BatchFormer: Learning to Explore Sample Relationships for Robust Representation Learning

Zhi Hou[1], Baosheng Yu[1], Dacheng Tao[2,1]
[1] School of Computer Science, Faculty of Engineering, The University of Sydney, Australia
[2] JD Explore Academy, China
zhou9878@uni.sydney.edu.au, baosheng.yu@sydney.edu.au, dacheng.tao@gmail.com

Abstract

Despite the success of deep neural networks, there are still many challenges in deep representation learning due to the data scarcity issues such as data imbalance, unseen distribution, and domain shift. To address the above-mentioned issues, a variety of methods have been devised to explore the sample relationships in a vanilla way (i.e., from the perspectives of either the input or the loss function), failing to explore the internal structure of deep neural networks for learning with sample relationships. Inspired by this, we propose to enable deep neural networks themselves with the ability to learn the sample relationships from each mini-batch. Specifically, we introduce a batch transformer module or BatchFormer, which is then applied into the batch dimension of each mini-batch to implicitly explore sample relationships during training. By doing this, the proposed method enables the collaboration of different samples, e.g., the head-class samples can also contribute to the learning of the tail classes for long-tailed recognition. Furthermore, to mitigate the gap between training and testing, we share the classifier between with or without the BatchFormer during training, which can thus be removed during testing. We perform extensive experiments on over ten datasets and the proposed method achieves significant improvements on different data scarcity applications without any bells and whistles, including the tasks of long-tailed recognition, compositional zero-shot learning, domain generalization, and contrastive learning. Code is made publicly available at https://github.com/zhihou7/BatchFormer.

1. Introduction

Despite the great success of deep neural networks for representation learning [23, 24], it heavily relies on collecting large-scale training data samples, which turns out to be non-trivial in real-world applications. Therefore, how to form robust deep representation learning under the data scarcity by exploring the sample relationships has received a lot of attention from the community, especially for the tasks without good training data to guarantee the generalization, such as long-tailed recognition [65], zero-shot learning [46], and domain generalization [8]. However, it still remains a great challenge to find a unified, flexible, and powerful way to explore the sample relationships for robust representation learning. An intuitive example revealing the effectiveness of sample relationships is shown in Figure 1.

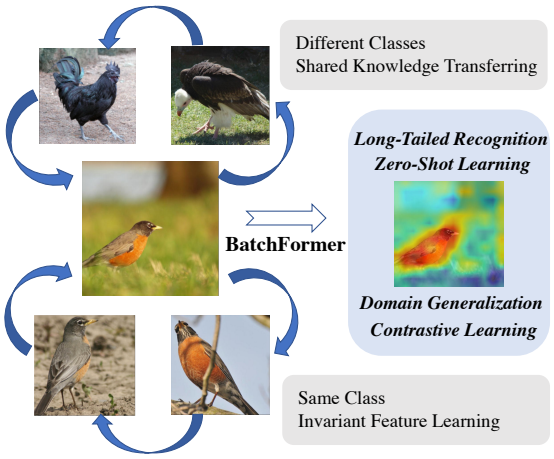

Figure 1. An illustration of the relationships between different images. Specifically, similar classes tend to share similar parts (*e.g.*, cock, robin, vulture share body shape and claw shape) and transferable augmentation (*e.g.*, angles). Therefore, transferring shared knowledge from head/seen classes to tail/unseen classes can facilitate long-tailed/zero-shot learning. In addition, exploring the invariant features between images belonging to the same class is also helpful for robust representation learning with a few samples.

Among recent data scarcity learning methods, sample relationships have been intensively explored using an explicit scheme from either regularization [18, 40, 54, 70] or knowledge transfer [63, 66, 78]. Specifically, a simple yet very effective way is to directly generate new data

samples/domains from existing training data [40], such as mixup [70], copy-paste [18], crossgrad [54], and compositional learning [2, 28, 37, 46]. Another way is to transfer knowledge between data samples, e.g., 1) transferring meta knowledge between head and tail classes for long-tailed recognition [43, 66]; 2) transferring the knowledge from seen classes to unseen classes for zero-shot learning [46, 67]; and 3) transferring invariant knowledge for domain generalization [1, 31, 49]. However, the above-mentioned methods explore sample relationships from either the input or output of deep neural networks, failing to enable deep neural networks themselves with the ability to explore sample relationships, i.e., there is no interaction from the view of batch dimension.

Enabling the learning on the batch dimension is not easy for deep neural networks due to the training and inference gap, i.e., we do not always have a mini-batch of data samples during testing. For example, batch normalization requires to always keep mini-batch training statistics, where the running mean and variance are then used to normalize testing samples [32]. Another example uses the feature memory to keep category centers during training, which is then used to enhance the tail/unseen categories during testing [43, 78]. Therefore, to explore sample relationships for robust representation learning, we propose to empower the deep neural networks with structural advances for sample relationship learning. Specifically, we try to capture and model the sample relationships in each mini-batch of training data samples by introducing a transformer into the batch dimension, and we refer to it as the Batch Transformer or BatchFormer. Furthermore, to mitigate the gap between training and testing, we utilize a shared classifier before and after the BatchFormer module to enforce the batch-invariant learning. By doing this, the BatchFormer module is only required during training, i.e., we do not need to change the inference structures of deep neural networks.

From the perspective of optimization, BatchFormer enables the information propagation of all features of the mini-batch samples. Therefore, all samples can contribute to the learning on any object categories, and the insight might be that this implicitly enriches current training samples with hallucinated features from the whole mini-batch (*e.g.*, the shared parts between two categories). For example, in long-tailed recognition, the hallucinated features may improve the feature space of the tail classes. Meanwhile, the loss function also emphasizes on the rare classes via propagating larger gradients of rare classes on other features in the mini-batch. In particular, we also find that BatchFormer brings two obvious changes to the learned representation, *i.e.*, it effectively facilitates the deep model to learn 1) comprehensive representations by focusing on almost all different parts of the object; and 2) invariant representations by focusing on the object itself rather than the complex background cues (See more empirical evaluations in appendix).

In this paper, our main contributions can be summarized as follow: 1) we propose to explore sample relationships from the perspective of the internal structure of deep neural networks; 2) we devise a simple yet effective module termed as BatchFormer, which is a plug-and-play module to explore sample relationships in each mini-batch; and 3) without any bells and whistles, we perform extensive experiments to demonstrate the effectiveness of BatchFormer in a variety of visual recognition tasks, including long-tailed recognition, zero-shot learning, domain generalization, and self-supervised representation learning.

