Rethinking the Augmentation Module in Contrastive Learning: Learning Hierarchical Augmentation Invariance with Expanded Views

Junbo Zhang Kaisheng Ma*
Tsinghua University
zhangjb21@mails.tsinghua.edu.cn kaisheng@mail.tsinghua.edu.cn

Abstract

A data augmentation module is utilized in contrastive learning to transform the given data example into two views, which is considered essential and irreplaceable. However, the pre-determined composition of multiple data augmentations brings two drawbacks. First, the artificial choice of augmentation types brings specific representational invariances to the model, which have different degrees of positive and negative effects on different downstream tasks. Treating each type of augmentation equally during training makes the model learn non-optimal representations for various downstream tasks and limits the flexibility to choose augmentation types beforehand. Second, the strong data augmentations used in classic contrastive learning methods may bring too much invariance in some cases, and fine-grained information that is essential to some downstream tasks may be lost. This paper proposes a general method to alleviate these two problems by considering "where" and "what" to contrast in a general contrastive learning framework. We first propose to learn different augmentation invariances at different depths of the model according to the importance of each data augmentation instead of learning representational invariances evenly in the backbone. We then propose to expand the contrast content with augmentation embeddings to reduce the misleading effects of strong data augmentations. Experiments based on several baseline methods demonstrate that we learn better representations for various benchmarks on classification, detection, and segmentation downstream tasks.

1. Introduction

Contrastive learning has been proved to be able to learn meaningful visual representations without human annotations [4, 10]. Original methods regard two views transformed from the same example as a positive pair and other examples in the batch or the memory bank as negative samples [30], then the model is trained with the contrastive loss. Many techniques were previously considered important in this process, such as strong data augmentations [3], the selection of negative samples [14, 34], momentum-update encoder [10, 11], and training details like large batch sizes and long training epochs [3]. However, recent works prove that useful visual representations can be learned without negative pairs, the momentum-update of parameters or large batch size [4, 10, 39]. The most indispensable process in contrastive learning is the data augmentation module. The essential principle of contrastive learning is to learn the representational invariance by making the network learn to be invariant to a set of data augmentations [29].

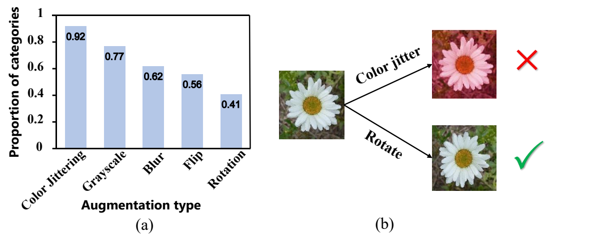

Figure 1. The influence of augmentation types. (a) The proportion of categories in ImageNet where the data augmentation has a positive effect. (b) An example of the flower where color invariance has a negative effect and rotation invariance has a positive effect. The conclusion is reversed for most categories in ImageNet.

Previous works show that the composition of multiple types of data augmentations is crucial for contrastive learning [3]. A specific set of augmentations is determined after extensive experiments to achieve the best results on large-scale datasets (e.g., ImageNet). In most recent works [1, 11, 16, 26, 33], the data augmentation pipeline consists of random cropping and resizing, horizontal flipping, color jittering, converting to grayscale, and Gaussian blurring. However, the pre-determined and artificial choices of augmentation types and augmentation strength bring the corresponding problems as follows.

In terms of the augmentation type, after selecting specific

*Corresponding Author.

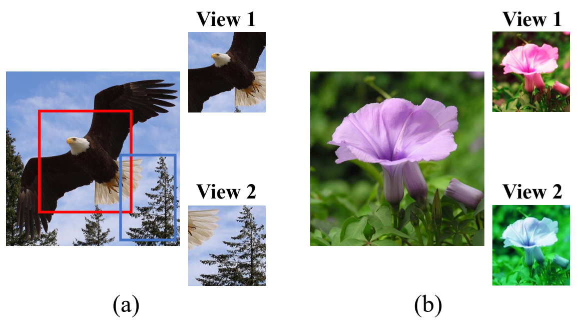

Figure 2. The influence of augmentation strength. (a) Aligning the two cropped views introduces misleading spatial information. (b) Aligning the two color-augmented views loses the fine-grained color information of flowers.

types, current contrastive learning methods treat each augmentation equally, and all the representational invariances are evenly distributed in the backbone. However, according to previous works, some augmentations are fundamental for contrastive learning [3, 10], such as cropping and color jittering, while others are less important or even harmful, such as blurring and rotating. Figure 1 (a) shows the proportion of categories in ImageNet where the data augmentation has a positive effect. Furthermore, each data augmentation may have a different impact on downstream tasks. As shown in Figure 1 (b), using color jittering may help with most categories in ImageNet but harms the representation learning of flowers, while adding rotation brings the opposite effect. Thus, simply getting rid of one augmentation (e.g., rotation) and keeping the other as in the classical augmentation pipeline assumes an implicit knowledge of invariance for no reason. Indiscriminately learning the augmentation invariances in the encoder makes it less flexible in choosing augmentation types beforehand and may lead to non-optimal representations for various downstream tasks.

