Neuron Coverage-Guided Domain Generalization

Chris Xing Tian, Haoliang Li, Xiaofei Xie, Yang Liu, and Shiqi Wang

Abstract—This paper focuses on the domain generalization task where domain knowledge is unavailable, and even worse, only samples from a single domain can be utilized during training. Our motivation originates from the recent progresses in deep neural network (DNN) testing, which has shown that maximizing neuron coverage of DNN can help to explore possible defects of DNN (i.e., misclassification). More specifically, by treating the DNN as a program and each neuron as a functional point of the code, during the network training we aim to improve the generalization capability by maximizing the neuron coverage of DNN with the gradient similarity regularization between the original and augmented samples. As such, the decision behavior of the DNN is optimized, avoiding the arbitrary neurons that are deleterious for the unseen samples, and leading to the trained DNN that can be better generalized to out-of-distribution samples. Extensive studies on various domain generalization tasks based on both single and multiple domain(s) setting demonstrate the effectiveness of our proposed approach compared with state-of-the-art baseline methods. We also analyze our method by conducting visualization based on network dissection. The results further provide useful evidence on the rationality and effectiveness of our approach.

Index Terms—Out-of-distribution, neuron coverage, gradient similarity

1 Introduction

It has been well-known that training a deep neural network (DNN) model with desirable generalization ability usually requires abundant labeled data [1], [2], and similar to other supervised learning models, the trained deep model may fail to generalize well on the out-of-distribution samples which are different from the domains where it is trained [3]. To overcome these limitations, transfer learning, which aims to transfer knowledge from some source domain(s) to boost the generalization performance of a learning model on a domain of interest (i.e., a target domain), has been proposed [4].

- *C.X.Tian and S.Wang are with the Department of Computer Science, City University of Hong Kong, Hong Kong SAR, China*
- *H. Li is with the Department of Electrical Engineering, City University of Hong Kong, Hong Kong SAR, China.*
- *X. Xie is with the School of Computing and Information Systems, Singapore Management University, Singapore*
- *Y. Liu are with the School of Computer Science and Engineering, Nanyang Technological University, Singapore.*
- *Corresponding authors: Shiqi Wang (e-mail: shiqwang@cityu.edu.hk)*

- *This work is supported in part by the National Natural Science Foundation of China under 62022002, in part by the Hong Kong Research Grants Council, Early Career Scheme (RGC ECS) under Grant 21211018, General Research Fund (GRF) under Grant 11203220.*
- *This work of H. Li was supported by CityU New Research Initiatives/Infrastructure Support from Central under the grant APRC 9610528.*
- *This work of Y. Liu is partially supported by the National Research Foundation, Prime Ministers Office, Singapore under its National Cybersecurity R&D Program (Award No. NRF2018NCR-NCR005-0001) and the NRF Investigatorship NRFI06-2020-0022-0001.*

However, in many applications, it may not be feasible to acquire the target domain data in advance. Domain generalization [5] is a promising technique which gains knowledge acquired from different domains, and applies it to previously unseen but related domains. Generally speaking, current research regarding domain generalization can be categorized into two streams. The first stream (e.g., [5]) aims at learning to extract a universal feature representation among domains through either distribution alignment or multi-task learning. The other stream (e.g., [6]) leverages the advantage of meta-learning methods for feature representation learning which was originally proposed for few-shot learning problem. During training, meta-train set and meta-test set are selected from source domains during each iteration to produce a more robust feature representation.

One major limitation of aforementioned methods lies in that the prior knowledge of domain information must be acquired in advance, such that one can leverage covariate shift [7] to learn shareable information among domains. In practice, the training data can be quite complicated that there may not be known and clear distinction among the domains, thus, the domain of each sample is ambiguous to define. Even worse, due to the privacy issue, one may only have data collected from one single domain, such that it is impractical to simulate covariate shift with large domain gap in this case.

To tackle the aforementioned problem, there exists works focusing on the worst-case formulation of domain generalization that the samples from only one single domain can be utilized during training stage. The main idea is conducting data augmentation to improve the generalization capability of DNN. In [8], an adversarial training mechanism was proposed by forcing the latent features of the original and augmented data to be lying on a similar manifold. Such mechanism has been further extended in [9], where a Wasserstein

distance constraint [10] was introduced to encourage out-of-distribution augmentation and a meta-learning method was adopted to learn shareable information between the original and augmented data. The goal of these works is to guarantee that the original data and its augmented data have the similar prediction/decision by the DNN. However, how to understand the misclassification behavior of an out-of-distribution sample and how to measure the similarity of the decision on two inputs are still open problems.

The coverage analysis has been recognized as the key driver in software testing, ensuring no bugs triggered in the high-coverage tests. Software bugs usually trigger the abnormal behaviors that cover the special control/data flow of the programs. In the traditional code testing, software bugs can be regarded as the out-of-distribution inputs which have different coverage compared with normal inputs. Inspired by this, the neuron coverage was further proposed to detect out-of-distribution samples that may have different coverage, which leads to the misclassification behavior of DNNs [11], [12]. The undetected bugs (i.e., misclassification) of DNN can be caused by the neurons which are inactivated during training stage but can be activated by the out-of-distribution samples during testing stage [11]. Thus, in order to improve the generalization capability of the DNN, we aim to maximize the coverage such that more neurons can be activated during the training phase. Moreover, in analogous to the traditional software, the out-of-distribution samples and the in-distribution samples should trigger the similar control flow or data flow of the DNN if they have similar semantic information or logic (i.e., same label information for recognition task). To this end, we propose to improve the generalization capability of DNNs by *maximizing the neuron coverage* of DNN with *gradient similarity regularization* between the original and the augmented samples, where neuron coverage and gradient can represent the control flow and data flow of DNN, respectively. We expect the trained DNN to be better generalized to out-of-distribution samples from unseen but related domains. Experimental results on domain generalization setting where domain knowledge is not available during the training stage (including the worst-case formulation where only one single domain is available) demonstrate the effectiveness of our proposed method. Last but not least, we also attempt to bridge the gap between the software testing and computer vision by providing analysis based on network visualization through network dissection [13]. The visualization results further justify the rationality and the effectiveness of our proposed method.

2 Related Works

2.1 Domain Generalization

The goal of domain generalization is to improve the generalization capability of the trained model towards the out-of-distribution samples from unseen but related domains. One main research direction of domain generalization is to learn shareable representations from samples of different domains. For example, Muandet *et al.* [5] proposed the Domain Invariant Component Analysis (DICA) algorithm based on multiple source-domain data. As such, the distribution mismatch across domains is minimized while the conditional function relationship is preserved. Ghifary *et al.* [14] proposed a multi-task autoencoder framework to learn domain invariant features by reconstructing the data from one domain to another. Motiian *et al.* [15] proposed to minimize the semantic alignment loss as well as the separation loss for domain generalization. Li *et al.* [16] proposed to learn CNN model through low-rank regularization. Carlucci *et al.* [17] proposed to leverage the advantage of self-supervised learning to learn generalized feature representation. Wang *et al.* [18] proposed to learn robust feature representation through statistics out by removing the grey-level co-occurrence information. More recently, Zhou *et al.* [19] proposed to conduct image generation across domains. Huang *et al.* [20] proposed a "dropout on gradient" mechanism for domain generalization.

Another direction is to conduct meta-learning (a.k.a. learning to learn) by simulating domain shift for meta-train and meta-test set to tackle the problem of domain generalization. In [6], the idea of meta learning, which was originally proposed for few-shot learning problem, was extended to the "unseen" domain setting. Balaji *et al.* [21] proposed an episodic training procedure by considering regularization based on domain specific network. Such idea was further extended with either advanced training mechanism [22], novel network regularization [23] or additional feature embedding loss (e.g., triplet loss) [24].

While the aforementioned techniques focusing on multiple domain setting, recently, the worst-case formulation of domain generalization that only single domain training data can be utilized has also drawn more and more attentions. In [8], an adversarial training mechanism was proposed with the regularization that augmented data and the original data lie on a similar manifold in terms of the semantic feature space. In [9], a joint data augmentation method was proposed based on adversarial training mechanism for within-domain data augmentation. Moreover, the Wasserstein autoencoder for out-of-manifold data augmentation has been presented in [10]. Self-supervised learning has also been proved to be effective in this case. For example, in [17], the authors have shown that better generalization capability can be achieved based on single domain scenario by performing the evaluations in the digit recognition task.