2. Related Work

Sample Relationship. There are diverse and firm relationships among different samples, which have been widely used via various types of strategies [27, 42, 45, 70]. Zhang *et al*. [70] propose to regularize the network to favor simple linear behavior in-between training examples with mixup. Mixup [70] merely considers the linear transformation between examples, while we investigate the relationship among examples in a non-linear way. The compositionality of samples has inspired massive approaches to improve the generalization of few-shot and zero-shot learning [22, 28, 46, 57], where the shared parts/attributes among different samples have been explored via prior label relationship knowledge. There are also some approaches via investigating sample/class relationships to conduct transductive inference [27, 42, 43, 45], *e.g.*, transductive few-shot classification [42], meta embedding [43, 78]. However, those approaches require to conduct inference with multiple samples (*e.g.*, query set, or bank features). Meanwhile, massive domain generalization methods [1, 31, 49] aims to find casual/invariant representations across domains, which we think internally utilizes the relationship among samples of the same class but different domains. However, those techniques are diverse and complex. By contrast, we propose to investigate relationships for robust representation learning in a simple yet effective way, and thus benefit those challenging tasks simultaneously.

Data Scarcity Learning. Learning with imperfect training data has turned out to be very challenging, which has been explored in a variety of data scarcity tasks. A very important problem is long-tailed recognition, where the data from different classes usually exhibit a long-tailed distribution: a large portion of classes have very few instances. Current long-tailed approaches can be roughly categorized into distribution re-balancing methods (*e.g.*, re-sampling [10, 20, 21], re-weighting [6, 15, 34, 56, 72]), ensemble of diverse branches/experts [7, 65, 73], and knowledge transfer [25, 43, 63, 66, 78]. Many knowledge transfer approaches have been developed for long-tailed recognition via meta learning [66], memory features [43, 78],

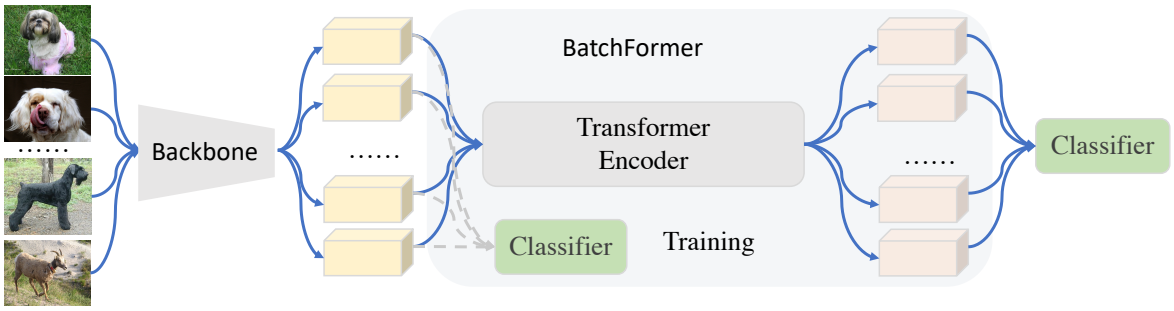

Figure 2. The main framework of representation learning with the proposed BatchFormer. Specifically, we apply a BatchFormer module between the feature extractor (*e.g.*, ResNet) and the classifier layer to explore the samples relationships. Furthermore, with a shared classifier before and after the BatchFormer during training for batch-invariant learning, we can then remove BatchFormer during testing.

and virtual data generation [25, 29, 63]. Nevertheless, those works usually depend on complex memory bank [43, 78], or complex learning strategies [66] or additional distillation step [25]. Another data scarcity tasks include zero-shot learning and domain generalization [5, 62, 75], which require to recognize new categories with unseen training data. Specifically, zero-shot learning aims to recognize unseen classes that do not training samples, while domain generalization targets at generalizing classes of seen domains to unseen domains. Current zero-shot learning approaches [38, 48, 64, 67] usually transfer knowledge of seen classes to unseen classes via modern techniques (*e.g.*, graph network [46], data generalization [79], and compositional learning [28, 46]) for unseen classes recognition. Recently, compositional zero-shot learning [2, 28, 37, 46] have been widely explored in different tasks, we thus mainly evaluate the proposed BatchFormer on compositional zero-shot learning [46]. Domain generalization techniques usually include data augmentation [75, 76], meta learning [3, 71], and disentangled/invariant representation learning [1, 9, 49, 50]. BatchFormer facilitates invariant representation learning, and thus illustrates effectiveness on multiple datasets.

Contrastive Self-Supervised Learning. Contrastive Learning [19] has increasingly achieved great success in computer vision for self-supervised learning, *e.g.*, [47], [26], [12, 13, 23] and [11]. Contrastive Learning aims to learn representations that attract similar samples and dispel different samples, where the similar samples and different samples are known according to some priors. However, we implicitly mine the relationships with Transformer Encoder network. Specifically, we also demonstrates the effectiveness of BatchFormer in contrastive learning, *e.g.*, MoCo-v2 [12] and MoCo-v3 [13].

3. Method

In this section, we first provide an overview of the main deep representation learning framework with the proposed BatchFormer module. We then introduce the proposed BatchFormer module in detail, including the encoder and the shared classifier. Lastly, we discuss the insights behind the proposed BatchFormer from the view of gradient flow.

3.1. Overview

The relationships between different samples are various and complex, while previous approaches have explored sample relationships using a straightforward way, such as the joint manipulation of different input images and the knowledge transfer using meta embedding or loss functions. Among existing methods, sample relationships are usually required to be defined in an explicit scheme before they can be used for learning, thus failing to automatically learn sample relationships for representation learning.

We consider sample relationships from a learning perspective, i.e., we aim to enable deep neural networks themselves with the ability to learn sample relationships from each mini-batch sample during the end-to-end deep representation learning. The main deep representation learning framework with the proposed new module is shown in Figure 2. Specifically, a backbone network is first used to learn representations for individual data samples, i.e., there is no interaction between different samples in each mini-batch. After this, a new module is introduced to model the relationships between different samples by using the cross-attention mechanism in transformer, and we thus referred to it as Batch Transformer or BatchFormer module. The output of BatchFormer is then used as the input of the final classifier. To fulfill the gap between training and testing, we also utilize an auxiliary classifier before the BatchFormer module, i.e., by sharing the weights between the final classifier and the auxiliary classifier, we are able to transfer the knowledge learned from sample relationships to the backbone and auxiliary classifier. Therefore, during testing we can remove the BatchFormer and directly use the auxiliary classifier for classification.

