Learning Transferrable and Interpretable Representations for Domain Generalization

Paper ID: 1757*

ABSTRACT

Conventional machine learning models are often vulnerable to samples with different distributions from the ones of training samples, which is known as domain shift. Domain Generalization (DG) aims to challenge this issue by training a model based on multiple source domains and generalizing it to arbitrary unseen target domains. In spite of remarkable empirical results made in DG, a majority of existing works lack a deep understanding of the feature representations learned in DG models, resulting in limited generalization ability when facing domains *out-of-distribution.* In this paper, we aim to learn a domain transformation space via a domain transformer network (DTN) which explicitly mines the relationship among multiple domains and constructs transferrable feature representations for down-stream tasks by interpreting each feature as a weighted combination of multiple domain-specific features. Our DTN is encouraged to meta-learn the properties and characteristics of domains during the training process based on multiple seen domains, making transformed feature representations more interpretable thus generalize better to unseen domains. Once the domain transformer network is constructed, the feature representations of unseen target domains can also be inferred adaptively by selectively combining the feature representations from the diverse set of seen domains. We conduct extensive experiments on five DG benchmarks and the results strongly demonstrate the effectiveness of our approach.

CCS CONCEPTS

• **Computing methodologies** → **Computer vision**; *Learning paradigms*; *Image representations.*

KEYWORDS

image recognition; domain generalization; self-attention; interpretable feature

ACM Reference Format:
Paper ID: 1757. 2021. Learning Transferrable and Interpretable Representations for Domain Generalization. In *Proceedings of the 29th ACM International Conference on Multimedia (MM '21), October 20–24, 2021, Chengdu, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

A basic assumption behind multimedia analysis and interpretation is that the training data and the testing data are independently and identically distributed (i.i.d.). Unfortunately, this is not often the case in many real-world applications due to a variety of factors such as background, illumination and object position, etc., which is a.k.a. *domain shift* [33]. Humans often excel in tackling problems under different conditions (modalities) while machines struggle, making it a major barrier to unleashing the practical applicability of machine learning models. To tackle this problem, many transfer learning-related research topics have been proposed, including domain adaptation (DA) [51], domain generalization (DG) [50] and meta-learning [47], etc. Although the *domain shift* problem has been extensively studied in the DA community in recent years, it requires access to a certain number of target samples in advance for guidance to adapt the model to a specific target domain. In contrast, DG aims to generalize models trained on multiple source domains to arbitrary unseen target domains without any prior information, which is more human-like yet challenging.

Existing DG approaches can be divided into several categories according to their technical components. The primary branch of DG methods leverage the core idea from DA techniques to align the distributions of multiple source domains [2, 12, 25, 26, 31]. These methods aim to exploit domain-invariant features from seen domains, hoping that the invariant factors will also be applicable to unseen ones. Although this idea is reasonable, these methods are still prone to overfit to seen domains like other data-driven algorithms due to the absence of target data. As an alternative, meta-learning-based methods [4, 11, 23] have been proposed in the literature. These methods train domain-invariant models by first constructing meta-test (target) domains using held-out source domains, then explicitly encourage the model to learn to adapt to these fictitious target domains. However, these constructed target domains are still limited to observed domains. Therefore, the generalization ability on novel domains cannot be guaranteed [58], especially the ones with large domain discrepancies.

Recently, Gulrajani et al. [14] investigate how useful are previous DG algorithms in a consistent experimental condition, and they accidentally find that the vanilla Empirical Risk Minimization (ERM) outperforms most state-of-the-art methods. Based on this conclusion, we have to doubt: *is domain-invariance all we need?* Along with Liu *et al.* [28], we conjecture the reason for this phenomenon is that purely pursuing domain-invariant features through DNNs may lead to spurious invariant correlation due to dataset bias. In other words, we cannot guarantee the learned invariant features are causally interpretable, thus failing to describe the characteristics of novel domains. On the contrary, humans have an innate ability to learn interpretable factors of data that expose semantic meaning, which are more likely to be useful for tasks such as transfer learning and zero-shot learning [20]. Considering the example in

[*] Anonymous author

MM '21, October 20–24, 2021, Chengdu, China

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

Fig. 1, where we have observed samples from three domains, e.g., *painting*, *cartoon* and *photo*, then facing the instances from a novel domain, i.e., *sketch*. Intuitively, considering the similarity between the instances from *sketch* and those from *cartoon* will result in a better prediction than merely considering invariant features among all seen domains, because when people see something never seen before, they will naturally retrieve similar pieces in their memories. Therefore, we wonder if we can interpret a novel domain by those seen ones based on their similarities. In this paper, we will give a positive answer. Our idea is inspired from Transformer [45], which leverages a self-attention mechanism to mine the semantic relevance of words in a sentence under a certain task, then obtain the semantic value of a word by a weighted combination of other ones. We follow this idea and leverage it in the context of DG. More precisely, we first obtain a set of domain-specific features (multi-domain representation) for each sample by multiple domain-specific backbones that pre-trained on each source domain, this multi-domain representation captures the semantic information of the sample from the perspectives of all source domains. Based on this, we develop a domain transformer network (DTN) to meta-learn the semantic structure between the domain of this sample and multiple seen domains in a shared domain transformation space by a self-attention mechanism. The properties and characteristics of each domain are revealed in such a space, we can therefore construct a transferrable representation for each sample by a weighted combination of relevant domain-specific sub-features from the multi-domain representation (as illustrated in Fig. 1). Experimental results in section 5 indicate that our DTN is able to learn a transferrable and interpretable feature appropriate for semantic matching across domains.

In summary, the contributions of this paper can be listed as follows: (1) We propose to learn transferrable and interpretable feature representations for domain generalization, which makes a sample to be interpreted as a weighted combination of multiple domain-specific features based on their importances. Therefore, our method is expected to be applicable for samples from unseen target domains with large domain discrepancies. (2) To practice the idea, we leverage a domain transformer network that meta-learns the semantic structure across domains by self-attention. (3) To evaluate our approach, we conduct extensive experiments on five widely used DG benchmarks. Experimental results show that our approach is able to achieve a new state-of-the-art performance.

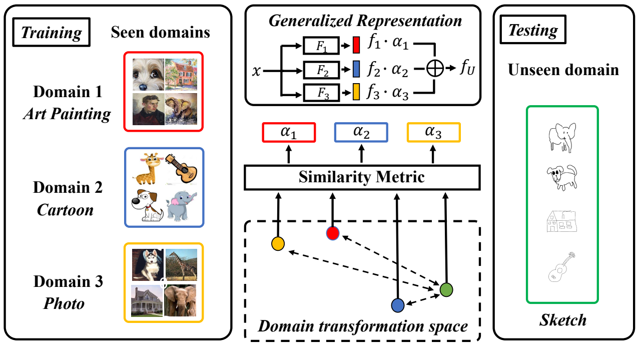


Figure 1: Illustration of our proposal. We learn a domain transformation space where each domain can be represented by a combination of other domains. Based on which the properties and characteristics of a novel target sample could also be analyzed by these known ones, thus enabling a more interpretable DG. We construct a transferrable representation f_U for each sample x based on the inter-domain relationship for subsequent classification.

