Quantitatively Measuring and Contrastively Exploring Heterogeneity for Domain Generalization

Yunze Tong
tyz01@zju.edu.cn
Zhejiang University
Hangzhou, China

Junkun Yuan
yuanjk@zju.edu.cn
Zhejiang University
Hangzhou, China

Min Zhang
zhangmin.milab@zju.edu.cn
Zhejiang University
Hangzhou, China

Didi Zhu
didi_zhu@zju.edu.cn
Zhejiang University
Hangzhou, China

Keli Zhang
zhangkeli1@huawei.com
Noah's Ark Lab, Huawei Technologies
Shenzhen, China

Fei Wu
wufei@zju.edu.cn
Zhejiang University
Hangzhou, China

Kun Kuang*
kunkuang@zju.edu.cn
Zhejiang University
Hangzhou, China

ABSTRACT

Domain generalization (DG) is a prevalent problem in real-world applications, which aims to train well-generalized models for unseen target domains by utilizing several source domains. Since domain labels, *i.e.*, which domain each data point is sampled from, naturally exist, most DG algorithms treat them as a kind of supervision information to improve the generalization performance. However, the original domain labels may not be the optimal supervision signal due to the lack of domain heterogeneity, *i.e.*, the diversity among domains. For example, a sample in one domain may be closer to another domain, its original label thus can be the noise to disturb the generalization learning. Although some methods try to solve it by re-dividing domains and applying the newly generated dividing pattern, the pattern they choose may not be the most heterogeneous due to the lack of the metric for heterogeneity. In this paper, we point out that domain heterogeneity mainly lies in variant features under the invariant learning framework. With contrastive learning, we propose a learning potential-guided metric for domain heterogeneity by promoting learning variant features. Then we notice the differences between seeking variance-based heterogeneity and training invariance-based generalizable model. We thus propose a novel method called **H**eterogeneity-based **T**wo-stage **C**ontrastive **L**earning (HTCL) for the DG task. In the first stage, we generate the most heterogeneous dividing pattern with our contrastive metric. In the second stage, we employ an invariance-aimed contrastive learning by re-building pairs with the stable relation hinted by domains and classes, which better utilizes generated domain labels for generalization learning. Extensive experiments show HTCL better digs heterogeneity and yields great generalization performance.

CCS CONCEPTS

• **Computing methodologies** → *Learning latent representations.*

KEYWORDS

Domain Generalization; Heterogeneity; Invariant Learning; Contrastive Learning

ACM Reference Format:
Yunze Tong, Junkun Yuan, Min Zhang, Didi Zhu, Keli Zhang, Fei Wu, and Kun Kuang. 2023. Quantitatively Measuring and Contrastively Exploring Heterogeneity for Domain Generalization. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23), August 6–10, 2023, Long Beach, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3580305.3599481

1 INTRODUCTION

Machine learning has exhibited extraordinary and super-human performance on various tasks [9, 16]. Its common paradigm, Empirical Risk Minimization (ERM) [53], trains models by minimizing the learning error on training data under the independent and identically distributed (i.i.d.) assumption. However, distribution shifts, *i.e.*, out-of-distribution (OOD) problems, usually exist between training and real-world testing data, which leads to the performance degradation of ERM-based methods [10, 13, 37, 45]. In order to solve the OOD generalization problem, researchers turn to developing methods for domain generalization (DG), a task that aims to generalize the models learned in multiple source domains to other unseen target domains. In DG, the features of samples can be divided into two parts: invariant (*a.k.a.* semantic or stable) features and variant (*a.k.a.* non-semantic or unstable) features. Invariant features are those having a domain-invariant relation with true class labels, while variant features shift with the change of domains. Thus, DG methods tend to perform invariant learning: identify the stable

*Corresponding author.

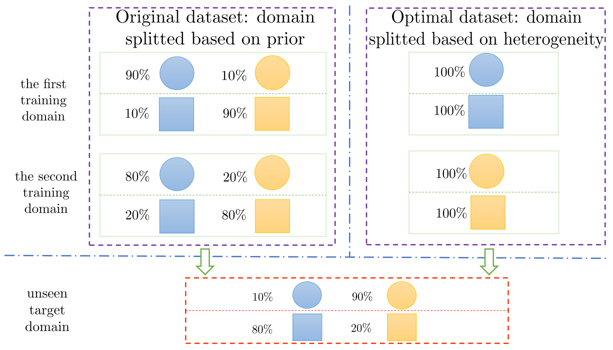

Original dataset: domain splitted based on prior
Optimal dataset: domain splitted based on heterogeneity
the first training domain
the second training domain
unseen target domain
90%
10%
10%
90%
80%
20%
20%
80%
100%
100%
100%
100%
10%
90%
80%
20%

Figure 1: The toy example for illustrating the problem of lack in heterogeneity. See main text for details.

relations among all source domains and predict class labels with invariant parts. ERM-based approaches may easily learn and rely on variant features for prediction [13] because they even do not tell which domain the training samples are from at all. Assuming that all data come from one distribution, they achieve high accuracies in seen domains but might degrade a lot in unseen target domains.

To tackle this issue, some methods turn to using the gap between different domains to surpass ERM. They utilize the original domain labels as extra supervision signals [1, 4, 35, 36, 48]. Data are separately treated according to which domain they are sampled from. Among them, some fine-grained methods try to minimize some statistical metrics (*e.g.*, MMD [22, 50] or Wasserstein distance [69]) among feature representations of different domains. Models trained by these methods are more robust to complex domains than ERM and show good performances in bridging the domain gap. However, these methods neglect the quality of the original domain labels they use, *i.e.*, the risk of lacking heterogeneity. We illustrate this phenomenon with Figure 1. Here we aim to predict classes by shapes, which are invariant features, while the colors are domain-variant. In realistic scenarios, we may follow a make-sense prior that making the samples within a domain diverse as the left of Figure 1. However, when performing DG, the left pattern will easily make the model predict labels with domain-variant features because their domains own two very similar distributions in colors. In fact, the optimal pattern is shown on the right of Figure 1, which makes the domain-variant features heterogeneous across domains and homogeneous within each domain. As illustrated above, sub-optimal domain labels will introduce bias during training and mislead the generalization learning. To reduce the bias, enlarging domain heterogeneity, *i.e.*, the variety across domains, is an effective approach. From this view, domain labels could be regarded as heterogeneity-focused dividing according to prior. If source domains are heterogeneous enough, models may be more robust to unseen domains because source sub-populations provide more kinds of data distribution during training. However, if source domains are homogeneous, the latent data distribution and predictive strategy within every source domain may tend to be similar. Models may tend to rely on a common predictive path for training, which is unstable for unseen distributions. Therefore, digging domain heterogeneity is significant for reducing bias in DG.

Recently, some methods notice the importance of data heterogeneity and turn to utilize heterogeneity to boost generalization learning. A more heterogeneous dataset is believed to be able to help disentangle features better and cut off the shortcut of neural networks to variant features [8]. They mostly generate a dividing pattern (*a.k.a.* infer environments or split groups) to re-separate training samples into newly generated domains. By applying their patterns, they achieve favorable accuracies [8, 31, 56]. However, there is no precise definition and metric for heterogeneity in the DG task yet. Without the metric during domain label generation, the chosen dividing pattern may be sub-optimal, which might even bring new noise and disturb generalization learning. In addition, their experiments are mainly based on synthetic or low-dimension data, which is insufficient to verify their potential in dividing domains and generalization abilities in real-world scenarios.

In this paper, we propose a quantitative learning potential-guided heterogeneity metric and introduce a heterogeneity-based two-stage DG algorithm through contrastive learning. We first point out that domain heterogeneity mainly lies in variant features in DG under the invariant learning framework. Our metric is calculated with the ratio of the average distance between the representations of different domains, which needs to construct and contrast the same-class pairs within each domain and across domains. When it involves measuring the heterogeneity of a pattern, we apply the metric to the features from a variance-focused model, which also indicates the potential to obtain heterogeneity. Our method comprises generating heterogeneous patterns and enhancing generalization with the pattern. We select the most heterogeneous dividing pattern from the generated ones in the first stage, which is measured with our heterogeneity metric quantitatively. The first stage contains two interactive modules: the heterogeneity exploration module and the pattern generation module. They are performed iteratively and boosted by each other. In the second stage, with the domain labels generated by the first stage, we construct positive pairs with same-class and different-domain data while negative ones indicate different-class data within the same domain. Then an invariance-aimed contrastive learning is employed to help train a well-generalized model for the DG task.

