

ABSTRACT

Title of dissertation: SELF SUPERVISED LEARNING
ON LARGE SCALE DATASETS

Shlok Mishra
Doctor of Philosophy, 2023

Dissertation directed by: Professor David Jacobs
Department of Computer Science

Humans and animals possess the remarkable ability to comprehend and perceive the world around them with minimal, if any, reliance on explicit labels. Much of the knowledge acquired by humans is obtained without the need for direct supervision, simply by processing extensive amounts of unlabeled data. This observation strongly suggests that enabling machines to grasp the world without the use of labels could represent a fundamental approach to artificial intelligence. However, the vast majority of advancements achieved by state-of-the-art deep neural networks have been fueled by their dependence on annotated datasets. The process of annotating datasets is both costly and impractical for numerous domains. This manuscript discusses various ways machines can be taught without using any labels using Self-Supervised Learning (SSL). We show that generally training machines without using any labels can result in less biased and more robust representations.

This manuscript deals with mainly four types of issues in SSL. The first problem we tackle is the over-emphasis of neural networks on low level shortcuts such as

texture. Consider the example of a sofa with texture of a leopard. State-of-the-art neural networks will often predict this sofa to be a leopard, instead of a sofa. Unlike humans, neural networks don't understand the shape of objects and often rely on low level cues. To solve this we propose two different methods. To reduce reliance on texture cues we firstly propose to suppress texture in images, which helps the neural networks to focus less on texture and more on higher level information such as shape. Secondly we augment the SSL learning methods with negative samples which contain only texture from the images. By augmenting with texture based images our method achieves better generalization, especially in out-of-domain settings.

The second problem we deal with is the poor performance of SSL methods on multi-object datasets like OpenImages. One of the fundamental reasons behind this is the cropping data augmentations that select sub-regions of an image to be used as positive samples. These positive samples are generally very meaningful since in object centric datasets they often contain semantic overlap between the views. However this doesn't hold for multi-object datasets since there could be multiple objects and the two views might not have semantic overlap. To remedy this we propose replacing one or both of the random crops with crops obtained from an object proposal algorithm. This encourages the network to learn more object aware representations that result in significant improvement over the random crop baselines.

Thirdly, current SSL networks generally treat objects and scenes using the same framework. However visually similar objects are close in the representation space, hence we argue that scenes and objects should follow a hierarchical structure

based on their compositionality. To solve this, we propose a contrastive learning framework where a Euclidean loss is used to learn object representations and a hyperbolic loss is used to encourage representations of scenes to lie close to representations of their constituent objects in a hyperbolic space. Our hyperbolic loss encourages the network to have a scene-object hypernymy by optimizing the magnitude of their norms.

Lastly, we address the challenge of training self-supervised learning (SSL) methods on vast real-world datasets like JFT. Currently, state-of-the-art SSL methods struggle to perform effectively on JFT due to its skewed data distribution. To address this issue, we present a novel approach that combines Masked Autoencoders and contrastive learning. We introduce CAN, which is a concise and conceptually clear fusion of three components: (C) contrastive learning, (A) masked autoencoders, and (N) the noise prediction approach commonly used in diffusion models. These learning mechanisms complement each other in the following ways: contrastive learning shapes the embedding space when processing a batch of images; masked autoencoders focus on reconstructing low-frequency spatial correlations within a single image; and noise prediction is employed to reconstruct high-frequency image components. When combined, our approach surpasses the performance of its individual constituents, MAE and SimCLR, across a wide range of downstream transfer learning and robustness tasks.

SELF SUPERVISED LEARNING ON LARGE SCALE DATASETS

by

Shlok Mishra

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2023

Advisory Committee:

Professor David Jacobs, Chair/Advisor

Dr Dilip Krishnan

Professor Furong Huang

Professor Abhinav Shrivastava

Professor Rama Chellappa, Dean's Representative

For my wife Akanksha

Acknowledgments

I would like to express my profound gratitude to my advisor, Prof. David Jacobs, for his infinite patience, constant motivation, guidance, and support provided throughout the last five years. He was always there to listen and discuss any ideas. His insights were helpful and helped me grow and understand the process of science. I am also thankful to Dr. Dilip Krishnan for discussing and teaching me how to think big in science and for always being there for me. I am also grateful to my other committee members, Professors Abhinav Shrivastava, Rama Chellapa, and Furong Huang for their time and feedback.

I would like to thank Dr. Abhishek Sharma for his constant support throughout my PhD. He is an amazing guide and taught me the ropes during the initial years of my PhD. I wish to thank Dr. Ankan Bansal, who was always ready to talk about my work provided key insights and helped me in the initial phase of my PhD and also how to navigate life. I was very lucky to have him as my first mentor.

I was fortunate to work with great mentors during my internships. These internships helped me understand the nitty-gritties of working on real-world problems in a fast-paced setting. I am deeply grateful for the mentorship provided by Dr Kuntal Sengupta, Dr. Vincent Chu, Dr. Sofien Bouaziz In my first internship at Google. Dr. Christian Haene, Dr. Hossam Isack in my second internship. Dr. Aaron Maschinot in my third internship.

Immeasurable thanks are due to Ms. Janice Perrone for managing all of the administrative requirements. She makes all of our lives much easier by constantly

working hard. I would also like to take the opportunity to thank the administrative staff of CS and UMIACS for their help throughout my graduate studies. In particular, I would like to thank Vivian Liu, Jodie Gray, and Petra Zapf. I am also grateful to the nice people in the CS Business Office and the staff in AVW and the Iribe Center for taking care of the big things so that I can focus on the tiny ones.