2 RELATED WORK

Typical DG. The majority of existing domain generalization methods can be roughly divided into three categories: *domain alignment-based* [25–27], *meta-learning-based* [4, 9, 23] and *data augmentation-based* [6, 49, 58] methods. These methods pursue invariance from feature-level, model-level and data-level, respectively. Specifically, *domain alignment-based* methods are derived from DA and aim to learn domain-invariant feature representations by aligning feature distributions across multiple seen domains, which can be realized by either minimizing a well-defined distance metric (e.g., Maximum Mean Discrepancy [13] and Contrastive Domain Discrepancy [19]), or adversarial learning [12]. For instance, Li *et al.* [25] leverage Adversarial Autoencoder (AAE) [30] and MMD for distribution alignment between every pair of source domains. Ganin *et al.* [12] propose Domain Adversarial Neural Networks (DANN) to align distributions between domains in an adversarial fashion, this method can be easily extended to DG via performing every pair-wise training. Some other methods further consider aligning conditional distributions for more fine-grained category-level alignment [26, 27]. *Meta-learning-based* methods follow an alternative route to tackle domain shift: they simulate unseen domains by splitting the training data into meta-training sets and meta-test sets, by which the domain adaptation ability can be explicitly learned. As an early work, Li *et al.* [23] propose to leverage meta-learning in DG based on MAML [11]. Similarly, Dou *et al.* [9] adopt the MAML paradigm as well as two complementary regularizers to learn the semantic structure of the feature space. Later, MetaReg [4] is proposed to meta-learn a regularizer that boosts the domain generalization performance. However, these methods only use seen domains as fictitious target domains, thus still suffering from overfitting. *Data augmentation-based* methods can effectively address this problem by enlarging the support of seen distributions in different ways. Volpi *et al.* [49] extend source distributions by generating adversarial perturbations on source data within a certain Wasserstein distance in semantic space. Rahman *et al.* [37] and Zhou *et al.* [58] propose two different ways to augment training data using GANs. EISNet [52] learns how to generalize across domains by a metric learning task and a self-supervised task. Carlucci *et al.* [6] introduce an auxiliary that aims to solve jigsaw puzzles of randomly permuted image patches, by which the generalization ability of features can be promoted. Similarly, Representation Self-Challenging (RSC) [17] aims to learn a robust feature representation by iteratively dropping out dominant features. However, it is still difficult to ensure that these heuristic methods indeed bring generalization ability.

Recently, Gulrajani *et al.* [14] evaluate many previous DG methods in a consistent setting and release a DG testbed termed DomainBed. It is thought-provoking that simple ERM can achieve better or comparable results to these elaborate approaches. Our

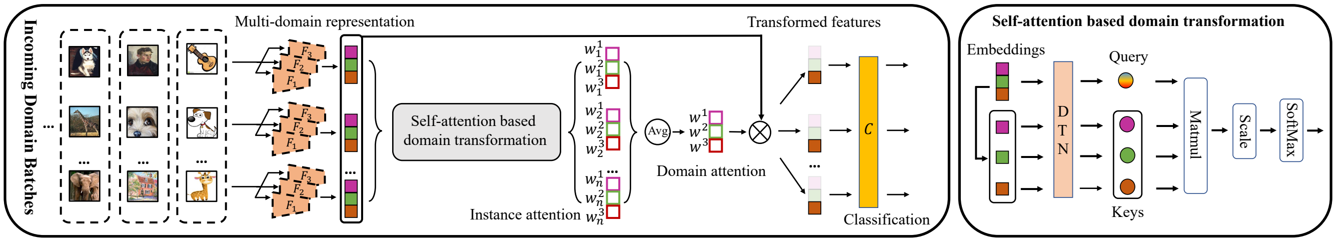


Figure 2: The pipeline of our proposed method. We refer to the example in Fig. 1 for a clear illustration, where we have 3 pre-trained feature extractors. Features in different colors denote different domain-specific sub-features in the multi-domain representation. Our DTN maps the multi-domain representation and domain-specific sub-features into a shared domain space, in which we perform the self-attention computation. We obtain the transferrable features of instances based on these scores and perform classification on the transformed features. Models in dash lines are frozen during training.

method can be regarded as a kind of meta-learning. However, different from the above ones that leverage the gradient signal as a regularizer, we explicitly pursue the interpretability of features, which is more promising to break out the shackle of seen domains.

Interpretable DG. There have been studies that criticize invariant feature-based approaches [18, 56]. Zhao *et al.* [56] claim that learning invariant representations can break the originally favorable underlying structure, i.e., leading to spurious invariant correlation. Recently, some studies propose to capture the causality in domain-invariant relations [3, 55], which are considered to provide interpretations in DG. Alternatively, some disentanglement-based DG methods [8, 36] decompose a feature to domain-invariant and domain-specific subsets. However, as pointed out by [50], interpretable domain generalization is still a challenge in future research. Similar to our work, methods in [7, 40, 41] argue that domain-specific factors also matter in DG. Specifically, [40] measures domain similarities in a transferrable domain space that is implicitly learned by domain-specific batch normalization layers. Different from them, we aim to learn such a domain space explicitly by interpreting one sample as a weighted combination of multiple domain-specific features.

Transformer is originally proposed for replacing RNN and CNN in Natural Language Processing (NLP) [45]. The core contribution of it is the self-attention mechanism that can be used to find the intra relationship between multiple components within an instance. Then it quickly gains popularity in other fields such as reinforcement learning [16, 48] and few-shot learning [29, 54], etc. Given its effectiveness, we want to establish a relationship across various domains by self-attention for DG.

3 PROPOSED METHOD

3.1 Problem Formulation

Let $\mathcal{X}$ and $\mathcal{Y}$ be the original input space and the output space (i.e, category number M) of a deep model, respectively. $\mathcal{D} = \{d_i\}_{i=1}^K$ denotes the set of K available source domains during training. Considering that all possible domains follow the same hyper-distribution $\mathcal{P}(X, Y)$ defined on $\mathcal{X} \times \mathcal{Y}$. Then each source domain d_i can be supported by a conditional distribution $P_{XY}^i = \mathcal{P}(X, Y|d_i)$. Our goal is to learn a predictor $f : \mathcal{X} \rightarrow \mathcal{Y}$ based on these seen distributions $\{P_{XY}^i\}_{i=1}^K$, with which we can predict the category $\hat{y}$ of a sample x by $\hat{y} = f(x)$. A common practice in DG is to decompose the predictor as $f = F \circ C$, where $F : \mathcal{X} \rightarrow \mathcal{F}$ is the feature extractor and $C : \mathcal{F} \rightarrow \mathcal{Y}$ is the classifier. In order to learn such a predictor, we construct a dataset $S^i = \{(x_j^i, y_j^i)\}_{j=1}^{n_i}$ for each domain d_i by identically and independently sampling n_i samples from P_{XY}^i. With these source datasets, the problem of DG is that how can we make the predictor able to perform well on target samples $\{(x_j^t, y_j^t)\}_{j=1}^{n_t}$ collected from $P_{XY}^t = \mathcal{P}(X, Y|d_t)$ during testing, where d_t can be any domain we never seen during training.

3.2 Domain Transformer Network

Although testing samples are collected from unseen domains, they still share some characteristics and properties with multiple source domains. For example, the samples from *cartoon* and those from *sketch* may be presented in different textures and colors, but they are all based on some simple shapes and lines. In this paper, we assume that these characteristics and properties are encoded in the discriminative feature space of each domain. The core idea of our method is to find a domain transformation space that associates all domains together, where we can measure the similarity between different domains. After that, the transferrable features can be constructed by adaptively retrieving and combining all relevant domain-specific features.