2.2 Code Coverage and Neuron Coverage

In traditional program, control-flow graph (CFG) is a representation of all paths that might be executed. Each node of the CFG is a basic block including a sequence of statements. The edge represents the jump (e.g, the if-else branch) from one basic block to another one. Data flow analysis aims to gather information regarding the possible set of values calculated at various points of the CFG. To test the programs, several coverage criteria (e.g., statement coverage, branch coverage) have been proposed based on the CFG. Motivated by the structure of traditional programs, Pei *et al.* [11] proposed the neuron coverage that measures the percentage of activated neurons with given input set. Furthermore, Ma *et al.* [12] then extended neuron coverage and proposed a set of more fine-grained neuron-based coverage criteria considering the distribution of the neuron outputs from training data. Unlike the aforementioned techniques which applied neuron coverage during the DNN testing stage, we

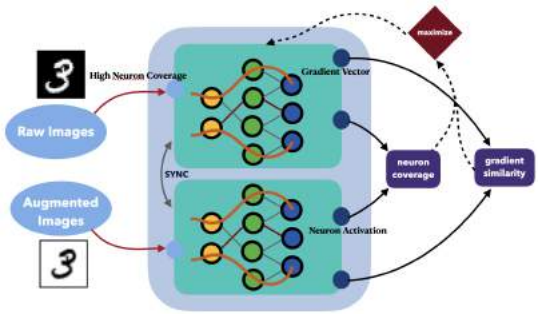


Fig. 1. Our proposed framework for domain generalization. Given the two different samples with similar semantics, we propose to maximize the neuron coverage of DNN with gradient similarity regularization between two samples with similar semantic information.

propose to maximize neuron coverage during the training stage for improving the generalization capability to unseen domains.

3 METHODOLOGY

3.1 Preliminary and Overview

We denote the source domain(s) on a joint space $\mathcal{X} \times \mathcal{Y}$ as $\mathcal{D}_S = \{\mathbf{x}_l^S, y_l^S\}_{l=1}^{N_S}$ with N_S labeled samples in total. We aim to learn a DNN model f which is parameterized by Θ with $\mathcal{D}_S$ only, and perform the classification task on the samples $\mathbf{x}^T$ from a related domain $\mathcal{D}_T$ without having any prior observations or knowledge about $\mathcal{D}_T$.

We provide a framework called Neuron Coverage-guided Domain Generalization (NCDG) to improve the generalization of DNN model f. Inspired by the CFG and data flow graph (DFG) in traditional programs, we propose novel loss functions, i.e, the *neuron coverage maximization loss* with the *gradient similarity loss* regularization. By optimizing the proposed loss function on inputs from two similar domains (i.e., the original source domain $\mathcal{D}_S$ and its augmented domain $\hat{\mathcal{D}}_S$), we expect the DNN to be better generalized to out-of-distribution samples. Herein, we introduce the two loss functions as well as the training procedure. The whole framework is summarized in Fig. 1.

3.2 Neuron Coverage Maximization

Existing works [11], [12] have demonstrated that the out-of-distribution samples exhibit different neuron coverage with the original data (i.e., different neurons are activated). To mitigate this issue, we propose to maximize the neuron coverage of the original data and the augmented data based on the following inspirations. First, maximizing the neuron coverage can improve the prediction stability by reducing the possibility that the out-of-distribution samples activate different neurons with the training data. We expect that the undetected bugs (i.e., misclassification) caused by inactive neurons, which are triggered by out-of-distribution samples during testing stage, can be reduced. Second, by maximizing the neuron coverage, we can further expect more overlapped activated neurons between the original data and the augmented data. As such, the neuron activation similarity between the original data and the augmented data is increased (i.e., the "control flow" is similar).

We denote n_i^j as the j-th neuron in the i-th layer of DNN f, where the output of n_i^j is denoted as $out(\mathbf{x}, n_i^j)$ given input $\mathbf{x}$. We further conduct max-min normalization of $out(\mathbf{x}, n_i^j)$ given the neuron outputs from the same layer $\mathcal{O}(\mathbf{x}, i) = \{out(\mathbf{x}, n_i^1), out(\mathbf{x}, n_i^2), ..., out(\mathbf{x}, n_i^{N_i})\}$, where N_i is the total number of neurons in the i-th layer. The normalized output of $out(\mathbf{x}, n_i^j)$ is represented as

$$\tilde{out}(\mathbf{x}, n_i^j) = \frac{out(\mathbf{x}, n_i^j) - min(\mathcal{O}(\mathbf{x}, i))}{max(\mathcal{O}(\mathbf{x}, i)) - min(\mathcal{O}(\mathbf{x}, i))}. \quad (1)$$

As such, the neuron n_i^j is considered to be activated only if $\tilde{out}(\mathbf{x}, n_i^j)$ is larger than a pre-defined threshold t.

Algorithm 1 The Computation of Neuron Coverage Loss

Input:
$\lambda \rightarrow$ parameter for neuron coverage;
$t \rightarrow$ threshold for neuron activation;
model $\rightarrow$ model for training;
neuron_act_map $\rightarrow$ records the neurons which have been ever activated, and it is initialized as empty map before the epoch.
Output:
neu_cov_loss $\rightarrow$ neuron coverage loss for current ended iteration, to be maximized in following iteration;

```
/* one training iteration starts */
neu_cov_loss := 0
for layer in model.layers do
    inactive_neurons = {}
    if layer not activation layer then
        continue
    end if
    max = maximal output value in layer.neurons
    min = minimal output value in layer.neurons
    for n in layer.neurons do
        n_val_scale = (n.output() - min) / (max - min)
        if neuron_act_map(n) or n_val_scale >t then
            neuron_act_map(n) = true
        else
            inactive_neurons.put(n, n_val_scale)
        end if
    end for
    neu_cov_loss += sum(inactive_neurons.outputs())
end for
return neu_cov_loss /*to be incorporated with L_c */
```

The neuron coverage is defined based on the proportion of neurons which have been activated given a batch of data [11], [12]. Moreover, the neuron is considered to be activated if there exists a sample from the batch where the corresponding output is larger than the threshold t. In analogy to the implication of low code coverage in software testing, low neuron coverage of a DNN may suggest that the incorrect DNN behaviors (i.e., misclassification) remains unexplored. Thus, in this scenario the DNN tends to be lack of generalization capability as errors can be induced once the inactivated neurons get activated during the test phase on target domains.

Directly maximizing the neuron coverage of DNN based on a batch of data may not be computationally feasible due to the *logical reasoning* involved [25]. To this end, we first

propose to relax the optimization by maximizing the average output of neuron given a batch together with the minimizing the standard classification loss (e.g., cross-entropy loss) $\mathcal{L}_c(f, \mathcal{D})$ to improve the generalization capability of DNN, which can be represented as

$$\mathcal{L}_{cov}(f, \mathcal{D}) = \mathcal{L}_c(f, \mathcal{D}) - \lambda \mathbb{E}_{\mathbf{x} \in \mathcal{D}} \sum_i \sum_j \tilde{out}(\mathbf{x}, n_i^j; t), \quad (2)$$

where $\mathcal{D}$ denotes the domain for training, λ is the parameter for neuron activation loss, and t is the threshold for neuron activation.

We further propose a bootstrapping-like mechanism to optimize Eq. 2 by exploring the *inactive neurons* for more efficient training. More specifically, we first initialize the inactivated set of the i-th layer $\mathcal{IS}_i$ to contain all neurons from the output of this layer at the beginning of a training epoch. We then update the $\mathcal{IS}_i$ by removing the neurons which have been activated during optimization (i.e., the output of neuron is higher than the threshold t). Thus, in each iteration, we only consider the neurons in $\mathcal{IS}_i$ to compute average output of neurons instead of adopting all neurons from the i-th layer. We repeat the process until the end of the epoch, and reset all neurons to be inactivated again at the beginning of the next epoch accordingly. The details of computing the neuron coverage loss are summarized in Algorithm 1.