In terms of the augmentation strength, SimCLR [3] shows that contrastive learning benefits from strong data augmentations. InfoMin principle [29] also argues that weak augmentations bring too much noise into the mutual information between two augmented views, which leads to worse generalization on the downstream task. Nevertheless, the representations learned using strong augmentations may sometimes bring unnecessary invariance into the backbone and lose fine-grained information essential to some downstream tasks. As shown in Figure 2, projecting the two cropped views into the same position in the feature space learns misleading spatial information and brings too much invariance. And projecting the two color-augmented views into the same position loses the fine-grained category information [6] (e.g., the color of morning glory). Although classical methods implicitly alleviate this problem by adding a projection head after the encoder, no explicit method has been proposed in unsupervised contrastive learning.

This paper proposes a generic method to tackle these two augmentation-related problems by considering "where" and "what" to contrast. First, we propose to treat various data augmentations differently and learn the "hierarchical augmentation invariance". By computing multiple contrastive losses at different depths of the encoder, we make the fundamental augmentation invariances more widely distributed and some generally insignificant invariances restricted to the deeper layers. By restricting the impact scope of data augmentation without weakening its strength, we demonstrate that adding a specific type of data augmentation that was not used in the classical augmentation set can simultaneously improve the performance on both large-scale datasets (e.g., ImageNet, COCO) and fine-grained datasets (e.g., VGG Flowers, iNaturelist-2019). Second, we propose to expand the contrast content with augmentation embeddings. By augmenting the original labels with input transformation, the label augmentation [18] method in supervised learning relaxes specific transformation invariant constraints and prevents the loss of transformation-related information. Inspired by this, we regard each view as the "label" of the other view and expand the extracted view features with corresponding augmentation embeddings. The encoded augmentation information helps to reduce the unnecessary invariance and make up for some lost fine-grained information. The small network for embedding augmentation parameters is simultaneously trained with the encoder and is discarded during the inference. Our analysis demonstrates that the augmentation embeddings learn useful information of specific augmentations and benefit the representation learning in various benchmarks.

We apply our method to several baseline contrastive learning architectures and evaluate the representations on various classification, detection, and segmentation benchmarks. Our results reveal that the proposed method consistently improves the performance compared to baselines on various downstream tasks.

2. Related work

2.1. Contrastive learning

Contrastive learning aims to learn generalizable and transferable representations from unlabeled data using contrastive pairs. The classic method constructs a positive pair and a negative pair for each data sample and optimizes the model with infoNCE loss [30]. The choice of negative samples is considered important in this process. MoCo [11] introduces a dynamic memory bank to record the embeddings of negative samples. AdCo [14] learns a set of negative samples by adversarial training. Other works propose to contrast the data with prototypical representations trying to find more suitable negative samples for contrastive learning [2, 19]. Besides negative samples, SimCLR [3] shows the large batch size and long training time are also crucial for contrastive learning. Beyond the aforementioned classic training framework, BYOL [10] introduces a slow-moving average network and shows that contrastive learning can be effective without using any negative samples. By introducing stop-gradient, SimSiam [4] shows that simple Siamese networks can learn meaningful representations even without negative pairs, large batches, or momentum encoders. Barlow Twins [39] further proposes a new contrastive learning objective without using stop-gradient, which also brings comparable results. Recently, some theoretical works have tried to understand how these new methods succeed in avoiding representational collapse [15, 27]. Although contrastive learning framework has been dramatically simplified, producing a contrastive pair with multiple types of data augmentations is still considered vital and indispensable in most works. The essence of contrastive learning is to learn multiple augmentation invariances in the representations so that these representations can be utilized successfully in a variety of downstream tasks.

2.2. Augmentation in contrastive learning

The augmentation module in contrastive learning transforms the given data sample into two correlated views. SimCLR [3] first shows that the composition of multiple data augmentations is crucial to yield effective representations, among which the composition of random cropping and color jittering stands out. The paper also conducts detailed experiments to confirm that unsupervised contrastive learning benefits from strong data augmentation and determines a certain set of augmentations that yield the best results on universal datasets. BYOL [10] proposes a novel architecture that is more robust to the choice of image augmentations. Its performance is much less affected than SimCLR when removing some augmentations. However, simply removing one type of augmentation in BYOL can still result in a 5% $\sim$ 25% decrease in accuracy. As the necessity of data augmentation is fully confirmed, many works explore what kind of data augmentation can lead to better representations in contrastive learning. SwAV [2] proposes a new 'multi-crop' augmentation strategy that mixes the views of different resolutions. CsMl [37] applies CutMix [38] augmentation to generate the views with cross-samples and multi-level representation. Besides, InfoMin [29] argues that the optimal augmentation strategy should reduce the mutual information between views while maintaining the task-relevant information. Since this 'optimal' augmentation strategy is strongly related to downstream tasks, InfoMin proposes to learn this optimal strategy in a semi-supervised way. Although this method does not work well in a complete unsupervised framework, it inspires us to think of whether using a pre-determined composition of data augmentations, as in most previous works, has some drawbacks on certain downstream tasks. A similar idea is shown in MaskCo [42], which argues that the implicit semantic consistency assumption in instance discrimination pre-training task may harm the performance of downstream tasks on unconstrained datasets. In terms of augmentations, the pre-determined data augmentations assume particular representation invariances that may not always be needed in downstream tasks. LooC [36] weakens this assumption implicitly by using a multi-head network. Nevertheless, the backbone of LOOC still has the same representational invariances. Instead, we reduce the drawbacks of both the pre-determined augmentation types and augmentation strength by considering "where" and "what" to contrast in contrastive learning.