Pretrain multiple source feature extractors. We begin our method by exploiting discriminative information in each source domain. With K labeled source datasets, we pre-train K feature extractors $(F_1, F_2, \cdots, F_K)$ by the standard M-way cross-entropy loss. More precisely, for source domain i, we have

$$\mathcal{L}_{pre}\left(S^i; F_i, C_i\right) = -\frac{1}{n_i} \sum_{j=1}^{n_i} \sum_{m=1}^{M} \mathbb{1}\left[m = y_j^i\right] \log C_i\left(F_i\left(x_j^i\right)\right), \tag{1}$$

where C_i is the classifier for domain i. After this step, we can obtain multiple discriminative feature spaces of source domains which are encoded by these pre-trained feature extractors. However, these domain-specific feature spaces usually contain a large portion of information that is biased to each source domain, thus features obtained through these feature extractors are also domain-biased and not able to generalize to unseen target domains. It is worth noting that although we need to train multiple feature representation networks, the total number of domains are limited, often less than 5, in real-world applications.

Self-attention based domain transformer network. To solve the problem, we propose to retrieve the most relevant features from these domain-specific features while discarding those that are relatively less relevant, thus making the transformed features more semantically-siginificant and generalizable. For this purpose, we introduce our domain transformer network (DTN) that is encouraged to meta-learn the semantic structure across multiple domains. Let $h(x)$ be the multi-domain representation obtained by concatenating all domain-specific representations, i.e.,

$$h(\mathbf{x}) = \text{concatenate}\left(F_1'(\mathbf{x}), \ldots, F_K'(\mathbf{x})\right), \tag{2}$$

where $F_i'(\mathbf{x})$ simply denotes $F_i(\mathbf{x})$ after ℓ_2-normalization since features output by different extractors usually have different norms. Our DTN can be served as an attention function that maps a query and a set of key-value pairs to output scores. For each input x_j^i, we refer to domain-specific features $\{F_m'(x_j^i)\}_{m=1}^K$ as the value vectors of multiple source domains, then the transferrable representation is computed as a weighted sum of the values, where the weight for each value is determined by the query and the key of that value. Hence, the tricky part is how to design the query and the key. In this paper, we compute them by neural networks. Specifically, we obtain the query q_j^i for sample x_j^i by

$$q_j^i = D_q(h(x_j^i)), \tag{3}$$

where D_q representats a transformation function (network) parametered by θ_q. Similarly, we construct the corresponding key for each x_j^i with respect to each domain m by

$$k_j^{i(m)} = D_k(F_m'(x_j^i)), \tag{4}$$

with D_k also a transformation function (network) parametered by θ_k. Based on the query and keys, we then compute the instance-level attention score for x_j^i with respect to domain m using standard scaled dot-product attention

$$w_j^{i(m)} = \frac{\exp\left(z_j^{i(m)}\right)}{\sum_{k=1}^{K} \exp\left(z_j^{i(k)}\right)}, z_j^{i(m)} = \frac{(q_j^i)^\top k_j^{i(m)}}{\sqrt{d_k}}, \tag{5}$$

with d_k the dimensionality of keys and queries. Through above procedure, all domain-specific feature spaces are mapped into a transferrable domain space by D_k and D_q, where we obtain the domain relationship in the form of attention scores.

Training with domain batch. Since our purpose is to learn the relationship across domains, we adopt the domain batch based training policy to deal with the intra-domain variation, i.e., each training batch consists of K domain batches, with each domain batch constructed by randomly sampling B samples from that domain. Given a training batch from domain i, we obtain domain-level attention score by averaging these per-sample attention scores,

$$w^{i(m)} = \frac{\sum_{j=1}^{B} w_j^{i(m)}}{B}. \tag{6}$$

Based on $w^{i(m)}$, we construct the transferrable feature for each sample by

$$f(x_j^i) = \sum_{m=1}^{K} w^{i(m)} F_m'(x_j^i). \tag{7}$$

Algorithm 1 Domain Transformer Networks

```
Require: K pre-trained feature extractors $\{F_i\}_{i=1}^K$, trainable
    domain ransformer network $\{D_k^1, D_q^1, D_k^2, D_q^2, \cdots, D_k^H, D_q^H\}$,
    trainable classifier $C_U$, the number of training batches $\mathcal{T}$, pa-
    rameter $\lambda_{reg}$, domain batch size $B$, learning rate $\alpha$.
Ensure: Optimal $\{D_k^1, D_q^1, D_k^2, D_q^2, \cdots, D_k^H, D_q^H, C_U\}$
for t ← 1 to $\mathcal{T}$ do
    for domain ← 1 to $K$ do
        Obtain the multi-domain representation $h(x)$ for each
            instance $x$ using $K$ pre-trained extractors as in Eq. (2).
        Compute the self-attention scores of each instance accord-
            ing to Eq. (3), (4) and (5).
        Infer domain-level attention scores by averaging the self-
            attention scores of instances according to Eq. (6).
        Compute the transformed representations of instances by
            Eq. (7) and Eq. (8).
        Compute the overall training loss $\mathcal{L} = \mathcal{L}_{cls} + \lambda_{reg}\mathcal{L}_{reg}$
        Update the parameters of our DTN and the classifier $C_U$.
    end for
end for
```

This training strategy is much like the task sampling strategy in the meta-learning paradigm, where a domain batch takes the form of a learning task.

Multi-head attention. In addition to capturing attention from a single aspect, we wish the model to be able to capture attention from multiple patterns so as to learn more robust representations. We replicate the domain attention procedure described above H times, then concatenate these H obtained features together as our final transferrable feature for the down-stream classification task. Formally,

$$f_U(x_j^i) = \text{concatenate}\left(f_1(x_j^i), f_2(x_j^i), \cdots, f_H(x_j^i)\right), \tag{8}$$

where $f_h(x_j^i)$ is the single-head attention for the h-th head, which can be obtained by Eq. (7). To differentiate these heads, we use $(D_k^1, D_q^1), (D_k^2, D_q^2), \cdots, (D_k^H, D_q^H)$ to denote the network components for each head, with parameters $(\theta_k^1, \theta_q^1), (\theta_k^2, \theta_q^2), \cdots, (\theta_k^H, \theta_q^H)$, respectively. Then our DTN can be represented by these subnetworks. Furthermore, to avoid duplication, we add a penalty term to encourage each head to focus on a unique aspect. Let $A \in \mathrm{R}^{H \times K}$ be the attention matrix, where the h-th row of A is the attention scores of the h-th head. Then our regularization term can be defined by

$$\mathcal{L}_{reg} = \sum_{i,j=1}^{H} \Gamma_{ij} - \sum_{i=1}^{H} \Gamma_{ii} = \sum_{i \neq j}^{H} \Gamma_{ij}, \tag{9}$$

where $\Gamma = A^\top A \in \mathrm{R}^{H \times H}$. Obviously, this term will encourage the diversity of attention scores of different heads.