I would like to say thanks to Anshul for working really hard with me on different projects and teaching me all the basics. The path to this dissertation was a tough one. My friends were always there to take me through all the highs and lows. I could not have done this without their support. I want to take this opportunity to thank my flatmates Aman, Ameya, Samyadeep, Manas, Sneha, Naman, and Ryan. They were always available for me and helped me throughout my PhD by always being available. I would also like to say thanks to my other friends Janit, Sweta, Pranav, Susmija, Songwei, Sai, Priyom, Nitish, Aadesh, Trisha, Kamal, Rudradeep, and Shishira for the constant support and great discussions on life, the universe, and everything. I am also grateful for the help and support provided by my lab-mates Koutilya, Vasu, Daniel, Ryan, and Jiaye.

My gratitude for my wife and family is ineffable. But for their support and sacrifices, I wouldn't be writing this.

Table of Contents

List of Tables	ix
List of Figures	xiv
1 Introduction	1
1.1 Objective	1
1.2 A Solution for Texture	2
1.3 A Solution for Multi-Object Scene images	4
1.4 Scaling Self-Supervised learning to large scale datasets:	6
2 Learning Visual Representations For Transfer Learning By Suppressing Texture	8
2.1 Introduction	8
2.2 Related Work	11
2.3 Methods	14
2.3.1 Anisotropic diffusion	15
2.3.2 Pix2Pix for suppressing texture	16
2.3.3 Other texture suppressing methods	17
2.4 Experiments	18
2.4.1 Self-Supervision for Transfer Learning	19
2.4.2 Transfer Learning for Supervised Learning	22
2.4.2.1 Across Domains	22
2.4.2.2 Same Domain	24
2.4.3 Analysis	24
2.4.3.1 Synthetic-DTD Dataset:	25
2.4.3.2 Visual Analysis:	25
2.4.3.3 Experiments on testing shape bias:	26
2.4.3.4 Results after removing high-frequency components:	27
2.4.3.5 Confidence of Models:	29
2.4.3.6 Training with Stylized ImageNet	29
2.4.3.7 Results on model robustness	30
2.5 Jigsaw Task	32
2.5.1 Method	32
2.5.2 Results using Jigsaw Pretraining	33
2.5.3 Jigsaw using Alexnet as backbone	34
2.5.4 Patch-wise Anisotropic Diffusion	34

2.6	Anisotropic Images	35
2.7	Saliency Maps	36
2.7.1	Sketch-ImageNet Saliency Maps	36
2.7.2	Saliency Maps for Anisotropic ImageNet and Standard ImageNet models	36
2.8	Implementation details	36
2.9	Label Corruption Task	39
2.10	Saliency Maps	39
2.11	Conclusion	40
3	Robust Contrastive Learning Using Negative Samples with Diminished Semantics	48
3.1	Introduction	48
3.2	Negative Samples with Diminished Semantics	52
3.2.1	Contrastive learning with non-semantic negatives	52
3.2.2	Texture-based negative sample generation	54
3.2.3	Patch-based negative sample generation	55
3.2.4	How hard are the texture-based and patch-based negative samples?	56
3.3	Experiments	57
3.3.1	ImageNet-100	58
3.3.2	ImageNet-1K	61
3.3.3	Memory bank size	62
3.4	Discussion	63
3.4.1	Controlling the shape-texture trade-off with α	63
3.4.2	Rethinking the shape-texture trade-off through class-based analysis	65
3.5	Related Work	67
3.6	ImageNet-Texture	69
3.6.1	Examples	70
3.6.2	Analysis	70
3.7	Discussion on the contrastive learning with non-semantic negatives	71
3.7.1	Comparison of two ways to apply α in NCE loss	71
3.7.2	Comparison of patch-based negatives and jigsaw-based pretext tasks	72
3.8	Experimental details and additional results	73
3.8.1	Implementation details	73
3.8.2	Results on ImageNet-C with different levels of severity	73
3.8.3	Additional ablations on the patch-based augmentations	74
3.8.3.1	Patch sizes	74
3.8.3.2	Data augmentations on individual patches	75
3.8.4	Hardness of texture-based negative samples	75
3.8.5	Hardness of patch-based negative samples using models trained with different α	76
3.8.6	More results on ImageNet-1K	83

3.9	Closing Remarks	83
4	Object-aware Cropping for Self-Supervised Learning	85
4.1	Introduction	85
4.2	Analysis of Self-Supervised Learning methods on the OpenImages	
	Dataset	88
4.2.1	Performance of SSL methods	89
4.2.2	Analysis and Motivation	90
4.3	Proposed Approach	93
4.3.1	Object Proposals	93
4.3.2	Loss Objectives	95
	4.3.2.1 Rotation task $\mathcal{L}_{rot}$	95
	4.3.2.2 Object localization task $\mathcal{L}_{loc}$	96
4.4	Results	97
4.5	Related Work	102
4.6	Details of OpenImages-Subset	104
4.7	Implementation details for Object localization task	105
4.8	Other object proposal methods	105
4.9	More Analysis on OpenImages	107
4.10	Analysis on Object-Scene crops:	109
4.11	Additional Results	111
4.12	Visualizations	114
4.13	Implementation Details	115
	4.13.1 Hyperparameters for different methods	118
4.14	Theoretical motivation	119
4.15	Conclusion	121
5	Hyperbolic Contrastive Learning for Visual Representations beyond Objects	122
5.1	Introduction	122
5.2	Method	126
5.2.1	Object-Centric Scene Hierarchy	126
5.2.2	Representation Learning beyond Objects	128
	5.2.2.1 Hyperbolic Space	128
	5.2.2.2 Hyperbolic Contrastive Learning	130
5.3	Experiments	133
5.3.1	Implementation Details	133
5.3.2	Main Results	135
5.3.3	Properties of Models Trained with HCL	136
	5.3.3.1 Label Uncertainty Quantification	137
	5.3.3.2 Out-of-Context Detection	139
5.4	Main Ablation Studies	140
5.5	Related Work	142
5.6	Experiment setups	144
5.7	Additional experimental results	146
	5.7.1 Robustness under Corruption.	146