Algorithm 1: Pytorch Code of BatchFormer.

```
def BatchFormer(x, y, encoder, is_training):
    # x:  input features with the shape [N, C]
    # encoder:  TransformerEncoderLayer(C,4,C,0.5)
    if not is_training:
        return x, y
    pre_x = x
    x = encoder(x.unsqueeze(1)).squeeze(1)
    x = torch.cat([pre_x, x], dim=0)
    y = torch.cat([y, y], dim=0)
    return x, y
```

3.2. BatchFormer

In this subsection, we introduce the detailed structures of the proposed BatchFormer. Specifically, the proposed BatchFormer module utilizes a stack of multiple transformer encoder layers to model the relationships between different samples.

Transformer Encoder. The transformer encoder includes multihead self-attention (MSA) and MLP blocks. A Layernorm (LN) is used after each block. Let $X \in R^{N \times C}$ denote a sequence of input features, where N is the length of the sequence and C indicates the dimension of input features. We then have the output of the transformer encoder as follows,

$$\hat{X}_l = LN(MSA(X_{l-1}) + X_{l-1}), \tag{1}$$

$$X_l = LN(MLP(\hat{X}_l) + \hat{X}_l), \tag{2}$$

where l indicates the index of layers in the transformer encoder. The multi-head attention layers have been widely used to model the relationships from channel and spatial dimensions [16, 30, 60]. Therefore, we argue that it can also be extended to explore the relationships in the batch dimension. As a result, different from typical usage of transformer layers, the input of BatchFormer will be first reshaped to enable the transformer layers working on the batch dimension of the input data. By doing this, the self-attention mechanism in transformer layers then becomes the cross-attention between different samples for BatchFormer.

Shared Classifier. Since we can not assume batch statistics for testing, such as sample relationships, there might be a gap between the features before and after the BatchFormer module. That is, we can not perform inference on new samples by directly removing the BatchFormer. Therefore, apart from the final classifier, we also introduce a new auxiliary classifier to not only learn from the final classifier but also keep consistent with the features before the BatchFormer. To achieves this, we simply share the parameters/weights between the auxiliary classifier and the final classifier. We refer to this simple yet effective strategy as "shared classifier". With the proposed "shared classifier", we can thus remove the BatchFormer module during testing, while still benefiting from the sample relationship learning using BatchFormer.

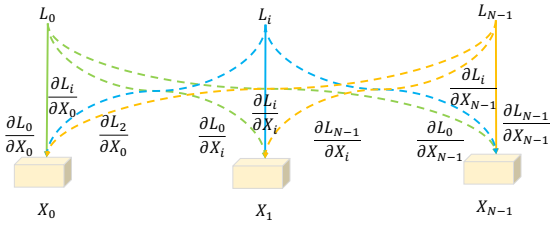

Figure 3. Gradient propagation with BatchFormer in a mini-batch. Dashed lines represent new gradient propagation among samples.

The proposed BatchFormer is a plug-and-play module for robust deep representation learning. During optimization, the proposed BatchFormer modules can be jointly optimized with other deep learning backbone networks (including both CNNs and Vision Transformers) in an end-to-end way. The proposed BatchFormer is also very easy to implement using typical deep learning softwares,. For example, we show how to implement BatchFormer with several lines of PyTorch code in Algorithm 1.

3.3. BatchFormer: A Gradient View

To better understand how the proposed BatchFormer helps representation learning by exploring sample relationships, we also provide an intuitive explanation from the perspective of gradient propagation for optimization. Intuitively, without BatchFormer, all losses only propagate gradients on the corresponding samples and categories, *i.e.*, one-to-one, while there are gradients on other samples with BatchFormer (the dashed line) as illustrated in Figure 3. Specifically, given samples $X(X = X_0, X_1, X_i, ..., X_{N-1})$ and the corresponding losses $L_0, L_1, L_i, ..., L_{N-1}$ in the mini-batch, with BatchFormer, we then have

$$\frac{\partial L_i}{\partial X} = \frac{\partial L_i}{\partial X_i} + \sum_{j \neq i}^{N-1} \frac{\partial L_i}{\partial X_j}. \tag{3}$$

That is, BatchFormer brings new gradient terms $\frac{\partial L_i}{\partial X_j}$, where $i \neq j$. From a perspective of gradient optimization, L_i also optimizes the network according to sample $X_j (j \neq i)$, that is a significant difference compared to the model without BatchFormer. In other word, $X_j (j \neq i)$ can be regarded as a virtual sample [25, 70] of y_i, where y_i is the label of X_i. We consider that both BatchFormer and Mixup [70] can be regarded as data-dependent augmentations. BatchFormer implicitly draws virtual examples from the vicinity distribution of samples via cross-attention module. From this perspective, BatchFormer has implicitly augmented $N - 1$ virtual samples for each label y_i via the relationship modeling among samples in the mini-batch. Previous approaches [3, 79] have demonstrated data augmentation is helpful for long-tailed recognition [25, 63], zero-shot

learning [79], and domain generalization [75, 76]. the virtual samples are largely helpful for tail classes since those classes lack of samples. Our gradients analysis in Section 5 also demonstrates the tail classes have larger gradients on other samples compared to head classes.

4. Experiments

In this section, we perform extensive experiments to demonstrate the effectiveness of BatchFormer for a variety of data scarcity learning tasks, including long-tailed recognition, zero-shot learning, domain generalization and contrastive learning. Appendix provides more results of other tasks, *e.g.*, image classification (include Visual Transformer [16, 58]), and more ablation studies.

4.1. Long-Tailed Recognition

Datasets. We use four popular datasets for long-tailed recognition as follows: 1) CIFAR-100-LT has 50,000 training images and 10,000 validation images with 100 categories. 2) ImageNet-LT [43] contains 115.8K images of 1000 classes from ImageNet 2012. The number of images in each class ranges from 5 to 1,280; 3) iNaturalist 2018 [59] is a large-scale fine-grained dataset with 437.5K images from 8,142 categories; 4) Places-LT [43] is a long-tailed scene classification dataset derived from the Places dataset [74] with 184.5K images from 365 categories with class instances ranging from 5 to 4,980.

