8 for DomainNet and Office-31 respectively, using the source domain that results in the best post-adaptation accuracy. SST(w/oSty) and SST(w/Sty) denotes models before adaptation, with stylization as an ablation. SST→TA(conf) and SST→TA(mhu) denote post-adaptation models utilizing confidence (conf) and multi-head-unanimity (mhu). Before adaptation, our variants outperform the *Source-Only* baseline, which is a ResNet-101 model simply trained on the source domain. Incorporating stylization demonstrates the advantages of exploiting domain specificity. Post adaptation, using the unanimity criterion yields better results.

d) Extensibility to domain-shifts (MSDA). We adapt the DG trained models using the unanimity criterion (*i.e. MST→TA(mhu)*) and report the post-adaptation accuracy in Table 13 and 9. Interestingly, pre-adaptation performance of MDN (last row in Table 6) is comparable to the state-of-the-art MSDA methods and even outperforms the results for some domains. The performance is further improved post adaptation, outperforming all competing methods. The unanimity criterion performs better than confidence thresholding, similar to SSDA. Low variance (Table 9) across 3 random seeds highlights the statistical significance of our gains.

5 Conclusion

We present a deployable multi-domain paradigm to simultaneously address a wide range of inductive and transductive transfer learning problems. Our unique effort to encourage head-specificity enabled us to retain a superior balance between domain-generic and domain-specific representations. For future work, we plan to investigate the effectiveness of such frameworks for continual learning in presence of diverse task and domain shifts.

Acknowledgements. This work is supported by Ministry of Electronics and Information Technology, Govt. of India (Grant No. 4(16)2019-ITEA) and a Google PhD Fellowship (Jogendra).

References

[1] Z. Feng, C. Xu, and D. Tao, "Self-supervised representation learning from multi-domain data," in *ICCV*, 2019.
[2] Z. Ren and Y. Jae Lee, "Cross-domain self-supervised multi-task feature learning using synthetic imagery," in *CVPR*, 2018.
[3] H. Bilen and A. Vedaldi, "Universal representations: The missing link between faces, text, planktons, and cat breeds," *arXiv preprint arXiv:1701.07275*, 2017.
[4] A. S. Sebag, L. Heinrich, M. Schoenauer, M. Sebag, L. F. Wu, and S. Altschuler, "Multi-domain adversarial learning," in *ICLR*, 2019.
[5] M. Joshi, M. Dredze, W. W. Cohen, and C. Rosé, "What's in a domain? multi-domain learning for multi-attribute data," in *HLT-NAACL*, 2013.
[6] M. Joshi, M. Dredze, W. W. Cohen, and C. Rose, "Multi-domain learning: When do domains matter?" in *EMNLP-CoNLL*, 2012.
[7] M. Dredze, A. Kulesza, and K. Crammer, "Multi-domain learning by confidence-weighted parameter combination," *Machine Learning*, vol. 79, pp. 123–149, 2009.
[8] J. N. Kundu, N. Venkat, A. Revanur, R. M V, and R. V. Babu, "Towards inheritable models for open-set domain adaptation," in *CVPR*, 2020.
[9] H. Salman, A. Ilyas, L. Engstrom, A. Kapoor, and A. Madry, "Do adversarially robust imagenet models transfer better?" in *NeurIPS*, 2020.
[10] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Unsupervised domain adaptation with residual transfer networks," in *NeurIPS*, 2016.

[11] H. Zhao, S. Zhang, G. Wu, J. M. Moura, J. P. Costeira, and G. J. Gordon, "Adversarial multiple source domain adaptation," in *NeurIPS*, 2018.

[12] F. M. Carlucci, A. D'Innocente, S. Bucci, B. Caputo, and T. Tommasi, "Domain generalization by solving jigsaw puzzles," in *CVPR*, 2019.

[13] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," in *ICML*, 2014.

[14] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: an astounding baseline for recognition," in *CVPR Workshop*, 2014.

[15] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *BMVC*, 2014.

[16] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *CVPR*, 2014.

[17] S. Kornblith, J. Shlens, and Q. V. Le, "Do better imagenet models transfer better?" in *CVPR*, 2019.