5.7.2	Fine-grained class classification	146
5.7.3	More ImageNet Examples	147
5.8	Additional ablation studies	150
5.9	Closing Remarks	151
6	A simple, efficient and scalable contrastive masked autoencoder for learning visual representations	153
6.1	Introduction	153
6.2	Related Work	156
6.3	A simple contrastive masked autoencoder framework	158
6.3.1	Overview of method	159
6.3.2	Contrastive learning objective	160
6.3.3	Patch reconstruction objective	160
6.3.4	Denoising objective	161
6.3.5	The combined objective function	163
6.3.6	Discussion on Efficiency	163
6.4	Results	164
6.4.1	Pre-training on uncured data: JFT-300M	164
6.4.2	Pre-training on ImageNet-21K	166
6.4.3	Pre-training on ImageNet-1K	167
6.4.4	Few-shot learning	168
6.4.5	Robustness to distribution shift	169
6.5	Hyperparameter analysis	169
6.6	Discussion	173
6.7	Additional Transfer Learning Results	173
6.7.1	ImageNet-21K pre-training.	178
6.8	Runtime of CAN compared to DnC	180
6.9	Hyperparameter settings	180
6.9.1	Pre-training hyperparameters	182
6.9.2	Finetuning and linear probe hyperparameters	184
7	Conclusion and Future work	186
7.1	Conclusion:	186
7.2	Future work:	188
7.2.1	Self-Supervised Learning and Generative Learning on multi-object scene datasets:	188
7.2.2	Self-Supervised Learning on Segment Anything:	189
7.2.3	Self-Supervised Learning in Robotics:	189
7.2.4	Image Text models and SSL:	190
	Bibliography	191

List of Tables

2.1	Comparison with MoCoV2 and Dense-CL in SSL. We note that using Anisotropic diffusion with improves performance on VOC detection and Semantic Segmentation (SS). We test on COCO-based metrics as used in [39]. We also improve performance over the baseline on the semantic segmentation (SS) task [155]. Although we have only focussed on MoCoV2 Dense-CL, our technique can potentially be extended to other state-of-art methods.	21
2.2	Transfer learning across different datasets. Note that our approach leads to improvements in both supervised and SSL set-up.	21
2.3	Experiments with Sketch-ImageNet. Use of Anisotropic ImageNet shows that our method is better at capturing representation that are less dependent on texture.	22
2.4	Comparison using different texture removing methods, with different hyper-parameters for Anisotropic diffusion methods. We observe that the most simple [185] performs the best, and removing more texture from images does not improve performance.	23
2.6	We show additional experiments on dataset ImageNet-C [100], which evaluates model robustness to common corruptions. We can see that by focussing less on texture our model is consistently more robust than the baseline Resnet model pre-trained on ImageNet data.	29
2.7	Comparison of our approach with Jigsaw baseline methods. Using our best model, we improve 2.52 mAP in VOC classification , 0.78 mAP on VOC detection and 8.1 mAP on VOC semantic segmentation(SS) over the baseline models. Note that Stylized ImageNet performs poorly on VOC classification due to the visual shortcuts.	35
2.8	ImageNet classification by finetuning the last FC layer. Features from the conv layers are kept unchanged. This experiment helps evaluate the quality of features learnt by the convolutional layers.	35
2.9	Experiments with Alexnet as the backbone. Ideas of anisotropic diffusion filter can extend to other architectures like Alexnet. The Anisotropic ImageNet model improves over the baseline by 0.67 mAP	35
2.10	Results on Label corruption task. We can see that our model consistently outperforms the baseline with larger improvement upon increasing the corruption probability.	37

2.11	Comparison between Stylized ImageNet and our Anisotropic ImageNet. Following [76], we use ResNet50 as our backbone. We fine-tune our models on only the ImageNet dataset. We can see that on ImageNet classification and object detection, Anisotropic ImageNet and Stylized ImageNet have very similar performance.	39
2.12	Experiments discussing the confidence and entropy of Anisotropic ImageNet and Standard ImageNet	39
3.1	Top-1 accuracy on the ImageNet-100 dataset and its OOD variants. For our main comparison using MoCo-v2 and BYOL, we also report the standard deviation of 3 runs. For MoCo models, k represents the size of memory bank.	59
3.2	Top-1 accuracy on the ImageNet-1K dataset and its sketch, stylized, rendition variants, and mCE on the ImageNet-C dataset.	62
3.3	Top-1 accuracy on the STL-10 dataset. [§] denotes results visually extracted from Figure 2 in [193].	62
3.4	Top-1 accuracy on ImageNet-C-100 dataset with different levels of severity.	74
3.5	Accuracy on ImageNet-100 validation set with different configurations of augmentations on the individual patches. Augmentations are cumulative across the columns (e.g. the “+ Vertical flip” model used horizontal and vertical flip).	75
3.6	Top-1 accuracy on the ImageNet-1K dataset and its sketch, stylized, rendition variants, and mCE on the ImageNet-C dataset.	83
4.1	We achieve superior performance on VOC detection when pre-training on our proposed OpenImages Hard Multi-object Subset OHMS dataset as compared to ImageNet trained models by 6.52mAP. Our proposed OAC also helps ImageNet (2nd last row), but it helps much more on the proposed OHMS multi-object dataset. ImageNet baselines has been trained for 100 epochs [202] and OpenImages model have been trained for same 100 ImageNet equivalent epochs.	88
4.2	Classification results on OHMS and Imagenet. For each SSL method, we show in parentheses the gap to fully supervised training (same number of epochs). The last row shows that our proposed approach using <i>obj-obj+dilate</i> cropping reduces the gap on OHMS by nearly half compared to the baselines, improving over the <i>scene-scene</i> cropping based SSL methods by between 6.8 to 9.4 mAP points. We also observe improvements on ImageNet as well.	90
4.3	Linear evaluation on our OHMS dataset with different pre-training strategies with MoCo-v2 (see Section 4.2 for details). Column 1 uses fully supervised learning on OHMS. We see that for self-supervised pre-training, no specific range of scene crops helps to close the large gap between SSL and supervised training. However, there is a sweet spot of scene crop range where MoCo-v2 performance is highest. . . .	93