To summarize, our contributions include:

- We point out the heterogeneity in DG, *i.e.*, the original domain labels may not be optimal when treated as supervision signals. Sub-optimal domain labels will introduce bias and disturb the generalization learning.
- Under the insight that heterogeneity mainly lies in variant parts, we propose a metric for measuring domain heterogeneity quantitatively. Through the features from a variance-focused model, our metric contrast the same-class pairs within each domain and across domains.
- We propose a heterogeneity-based two-stage contrastive learning method for DG. It incorporates both generating heterogeneous dividing patterns and utilizing generated domain labels to guide generalization learning.
- Extensive experiments exhibit the state-of-the-art performance of our method. Ablation studies also demonstrate the significance of each stage in our method.

2 RELATED WORK

We first provide an overview of the related work involved with our setting and method.

Domain generalization. Domain generalization (DG) aims to train a well-generalized model which can avoid the performance degradation in unseen domains [62]. Domains are given as natural input in DG, which means models can access more information. However, sub-optimal signals will also induce distribution shift [70] or make the model rely on spurious features [1]. That's why DG may be more hard to solve compared to ordinary classification tasks. Several different approaches are proposed to tackle this problem. Some works use data augmentation to enlarge source data space [55, 58, 59, 61, 66, 70]. Methods based on GroupDRO [48] or IRM [1] turn to optimize one or some specific groups' risk to achieve higher generalization [29, 30, 72]. There are also some methods employing feature alignment [2, 49, 73] or decorrelation [28] to guide the learning procedure.

Heterogeneity in domain generalization. Data heterogeneity broadly exists in various fields. In DG, heterogeneity mainly refers to the diversity of each domain. It is caused by the distribution shifts of data when dividing the implicit whole data distribution into different sub-populations, *i.e.*, domains. Though there are some formulations for data heterogeneity in other machine learning problems [26, 33, 51], domain heterogeneity has no precise and uniform metric till now. EIIL [8] is one of the first methods that notice the gain brought by re-dividing domains in DG. It infers domain labels with a biased model. HRM [31] designs a framework where dividing domains and performing invariant learning are optimized jointly. KerHRM [32] develops HRM by integrating the procedure with kernel methods to better capture features. IP-IRM [56] also takes the strategy of dividing domains to better disentangle features from a group-theoretic view. Above novel methods achieve favorable performances on synthetic or low-dimension data. However, they all treat generating dividing patterns as a middle process and haven't proposed the metric for domain heterogeneity. In addition, their potentials in high-dimension data, which are less decomposed on the raw feature level, are not fully verified. These are the issues we aim to make up in this paper.

Invariant learning. Invariant learning actually can be seen as one of the effective approaches in DG. Its core is differentiating invariant and variant parts in an image and making the model tend to learn the invariant features across different domains. Following this line, some works [34, 38, 39, 63, 74] achieve the goal by the pre-defined causal graph to learn the key features involving class labels. Some works also consider the disentangled method [24, 40, 42, 44, 57, 65, 71] aiming to split the features completely. Besides, ZIN [30] proposes to use explicit auxiliary information to help learn stable and invariant features, which is also a promising direction.

Contrastive learning. Contrastive learning (CL) contrasts semantically similar and dissimilar pairs of samples, which aims to map positive sample pairs closer while pushing apart negative ones in the feature space. The prototype of CL is the architecture of Contrastive Predictive Coding (CPC) [43]. It contains InfoNCE loss which can be optimized to maximize a lower bound on mutual information in the theoretical guarantee. Contrastive learning has already achieved great success in self-supervised learning [6, 15] due to the need of modeling unlabeled data. It has been applied in various tasks due to the ability to make the hidden representations of the samples from the same class close to each other [23, 25, 64, 68].

3 PROBLEM SETTING

We formalize the problem setting in domain generalization (DG) task. Suppose that we have source data $\mathcal{D} = \mathcal{X} \times \mathcal{Y}$ for a classification task, every sample x_i follows $x_i \in \mathcal{X}$ and its label y_i follows $y_i \in \mathcal{Y}$. Typically, we need a non-linear feature extractor ϕ and a classifier w. ϕ will map the sample from input space to representation space, which can be formalized as $\phi : \mathcal{X} \to \mathcal{H}$. $\mathcal{H}$ denotes the representation space here. Then $\phi(x_i) \in \mathbb{R}^d$ denotes the d-dimension feature representation of x_i. $w(\phi(x_i)) \in \mathbb{R}^{|\mathcal{Y}|}$ denotes the predicted possibilities for all labels. In DG, every sample has its initial domain label. Let ε^{tr} be the training domains (*a.k.a.* environments or groups). Then we use $d_{x_i} \in \varepsilon^{tr}$ to denote the domain label of the training sample x_i.

Some solutions don't rely on domain labels. For example, Empirical Risk Minimization (ERM) [53] minimizes the loss over all the samples no matter which domain label they have:

$$\mathcal{L}_{ERM} = E_{(x_i,y_i)\in\mathcal{D}}[\ell(w(\phi(x_i)), y_i)], \tag{1}$$

where ℓ is the Cross Entropy loss function for classification. For some other methods which utilize domain labels, the risk will have an extra penalty term. Take Invariant Risk Minimization (IRM) [1] as an example,

$$\mathcal{L}_{IRM} = \sum_{\epsilon\in\varepsilon^{tr}} R^{\epsilon}(\phi, w) + \lambda \left\|\nabla_{\bar{w}} R^{\epsilon}(\bar{w}\circ\phi)\right\|. \tag{2}$$

Here $R^{\epsilon}(\phi, w) = E_{\{(x_i,y_i)|d_{x_i}=\epsilon\}}[\ell(w(\phi(x_i)), y_i)]$ denotes the per-domain classification loss. The second term in Equation (2) enforces simultaneous optimality of the ϕ and w in every domain. $\bar{w}$ is a constant scalar multiplier of 1.0 for each output dimension.

No matter whether to use domain labels, the aims of DG methods are consistent: training a robust model which can generalize to unseen domains with only several source domains.

4 DOMAIN HETEROGENEITY

4.1 Variance-based Heterogeneity

In domain generalization (DG), all the samples are always given with their specific domain labels due to the need of differentiating source domains during training. Therefore, domain labels reflect the process of splitting data into several sub-populations, which denotes domain heterogeneity. Domain heterogeneity has a close connection with training models. Suppose that all the training and testing data form an implicit overall distribution. After splitting them into several domains, the data points in each domain will form a complete data distribution and have their specific predictive mechanisms. If the source domains are homogeneous, the latent distribution and predictive strategy within every source domain may tend to be similar. Models trained in these domains will also tend to learn a common predictive mechanism because they can hardly access novel distributions. Therefore, when facing the target unseen domains, the robustness of the model may not be favorable. Therefore, pursuing domain heterogeneity is important for DG.

Here we explain where domain heterogeneity lies. Following the idea of invariant learning, the feature representation obtained from supervised learning is composed of invariant features and variant features. The invariant parts will help predict labels while the variant parts may harm training. Therefore, common methods use techniques to make the model sensitive to the invariant parts in latent space as much as possible instead of the variant ones. For a pair of same-label samples, their invariant parts should be naturally close because they own the same class label, which highly involves invariant features. While their variant parts can show arbitrary relations. For example, two samples whose labels are both dogs may have similar invariant parts like dogs' noses. But as for variant parts, they can be various. They can be similar if they both have grass as the background while they can be totally different if one is in the grass and the other is in the water.[1] An ideal heterogeneous dividing pattern focuses more on the **variant** parts. The reason is that: the target unseen domain will share the same invariant information (because they are highly involved with the label information, which is shared among every domain) with the source domains, while the variant parts from the target domain may be unseen during training. Thus, if we make the variant parts of source domains homogeneous within a single domain and heterogeneous among different domains, we will maximally imitate the situation of facing unseen domains during training. Therefore, our goal comes to: **re-divide domain labels for every sample to form a more heterogeneous dividing pattern than original one**.

4.2 The Metric of Heterogeneity

Since the heterogeneity among source domains counts, how to measure it quantitatively becomes the main problem. Obviously, different dividing patterns (different allocation of domain labels for all samples) have different heterogeneity because of the bias in the variant parts. Suppose that we let a model with fixed architecture learn the representation with a dividing pattern. If the dividing pattern is consistent with the assumptions in section 4.1, *i.e.*, the variant parts are homogeneous enough within each domain and heterogeneous enough across different domains, then the following requirement should be fulfilled: for any pair of same-class samples, the distance between their representations should be as small as possible if they are from the same domain while the distance should be as large as possible if they are from different domains.