[18] M. Huh, P. Agrawal, and A. A. Efros, "What makes imagenet good for transfer learning?" *arXiv preprint arXiv:1608.08614*, 2016.

[19] H. Azizpour, A. S. Razavian, J. Sullivan, A. Maki, and S. Carlsson, "Factors of transferability for a generic convnet representation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 9, pp. 1790–1802, 2015.

[20] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *NeurIPS*, 2014.

[21] M. Ghifary, D. Balduzzi, W. B. Kleijn, and M. Zhang, "Scatter component analysis: A unified framework for domain adaptation and domain generalization," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 7, pp. 1414–1430, 2016.

[22] H. Li, S. J. Pan, S. Wang, and A. C. Kot, "Domain generalization with adversarial feature learning," in *CVPR*, June 2018.

[23] S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto, "Unified deep supervised domain adaptation and generalization," in *ICCV*, 2017.

[24] S. Shankar, V. Piratla, S. Chakrabarti, S. Chaudhuri, P. Jyothi, and S. Sarawagi, "Generalizing across domains via cross-gradient training," in *ICLR*, 2018.

[25] R. Volpi, H. Namkoong, O. Sener, J. C. Duchi, V. Murino, and S. Savarese, "Generalizing to unseen domains via adversarial data augmentation," in *NeurIPS*, 2018.

[26] D. Li, J. Zhang, Y. Yang, C. Liu, Y.-Z. Song, and T. M. Hospedales, "Episodic training for domain generalization," in *ICCV*, 2019.

[27] M. Ghifary, W. Bastiaan Kleijn, M. Zhang, and D. Balduzzi, "Domain generalization for object recognition with multi-task autoencoders," in *ICCV*, 2015.

[28] A. Khosla, T. Zhou, T. Malisiewicz, A. A. Efros, and A. Torralba, "Undoing the damage of dataset bias," in *ECCV*, 2012.

[29] Y. Balaji, S. Sankaranarayanan, and R. Chellappa, "Metareg: Towards domain generalization using meta-regularization," in *NeurIPS*, 2018.

[30] Q. Dou, D. C. de Castro, K. Kamnitsas, and B. Glocker, "Domain generalization via model-agnostic learning of semantic features," in *NeurIPS*, 2019.

[31] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales, "Learning to generalize: Meta-learning for domain generalization," *AAAI*, 2018.

[32] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang, "Moment matching for multi-source domain adaptation," in *ICCV*, 2019.

[33] R. Xu, Z. Chen, W. Zuo, J. Yan, and L. Lin, "Deep cocktail network: Multi-source unsupervised domain adaptation with category shift," in *CVPR*, 2018.

[34] N. Dvornik, C. Schmid, and J. Mairal, "Selecting relevant features from a multi-domain representation for few-shot classification," in *ECCV*, 2020.

[35] L. Liu, W. Hamilton, G. Long, J. Jiang, and H. Larochelle, "A universal representation transformer layer for few-shot image classification," in *ICLR*, 2021.

[36] S.-A. Rebuffi, H. Bilen, and A. Vedaldi, "Learning multiple visual domains with residual adapters," in *NeurIPS*, 2017.

[37] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville, "Film: Visual reasoning with a general conditioning layer," in *AAAI*, 2018.

[38] S.-A. Rebuffi, H. Bilen, and A. Vedaldi, "Efficient parametrization of multi-domain deep neural networks," in *CVPR*, 2018.

[39] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015.

[40] V. Vapnik, "Principles of risk minimization for learning theory," in *NeurIPS*, 1992.

[41] H. Wang, S. Ge, Z. Lipton, and E. P. Xing, "Learning robust global representations by penalizing local predictive power," in *NeurIPS*, 2019.

[42] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009.

[43] D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo *et al.*, "The many faces of robustness: A critical analysis of out-of-distribution generalization," *ICCV*, 2021.

[44] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel, "Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness," in *ICLR*, 2019.

[45] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *ICCV*, 2017.

[46] X. Soria, E. Riba, and A. Sappa, "Dense extreme inception network: Towards a robust cnn model for edge detection," in *WACV*, 2020.