3. Method

We first review the three indispensable components in the general contrastive learning framework in Section 3.1: the augmentation module, the siamese structure, and the contrastive loss. Then we illustrate the overall architecture in Section 3.2 and show that the proposed method can be combined with any approach that fits within the general contrastive learning framework. Next, we thoroughly introduce the proposed methods dealing with the problems of augmentation types and augmentation strength in Section 3.3 and 3.4, respectively.

3.1. General contrastive learning framework

We first briefly review the three components in the general framework. Firstly, a data augmentation module transforms the given data example into two views randomly. These two correlated views are then fed to the backbone and are used to compute the contrastive loss. The composition of multiple data augmentations widely used in most previous works contains random cropping and resizing, horizontal flipping, color jittering, converting to grayscale, and Gaussian blurring. Secondly, two neural-network-based encoders form a siamese structure that extracts representation vectors from the two views. Note that the two encoders can share their weights [4, 39] or can be two separate networks with the same structure [11, 22, 31]. The encoders can be optimized by backpropagation or the momentum-based moving average of parameters [10]. Thirdly, a contrastive loss is used to define a contrastive prediction task. Several contrastive losses have been proposed in previous works, such as InfoNCE [30] computed with the negative samples or prototypes [2, 19], negative cosine similarity between two positive view features [4], and the similarity between the cross-correlation matrix of views and the identity matrix [39]. The core of these objective functions is to narrow the distance between two views in the feature space.

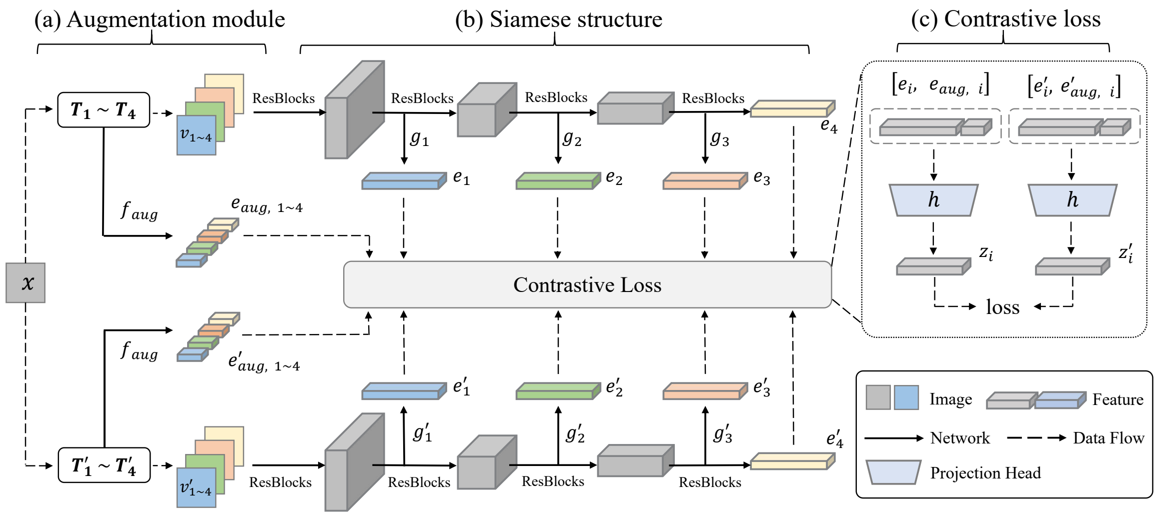

Figure 3. The improvements to the three components in the general contrastive learning framework. (a) Augmentation module: We generate multiple pairs of views with an add-one strategy and utilize a small network to embed the augmentation parameters. (b) Siamese structure: We divide the backbone into several stages according to the depth. Multiple pairs of view features shown in different colors are extracted at different stages. (c) Contrastive loss: We introduce multiple contrastive losses at different stages using multiple pairs of view features. The view features are expanded with corresponding augmentation embeddings before the projection head. The expanded features can then be used to calculate contrastive loss as in many previous works.

3.2. Overall architecture of our method

The overview of our method is shown in Figure 3. We introduce three improvements to the contrastive learning framework. First, instead of transforming the data sample into one pair of views, we produce multiple augmentation modules with add-one strategy and transform the image into multiple pairs of views as in Figure 3 (a). "Add-one" means each augmentation module has one more augmentation type than previous one. A small network is used to embed specific augmentation parameters for subsequent improvements. Second, the siamese structure is divided into several stages according to the depth as in Figure 3 (b). The multiple pairs of view features will subsequently be extracted at different stages and used to compute multiple contrastive losses, which "supervise" different parts of the encoder. With these two improvements, hierarchical augmentation invariance can be learned in the encoder. Third, we expand the view features with augmentation embeddings while keeping the original loss function unchanged as in Figure 3 (c). By doing so, the view information is combined with the augmentation information. Note that these three improvements can be combined with all variants of the current contrastive learning framework. We next introduce the first two improvements in Section 3.3 and the third improvement in Section 3.4.