3.3 Overall Objective

Our DTN can be trained on top of any pre-trained feature extractors. We do not perform any fine-tuning on these feature extractors at training, thus the parameters that need to be learned are quite less than previous DG methods. In order to meta-learn the domain-level relationship better, we construct a domain batch by sampling instances from a specific domain, each training batch is composed of K domain batches with an equal number B of samples. For every batch with respect to domain i, we train our DTN based on the standard classification loss,

$$\mathcal{L}_{cls} = -\tfrac{1}{B} \textstyle\sum_{j=1}^{B} \sum_{c=1}^{M} \mathbb{1} \left[c = y_j^i\right] \log C_U \left(f_U(x_j^i)\right), \tag{10}$$

where C_U is the classifier works on transformed features parametered by θ_c. The overall training loss is shown as follows,

$$\mathcal{L} = \mathcal{L}_{cls} + \lambda_{reg}\mathcal{L}_{reg}. \tag{11}$$

with λ_{reg} the penalty coefficient. After training, we apply the same principle for target samples for testing. Although they are never seen before, it is hopeful to interpret the target domain by the source domains in the transformed domain space (where we perform the self-attention computation) since our DTN has learned the underlying semantic structure among domains. For a clear understanding, we depict the pipeline of our method in Fig. 2 and summary the optimization steps of the proposed method in Algorithm 1.

4 THEORETICAL ANALYSIS

In this section, we make a brief analysis of our proposed method by establishing the relationship between our method and the generalization bound of domain generalization. In this paper, we consider the covariate shift across different domains and measure the risk of a specific target domain. Covariate shift assumes that all possible domains follow a hyper-distribution $\mathcal{P}(X, Y)$ and share the same classifier, as we discussed in Section 3.1. With this assumption, Albuquerque *et al.* [1] consider approximating the target domain distribution P_{XY}^t in the convex hull Ω of source domain distributions, i.e., $\Omega := \left\{\sum_{i=1}^{K} \pi_i P_{X,Y}^i \mid \pi \in \Delta_K\right\}$, with Δ_K the $(K-1)$-dimensional simplex so that $\sum_{i=1}^{K} \pi_i = 1$. Let $R_S^1(h), R_S^2(h), \cdots, R_S^K(h)$ be the risks on multiple source domains with regards to hypothesis h, and $R_T(h)$ be the target risk. We denote the best approximated target distribution as $\overline{P}_{XY}^t$, which is the closest element in Ω approximated by $\pi^* = \text{argmin}_{\pi_1,\dots,\pi_K} d_{\mathcal{H}}\left(P_{XY}^t, \sum_{i=1}^{K} \pi_i P_{XY}^i\right)$, where $d_{\mathcal{H}}$ is the $\mathcal{H}$-divergence that widely used to measure the distribution divergence in domain adaptation. We then have the following derivation for the generalization bound of $R_T(h)$:

Theorem 1. *(Generalization to unseen domains [1]). Let* $\gamma := d_{\mathcal{H}}\left(\overline{P}_{XY}^t, P_{XY}^t\right)$ *with minimizer* π^* *be the distance of* P_{XY}^t *from the convex hull* Ω*, and* $\overline{P}_{XY}^t := \sum_{i=1}^{K} \pi_i^* P_{XY}^i$ *be the best approximator within* Ω*. Let* $\epsilon := \sup_{P'_X, P''_X \in \Lambda} d_{\mathcal{H}}\left(P'_{XY}, P''_{XY}\right)$ *be the diameter of* Ω*. Then it holds that*

$$R_T(h) \leq \sum_{i=1}^{K} \pi_i^* R_S^i(h) + \frac{\gamma + \epsilon}{2} + \lambda_{\mathcal{H},\left(P_{XY}^t, \overline{P}_{XY}^t\right)}$$

where $\lambda_{\mathcal{H},\left(P_{XY}^t, \overline{P}_{XY}^t\right)}$ *is the ideal joint risk of the target domain and the domain with the best approximator distribution* $\overline{P}_{XY}^t$ *.*

The proof can be found in [1]. Theorem 1 shows that the generalization ability of the model is upper-bounded by source risks and the approximation degree to the target distribution using source distributions. This directly motivates us to minimize source risks by the shared classifier C_U as well as explicitly learn to approximate a domain by a weighted combination of these seen domains via self-attention. In this format, the target domain is also hoped to be represented by these source domains, thus reducing γ and ϵ on the transformed feature space.

5 EXPERIMENTS

5.1 Data Preparation

We verify the performance of our DTN on five DG datasets: **Office-31** [38], **PACS** [24], **VLCS** [10], **OfficeHome** [46] and **DomainNet** [35]. Specifically, **Office-31** is a popular dataset widely adopted by domain adaptation methods. It is constructed by images from three distinct datasets: Amazon (A), Webcam (W) and DSLR (D). 31 classes are shared by these domains and 2817, 795, 498 images are involved in each domain, respectively. **PACS** is proposed, in particular, for domain generalization, which consists of 7 categories across 4 domains: Photo (P), Art painting (A), Cartoon (C) and Sketch(S). **VLCS** consists of images from 4 domains: VOC2007 (V), LabelMe (L), Caltech101 (C), and SUN09 (S). There are totally 10,729 samples in 5 categories. **OfficeHome** is a challenging dataset collected from 4 distinct domains: Artistic (Ar), Clipart (Cl), Product (Pr), and Real-World (Rw). A large number of categories (65) makes it more challenging for DG. **DomainNet** is the largest and hardest dataset so far, it has 586,575 examples and shares 345 categories among 6 domains, namely, Clipart (C), Infographic (I), Painting (P), Quickdraw (Q), Real (R) and Sketch (S).

For a fair comparison, we follow the dataset splitting and testing strategy adopt in [14]. More precisely, for all datasets, we choose one domain as the held-out target domain and use all other domains as source domains. For each domain (no matter source or target domains), we split the data into a training set (80%) and a validation set (20%), we use the larger split of the target domain for final evaluation. The data augmentation protocol is also derived from [14], which has been adopted by previous state-of-the-art DG work.

5.2 Implementation Details

Implementation platform. We choose PyTorch [34] framework to implement our method. One NIVIDIA GeForce RTX 2080 Ti GPU is employed as our hardware platform. We choose the DomainBed [14] as our test-bed for a fair comparison. The network architecture and detail experimental settings are detailed as follows.

Network architecture. We choose the ImageNet-pretrained ResNet-50 [15] as our feature extractor for all experiments. The output dimension is 2048. The classifier is a four-layer full-connect network with input dimension $d_c = 2048*H$, and hidden dimensions $1/2d_c$, $1/4d_c$, respectively. For our DTN, we use a single linear layer for $\{D_k^i\}_{i=1}^H$ and $\{D_q^i\}_{i=1}^H$. The dimensionality of keys and queries are set to 2048 in this paper since we empirically found it works well in this value. In addition, we freeze all batch normalization layers before training with the same reason in [14].

Implementation details for pre-training feature extractors. We use the same network architectures as we described above to pre-train feature extractors. For optimization, SGD optimizer is employed with learning rate $1e-2$ and weight decay $7e-4$ for all datasets. The batch size is set to 32 and the maximum number of the training iteration is set to 3,000 except 20,000 for DomainNet

Table 1: Accuracy(%) on Office-31 dataset for domain generalization (ResNet-50). Best in bold.