While the neuron coverage can be used to measure the decision similarity, it still cannot model the data flow through skip connections over certain layers (e.g., residue connection module [26]). We propose to adapt the *gradient information* to represent the data flow of the DNN, as it can track the neuron output and the information flow across the shortcut [27]. More specifically, we use gradient of neuron coverage loss (i.e., standard classification loss with neuron activation maximization) for regularization purpose, which is denoted as $\frac{\partial L_{cov}(f, \mathcal{D}_S)}{\partial \Theta}$ and can be obtained by optimizing $\mathcal{L}_{cov}$ through backpropagation. Based on gradient information, we can measure the similarity between the original data and the augmented data in a finer-grained way.

Given the original source samples as $\mathcal{D}_S = \{\mathbf{x}_i^S, y_i^S\}_{i=1}^{N_S}$, we first augment the data through data augmentation techniques to obtain $\hat{\mathcal{D}}_S$ (noted that $\hat{\mathcal{D}}_S$ and $\mathcal{D}_S$ are one-one correspondence). Subsequently, at each iteration, we train the model on the minibatch of $\mathcal{D}_S$ and $\hat{\mathcal{D}}_S$ separately with $\mathcal{L}_{cov}$ to obtain the gradients $\frac{\partial L_{cov}(f, \mathcal{D}_S)}{\partial \Theta}$ and $\frac{\partial L_{cov}(f, \hat{\mathcal{D}}_S)}{\partial \Theta}$, respectively. We model the gradient similarity between $\mathcal{D}_S$ and its augmented version $\hat{\mathcal{D}}_S$ based on L2 norm as

$$\mathcal{L}_{sim}(f) = \left\| \frac{\partial L_{cov}(f, \mathcal{D}_S)}{\partial \Theta} - \frac{\partial L_{cov}(f, \hat{\mathcal{D}}_S)}{\partial \Theta} \right\|_2, \quad (3)$$

where the update of Eq. 3 involves a gradient through a gradient and requires an additional backward pass through the model [28].

Discussion: It is worth noting that our proposed neuron coverage is computed based on the whole training dataset, which aims to increase the number of activated neurons (i.e., which have been activated by at least one sample from the training set). Our proposed method also does not contradict with dropout, as dropout is randomly applied on a batch (or a single sample). Our proposed gradient similarity regularization term is close to the contrastive learning (e.g., [29], [30]) which aims to learn a representation by maximizing similarity and dissimilarity of data samples organized into similar and dissimilar pairs. Unlike these methods which map the data into a single (or multiple) embedding space(s) where contrastive learning is performed, our proposed method is based on the neuron coverage with similarity regularization by mapping the data into neural tangent kernel space [31] (i.e., mapping the data from the original space to the kernel space represented by the gradient corresponding to the loss function). Our gradient similarity regularization is also different from [6]. In [6], the gradient regularization is conducted from the perspective of meta-learning through first-order Taylor expansion based on average gradient through gradient descent, which requires the "direction of improvement in each set of domains is similar". However, our proposed method is motivated by the data flow similarity, which requires gradient regularization in a one-to-one correspondence manner. Last, while one may argue that removing inactivated neurons through network pruning can avoid unexpected activation, it has been shown in [32] that network pruning can even worsen the performance under cross-domain setting.

Algorithm 2 Training Procedure of NCDG

Input:
$\mathcal{D}_S \rightarrow$ original training samples batch;
$\hat{\mathcal{D}}_S \rightarrow$ augmented training samples batch, one-one correspondence with $\mathcal{D}_S$;
$\beta \rightarrow$ parameter balancing coupled neuron coverage loss and gradient similarity loss;
$\lambda, t \rightarrow$ parameters for neuron coverage loss

/* *main procedure starts* */
for ite **in** iterations **do**
  Raw-train: Gradients $\nabla_\Theta = \frac{\partial \mathcal{L}_{cov}(f, \mathcal{D}_S)}{\partial \Theta}$
  Augmented-train: Gradients $\hat{\nabla}_\Theta = \frac{\partial \mathcal{L}_{cov}(f, \hat{\mathcal{D}}_S)}{\partial \Theta}$
  optimization: Update Θ:
  $\Theta = \Theta - \eta \cdot \frac{\partial \left(\mathcal{L}_{cov}(f, \mathcal{D}_S) + \mathcal{L}_{cov}(f, \hat{\mathcal{D}}_S) + \beta \cdot \left\| \nabla_\Theta - \hat{\nabla}_\Theta \right\|_2 \right)}{\partial \Theta}$
end for
end procedure

3.3 Model Training

In this subsection, we summarize the proposed method NCDG. In particular, the neuron coverage losses for $\mathcal{D}_S$ and $\hat{\mathcal{D}}_S$ are optimized, containing a standard classification loss with a neuron activation maximization regularization term. Moreover, a gradient similarity loss based on gradient between $\mathcal{D}_S$ and $\hat{\mathcal{D}}_S$ is incorporated. As such, the final objective is given by

$$\mathcal{L}_{NCDG} = \mathcal{L}_{cov}(f, \mathcal{D}_S) + \mathcal{L}_{cov}(f, \hat{\mathcal{D}}_S) + \beta \mathcal{L}_{sim}(f, \mathcal{D}_S, \hat{\mathcal{D}}_S), \quad (4)$$

where β is a parameter for balancing between neuron coverage loss and neuron gradient similarity loss. We also show the training details in Algorithm 2.

It is worth mentioning that our proposed algorithm involves "gradient through a gradient" operation, where the computational cost increases linearly upon the number of

model layers. Nevertheless, our computational cost is still tractable according to the following analyses.

1) The outputs of neurons and gradients share the same size. By treating the neuron output and gradients as the input of the loss function, our proposed coverage loss shares the similar computational cost compared with the gradient-regularization based technique *when conducting backward computation*.
2) In analogous to other domain adaptation techniques that conduct domain alignment on latent feature spaces, our proposed coverage loss does not necessarily need to maximize the coverage for *all* neurons. For example, while we consider all neurons in the Digit Recognition task, only the neurons in the convolutional layers are considered for Object Recognition task. Therefore, one can further reduce the computational cost by only involving partial neurons for coverage maximization. How to select the neurons for coverage maximization will be investigated in our future work.
3) While the computational cost of our proposed method is higher than that of the vanilla model (i.e., directly training the model with cross-entropy loss), *the computational cost of our method is conceptually the same as vanilla model during the inference stage*.

4 EXPERIMENTS

4.1 Single-Source Domain Generalization

To evaluate the performance of NCDG in domain generalization, we first focus on a worst-case scenario, namely single-source domain generalization (SSDG), and compare our NCDG with state-of-the-art SSDG methods, including JiGen [17], GUD [8] and the recently proposed M-ADA [9], as well as the Empirical Risk Minimization (ERM) baseline [33].

4.1.1 SSGD Evaluation on Digit Recognition

We first evaluate on a standard SSDG benchmark *Digits* [8]. In particular, the model is trained on one single source dataset: MNIST [34] and tested on other four digital datasets including SVHN [35], MNIST-M [36], SYN [36] and USPS [37] all at once. Following the experimental protocol of prior SSDG works including GUD [8] and M-ADA [9], we select the first 10,000 images from MNIST train split for training, and then evaluate the classification accuracy on the test split of the four testing datasets as four different domains. All images are resized to 32×32. We also convert MNIST and USPS from grey scale to RGB image by duplicating the grey channel twice.

Following the same protocol, a ConvNet with architecture *conv-pool-conv-pool-fc-fc-softmax* is used as the training model. Adam optimizer with learning rate of 0.0001 and batch size of 32 is adopted. The model is trained for 32 epochs, which is equivalent to 10,000 iterations. For NCDG, we track the activation of neurons in all layers before the softmax and set activation threshold t to 0.005 and other parameters as $\lambda = 0.1, \beta = 0.01$. Since original MNIST samples are grey-scale images, we choose to apply intensity reversing [38] for augmentation purpose[1]. To further validate the proposed method, we compare it with another baseline model (Vanilla**) trained directly on original and augmented MNIST data without domain generalization.

TABLE 1
SSDG classification accuracy (%) on *Digits*. The superscript $^{+}$ denotes the augmentation with our proposed loss. The superscript ** denotes the baselines with pixel intensity reversing. In particular, the Vanilla** denotes the Vanilla scheme (only cross-entropy loss) with the pixel intensity reversing. By comparing the NCDG with the four schemes augmented with pixel intensity reversing (Vanilla**, JiGen**, GUD**, M-ADA**), it is apparent that the proposed loss achieves better performance under the same augmentation method. By comparing the GUD^{+} and GUD, as well as the M-ADA^{+} with M-ADA, significant performance improvement originating from our proposed loss is observed.