3.3. Hierarchical augmentation invariance

In order to alleviate the problem of augmentation types as discussed in Section 1, we propose to learn the hierarchical augmentation invariances by applying different invariances to different depths of the encoder. Specifically, we take an add-one strategy based on the importance of each augmentation type to produce multiple augmentation modules. Thus, each augmentation module has one more augmentation type than the previous module, and the last module contains all the types of augmentations as in the classical pipeline. In this paper, we divide the backbone into four stages according to the depth as previous works [40, 41]. We formulate the complete augmentation pipeline in most previous works as $T = \text{Compose}\{t_0, t_1, t_2, t_3, t_4\}$, where t_0 contains the base augmentations and $t_1 \sim t_4$ each represents one type of augmentation. So the four augmentation modules T_1, T_2, T_3, T_4 generated by add-one strategy can be formulated as:

$$\begin{aligned} T_1 &= \text{Compose}\{t_0, t_1\}, \\ T_2 &= \text{Compose}\{t_0, t_1, t_2\}, \\ T_3 &= \text{Compose}\{t_0, t_1, t_2, t_3\}, \\ T_4 &= \text{Compose}\{t_0, t_1, t_2, t_3, t_4\}. \end{aligned} \tag{1}$$

With these modules, we augment the given data sample x into eight views as: $v_i = T_i(x)$, $v_i^{'} = T_i^{'}(x)$, $i = 1, 2, 3, 4$, where T_i and $T_i^{'}$ have the same augmentation types but are two random instances. In this way, we obtain four pairs of views. Aligning each pair of view features brings a specific composition of augmentation invariance.

With these pairs of views, we propose to compute multiple contrastive losses at different depths of the encoder.

Optimizing each loss function brings specific invariance to a subpart of the encoder. We introduce the contrastive losses at the end of each stage for the ResNet backbone. Specifically, we add several convolutional layers to the shallow stages to get features of the same shape as the last stage output. These extra layers also help to solve gradient competition issue [25] and extract more meaningful visual representations. Then the four pairs of view features are extracted by the four subparts of the encoder respectively, which can be formulated as:

$$e_i = g_i(f_i(v_i));\ e_i^{'} = g_i^{'}(f_i(v_i^{'})),\ i = 1, 2, 3, 4, \tag{2}$$

where f_i represents the four stages of the backbone and g_i represents the extra convolutional layers. We do not add any layers to the last stage, so g_4 and $g_4^{'}$ can be instantiated as an identity function. As in most contrastive learning works, we use several MLP based projection heads h_i to map each view features to the space where contrastive loss is applied:

$$z_i = h_i(e_i);\ z_i^{'} = h_i(e_i^{'}). \tag{3}$$

Thus, the overall loss function is fomulated as follows where $L_{contrast}$ could be any of the contrastive losses:

$$L_i = L_{contrast}(z_i, z_i^{'}), \quad L_{overall} = \sum_{i=1}^{4} L_i. \tag{4}$$

In this way, the invariances of augmentation t_0 and t_1 are most widely distributed, while the invariance of t_4 is restricted to the deepest stage of the encoder.

In this paper, we instantiate t_0 as random cropping and resizing since ImageNet images are of different sizes, and cropping is considered the most fundamental augmentation for contrastive learning. $t_1 \sim t_4$ are selected without repetition in the following four augmentations as the classical pipeline: horizontal flipping, color jittering, converting to grayscale, and Gaussian blurring. Consistent with our intuition and motivation, the empirical study finds that instantiating t_i according to the importance of each augmentation brings the best results, as shown in Section 5.1.

3.4. Feature expansion with augmentation embeddings

Some previous works take self-supervised tasks as auxiliary training in order to boost the performance of supervised learning [18]. These works utilize "label augmentation" to remove the unnecessary invariance brought by the self-supervised tasks. They augment the class label y with the augmentation parameters used for the corresponding image. More specifically, the classifier should jointly predict the original label and the self-supervised label defined by the augmentation parameters (e.g., the rotation degree of $0°, 90°, 180°$ and $270°$).

There is no class label in self-supervised learning, and the augmentation parameters are contiguous float numbers in most cases. Thus, we propose to regard each view as the "label" of the other view and embed the augmentation parameters by a small network. In this way, we can expand each view feature with its augmentation embedding to remove the unnecessary invariance properly.

In practice, we utilize a linear layer f_{aug} to embed the parameters of each augmentation to a vector. Taking color jittering as an example, we first get the four factors of brightness, contrast, saturation, and hue as $[b, c, s, h]$. We concatenate these scalars and embed them using a linear layer with the input size of 4: $e_{aug} = f_{aug}([b, c, s, h])$. Then we concatenate the view feature and the augmentation embedding before feeding them to the projection head. So the Equation (3) is changed into:

$$z_i = h_i([e_i, e_{aug,i}]);\ z_i^{'} = h_i([e_i^{'}, e_{aug,i}^{'}]). \tag{5}$$

Previous works conjecture an intuitive understanding of the projection head: induced by the contrastive loss, the projection head removes information that may be useful for the downstream task, such as the color of objects [3]. Thus, more useful information can be maintained in the backbone f. With the proposed feature expansion procedure, the projection head should consider both the feature and color-related information to meet the contrastive learning objective. In other words, we explicitly facilitate the projection head to do the job of removing color-related information. Therefore, useful information is more appropriately stored in the backbone f.