Baselines. We use the following three baseline methods: Balanced Softmax [52], RIDE [65] and PaCo [14]. If not otherwise stated, we follow the same settings used in previous methods, and the proposed BatchFormer is removed during testing. Particularly, there is a small difference when comparing with RIDE [65]. we train the network with a batch size of 400 on 4 V100 GPUs for 100 epochs with an initial learning rate of 0.1 on ImageNet-LT and 0.2 on iNaturalist 2018, respectively. The learning rare is decayed with cosine schedule on iNaturalist 2018. We report the performance of BatchFormer based on 3 experts RIDE. All experiments are conducted on V100 GPU with PyTorch. See other details in Appendix A.

Results on CIFAR-100-LT. Table 1 demonstrates the proposed BatchFormer module is orthogonal to the state-of-the-art methods, *e.g.*, Balanced Softmax and Paco. We notice that BatchFormer largely improves Balanced Softmax by 2.4% for few classes when imbalance ratio is 100, and by 1.8% on medium classes and by 1.2% on few classes respectively when imbalance ratio is 200. Besides, the results of PaCo on imbalance ratio 200 also increase by 1.5% on medium classes and 0.7% on few classes with BatchFormer. For ratio 100, BatchFormer mainly improves many classes since Paco has achieved good performance on few classes. Please refer to Appendix for more explanations (*i.e.*, the comparison without PaCo loss, but with strong data augmentation). Overall, BatchFormer improves the recognition of tail classes while maintaining the performance of head classes. Meanwhile BatchFormer is pluggable to current popular methods on CIFAR-100-LT.

Table 1. Illustration of imbalance ratio 100 and 200 on CIFAR-100-LT. $*$ means we train the method with the released code in one stage (*e.g.*, balanced softmax [52]). Med means Medium category. RIDE* means we use 3 experts.

Method	100				200			
	All	Many	Med	Few	All	Many	Med	Few
RIDE* [65]	48.0	68.1	49.2	23.9	-	-	-	-
Balanced$*$ [52]	50.7	68.0	49.7	31.9	46.4	70.0	51.5	24.3
+ BatchFormer	**51.7**	**68.4**	49.3	**34.3**	**47.5**	**70.2**	**53.3**	**25.5**
Paco [14]	51.9	63.9	**53.0**	**36.5**	47.1	68.1	51.5	27.5
+ BatchFormer	**52.4**	**68.4**	52.1	34.0	**47.8**	68.1	**53.0**	**28.2**

Table 2. Illustration of ResNet-10 and ResNet-50 on ImageNet-LT. $*$ means we train the net work with the released code by one-stage. RIDE-3e means we use three experts in RIDE.

Method	ResNet-10				ResNet-50			
	All	Many	Med	Few	All	Many	Med	Few
OLTR [43]	35.6	43.2	35.1	18.5	-	-	-	
LFME [68]	38.8	47.0	37.9	19.2	-	-	-	-
Balanced$*$ [52]	41.0	52.6	38.3	18.0	50.1	61.1	47.5	27.6
+ BatchFormer†	**43.2**	**52.8**	**40.7**	**24.9**	**51.1**	**61.4**	**47.8**	**33.6**
RIDE-3e $*$ [65]	44.7	57.0	40.3	25.5	53.6	64.9	50.4	33.2
+ BatchFormer	**45.7**	56.3	**42.1**	**28.3**	**54.1**	64.3	**51.4**	**35.1**
PaCo [14]	-	-	-	-	57.0	64.8	55.9	39.1
+ BatchFormer	-	-	-	-	**57.4**	62.7	**56.7**	**42.1**
two stage								
RIDE-3e [65]	45.9	57.6	41.7	28.0	54.9	66.2	51.7	34.9
+ BatchFormer	**47.6**	55.3	**45.5**	**33.3**	**55.7**	64.6	**53.4**	**39.0**

Results on ImageNet-LT. As illustrated in Table 2, BatchFormer largely improves Balanced Softmax by 2.4% on medium classes and **6.9%** on few classes respectively under ResNet-10 backbone. Meanwhile, under ResNet-50 backbone, BatchFormer largely improves Balanced Softmax on the Few category by **5%**.

When BatchFormer is applied in RIDE [65], the results on medium classes and few classes increase by 1.8% and 2.8% respectively under ResNet-10 backbone. BatchFormer also increases medium and few classes by 1% and 1.9% respectively under ResNet-50. BatchFormer achieves clear improvement overall, while the performance on many classes drops a bit. Furthermore, BatchFormer also effectively improves RIDE under two-stage training strategy (RIDE uses a larger model to teach small model with distilling loss). Here, different from RIDE, we use a pre-trained model (the same model) to initialize the model and train the model again with BatchFormer (See details in Appendix).

PaCo [14] is recently introduced for Long-Tailed Recog-

Table 3. Illustration on iNaturalist 2018. $*$ means we train the method with the released code in one stage. RIDE-3e means RIDE with 3 experts.

Methods	All	Many	Medium	Few
BBN [73]	66.3	49.4	70.8	65.3
cRT [36]	65.2	69.0	66.0	63.2
RIDE-3e [65]	72.2	70.2	72.2	72.7
PaCo [14]	73.2	-	-	-
RIDE-3e$*$ [65]	72.5	68.1	72.7	73.2
+ Batchformer	**74.1**	65.5	**74.5**	**75.8**

Table 4. Illustration of Places-LT (Backbone is ResNet-152). $*$ means the result that we train the method with the released code.

Methods	All	Many	Medium	Few
OLTR [43]	35.9	44.7	37.0	25.3
τ-normalized [36]	37.9	37.8	40.7	31.8
BALMS* [52]	37.8	41.4	38.8	29.1
+ Batchformer	**38.2**	39.5	38.3	**35.7**
PaCo [14]	41.2	37.5	**47.2**	33.9
+ Batchformer	**41.6**	**44.0**	43.1	33.7

nition with Supervised Contrastive Learning. We also find BatchFormer is able to facilitate ImageNet-LT on medium classes and few classes. Noticeably, PaCo uses a strong data augmentation strategy from Supervised Contrastive Learning with 400 training epochs. We think data augmentation limits the improvement of BatchFormer on ImageNet-LT. Particularly, Appendix also shows BatchFormer achieves comparable results on balanced Full ImageNet.