Algorithm	A	D	W	Avg
ERM [44]	57.2 ± 0.2	98.2 ± 0.3	94.0 ± 0.5	83.1
Mixup [53]	53.2 ± 0.7	**98.9 ± 0.8**	96.7 ± 0.2	82.9
GroupDRO [39]	57.3 ± 0.3	97.5 ± 0.4	92.1 ± 0.1	82.3
MLDG [23]	54.7 ± 0.5	**98.9 ± 0.9**	95.4 ± 0.2	83.0
MMD [25]	55.9 ± 0.8	97.2 ± 0.2	**96.8 ± 0.3**	83.3
MTL [5]	54.9 ± 0.3	98.5 ± 0.4	94.5 ± 0.7	82.7
SagNet [32]	59.2 ± 0.8	97.0 ± 0.4	93.8 ± 0.3	83.3
ARM [55]	48.4 ± 0.6	98.2 ± 0.5	92.7 ± 0.1	79.8
VREx [22]	57.9 ± 1.2	98.1 ± 0.2	94.9 ± 0.5	83.6
RSC [17]	45.2 ± 0.6	97.9 ± 0.9	94.4 ± 0.2	79.2
DTN (Ours)	**59.4 ± 0.3**	98.0 ± 0.6	94.8 ± 0.3	**84.1**

Table 2: Accuracy(%) on PACS dataset for domain generalization (ResNet-50). Methods marked by † mean that the results are collected from original papers. Best in bold.

Algorithm	A	C	P	S	Avg
ERM [44]	84.7 ± 0.4	80.8 ± 0.6	97.2 ± 0.3	79.3 ± 1.0	85.5
MLDG [23]	85.5 ± 1.4	80.1 ± 1.7	97.4 ± 0.3	76.6 ± 1.1	84.9
MMD [25]	86.1 ± 1.4	79.4 ± 0.9	96.6 ± 0.2	76.5 ± 0.5	84.6
DANN [12]	86.4 ± 0.8	77.4 ± 0.8	97.3 ± 0.4	73.5 ± 2.3	83.6
SagNet [32]	87.4 ± 1.0	80.7 ± 0.6	97.1 ± 0.1	80.0 ± 0.4	86.3
ARM [55]	86.8 ± 0.6	76.8 ± 0.5	97.4 ± 0.3	79.3 ± 1.2	85.1
VREx [22]	86.0 ± 1.6	79.1 ± 0.6	96.9 ± 0.5	77.7 ± 1.7	84.9
RSC [17]	85.4 ± 0.8	79.7 ± 1.8	97.6 ± 0.3	78.2 ± 1.2	85.2
EisNet † [52]	86.6 ± 1.4	81.5 ± 0.6	97.1 ± 0.4	78.1 ± 1.4	85.8
ER † [57]	87.5 ± 1.0	79.3 ± 1.4	98.3 ± 0.1	76.3 ± 0.7	85.3
DMG † [7]	82.6	78.1	94.5	78.3	83.4
DSON † [41]	87.0	80.6	96.0	**82.9**	86.6
MASF † [9]	82.9	80.5	95.0	72.3	82.7
MetaReg † [4]	87.2	79.2	97.6	70.3	83.6
DTN (Ours)	**88.5 ± 0.2**	**82.7 ± 0.1**	**98.3 ± 0.4**	81.5 ± 0.7	**87.8**

since it is quite large and therefore hard to converge. We divide the learning rate by 0.1 after 80% epochs. Finally, we choose the model with the highest accuracy on validation set for subsequent training.

Implementation details for training DTN. We use Adam optimizer [21] to train our DTN and classifier, the initial learning rates are set to $1e-2$ and $5e-5$, respectively. We adopt the same learning rate scheduling strategy as described above, and we do not use weight decay in experiments. The number of attention heads is set to 4. The domain batch size is set to 32 and the maximum number of training iterations is 5,000 expect 10,000 for DomainNet. We set $\lambda_{reg} = 0.1$ for all experiments. During training, we test our model every 300 iterations except 1000 iterations for DomainNet. For model selection, we use the *training-domain validation* strategy, i.e., we train models using the source training sets, and choose the model maximizing the accuracy on the union of source validation sets for testing. We repeated every experiment three times and report the mean accuracy as well as the standard deviation.

5.3 Comparison with Other Methods

We compare our method with previous methods that have considerable impact in DG in recent years. For a fair comparison, our experiments are conducted on DomainBed under the same experimental settings as [14]. The results of Empirical Risk Minimization (ERM, [44]), Group Distributionally Robust Optimization (GroupDRO, [39]), Inter-domain Mixup (Mixup, [53]), Meta-Learning for Domain Generalization (MLDG, [23]), DomainAdversarial Neural Networks (DANN, [12]), Class-conditional DANN (C-DANN, [27]), Deep CORrelation ALignment (CORAL, [42]), Maximum Mean Discrepancy (MMD, [25]), Invariant Risk Minimization (IRM, [3]), Adaptive Risk Minimization (ARM, [55]), Marginal Transfer Learning (MTL, [5]), Style-Agnostic Networks (SagNet, [32]), and Representation Self Challenging [17]) are cited from [14] except for **Office-31**, on which we conduct experiments and report the results reproduced by us. For **PACS**, we also compare our method with EisNet [52], Entropy Regularization (ER) [57], DMG [7], DSON [41], MASF [9] and MetaReg [4] which also use ResNet-50 as the backbone, the results of these methods are directly cited from original papers. It is worth noting that from the results reported in previous DG research over the years, one can easily find that even a marginal improvement (e.g., 1%) on average accuracy is challenging in the DG community.

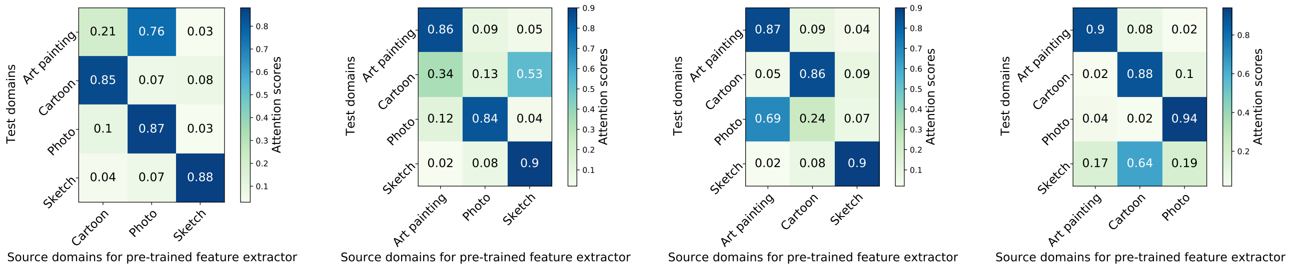
Table 3: Accuracy(%) on VLCS dataset for domain generalization (ResNet-50). Best in bold.

