Generalization Beyond Feature Alignment: Concept Activation-Guided Contrastive Learning

Yibing Liu, Chris Xing Tian, Haoliang Li, Member, IEEE, and Shiqi Wang, Senior Member, IEEE

Abstract—Learning invariant representations via contrastive learning has seen state-of-the-art performance in domain generalization (DG). Despite such success, in this paper, we find that its core learning strategy – feature alignment – could heavily hinder model generalization. Drawing insights in neuron interpretability, we characterize this problem from a neuron activation view. Specifically, by treating feature elements as neuron activation states, we show that conventional alignment methods tend to deteriorate the diversity of learned invariant features, as they indiscriminately minimize all neuron activation differences. This instead ignores rich relations among neurons – many of them often identify the same visual concepts despite differing activation patterns. With this finding, we present a simple yet effective approach, Concept Contrast (CoCo), which relaxes element-wise feature alignments by contrasting high-level concepts encoded in neurons. Our CoCo performs in a plug-and-play fashion, thus it can be integrated into any contrastive method in DG. We evaluate CoCo over four canonical contrastive methods, showing that CoCo promotes the diversity of feature representations and consistently improves model generalization capability. By decoupling this success through neuron coverage analysis, we further find that CoCo potentially invokes more meaningful neurons during training, thereby improving model learning.

Index Terms—Domain generalization, neuron activation, contrastive learning, concept contrast.

I. INTRODUCTION

OVER the past decades machine learning systems have achieved remarkable success [1], [2], [3], [4], with the assumption that training and test data are sampled from independent and identical distributions. In real-world applications where data acquisition condition largely varies, however, this assumption rarely holds. The distribution shifts between training and test data thus appear, leading to the degraded performance on well-trained models. To this end, domain generalization (DG) [5] proposes to learn models that generalize well on unseen domains by leveraging a collection of source domains during training.

In contrast to domain adaptation [6] which assumes target data is accessible, DG considers a more realistic scenario where only source domains can be utilized during the training phase. As such, there is much hope resting on that the learned feature representations could keep invariance on the unseen targets [7], [8], [9], [10], [11], [12], thereby powering the predictor trained merely on source domains to perform well. Benefiting from the recent advances in self-supervised learning [13], [14], [15], [16], contrastive-based learning has been proposed as a promising solution [17], [18], [19], [20], [21], [22], [23], [24], [25]. A canonical paradigm underlying this theme is to exploit rich sample-to-sample relations from different domains, whereby the feature representations of same-class samples (positive pairs) are aligned together, while the different-class ones (negative pairs) are brought apart. Since feature distance is explicitly optimized over domain shifts, the model could eventually learn class-conditional feature invariance via such contrasts. However, in this paper, we find that conventional feature alignment strategies of contrastive learning would instead hinder the model generalization.

To illustrate this point, we draw insights from the recent progress in neuron interpretability [26], [27], [28], [29], [30], [31], [32], [33], which shows that neurons (e.g., convolutional channels) in the network often emerge as concept detectors and identify the meaningful visual concepts (See Figure 2 for examples). Given such rich semantic meanings of neurons, however, we find that current contrastive methods mostly perform via indiscriminately minimizing all neuron activation differences, i.e., $\|z^- - z^+\|^2$ in Figure 1. This instead ignores intricate relations among neurons: even if all neuron activations differ given two images, they can still encode the same semantic information. For instance, in Figure 1, both $z$ and $z^+$ capture the same visual concept “dog nose” (either via the 677-th or 136-th neuron). However, their neuron activations exhibit stark contrasts, e.g., the 677-th neuron is only activated in $z$ but not in $z^+$. From this standpoint, we thus argue that such direct alignment methods – overly stressing the same

\begin{footnote}{1}\text{Neuron activations commonly refer to the outputs of neurons following prior work [26], [27], [28], [29], [30], [31], [32], [33]. They can also be viewed as feature elements, as features typically stem from neurons (e.g., convolutional channels) in the network.}\end{footnote}
Fig. 1. (a) Contrastive learning pipeline for DG, where a positive pair of images \((x, x^+)\) are aligned through the contrast loss \(L\). (b) Feature contrast calculates \(L\) proportionally to the direct feature difference \(|z - z^+|^2\), which excessively emphasizes the same neuron activations at the same positions. This can result in collapsed feature representations and degrade model generalization. (c) Our Concept Contrast (CoCo) calculates \(L\) proportionally to the high-level concept difference \(|v - v^+|^2\), thereby relaxing constraints on individual feature elements and promoting diversified feature representations.

neuron activations – could constrain the diversity of learned features. This further leads to the collapsed representations and deteriorates model generalization capability. We provide the empirical analysis of this issue in Figure 5.

In this paper, we present Concept Contrast (CoCo) to address the above problem. Instead of performing element-wise alignments at the feature level, our CoCo contrasts high-level concepts encoded in neurons. Concretely, as many neurons in the network often operate in a similar fashion, we first model relations among neurons by employing neuron summarization techniques [28], [34], whereby different neurons identifying the similar visual concepts are clustered. To further retrieve concept representations from features, we then introduce concept activation vectors (CAVs), which aggregates activation states from same-cluster neurons. Finally, by optimizing the divergence between CAVs, we carry out contrastive loss on the concept level. Note that since our method does not impose extra parameters, it can be applied into any contrastive method in DG with easy integration.

To validate the effectiveness of CoCo, we conduct extensive experiments with four typical contrastive methods over various DG benchmarks. The quantitative and qualitative results demonstrate that CoCo not only boosts invariant feature learning, but also consistently improves model performance across datasets and methods. In addition, through neuron coverage analysis [35], [36], [37], we further find that CoCo potentially invokes more meaningful neurons during training, e.g., +11.5% neuron coverage ratio on SelfReg [21] over PACS dataset [38]. Our code is available at https://github.com/bierone/CoCo.

II. RELATED WORK

A. Domain Generalization

The main goal of domain generalization (DG) is to learn models that can overcome distribution shifts between training and test domains. Since target data is assumed not accessible under this setting, recent studies mostly center on learning invariant representations over single or multiple source domains [7], [10], [11], [12], [39], [40], [41], [42], [43], [44], [45]. In contrast, another direction attempts to improve generalization via classical learning strategies, such as meta learning [46], [47], [48], ensemble learning [49], [50], and gradient-based learning [8], [9], [35]. Some works also employ data augmentation techniques to enlarge data space and mitigate the overfitting problem in representation learning [51], [52], [53].