This paper also attempts to embed cropping parameters and combine the embeddings with color augmentation embeddings. For random cropping, we get four parameters representing the cropping box's location, height, and width $[x, y, h, w]$ and encode them in the same way as above. The detailed comparison of different augmentation embeddings is shown in Section 5.3.

4. Experiments

4.1. Experiment settings

Datasets. Most of our study for unsupervised pretraining is done using the ImageNet ILSVRC-2012 dataset [5]. Additional pretraining experiments for ablation are done using a subset of ImageNet, 100-category ImageNet (IN-100). The split of this subset follows CMC [28], which contains about 125K images of 100 classes. The linear evaluation for the classification task is conducted on the ImageNet and some other datasets to measure the transfer capabilities as follows. The Caltech-UCSD Birds 2011 (CUB-200) dataset [32], a fine-grained classification dataset of 200 bird species. VGG Flowers (Flowers-102) [23], a

Table 1. Top-1 and top-5 accuracies (%) under linear evaluation. All models use a ResNet-50 encoder pre-trained on ImageNet.

Model	Method	ImageNet		CUB-200		Flower-102		iNat-2019		Car-196	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
ResNet50	SimSiam [11]	69.9	89.3	38.8	68.2	89.9	97.4	32.1	58.4	50.5	76.7
	SimSiam + Ours	**70.1**	**89.5**	**42.2**	**72.0**	**92.3**	**98.4**	**38.1**	**65.3**	**51.9**	**77.0**
ResNet50	BT [39]	67.0	87.6	33.8	62.9	89.1	97.2	31.9	58.1	37.2	64.3
	BT + Ours	**67.1**	**87.9**	**35.8**	**65.1**	**91.2**	**97.8**	**36.0**	**63.6**	**38.4**	**64.7**

Table 2. Top-1 and top-5 accuracies (%) under linear evaluation. All models use a ResNet-34 encoder pre-trained on ImageNet-100.

Model	Method	ImageNet-100		CUB-200		Flower-102		iNat-2019		Car-196	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
ResNet34	SimSiam [11]	63.9	88.7	29.1	58.5	76.1	91.9	16.9	36.6	19.6	42.0
	SimSiam + Ours	**67.1**	**90.5**	**33.9**	**64.2**	**83.6**	**94.5**	**23.2**	**44.5**	**20.7**	**42.7**
ResNet34	BYOL [10]	66.5	89.1	30.0	59.4	76.6	92.3	17.8	37.6	20.5	42.6
	BYOL + Ours	**69.3**	**91.3**	**33.6**	**64.0**	**83.2**	**94.1**	**24.4**	**47.5**	**21.9**	**43.8**
ResNet34	BT [39]	67.3	89.8	27.5	55.1	76.4	91.7	17.1	37.2	15.5	34.1
	BT + Ours	**69.1**	**91.7**	**29.4**	**58.5**	**82.1**	**94.9**	**23.3**	**45.0**	**17.0**	**36.2**

fine-grained dataset of 102 flower categories. The iNaturelist 2019 dataset (iNat-2019) [13], a large-scale dataset with 268,243 images, which contains 1010 species of natural plants and animals. The Stanford Cars dataset (Car-196) [17], a fine-grained dataset which contains 16,185 images of 196 classes of cars. We measure the transfer capabilities to other tasks on two famous benchmarks, including VOC 2007 [7] for object detection and COCO [20] for both object detection and instance segmentation.

Implementation details. To learn the hierarchical augmentation invariance, we divide the backbone into four stages and extract the feature maps before each downsampling layer. The data augmentation $t_1 \sim t_4$ in the main experiments are color jittering, converting to grayscale, blurring, and horizontal flipping, respectively. The chosen strategy of these augmentation types is discussed in Section 5.1. To produce view features in different stages, we add several extra convolutional layers in shallow stages (the number of conv layers is 3,2,1 for the first three stages, respectively), which are discarded in the inference period.

The MLP-based projection head has the same structure as many previous works, which has three fully-connected layers. Note that we also add three projection heads to project the view features in shallow stages.

To embed the augmentation parameters, we use a linear layer followed by BN and ReLU to project the parameters into a 512-dim vector for ImageNet pre-training experiments. This small network is also discarded during inference. In the main experiments, we expand the view features with the augmentation embedding in all stages that contain the corresponding augmentation invariance.

Unsupervised pre-training. We pre-train the ResNet-50 backbone on the 1000-category ImageNet training set following the classical protocol. We also pre-train the ResNet-34 backbone on 100-category ImageNet for ablation study using the same hyperparameter settings. Specifically, following SimSiam, we pre-train all the models for 200 epochs with the SGD optimizer. We use a learning rate of $lr \times$ BatchSize / 256 (linear scaling [8]), with a base $lr = 0.05$. The learning rate has a cosine decay schedule [3, 21]. The weight decay is 0.0001 and the SGD momentum is 0.9. The batch size is 256. For the sake of fair and direct comparison, we use the same settings for each baseline and our method, and only apply the three improvements to the baselines mentioned above.