Algorithm	C	L	S	V	Avg
ERM [44]	97.7 ± 0.4	64.3 ± 0.9	73.4 ± 0.5	74.6 ± 1.3	77.5
IRM [3]	98.6 ± 0.1	64.9 ± 0.9	73.4 ± 0.6	77.3 ± 0.9	78.5
MLDG [23]	97.4 ± 0.2	65.2 ± 0.7	71.0 ± 1.4	75.3 ± 1.0	77.2
CORAL [42]	98.3 ± 0.1	**66.1 ± 1.2**	73.4 ± 0.3	77.5 ± 1.2	78.8
MMD [25]	97.7 ± 0.1	64.0 ± 1.1	72.8 ± 0.2	75.3 ± 3.3	77.5
DANN [12]	99.0 ± 0.3	65.1 ± 1.4	73.1 ± 0.3	77.2 ± 0.6	78.6
CDANN [27]	97.1 ± 0.3	65.1 ± 1.2	70.7 ± 0.8	77.1 ± 1.5	77.5
SagNet [32]	97.9 ± 0.4	64.5 ± 0.5	71.4 ± 1.3	77.5 ± 0.5	77.8
VREx [22]	98.4 ± 0.3	64.4 ± 1.4	74.1 ± 0.4	76.2 ± 1.3	78.3
RSC [17]	97.9 ± 0.1	62.5 ± 0.7	72.3 ± 1.2	75.6 ± 0.8	77.1
DTN (Ours)	**98.7 ± 0.1**	64.6 ± 0.2	**75.5 ± 0.4**	**78.0 ± 0.3**	**79.2**

Table 4: Accuracy(%) on OfficeHome dataset for domain generalization (ResNet-50). Best in bold.

Algorithm	Ar	Cl	Pr	Rw	Avg
ERM [44]	61.3 ± 0.7	52.4 ± 0.3	75.8 ± 0.1	76.6 ± 0.3	66.5
IRM [3]	58.9 ± 2.3	52.2 ± 1.6	72.1 ± 2.9	74.0 ± 2.5	64.3
GroupDRO [39]	60.4 ± 0.7	52.7 ± 1.0	75.0 ± 0.7	76.0 ± 0.7	66.0
MLDG [23]	61.5 ± 0.9	53.2 ± 0.6	75.0 ± 1.2	77.5 ± 0.4	66.8
MMD [25]	60.4 ± 0.2	53.3 ± 0.3	74.3 ± 0.1	77.4 ± 0.6	66.3
DANN [12]	59.9 ± 1.3	53.0 ± 0.3	73.6 ± 0.7	76.9 ± 0.5	65.9
CDANN [27]	61.5 ± 1.4	50.4 ± 2.4	74.4 ± 0.9	76.6 ± 0.8	65.8
MTL [5]	61.5 ± 0.7	52.4 ± 0.6	74.9 ± 0.4	76.8 ± 0.4	66.4
ARM [55]	58.9 ± 0.8	51.0 ± 0.5	74.1 ± 0.1	75.2 ± 0.3	64.8
VREx [22]	60.7 ± 0.9	53.0 ± 0.9	75.3 ± 0.1	76.6 ± 0.5	66.4
RSC [17]	60.7 ± 1.4	51.4 ± 0.3	74.8 ± 1.1	75.1 ± 1.3	65.5
DTN (Ours)	**62.1 ± 0.3**	**54.2 ± 0.1**	**76.8 ± 0.2**	**78.4 ± 0.5**	**67.9**

The quantitative results of five datasets are reported in Table 1, 2, 3, 4, 5, respectively. It is obvious that our method achieves results surpassing or comparable to previous state-of-the-arts for all datasets. Specifically, our DTN boosts the strong baseline ERM and those domain-invariant feature based methods by a clear margin

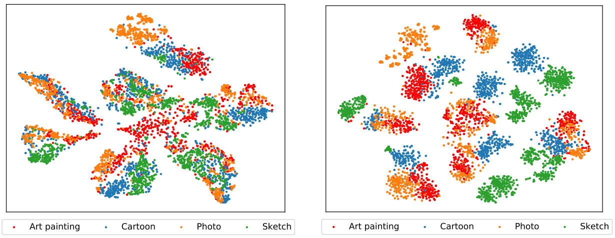


(a) Art painting (b) Cartoon (c) Photo (d) Sketch

Figure 3: The averaged attention scores obtained by the first head of our DTN when the target domain is (a) Art painting, (b) Cartoon, (c) Photo and (d) Sketch. The attention scores of seen domains are got by using the corresponding validation set.

Table 5: Accuracy(%) on DomainNet dataset for domain generalization (ResNet-50). Best in bold.

Algorithm	clip	info	paint	quick	real	sketch	Avg
ERM [44]	58.1 ± 0.3	18.8 ± 0.3	46.7 ± 0.3	12.2 ± 0.4	59.6 ± 0.1	49.8 ± 0.4	40.9
IRM [3]	48.5 ± 2.8	15.0 ± 1.5	38.3 ± 4.3	10.9 ± 0.5	48.2 ± 5.2	42.3 ± 3.1	33.9
Mixup [53]	55.7 ± 0.3	18.5 ± 0.5	44.3 ± 0.5	12.5 ± 0.4	55.8 ± 0.3	48.2 ± 0.5	39.2
MLDG [23]	59.1 ± 0.2	19.1 ± 0.3	45.8 ± 0.7	13.4 ± 0.3	59.6 ± 0.2	50.2 ± 0.4	41.2
CORAL [42]	59.2 ± 0.1	**19.7 ± 0.2**	46.6 ± 0.3	13.4 ± 0.4	59.8 ± 0.2	50.1 ± 0.6	41.5
DANN [12]	53.1 ± 0.2	18.3 ± 0.1	44.2 ± 0.7	11.8 ± 0.1	55.5 ± 0.4	46.8 ± 0.6	38.3
CDANN [27]	54.6 ± 0.4	17.3 ± 0.1	43.7 ± 0.9	12.1 ± 0.7	56.2 ± 0.4	45.9 ± 0.5	38.3
MTL [5]	57.9 ± 0.5	18.5 ± 0.4	46.0 ± 0.1	12.5 ± 0.1	59.5 ± 0.3	49.2 ± 0.1	40.6
RSC [17]	55.0 ± 1.2	18.3 ± 0.5	44.4 ± 0.6	12.2 ± 0.2	55.7 ± 0.7	47.8 ± 0.9	38.9
DTN (Ours)	**62.0 ± 0.2**	19.6 ± 0.2	**50.3 ± 0.2**	**13.7 ± 0.3**	**60.4 ± 0.4**	**50.9 ± 0.5**	**42.8**



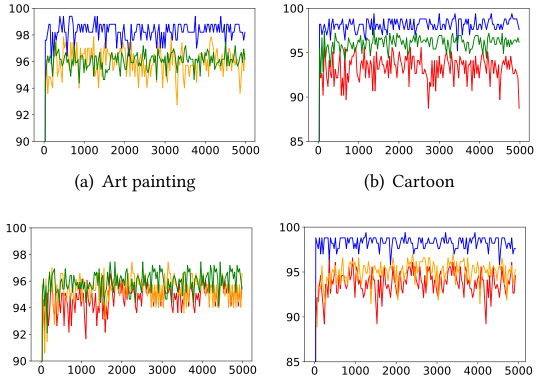
(a) ERM (b) Ours

Figure 4: Feature visualization on PACS by t-SNE [43]. We choose *Art painting* as the unseen target domain. Different colors represent different domains. (a) Features learned by simple ERM algorithm. (b) Features learned by our proposed method. Best viewed in color.

(more than 1%) with respect to the mean accuracy for all datasets. For the small dataset such as **Office-31**, our superiority is relatively marginal, we conjecture the reason is that there are only two source domains in **Office-31**, lacking domain diversity makes it hard to express the target domain through the source ones. On the contrary, for large-scale datasets such as **PACS** and **DomainNet**, our method obtains large improvements over the competitors, outperforming the second-best ones by 1.2% (DSON) and 1.6% (MLDG) with respect to the mean accuracy, respectively. In Table 5, it is desirable that our method can achieve significant improvements on some difficult categories in **DomainNet** such as *Clipart* and *Painting*, with +2.8% and +3.6% improvements compared with the second-best methods. This justifies that learning with diverse feature extractors facilitates the accuracy of our method. For **VLCS** and **OfficeHome**, our DTN also achieves comparable results and achieves best mean accuracies with 79.2% and 67.9%, respectively. These quantitative results strongly demonstrate the effectiveness of our method for domain generalization.