Considering that the heterogeneity lies in the variant features as stated above, we characterize the heterogeneity quantitatively with the proportion between the distances of different groups of the same-class features:

$$\mathcal{L}_H = \sum_{(x,y)} log(\frac{E_{\{(x',y')|d_{x'}=d_x, y'=y\}}[||\phi(x)-\phi(x')||_2]}{E_{\{(x'',y'')|d_{x''}\neq d_x, y''=y\}}[||\phi(x)-\phi(x'')||_2]}). \quad (3)$$

For a sample (x, y), we collect all its same-class sample ($y = y' = y''$). Then we set x's positive (negative) pair samples x' (x'') as the ones which are from the same (different) domain $d_{x'} = d_x$ ($d_{x''} \neq d_x$). The more heterogeneous the dividing pattern is, the less will the numerators become and the more will the denominators become. $\mathcal{L}_H$ thus will become less. We use Equation (3) to judge whether the current dividing pattern is more heterogeneous in the iterative process. We calculate the distance between the features instead of the original data here. Recall section 4.1, we can have such intuition because we can differentiate the variant or invariant part just with original images. But as for a fixed model, the representation it learns will lose information because of the reduction of the dimensions. From the view of neural networks, it is the similarity between low-dimension representation mainly works. As a result, if we want to measure heterogeneity, we should concentrate on the similarity between full-trained representations instead of the original data. Therefore, we put stress on the potential for learning heterogeneous features with the given dividing pattern. We then use an already-trained model ϕ to generate features instead of using other methods (*e.g.*, kernel tricks) only performing on the data itself.

Then it comes to how to train ϕ. Note that our metric for heterogeneity stresses the potential to learn the best variant representation in a given dividing pattern. Therefore, training variance-based ϕ should minimize the loss containing classification error and the guidance for exploring heterogeneity, *i.e.*, variant features in DG:

$$\mathcal{L}_{var} = \sum_{(x,y)} \ell(w(\phi(x)), y) + \lambda_1 \mathcal{L}_H. \quad (4)$$

The first term ℓ makes sure ϕ focus on the feature which carries helpful information for classification (no matter if it is an invariant feature or variant feature). The second term is our metric for heterogeneity. Minimizing it promotes the variant features becoming different across different domains and becoming similar in the same domain. Hyperparameter λ_1 balances the two terms.

4.3 The Utilization of Heterogeneity

Given an ideal heterogeneous dividing pattern following our metric, it comes to training feature extractor ϕ and classifier w for the classification task. Here we aim to make feature extractor ϕ focus more on the invariant parts in each image. With the reorganized domain labels from the last stage, the variant parts of the samples from the same domain would be homogeneous. Therefore, we transfer our target to other two kinds of sample pairs: the sample pairs with the same domain label and different class labels, and the sample pairs with different domain labels and the same class label. It is obvious that in representation space, the features should follow the strategy of making intra-class distance small and inter-class distance large. We continue using the dog's example in section 4.1 to explain the motivation. Here we have already obtained a heterogeneous dividing pattern by reorganizing domain labels. The model should learn the invariant features (*e.g.*, animals' noses or hairs) which will help distinguish whether the sample is a dog or cat, not the variant features (*e.g.*, the background or the style of the image) which tend to be homogeneous within a single domain and heterogeneous across different domains. Suppose that there are two domains that are heterogeneous, the first domain has images of cats or dogs in the grass and the second domain has same-label images whose background is water. ϕ fulfilling $dist(\phi(x_{dog,grass}), \phi(x_{dog,water})) \ll dist(\phi(x_{dog,grass}), \phi(x_{cat,grass}))$ would be more appropriate for predicting the label. To achieve the goal of encoding more invariant features rather than variant features, we leverage two specific kinds of sample pairs mentioned above. For $(x, y) \in \mathcal{D}$, its negative pair samples are images having different class labels from y and the same domain label as d_x:

$$P_x^- = \{x' | d_{x'} = d_x, y' \neq y\}.$$

[1]In some DG datasets, the variant parts mainly refers to the style of the whole picture instead of the background in our example.

The positive pair samples are images having the same class label as y and different domain labels from d_x:

$$P_x^+ = \{x' | d_{x'} \neq d_x, y' = y\}.$$

We use Maximum Mean Discrepancy (MMD) as an estimation of the similarity of a pair of features. Then we write a penalty term, which contrasts the pairs to promote utilizing domain heterogeneity, as:

$$-\sum_x log \frac{\sum_{x_- \in P_x^-} mmd(\phi(x), \phi(x_-))}{\sum_{x_- \in P_x^-} mmd(\phi(x), \phi(x_-)) + \sum_{x_+ \in P_x^+} mmd(\phi(x), \phi(x_+))}. \quad (5)$$

Here we compare the metric in Equation (3) and the regularization term in Equation (5) again. Their aims are totally different. In Equation (3), the core lies in utilizing the relations among variant features, which in turn helps train ϕ for digging the potential of enlarging heterogeneity in Equation (4). However, when it involves utilizing heterogeneity, the ultimate goal is training the best feature extractor ϕ and classifier w for classification. In DG, the ideal ϕ will only encode invariant features and not encode variant features to representation space at all. So our aim is to make the feature extractor focus more on the invariant features.

5 PROPOSED METHODS

Our heterogeneity-based two-stage contrastive learning (HTCL) method is based on the problem setting in section 3 and the metric in section 4.2. Following the analysis above, we treat the procedure of exploring heterogeneity and the one of training models with newly generated domain labels separately. The first stage aims to generate a heterogeneous dividing pattern. It can be seen as a preprocessing stage that only re-divides domain labels and outputs the most heterogeneous ones, which corresponds to section 4.2. The second stage receives the domain labels from the first stage as input. During its training process, it utilizes the contrastive loss mentioned in section 4.3 to help learn invariant features. We detail these two stages in section 5.1 and 5.2. Note that the two stages have no iteration process. They are performed only once in a consistent order while the modules **within** the first stage have an iterative process. That's a difference from existing methods [8, 31, 32, 56] mentioned in section 1, which also aim to generate domain labels by themselves but incorporate dividing domains and invariant learning together. Algorithm (1) shows the whole process of HTCL.

5.1 Heterogeneous Dividing Pattern Generation

With the training set and original domain labels, we explore heterogeneity by dividing images into $|\varepsilon^{tr}|$ disjoint domains. However, exploring heterogeneity with variant features and measuring heterogeneity can't be optimized simultaneously. As a result, we design two interactive modules and make them work alternately.

5.1.1 Heterogeneity exploration module. A fixed dividing pattern and raw samples are given as input. Our goal includes:

- learning an adequate ϕ guided by Equation (4) and generating the low-dimension representation $\sum_{x_i} \phi(x_i)$ of all samples for the next pattern generation module.
- measuring the heterogeneity of the input dividing pattern with the metric $\mathcal{L}_H$ quantitatively.

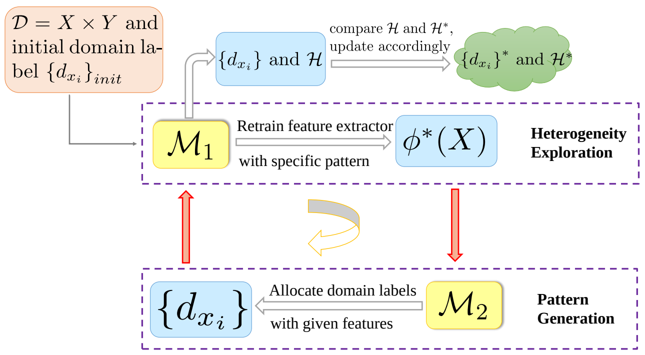

$\mathcal{D} = X \times Y$ and initial domain label $\{d_{x_i}\}_{init}$
$\{d_{x_i}\}$ and $\mathcal{H}$
compare $\mathcal{H}$ and $\mathcal{H}^*$, update accordingly
$\{d_{x_i}\}^*$ and $\mathcal{H}^*$
$\mathcal{M}_1$
Retrain feature extractor
with specific pattern
$\phi^*(X)$
Heterogeneity Exploration
$\{d_{x_i}\}$
Allocate domain labels
with given features
$\mathcal{M}_2$
Pattern Generation

Figure 2: The framework for the first stage of HTCL. $\mathcal{M}_1$ denotes the heterogeneity exploration module, and $\mathcal{M}_2$ refers to the pattern generation module. $\mathcal{M}_1$ outputs a well-trained feature representation for $\mathcal{M}_2$. $\mathcal{M}_2$ returns a newly generated dividing pattern to $\mathcal{M}_1$. The input and output of each module are shown in blue boxes. The final optimal output $\{d_{x_i}\}^*$, which will be used as the domain labels in ultimate invariant learning, is shown in green box. It updates according to the heterogeneity metric during the iterations.

For the first goal, we can just update ϕ and w through Equation (4). When it refers to constructing the corresponding positive and negative pairs for $\mathcal{L}_{var}$, we only construct them within every batch instead of the whole dataset.