4.4	Crop approaches on OHMS: using LOD crops to generate one view, and a dilated crop for the other positive, we are able to reduce the difference between SSL and Supervised Learning by close to 50% (compare the last two rows to the first row). Using ground-truth boxes to generate crops from OHMS improves the pre-training performance marginally compared to LOD crops. Our Obj-Obj+Dilate outperforms the Scene-Scene baseline by significant margin on the OHMS dataset.	98
4.5	Object detection (first 3 columns) and Semantic Segmentation (last 3 columns) results on COCO dataset. All SSL models have been pre-trained on COCO dataset and then finetuned on COCO. All other methods are run for 90K, finetuning iterations. For any SSL method, we compare (BYOL, Moco-v2, Dense-CL) adding Object-Aware-Cropping (OAC) crop improves performance.	99
4.6	Object detection results on VOC dataset (OHMS pre-training, using LOD proposals). All models have been pre-trained on OHMS and then fine-tuned on VOC. We can see that all of our proposed components improve upon the baseline scene scene crop and combining all of them improves upon the baseline by 3.1 mAP.	100
4.7	Object detection (first 3 columns) and semantic segmentation (last 3 columns) results on COCO (first 2 rows) and VOC (last 2 rows). All SSL models have been pre-trained on complete OpenImages dataset(1.9 million images) for 75 epochs and then finetuned on COCO and VOC dataset.	100
4.8	Object detection results on VOC dataset (COCO pre-training). All models have been pre-trained on COCO and then fine-tuned on VOC. For both MoCo-v2 and BYOL, replacing the default scene crops with object-scene crops results in a consistent improvement.	112
4.9	Object detection results on VOC dataset (OpenImages pre-training). All models have been pre-trained on OpenImages and then fine-tuned on VOC. Replacing the default scene crops with Obj-Scene crops results in a consistent improvement.	112
4.10	Varying the number of proposals generated by BING. All models are pre-trained on COCO and then fine-tuned on VOC Object Detection. Increasing the number of proposals provides a consistent boost. We used 10 proposals as this was close to the average number of objects per image in OpenImages.	113
4.11	Varying dilation i.e δ on the COCO dataset for Obj-Obj+Dilate crop strategy. All models have been pre-trained on COCO and then fine-tuned on VOC Object Detection. $\delta = 0$ corresponds to Obj-Obj cropping and larger δ is very similar to Obj-Scene. A sweet spot exists between the extremes.	113
4.12	Obj-Obj+Dilate crop pre-training consistently outperforms Obj-Scene for transfer to various downstream datasets for classification (COCO pre-training).	113

4.13	Impact of varying scene crop range in the object-scene crop.	113
4.14	Results after pre-training for 200 epochs. These results are highly indicative of results we get after longer training and can be useful for comparison in less compute settings.	114
4.15	Object detection results on COCO (top 2 rows) and VOC (bottom 2 rows). All SSL models have been pre-trained on ImageNet for 200 epochs and then fine-tuned on COCO and VOC.	114
5.1	Classification results with linear evaluation. Our model improves scene-level classification on the VOC [68] and INPMix [269] datasets, and object-level classification on ImageNet-100 [219] and ImageNet-1k [56] datasets.	134
5.2	Object detection and Semantic Segmentation results. Our model improves on both tasks on COCO [148] and VOC [68] datasets.	135
5.3	NDCG scores of the image rankings based on the different indicators and models, and evaluated by the the number of labels per image.	137
5.4	Ablation on the similarity measure and hierarchy center.	140
5.5	Ablation on the losses trade-off.	141
5.6	Ablation on the RSGD versus SGD optimizers.	141
5.7	Fine grained classification results.	146
5.8	Results by varying the radius r of Poincaré ball. $c = \frac{1}{r^2}$	151
5.9	Different configurations of the head in the Euclidean and hyperbolic branches.	151
5.10	Results of hyperbolic linear evaluation with different optimizers and learning rates.	151
6.1	JFT-300M pre-training: Comparison to the state of the art on ImageNet linear probe. CAN outperforms all methods except DnC, which uses a complicated multi-stage training process. Computation is measured as ImageNet-equivalent epochs. [†] Our implementation of [34] and [90].	164
6.2	Finetune and linear probe results with pre-training on ImageNet-1K. Note that CAN does not use multi-crop augmentation or momentum encoder. [†] Our implementation of [34] and [90]. [*] Quoted from [38].	167
6.3	Loss complementarity. CAN training achieves <i>lower</i> training loss for both contrastive and reconstruction than individual training. All methods use 50% masking for fair comparison.	171
6.4	Denoising objective. “Full” denotes the entire method as described in Section 6.3.4	172
6.5	CAN loss terms. We remove each of the three loss terms in CAN one by one.	172
6.6	IN-21K pre-training. Linear probe performance on IN-1K. ViT-B models pre-trained for 800 IN-1K equivalent epochs.	179