5.4 Further Analysis

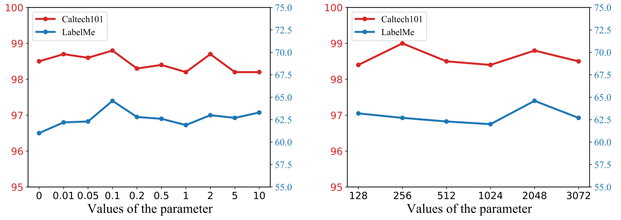
Interpreting attention scores. To further analyze whether our method really learns interpretable attention, we visualized the attention scores output by our DTN when tested on **PACS**. We empirically found that all domains share the same attention scores in head 2,3,4 and vary in head 1. Thus we choose to report the attention of head 1 in Fig. 3. The reported attention score of each test domain is the global average over all validation samples in that domain. We can see that the validation samples from these seen domains are prone to put most of the attention (about 85% 90%) on the feature extractor pre-trained on that domain. For the unseen domain, it shows a relatively smooth attention score among the seen domains which are related to that domain. For instance, Fig. 3(b) shows the samples from *Cartoon* tend to blend the representations from *Art painting* and *Sketch*, we conjecture that the rich colors in *Art painting* and the line style in *Sketch* make up the style of *Cartoon*. Similarly, Fig. 3(a) indicates that *Art painting* is like a combination



(a) Art painting (b) Cartoon

(c) Photo (d) Sketch

Figure 5: The validation accuracies for all domains during training. We conduct experiments on PACS dataset and report results when the target domain is (a) Art painting (red), (b) Cartoon (orange), (c) Photo (blue) and (d) Sketch (green). The X-axis and Y-axis represent the training iteration and accuracy, respectively. Best viewed in color.



(a) λ_{reg} (b) d_k

Figure 6: The accuracies of the proposed method on VLCS dataset with different values of λ_{reg} and d_k. We take tasks C and L as examples. Best viewed in color.

of *Photo* and *Cartoon*. These attention scores encode semantics that can be understood by humans, indicating that our DTN is able to mine the semantical structure among diverse domains.

Feature visualization. To vividly illustrate why our method works in DG, we visualize the feature representations of vanilla ERM and our method in Fig. 4. Comparing Fig. 4 (a) and Fig. 4 (b), we can observe that simple ERM is prone to align the distributions of all seen domains since it regards them as a whole. However, a large portion of the target samples (i.e., *Art painting*) will fall outside the support of source distributions (in the center of Fig. 4 (a)). From Fig. 4 (b), it is easy to observe that after the transformation by our DTN, the features of *Art painting* will fall around the distributions of related domains such as *Photo* and *Cartoon*, which echos our analysis in attention scores. These experimental results show that our proposed method is effective in learning transferrable and interpretable features for domain generalization.

Training stability. Fig. 5 reports the validation accuracies of the seen domains over training iterations. We evaluate all tasks on **PACS** dataset. It can be easily found that the accuracies of all validation tasks follow a very similar trend, i.e., quickly growing up and then converging after about 500 steps. Generally, the training of our method is smooth and stable.

Table 6: Mean accuracy on VLCS dataset using different number of attention heads. Best in bold.

Heads	1	2	3	4	5	6	7	8
C	98.5	99.2	**98.9**	98.8	98.6	98.2	97.8	97.8
L	63.2	**64.6**	**64.6**	**64.6**	63.3	63.4	62.5	61.0
S	74.3	74.9	73.3	75.5	74.2	**76.3**	74.2	75.7
V	75.2	76.1	72.5	78.0	78.2	**78.4**	77.5	77.4
Avg.	77.8	78.7	77.3	**79.2**	78.6	79.0	78.0	77.9

Table 7: Ablation study on VLCS dataset. The w/o is short for without. Best in bold.

Method	C	L	V	S	Avg
w/o DTN	**99.0**	59.9	73.7	69.8	75.6
w/o $\mathcal{L}_{reg}$	98.5	61.0	74.3	77.9	77.9
Full Model	98.8	**64.6**	**75.5**	**78.0**	**79.2**

Parameter sensitivity. Our method involves three hyperparameters, i.e., λ_{reg} which controls the contribution of the regularization loss $\mathcal{L}_{reg}$, the dimensionality of keys and queries d_k, and the number of attention heads H. We investigate the sensitivity of these parameters to better understand the impact of them. Specifically, we evaluate each parameter by fixing others as the default value in Section 5.2. Fig. 6(a) shows that with the increase of λ_{reg}, the accuracies of tasks first grow up and then tend to be stable, it reaches the maximum when $\lambda_{reg} = 0.1$. It is obvious that our regularization term indeed benefits the training if we compare the performances when $\lambda_{reg} = 0$ and $\lambda_{reg} = 0.1$, respectively. For d_k, we run a grid-search in $\{128, 256, 512, 1024, 2048, 3072\}$. Fig. 6(b) shows the accuracy curve does not present a significant fluctuation when d_k varies. It demonstrates that our method can perform well in a wide range of d_k. To investigate the effect of the number of attention heads, we turned H from 1 to 8 on **VLCS** and report the mean accuracy over all tasks in Table. 6. The accuracy first grows up when $H < 4$ then begins to decline slightly when $H > 4$. Generally, our method is not sensitive to these hyperparameters.

Ablation study. We investigate the effectiveness of our DTN and $\mathcal{L}_{reg}$ by removing them respectively. By w/o DTN we mean the attention scores obtained by DTN are not applied in the current implementation, i.e., all source domains have equal contributions. It is worth noting that our $\mathcal{L}_{reg}$ relies on the DTN, it only can be removed when DTN works. From Fig. 7, we can conclude that our DTN significantly benefits the accuracy of DG, and the regularization term $\mathcal{L}_{reg}$ further facilitates it.

6 CONCLUSION

We present an effective method to combine feature representations from multiple domains for domain generalization. Our method aims to meta-learn the semantic relationship across domains, so as to adaptively select the relevant features from the diverse set of seen domains. In this way, we hope to interpret the features of unseen target domains as a weighted combination of those seen domain-specific features. The learning strategy is based on the self-attention mechanism that proposed in Transformer. Extensive experiments on five widely used domain generalization datasets demonstrate the advantage of our method.

REFERENCES

[1] Isabela Albuquerque, João Monteiro, Mohammad Darvishi, Tiago H Falk, and Ioannis Mitliagkas. 2019. Generalizing to unseen domains via distribution matching. *arXiv preprint arXiv:1911.00804* (2019).

[2] Isabela Albuquerque, João Monteiro, Tiago H Falk, and Ioannis Mitliagkas. 2019. Adversarial target-invariant representation learning for domain generalization. *arXiv preprint arXiv:1911.00804* (2019).

[3] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. 2019. Invariant risk minimization. *arXiv preprint arXiv:1907.02893* (2019).

[4] Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. 2018. Metareg: Towards domain generalization using meta-regularization. *Advances in Neural Information Processing Systems* 31 (2018), 998–1008.