Here we re-write $\mathcal{L}_H$ which has been mentioned in Equation (3). $\mathcal{L}_H$ is our heterogeneity metric, and we also apply it in Equation (4) to guide heterogeneity exploration. Considering the calculation is conducted per batch, we re-formalize $\mathcal{L}_H$ more detailedly as:

$$\mathcal{L}_H = -\sum_{(X_i, y_i)} \sum_{(d_m, d_n)} \log \frac{dist(\phi(X_i^{d_m}), \phi(X_i^{d_n}))}{dist(\phi(X_i^{d_m})) + dist(\phi(X_i^{d_n}))}. \quad (6)$$

We denote a single batch by $B = \{(x, y)^{|B|}\} \subset \mathcal{D}$. For each kind of label $y_i \in \mathcal{Y}$, X_i denotes the set of all images which own class label y_i in the batch B. d_{X_i} denotes the set of all domains which contain at least one sample in X_i. $\forall (d_m, d_n)$ s.t. $d_m \in d_{X_i}, d_n \in d_{X_i}, d_m \neq d_n$, $X_i^{d_m}$ and $X_i^{d_n}$ denotes the samples whose domain labels are d_m and d_n separately. For convenience, here we set the input of feature extractor ϕ as a set of images and every dimension of the features denotes the predicted probability of each label for a single image. Then we set function $dist(\cdot, \cdot)$ to measure the average distance between different kinds of representation:

$$dist(\phi_1, \phi_2) = \frac{\sum_{i=0}^{|\phi_1|-1} \sum_{j=0}^{|\phi_2|-1} \|\phi_1[i] - \phi_2[j]\|_2}{|\phi_1| \times |\phi_2|}. \quad (7)$$

In above equation, ϕ_1 has a shape of $|\phi_1| \times d$ and the entry $\phi_1[i]$ denotes the representation of i-th image in ϕ_1. ϕ_2 has a similar form. We then calculate the average distance between different representations. Similarly, we can measure the average distance among same-domain and same-label samples as follows:

$$dist(\phi) = \frac{\sum_{i=0}^{|\phi|-2} \sum_{j=i+1}^{|\phi|-1} \|\phi[i] - \phi[j]\|_2}{|\phi|^2 - |\phi|}. \quad (8)$$

After training for several epochs, we obtain the representation with the final ϕ^* as the optimal feature extractor. Then it comes to the second goal: measuring the heterogeneity. Similarly, we calculate the ratio batchwisely and finally take the expectation among all batches' counted pairs to avoid the problem brought by batches' random shuffle:

$$H = E_{B \subset \mathcal{D}}[\mathcal{L}_H]. \tag{9}$$

Thus, H specifies the quantity of the heterogeneity in a given dividing pattern. Note that when performing Equation (9), we use a trained feature extractor ϕ^* to replace ϕ in $\mathcal{L}_H$, which makes sure Equation (9) reflect the maximum potential of the variance-based model in such a dividing pattern.

Remark. *When it refers to heterogeneity, the subjective is feature representation rather than raw data itself. A fully-trained representation reflects the learning potential of a fixed model with a given dividing pattern. So we choose to train ϕ in such above Equation (4) manner before measuring heterogeneity instead of directly performing clustering methods (e.g., density-based) on raw data.*

5.1.2 Pattern generation module. In this module, input $\phi^*(\mathcal{X})$ is the low-dimension representation of all the samples from the last module. We divide them into $|\varepsilon^{tr}|$ disjoint domains to form a new dividing pattern. We use a simple multilayer perceptron (MLP) $f(\cdot)$ to map representation to $|\varepsilon^{tr}|$ domains with probability: $f(\phi^*(\mathcal{X})) \in \mathbb{R}^{|\mathcal{X}| \times |\varepsilon^{tr}|}$. No supervised signals are used in this module. But we should avoid two situations:

Generating process is influenced by invariant features too much. Considering that heterogeneity exploration module inevitably uses class-label as supervision information, the input of this module will naturally carry invariant features. Suppose that ϕ^* can mainly extract invariant features of samples x_1, x_2, x_3 in the early stage. If x_1 and x_2 share the same class while x_3 doesn't, the distance between $\phi^*(x_1)$ and $\phi^*(x_2)$ will naturally be less than the distance between $\phi^*(x_1)$ and $\phi^*(x_3)$ no matter whether x_1 and x_3 share the similar variant features or not. In a word, same-label samples tend to be close to each other. Then they may be divided into one domain when invariant features dominate a main position in the representation space. There are two drawbacks. On the one hand, when feeding this kind of dividing pattern into heterogeneity exploration module, we can't construct effective pairs to boost the process of learning variant features and measuring heterogeneity. On the other hand, if domain labels are highly involved with class labels, the existence of domain labels will be of no help to the final process of learning semantic invariant features. Commonly, the number of domains is less than the number of labels[2]. From the view of information theory, the information brought by domain labels is the subset of the information brought by class labels. Then our final training process will degrade to ERM. Therefore, we should avoid this module generating dividing patterns by actual classes.

The numbers of samples contained in each domain differ a lot. If a domain contains few samples while another domain contains a lot, it will cause similar problems mentioned in the last point. Thus, we should also balance the numbers in all the domains.

The overall predictive probability of mapped domains, *i.e.*, $f(\phi^*(\mathcal{X}))$, has a shape of $\mathbb{R}^{|\mathcal{X}| \times |\varepsilon^{tr}|}$. We calculate along the first dimension to obtain all samples' average probability distribution: $f(\phi^*(\mathcal{X}))_{avg} \in \mathbb{R}^{1 \times |\varepsilon^{tr}|}$. Then we guide $f(\cdot)$ which generates candidate domain labels as follows:

$$\mathcal{L}_{divide} = H(f(\phi^*(\mathcal{X}))_{avg}) + (\frac{1}{|\epsilon^{tr}|} - \min(f(\phi^*(\mathcal{X}))_{avg})). \tag{10}$$

Note that ϕ^* is fixed here and we only update $f(\cdot)$. $H(p)$ denotes the entropy of the distribution p. The first term uses entropy on classes to encourage the same-label samples not to shrink to a single domain, which reduces the influence of invariant features. The second term is a penalty term to avoid the minimum sample number in one domain becoming too small.

5.1.3 Summary of generating a heterogeneous dividing pattern. Generating a heterogeneous dividing pattern is the core stage in our method. We design two interactive modules to iteratively learn the optimal dividing pattern which contains more variant information. At the start, we use original domain labels as the initial dividing pattern and send them to the heterogeneity exploration module. Then we update ϕ with Equation (4) and measure the heterogeneity with Equation (9). After obtaining the optimal feature representation, we send it into the pattern generation module to generate a new dividing pattern with Equation (10). The newly generated dividing pattern is then sent as input to the heterogeneity exploration module again. We repeat this procedure several times to learn variant features as possible and select the best dividing pattern which obtains the minimum value in Equation (9). Figure 2 illustrates the whole framework.

Algorithm 1 The pseudo code of HTCL

```
Input: training samples $\mathcal{D} = \mathcal{X} \times \mathcal{Y}$, original domain labels
    $\{d_{x_i}\}_{init}$, hyperparamters $\lambda_1, \lambda_{cont}, \lambda_{mmd}, T_1, T_2$
  // The first stage: heterogeneous dividing pattern generation
  Initialization: $\phi, w$ initialized with pretrained ResNet, $\{d_{x_i}\} \leftarrow$
    $\{d_{x_i}\}_{init}$, $H^* \leftarrow 0$
  repeat
    Update $\phi$ based on Equation (4) till convergence
    $\phi^* \leftarrow \phi$
    Measure heterogeneity of $\{d_{x_i}\}$ with Equation (9), obtain $H$
    if $H < H^* \vee H^* == 0$ then
      $H^* = H$
      $\{d_{x_i}\}^* = \{d_{x_i}\}$
    end if
    Generate a new dividing pattern $\{d_{x_i}\}$ with Equation (10)
  until reach $T_1$ iterations
  // The second stage: prediction with heterogeneous dividing pattern
  Initialization: $\phi, w$ initialized with pretrained ResNet again,
    $\{d_{x_i}\}^*$ received from the first stage
  repeat
    Update $\phi, w$ with Equation (14) using $\{d_{x_i}\}^*$ as domain labels
  until reach $T_2$ iterations
Output: $\phi, w$
```

[2]Actually, all the datasets we used in the experiment fulfill this condition

Table 1: The comparison with other DG methods in standard DomainBed [14] benchmark. We report average accuracy on all target domains under three runs. The results of baselines are from their original corresponding literatures or DomainBed. We highlight the best results with boldface.