Results on iNaturalist 2018. We mainly evaluate BatchFormer on RIDE since Balanced Softmax has limited performance on iNaturalist 2018 and PaCo requires over 36 GPU days to converge (See Appendix). We train RIDE with cosine decay learning-rate scheduler by one-stage and achieve a strong baseline with 72.5% (better than the reported) as illustrated in Table 3. Particularly, BatchFormer further improves the Medium category and Few Category by 1.8% and 2.6% respectively. Here, for a fair comparison to previous work, we mainly evaluate BatchFormer RIDE with 3 experts which has similar GFlops to previous work.

Results on Places-LT. Table 4 illustrates BatchFormer improves BALMS on Few category. Besides, BatchFormer effectively improves PaCo [14] which is the new state-of-the-art on Places-LT. Here, different from CIFAR-100-LT and ImageNet-LT, BatchFormer mainly improves Many category. This might be because the result of PaCo on Many category is very worse and BatchFormer can re-balance the imbalanced datasets.

4.2. Zero-Shot Learning

We evaluate BatchFormer on compositional zero-shot learning [46]. All experiments are evaluated with one V100 GPU. The learning rare, epochs, optimizer are fully similar to [46]. See details in Appendix A.

Datasets. The experiments are performed on three datasets: 1) MIT-States [33] consists of 30,000 training images of natural objects with 1,262 seen compositions (23.8 image per composition, 115 states and 245 objects), and 13,000 test images with 400 seen compositions and 400 unseen compositions; 2) UT-Zappos [69] includes 23,000 training images of shoes catalogue and we use the splits from [51]. UT-Zappos has 83 seen compositions for training (277.1 images per composition, 16 states, 12 objects) and 18 seen compositions and unseen compositions in test set; 3) C-GQA [46] provides 26,000 training images with 6,963 seen compositions (3.7 images per composition, 453 states, 870 objects) and 3,000 test images with 18 seen compositions and unseen compositions.

Metrics. We adopt the evaluation protocol of [51] and report the Area Under the Curve (AUC) (in %) between the accuracy on seen and unseen compositions. Similar to [46], we also report unseen accuracy and seen accuracy, as well as the best harmonic mean.

Table 5 demonstrates BatchFormer effectively improves the AUC and HM among all the datasets compared to the baseline. Particularly, for a fair comparison, we use the released code under the same setting to reproduce [46] as the baseline. We notice BatchFormer mainly improves the Seen category on MIT-States and C-GQA, while BatchFormer largely improves the unseen category by nearly **5%**. This might be because the number of seen composition instances on MIT-States and C-GQA is few, *e.g.* 23.8 image per seen composition on MIT-States and 3.7 images per seen composition. In other words, the recognition of seen compositions on MIT-States and C-GQA is few-shot learning. We think BatchFormer on the two datasets mainly finds invariant features among images of the same class.

4.3. Domain Generalization

By exploring sample relationships of the same class, it is easier to find the invariant features and thus improve the domain generalization. We first demonstrate BatchFormer effectively improves the baseline (without domain generalization techniques) on PACS [39] under ResNet-18. Then, we apply BatchFormer to recent domain generalization, *e.g.*, SWAD [8], and shows the effectiveness of BatchFormer.

Datasets. We illustrates the application of BatchFormer on several domain generalization datasets: 1) PACS [39] covers 7 object categories and 4 domains (Photo, Art Paintings, Cartoon and Sketches), 2) VLCS [17] (4 domains, 5 classes, 10,729 images), OfficeHome [61] (4 domains, 65 classes and 15,588 images), and TerraIncognita [4] (4 domains, 10 classes and 24,788 images).

Details. For baseline, we train the network under ResNet-18 with SGD (30 epochs, initial learning rate

Table 5. Illustration of Batchformer on Compositional Zero-Shot Learning. * means we use the released code to reproduce the results. S means seen, U means unseen, s means state and o means object. The results of [41] are copied from [46].

Method	MIT-States						UT-Zap50K						C-GQA					
	AUC	HM	S	U	s	o	AUC	HM	S	U	s	o	AUC	HM	S	U	s	o
CompCos [44]	4.5	16.4	25.3	24.6	27.9	31.8	28.7	43.1	59.8	62.5	44.7	73.5	-	-	-	-	-	-
CGE [46]	6.5	21.4	32.8	28.0	30.1	34.7	33.5	60.5	64.5	71.5	48.7	76.2	3.6	14.5	31.4	14.0	15.2	30.4
CGE*	6.3	20.0	31.6	27.3	30.3	34.5	31.5	46.5	60.3	64.5	46.3	74.4	3.7	14.9	30.8	14.7	**15.8**	29.0
+BatchFormer	**6.7**	**20.6**	**33.2**	**27.7**	**30.8**	**34.7**	**34.6**	**49.0**	**62.5**	**69.2**	**49.7**	**75.6**	**3.8**	**15.5**	**31.3**	14.7	15.3	**30.0**

Table 6. Illustration of BatchFormer for Domain Generalization under different works on PACS [39].

Methods	art_paint	cartoon	sketches	photo	Avg.
Baseline	79.9± 1.0	73.0±1.5	67.7± 3.0	95.7±0.4	79.1
+BatchFormer	**80.4**±0.2	**73.8**±2.0	**68.6**±1.8	**96.3**±0.2	**79.8**
CORAL [55]	79.2±1.7	**75.5** ±1.1	71.4±3.1	94.7±0.3	80.2
+BatchFormer	**80.6**±0.9	74.7±1.9	**73.1**±0.3	**95.1**±0.3	**80.9**
MixStyle [77]	81.7±0.1	**76.8**±0.0	80.8±0.0	93.1±0.0	83.1
+BatchFormer	**84.8** ±0.4	75.3±0.0	**81.1** ±0.4	**93.6**±0.0	**83.7**

Table 7. Illustration of BatchFormer for Domain Generalization based on recent work [8] (ResNet-18). We show the average results of different domains. Terra is TerraIncognita. See more results in Appendix.