[5] Gilles Blanchard, Gyemin Lee, and Clayton Scott. 2011. Generalizing from several related classification tasks to a new unlabeled sample. *Advances in neural information processing systems* 24 (2011), 2178–2186.

[6] Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. 2019. Domain generalization by solving jigsaw puzzles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 2229–2238.

[7] Prithvijit Chattopadhyay, Yogesh Balaji, and Judy Hoffman. 2020. Learning to balance specificity and invariance for in and out of domain generalization. In *European Conference on Computer Vision.* Springer, 301–318.

[8] Aniket Anand Deshmukh, Ankit Bansal, and Akash Rastogi. 2018. Domain2vec: Deep domain generalization. *arXiv preprint arXiv:1807.02919* (2018).

[9] Qi Dou, Daniel C Castro, Konstantinos Kamnitsas, and Ben Glocker. 2019. Domain generalization via model-agnostic learning of semantic features. *arXiv preprint arXiv:1910.13580* (2019).

[10] Chen Fang, Ye Xu, and Daniel N Rockmore. 2013. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *Proceedings of the IEEE International Conference on Computer Vision.* 1657–1664.

[11] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning.* PMLR, 1126–1135.

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

[13] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. 2012. A kernel two-sample test. *The Journal of Machine Learning Research* 13, 1 (2012), 723–773.

[14] Ishaan Gulrajani and David Lopez-Paz. 2020. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434* (2020).

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* 770–778.

[16] Siyi Hu, Fengda Zhu, Xiaojun Chang, and Xiaodan Liang. 2021. Updet: Universal multi-agent reinforcement learning via policy decoupling with transformers. *arXiv preprint arXiv:2101.08001* (2021).

[17] Zeyi Huang, Haohan Wang, Eric P Xing, and Dong Huang. 2020. Self-challenging improves cross-domain generalization. *arXiv preprint arXiv:2007.02454* 2 (2020).

[18] Fredrik D Johansson, David Sontag, and Rajesh Ranganath. 2019. Support and invertibility in domain-invariant representations. In *The 22nd International Conference on Artificial Intelligence and Statistics.* PMLR, 527–536.

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

[20] Hyunjik Kim and Andriy Mnih. 2018. Disentangling by factorising. In *International Conference on Machine Learning.* PMLR, 2649–2658.

[21] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[22] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. 2020. Out-of-distribution generalization via risk extrapolation (rex). *arXiv preprint arXiv:2003.00688* (2020).

[23] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. 2018. Learning to generalize: Meta-learning for domain generalization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[24] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. 2017. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision.* 5542–5550.

[25] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. 2018. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 5400–5409.

[26] Ya Li, Mingming Gong, Xinmei Tian, Tongliang Liu, and Dacheng Tao. 2018. Domain generalization via conditional invariant representations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[27] Ya Li, Xinmei Tian, Mingming Gong, Yajing Liu, Tongliang Liu, Kun Zhang, and Dacheng Tao. 2018. Deep domain generalization via conditional invariant adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV).* 624–639.

[28] Chang Liu, Xinwei Sun, Jindong Wang, Tao Li, Tao Qin, Wei Chen, and Tie-Yan Liu. 2020. Learning Causal Semantic Representation for Out-of-Distribution Prediction. *arXiv preprint arXiv:2011.01681* (2020).

[29] Lu Liu, William Hamilton, Guodong Long, Jing Jiang, and Hugo Larochelle. 2020. A universal representation transformer layer for few-shot image classification. *arXiv preprint arXiv:2006.11702* (2020).

[30] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. 2015. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644* (2015).

[31] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. 2013. Domain generalization via invariant feature representation. In *International Conference on Machine Learning.* PMLR, 10–18.

[32] Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. 2019. Reducing domain gap via style-agnostic networks. *arXiv preprint arXiv:1910.11645* (2019).

[33] Sinno Jialin Pan and Qiang Yang. 2009. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22, 10 (2009), 1345–1359.

[34] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703* (2019).

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

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

[37] Mohammad Mahfujur Rahman, Clinton Fookes, Mahsa Baktashmotlagh, and Sridha Sridharan. 2019. Multi-component image translation for deep domain generalization. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV).* IEEE, 579–588.

[38] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. 2010. Adapting visual category models to new domains. In *European conference on computer vision.* Springer, 213–226.

[39] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. 2019. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731* (2019).

[40] Mattia Segù, Alessio Tonioni, and Federico Tombari. 2020. Batch Normalization Embeddings for Deep Domain Generalization. *arXiv preprint arXiv:2011.12672* (2020).

[41] Seonguk Seo, Yumin Suh, Dongwan Kim, Jongwoo Han, and Bohyung Han. 2019. Learning to optimize domain specific normalization for domain generalization. *arXiv preprint arXiv:1907.04275* 3, 6 (2019), 7.

[42] Baochen Sun and Kate Saenko. 2016. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision.* Springer, 443–450.

[43] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (2008).

[44] Vladimir Vapnik and Vlamimir Vapnik. 1998. Statistical learning theory Wiley. *New York* 1, 624 (1998), 2.

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762* (2017).

[46] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. 2017. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* 5018–5027.

[47] Ricardo Vilalta and Youssef Drissi. 2002. A perspective view and survey of meta-learning. *Artificial intelligence review* 18, 2 (2002), 77–95.

[48] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature* 575, 7782 (2019), 350–354.

[49] Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John Duchi, Vittorio Murino, and Silvio Savarese. 2018. Generalizing to unseen domains via adversarial data augmentation. *arXiv preprint arXiv:1805.12018* (2018).

[50] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, and Tao Qin. 2021. Generalizing to Unseen Domains: A Survey on Domain Generalization. *arXiv preprint arXiv:2103.03097* (2021).

[51] Mei Wang and Weihong Deng. 2018. Deep visual domain adaptation: A survey. *Neurocomputing* 312 (2018), 135–153.

[52] Shujun Wang, Lequan Yu, Caizi Li, Chi-Wing Fu, and Pheng-Ann Heng. 2020. Learning from Extrinsic and Intrinsic Supervisions for Domain Generalization. In *European Conference on Computer Vision.* Springer, 159–176.

[53] Minghao Xu, Jian Zhang, Bingbing Ni, Teng Li, Chengjie Wang, Qi Tian, and Wenjun Zhang. 2020. Adversarial domain adaptation with domain mixup. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 6502–6509.

[54] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. 2020. Few-shot learning via embedding adaptation with set-to-set functions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8808–8817.

[55] Marvin Zhang, Henrik Marklund, Abhishek Gupta, Sergey Levine, and Chelsea Finn. 2020. Adaptive Risk Minimization: A Meta-Learning Approach for Tackling Group Shift. *arXiv preprint arXiv:2007.02931* (2020).

[56] Han Zhao, Remi Tachet Des Combes, Kun Zhang, and Geoffrey Gordon. 2019. On learning invariant representations for domain adaptation. In *International Conference on Machine Learning*. PMLR, 7523–7532.

[57] Shanshan Zhao, Mingming Gong, Tongliang Liu, Huan Fu, and Dacheng Tao. 2020. Domain Generalization via Entropy Regularization. *Advances in Neural Information Processing Systems* 33 (2020).

[58] Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. 2020. Deep domain-adversarial image generation for domain generalisation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 13025–13032.