Method	SVHN	MNIST-M	SYN	USPS	Avg.
ERM	27.8	52.7	39.6	76.9	49.3
JiGen	33.8	57.8	43.8	77.2	53.1
GUD	35.5	60.4	45.3	77.3	54.6
M-ADA	42.6	67.9	49.0	78.5	59.5
GUD^{+}	40.3	62.1	46.4	80.3	57.3
M-ADA^{+}	47.7	70.7	51.9	81.9	62.5
Vanilla**	54.2	73.9	58.6	90.9	69.4
JiGen**	53.7	75.2	60.1	91.9	70.2
GUD**	56.3	77.1	62.3	90.3	71.5
M-ADA**	58.0	**78.1**	60.9	91.1	72.0
NCDG	**59.7**	77.4	**63.8**	**92.6**	**73.4**

We report the results in Table 1. As we can observe, all domain generalization techniques can outperform the baseline model ERM, and NCDG can achieve significant performance improvement in all scenarios. On the other hand, we can also achieve desired performance by directly training on the original and augmented data with cross-entropy loss (i.e., Vanilla**). We conjecture that the reason lies in the intensity reversing which can enhance the structure information for digit recognition. By involving the proposed domain generalization regularization term, we can achieve 4% performance improvement on average compared with directly training on the original and augmented MNIST.

We also compare the proposed method with the baseline schemes augmented by pixel intensity reversing (JiGen**, GUD**, M-ADA**). As we can observe, while better performance can be achieved for baseline techniques with pixel intensity reversing, our proposed method can still better performance in most of the cases.

Last but not the least, we justify that our method can be applied with other augmentation based methods. To this end, we consider the adversarial augmentation method proposed in GUD [8] and M-ADA [9] and apply our proposed loss for training. We observe that the improvement can be achieved by a large margin compared with GUD and M-ADA, which further illustrates the effectiveness of our proposed method.

4.1.2 SSDG Evaluation on PACS

We consider a more realistic dataset: PACS [16] for evaluation. PACS covers objects of 7 different classes: *dog, elephant, giraffe, guitar, house, horse, person* and consists of 4 domains

1. It is worth mentioning that other augmentation methods can also be applied.

TABLE 2
SSDG classification accuracy (%) on PACS dataset with Resnet-18 as backbone model. Each row indicates the result of training on a single source domain and testing on the other three domains.

Source Domain	Method	Photo	Art_painting	Cartoon	Sketch	Avg.
Photo	DeepAll	/	66.0	26.7	35.0	42.5
	JiGen	/	64.1	23.9	32.9	40.3
	GUD	/	55.8	**33.3**	45.6	44.9
	M-ADA	/	64.3	29.8	35.2	43.1
	NCDG	/	**68.8**	29.8	**48.6**	**49.0**
Art_painting	DeepAll	**96.5**	/	60.6	52.5	69.9
	JiGen	95.5	/	60.1	50.2	68.6
	GUD	93.7	/	61.1	56.2	70.4
	M-ADA	95.0	/	61.5	47.6	68.0
	NCDG	95.0	/	**68.6**	**66.4**	**76.6**
Cartoon	DeepAll	**87.4**	67.6	/	68.3	74.5
	JiGen	85.1	65.5	/	65.7	72.1
	GUD	86.5	67.2	/	68.5	73.1
	M-ADA	83.1	66.4	/	66.3	71.9
	NCDG	85.8	**71.6**	/	**71.9**	**76.4**
Sketch	DeepAll	42.0	32.2	54.2	/	42.8
	JiGen	47.2	35.5	51.8	/	44.8
	GUD	32.9	23.1	37.5	/	31.2
	M-ADA	36.9	22.0	42.6	/	33.9
	NCDG	**47.9**	**45.6**	**65.8**	/	**53.1**

including *Art-Painting*, *Cartoon*, *Photo* and *Sketch*. For SSDG evaluation, the images for training are from a single domain, and the performance of the model is evaluated on another domain.

We train NCDG on PACS datasets using backbone architecture Resnet-18 [26]. We set the learning rate as 0.001, the batch size as 32 by training with SGD optimizer. Due to the skip connection existing in Resnet-18, we use the output from the first convolutional layer and the outputs of all the four Resnet blocks before the final fully connected layer for neuron coverage measurement. We consider the original PACS images together with the images processed by the standard augmentation techniques (e.g., image cropping, flipping, jittering), which are widely adopted in ImageNet challenge [39] and other domain generalization techniques (e.g., [9], [17]), as the input for fair comparison. The parameters are set as $\lambda = 0.1, t = 0.005, \beta = 0.01$. For baseline implementation, we adopt the open-source codes and report the best performance obtained by tuning the parameters in a wide range.

The results are reported in Table 2, and we can observe that the proposed method can achieve significantly better performance in all scenarios. Compared with the baseline by directly training on augmented data (i.e., DeepAll), the other single domain generalization methods suffer from performance drop, especially by considering *Sketch* as the source domain. This may originate from the large domain gap between source and target domains. In this case, data augmentation through adversarial training may not be helpful. While JiGen [17] considers jigsaw puzzle shuffling as a regularization term, it may still suffer from overfitting problem due to the large domain gap, as the fine-grain information learned on the source domain through jigsaw puzzle shuffling still belongs to the source domain, which may not be able to be generalized to the target domain.

4.1.3 SSDG (Robustness) Evaluation on CIFAR-10-C

CIFAR-10-C [40] is a typical robustness benchmark consisting of 19 corruptions types with five levels of serveries. The robustness evaluation strategy is similar to SSDG evaluation: each corruption type applied to the original data can be considered as a different domain. We follow the settings of [9] to train our model on CIFAR-10 and evaluate the model on CIFAR-10-C under the highest corruption severity "5". The Wide Residual Network (WRN) [41] with 16 layers and the width 4 is used as the NCDG backbone. The outputs from the WRN blocks before the final fully connected layer are considered for coverage loss computing, and the hyper-parameters are set as $\lambda = 0.01, t = 0.01, \beta = 0.01$. Other detailed training settings including the SGD optimizer, the decayed learning rate etc., are the same as [9]. We adopt Augmix for augmentation purpose due to the reason that Augmix is one of the standard techniques for robust deep neural network training [42].

The results are shown in the Table. 3. Based on the results, we have the following observations. First, Augmix can significantly outperform M-ADA, which suggests that the task prior knowledge is important for generalization capability improvement. Second, both NCDG$_{w/o\ Grad}$ (our proposed method without gradient regularization loss) and NCDG$_{w/o\ Cov}$ (our proposed method without coverage maximization loss) can lead to performance improvement, which verifies the effectiveness of our proposed coverage maximization term and gradient similarity regularization term. Third, by jointly conducting NCDG$_{w/o\ Grad}$ and NCDG$_{w/o\ Cov}$, we can achieve the best performance in most of the cases, further demonstrating the effectiveness of our proposed framework for domain generalization tasks.

4.1.4 SSDG Evaluation on Cross-Domain Segmentation

We consider a challenging setting on cross-domain semantic image segmentation task between the synthetic dataset GTA5 [44], SYNTHIARAND-CITYSCAPES (SYNTHIA) [45] and real-world dataset CITYSCAPES [46], where the synthetic dataset is used as source domain and the CITYSCAPES dataset is used as unseen target domain. In particularly, we adopt DRN-C-26 [47] as the backbone, crop the image with the size 600 $\times$ 600, and apply SGD with learning rate 0.001 and momentum 0.9 with a batch size of 8 for training purpose. We follow [43] by applying domain randomization (DR) for augmentation purpose. We report mean intersection-over-union (mIoU) for evaluation and the results are shown in Table 4. As we can see, our proposed method can achieve consistently better performance on segmentation task where unlabeled target domain data. It is worth noting that we also conduct adversarial learning [8], [9] for data augmentation but find it cannot achieve desired performance compared with DR. We conjecture the reason that adversarial learning may not handle the large domain shift between synthetic data and real-world data. Furthermore, we consider meta-learning (ML), which was adopted in [9], as another baseline based on the original and augmented data through DR. Based on the results, we find our proposed NCDG can also achieve better performance, which further justifies the effectiveness of our proposed method. We then consider the domain adaptation setting where the prior knowledge in target domain is available. Particularly, we follow [43] by applying CycleGAN for style transfer, and subsequently apply our proposed method based on the new data. As we can see, our method is also effective by combining with domain adaptation techniques,