B. Contrastive Learning

Contrastive learning has recently emerged as a powerful strategy in various tasks, such as kernel learning [54] and self-supervised learning [13], [14], [15], [16], and domain adaptation [55]. Specific to the DG problem, CCSA [17] and MASF [18] are first proposed to improve class-level
feature invariance by aligning the feature distributions of same-class samples. Subsequently, SelfReg [21] argues the effects of negative contrast pairs and merely learns features with positive pairs. PCL [56] further finds that the large distribution gap between source domains could make contrastive learning ineffective, and utilizes proxy-based loss to replace typical simple-to-simple contrast loss. On the other end of the spectrum, STEAM [57] employs contrastive loss to learn inter-domain and intra-domain style invariance. PDEN [24] and CSG [22] also apply contrastive learning to single-domain DG for generalized feature learning. Contrary to these works, in this paper, we extend the idea of neuron explainability to feature learning and targets at improving the model generalization ability.

D. Neuron Coverage

Neuron coverage refers to the percentage of activated neurons given an input set. Motivated by the code coverage in system testing, it is first introduced by [36] to simulate the bug testing in neural networks. The high level idea is that if a network performs with larger neuron coverage during training, there would be fewer undetected bugs (e.g., misclassification) triggered in the test phase. In light of this, a number of approaches dive into testing deep learning systems with neuron coverage criteria [37], [62], [63], [64]. In DG setting, [35] recently introduces neuron coverage as a measurement of model generalization capability, and proposes to maximize neuron coverage in the training. In this paper, we also demonstrate that CoCo has great potential to improve neuron coverage and further improves the model generalization performance.

III. METHODOLOGY

We consider the common formulation of domain generalization (DG), where multiple source domains are available during training. In our formal setup, let $D = \{D_1, D_2, \ldots, D_S\}$ be the training set consisting of $S$ source domains on the joint space $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ respectively denote the input and label spaces. We characterize each domain as $D_i = \{(x_i^j, y_i)^j\}_{j=1}^{N_i}$, where $N_i$ is the number of data samples. Without any prior knowledge on the unseen targets, we aim to learn a generalized model $F_{0,\phi}$ which comprises of a feature extractor $f_0 : \mathcal{X} \rightarrow \Gamma$ and a task-related predictor $g_\phi : \Gamma \rightarrow \mathbb{R}^C$, where $\Gamma$ indicates the feature space and $C$ is the number of classes in $\mathcal{Y}$.

In this section, we first motivate our method by reviewing typical contrastive methods in DG, which perform with element-wise feature contrasts (Section III-A). Then, we introduce details of Concept Contrast (CoCo) in Section III-B.
Fig. 3. Conceptual comparison of feature contrast and our concept contrast. Here we utilize three colors (yellow, blue, and green) to differentiate the roles acted by neurons, and gray ones are inactivated neurons. (a) Element-wise feature contrast requires all neurons to be aligned at the same degree, even though activated neurons (e.g., yellow ones) in two feature vectors have encoded the same visual concepts. (b) CoCo abstracts high-level concepts encoded in neurons and relaxes feature contrast by merely optimizing concept-level differences.

A. Preliminaries

Feature contrast for domain generalization mainly targets at learning class-conditional invariant representations over domain shifts [17], [18], [19], [21], [25]. The key idea is to explicitly optimize the feature distance that aligns the same-class samples from different domains (positive pairs) closer, while pushing different-class ones (negative pairs) apart. Formally, given an anchor sample \( x_i \) within a minibatch, we define its feature embedding as \( z_i = f_\theta(x_i) \). The positive pair is then constructed by selecting another same-class sample \( x^+_i \) in the batch, and negative samples are the remaining ones with different classes. By using the typical InfoNCE loss [65], a feature-level contrast can be given by:

\[
L_f = -\sum_i \log \frac{\exp(-|z_i - z^+_i|^2/2)}{\exp(-|z_i - z^+_i|^2/2) + \sum_{j \neq i} \exp(-|z_i - z^+_j|^2/2)},
\]

where \( z^+_i \) and \( z^-_j \) are the embedding vectors of positive and negative samples, respectively. Note that here \(-|z_i - z^+_i|^2/2 \) can be directly replaced by the inner product \( z^T_i z_i + z^T_i z^+_i \), as all feature embeddings are normalized to unit vectors.

Since all feature elements stem from the neuron output, it is natural to treat feature embeddings as activation strengths of neurons, i.e., activation states [28], [35]. However, given rich relations among neurons (See Figure 2), feature contrast performs in an element-wise way, which aligns all neurons indiscriminately, i.e., \( |z_i - z^+_i|^2 \) in Equation 1. This inevitably enforces strong constraints on all neurons, thus limiting representation ability of learned invariant features.

B. Generalization via Concept Contrast

From the above perspective, we propose concept contrast (CoCo), which relaxes element-wise feature alignments via contrasting high-level concepts encoded in neurons. Our hypothesis is that as long as two feature vectors encode the same visual concepts, they can be quantified with less difference despite the divergence between their feature elements.

We provide a conceptual comparison with feature contrast in Figure 3. In detail, given an initialized model, CoCo first identifies the relations among neurons, whereby a set of concept clusters \( U^* \) is summarized. Next, to perform the contrastive loss, we abstract features to concept activation vectors (CAVs) based on \( U^* \), thereby optimizing divergence at the concept level. We introduce them in the following.

C. Neuron Summarization

The details of neuron summarization are provided in Algorithm 1 and Figure 4. The central idea is to identify concepts encoded in neurons, and group the ones sharing similar roles. Since there are no pre-defined concepts, here we assume that annotated labels in datasets can be the guidance. To further refine such class-level concepts, we take advantage of domain information. That is, under DG settings where domains broadly vary in styles, some domain-specific concepts often emerge in each class.

Algorithm 1 Neuron Summarization

| Algorithm 1 Neuron Summarization |
|----------------------------------|
| **Input:** \( f_\theta \rightarrow \) feature extractor; \( F_{\theta, \phi} \rightarrow \) DNN model; \( \tau \rightarrow \) threshold for neuron activation; \( D = \{D_s\}_{s=1}^S \rightarrow \) training dataset with \( C \) classes and \( S \) domains. |
| **Output:** \( U^* \rightarrow \) concept clusters where each cluster consists of neurons identifying the same visual concepts. |
| 1: \( \triangleright \) Main procedure starts |
| 2: \( U = \{\} \) |
| 3: for \( c \leftarrow 1, \ldots, C \) do |
| 4: \( U_c = \{\} \) |
| 5: for \( s \leftarrow 1, \ldots, S \) do |
| 6: \( \triangleright \) Step1: Collecting neuron states |
| 7: \( X_{c,s} := \{x^c_s \in D_s | y^c_s = c \wedge F_{\theta, \phi}(x^c_s) = c\} \) |
| 8: \( Z_{c,s} = [f_\theta(x^c_s), \ldots, f_\theta(x^c_s)] \) |
| 9: \( \triangleright \) Step2: Filtering noisy neuron activations |
| 10: \( \text{cond} \rightarrow (Z_{c,s} \leq \tau) \Rightarrow (Z_{c,s}[\text{cond}] = 0) \) |
| 11: \( \triangleright \) Step3: Clustering neurons unsupervisedly |
| 12: \( U_{c,s} = \text{K-Means}(Z_{c,s}) \) |
| 13: \( U_c = U_c \cup U_{c,s} \) |
| 14: end for |
| 15: Compute \( U^*_c \) by merging domain-level overlapped clusters in \( U_c \) based on Equation (2) |
| 16: \( U = U \cup U^*_c \) |
| 17: end for |
| 18: Compute \( U^* \) by merging class-level overlapped clusters in \( U \) based on Equation (2) |
| 19: \( \text{return} \ U^* \) |

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
class-level coarse concepts under different domains and then group neurons pertaining to such class-domain conditioned concepts to get the neuron clusters $U^c$.