Methods	PACS	VLCS	OfficeHome	Terra
SWAD* [8]	82.9	76.3	62.1	42.1
+ BatchFormer	**83.7**	**76.9**	**64.3**	**44.8**

Table 8. Illustration of BatchFormer for Contrastive Learning (MoCo [12, 13, 23] under ResNet50) on linear classification.

Methods	Epochs	Top-1	Top-5
MoCo-v2 [12]	200	67.5	-
MoCo-v2 [12]+BatchFormer	200	**68.4**	88.5
MoCo-v3 [13]	100	68.9	-
MoCo-v3 [13]+BatchFormer	100	**69.8**	89.5

0.001), and drop the learning rate at 24 epochs. We also use the popular data augmentations, *e.g.*, flip, color jiter and scale. We train the network 5 times and report the average. Others methods are compared based on [35]. For a fair comparison with SWAD [8], we follow the optimization methods of [8] and use the released code of [8] to evaluate BatchFormer on ResNet-18. See more details in Appendix.

Table 6 illustrates BatchFormer consistently improves baseline , CORAL [55] and MixStyle [77]. BatchFormer also clearly improves recent work [8] on four datasets in Table 7. Particularly, BatchFormer improves [8] by over 2% on OfficeHome and TerraIncognita. This illustrates BatchFormer is able to facilitate invariant representation learning, and improve the generalization across domains.

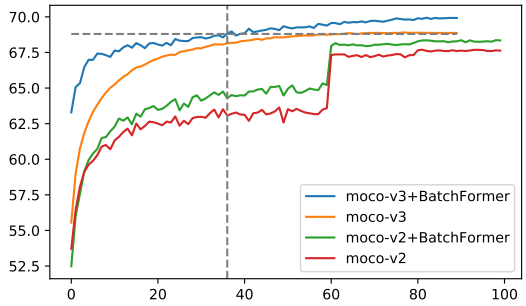

Figure 4. Convergence of BatchFormer under linear classification. The batch size and optimizer are the same as [13].

4.4. Self-Supervised Learning

Contrastive Learning aims to learn representations that attract similar samples and dispel different samples, while BatchFormer builds a Transformer Network among samples to implicitly explore the sample relationships for representation learning. BatchFormer can also be applied to Contrastive Learning. We mainly evaluate BatchFormer with MoCo-v2 [12] and MoCo-v3 [13] on linear classification protocol. Object detection result is provided in Appendix.

Table 8 shows BatchFormer is also able to improve the representation learning of self-supervised learning, *e.g.*, BatchFormer consistently improves MoCo-v2 and MoCo-v3 by around 1.% on ImageNet. Figure 4 shows the linear classification of BatchFormer pre-trained model convergences faster, *e.g.*, BatchFormer pre-trained model achieves the performance of moco-v3 with only 38 epochs.

4.5. Ablation Studies

Batch size. The batch size represents the size of Transformer. Table 9 shows BatchFormer is less sensitive to the batch size when batch size is less than 128. We find batch size 512 achieves better performance on Few category.

Shared Classifier. Interestingly, Table 10 demonstrates BatchFormer without shared classifier achieves similar performance on three categories, while BatchFormer with shared classifier maintains the performance on Many category. This demonstrates the effectiveness of shared classifier and the ability of re-balancing of BatchFormer.

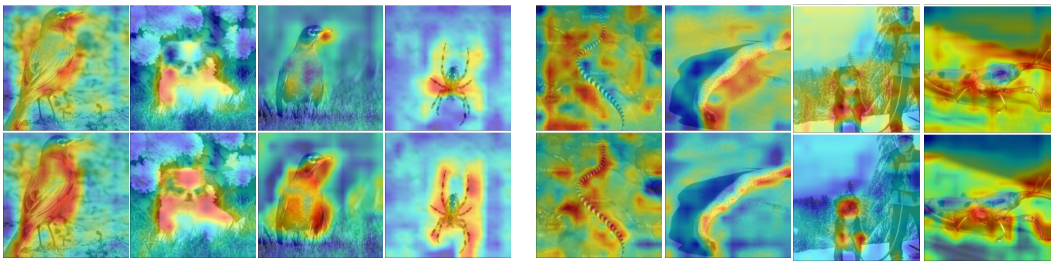

Figure 5. Grad-Cam demonstration of BatchFormer on low-shot test images based on [52]. The first row is baseline, while the second row is BatchFormer. The left images show BatchFormer enables the model pay attention on more details when the scene is simple and clean, while the right images show BatchFormer faciliates the model ignore the spurious correlation in the image. More figures are in Appendix.

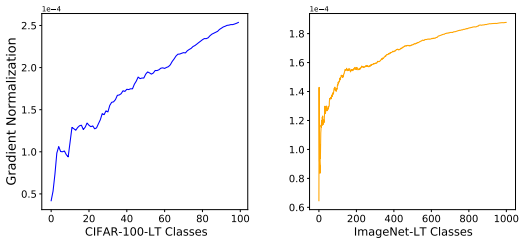

Figure 6. The gradient of each class to other images in mini-batch on CIFAR-100-LT and ImageNet-LT (based on [52]). For each class, we obtain the gradient normalization of the loss of each label to other images in the mini-batches, and then average the gradients of each class in the test set. The classes are sorted by descending order according to the number of instances in the training set.

Table 9. Ablation Studies of Batch Size based on Balanced Softmax [52]. The backbone is ResNet-10 under one GPU. learning rare is linearly decreased with batch size.

Batch Size	16	32	64	128	256	512
All	43.2	**43.6**	43.3	43.2	42.4	42.5
Many	**53.8**	53.7	52.9	52.8	52.2	52.0
Medium	40.2	40.7	**40.8**	40.7	39.5	39.6
Few	23.9	24.9	25.3	24.9	25.2	**25.8**

Table 10. Ablation Studies on ImageNet-LT based on Balanced Softmax [52]. The backbone is ResNet-50.

Method	All	Many	Medium	Few
BatchFormer	**50.9**	**60.7**	**47.7**	34.1
w/o Shared Classifier	42.4	41.3	43.3	**42.2**

5. Visualization Analysis

Visualized Comparison. We illustrate the visualized comparison with Grad-CAM [53] between Baseline and BatchFormer in Figure 5. We find BatchFormer focuses on more details of objects and ignores spurious correlations. On the one hand, when the image includes complex scence with many disturbing factors, BatchFormer effectively improves the attention of the network on the corresponding object regions (*e.g.*, sea snake on the sandbeach, dog on the snow and insect on the leaf in Figure 5 right). On the other hand, BatchFormer also pays more attention on regions of the object when the scene is clear (*e.g.*, the bird, dog, and spider in Figure 5 left). See more examples in Appendix.