TABLE 3
Robustness comparison on *CIFAR-10-C*. We report the classification accuracy (%) of 19 corruptions under the severest corruption level "5"

	Weather			**Blur**					**Noise**			
	Fog	Snow	Frost	Zoom	Defocus	Glass	Gaussian	Motion	Speckle	Shot	Impulse	Gaussian
ERM	65.9	74.3	61.6	60.0	53.7	49.4	30.7	63.8	41.3	35.4	25.7	29.0
M-ADA	69.4	80.6	76.7	68.0	61.2	61.6	47.3	64.2	60.9	60.6	45.2	56.9
Augmix	80.3	82.2	78.3	88.0	88.9	63.5	85.2	86.4	70.2	66.0	59.3	58.0
NCDG$_{w/o\ Grad}$	80.1	81.6	80.2	88.0	**89.2**	66.1	**85.3**	86.2	72.6	69.1	62.1	63.4
NCDG$_{w/o\ Cov}$	**81.5**	83.2	81.2	87.7	88.8	66.3	84.5	86.2	70.7	67.7	60.9	60.5
NCDG	81.1	**83.5**	**82.1**	**88.1**	89.0	**68.0**	85.2	**86.5**	**74.7**	**71.7**	**66.8**	**66.2**

	Digital							
	Jpeg	Pixelate	Spatter	Elastic	Brightness	Saturate	Contrast	Avg.
ERM	69.9	41.1	75.4	72.4	91.3	89.1	36.9	56.2
M-ADA	77.1	52.3	80.6	75.6	90.8	87.6	29.7	65.6
Augmix	78.5	63.0	87.4	77.3	91.6	89.8	62.0	76.6
NCDG$_{w/o\ Grad}$	78.9	63.0	88.5	77.8	92.1	90.9	62.1	77.8
NCDG$_{w/o\ Cov}$	**79.1**	61.3	**89.0**	78.4	**93.0**	**92.0**	63.0	77.6
NCDG	78.7	**63.4**	88.6	**80.2**	92.2	89.9	**69.1**	**79.2**

TABLE 4
Performance comparisons on cross-domain semantic image segmentation.

		With Target Domain Data		W/O Target Domain Data		
Source	DeepAll	CycleGAN [43]	NCDG+CycleGAN	DR	DR+ML	NCDG
GTA5	22.7	39.6	41.4	24.9	25.3	**27.0**
SYNTHIA	18.3	27.1	30.2	19.5	19.6	**20.1**

which further justifies the effectiveness of our proposed method.

Last but not the least, we also consider more strong competitors to justify the effectiveness of our proposed method. Specifically, we apply our proposed method to the Deeplab v3+ [48], Deeplab v3+ with IBN layer [49] and RobustNet [50] (i.e., Deeplab v3+ with ISW) with Resnet50 as the backbone (as suggested in [50]) by considering GTAV as source domain and CityScape as target domain. The results are shown below. As we can observe, our proposed method can consistently achieve the best performance in all scenarios, which show the effectiveness of our proposed method. It is also worth noting that our proposed method can achieve the state-of-the-art performance by comparing with the recent method Robostnet.

TABLE 5
Performance comparisons on cross-domain semantic image segmentation on DeepLabv3+ with Resnet50 as backbone.

Method	Baseline	IBN	ISW
w/o coverage	29.7	33.9	36.6
w/ coverage	**34.7**	**35.6**	**37.2**

4.1.5 Component Analysis

To investigate the specific role of each NCDG objective loss component, besides the results we show in Table 3, we also conduct detailed ablation experiments on PACS datasets using SSDG settings based on Resnet-18. Specifically, we evaluate the performance of NCDG without the gradient regularization term as well as NCDG without coverage maximization. The latter is achieved by setting neuron coverage weight λ to zero to eliminate the influences of neuron coverage loss. The results are shown in Table 6.

As we can see, generally, both coverage loss and similarity regularization contribute to the final performance by comparing with the DeepAll baseline in Table 2. We also observe that, in most of the cases, the performance can be further boosted by combining the two together, which further justify the effectiveness of our proposed method. However, in some other cases, our proposed gradient regularization method may not be helpful, especially when using "Photo" as the target domain. We conjecture that there are two possible reasons.

- Compared with the target domain, the sample diversity on source domain as well as its corresponding augmented domain may not be that high. Therefore, applying gradient regularization on source domain may not be able to capture reliable decision behavior of the target.
- When using Art-Painting and Cartoon as source domains, we can already achieve a relatively high performance by using Photo as target domain with coverage maximization only. Therefore, applying gradient regularization may lead to over-fitting problem on source domain.

It is also interesting to analyze impact of applying neuron coverage loss at different layers of the network. To this end, we use sketch as source domain and the others as target domains. In our original experiment setting, we use the outputs from the first convolutional layer and all the four Resnet blocks for coverage loss computing. To further understand the relationship between the neuron location and final performance, we conduct ablation studies by considering different number of layer outputs, where $conv_0$ denotes the output of the first convolutional layer, and $conv_{0,1,..,k}$ denotes the output of the first convolutional layer along with the outputs of Resnet block(s) from 1 to

TABLE 6
Component analysis on PACS dataset.

Source Domain	Method	Photo	Art_painting	Cartoon	Sketch	Avg.
Resnet-18						
Photo	NCDG$_{w/o\ Grad}$	/	65.1	25.9	39.9	43.6
	NCDG$_{w/o\ Cov}$	/	67.0	29.0	45.9	47.3
	NCDG	/	**68.8**	**29.8-**	**48.6**	**49.0**
Art_painting	NCDG$_{w/o\ Grad}$	**96.4**	/	65.7	59.0	73.7
	NCDG$_{w/o\ Cov}$	93.8	/	67.4	63.7	75.0
	NCDG	95.0	/	**68.6**	**66.4**	**76.6**
Cartoon	NCDG$_{w/o\ Grad}$	**88.6**	70.9	/	69.3	76.3
	NCDG$_{w/o\ Cov}$	86.2	69.3	/	**72.6**	76.0
	NCDG	85.8	**71.6**	/	71.9	**76.4**
Sketch	NCDG$_{w/o\ Grad}$	**49.2**	35.8	51.9	/	45.6
	NCDG$_{w/o\ Cov}$	45.9	36.8	53.4	/	45.4
	NCDG	47.9	**45.6**	**65.8**	/	**53.1**

k. The results are shown in Table 7. As we can see, the performances can be variant by considering different layers. In object recognition task based on PACS, we observe that, in most of the cases, better performance can be achieved by considering more neurons for coverage loss computing.

Besides the loss components, we are also interested in whether our proposed method can be applied to the network with different activation functions. To this end, we further consider PReLU and ELU on the single domain generalization task based on PACS dataset. In particular, as shown in Table 8, Sketch domain is used as the source domain and the other domains are treated as target. As we can see, our proposed method can consistently achieve better performance when using PReLU and ELU as the activation functions.

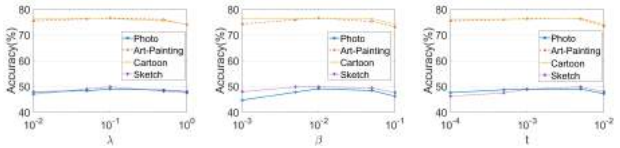
TABLE 7
Layer analysis of Resnet-18 on *PACS* (Sketch as the source domain).

Layers	Photo	Art	Cartoon	Avg.
$conv_0$	44.9	33.1	62.2	46.7
$conv_{0,1}$	46.8	35.5	63.3	48.5
$conv_{0,1,2}$	45.3	44.2	64.5	51.4
$conv_{0,1,2,3}$	**48.7**	44.7	63.9	52.4
$conv_{0,1,2,3,4}$	47.9	**45.6**	**65.8**	**53.1**

TABLE 8
SSDG comparison between different activation functions of Resnet-18 on *PACS* (Sketch as the source domain).