Algorithm	PACS	VLCS	OfficeHome	TerraIncognita	**Avg.**
ERM [53]	$85.5_{\pm 0.2}$	$77.5_{\pm 0.4}$	$66.5_{\pm 0.3}$	$46.1_{\pm 1.8}$	68.9
IRM [1]	$83.5_{\pm 0.8}$	$78.5_{\pm 0.5}$	$64.3_{\pm 2.2}$	$47.6_{\pm 0.8}$	68.4
GroupDRO [48]	$84.4_{\pm 0.8}$	$76.7_{\pm 0.6}$	$66.0_{\pm 0.7}$	$43.2_{\pm 1.1}$	67.5
Mixup [60]	$84.6_{\pm 0.6}$	$77.4_{\pm 0.6}$	$68.1_{\pm 0.3}$	$47.9_{\pm 0.8}$	69.5
MLDG [20]	$84.9_{\pm 1.0}$	$77.2_{\pm 0.4}$	$66.8_{\pm 0.6}$	$47.7_{\pm 0.9}$	69.2
CORAL [50]	$86.2_{\pm 0.3}$	$\mathbf{78.8}_{\pm 0.6}$	$68.7_{\pm 0.3}$	$47.6_{\pm 1.0}$	70.4
MMD [22]	$84.6_{\pm 0.5}$	$77.5_{\pm 0.9}$	$66.3_{\pm 0.1}$	$42.2_{\pm 1.6}$	67.7
DANN [12]	$83.7_{\pm 0.4}$	$78.6_{\pm 0.4}$	$65.9_{\pm 0.6}$	$46.7_{\pm 0.5}$	68.7
CDANN [27]	$82.6_{\pm 0.9}$	$77.5_{\pm 0.1}$	$65.7_{\pm 1.3}$	$45.8_{\pm 1.6}$	67.9
MTL [4]	$84.6_{\pm 0.5}$	$77.2_{\pm 0.4}$	$66.4_{\pm 0.5}$	$45.6_{\pm 1.2}$	68.5
SagNet [41]	$86.3_{\pm 0.2}$	$77.8_{\pm 0.5}$	$68.1_{\pm 0.1}$	$48.6_{\pm 1.0}$	70.2
ARM [67]	$85.1_{\pm 0.4}$	$77.6_{\pm 0.3}$	$64.8_{\pm 0.3}$	$45.5_{\pm 0.3}$	68.3
VREx [19]	$84.9_{\pm 0.6}$	$78.3_{\pm 0.2}$	$66.4_{\pm 0.6}$	$46.4_{\pm 0.6}$	69.0
RSC [17]	$85.2_{\pm 0.9}$	$77.1_{\pm 0.5}$	$65.5_{\pm 0.9}$	$46.6_{\pm 1.0}$	68.6
CAD [46]	$85.2_{\pm 0.9}$	$78.0_{\pm 0.5}$	$67.4_{\pm 0.2}$	$47.3_{\pm 2.2}$	69.5
CausalRL-CORAL [7]	$85.8_{\pm 0.1}$	$77.5_{\pm 0.6}$	$68.6_{\pm 0.3}$	$47.3_{\pm 0.8}$	69.8
CausalRL-MMD [7]	$84.0_{\pm 0.8}$	$77.6_{\pm 0.4}$	$65.7_{\pm 0.6}$	$46.3_{\pm 0.9}$	68.4
HTCL (Ours)	$\mathbf{88.6}_{\pm 0.3}$	$77.6_{\pm 0.5}$	$\mathbf{71.3}_{\pm 0.6}$	$\mathbf{50.9}_{\pm 1.9}$	**72.1**

5.2 Prediction with Heterogeneous Dividing Pattern

In this part, our aim changes from learning a heterogeneous dividing pattern to learning invariant features from the samples to help predict the labels in the unseen target domain. Apart from the standard classification loss, we add a distance-based term to prevent the representations between different domains too far. We measure the distance between two different representations as:

$$mmd(D_1, D_2) = \frac{1}{4d^2} \|Cov(D_1) - Cov(D_2)\|_F^2 . \tag{11}$$

$Cov(\cdot)$ means covariances and $\|\cdot\|_F^2$ denotes the squared matrix Frobenius norm. d still denotes the number of dimensions of the feature representation. The invariance-based contrastive loss for better utilizing the heterogeneous domain labels takes the form as:

$$\mathcal{L}_{cont} = \sum_{\epsilon \in \varepsilon^{tr}} \sum_{y \in \mathcal{Y}} \log(1 + \frac{mmd(\phi(X_s), \phi(X_{pos}))}{mmd(\phi(X_s), \phi(X_{neg}))}). \tag{12}$$

Consistent to section 4.3, here X_s denotes all the source samples belonging to domain ϵ and owning label y. X_{pos} contains all the postive pair samples for X_s, which are not from domain ϵ but own the same class as y. X_{neg} are from ϵ while have different classes from y. Then we calculate all the distances between different domain representations like the method in CORAL [50]:

$$\mathcal{L}_{mmd} = \sum_{\epsilon_1, \epsilon_2 \in \varepsilon^{tr},\ \epsilon_1 \neq \epsilon_2} mmd(\phi(X_{\epsilon_1}), \phi(X_{\epsilon_2})). \tag{13}$$

Then the ultimate objective function for DG turns to:

$$\mathcal{L}_{predict} = \sum_{(x,y)} \ell(w(\phi(x)), y) + \lambda_{cont} \mathcal{L}_{cont} + \lambda_{mmd} \mathcal{L}_{mmd}. \tag{14}$$

λ_{cont} and λ_{mmd} are the hyperparameters to balance each guidance.

6 EXPERIMENTS

6.1 Experimental Settings

We first introduce the common settings of our experiments.

Datasets. We use four kinds of domain generalization (DG) datasets to evaluate our proposed HTCL method:

- PACS [21] contains 9991 images shared by 7 classes and 4 domains {art, cartoon, photo, and sketch}.
- OfficeHome [54] contains 15588 images shared by 65 classes and 4 domains {art, clipart, product, real}.
- VLCS [11] contains 10729 images shared by 5 classes and 4 domains {VOC2007, LabelMe, Caltech101, SUN09}.
- TerraIncognita [3] comprises photos of wild animals taken by cameras at different locations. Following [14], we use 4 domains {L100, L38, L43, L46}, which contains 24330 images shared by 10 classes.

These four datasets show different kinds of shift in DG. PACS and OfficeHome are mainly distinguished by images' style by human eyes, while VLCS and TerraIncognita divide domains by different backgrounds, which involves spurious correlation with actual labels. Note that our method stresses exploring the heterogeneity in representation space. The method therefore can be fit for both two kinds of shift because representation is needed for both learning processes, which gains an advantage over only style-based or only causality-based DG methods.

Evaluation metric. We use the training and evaluation protocol presented by DomainBed benchmark [14]. In DG, a domain is chosen as unseen target domain and other domains can be seen as source domains for training. Following the instruction of the benchmark, we split each source domain into 8:2 training/validation splits and integrate the validation subsets of each source domain to create an overall validation set, which is used for validation. The ultimate chosen model is tested on the unseen target domain, and we record the mean and standard deviation of out-of-domain classification accuracies from three different runs with different train-validation splits. For one dataset, we set its every domain as test domain once to record the average accuracy and integrate the prediction accuracy of every domain by a second averaging to stand for the performance.

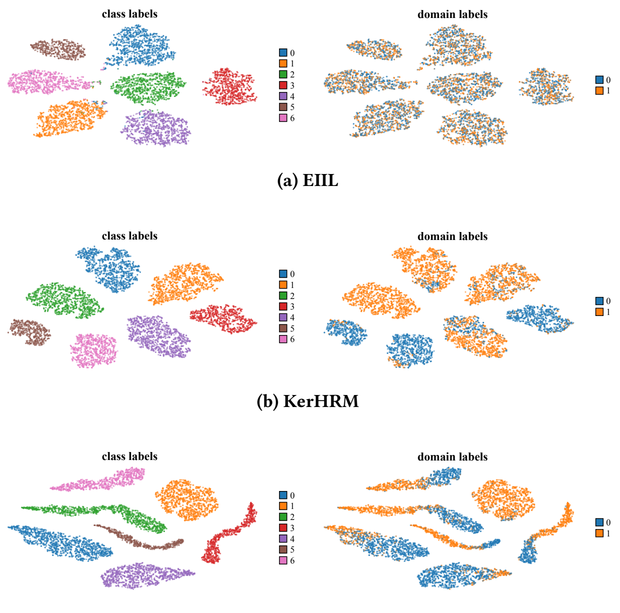
Implementation details. Our method comprises two stages: generating a heterogeneous dividing pattern and training a prediction model. In the first phase, we use ResNet-18 [16] pre-trained on ImageNet [47] as the backbone feature extractor ϕ. We change ResNet-18's last FC layer's output to a low 64-dimension for saving computation time. We additionally set a classifier whose input dimension is 64 to help classification. For generating new dividing patterns, we use a multilayer perceptron (MLP) to divide samples into domains. The MLP has 3 hidden layers with 256 hidden units. Finally, we send the optimal domain labels for all the training samples to the ultimate training stage and the networks trained in heterogeneity generation stage are dropped. As for the hyper-paramters referred in Algorithm 1, we set $T_1 = 5, \lambda_1 = 0.01, \lambda_{cont} = 1, \lambda_{mmd} = 1$ from HTCL. The value of T_2 follows the default value in DomainBed. When training the ultimate model for predicting, we follow DomainBed's strategy. We use ResNet-50 [16] pre-trained on ImageNet [47] as the backbone network for all datasets, and we use an additional FC layer to map features to classes as a classifier. As for model selection, we turn to SWAD [5] for weight averaging.