Linear evaluation, detection and segmentation. We follow the linear evaluation protocol [9], in which the pre-trained model is fixed and only the additional linear classification layer is fine-tuned. For detection and segmentation tasks, we fine-tune the pre-trained models end-to-end in the target datasets. A Faster R-CNN detector [24] is used for VOC and a Mask R-CNN detector [12] is used for COCO both with the ResNet50-C4 backbone implemented in Detectron2 [35]. More details can be found in the appendix.

4.2. Results

First, we show the results of the linear evaluation in classification tasks where the model is pre-trained on ImageNet. In Table 1, we compare our methods with corresponding baselines using the backbone of ResNet-50 pre-trained on ImageNet-1000. The top-1 and top-5 accuracies on several benchmarks are reported. Our methods significantly improve the performance on various classification downstream tasks, especially on fine-grained datasets, possibly because the representations from fine-grained datasets are more sensitive to data augmentations. For example, color

Table 3. Transfer learning results on detection and segmentation. All models use a ResNet-50 encoder pre-trained on ImageNet.

Method	VOC 07 det			COCO det			COCO instance seg		
	AP_{50}	AP	AP_{75}	AP_{50}	AP	AP_{75}	AP^{mk}_{50}	AP^{mk}	AP^{mk}_{75}
SimSiam [11]	**72.3**	45.8	49.4	55.0	36.0	38.6	51.8	31.9	33.9
SimSiam + Ours	72.0	**46.6**	**50.9**	**56.9**	**37.6**	**40.7**	**53.5**	**33.1**	**35.2**
BT [39]	71.2	45.6	49.4	59.6	40.0	43.0	56.1	35.1	37.2
BT + Ours	**71.7**	**46.0**	**49.7**	**60.0**	**40.2**	**43.5**	**56.7**	**35.1**	**37.6**

information is more crucial to distinguish between flowers in Flower-102 and natural species in iNaturalist-2019. We also compare our methods with more baselines and different model architectures whose results are shown in Table 2. Here we use the backbone of ResNet-34 pre-trained on ImageNet-100. The proposed method still leads to significant accuracy boost compared with the baseline models, reflecting the stable and universal effect of our method.

Next, we evaluate our representations for the localization based tasks of object detection and instance segmentation. For VOC 07 detection, we fine-tune on trainval2007 and report results on test2007 using the standard $\mathrm{AP}_{50}, \mathrm{AP}, \mathrm{AP}_{75}$ metric. For COCO detection and segmentation, we fine-tune on COCO 2017 train and report results on COCO 2017 val. Table 3 shows that our method boosts the accuracies on both tasks, suggesting that our method improves the generalizability of the representations beyond classification tasks.

Above, we report the representation learning results using the exact composition of data augmentations as in classic contrastive learning methods. Next, we show that by restricting the impact scope of data augmentation, adding a specific type of data augmentation that was not used in the classical augmentation pipeline (e.g., rotation) can simultaneously improve the performance on both large-scale datasets and fine-grained datasets. We assume that classic contrastive learning methods introducing rotation data augmentation will lead to severe performance degradation because they treat each data augmentation equally. Instead, we restrict rotation invariances to the deeper layers of the model without weakening its augmentation strength. Following SimCLR [3], the rotation augmentation randomly rotates the image with the probability of 0.5 with one of $\{0°, 90°, 180°, 270°\}$. Table 4 shows the results of our methods compared with SimSiam baseline with and without rotation augmentation. First, by restricting rotation invariance to the deepest stage, our method outperforms the SimSiam [11] baseline with or without rotation augmentation. Second, compared with our method without rotation, adding rotation augmentation improves the performance on fine-grained datasets such as CUB-200 and Flower-102. Besides, adding rotation augmentation in our method brings a significantly smaller negative impact on the universal dataset (e.g., ImageNet) than the baseline. We also compare the results of

Table 4. The impact on accuracy (%) of adding random rotation to the augmentation pipeline. Rotation-i $(\mathrm{i} = 1, 2, 3, 4)$ means the rotation invariance is added to the model from the i^{th} stage.

Method	ImageNet-100	CUB-200	Flower-102
SimSiam [11]	63.9	29.1	76.1
SimSiam + Ours	**67.1**	33.9	83.6
SimSiam [11] + rotation	61.3	28.4	76.4
Ours + rotation-4	66.2	**34.1**	**84.7**
Ours + rotation-1	60.9	31.8	84.2
Ours + rotation-2	62.0	32.0	84.7
Ours + rotation-3	64.3	32.9	84.0

Table 5. Linear evaluation accuracy (%) under different arrangement modes of augmentation types. (C: color jittering; G: converting to grayscale; B: blurring; F: flipping)

Arrangement	ImageNet-100		CUB-200	
	Top-1	Top-5	Top-1	Top-5
[G, B, F, C]	64.5	88.0	32.9	63.6
[B, F, C, G]	65.4	89.3	33.5	64.0
[F, C, G, B]	66.3	90.0	33.2	63.8
[C, G, B, F]	**67.1**	**90.5**	**33.9**	**64.2**

adding rotation invariance starting from different stages of the model, which are shown in the last three lines in Table 4. For our method, rotation-i means that the rotation invariance is added to the model from the i^{th} stage. These results indicate that enlarging the distribution of rotation invariance in the model is harmful to representation learning, which is in line with our intuition and motivation.