Activation Func	Photo	Art	Cartoon	Avg.
DeepAll$_{PReLU}$	44.7	34.3	52.1	43.7
NCDG$_{PReLU}$	45.0	36.5	61.7	47.7
DeepAll$_{ELU}$	43.7	33.5	52.1	43.1
NCDG$_{ELU}$	45.0	37.8	60.9	47.9
DeepAll$_{ReLU}$	42.0	32.2	54.2	42.8
NCDG$_{ReLU}$	**47.9**	**45.6**	**65.8**	**53.1**

It is worth noting that feature selection can also benefit generalization capability [51], and our proposed method is conceptually different with the feature selection as we require all neurons to be activated by at least one training sample. Therefore, we are also interested in the superiority of our proposed coverage loss against feature selection based techniques. To this end, we replace our coverage loss with L1 regularization [51] term on hidden layers which are also considered for the coverage loss. In particular, we consider the single domain generalization task by using PACS



(a) (b) (c)

Fig. 2. Parameter sensitivity analysis by varying (a) λ, (b) β, (c) t. Each curve denotes the performance by considering the domain shown in legend as source domain. The performance is reported by averaging the results on other three domains.

benchmark, where Sketch is used as the source domain, and the others are treated as target domains. We tune the hyper-parameter of L1 regularization in a wide range to report the best performance we can obtain. As we can see from Table 7, L1 regularization can generally achieve better performance compared with DeepAll baseline, which shows that feature selection can help with generalization performance. Nevertheless, our proposed method can achieve much better performance compared with results using L1 regularization, which shows the effectiveness of our proposed method.

TABLE 9
SSDG Comparison between NCDG proposed loss and L1 regularization loss on PACS dataset (Sketch domain as the source)

Method	Photo	Art	Cartoon	Avg.
DeepAll	42.0	32.2	54.2	42.8
L1	41.5	32.7	64.0	46.1
NCDG	**47.9**	**45.6**	**65.8**	**53.1**

Last but not the least, it is beneficial to look at divergence between source and target domain. To this end, we calculate the Kullback–Leibler (KL) divergence between source and target domain by using "sketch" as source domain and the other three as target domain. We use the output of the last Resnet block as the feature space. The results are shown as follow. We can draw the conclusion that the distribution distance between source and target domain based on KL divergence can be decreased when applying our method compared with "DeepAll".

TABLE 10
Kullback–Leibler divergences between the source domain: Sketch, with the other three target domains of DeepAll and NCDG models.

Method	Photo	Art	Cartoon
DeepAll	0.0044	0.0043	0.0027
NCDG	**0.0030**	**0.0022**	**0.0018**

4.1.6 Parameter Sensitivity Analysis

qWe conduct parameter sensitivity study on PACS with Resnet-18 based on SSGD setting. More specifically, we use one domain from PACS as source and report the average classification accuracy on the other three domains. The results are shown in Fig. 2 by varying the parameters. As we can see, the final performance is stable based on different parameters, which shows the robustness of our proposed method. However, extreme parameters may lead to significant performance drop. For example, when lowering β which controls "similarity", the performance decreases as

TABLE 11
MSDG classification accuracy (%) on Office-Home datasets. Each column indicates the results on a given target domain.

Office-Home	Art	Clipart	Product	Real-World	Avg.
DeepAll	55.6	42.4	70.3	70.9	59.8
D-SAM	58.0	44.4	69.2	71.5	60.8
JiGen	53.0	47.5	71.5	72.8	61.2
L2A-OT	**60.6**	50.1	74.8	**77.0**	65.6
DDAIG	59.2	52.3	74.6	76.0	65.5
RSC	58.4	47.9	71.6	74.5	63.1
DSON	59.4	45.7	71.8	74.7	62.9
NCDG	59.8	**53.1**	**75.3**	76.3	**66.1**

TABLE 12
MSDG classification accuracy (%) on VLCS. Each column indicates the results on a given target domain.

VLCS	Caltech	LabelMe	Pascal	Sun	Avg.
DeepAll	85.7	61.3	62.7	59.3	67.3
TF	93.6	63.4	70.0	61.3	72.1
MMD-AAE	94.4	62.6	67.7	64.4	72.3
D-SAM	91.8	57.0	58.6	60.8	67.0
JiGen	96.9	60.9	70.6	64.3	73.2
RSC	**97.6**	61.9	**73.9**	68.3	75.4
EISNet	97.3	63.5	69.8	68.0	74.7
NCDG	97.2	**67.6**	70.7	**68.7**	**76.1**

the effectiveness of similarity regularization can be reduced by lowering β. On the other hand, the performance also drops when $\beta >= 0.05$, as in this scenario the objective over-fits to the similarity regularization term.

4.2 Multi-Source Domain Generalization

We further validate the proposed method NCDG with a more general setting, where multiple source domains are available during training, i.e., Multi-Source Domain Generalization (MSDG). We treat the training data as a compound source domain where domain knowledge is not available in the model training of NCDG. The evaluations are conducted on PACS as well as two other benchmark datasets Office-Home [52] and VLCS [7].

It is worth noting that the aforementioned benchmark datasets contain different numbers of categories ranging from 5 to 65. Such difference renders a well-established testbed to verify the scalability of NCDG as diversities of object categories are considered. Following the existing setting, we apply the leave-one-out strategy for training and evaluation, such that one domain is selected as the target while the remaining domains are treated as source. We also follow the same data augmentation and neuron coverage computing strategy as PACS based on SSDG setting.

We compare NCDG against several state-of-the-art domain generalization baselines, including TF [16], MMD-AAE [53], D-SAM [54], JiGen [17], L2A-OT [19], DDAIG [55], EISNet [56], DSON [57], InfoDrop [58] and RSC [20]. In particular, both L2A-OT and DDAIG require specific prior domain knowledge regarding the source training data for adversarial domain generation.

4.2.1 MSDG Evaluation on Office-Home

Following the setting of existing DG methods [17], [19], for each domain, we randomly select 90% for training and 10% for validation, using ImageNet [39] pre-trained Resnet-18 [26] as the backbone model with training setting as PACS SSDG experiment. The introduced hyper-parameters in NCDG are set as $\lambda = 1, t = 0.005, \beta = 0.01$ for training purpose.

The results are shown in Table 11. As we can see, by comparing with the baseline methods D-SAM and JiGen which do not require domain knowledge during training stage, we can achieve significantly improvement in all scenarios. On the other hand, by comparing with the baselines where domain knowledge is utilized, we can also achieve competitive performance and achieve the best performance on average, indicating the effectiveness of our proposed method.

TABLE 13
MSDG classification accuracy (%) on PACS. Each column indicates the results on a given target domain.

PACS	Photo	Art painting	Cartoon	Sketch	Avg.
AlexNet					
DeepAll	87.7	63.3	63.1	54.1	67.1
D-SAM	85.6	63.9	69.4	64.7	71.2
JiGen	89.0	67.6	71.7	65.2	73.4
RSC	90.9	**71.6**	**75.1**	66.6	76.1
EISNet	**91.2**	70.4	71.2	70.3	75.9
NCDG	89.0	68.9	74.7	**72.9**	**76.4**
Resnet-18					
DeepAll	94.3	77.4	75.7	69.6	79.2
D-SAM	95.3	77.3	72.4	77.8	80.7
JiGen	96.0	79.4	75.3	71.4	80.5
L2A-OT	**96.2**	83.3	78.2	73.6	82.8
DDAIG	95.3	84.2	78.1	74.7	83.1
RSC	96.0	83.4	80.3	80.9	85.2
EISNet	95.9	81.9	76.4	74.3	82.2
InfoDrop	96.1	80.3	76.5	76.4	82.3
DSON	95.9	**84.7**	77.7	**82.3**	85.1
NCDG	95.4	82.3	**82.3**	82.1	**86.2**

4.2.2 MSDG Evaluation on VLCS

Following [17], [53], we choose ImageNet pre-trained AlexNet [59] as the backbone for evaluation, with all images resized to 225×225. For parameter selection, we choose $\lambda = 0.1, t = 0.001, \beta = 0.01$. We also use SGD for model training and the setting is the same as Office-Home.

The results are reported in Table 12. As we can see, we can achieve competitive performance when using *Caltech*, *Pascal* and *Sun* as target domains, and significant improvement can be achieved by evaluating on *LabelMe*. Overall, we can achieve the best performance on average by comparing with all other baselines.

4.2.3 MSDG Evaluation on PACS

To demonstrate the model-agnostic generalization ability that NCDG brings, we follow [17] to train NCDG on PACS using backbone architectures Resnet-18 and AlexNet individually. For Resnet-18, we set the parameters in NCDG as $\lambda = 1, t = 0.005, \beta = 0.01$, and $\lambda = 0.1, t = 0.001, \beta = 0.01$ are used for AlexNet. Other training details are the same as those for Office-Home based on ResNet-18 and VLCS based on AlexNet.