6.2 Main Results

We first demonstrate our results and compare HTCL with various domain generalization methods in Table 1. Our method achieves the best performance on most of the datasets. The performance of our method exceeds that of CORAL [50] by 1.7% on average. Specifically, the performance gain on TerraIncognita, which is the hardest to predict among all these four dataset, is delightful, reaching 2.3% over SagNet [41] method. All the above comparisons reveal the effect of our method and further demonstrate the improvement brought by seeking and utilizing heterogeneity.

6.3 Comparison with Other Similar Methods

Our method aims for utilizing heterogeneity to help prediction and we implement this goal by generating a heterogeneous dividing pattern, *i.e.*, domain labels. This strategy of re-allocating domain labels is shared by several previous methods. Thus, we compare the performances of these methods under the DomainBed framework. These previous methods have no official version for the DG datasets we used and they mainly report their performances on low-dimension data. So we re-implement them with the help of their code and integrate them into the benchmark framework. As for the hyperparameters, we mainly follow their recommendation and finetune some of them for better performances. The comparison results on the PACS dataset, which is one of the most fashioned DG datasets, are shown in Table 3. We report both their performances on target seen domains (In-domain) and performances on target unseen domains (Out-of-domain). We can find that our method is more suitable for DG tasks than other previous methods. All



(a) EIIL

(b) KerHRM

(c) HTCL (Ours)

Figure 3: T-SNE [52] of the feature representations during generating domain labels in three methods. All these three methods generate dividing patterns with the help of learned feature representations. We obtain the features and compare samples' relative distances with t-SNE. The colors of left column reflect classes while the ones of right denote domains.

these methods enjoy a rather high In-domain accuracy. However, as for out-of-domain performances, our method shows better performances than KerHRM [32] by 4.7%. In addition, our method's standard deviation on testing accuracies is also lower than other methods, which indicates the robustness of HTCL. The results confirm the problem mentioned in section 1: though the novel ideas of these methods are effective in their experiments on synthetic or low-dimension data, the representation in DG datasets is more complex and hard to disentangle. Therefore, some tricks in the feature level will degrade in bigger DG datasets. That's why we design our method with the help of supervision information to obtain a better representation for classification as mentioned in section 4.2.

To further explore these methods' process of dividing domain labels, we use t-SNE [52] to visualize the feature representations of three methods: EIIL, KerHRM, and our HTCL. All these three methods generate new dividing patterns based on the learned features in an interactive module. In other words, the features they learn will influence the process of domain label generation. We obtain the features which decide the final domain labels in these methods and assess their quality. Figure 3 illustrates the scattering results of their features with class labels and domain labels separately by different colors. It can be seen that the features of the three methods all form clusters with class labels. While as for the right figures which are differentiated with domain labels, it can be seen that EIIL tends to separate the same-class samples into different domains too fairly. The domains split by KerHRM follow the strategy to lower intra-domain distances and enlarge inter-domain distances better than EIIL. However, the same-class samples may tend to be divided

Table 2: Summary of the ablation study on PACS. The second row block contains the ablation on the modules of the first stage (generating heterogeneous dividing pattern), while the third row block denotes the ablation on the second stage (predicting with heterogeneous dividing pattern). We record the degradation compared with original method in the last column.

	A	C	P	S	Avg. (Δ)
without generating heterogeneous dividing pattern	89.4 ±0.3	83.0 ±0.5	97.7 ±0.1	82.9 ±1.0	88.2 (-0.4)
replace the guidance of pattern generation with K-Means	90.3 ±1.0	82.6 ±1.0	97.0 ±0.3	83.1 ±0.7	88.2 (-0.4)
without predicting with heterogeneous dividing pattern	90.1 ±0.4	82.5 ±0.2	97.3 ±0.2	83.2 ±1.3	88.3 (-0.3)
HTCL (Original)	89.7 ±0.6	82.9 ±0.4	97.5 ±0.5	84.1 ±0.4	88.6

into one domain in KerHRM, which has the risk of class imbalance among domains. In our method, the data within every manifold is divided into both domains more or less, which indicates the role of our guidance (Equation 10) in generating candidate patterns.

Table 3: The comparison results among the methods which also generate new dividing pattern for DG on PACS [21] dataset. In-domain results denote the testing performance on the target seen domains. Out-of-domain results denote that on the target unseen domain. The standard deviation of testing runs is reported after the average testing accuracies.

	Out-of-domain	In-domain
EIIL [8]	81.7 ±0.6	93.2 ±0.3
IP-IRM [56]	81.7 ±0.4	97.1 ±0.6
KerHRM [32]	83.9 ±2.3	97.5 ±0.1
HTCL (Ours)	**88.6** ±0.3	**97.8** ±0.1

6.4 Ablation Study

Considering that the process of generating dividing patterns has no interaction with ultimate training, it is necessary to split the phases to evaluate our methods. We conduct ablation studies to investigate every module's role in helping enhance generalization. We consider three factors: totally dropping the first stage that generates heterogeneous dividing patterns, replacing the candidate pattern generating guidance (Equation (10)) with simple K-Means cluster algorithm [18] in the first stage, and dropping the contrastive term in the ultimate training stage which aims for utilizing heterogeneity. The results of performances on PACS are listed in Table 2. It confirms that every lack of the module will lower the generalization ability of the model compared to the original HTCL method.

6.4.1 Effects of applying heterogeneous dividing pattern. We first totally drop the stage of re-allocating domain labels. Instead, we use the original domain labels, which turns into the common setting of DomainBed. As shown in Table 2, dropping the procedure of digging heterogeneity lowers the predictive accuracy by 0.4%, which shows the significance of creating domain-heterogeneous dataset in DG.

6.4.2 Effects of our guidance to generate specific candidate patterns. Generating candidate dividing patterns is necessary for learning variant features. In pattern generation module, we design Equation (10) to guide the split of domains. Here we replace this module with a common K-Means cluster algorithm on the given low-dimension representation. As seen in Table 2, our original pattern generation method outperforms the K-Means method by 0.4%. In fact, simply using the K-means algorithm to generate candidate dividing patterns has similar results with totally dropping the first module (both of them obtain 88.2% testing accuracy on PACS).

6.4.3 Effects of the contrastive term for utilizing heterogeneity. As for the ultimate training, we add an extra contrastive term in the loss function to improve model's generalization ability. We conduct the experiments without this term. As shown in Table 2, the average accuracy reduces from 88.6% to 88.3% when dropping this term.

6.4.4 Summary of the ablation study. Above ablation study confirms that every part of HTCL plays a role in helping generalization. There is another observation. In PACS, when setting target unseen domain as C (cartoon) and S (sketch), models' testing accuracies are worse than setting that as A (art) and P (photo), which means samples from C or S are more hard to perform prediction when being set as target domain. Obviously, enhancing the testing accuracies of these domains is more valuable. We note that maintaining the original HTCL framework outperforms in these domains, which indicates our method achieve robustness by pursuing heterogeneity.

7 CONCLUSION

In this work, we comprehensively consider the role of domain labels in the domain generalization (DG) task and explain why domain-heterogeneous datasets can help model obtain better performances on unseen domains. By building a connection between variant features and heterogeneity, we propose a metric for measuring heterogeneity quantitatively with contrastive learning. Besides, we notice that an invariance-aimed contrastive learning performed in the ultimate training will make model better utilize the information brought by heterogeneous domains. Thus, we integrate both digging heterogeneity and utilizing heterogeneity in one framework to help train well-generalized models. We denote this integrated method as heterogeneity-based two-stage contrastive learning (HTCL) for DG. Extensive experimental results show the effectiveness of HTCL on complex DG datasets.

ACKNOWLEDGMENTS

This work was supported in part by National Natural Science Foundation of China (62006207, 62037001, U20A20387), Young Elite Scientists Sponsorship Program by CAST (2021QNRC001), Zhejiang Province Natural Science Foundation (LQ21F020020), Project by Shanghai AI Laboratory (P22KS00111), Program of Zhejiang Province Science and Technology (2022C01044), the StarryNight Science Fund of Zhejiang University Shanghai Institute for Advanced Study (SN-ZJU-SIAS-0010), and the Fundamental Research Funds for the Central Universities (226-2022-00142, 226-2022-00051).

REFERENCES

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

[2] Haoyue Bai, Rui Sun, Lanqing Hong, Fengwei Zhou, Nanyang Ye, Han-Jia Ye, S-H Gary Chan, and Zhenguo Li. 2021. Decaug: Out-of-distribution generalization via decomposed feature representation and semantic augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 6705–6713.

[3] Sara Beery, Grant Van Horn, and Pietro Perona. 2018. Recognition in terra incognita. In *Proceedings of the European conference on computer vision (ECCV)*. 456–473.