[47] A. B. Jung, K. Wada, J. Crall, S. Tanaka, J. Graving, C. Reinders, S. Yadav, J. Banerjee, G. Vecsei, A. Kraft, Z. Rui, J. Borovec, C. Vallentin, S. Zhydenko, K. Pfeiffer, B. Cook, I. Fernández, F.-M. De Rainville, C.-H. Weng, A. Ayala-Acevedo, R. Meudec, M. Laporte *et al.*, "imgaug," https://github.com/aleju/imgaug, 2020, online; accessed 01-Feb-2020.

[48] D.-H. Lee, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *ICML-W*, 2013.

[49] K. Saito, Y. Ushiku, and T. Harada, "Asymmetric tri-training for unsupervised domain adaptation," in *ICML*, 2017.

[50] Y. Li, L. Yuan, and N. Vasconcelos, "Bidirectional learning for domain adaptation of semantic segmentation," in *CVPR*, 2019.

[51] K. Mei, C. Zhu, J. Zou, and S. Zhang, "Instance adaptive self-training for unsupervised domain adaptation," in *ECCV*, 2020.

[52] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *ECCV*, 2010.

[53] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales, "Deeper, broader and artier domain generalization," in *ICCV*, 2017.

[54] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *CVPR*, 2017.

[55] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi, "Fine-grained visual classification of aircraft," Tech. Rep., 2013.

[56] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3d object representations for fine-grained categorization," in *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, 2013.

[57] A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.

[58] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," in *CVPR workshop*, 2004.

[59] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," *Caltech Technical Report*, 2007.

[60] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi, "Describing textures in the wild," in *CVPR*, 2014.

[61] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *ICVGIP*, 2008.

[62] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. Jawahar, "Cats and dogs," in *CVPR*, 2012.

[63] Q. Xu, R. Zhang, Y. Zhang, Y. Wang, and Q. Tian, "A fourier-based framework for domain generalization," in *CVPR*, 2021.

[64] Y. Li, L. Yuan, Y. Chen, P. Wang, and N. Vasconcelos, "Dynamic transfer for multi-source domain adaptation," in *CVPR*, 2021.

[65] W.-H. Li, X. Liu, and H. Bilen, "Universal representation learning from multiple domains for few-shot classification," in *ICCV*, 2021.

[66] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *CVPR*, 2017.

[67] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[68] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[69] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[70] P. Chattopadhyay, Y. Balaji, and J. Hoffman, "Learning to balance specificity and invariance for in and out of domain generalization," in *ECCV*, 2020.

[71] Y. Chen, Y. Wang, Y. Pan, T. Yao, X. Tian, and T. Mei, "A style and semantic memory mechanism for domain generalization," in *ICCV*, 2021.

[72] Z. Huang, H. Wang, E. P. Xing, and D. Huang, "Self-challenging improves cross-domain generalization," in *ECCV*, 2020.

[73] K. Zhou, Y. Yang, T. Hospedales, and T. Xiang, "Learning to generate novel domains for domain generalization," in *ECCV*, 2020.

[74] K. Zhou, Y. Yang, T. M. Hospedales, and T. Xiang, "Deep domain-adversarial image generation for domain generalisation." in *AAAI*, 2020.

[75] S. Seo, Y. Suh, D. Kim, G. Kim, J. Han, and B. Han, "Learning to optimize domain specific normalization for domain generalization," in *ECCV*, 2020.

[76] F.-E. Yang, Y.-C. Cheng, Z.-Y. Shiau, and Y.-C. F. Wang, "Adversarial teacher-student representation learning for domain generalization," in *NeurIPS*, 2021.

[77] K. Zhou, Y. Yang, Y. Qiao, and T. Xiang, "Domain generalization with mixstyle," in *ICLR*, 2021.

[78] G. Wu and S. Gong, "Collaborative optimization and aggregation for decentralized domain generalization and adaptation," in *ICCV*, 2021.

[79] G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann, "Contrastive adaptation network for unsupervised domain adaptation," in *CVPR*, 2019.