We start by constructing a set of class-domain conditioned clusters $U_{c,s} = \{u_{i}^{c,s}\}_{i=1}^{K}$, where each cluster $u_{i}^{c,s}$ comprises neurons identifying a domain-specific concept in class $c$ under domain $s$. Inspired by the recent findings that neurons detecting the same concepts often co-activate many common images [28], [34], we build $U_{c,s}$ via grouping neurons which share a set of top-activated images. In detail, given a set of training samples $X_{c,s}$ under domain $s$ and class $c$, three steps are performed:

1. **Step1 (Collecting):** By treating feature elements as neuron activation states, we first collect a neuron activation matrix $Z_{c,s} = \{f_{\theta}(x_n^{c,s})\}_{n=1}^{N}$, where each neuron $f_{\theta}(x_n^{c,s})$ is the feature vector for sample $x_n^{c,s}$ through the feature extractor $f_{\theta}$, which is a set of images that activate neuron $n$. $N$ indicates the number of neurons in the last feature layer.

2. **Step2 (Filtering):** Following the prevalent formulation of neuron activations [26], [32], [33], [35], [64], we introduce a threshold $t$ to filter out invalid and noisy neuron activations in $Z_{c,s}$, such that neurons with states less (or equal) than $t$ would be marked as 0, indicating non-activated states.

3. **Step3 (Clustering):** After the filtering process, $n$-th row in $Z_{c,s}$ would correspond to the set of images that activate $n$-th neuron (which we would refer to as the **stimuli set** of $n$-th neuron). In this sense, we finally apply unsupervised clustering (e.g., K-Means) to group neurons based on their stimuli sets, forming preliminary clusters $U_{c,s}$.

Since different domains may enjoy similar visual concepts, we then merge overlapped clusters across domains. In particular, following [34], for each class $c$, we calculate Jaccard similarity on the union set of clusters $U_c = \bigcup_{s \in \mathcal{S}} U_{c,s},$

$$\text{Jaccard}(u_i^c, u_j^c) = \frac{|u_i^c \cap u_j^c|}{|u_i^c \cup u_j^c|},$$

where $u_i^c \in U_c$ and $u_j^c \in U_c$ denote clusters under class $c$. The clusters with a similarity score closer to 1 are merged. After that, we get the refined class-level clusters $U_c^*$.

Likewise, the overlap among class-level concepts is also considered. As such, by performing Equation 2 again on the $U = \bigcup_{c \in \mathcal{C}} U_c^*$, we get the final set of neuron clusters $U^*$. Lastly, as neurons may not limit to the single-concept detection, we calculate the contribution of neurons in each concept cluster $u_i \in U^*$. In detail, for each neuron $n \in u_i$, we calculate its weight as the overlap in activated images:

$$w_n = \frac{|X_n|}{|\bigcup_{n \in u_i} X_n|},$$

where $X_n = \{x|\forall x \in X, f_{\theta}(x) > t\}$ is a set of images that activate neuron $n$. $X$ denotes all training samples. Note that since CoCo works in a finetuning fashion when neuron concepts evolve slowly [33], we typically update $U^*$ every 1,000 steps, with a maximum of 5,000 training steps.

### D. Concept Contrast

After modeling the relations among neurons, we then perform concept-level contrastive learning by abstracting concept activation vectors (CAVs) from feature embeddings. In particular, we define $i$-th element of CAV $v$ as $v_i$, which connects with the activation state of $i$-th concept identified in $U^*$. In this sense, given a feature vector $z$, we calculate $v_i$ as the weighted sum of all neuron states in $i$-th concept cluster $u_i \in U^*$.

$$v_i = \sum_{n \in u_i} w_n z_n,$$

where $n$ is a neuron in concept cluster $u_i$, and $w_n$ denotes its corresponding weight. $z_n$ is the $n$-th element in the feature vector, which is also treated as activation state of neuron $n$. As the philosophy behind the CoCo is to contrast concept states, we simply replace all feature vectors $z$ by $v$ in Eq. (1), and then the concept-level contrastive loss $L_c$ can be obtained.

### E. Discussions

It is worth noting that due to domain gaps in DG settings, a large variance among same-class samples often challenges the learning of class-level invariant features [56]. The direct consequence can be the loss of diversity in feature representations, which could lead to the overfitting problem in generalization [22]. However, tackling this issue is always non-trivial. Our approach CoCo can be considered as a relaxation on the contrastive feature learning in DG, which overcomes this problem from a new view: learning **invariant concepts** instead of **invariant features**. Since concept comparison relaxes the constraints on feature elements, the diversity of learned features can be improved. Lastly, as there are no extra parameters imposed by CoCo, it allows easy integration into
any contrastive method in DG, enabling enhanced feature learning.

IV. EXPERIMENTS

In this section, we first detail the implementation of our CoCo (Section IV-A), and then introduce the feature diversity metric, which is utilized to evaluate learned invariant features (Section IV-B). In Section IV-C, we demonstrate the effectiveness of our CoCo on the four datasets of DomainBed benchmark [66]. The further experiments on the task of synthetic-to-real generalization [67] are also elaborated, as this task relies on the diversity of learned features (Section IV-D).