The results in Table 13 reveal that our proposed method can achieve better performance when considering *Cartoon* and *Sketch* as the target domains, and achieve competitive performance when using *Photo* and *Art Painting* as target

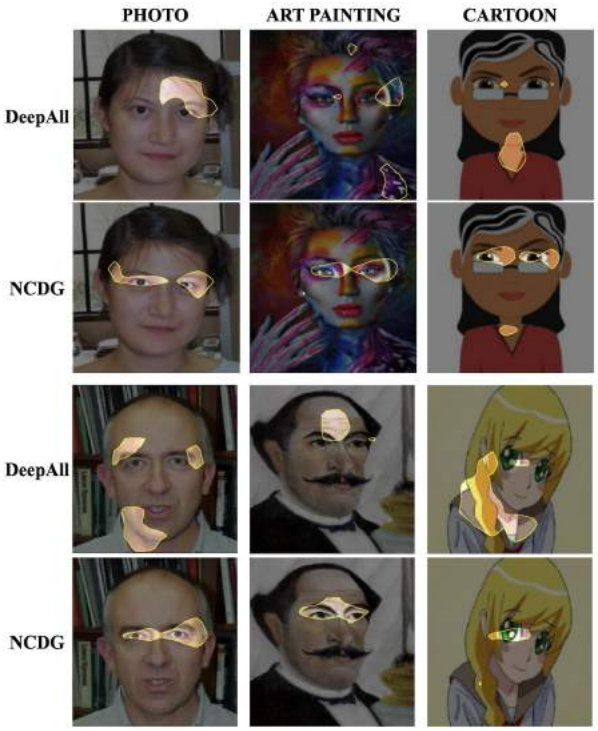


Fig. 3. Visualization results of network dissection. Each column indicates images from one target domain and each row shows the visualization result of unit 170 by either DeepAll model or our proposed NCDG.

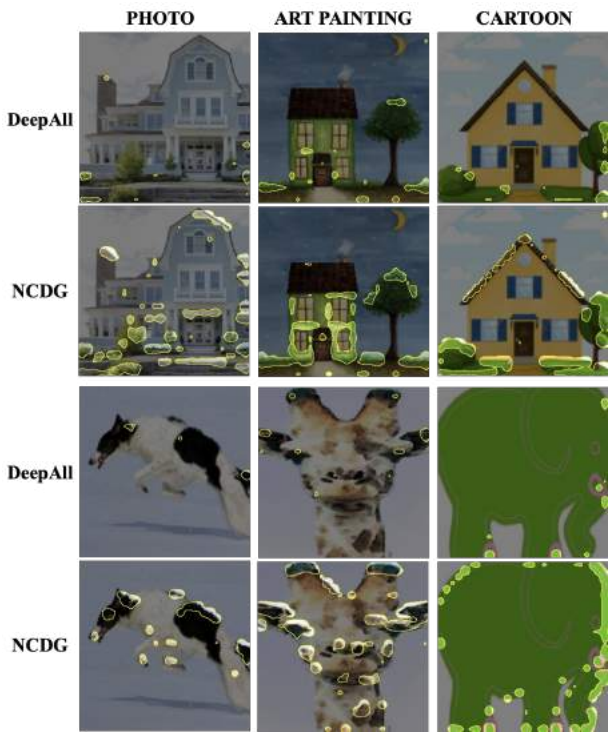


Fig. 4. Visualization results of network dissection. Each column indicates images from one target domain and each row shows the visualization result of ResNet-18 block 2 unit 20 by either DeepAll model or our proposed NCDG.

domains by comparing the baselines leveraging domain information. We can also achieve the best performance on average, which further shows the effectiveness of our proposed method.

4.3 Connection with Network Dissection

Here we provide in-depth analyses regarding the effectiveness of our proposed method. In particularly, we aim to justify our motivation based on network dissection [13], which is to identify the semantics information of individual hidden units (i.e., channel) of the image classification network.

Recall that our proposed method is built upon the assumption that an out-of-distribution sample for evaluation may trigger the inactive neuron, which further leads to misclassification behavior. To better understand how this behavior is triggered, we can examine the unit of network trained by both DeepAll strategy (i.e., directly training with cross-entropy loss) and our proposed algorithm. Specifically, we consider using PACS dataset based on single domain generalization setting trained with Resnet-18 as an example, where the network is trained based on Sketch domain and evaluated on the other three domains. We choose two images belonging to the "person" category, which are misclassified by DeepAll but can be correctly predicted by our proposed method, from each target domain. We further visualize unit 170 from the output of block 3 of Resnet-18 by feeding the images to the network, as we empirically find that unit 170 is not activated if we simply train Resnet-18 based on Sketch domain by using DeepAll strategy. By contrast, the unit can be activated by using our proposed method. The visualization results are shown in Fig. 3.

As we can observe, all testing samples can activate the unit 170 of Resnet-18 block 3. No semantic information is observed from unit 170 when directly training the network with DeepAll, thus it is reasonable that the network predicts wrongly as the activated unit 170 triggers unexpected behavior. We can also observe that the concept of "eye" can be consistently extracted by our proposed method across samples, and such concept has a strong connection with the "person" category, which further drives the network to predict correctly. Such observation justifies our motivation that maximizing the neuron coverage facilitates the reduction on the number of possible defects introduced by out-of-distribution sample. On the other hand, our observation is also consistent with the finding in [13] that the classification performance of a certain category can be explained by the units that identify visual concepts of this class.

Besides focusing on the output of ResNet-18 block 3 which captures semantic information, we are also interested in the low-level block output. To this end, we visualize the unit 20 of ResNet-18 block 2 by considering sketch as source domain and the remaining ones as target domain based on different categories. Based on the results in Fig. 4, we observe that our proposed method can explore some edge and contour information, which can benefit object recognition. Again, it can be difficult for us to observe useful information extracted by DeepAll baseline. Thus, it is reasonable that the network predicts wrongly if the given neuron is activated by out-of-distribution samples.

5 Conclusions

In this paper, we propose to improve the generalization capability of DNN from the perspective of neuron coverage maximization. By modeling the DNN as a program, we propose to maximize the neuron coverage (i.e., control flow) of DNN with the gradient (i.e., data flow) similarity regularization between the original data and the augmented data during the training stage, such that the trained DNN can be better generalized to the out-of-distribution samples. Extensive experiments on various domain generalization tasks verify the effectiveness of our proposed method.

References

[1] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," *arXiv preprint arXiv:1611.03530*, 2016.

[2] K. Kawaguchi, L. P. Kaelbling, and Y. Bengio, "Generalization in deep learning," *arXiv preprint arXiv:1710.05468*, 2017.

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

[4] S. J. Pan and Q. Yang, "A survey on transfer learning," *TKDE*, 2010.

[5] K. Muandet, D. Balduzzi, and B. Schölkopf, "Domain generalization via invariant feature representation," in *ICML*, 2013.

[6] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales, "Learning to generalize: Meta-learning for domain generalization," *arXiv preprint arXiv:1710.03463*, 2017.

[7] A. Torralba and A. A. Efros, "Unbiased look at dataset bias," in *CVPR*, 2011.

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

[9] F. Qiao, L. Zhao, and X. Peng, "Learning to learn single domain generalization," in *CVPR*, 2020.

[10] I. Tolstikhin, O. Bousquet, S. Gelly, and B. Schoelkopf, "Wasserstein auto-encoders," *arXiv preprint arXiv:1711.01558*, 2017.

[11] K. Pei, Y. Cao, J. Yang, and S. Jana, "Deepxplore: Automated whitebox testing of deep learning systems," in *SOSP*, 2017.

[12] L. Ma, F. Juefei-Xu, F. Zhang, J. Sun, M. Xue, B. Li, C. Chen, T. Su, L. Li, Y. Liu *et al.*, "Deepgauge: Multi-granularity testing criteria for deep learning systems," in *ASE*, 2018.

[13] D. Bau, J.-Y. Zhu, H. Strobelt, A. Lapedriza, B. Zhou, and A. Torralba, "Understanding the role of individual units in a deep neural network," *Proceedings of the National Academy of Sciences*, vol. 117, no. 48, pp. 30 071–30 078, 2020.