[80] Y. Wu, D. Inkpen, and A. El-Roby, "Dual mixup regularized learning for adversarial domain adaptation," in *ECCV*, 2020.

[81] L. Hu, M. Kan, S. Shan, and X. Chen, "Unsupervised domain adaptation with hierarchical gradient synchronization," in *CVPR*, 2020.

[82] Y. Hou and L. Zheng, "Visualizing adapted knowledge in domain transfer," in *CVPR*, 2021.

[83] X. Xu, X. Zhou, R. Venkatesan, G. Swaminathan, and O. Majumder, "d-sne: Domain adaptation using stochastic neighborhood embedding," in *CVPR*, 2019.

[84] H. Tang, K. Chen, and K. Jia, "Unsupervised domain adaptation via structurally regularized deep clustering," in *CVPR*, 2020.

[85] X. Gu, J. Sun, and Z. Xu, "Spherical space domain adaptation with robust pseudo-label loss," in *CVPR*, 2020.

[86] J. Liang, D. Hu, Y. Wang, R. He, and J. Feng, "Source data-absent unsupervised domain adaptation through hypothesis transfer and labeling transfer," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[87] J. Liang, D. Hu, and J. Feng, "Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation," in *ICML*, 2020.

[88] J. Huang, D. Guan, A. Xiao, and S. Lu, "Rda: Robust domain adaptation via fourier adversarial attacking," in *ICCV*, 2021.

[89] X. Jin, C. Lan, W. Zeng, and Z. Chen, "Re-energizing domain discriminator with sample relabeling for adversarial domain adaptation," in *ICCV*, 2021.

[90] M. Awais, F. Zhou, H. Xu, L. Hong, P. Luo, S.-H. Bae, and Z. Li, "Adversarial robustness for unsupervised domain adaptation," in *ICCV*, 2021.

[91] J. Na, H. Jung, H. J. Chang, and W. Hwang, "Fixbi: Bridging domain spaces for unsupervised domain adaptation," in *CVPR*, 2021.

[92] J. Gao, Y. Hua, G. Hu, C. Wang, and N. M. Robertson, "Reducing distributional uncertainty by mutual information maximisation and transferable feature learning," in *ECCV*, 2020.

[93] S. M. Ahmed, D. S. Raychaudhuri, S. Paul, S. Oymak, and A. Roy-Chowdhury, "Unsupervised multi-source domain adaptation without access to source data," in *CVPR*, 2021.

[94] S. Aggarwal, J. N. Kundu, V. B. Radhakrishnan, and A. Chakraborty, "WAMDA: weighted alignment of sources for multi-source domain adaptation," in *BMVC*, 2020.

[95] J. Dong, Z. Fang, A. Liu, G. Sun, and T. Liu, "Confident anchor-induced multi-source free domain adaptation," in *NeurIPS*, 2021.

[96] J. Wen, R. Greiner, and D. Schuurmans, "Domain aggregation networks for multi-source domain adaptation," in *ICML*, 2020.

[97] N. Venkat, J. N. Kundu, D. K. Singh, A. Revanur, and R. Venkatesh-Babu, "Your classifier can secretly suffice multi-source domain adaptation," in *NeurIPS*, 2020.

[98] G. Y. Park and S. W. Lee, "Information-theoretic regularization for multi-source domain adaptation," in *ICCV*, 2021.

[99] M. Scalbert, M. Vakalopoulou, F. Couzinié-Devy, and P. S. Centrale-Supélec, "Multi-source domain adaptation via supervised contrastive learning and confident consistency regularization," in *BMVC*, 2021.

[100] Z. Du, J. Li, H. Su, L. Zhu, and K. Lu, "Cross-domain gradient discrepancy minimization for unsupervised domain adaptation," in *CVPR*, 2021.

[101] X. Fan, Q. Wang, J. Ke, F. Yang, B. Gong, and M. Zhou, "Adversarially adaptive normalization for single domain generalization," in *CVPR*, 2021.

[102] Z. Wang, Y. Luo, R. Qiu, Z. Huang, and M. Baktashmotlagh, "Learning to diversify for single domain generalization," in *ICCV*, 2021.