A. Implementation Details

1) Training: Following prevalent formulations of DG approaches [7], [17], [22], we train models comprising of a ImageNet-pretrained feature extractor (e.g., ResNet50 [3]), and a predictor (e.g., MLP). The whole training process consists of two stages: 1) we first initialize a model trained with the classical ERM strategy [70]; and then 2) we fine-tune the model with CoCo-based contrastive loss \( \mathcal{L}_c \), where the clustering is typically performed every 1000 training steps. Most of our implementations are built upon the DomainBed official code using PyTorch.\(^3\) For Synthetic-to-Real generalization, we base our implementation on the source code of CSG.\(^4\)

2) Hyperparameters: For all employed contrastive methods, we utilize the default hyperparameters of the original paper. As aforementioned, the implementation of CoCo involves two hyperparameters: the threshold \( t \) and the number of clusters \( K \). Following [26], we dynamically set \( t \) as the top 1% quantile level of neuron outputs. To accelerate the neuron clustering, we further set neuron states larger than the threshold \( t \) as 1. The search space for \( K \) is \([1, 2, 5, 10]\). Besides, we discard neurons with activation times below \( \lambda X_{c,i} \) to account for abnormal neuron activations. \( \lambda \) is the minimum proportion of activated samples required for a valid neuron, and the corresponding search space is \([1\%, 5\%, 10\%, 50\%]\). All hyperparameters are randomly searched following the protocol of DomainBed [66], thereby facilitating method adaptation. Following [34], we merge clusters with a Jaccard similarity greater than 0.8 to address overlapping clusters.

B. Feature Diversity Measurement

While contrastive methods are effective to learn invariant features in DG, they have long been argued with the problem of representation collapse [14], [76], thus limiting the diversity of learned features. Following [22], we employ the hyperspherical potential energy [77] to quantitatively measure the diversity of learned invariant features,

\[
E_\xi = \frac{1}{N} \sum_{i=1}^{N} \sum_{j \neq i} \frac{1}{s} \log(||z_i - z_j||^{-s}), \quad s > 0
\]

where \( \xi \) denotes the power factor, and \( N \) is the number of data samples. A lower potential energy \( E_\xi \) would represent the more diversified features. We utilize \( \xi = 0 \) in this paper.

C. Results on DomainBed Benchmark

1) Settings: We first evaluate our CoCo on the DomainBed benchmark [66], which is an arguably fairer testbed and rigorously evaluates DG approaches under the same conditions. As digital images may not contain abundant visual concepts, we experiment with our method on non-MINIST datasets following [25], and four datasets are employed: PACS (4 domains, 9,991 images) [38], Office-Home (4 domains, 15,588 images) [78], TerraInc (4 domains, 24,788 images) [79], and DomainNet (6 domains, 586,575 images) [80].

We incorporate CoCo into four typical contrastive methods: SupCL [15], PCL [56], SelfReg [21] and CondCAD [25] on DomainBed. For a fair comparison, we follow the official source codes and run over all the required processing steps to get results. To report final scores, we employ training-domain validation sets for best model selection, and average our results over three random seeds.

2) Effectiveness in Improving Invariant Feature Learning:

As discussed previously (e.g., Figure 1), feature contrast learns class-conditional invariant features by imposing constraints on all neurons, which may suffer from the overfitting problem in feature representations. In this regard, we first verify such unwanted behavior, and then show the effectiveness of CoCo in improving invariant feature learning. Specifically, following [22] and [81] we plot the class-level feature diversity for two feature contrast methods, SelfReg and CondCAD, with and without CoCo.

---

\(^3\)https://github.com/facebookresearch/DomainBed
\(^4\)https://github.com/NVlabs/CSG
TABLE I

| Algorithm          | Art | Cartoon | Sketch | Average |
|--------------------|-----|---------|--------|---------|
| CORAL [43]         | 8.3 | 80.7    | 97.5   | 78.9    | 86.2    |
| DANN [41]          | 8.6 | 77.4    | 97.3   | 73.5    | 83.6    |
| MMD [39]           | 8.1 | 79.4    | 96.6   | 76.5    | 84.6    |
| CDANN [40]         | 8.65| 75.5    | 96.8   | 73.5    | 82.6    |
| MLDG [47]          | 8.5 | 80.1    | 97.4   | 76.6    | 84.9    |
| IRM [7]            | 8.4 | 76.4    | 96.7   | 76.1    | 83.5    |
| RSC [68]           | 8.5 | 79.7    | 97.6   | 78.2    | 85.2    |
| GroupDRO [69]      | 8.35| 79.1    | 97.6   | 78.3    | 84.4    |
| ARM [46]           | 8.6 | 76.8    | 97.4   | 79.3    | 85.1    |
| ERM [70]           | 8.47| 80.8    | 97.2   | 79.3    | 85.5    |
| MTV [71]           | 8.75| 77.1    | 96.4   | 77.3    | 84.6    |
| VREx [45]          | 8.6 | 79.1    | 96.9   | 77.7    | 84.9    |
| And-mask [72]      | 8.5 | 79.2    | 96.9   | 76.2    | 84.4    |
| SAND-mask [73]     | 8.3 | 79.2    | 96.3   | 76.9    | 84.6    |
| SagNet [74]        | 8.74| 80.7    | 97.1   | 80.0    | 86.3    |
| Fishr [9]          | 8.4 | 78.7    | 97.0   | 77.8    | 85.5    |
| DGRIR [75]         | /   | /       | /      | /       | 84.7    |
| PCL [56]           | 8.76| 79.9    | 96.4   | 77.4    | 85.3    |

As illustrated in Figure 5, the class-level features learned via feature-level contrast methods mostly center on a narrow subspace, which thus causes the collapsed representations with the overfitting problem. In comparison, by adopting concept-level contrasts (CoCo), feature representations are improved to span the whole space. This further introduces the higher diversity in class-level invariant features (i.e., lower hyperspherical energy $E_\xi$), and demonstrates the effectiveness of our CoCo in improving feature learning.

3) Effectiveness in Improving Generalization Ability: We further compare our approaches with recent state-of-the-art (SoTA) methods on four datasets of DomainBed. Table I-IV illustrate the results, where gray color denotes the methods designed for unsupervised domain adaptation, which peek target domains during training. Through the results, the following observations can be drawn:

Firstly, by incorporating CoCo into existing contrastive learning methods, we achieve a new state-of-the-art performance across four DomainBed datasets. It even outperforms CORAL [43], which peeks data from target domains.

Secondly, CoCo consistently enhances the generalization capability of contrastive methods (SupCL, PCL, SelfReg, CondCAD) across the four DomainBed datasets. For instance, by incorporating CoCo into SelfReg and CondCAD, a 2.4% and 1.2% improvement can be obtained on the TerraInc dataset, respectively. This highlights the effectiveness of CoCo.

Thirdly, in comparison with other employed contrastive methods, CoCo often introduces a larger improvement over SelfReg. We conjecture that the reason lies in the training strategy of SelfReg, which solely utilizes the positive feature alignment during training. This introduces a higher risk of overfitting, as verified by the severe collapse issue in Figure 5.

In this sense, incorporating CoCo takes more benefits as overfitting is largely alleviated.