[4] Gilles Blanchard, Aniket Anand Deshmukh, Urun Dogan, Gyemin Lee, and Clayton Scott. 2021. Domain Generalization by Marginal Transfer Learning. *Journal of Machine Learning Research* 22, 2 (2021), 1–55.

[5] Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. 2021. SWAD: Domain Generalization by Seeking Flat Minima. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*. PMLR, 1597–1607.

[7] Mathieu Chevalley, Charlotte Bunne, Andreas Krause, and Stefan Bauer. 2022. Invariant causal mechanisms through distribution matching. *arXiv preprint arXiv:2206.11646* (2022).

[8] Elliot Creager, Jörn-Henrik Jacobsen, and Richard Zemel. 2021. Environment Inference for Invariant Learning. In *International Conference on Machine Learning*.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[10] Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Jacob Steinhardt, and Aleksander Madry. 2020. Identifying statistical bias in dataset replication. In *International Conference on Machine Learning*. PMLR, 2922–2932.

[11] 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.

[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. *Journal of machine learning research* 17, 1 (2016), 2096–2030.

[13] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence* 2, 11 (2020), 665–673.

[14] Ishaan Gulrajani and David Lopez-Paz. 2021. In Search of Lost Domain Generalization. In *International Conference on Learning Representations*.

[15] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 9729–9738.

[16] 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.

[17] Zeyi Huang, Haohan Wang, Eric P Xing, and Dong Huang. 2020. Self-challenging improves cross-domain generalization. *European Conference on Computer Vision* 2 (2020).

[18] K. Krishna and M. Narasimha Murty. 1999. Genetic K-means algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 29, 3 (1999), 433–439. https://doi.org/10.1109/3477.764879

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

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

[21] 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.

[22] 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.

[23] Mengze Li, Han Wang, Wenqiao Zhang, Jiaxu Miao, Wei Ji, Zhou Zhao, Shengyu Zhang, and Fei Wu. 2023. WINNER: Weakly-supervised hIerarchical decompositioN and aligNment for spatio-tEmporal video gRounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

[24] Mengze Li, Tianbao Wang, Jiahe Xu, Kairong Han, Shengyu Zhang, Zhou Zhao, Jiaxu Miao, Wenqiao Zhang, Shiliang Pu, and Fei Wu. 2023. Multi-modal Action Chain Abductive Reasoning. In *Proceedings of the 61th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

[25] Mengze Li, Tianbao Wang, Haoyu Zhang, Shengyu Zhang, Zhou Zhao, Wenqiao Zhang, Jiaxu Miao, Shiliang Pu, and Fei Wu. 2022. Hero: Hierarchical spatio-temporal reasoning with contrastive action correspondence for end-to-end video object grounding. In *Proceedings of the 30th ACM International Conference on Multimedia*. 3801–3810.

[26] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems* 2 (2020), 429–450.

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

[28] Yufan Liao, Qi Wu, and Xing Yan. 2022. Decorr: Environment Partitioning for Invariant Learning and OOD Generalization. *arXiv preprint arXiv:2211.10054* (2022).

[29] Yong Lin, Hanze Dong, Hao Wang, and Tong Zhang. 2022. Bayesian invariant risk minimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 16021–16030.

[30] Yong Lin, Shengyu Zhu, Lu Tan, and Peng Cui. 2022. ZIN: When and How to Learn Invariance Without Environment Partition? *Advances in Neural Information Processing Systems* 35 (2022), 24529–24542.

[31] Jiashuo Liu, Zheyuan Hu, Peng Cui, Bo Li, and Zheyan Shen. 2021. Heterogeneous risk minimization. In *International Conference on Machine Learning*. PMLR, 6804–6814.

[32] Jiashuo Liu, Zheyuan Hu, Peng Cui, Bo Li, and Zheyan Shen. 2021. Kernelized heterogeneous risk minimization. *arXiv preprint arXiv:2110.12425* (2021).

[33] Jiashuo Liu, Jiayun Wu, Renjie Pi, Renzhe Xu, Xingxuan Zhang, Bo Li, and Peng Cui. 2023. Measure the Predictive Heterogeneity. In *International Conference on Learning Representations*. https://openreview.net/forum?id=g2oB_k-18b

[34] Fangrui Lv, Jian Liang, Shuang Li, Bin Zang, Chi Harold Liu, Ziteng Wang, and Di Liu. 2022. Causality Inspired Representation Learning for Domain Generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8046–8056.

[35] Zheqi Lv, Zhengyu Chen, Shengyu Zhang, Kun Kuang, Wenqiao Zhang, Mengze Li, Beng Chin Ooi, and Fei Wu. 2023. IDEAL: Toward High-efficiency Device-Cloud Collaborative and Dynamic Recommendation System. *arXiv preprint arXiv:2302.07335* (2023).

[36] Zheqi Lv, Feng Wang, Shengyu Zhang, Kun Kuang, Hongxia Yang, and Fei Wu. 2022. Personalizing Intervened Network for Long-tailed Sequential User Behavior Modeling. *arXiv preprint arXiv:2208.09130* (2022).

[37] Zheqi Lv, Wenqiao Zhang, Shengyu Zhang, Kun Kuang, Feng Wang, Yongwei Wang, Zhengyu Chen, Tao Shen, Hongxia Yang, Beng Chin Ooi, and Fei Wu. 2023. DUET: A Tuning-Free Device-Cloud Collaborative Parameters Generation Framework for Efficient Device Model Generalization. In *Proceedings of the ACM Web Conference 2023*.

[38] Divyat Mahajan, Shruti Tople, and Amit Sharma. 2021. Domain generalization using causal matching. In *International Conference on Machine Learning*. PMLR, 7313–7324.

[39] Qiaowei Miao, Junkun Yuan, and Kun Kuang. 2022. Domain Generalization via Contrastive Causal Learning. *arXiv preprint arXiv:2210.02655* (2022).

[40] Milton Llera Montero, Casimir JH Ludwig, Rui Ponte Costa, Gaurav Malhotra, and Jeffrey Bowers. 2020. The role of disentanglement in generalisation. In *International Conference on Learning Representations*.

[41] Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. 2021. Reducing Domain Gap by Reducing Style Bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8690–8699.

[42] Ziwei Niu, Junkun Yuan, Xu Ma, Yingying Xu, Jing Liu, Yen-Wei Chen, Ruofeng Tong, and Lanfen Lin. 2023. Knowledge Distillation-based Domain-invariant Representation Learning for Domain Generalization. *IEEE Transactions on Multimedia* (2023).

[43] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748* (2018).

[44] Vihari Piratla, Praneeth Netrapalli, and Sunita Sarawagi. 2020. Efficient domain generalization via common-specific low-rank decomposition. In *International Conference on Machine Learning*. PMLR, 7728–7738.

[45] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. 2019. Do imagenet classifiers generalize to imagenet?. In *International Conference on Machine Learning*. PMLR, 5389–5400.

[46] Yangjun Ruan, Yann Dubois, and Chris J Maddison. 2021. Optimal representations for covariate shift. *arXiv preprint arXiv:2201.00057* (2021).

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

[48] Shiori Sagawa*, Pang Wei Koh*, Tatsunori B. Hashimoto, and Percy Liang. 2020. Distributionally Robust Neural Networks. In *International Conference on Learning Representations*.

[49] Yuge Shi, Jeffrey Seely, Philip Torr, Siddharth N, Awni Hannun, Nicolas Usunier, and Gabriel Synnaeve. 2022. Gradient Matching for Domain Generalization. In *International Conference on Learning Representations*.

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

[51] Christopher Tran and Elena Zheleva. 2022. Improving Data-driven Heterogeneous Treatment Effect Estimation Under Structure Uncertainty. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1787–1797.

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

[53] Vladimir N Vapnik. 1999. An overview of statistical learning theory. *IEEE transactions on neural networks* 10, 5 (1999), 988–999.

[54] 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.

[55] Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John C Duchi, Vittorio Murino, and Silvio Savarese. 2018. Generalizing to unseen domains via adversarial data augmentation. *Advances in neural information processing systems* 31 (2018).

[56] Tan Wang, Zhongqi Yue, Jianqiang Huang, Qianru Sun, and Hanwang Zhang. 2021. Self-supervised learning disentangled group representation as feature. *Advances in Neural Information Processing Systems* 34 (2021), 18225–18240.

[57] Anpeng Wu, Junkun Yuan, Kun Kuang, Bo Li, Runze Wu, Qiang Zhu, Yueting Zhuang, and Fei Wu. 2023. Learning Decomposed Representations for Treatment Effect Estimation. *IEEE Transactions on Knowledge and Data Engineering* 35, 5 (2023), 4989–5001. https://doi.org/10.1109/TKDE.2022.3150807

[58] Jun Wu and Jingrui He. 2021. Indirect Invisible Poisoning Attacks on Domain Adaptation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 1852–1862.