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

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

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

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

[18] H. Wang, Z. He, Z. C. Lipton, and E. P. Xing, "Learning robust representations by projecting superficial statistics out," *ICLR*, 2019.

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

[20] Z. Huang, H. Wang, E. P. Xing, and D. Huang, "Self-challenging improves cross-domain generalization," *arXiv preprint arXiv:2007.02454*, 2020.

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

[22] D. Li, J. Zhang, Y. Yang, C. Liu, Y.-Z. Song, and T. M. Hospedales, "Episodic training for domain generalization," *arXiv preprint arXiv:1902.00113*, 2019.

[23] Y. Li, Y. Yang, W. Zhou, and T. M. Hospedales, "Feature-critic networks for heterogeneous domain generalization," *arXiv preprint arXiv:1901.11448*, 2019.

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

[25] L. Serafini and A. d. Garcez, "Logic tensor networks: Deep learning and logical reasoning from data and knowledge," *arXiv preprint arXiv:1606.04422*, 2016.

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

[27] ——, "Identity mappings in deep residual networks," in *ECCV*, 2016.

[28] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *ICML*, 2017.

[29] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *CVPR*, 2005.

[30] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[31] A. Jacot, F. Gabriel, and C. Hongler, "Neural tangent kernel: Convergence and generalization in neural networks," in *NeurIPS*, 2018.

[32] S. Chen, W. Wang, and S. J. Pan, "Cooperative pruning in cross-domain deep neural network compression." in *IJCAI*, 2019.

[33] V. Koltchinskii, *Oracle Inequalities in Empirical Risk Minimization and Sparse Recovery Problems: Ecole d'Eté de Probabilités de Saint-Flour XXXVIII-2008*. Springer Science & Business Media, 2011, vol. 2033.

[34] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[35] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," 2011.

[36] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *ICML*, 2015.

[37] J. S. Denker, W. Gardner, H. P. Graf, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jackel, H. S. Baird, and I. Guyon, "Neural network recognizer for hand-written zip code digits," in *NeurIPS*, 1989.

[38] R. C. Gonzalez, R. E. Woods, and S. L. Eddins, *Digital image processing using MATLAB*. Pearson Education India, 2004.

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

[40] D. Hendrycks and T. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," *arXiv preprint arXiv:1903.12261*, 2019.

[41] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.

[42] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan, "Augmix: A simple data processing method to improve robustness and uncertainty," *arXiv preprint arXiv:1912.02781*, 2019.

[43] A. Dundar, M.-Y. Liu, T.-C. Wang, J. Zedlewski, and J. Kautz, "Domain stylization: A strong, simple baseline for synthetic to real image domain adaptation," *arXiv preprint arXiv:1807.09384*, 2018.

[44] S. R. Richter, V. Vineet, S. Roth, and V. Koltun, "Playing for data: Ground truth from computer games," in *ECCV*, 2016.

[45] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, "The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in *CVPR*, 2016.

[46] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *CVPR*, 2016.

[47] F. Yu, V. Koltun, and T. Funkhouser, "Dilated residual networks," in *CVPR*, 2017.

[48] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–818.

[49] X. Pan, P. Luo, J. Shi, and X. Tang, "Two at once: Enhancing learning and generalization capacities via ibn-net," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 464–479.

[50] S. Choi, S. Jung, H. Yun, J. T. Kim, S. Kim, and J. Choo, "Robustnet: Improving domain generalization in urban-scene segmentation via instance selective whitening," in *Proceedings of the IEEE/CVF*

Conference on Computer Vision and Pattern Recognition, 2021, pp. 11 580–11 590.
[51] J. Gui, Z. Sun, S. Ji, D. Tao, and T. Tan, "Feature selection based on structured sparsity: A comprehensive study," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 7, pp. 1490–1507, 2016.
[52] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *CVPR*, 2017.
[53] H. Li, S. Jialin Pan, S. Wang, and A. C. Kot, "Domain generalization with adversarial feature learning," in *CVPR*, 2018.
[54] A. D'Innocente and B. Caputo, "Domain generalization with domain-specific aggregation modules," in *German Conference on Pattern Recognition*. Springer, 2018, pp. 187–198.
[55] K. Zhou, Y. Yang, T. Hospedales, and T. Xiang, "Learning to generate novel domains for domain generalization," *arXiv preprint arXiv:2007.03304*, 2020.
[56] S. Wang, L. Yu, C. Li, C.-W. Fu, and P.-A. Heng, "Learning from extrinsic and intrinsic supervisions for domain generalization," in *European Conference on Computer Vision*. Springer, 2020, pp. 159–176.
[57] S. Seo, Y. Suh, D. Kim, G. Kim, J. Han, and B. Han, "Learning to optimize domain specific normalization for domain generalization," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*. Springer, 2020, pp. 68–83.
[58] B. Shi, D. Zhang, Q. Dai, Z. Zhu, Y. Mu, and J. Wang, "Informative dropout for robust representation learning: A shape-bias perspective," in *International Conference on Machine Learning*. PMLR, 2020, pp. 8828–8839.
[59] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NeurIPS*, 2012.

Chris Xing TIAN received the B.S. degree in computer science from Beihang University in 2015 and the M.S. degree in computer science from National University of Singapore in 2017. From 2017 to 2020, he was a senior software engineer in iQIYI Inc. He is currently a PhD student supervised by Dr. Shiqi Wang, in the Department of Computer Science, City University of Hong Kong.

Haoliang Li received the B.S. degree in communication engineering from University of Electronic Science and Technology of China (UESTC) in 2013, and his Ph.D. degree from Nanyang Technological University (NTU), Singapore in 2018. He is currently an assistant professor in Department of Electrical Engineering, City University of Hong Kong. His research mainly focuses on AI security, multimedia forensics and transfer learning. His research works appear in international journals/conferences such as TPAMI, IJCV, TIFS, NeurIPS, CVPR and AAAI. He received the Wallenberg-NTU presidential postdoc fellowship in 2019, doctoral innovation award in 2019, and VCIP best paper award in 2020.

Xiaofei Xie received his Ph.D, M.E. and B.E. from Tianjin University. He is currently an assistant professor in Singapore Management University, Singapore. His research mainly focuses on program analysis, traditional software testing and quality assurance analysis of artificial intelligence. He has published some top tier conference/journal papers relevant to software analysis in ICSE, ISSTA, FSE, ASE, TDSC, TIFS, TSE, IJCAI, NeurIPS, ICML and CCS. In particular, he won two ACM SIGSOFT Distinguished Paper Awards in FSE'16 and ASE'19.

Yang Liu graduated in 2005 with a Bachelor of Computing (Honours) in the National University of Singapore (NUS). In 2010, he obtained his Ph.D. and started his post-doctoral work in NUS and MIT. In 2012, he joined Nanyang Technological University (NTU), and currently is a full professor and Director of the cybersecurity lab in NTU. Dr. Liu specializes in software verification, security and software engineering. His research has bridged the gap between the theory and practical usage of formal methods and program analysis to evaluate the design and implementation of software for high assurance and security. By now, he has more than 270 publications in top tier conferences and journals. He has received a number of prestigious awards including MSRA Fellowship, TRF Fellowship, Nanyang Assistant Professor, Tan Chin Tuan Fellowship, Nanyang Research Award and 8 best paper awards in top conferences like ASE, FSE and ICSE.

Shiqi Wang (Member, IEEE) received the B.S. degree in computer science from the Harbin Institute of Technology in 2008 and the Ph.D. degree in computer application technology from Peking University in 2014. From 2014 to 2016, he was a Post-Doctoral Fellow with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. From 2016 to 2017, he was a Research Fellow with the Rapid-Rich Object Search Laboratory, Nanyang Technological University, Singapore. He is currently an Assistant Professor with the Department of Computer Science, City University of Hong Kong. He has proposed over 50 technical proposals to ISO/MPEG, ITU-T, and AVS standards, and authored/coauthored more than 200 refereed journal articles/conference papers. He received the 2021 IEEE multimedia rising star award, the Best Paper Award from IEEE VCIP 2019, ICME 2019, IEEE Multimedia 2018, and PCM 2017 and is the coauthor of an article that received the Best Student Paper Award in the IEEE ICIP 2018. His research interests include video compression, image/video quality assessment, and image/video search and analysis.