D. Results on Synthetic-to-Real Generalization

1) Settings: We also consider the task of synthetic-to-real generalization to further validate the effectiveness of our approach, whereby the training set merely consists of synthetic
images rendered by 3D models; whereas the target data is all collected from real images. Following [67], we conduct experiments on the benchmark: VisDA-17→COCO. In particular, the training and validation sets in VisDA dataset [85] are employed as the source domain and target domain, respectively. Similar to theDomainBed experiments, we evaluate our method by incorporating it into a novel feature-contrast approach CSG [22], and each processing step is carefully followed.

2) Effectiveness in Improving Generalization Ability: Since this task is to adapt synthetically trained models to real images, it has been shown that improving feature diversity can be the key to prevent overfitting and improve model generalization [22], [67], [84]. From this perspective, we measure the feature diversity on target data, and compare our method with recent SoTA in Table V.

It can be seen that CoCo not only improves diversity of learned invariant features over CSG, but also enhances generalization ability of trained models. This highlights the effectiveness of CoCo again.

V. Analysis

A. Architecture Analysis

The key to our approach CoCo is dissecting neurons, and grouping the ones identifying the same concepts. As this approach hinges on the concept detection ability of neurons, we hypothesize that with a better pretrained model as backbone, CoCo could bring more remarkable improvement. To this end, we further experiment with our method on two CLIP models [87] pretrained with ResNet-50 [3] and ViT-B/32 [86] over DomainBed. As shown in Table VI, we can see that our CoCo still consistently improves contrastive methods over all datasets, and performs the best on the average accuracy. In addition, compared to the results based on ResNet50 in Table I-IV, it can be found that incorporating CoCo into CLIP-based backbones could cause higher improvement in many cases, e.g., +2.0% on CLIP-ResNet50 vs. +0.9% on ResNet50 over DomainNet. Such results confirm our hypothesis and further shows the great potential of CoCo in model generalization.

B. SWAD Analysis

Stochastic Weight Averaging Densely (SWAD) [88] has recently been widely utilized in various domain generalization approaches [21], [56], [89] as an efficient training technique, which enhances model generalization ability through dense model ensemble. In order to further understand the potential of CoCo, we conduct a series of experiments based on SWAD. As shown in Table VII and Table VIII, we implement CoCo on the plain supervised contrastive loss (SupCL) [15], and compare it with the recent SoTA, proxy-based contrastive method (PCL) [56]. It can be seen that CoCo not only significantly improves SupCL on both PACS and TerraInc datasets, but also outperforms PCL by a large margin. For instance, on the TerraInc dataset, incorporating CoCo results in a +4.2% improvement in average accuracy for SupCL, surpassing PCL by 1.8%. This finding underscores the potential of CoCo to further enhance model generalization ability.

C. Projection Head Analysis

Neuron summarization algorithm works by mapping the feature vector to the low-dim concept activation vector. This process is similar to the projection head (e.g., linear projection layer), which is widely used in self-supervised learning to enhance the learned feature representations. From this perspective, we further compare our CoCo with the widely-adopted projection approach (referred to as ProjHead), which implicitly transforms features into a low-dim space. To ensure a fair comparison, we implement ProjHead following [13] and [15], which comprises an MLP with one hidden layer and a ReLU function. The output size of ProjHead is also set as the same as our CoCo, e.g., both methods abstract feature from 2048-dim vector to 300-dim compact representation.

We provide the results on PACS [38] and TerraInc [79] datasets in Table IX. As can be seen, the implicit ProjHead only shows minimal impact on improving the contrastive learning algorithms. Instead, our Neuron Summarization...
D. Clustering Method Analysis

Head thus may not introduce substantial improvements. In our supervised learning regime, where features are self-supervised learning, relies on implicit feature transformations. In contrast, the projection head, although effective in this improvement lies in the nature of CoCo, which operates by explicitly clustering neurons based on their semantic meanings. As previously discussed, CoCo performs neuron clustering algorithm (CoCo) exhibits a significant improvement over the employed baselines, highlighting its superiority over the simple projection approach. We conjecture that the reason for this improvement lies in the nature of CoCo, which operates by explicitly clustering neurons based on their semantic meanings. In contrast, the projection head, although effective in self-supervised learning, relies on implicit feature transformations. In our supervised learning regime, where features are already optimized using labeled data, the additional projection head thus may not introduce substantial improvements.

E. Hyperparameter Analysis

As illustrated in Algorithm 1, our neuron clustering strategy takes advantage of annotated information, such as class and domain information, to model fine-grained concepts. To further evaluate the effectiveness of this approach, we run additional ablations. Table X illustrates the results, where we perform an ablation study by comparing the clustering method of CoCo with and without the use of annotations. As can be observed, CoCo achieves superior results when both domain and class annotations are employed. Additionally, even in the absence of annotations, CoCo still improves the baseline contrastive performance. Following [35], [36], and [37], we denote the neuron coverage as the proportion of activated images. Both scenarios hinder the effective clustering of neurons, which in turn degrades the performance of CoCo.

F. Computational Cost Analysis

As previously discussed, CoCo performs neuron clustering in an infrequent manner, i.e., clustering is executed every 1,000 training steps. This infrequent process effectively reduces the extra computational burden associated with CoCo. To gain a quantitative understanding of such computational cost, we compare the computation time of feature contrast method (CondCAD) and our concept contrast (CondCAD+CoCo). Table XI illustrates the results on a single RTX 3090. As can be seen, the extra computation time introduced by CoCo is merely 7.6 seconds for every 1,000 training steps. This amounts to approximately 3.6% additional running time when compared to the feature contrast method. Given such minimal impact on computational resources, CoCo proves to be an efficient technique for enhancing the model generalization ability.

G. Neuron Coverage Analysis

1) Neuron Coverage Criteria: Accuracy as the indicator of model generalization ability has recently been argued with a large variance in the final results [25], [66]. In this sense, we take another criterion, neuron coverage [36], to evaluate the generalization ability of our method. Following [35], [36], and [37], we denote the neuron coverage as the proportion of activated neurons under the training set,

$$NCov = \frac{|\{n|\forall x \in X, f_\theta(x) > \tau\}|}{N},$$  

where $X$ denotes all training samples, $n$ is the neuron id, and $N$ is the number of neurons in the last feature layer. The high level idea is that if a model performs with large neuron coverage during training, there would be fewer undetected bugs (e.g., misclassification) triggered by unseen samples.