[103] A. T. Nguyen, T. Tran, Y. Gal, and A. G. Baydin, "Domain invariant representation learning with domain density transformations," in *NeurIPS*, 2021.

[104] A. Robey, G. Pappas, and H. Hassani, "Model-based domain generalization," in *NeurIPS*, 2021.

[105] D. Kim, Y. Yoo, S. Park, J. Kim, and J. Lee, "Selfreg: Self-supervised contrastive regularization for domain generalization," in *ICCV*, 2021.

[106] S. Zhao, G. Wang, S. Zhang, Y. Gu, Y. Li, Z. Song, P. Xu, R. Hu, H. Chai, and K. Keutzer, "Multi-source distilling domain adaptation," in *AAAI*, 2020.

[107] H. Wang, M. Xu, B. Ni, and W. Zhang, "Learning to combine: Knowledge aggregation for multi-source domain adaptation," in *ECCV*, 2020.

[108] L. Yang, Y. Balaji, S.-N. Lim, and A. Shrivastava, "Curriculum manager for source selection in multi-source domain adaptation," in *ECCV*, 2020.

[109] Y. Fu, M. Zhang, X. Xu, Z. Cao, C. Ma, Y. Ji, K. Zuo, and H. Lu, "Partial feature selection and alignment for multi-source domain adaptation," in *CVPR*, 2021.

[110] H. Feng, Z. You, M. Chen, T.-Y. Zhang, M. Zhu, F. Wu, C. Wu, and W. Chen, "Kd3a: Unsupervised multi-source decentralized domain adaptation via knowledge distillation," in *ICML*, 2021.

[111] R. Li, X. Jia, J. He, S. Chen, and Q. Hu, "T-svdnet: Exploring high-order prototypical correlations for multi-source domain adaptation," in *ICCV*, 2021.

[112] V.-A. Nguyen, T. Nguyen, T. Le, Q. H. Tran, and D. Phung, "Stem: An approach to multi-source domain adaptation with guarantees," in *ICCV*, 2021.

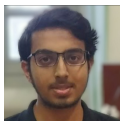
[113] W. Ahmed, P. Morerio, and V. Murino, "Cleaning noisy labels by negative ensemble learning for source-free unsupervised domain adaptation," in *WACV*, 2022.

[114] O. Amosy and G. Chechik, "Coupled training for multi-source domain adaptation," in *WACV*, 2022.

Jogendra Nath Kundu holds a Bachelor's from Biju Patnaik University of Technology. He is currently pursuing Ph.D. at the Indian Institute of Science, Bangalore, and is advised by Prof. R. Venkatesh Babu. His research interests include computer vision and machine learning.

Abhinav Agarwalla is a Masters student at Carnegie Mellon University. Before that, he was a project assistant at Video Analytics Lab, Indian Institute of Science, Bangalore. He received his undergraduate degree from IIT Kharagpur. His research interests includes computer vision and robotics.

Suvaansh Bhambri is a project assistant at Video Analytics Lab, Indian Institute of Science, Bangalore. He received his Bachelors in Electrical Engineering from Indian Institute of Technology, Roorkee. His research interests include machine learning and computer vision.



Akshay Kulkarni is a project assistant at Video Analytics Lab, Indian Institute of Science, Bangalore. He received his Bachelors from Visvesvaraya National Institute of Technology, Nagpur. His research interests include deep learning, computer vision and robotics.

Varun Jampani is a researcher at Google Research in Cambridge, US. Prior to that, he was a researcher at NVIDIA. He obtained his PhD at Max Planck Institute for Intelligent Systems (MPI) in Tübingen, Germany. His research involves content-adaptive neural networks and self-supervised visual discovery.

R. Venkatesh Babu received his Ph.D from Dept. of Electrical Engineering, IISc, Bangalore. Thereafter, he held postdoctoral positions at NTNU, Norway and IRISA/INRIA, France. Subsequently, he worked as a research fellow at NTU, Singapore. He is currently a Professor at Dept. of CDS and convener of VAL, IISc. His interests span vision, image/video processing, ML, and multimedia.