5. Discussion

In this section, we analyze our methods in detail and explain why each of our improvement is effective.

5.1. Why hierarchical augmentation works?

We assume that data augmentations have different degrees of both positive and negative effects on downstream tasks, which should be treated differently during training. So we propose to make the fundamental augmentation invariances more widely distributed in the encoder, and restrict some insignificant invariances to the deeper layers, namely learning hierarchical augmentation invariances. To verify this assumption, we change the chosen of $t_1 \sim t_4$ in Equation 1 and compare the results of linear evaluation on ImageNet and CUB-200. Table 5 shows the results of four arrangement modes of augmentations, where C/G/B/F represent color jittering, converting to grayscale, blurring, and flipping, respectively. The results reveal that instantiating $t_1 \sim t_4$ according to the importance of each augmentation brings the best results, which confirms our assumption.

Besides, we compare our method with another two base-

Table 6. Comparison of accuracy (%) with two designed baselines.

Method	ImageNet-100		CUB-200	
	Top-1	Top-5	Top-1	Top-5
Uniform	64.0	88.8	29.3	58.9
Hierarchical strength	63.6	88.2	27.1	56.4
Hierarchical type (ours)	**67.1**	**90.5**	**33.9**	**64.2**

Table 7. Accuracy (%) on the pretext task of predicting the transformation applied during the training using different representations. The accuracy of random guess is 10%.

Task	Representation	Aug embedding	Accuracy (%)
	e	×	25.0
Color	$h(e)$	✓	12.1
	e	✓	**65.3**

lines, namely "Uniform" and "Hierarchical strength" in Table 6. For "Uniform", we also apply multiple contrastive losses at different depths of the model. However, the augmentation invariances are uniform in different stages,which means that $T_1 \sim T_4$ have the same composition of $t_1 \sim t_4$ augmentations in Equation 1. For "Hierarchical strength", we make all types of invariances distributed in all model stages, but the augmentation strength is stronger in deeper stages than the shallower stages. The training details are contained in the appendix. The results show that our method outperforms the two baselines. It confirms that only applying multiple objectives is not enough, and the add-one strategy is effective. Moreover, simply weakening the augmentation strength at shallow stages does not work since contrastive learning benefits from strong data augmentations.

5.2. Why feature extension with augmentation embeddings works?

Previous work [3] conjectures that the projection head h can implicitly remove information that may be useful for the downstream tasks. So the feature e before the projection head h in Equation 3 maintains more useful information and is a better representation for downstream tasks. We expand each view feature e with its augmentation embeddings. This modification explicitly promotes the projection head h to remove useful augmentation-related information induced by the contrastive loss. So more information can be formed and maintained in e. To verify this hypothesis, following SimCLR [3], we conduct experiments that use either $h(e)$ or e trained with or without feature expansion to learn to predict the transformation applied during training. Specifically, we evenly divide the color jittering augmentation into ten categories according to its strength. Then we take the view features and predict the relative distance of augmentation strength applied to them, which is a more difficult task than the original task in SimCLR [3]. Table 7 shows that e trained with feature expansion contains much more information about the transformation applied, while the other representations lose the information.

Table 8. Ablation study on embedding types (acc. /%).

Embedding		Dataset	
Color	Crop	IN-100	CUB
×	×	66.2	31.9
✓	×	**67.1**	33.9
×	✓	66.9	32.9
✓	✓	66.7	**34.0**

Table 9. Ablation study on the two modules (acc. /%).

Hier. aug.	Aug. emb.	Acc.
×	×	63.9
×	✓	66.4
✓	×	66.2
✓	✓	**67.1**

5.3. Ablation study

In this section, we conduct an ablation study on classification tasks with the model pre-trained on ImageNet-100. As random cropping and color jittering are fundamental augmentations for contrastive learning, we also attempt to embed cropping parameters and combine them with color augmentation embeddings as introduced in Section 3.4. Table 8 shows that embedding cropping parameters also leads to an accuracy boost compared with baseline but is less effective than embedding color parameters. Combining cropping embeddings with color embeddings has an unstable effect on different benchmarks. This is probably because we simply concatenate the two types of embeddings channel-wise during training, which could be improved in future work. The ablation study results shown in Table 9 further demonstrate the effectiveness of the two proposed modules in our method, where "Hier. aug." represents hierarchical augmentation invariance introduced in Section 3.3 and "Aug. emb." represents the feature expansion with augmentation embeddings introduced in Section 3.4.

6. Conclusion

In this paper, we consider the drawbacks of the augmentation module in terms of type and strength. We first propose treating each augmentation differently and learning the hierarchical invariance in the backbone, making it more flexible in choosing augmentation types beforehand. We then propose expanding the view features with corresponding augmentation embeddings, which contribute to maintaining useful fine-grained information in view features. The proposed method can be combined with any contrastive learning method following the general framework. Experiments on several classification, detection and segmentation downstream tasks demonstrate that the proposed method leads to a consistent and significant accuracy boost. We also conduct analytical and ablation experiments to explore the effectiveness of each component in our method. Some limitations of our method are included in the appendix.