Gradients Analysis. BatchFormer has increased new gradient backward: the loss of each label has the gradients on other images. In other words, we have implicitly augmented the samples for the class of each image in the mini-batches. The other images in the mini-batch can also be regarded as the virtual instances of current image class. The gradient is firmly related to the effect of each image label on other images. Figure 6 illustrates the rarer the class is, the larger the gradients of the class have on other images in the mini-batch. Thus, BatchFormer actually utilizes the other images to facilitate low-shot recognition via increasing gradients of few-shot labels on other images.

6. Conclusion and Future work

We propose to enable deep neural networks themselves with the ability to explore the sample relationships from each mini-batch. Specifically, we regards each image (batch dimension) in the mini-batch as a node of a sequence, and then build a Transformer Encoder Network among the images to mine the relationships among the images in the mini-batch. BatchFormer enables the gradient propagation of each label to all images in the mini-batch, which can be regarded as virtual sample augmentation, and thus improve the representation learning. We further introduce a shared classifier before and after the BatchFormer during training, which can thus be removed during testing. We demonstrate the effectiveness of BatchFormer on over ten datasets and BatchFormer achieves significant improvements on different tasks, including long-tailed recognition, zero-shot learning, domain generalization, and contrastive learning.

Limitations The improvement of current BatchFormer on models with strong data augmentation and balanced distribution is limited.

Acknowledgements Dr. Baosheng Yu and Mr. Zhi Hou are supported by ARC FL-170100117.

References

[1] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019. 2, 3

[2] Yuval Atzmon, Felix Kreuk, Uri Shalit, and Gal Chechik. A causal view of compositional zero-shot recognition. In *NeurIPS*, 2020. 2, 3

[3] Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. Metareg: Towards domain generalization using meta-regularization. In *NIPS*, volume 31, pages 998–1008, 2018. 3, 4

[4] Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In *ECCV*, pages 456–473, 2018. 7

[5] Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from several related classification tasks to a new unlabeled sample. *NIPS*, 24:2178–2186, 2011. 3

[6] Jonathon Byrd and Zachary Lipton. What is the effect of importance weighting in deep learning? In *ICML*, pages 872–881. PMLR, 2019. 2

[7] Jiarui Cai, Yizhou Wang, and Jenq-Neng Hwang. Ace: Ally complementary experts for solving long-tailed recognition in one-shot. In *ICCV*, 2021. 2

[8] Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. Swad: Domain generalization by seeking flat minima. In *NeurIPS*, 2021. 1, 6, 7

[9] Prithvijit Chattopadhyay, Yogesh Balaji, and Judy Hoffman. Learning to balance specificity and invariance for in and out of domain generalization. In *ECCV*, pages 301–318. Springer, 2020. 3

[10] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002. 2

[11] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, pages 1597–1607. PMLR, 2020. 3

[12] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 3, 7

[13] Xinlei Chen*, Saining Xie*, and Kaiming He. An empirical study of training self-supervised vision transformers. In *CVPR*, 2021. 3, 7

[14] Jiequan Cui, Zhisheng Zhong, Shu Liu, Bei Yu, and Jiaya Jia. Parametric contrastive learning. In *ICCV*, 2021. 5, 6

[15] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *CVPR*, pages 9268–9277, 2019. 2

[16] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2020. 4, 5

[17] Chen Fang, Ye Xu, and Daniel N Rockmore. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *ICCV*, pages 1657–1664, 2013. 6

[18] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D Cubuk, Quoc V Le, and Barret Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation. In *CVPR*, pages 2918–2928, 2021. 1, 2

[19] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, volume 2, pages 1735–1742. IEEE, 2006. 3

[20] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing*, pages 878–887. Springer, 2005. 2

[21] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009. 2

[22] Ju He, Adam Kortylewski, and Alan Yuille. Compas: Representation learning with compositional part sharing for few-shot classification. *arXiv preprint arXiv:2101.11878*, 2021. 2

[23] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020. 1, 3, 7

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1

[25] Yin-Yin He, Jianxin Wu, and Xiu-Shen Wei. Distilling virtual examples for long-tailed recognition. In *ICCV*, 2021. 2, 3, 4, 5

[26] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *ICLR*, 2019. 3

[27] Ruibing Hou, Hong Chang, Bingpeng Ma, Shiguang Shan, and Xilin Chen. Cross attention network for few-shot classification. *NeurIPS*, 2019. 2

[28] Zhi Hou, Xiaojiang Peng, Yu Qiao, and Dacheng Tao. Visual compositional learning for human-object interaction detection. In *ECCV*, pages 584–600. Springer, 2020. 2, 3

[29] Zhi Hou, Baosheng Yu, Yu Qiao, Xiaojiang Peng, and Dacheng Tao. Detecting human-object interaction via fabricated compositional learning. In *CVPR*, pages 14646–14655, 2021. 2

[30] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, June 2018. 4

[31] Zeyi Huang, Haohan Wang, Eric P Xing, and Dong Huang. Self-challenging improves cross-domain generalization. In *ECCV*, pages 124–140. Springer, 2020. 2

[32] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456. PMLR, 2015. 2

[33] Phillip Isola, Joseph J Lim, and Edward H Adelson. Discovering states and transformations in image collections. In *CVPR*, pages 1383–1391, 2015. 6

[34] Muhammad Abdullah Jamal, Matthew Brown, Ming-Hsuan Yang, Liqiang Wang, and Boqing Gong. Rethinking class-balanced methods for long-tailed visual recognition from a

domain adaptation perspective. In *CVPR*, pages 7610–7619, 2020. 2

[35] Junguang Jiang, Baixu Chen, Bo Fu, and Mingsheng Long. Transfer-learning-library. https://github.com/thuml/Transfer-Learning-Library, 2020. 7

[36] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. *ICLR*, 2020. 6

[37] Keizo Kato, Yin Li, and Abhinav Gupta. Compositional learning for human object interaction. In *ECCV*, pages 234–251, 2018. 2, 3

[38] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, pages 951–958. IEEE, 2009. 3