**TABLE VI**

ARCHITECTURE ANALYSIS ON DOMAINBED BENCHMARK. TWO PRETRAINED CLIP MODELS WITH RESNET50 [3] AND ViT-B/32 [86] ARE RESPECTIVELY EMPLOYED. BASELINE DENOTES THE LINEAR CLASSIFIER TRAINED ON FROZEN FEATURE EXTRACTOR.

| Backbone | Algorithm | PACS [38] | OfficeHome [78] | TerraInc [79] | DomainNet [80] | Average |
|----------|-----------|-----------|-----------------|---------------|----------------|---------|
| CLIP-ResNet50 | Baseline | 90.3 ± 0.2 | 70.6 ± 0.1 | 29.6 ± 0.8 | 47.7 ± 0.0 | 59.6 |
|           | CondCAD [25] | 90.0 ± 0.6 | 70.5 ± 0.3 | 30.3 ± 0.9 | 45.5 ± 2.1 | 59.1 |
|           | +CoCo (Ours) | 90.8 ± 0.4 | 70.8 ± 0.1 | 32.6 ± 1.2 | 47.5 ± 0.1 | 60.4 |
| CLIP-ViT-B/32 | Baseline | 93.5 ± 0.8 | 79.4 ± 0.2 | 37.5 ± 0.7 | 50.1 ± 1.1 | 65.1 |
|           | CondCAD [25] | 93.5 ± 0.7 | 79.7 ± 0.2 | 37.4 ± 1.2 | 51.7 ± 1.4 | 65.6 |
|           | +CoCo (Ours) | 94.1 ± 0.3 | 79.8 ± 0.1 | 38.3 ± 0.5 | 53.3 ± 0.1 | 66.4 |

**TABLE VII**

SWAD-BASED RESULTS ON PACS DATASET WITH RESNET-50.

| Algorithm | Art | Cartoon | Photo | Sketch | Avg |
|-----------|-----|--------|------|--------|-----|
| SWAD [88] | 89.3 | 83.4 | 97.3 | 82.5 | 88.1 |
| +PCL [56] | 90.2 | 83.9 | 98.1 | 82.6 | 88.7 |
| +SupCL [15] | 90.1 | 81.8 | 97.6 | 83.2 | 88.2 |
| +SupCL+CoCo | 91.6 | 85.1 | 97.6 | 85.2 | 89.9 |

**TABLE VIII**

SWAD-BASED RESULTS ON TERRAINC DATASET WITH RESNET-50.

| Algorithm | Loc100 | Loc38 | Loc43 | Loc46 | Avg |
|-----------|--------|------|------|------|-----|
| SWAD [88] | 55.4 | 44.9 | 59.7 | 39.9 | 50.0 |
| +PCL [56] | 58.7 | 46.3 | 60.0 | 43.6 | 52.2 |
| +SupCL [15] | 56.6 | 44.6 | 57.1 | 41.1 | 49.8 |
| +SupCL+CoCo | 59.9 | 53.1 | 58.5 | 44.4 | 54.0 |
TABLE IX
PROJECTION HEAD ANALYSIS ON THE PACS [38] AND TERRAInc [79] DATASETS. RESULTS ARE AVERAGED OVER THREE RANDOM TRIALS

| Algorithm                  | PACS Dataset (%) | Terrainc Dataset (%) | Average |
|----------------------------|------------------|-----------------------|---------|
|                            | Art   | Cartoon | Photo | Sketch | Avg. | Loc100 | Loc38 | Loc43 | Loc46 | Avg. |
| ERM                        | 84.7  | 80.8   | 97.2  | 79.3    | 85.5 | 49.8   | 42.1  | 56.9  | 37.5  | 46.1 |
| SupCL [15]                 | 83.8  | 80.7   | 95.7  | 78.9    | 84.7 | 57.2   | 38.2  | 52.8  | 33.9  | 45.5 |
| +ProjHead                  | 84.4  | 80.0   | 95.6  | 79.5    | 84.9 | 56.8   | 37.4  | 52.9  | 35.5  | 45.7 |
| +CoCo (Ours)              | 84.8  | 81.6   | 96.3  | 80.6    | 85.8 | 57.4   | 44.3  | 52.3  | 36.4  | 47.6 |
| SelfReg [21]              | 85.6  | 82.1   | 97.0  | 77.7    | 85.6 | 48.8   | 41.3  | 57.3  | 40.6  | 47.0 |
| +ProjHead                  | 85.6  | 80.0   | 94.9  | 81.4    | 85.5 | 50.9   | 41.4  | 57.3  | 39.5  | 47.3 |
| +CoCo (Ours)              | 88.1  | 81.1   | 96.5  | 81.7    | 86.9 | 54.3   | 44.8  | 57.9  | 40.8  | 49.4 |
| CondCAD [25]              | 88.4  | 78.4   | 96.8  | 76.4    | 85.0 | 54.1   | 42.8  | 56.3  | 36.4  | 47.4 |
| +ProjHead                  | 87.4  | 79.5   | 96.3  | 76.6    | 85.0 | 52.8   | 43.6  | 53.1  | 38.4  | 47.0 |
| +CoCo (Ours)              | 87.9  | 80.7   | 96.9  | 79.9    | 86.4 | 53.9   | 42.0  | 57.2  | 41.4  | 48.6 |

Fig. 6. Neuron coverage analysis on the PACS [38], OfficeHome [78], and VLCS [90] datasets. Following [35], [36], and [37], we denote neuron coverage as the proportion of activated neurons under the training set. All curves are obtained by finetuning the same initialized ERM model with 300 pre-training steps. CoCo (Rand) implies that neurons in the same concept cluster are randomly selected.

TABLE X
ABALATION STUDIES ON CLUSTERING METHODS OF CoCo OVER PACS DATASET [38]. BACKBONE: ResNet-50

| Algorithm                  | Art   | Cartoon | Photo | Sketch | Avg. |
|----------------------------|-------|---------|-------|--------|------|
| SelfReg [21]              | 87.9  | 79.4    | 96.8  | 78.3   | 85.6 |
| -CoCo (w/o class & domain)| 90.3  | 77.7    | 96.3  | 78.7   | 85.8 |
| -CoCo (w/o domain)        | 88.3  | 79.8    | 96.3  | 80.8   | 86.3 |
| -CoCo (w/o class)         | 89.7  | 76.7    | 96.8  | 83.4   | 86.6 |
| -CoCo (Ours)              | 88.1  | 81.1    | 96.5  | 81.7   | 86.9 |

TABLE XI
COMPARISON OF COMPUTATION TIME (1,000 STEPS) BETWEEN FEATURE CONTRAST AND CONCEPT CONTRAST

| Operation                  | Computation Time (sec) |
|----------------------------|------------------------|
| Feature Contrast Computation | 212.6s                 |
| Concept Contrast Computation | 212.4s                 |
| Total                      | 220.2s                 |

2) Results: Figure 6 illustrates the neuron coverage curve of our CoCo based on SelfReg. For a fair comparison, we implement all baseline methods on the same pretrained ERM model, and then compare the change in the neuron coverage. As can be seen, by introducing our CoCo, the coverage ratio on all datasets presents an increasing trend, which further achieves the highest one among all the methods. This proves the effectiveness of our CoCo again. In addition, if we ablate the CoCo with random concept clusters, a significant gap in the coverage ratio can be observed, which highlights the advantages of our concept clustering method. Lastly, it can also be seen that the results presented by neuron coverage are consistent with the accuracy in Table I, i.e., SelfReg+CoCo > SelfReg > ERM. This indicates that coverage ratio can be a reasonable measurement of model generalization ability.
To further understand the benefits of a broader neuron coverage, we dissect newly activated neurons by comparing their activation maps with their inactivated versions. We provide the visualizations in Figure 7, where the image that activates a neuron (e.g., unit #125) the most across the entire dataset is visualized following [26]. As can be seen, by incorporating CoCo, inactivated neurons could evolve as concept detectors and identify the meaningful information in images, e.g., the unit 312 in CondCAD+CoCo evolves to detect birds in step 3000. This is in stark contrast to their inactivated period (e.g., step 300), when they merely encode insignificant regions, and thus can easily mislead a model if they are activated during test phases. As such, we can confirm our previous conclusions that larger coverage ratio introduced by CoCo promotes the model generalization ability.