References

[1] Guy Bukchin, Eli Schwartz, Kate Saenko, Ori Shahar, Rogério Schmidt Feris, Raja Giryes, and Leonid Karlinsky. Fine-grained angular contrastive learning with coarse labels. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8726–8736, 2021. 1

[2] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. 2020. 3

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *ArXiv*, abs/2002.05709, 2020. 1, 2, 3, 5, 6, 7, 8

[4] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, 2021. 1, 3

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, K. Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 5

[6] Linus Ericsson, Henry Gouk, and Timothy M. Hospedales. How well do self-supervised models transfer? *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5410–5419, 2021. 2

[7] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88:303–338, 2009. 6

[8] Priya Goyal, Piotr Dollár, Ross B. Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *ArXiv*, abs/1706.02677, 2017. 6

[9] Priya Goyal, Dhruv Kumar Mahajan, Abhinav Gupta, and Ishan Misra. Scaling and benchmarking self-supervised visual representation learning. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6390–6399, 2019. 6

[10] Jean-Bastien Grill, Florian Strub, Florent Altch'e, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. *NeurIPS*, abs/2006.07733, 2020. 1, 2, 3, 6

[11] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9726–9735, 2020. 1, 2, 3, 6, 7

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask r-cnn. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42:386–397, 2020. 6

[13] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alexander Shepard, Hartwig Adam, Pietro Perona, and Serge J. Belongie. The inaturalist species classification and detection dataset. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8769–8778, 2018. 6

[14] Qianjiang Hu, Xiao Wang, Wei Hu, and Guo-Jun Qi. Adco: Adversarial contrast for efficient learning of unsupervised representations from self-trained negative adversaries. 2021. 1, 2

[15] Tianyu Hua, Wenxiao Wang, Zihui Xue, Yue Wang, Sucheng Ren, and Han Zhao. On feature decorrelation in self-supervised learning. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 3

[16] Ziyu Jiang, Tianlong Chen, Bobak J. Mortazavi, and Zhangyang Wang. Self-damaging contrastive learning. 2021. 1

[17] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. *2013 IEEE International Conference on Computer Vision Workshops*, pages 554–561, 2013. 6

[18] Hankook Lee, Sung Ju Hwang, and Jinwoo Shin. Self-supervised label augmentation via input transformations. In *ICML*, 2020. 2, 5

[19] Junnan Li, Pan Zhou, Caiming Xiong, Richard Socher, and Steven C. H. Hoi. Prototypical contrastive learning of unsupervised representations. 2021. 3

[20] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 6

[21] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv: Learning*, 2017. 6

[22] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6706–6716, 2020. 3

[23] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729, 2008. 5

[24] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:1137–1149, 2015. 6

[25] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, and Y. Bengio. Fitnets: Hints for thin deep nets. In *ICLR*, 2015. 5

[26] Alessandro Sordoni, Nouha Dziri, Hannes Schulz, Geoffrey J. Gordon, Philip Bachman, and Rémi Tachet des Combes. Decomposed mutual information estimation for contrastive representation learning. 2021. 1

[27] Yuandong Tian, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning dynamics without contrastive pairs. 2021. 3

[28] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *ECCV*, 2020. 5

[29] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning. 2020. 1, 2, 3

[30] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748, 2018. 1, 2, 3

[31] Vikas Verma, Minh-Thang Luong, Kenji Kawaguchi, Hieu Pham, and Quoc V. Le. Towards domain-agnostic contrastive learning. In *ICML*, 2021. 3

[32] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge J. Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 5

[33] Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense contrastive learning for self-supervised visual pre-training. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3023–3032, 2021. 1

[34] Mike Wu, Chengxu Zhuang, Milan Mosse, Daniel L. K. Yamins, and Noah D. Goodman. On mutual information in contrastive learning for visual representations. *ArXiv*, abs/2005.13149, 2020. 1

[35] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. 2019. 6

[36] Tete Xiao, Xiaolong Wang, Alexei A. Efros, and Trevor Darrell. What should not be contrastive in contrastive learning. 2021. 3

[37] Haohang Xu, Xiaopeng Zhang, Hao Li, Lingxi Xie, Hongkai Xiong, and Qi Tian. Seed the views: Hierarchical semantic alignment for contrastive representation learning. *ArXiv*, 2021. 3

[38] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Young Joon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6022–6031, 2019. 3

[39] Jure Zbontar, Li Jing, Ishan Misra, Yann André LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *ICML*, 2021. 1, 3, 6, 7

[40] Linfeng Zhang, Yukang Shi, Zuoqiang Shi, Kaisheng Ma, and Chenglong Bao. Task-oriented feature distillation. In *NeurIPS*, 2020. 4

[41] Linfeng Zhang, Jiebo Song, Anni Gao, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3712–3721, 2019. 4

[42] Yucheng Zhao, Guangting Wang, Chong Luo, Wenjun Zeng, and Zhengjun Zha. Self-supervised visual representations learning by contrastive mask prediction. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 3