6.7	Hyperparameters for CAN pre-training on ImageNet. Note that we use lower learning rate for ViT-L as compared to ViT-B, following [212]. We use the same hyper-parameters for SimCLR pre-training, except for masking rate that is set to 0.	183
6.8	Hyperparameters for CAN pre-training on JFT-300M and IN-21K. Note that we use lower learning rate for ViT-L as compared to ViT-B, following [212]. We use the same hyper-parameters for SimCLR pre-training, except for masking rate that is set to 0.	183
6.9	Hyperparameters for MAE pre-training on IN-1K. These parameters are exactly the same as in [90].	183
6.10	Hyperparameters for MAE pre-training on JFT-300M and IN-21K with ViT-L models. The only difference from the IN-1K configuration is the learning rate, which we reduced to stabilize training.	184
6.11	Hyperparameters for finetuning CAN and SimCLR pre-trained models on ImageNet. We use the same hyperparameters for ViT-B and ViT-L, for JFT-300M, IN-21K and IN-1K pre-trained models. For MAE, we use the settings from Table 9 of [90].	185
6.12	Hyperparameters for linear probing pre-trained models on ImageNet. We use the same hyperparameters for ViT-B and ViT-L, and for JFT-300M, IN-21K and IN-1K pre-trained models. Note that these hyperparameters are same as reported in [90].	185

List of Figures

2.1	An overview of our approach. We propose to augment the ImageNet dataset by adding Anisotropic diffused images. The use of this augmentation helps the network rely less on texture information and increases performance in several settings.	9
2.2	Four different methods for reducing texture in images.	11
2.3	The figure shows the pipeline for training a Pix2Pix model with the classification loss. The input to the Pix2Pix network is a normal ImageNet image and the output is an Anisotropic image. Both of these images are used further for the classification task.	17
2.4	Saliency maps using GradCam. The text on the left of the row indicates whether the Anisotropic model or ImageNet model was used. The figure shows the saliency maps where the Anisotropic model gave correct predictions, and the ImageNet model gave wrong predictions. The failure of the ImageNet model might be due to it not attending to the whole object.	26
2.5	Examples of images from Sketch-ImageNet. Images have very little or no texture, which implies texture will have little to no impact on object classification.	27
2.6	Training plots for the both Stylized ImageNet and Standard ImageNet. The plot shows that the model trained on Stylized ImageNet quickly overfits by finding shortcuts after around 6000 steps. Therefore, it gives poor performance on downstream tasks by relying on texture based shortcuts.	28
2.7	Results on Label corruption task. The bottom two line corresponds to when the gold fraction is 0.05, and the top two lines correspond to the case when the gold fraction is 0.1. We can see that our model consistently outperforms the baseline with a larger improvement upon increasing the corruption probability.	30
2.8	Results after removing high-frequency components from images. We can see that our proposed anisotropic filtering leads to models that are less reliant on high-frequency components in an image. When we remove the highest frequency components from the image(far-right), our model, outperforms baseline by close to 2.1%. As we remove more high-frequency components, both the model suffers, but our model consistently performs better.	31

2.9	Saliency maps on three different set of images. The text on the left of the row indicates whether Anisotropic model or ImageNet model was used. The first two rows show the saliency maps where both model gave correct predictions. We can see from saliency maps that the Anisotropic model has more diffused saliency maps. The second two rows show the saliency maps where Anisotropic model gave correct predictions and ImageNet model gave wrong predictions. The failure of ImageNet model might be due to it not attending to whole object. The last two rows show the saliency maps where Anisotropic model gives incorrect predictions and ImageNet model gives correct predictions. Even in this failure mode, the Anisotropic model gives diffused saliency maps.	42
2.10	Saliency maps when our technique gives the correct prediction and baseline approach gives incorrect label. The top row gives the saliency maps for our model and the bottom one shows the corresponding saliency maps for the model trained on imagenet alone. We can see from saliency maps that Anisotropic model has bigger saliency maps which might be the reason for the correct prediction.	43
2.11	Original images (left)and images obtained after anisotropic diffusion (right). Most of the texture information in the images has been smoothed out by the filter while retaining the shape information. This forces the network to capture higher-level semantics without relying on low-level texture cues	44
2.12	Original images (left)and images obtained after anisotropic diffusion (right). Most of the texture information in the images has been smoothed out by the filter while retaining the shape information. This forces the network to capture higher-level semantics without relying on low-level texture cues	45
2.13	Saliency maps on few randomly selected images from Sketch-ImageNet. We can see from saliency maps that Anisotropic model has bigger saliency maps which might be the reason for the correct prediction. .	46
2.14	Saliency maps when Anistropic Model had correct predictions and ImageNet model has wrong predictions.	46
2.15	Saliency maps when both model have wrong predictions.	46
2.16	Saliency maps when both model have correct predictions.	47
2.17	Saliency maps when ImageNet model has correct predictions and Anistropic model has wrong predictions.	47
3.1	We propose to construct negative samples (NS) from input images for contrastive learning with augmentations that only preserve non-semantic information such as texture and local features.	49
3.2	The histogram of cosine similarity between the representations of query sample and its paired samples, namely positive sample and standard, texture-based, and patch-based negative samples (NS), using MoCo-v2 model trained on the ImageNet-1K dataset for 200 epochs.	57