H. Qualitative Feature Analysis

In this section, we provide the qualitative analysis regarding the feature learning of our CoCo. Since the target of CoCo is to overcome the feature collapse and improve the feature diversity, we visualize the learned representations of the CondCAD in the scenarios with or without using CoCo, as depicted in Figure 9. The key observations are summarized below.

Firstly, when comparing the feature visualizations in Figure 9 (a) and Figure 9 (b), it can be found that CoCo effectively improves the diversity of feature vectors, i.e., the clusters of using CoCo appear much sparser than the ones without using CoCo. Secondly, by comparing the feature visualization (Figure 9 (b)) and concept visualization (Figure 9 (c)), we can observe that while CoCo learns feature representations in a more diverse manner, it still maintains highly compact concept representations. This indicates that CoCo effectively performs invariant concept learning, preserving feature invariance at a higher concept level while enhancing the feature diversity.

In this section, we provide the qualitative analysis regarding the feature learning of our CoCo. Since the target of CoCo is to overcome the feature collapse and improve the feature diversity, we visualize the learned representations of the CondCAD in the scenarios with or without using CoCo, as depicted in Figure 9. The key observations are summarized below.

Firstly, when comparing the feature visualizations in Figure 9 (a) and Figure 9 (b), it can be found that CoCo effectively improves the diversity of feature vectors, i.e., the clusters of using CoCo appear much sparser than the ones without using CoCo. Secondly, by comparing the feature visualization (Figure 9 (b)) and concept visualization (Figure 9 (c)), we can observe that while CoCo learns feature representations in a more diverse manner, it still maintains highly compact concept representations. This indicates that CoCo effectively performs invariant concept learning, preserving feature invariance at a higher concept level while enhancing the feature diversity.

**VI. DISCUSSIONS**

In this section, we discuss the feasibility of CoCo in other visual applications, such as image captioning [91], [92], retrieval [93], [94], and segmentation [95], [96]. As illustrated in our previous experiments, the benefit of CoCo lies in the ability to enhance feature diversity and capture rich neuron relations. In this sense, for image captioning, this can potentially contribute to improved generalization ability, leading to a robust caption generation process across diverse visual scenarios. Similarly, in segmentation, CoCo increases feature diversity, which could lead to more diverse and semantically coherent segmentation results. In retrieval, CoCo can take advantage of high-level concept abstraction. This may enhance
the retrieval performance by finding visually similar images based on high-level concept similarities.

However, it is also important to acknowledge the limitations of CoCo in these tasks. While it promotes feature diversity, CoCo may not fully address the specific requirements and challenges of each application. For example, in image captioning, generating linguistically coherent and contextually appropriate captions requires additional considerations beyond feature diversity, which cannot be fulfilled by CoCo. It is thus essential to carefully evaluate and adapt CoCo to different visual applications to ensure its effectiveness.

VII. CONCLUSION

This paper explores the contrastive feature alignment in domain generalization from a neuron activation view. We presented a simple yet effective approach, Concept Contrast (CoCo), to relax element-wise feature alignments by contrasting high-level concepts encoded in neurons. Through the comparison with feature contrast methods, we demonstrated the capability of our CoCo to alleviate overfitting problem in invariant feature learning. The results on DomainBed and synthetic-to-real generalization further show that CoCo not only consistently improves model performance, but also contributes to a broader neuron coverage for model generalization.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in Proc. 26th Annu. Conf. Neural Inf. Process. Syst., 2012, pp. 1106–1114.

[2] C. Szegedy et al., “Going deeper with convolutions,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Sep. 2015, pp. 1–9.

[3] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770–778.

[4] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in Proc. ICLR, 2015.

[5] G. Blanchard, G. Lee, and C. Scott, “Generalizing from several related classification tasks to a new unlabeled sample,” in Proc. NeurIPS, 2011, pp. 2178–2186.

[6] J. Blitzer, R. McDonald, and F. Pereira, “Domain adaptation with structural correspondence learning,” in Proc. Conf. Empirical Methods Natural Lang. Process., 2006, pp. 120–128.

[7] M. Arjovsky, L. Bottou, I. Gulrajani, and D. Lopez-Paz, “Invariant risk minimization,” 2019, arXiv:1907.02893.

[8] Y. Shi et al., “Gradient matching for domain generalization,” in Proc. ICLR, 2022.

[9] A. Rame, C. Dancette, and M. Cord, “Fishr: Invariant gradient variances for out-of-distribution generalization,” in Proc. Int. Conf. Mach. Learn., 2022, pp. 18347–18377.

[10] Z. Ding and Y. Fu, “Deep domain generalization with structured low-rank constraint,” IEEE Trans. Image Process., vol. 27, no. 1, pp. 304–313, Jan. 2018.

[11] H. Xia, T. Jing, and Z. Ding, “Generative inference network for imbalanced domain generalization,” IEEE Trans. Image Process., vol. 32, pp. 1694–1704, 2023.

[12] Q. Dou et al., “Coarse-to-fine Domain Generalization via Multimodal Joint Learning,” in Proc. Int. Conf. Mach. Learn., 2020, pp. 235–250, 2023.

[13] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A simple framework for contrastive learning of visual representations,” in Proc. Int. Conf. Mach. Learn., 2020, pp. 1597–1607.

[14] J. Grill et al., “Bootstrap your own latent—A new approach to self-supervised learning,” in Proc. NeurIPS, 2020, pp. 21271–21284.

[15] P. Khosla et al., “Supervised contrastive learning,” in Proc. NIPS, 2020, pp. 18661–18673.

[16] F. Wang, T. Kong, R. Zhang, H. Liu, and H. Li, “Self-supervised learning by estimating twin class distribution,” IEEE Trans. Image Process., vol. 32, pp. 2228–2236, 2023.