[39] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *ICCV*, pages 5542–5550, 2017. 6, 7

[40] Shuang Li, Kaixiong Gong, Chi Harold Liu, Yulin Wang, Feng Qiao, and Xinjing Cheng. Metasaug: Meta semantic augmentation for long-tailed visual recognition. In *CVPR*, pages 5212–5221, 2021. 1, 2

[41] Yong-Lu Li, Yue Xu, Xiaohan Mao, and Cewu Lu. Symmetry and group in attribute-object compositions. In *CVPR*, pages 11316–11325, 2020. 7

[42] Yanbin Liu, Juho Lee, Minseop Park, Saehoon Kim, Eunho Yang, Sung Ju Hwang, and Yi Yang. Learning to propagate labels: Transductive propagation network for few-shot learning. 2018. 2

[43] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X. Yu. Large-scale long-tailed recognition in an open world. In *CVPR*, 2019. 2, 3, 5, 6

[44] Massimiliano Mancini, Muhammad Ferjad Naeem, Yongqin Xian, and Zeynep Akata. Open world compositional zero-shot learning. In *CVPR*, pages 5222–5230, 2021. 7

[45] Arnab Kumar Mondal, Vineet Jain, and Kaleem Siddiqi. Mini-batch graphs for robust image classification. *arXiv preprint arXiv:2105.03237*, 2021. 2

[46] MF Naeem, Y Xian, F Tombari, and Zeynep Akata. Learning graph embeddings for compositional zero-shot learning. In *CVPR*. IEEE, 2021. 1, 2, 3, 6, 7

[47] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. In *NeurIPS*, 2019. 3

[48] Mark M Palatucci, Dean A Pomerleau, Geoffrey E Hinton, and Tom Mitchell. Zero-shot learning with semantic output codes. 2009. 3

[49] Xingchao Peng, Zijun Huang, Ximeng Sun, and Kate Saenko. Domain agnostic learning with disentangled representations. In *ICML*, pages 5102–5112. PMLR, 2019. 2, 3

[50] Vihari Piratla, Praneeth Netrapalli, and Sunita Sarawagi. Efficient domain generalization via common-specific low-rank decomposition. In *ICML*, pages 7728–7738. PMLR, 2020. 3

[51] Senthil Purushwalkam, Maximilian Nickel, Abhinav Gupta, and Marc'Aurelio Ranzato. Task-driven modular networks for zero-shot compositional learning. In *ICCV*, pages 3593–3602, 2019. 6

[52] Jiawei Ren, Cunjun Yu, Shunan Sheng, Xiao Ma, Haiyu Zhao, Shuai Yi, and Hongsheng Li. Balanced meta-softmax for long-tailed visual recognition. In *NeurIPS*, 2020. 5, 6, 8

[53] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, pages 618–626, 2017. 8

[54] Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. Generalizing across domains via cross-gradient training. In *ICLR*, 2018. 1, 2

[55] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *ECCV*, 2016. 7

[56] Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. Equalization loss for long-tailed object recognition. In *CVPR*, pages 11662–11671, 2020. 2

[57] Pavel Tokmakov, Yu-Xiong Wang, and Martial Hebert. Learning compositional representations for few-shot recognition. In *CVPR*, pages 6372–6381, 2019. 2

[58] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & distillation through attention. In *ICML*, pages 10347–10357, 2021. 5

[59] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *CVPR*, pages 8769–8778, 2018. 5

[60] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017. 4

[61] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, pages 5018–5027, 2017. 6

[62] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Wenjun Zeng, and Tao Qin. Generalizing to unseen domains: A survey on domain generalization. *arXiv preprint arXiv:2103.03097*, 2021. 3

[63] Jianfeng Wang, Thomas Lukasiewicz, Xiaolin Hu, Jianfei Cai, and Zhenghua Xu. Rsg: A simple but effective module for learning imbalanced datasets. In *CVPR*, pages 3784–3793, 2021. 1, 2, 5

[64] Wei Wang, Vincent W Zheng, Han Yu, and Chunyan Miao. A survey of zero-shot learning: Settings, methods, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–37, 2019. 3

[65] Xudong Wang, Long Lian, Zhongqi Miao, Ziwei Liu, and Stella X Yu. Long-tailed recognition by routing diverse distribution-aware experts. In *ICLR*, 2021. 1, 2, 5, 6

[66] Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. Learning to model the tail. In *NIPS*, pages 7032–7042, 2017. 1, 2, 3

[67] Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *PAMI*, 41(9):2251–2265, 2018. 2, 3

[68] Liuyu Xiang, Guiguang Ding, and Jungong Han. Learning from multiple experts: Self-paced knowledge distillation for long-tailed classification. In *ECCV*, pages 247–263. Springer, 2020. 5

[69] Aron Yu and Kristen Grauman. Fine-grained visual comparisons with local learning. In *CVPR*, pages 192–199, 2014. 6

[70] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *ICLR*, 2018. 1, 2, 4

[71] Yuyang Zhao, Zhun Zhong, Fengxiang Yang, Zhiming Luo, Yaojin Lin, Shaozi Li, and Nicu Sebe. Learning to generalize unseen domains via memory-based multi-source meta-learning for person re-identification. In *CVPR*, pages 6277–6286, 2021. 3

[72] Zhisheng Zhong, Jiequan Cui, Shu Liu, and Jiaya Jia. Improving calibration for long-tailed recognition. In *CVPR*, pages 16489–16498, 2021. 2

[73] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *CVPR*, pages 9719–9728, 2020. 2, 6

[74] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *PAMI*, 40(6):1452–1464, 2017. 5

[75] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization: A survey. *arXiv preprint arXiv:2103.02503*, 2021. 3, 5

[76] Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Learning to generate novel domains for domain generalization. In *ECCV*, pages 561–578. Springer, 2020. 3, 5

[77] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain generalization with mixstyle. *ICLR*, 2021. 7

[78] Linchao Zhu and Yi Yang. Inflated episodic memory with region self-attention for long-tailed visual recognition. In *CVPR*, pages 4344–4353, 2020. 1, 2, 3

[79] Yizhe Zhu, Mohamed Elhoseiny, Bingchen Liu, Xi Peng, and Ahmed Elgammal. A generative adversarial approach for zero-shot learning from noisy texts. In *CVPR*, pages 1004–1013, 2018. 3, 4, 5