3.3	The histogram of cosine similarity between the representations of query sample and its paired positive sample, standard, texture-based, and patch-based negative samples (NS), using models trained without (blue) and with patch-based negative samples (red: $\alpha = 2$, green: $\alpha = 3$).	60
3.4	Top-1 accuracy on the STL-10 dataset with different memory bank sizes.	62
3.5	Top-1 accuracy on the ImageNet-100 dataset with different memory bank sizes.	62
3.6	Larger α monotonically increases the model bias to shape features over texture features. Model performance is impacted by such a trade-off differently under different settings. In all scenarios, slightly calibrated shape bias improves model performance. The test settings represented in the red lines gain more from the increased shape bias than the settings represented in the green lines.	64
3.7	A ResNet-50 model trained on the ImageNet dataset achieve decent accuracy when only texture features are available on some classes. A normal image and its texture version are displayed for some of these classes. The caption indicates the class ID, name, and accuracy on texture images.	66
3.7	Randomly picked images from our ImageNet-Texture dataset and their corresponding original images from the ImageNet dataset.	77
3.8	More randomly picked images from our ImageNet-Texture dataset and their corresponding original images from the ImageNet dataset.	78
3.9	(a) Accuracy on the ImageNet-Texture versus standard ImageNet with different model architectures. (b) Histogram of accuracy on the individual ImageNet-Texture classes with ResNet-50.	79
3.10	On some classes, a ResNet-50 model trained on the standard ImageNet dataset can achieve $> 50\%$ accuracy when only texture features are available. For each class, one sample image and its texture version are displayed. The caption of each image pair indicates the ImageNet class ID, class name and ResNet-50 accuracy on the texture images.	80
3.11	Test accuracy of patch-based non-semantic augmentations with patch sizes sampled from different uniform distribution. The x-axis indicates the lower boundary of the sampling interval and the y-axis indicates the higher boundary.	81
3.12	The histogram of cosine similarity between the representations of different kinds of pairs using models trained without (blue) and with (red) texture-based negative samples.	81
3.13	The histogram of cosine similarity between the representations of different kinds of pairs using models trained with patch-based negative samples and different α values from 0 (top) to 5 (bottom).	82

4.1	Illustration of object aware cropping. Top-Left: We show the original image with random crops overlaid. Bottom (red panel): Overlap between random crops tend to miss the object of interest. Top-Right: We show crops generated from the LOD [238] algorithm and also the dilated object crop. Bottom-Right (green panel): We use LOD-based object-aware crops. These incorporates both object and scene information into the MoCo-v2 (or other SSL frameworks).	87
4.2	Analysis of OHMS data distribution. Left: Performance of supervised and MoCo-v2 pre-training as a function of the scale of the objects; we plot the log of the average of pixels against the sum of AP for each class. We see no discernible pattern of performance of MoCo-v2 or supervised learning as a function of object scale. Right: Performance of supervised learning and MoCo-v2 as a function of the number of instances in a class; we plot the log number of instances in a class against the sum of AP for that class. We do not see any discernible pattern of performance difference as a function of class size.	92
4.3	Left: distribution of number (log) of images in each class for each of 208 selected classes. Middle: distribution after removing classes which have more than 10k images. Right: scatter plot of the height and width.	107
4.4	Scatter plot of the height and width of the OpenImages Dataset.. . . .	108
4.5	Results on COCO then transferred to VOC. Left: Varying the temperature parameter. Right: Varying radius for object-object crop.	109
4.6	Left: We vary the amount of overlap between object and scene crops, as shown. Optimal VOC detection performance is achieved around 58% overlap, decreasing on either side. The purple dot indicates overlap for scene-scene crops (66%). Right: We vary the radius of the object-object crops and observe a similar phenomenon of a “sweet spot” distance between the object crops at which downstream performance is maximized.	110
4.7	Visualization of BING and random crops.	115
4.8	Visualization of BING and random crops.	115
4.9	Visualization of BING and random crops.	116
4.10	Visualization of BING and random crops.	116
4.11	Visualization of BING and random crops.	117
4.12	First column: Two samples from OpenImages dataset. Second and Third Columns: Random scene crops; Fourth and fifth columns: Random object crops generated by the BING [42] algorithm. We see that the object crop tends to center on objects in the image that are often missed by the scene crops.	117
5.1	Illustration of the representation space learned by our models. Object images of the same class tend to gather near the center around similar directions, while the scene images are far away in these directions with larger norms.	124

5.2	Our Hyperbolic Contrastive Learning (HCL) framework has two branches: given a scene image, two object regions are cropped to learn the object representations with a loss defined in the Euclidean space focusing on the representation directions. A scene region as well as a contained object region are used to learn the scene representations with a loss defined in the hyperbolic space that affects the representation norms.	131
5.4	Images from ImageNet training set. The 5 images on the left have the smallest representation norms among all the images from the same class, and the 5 on the right have the largest norms.	137
5.3	Average representation norms of images with different number of labels in ImageNet-ReaL [11].	137
5.5	Out-of-context images from the SUN09 dataset [43]. The bounding box of each object, as well as its hyperbolic distance to the scene are displayed. The regular objects are in blue and the out-of-context objects are in purple. Note that the out-of-context objects tend to have large distances.	140
5.6	More images from ImageNet training set sorted by their representation norms.	148
5.7	More images from ImageNet training set sorted by their representation norms.	149
6.1	Left: CAN scales better than SimCLR since it uses masked inputs. Middle and right: CAN outperforms SimCLR and MAE on ImageNet linear probe and finetune evaluations for ViT-L models when pre-training on uncured data such as JFT-300M.	154
6.2	The CAN framework: Two views of an image are generated, 50% of patches randomly masked in each, and noise is added to patches. An encoder is trained to solve three tasks: 1) Reconstruction: encoded patches are passed to a decoder that reconstructs missing patches, 2) Denoise: reconstructs the noise added to unmasked patches, and 3) Contrast: pooled patches are passed to a contrastive loss, using in-batch samples as negatives [34].	158
6.3	Denoising: Both the encoded patches and the noise level σ are passed to the decoder by passing σ through an MLP, and adding the result to each embedded token.	162
6.4	Few-shot: ViT-L models pre-trained on JFT-300M for 5000 epochs are evaluated on 9 datasets in few-shot setting (10-shot and 25-shot). CAN outperforms MAE and SimCLR.	168
6.5	Robustness: Evaluating performance under distribution shifts with respect to models finetuned on IN-1K. Validation performance of ViT-L models is reported on 7 different datasets.	169
6.6	CAN and SimCLR with different masking rates. ViT-B models are pre-trained for 100 epochs on IN-1K (left), and 800 epochs on JFT-300M (right).	170