[17] S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto, “Unified deep supervised domain adaptation and generalization,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 5716–5726.

[18] Q. Dou, D. C. de Castro, K. Kannitissas, and B. Glöckner, “Domain generalization via multimodal joint learning of semantic features,” in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2019, pp. 6447–6458.

[19] C. Yoon, G. Hamarneh, and R. Garbi, “Generalizable feature learning in the presence of data bias and domain class imbalance with application to skin lesion classification,” in Proc. MICCAI. Cham, Switzerland: Springer, 2019, pp. 365–373.

[20] S. Wang, L. Yu, C. Li, C.-W. Fu, and P.-A. Heng, “Learning from extrinsic and intrinsic supervisions for domain generalization,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2020, pp. 159–176.

[21] D. Kim, Y. Yoo, S. Park, J. Kim, and J. Lee, “SelfReg: Self-supervised contrastive regularization for domain generalization,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2021, pp. 9599–9608.

[22] W. Chen, “Contrastive syn-to-real generalization,” in Proc. ICLR, 2021.

[23] D. Mahajan, S. Tople, and A. Sharma, “Domain generalization using causal matching,” in Proc. Int. Conf. Mach. Learn., 2021, pp. 7313–7324.

[24] L. Li et al., “Progressive domain expansion network for single domain generalization,” in Proc. IEEE/CVF Conf. Vis. Pattern Recognit. (CVPR), Jun. 2021, pp. 224–233.

[25] Y. Ruan, Y. Dubois, and C. J. Maddison, “Optimal representations for covariate shift,” in Proc. ICLR, 2022.

[26] 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,” Proc. Nat. Acad. Sci. USA, vol. 117, no. 48, pp. 30071–30078, Dec. 2020.

[27] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, “Network dissection: Quantifying interpretability of deep visual representations,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 3319–3327.

[28] H. Park et al., “Concept evolution in deep learning training: A unified interpretation framework and discoveries,” 2022, arXiv:2203.16475.

[29] A. M. Nguyen, A. Dosovitskiy, J. Yosinski, T. Brox, and J. Clune, “Synthesizing the preferred inputs for neurons in neural networks via deep generator networks,” in Proc. NeurIPS, 2016, pp. 3387–3395.

[30] A. Nguyen, J. Yosinski, and J. Clune, “Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks,” 2016, arXiv:1602.03616.

[31] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Object detectors emerge in deep scene CNNs,” in Proc. ICLR, 2015.

[32] E. Hernandez, S. Schwettmann, D. Bau, T. Bagashvili, A. Torralba, and J. Andreas, “Natural language descriptions of deep visual features,” in Proc. ICLR, 2022.

[33] B. Zhou, D. Bau, A. Oliva, and A. Torralba, “Interpreting deep visual representations via network dissection,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 41, no. 9, pp. 2131–2145, Sep. 2019.

[34] H. Park et al., “NeuroCartography: Scalable automatic visual summarization of concepts in deep neural networks,” IEEE Trans. Vis. Comput. Graphics, vol. 28, no. 1, pp. 813–823, Jan. 2022.

[35] C. X. Tian, H. Li, X. Xie, Y. Liu, and S. Wang, “Neuron coverage-guided domain generalization,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 45, no. 1, pp. 1302–1311, Jan. 2023.

[36] K. Pei, Y. Cao, J. Yang, and J. Jana, “DeepXplor: Automated whitebox testing of deep learning systems,” in Proc. 26th Symp. Operating Syst. Princ., Oct. 2017, pp. 1–18.

[37] L. Ma et al., “DeepGauge: Multi-granularity testing criteria for deep learning systems,” in Proc. 33rd IEEE/ACM Int. Conf. Automated Softw. Eng. (ASE), Sep. 2018, pp. 120–131.

[38] D. Li, Y. Yang, Y. Song, and T. M. Hospedales, “Deeper, broader and artier domain generalization,” in Proc. Conf. Mach. Learn. Res. (ICML), Oct. 2017, pp. 5543–5551.

[39] H. Li, S. J. Pan, S. Wang, and A. C. Kot, “Domain generalization with adversarial feature learning,” in Proc. IEEE/CVF Conf. Vis. Pattern Recognit., Jun. 2018, pp. 5400–5409.

[40] Y. Li, M. Gong, X. Tian, T. Liu, and D. Tao, “Domain generalization via conditional invariant representations,” in Proc. AAAI Conf. Artif. Intell., 2018, vol. 32, no. 1, pp. 3579–3587.

[41] Y. Ganin et al., “Domain-adversarial training of neural networks,” J. Mach. Learn. Res., vol. 17, p. 59, May 2016.
[95] S. Liu et al., “Rotated multi-scale interaction network for referring remote sensing image segmentation,” in Proc. CVPR, Jun. 2024, pp. 26658–26668.

[96] T. Guo, H. Wang, Y. Ma, J. Ji, and X. Sun, “Improving panoptic narrative grounding by harnessing semantic relationships and visual confirmation,” in Proc. AAAI, 2024, pp. 1985–1993.

Yibing Liu received the B.E. degree from the School of Computer Science and Technology, Shandong University, in 2020. He is currently pursuing the Ph.D. degree with the Department of Computer Science, City University of Hong Kong. He is also a Visiting Scholar with The University of Tokyo. His research interests include machine intelligence and explainable machine learning.

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 the National University of Singapore in 2017. He is currently pursuing the Ph.D. degree with the Department of Computer Science, City University of Hong Kong, under the supervision of Dr. Shiqi Wang. From 2017 to 2020, he was a Senior Software Engineer with iQIYI Inc.

Haoliang Li (Member, IEEE) received the B.S. degree in communication engineering from the University of Electronic Science and Technology of China (UESTC) in 2013 and the Ph.D. degree from Nanyang Technological University (NTU), Singapore, in 2018. He is currently an Assistant Professor with the 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 IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, International Journal of Computer Vision, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, ICML, NeurIPS, and CVPR. He received the Wallenberg-NTU Presidential Postdoctoral Fellowship in 2019, the Doctoral Innovation Award in 2019, and the VCIP Best Paper Award in 2020.

Shiqi Wang (Senior Member, IEEE) is currently an Associate Professor with the Department of Computer Science, City University of Hong Kong. He has proposed more than 50 technical proposals to ISO/MPEG, ITU-T, and AVS standards. He has authored or coauthored more than 300 refereed journal articles/conference papers. His research interests include video compression, image/video quality assessment, and image/video search and analysis. He received the Best Paper Award from IEEE VCIP 2019, ICME 2019, IEEE Multimedia 2018, and PCM 2017. His coauthored article received the Best Student Paper Award in the IEEE ICIP 2018. He was a recipient of the IEEE Multimedia Rising Star Award in ICME 2021.