[59] Qinwei Xu, Ruipeng Zhang, Ya Zhang, Yanfeng Wang, and Qi Tian. 2021. A fourier-based framework for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14383–14392.

[60] Shen Yan, Huan Song, Nanxiang Li, Lincan Zou, and Liu Ren. 2020. Improve unsupervised domain adaptation with mixup training. *arXiv preprint arXiv:2001.00677* (2020).

[61] Junkun Yuan, Xu Ma, Defang Chen, Kun Kuang, Fei Wu, and Lanfen Lin. 2022. Label-Efficient Domain Generalization via Collaborative Exploration and Generalization. In *Proceedings of the 30th ACM International Conference on Multimedia*. 2361–2370.

[62] Junkun Yuan, Xu Ma, Defang Chen, Kun Kuang, Fei Wu, and Lanfen Lin. 2023. Domain-specific bias filtering for single labeled domain generalization. *International Journal of Computer Vision* 131, 2 (2023), 552–571.

[63] Junkun Yuan, Xu Ma, Ruoxuan Xiong, Mingming Gong, Xiangyu Liu, Fei Wu, Lanfen Lin, and Kun Kuang. 2023. Instrumental Variable-Driven Domain Generalization with Unobserved Confounders. *ACM Transactions on Knowledge Discovery from Data* (2023).

[64] Fengda Zhang, Kun Kuang, Long Chen, Yuxuan Liu, Chao Wu, and Jun Xiao. 2023. Fairness-aware Contrastive Learning with Partially Annotated Sensitive Attributes. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=woa783QMul

[65] Hanlin Zhang, Yi-Fan Zhang, Weiyang Liu, Adrian Weller, Bernhard Schölkopf, and Eric P Xing. 2022. Towards principled disentanglement for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8024–8034.

[66] Min Zhang, Siteng Huang, Wenbin Li, and Donglin Wang. 2022. Tree Structure-Aware Few-Shot Image Classification via Hierarchical Aggregation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XX*. Springer, 453–470.

[67] 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).

[68] Shengyu Zhang, Dong Yao, Zhou Zhao, Tat-Seng Chua, and Fei Wu. 2021. Causerec: Counterfactual user sequence synthesis for sequential recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 367–377.

[69] Fan Zhou, Zhuqing Jiang, Changjian Shui, Boyu Wang, and Brahim Chaib-draa. 2020. Domain generalization with optimal transport and metric learning. *arXiv preprint arXiv:2007.10573* (2020).

[70] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. 2021. Domain Generalization with MixStyle. In *International Conference on Learning Representations*.

[71] Xiao Zhou, Yong Lin, Renjie Pi, Weizhong Zhang, Renzhe Xu, Peng Cui, and Tong Zhang. 2022. Model agnostic sample reweighting for out-of-distribution learning. In *International Conference on Machine Learning*. PMLR, 27203–27221.

[72] Xiao Zhou, Yong Lin, Weizhong Zhang, and Tong Zhang. 2022. Sparse invariant risk minimization. In *International Conference on Machine Learning*. PMLR, 27222–27244.

[73] Didi Zhu, Yincuan Li, Junkun Yuan, Zexi Li, Yunfeng Shao, Kun Kuang, and Chao Wu. 2023. Universal Domain Adaptation via Compressive Attention Matching. *arXiv preprint arXiv:2304.11862* (2023).

[74] Zhao Ziyu, Kun Kuang, Bo Li, Peng Cui, Runze Wu, Jun Xiao, and Fei Wu. 2023. Differentiated matching for individual and average treatment effect estimation. *Data Mining and Knowledge Discovery* 37, 1 (2023), 205–227.

A THE EFFECTIVENESS OF HETEROGENEOUS DIVIDING PATTERN

Since generating heterogeneous dividing pattern does not require any modification on ultimate training and model architecture, we combine this procedure with other four methods to verify the effectiveness of heterogeneous domain labels. These methods (GroupDRO [48], CORAL [50], MMD [22], MTL [4]) need to distinguish different source domains during their training procedures. We generate new heterogeneous domain labels by the first stage of HTCL. Then we apply these labels to those DG methods and don't change their original backbones. The comparison is shown in Table 4. By applying the procedure, all methods' performances are improved. The gain comes to at least 2%, which shows the huge potential brought by generating heterogeneous dividing pattern.

Table 4: The results of adding our newly generated dividing pattern to common DG methods on PACS [21] dataset. The heterogeneous domain labels are generated by HTCL's first stage. We then apply them on common DG methods. The gain of accuracies is reported in the last column.

	Original	With new domain labels	Δ
GroupDRO [48]	84.4 ± 0.8	88.1 ± 0.2	+3.7
CORAL [50]	86.2 ± 0.3	88.3 ± 0.4	+2.1
MMD [22]	84.7 ± 0.5	87.4 ± 0.1	+2.7
MTL [4]	84.6 ± 0.5	88.4 ± 0.4	+3.8

B SENSITIVITY ANALYSIS

In this subsection, we study the model sensitivity with respect to the hyper-parameters referred in Algorithm (1). Table 5 demonstrates the comparison results. The changes of all hyper-parameters don't affect the performances too much and they can all surpass current baselines. As for λ_{mmd} and T_2 in Algorithm (1), we set them to fixed 0.5 and 5000 (iterations) respectively as DomainBed [14] pre-defined to achieve a fair comparison. The visualization for the sensitivity analysis is in Figure 4. To demonstrate the results of λ_1 intuitively, we take the logarithm of its selected values and then map them on the x-axis. It can be seen that our method is robust to the value change of all hyper-parameters.

Table 5: The sensitivity analysis for T_1, λ_1 and λ_{cont} on PACS [21] dataset.

T_1	1	3	5 (default)	7	9
Test Acc. (%)	88.2	87.7	88.6	87.8	88.5
λ_1	0	1e-3	1e-2 (default)	1e-1	1
Test Acc. (%)	87.7	87.3	88.6	88.1	87.7
λ_{cont}	0.5	1.0 (default)	1.5	2.0	2.5
Test Acc. (%)	87.5	88.6	87.2	87.6	87.4

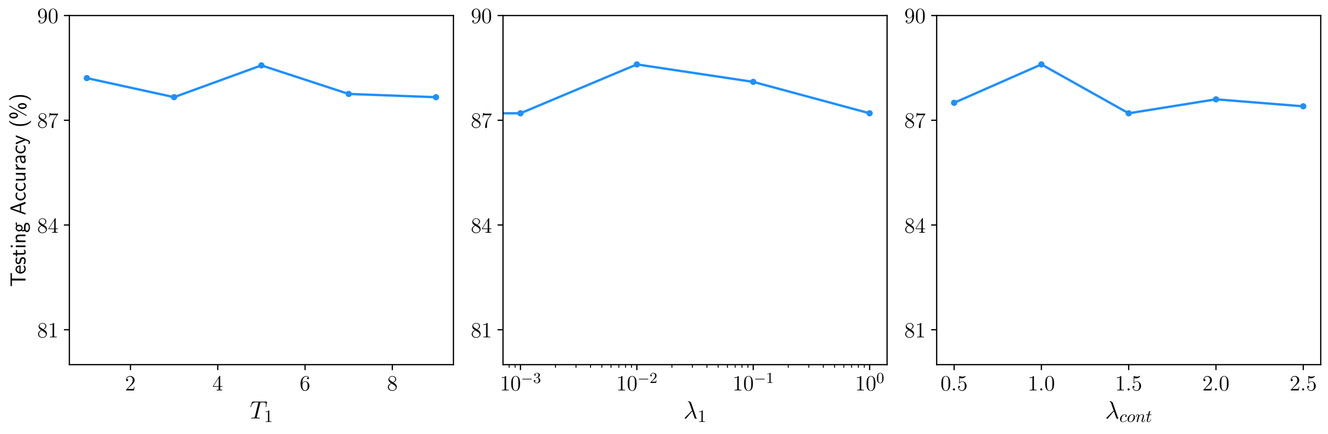


Figure 4: Results of the sensitivity analysis with respect to different hyper-parameters.

C FURTHER DISCUSSION ON THE MAIN RESULTS

Here we further analyze the main results in Table 1. As Table 1 illustrates, HTCL doesn't show the superior performance on VLCS [11] dataset. We think it is the original data heterogeneity that limits the performance of our method in specific datasets. Different from other three datasets, data of each domain of VLCS are collected from a specific dataset, which indirectly increases the original heterogeneity of the whole dataset. In the first stage of HTCL, we aim to replace original dividing pattern with our newly generated heterogeneous pattern. However, when the given pattern is already heterogeneous enough like VLCS[11], The performance gain brought by heterogeneous dividing pattern generation will naturally be reduced. Therefore, considering the influence of original data heterogeneity before applying our method may be our future improvement.