6.7	ViT-B models pre-trained on IN-1K for 100 epochs. Left: The best contrastive loss weight is small but non-negative. Middle: A wide range of $\sigma_{\max}$ values improve over no-noise. Right: Performance is not sensitive to the denoising loss weight.	170
6.8	ViT-L models pre-trained on JFT-300M for 800 and 1600 epochs respectively, evaluated on 7 datasets with distribution shifts from IN-1K.	174
6.9	Top: ViT-B models pre-trained on JFT-300M for 800 epochs, evaluated on 7 datasets with distribution shifts from IN-1K. Bottom: Comparison of our JFT-300M pre-trained ViT-B model to training ViT-B from scratch on IN-1K. We compare to standard supervised cross-entropy training with Mixup, and to PyramidAT [105], which uses an adversarial training method. CAN considerably outperforms supervised training, and beats PyramidAT in 6 out of 7 cases without requiring adversarial training.	175
6.10	Few shot: ViT-L models pre-trained on JFT-300M for 5000 epochs evaluated on 9 few-shot learning tasks. Results accompany the 10- and 25-shot results in Figure 6.4.	175
6.11	Few shot: ViT-L models pre-trained on JFT-300M for 800 epochs are evaluated on 9 few-shot learning tasks.	176
6.12	Few shot: ViT-L models pre-trained on JFT-300M for 1600 epochs are evaluated on 9 few-shot learning tasks.	176
6.13	Few shot: ViT-B models pre-trained on JFT-300M for 800 epochs are evaluated on 9 few-shot learning tasks.	177
6.14	Few shot: ViT-L models pre-trained on IN-1K for 800 epochs are evaluated on 9 few-shot learning tasks.	177
6.15	Robustness: ViT-L models pre-trained on IN-21K for 800 (IN-1K equivalent) epochs are first finetuned on IN-1K. The models are then evaluated on 7 test datasets with different distribution shifts from IN-1K.	179
6.16	Few shot: ViT-L models pre-trained on IN-21K for 800 (IN-1K equivalent) epochs are evaluated on 9 few-shot learning tasks.	179

Chapter 1: Introduction

1.1 Objective

Self-Supervised learning in recent years has made great progress and has shown that it is possible to learn high-level representations of object categories from unlabelled images [35,91]. The most studied benchmark dataset in self-supervised learning is ImageNet [56], where representations learned from unlabeled images can surpass supervised representations, both in terms of their data-efficiency and transfer-learning performance [27,91]. The ultimate goal of SSL is to work well with large scale uncurated datasets. However SSL methods show sub-par performance when pre-trained on large scale uncurated datasets like IG-1B [91] and JFT [214]. There are multiple challenges in SSL that have hampered its progress. In this manuscript we discuss three main challenges. The first limitation of these methods is their over-reliance on low level cues like texture and second is poor performance on multi-object scene datasets like OpenImages [137], COCO [148]. In this manuscript we firstly deal with the problem of removing low-level shortcuts such as texture and then we work on the problem of training SSL methods on multi-object scene images. Lastly we discuss how to efficiently scale SSL methods on large scale datasets like JFT [214].

1.2 A Solution for Texture

We begin in Chapter 2 by addressing the problem of texture bias in CNNs . Recent literature [77] has shown that features obtained from supervised training of CNNs may over-emphasize texture rather than encoding high-level information. In self-supervised learning in particular, texture as a low-level cue may provide shortcuts that prevent the network from learning higher-level representations. We hypothesize that retaining more edge information and suppressing texture can help in alleviating these problems. To this end, we propose to use a simple classical idea based on anisotropic diffusion to augment training using images with suppressed texture. We empirically show that augmenting ImageNet with texture suppressed ImageNet improves results on image classification with five diverse datasets in both supervised or self-supervised learning tasks such as MoCoV2 [92] and Dense-CL. Our method is particularly effective for transfer learning tasks, and we observe improved performance on twelve transfer learning datasets. The large improvements (up to 11.49%) on the Sketch-ImageNet [241], and Synthetic-DTD datasets [174], and additional visual analyses of saliency maps suggest that our approach helps in learning better representations that transfer well to downstream tasks. We show that our method is simple to implement and can be integrated into various computer vision tasks easily. We also compare our approach with Gaussian blur augmentation, recently used in [35, 39], and show that Anisotropic-filtering for texture suppression is better because isotropic Gaussian blur can potentially suppress edges and other higher-level semantic information as well.

Contrastive learning has been the main driving force behind the exceptional progress of unsupervised learning. However contrastive learning methods are also prone to depend on low-level features that humans deem non-semantic. This dependency has been conjectured to induce a lack of robustness to image perturbations or domain shift. In Chapter 3, we show that by generating carefully designed negative samples, contrastive learning can learn more robust representations with less dependence on such features. Contrastive learning utilizes positive pairs that preserve semantic information while perturbing superficial features in the training images. Similarly, we propose to generate negative samples in a reversed way, where only the superfluous instead of the semantic features are preserved. We develop two methods, texture-based and patch-based augmentations, to generate negative samples. We evaluate our methods with two contrastive learning methods, MoCo [39, 92] and BYOL [85], on three datasets, ImageNet-100, ImageNet-1K and STL-10. When using our proposed augmentations to generate negative samples and minimizing their representation similarity to the input images, we notice a consistent improvement in generalization performance over backbone methods [39, 85] and previous negative example generation strategies [122, 193], especially under out-of-distribution (OOD) settings. We also analyze our method and the generated texture-based samples, showing that texture features are indispensable in classifying particular ImageNet classes and especially finer classes. A larger texture bias benefits more fine-grained classification such as dog breeds, while larger shape bias helps shape based datasets like ImageNet-Sketch [241].

1.3 A Solution for Multi-Object Scene images

We next address the issue of training SSL on multi-object datasets in Chapter 4. SSL methods traditionally have been shown to work well on object centered datasets like ImageNet, but they perform poorly on complicated multi-object scene images. A core component of the recent success of self-supervised learning is cropping data augmentation, which selects sub-regions of an image to be used as positive views in the self-supervised loss. The underlying assumption is that randomly cropped and resized regions of a given image share information about the objects of interest, which is captured by the learned representation. This assumption is mostly satisfied in datasets such as ImageNet where there is a large, centered object, which is highly likely to be present in random crops of the full image. However, in other datasets such as OpenImages or COCO, which are more representative of real world uncurated data, there are typically multiple small objects in an image. In this work, we show that self-supervised learning based on the usual random cropping performs poorly on such datasets (measured by the difference from fully-supervised learning). Instead of using pairs of random crops, we propose to leverage an unsupervised object proposal technique; the first view is a crop obtained from this algorithm, and the second view is a dilated version of the first view. This encourages the self-supervised model to learn both object and scene level semantic representations. Using this approach, which we call *object-aware cropping*, results in significant improvements over random scene cropping on classification and object detection benchmarks. We conduct a number of experiments incorporating object-aware cropping, finding consis-

tent improvements on state of the art self-supervised methods such as MoCo-v2 [39], BYOL [86] and Dense-CL [245] across varied datasets and tasks. When we pre-train on OpenImages, our approach achieves an improvement of 8.8% mAP over random scene cropping (both methods using MoCo-v2). We also show significant improvements on COCO and PASCAL-VOC object detection and segmentation tasks over the state-of-the-art self-supervised learning approaches. Our approach is efficient, simple and general, and can be used in most existing contrastive and non-contrastive self-supervised learning frameworks.

Self-Supervised learning methods generally treat objects and scenes using the same lens. In Chapter 5, we focus on learning representations for objects and scenes explicitly in the same space. Motivated by the observation that visually similar objects are close in the representation space, we argue that the scenes and objects should instead follow a hierarchical structure based on their compositionality. To exploit such a structure, we propose a contrastive learning framework where a Euclidean loss is used to learn object representations and a hyperbolic loss is used to encourage representations of scenes to lie close to representations of their constituent objects in a hyperbolic space. This novel hyperbolic objective encourages the scene-object hypernymy among the representations by optimizing the magnitude of their norms. We show that when pretraining on the COCO and OpenImages datasets using MoCo-v2 [91], ORL [254] and Dense-CL [246] pre-text tasks, the hyperbolic loss improves downstream performance across multiple datasets and tasks, including image classification, object detection, and semantic segmentation. We also show that the properties of the learned representations allow us to solve various vision

tasks that involve the interaction between scenes and objects in a zero-shot way. We show results on label uncertainty quantification and also out-of-context object detection on the SUN09 dataset [43].

1.4 Scaling Self-Supervised learning to large scale datasets:

In Chapter 6 we introduce CAN, a simple, effective, and scalable approach for self-supervised visual representation learning. CAN is an elegant fusion of (C) contrastive learning, (A) masked autoencoders, and (N) the noise prediction technique from diffusion models. These components work harmoniously: contrastive learning shapes the embedding space across a batch of image samples, masked autoencoders focus on reconstructing low-frequency spatial correlations within individual images, and noise prediction encourages the reconstruction of high-frequency components. The result is a robust, scalable, and straightforward algorithm. During training, we symmetrically mask 50% of patches in both views, greatly enhancing efficiency compared to previous contrastive learning methods. Extensive empirical experiments demonstrate CAN’s strong performance in downstream tasks, including linear and fine-tuning evaluations for transfer learning and robustness. CAN outperforms MAE and SimCLR when pre-training on ImageNet, and it shines when pre-training on larger, uncured datasets like JFT-300M. For instance, in a linear probe on ImageNet, CAN achieves an accuracy of 75.4%, outperforming SimCLR at 73.4% and MAE at 64.1%. In terms of finetuned performance on ImageNet with our ViT-L model, CAN reaches 86.1%, surpassing SimCLR’s 85.5% and MAE’s 85.4%. Fur-

thermore, CAN exhibits significantly lower FLOPs (floating-point operations) load compared to SimCLR, particularly in ViT-L models.

Chapter 2: Learning Visual Representations For Transfer Learning By Suppressing Texture

2.1 Introduction

Deep convolutional neural networks (CNNs) learn powerful visual features that have resulted in significant improvements on many computer vision tasks such as semantic segmentation [204], object recognition [134], and object detection [190]. However, CNNs often fail to generalize well across datasets under domain-shift due to varied lighting, sensor resolution, spectral-response etc. One of the reasons for this poor generalization is CNNs’ over-reliance on low-level cues like texture [76].¹ These low-level cues and texture biases have been identified as grave challenges to various learning paradigms ranging from supervised learning [15, 76, 192] to self-supervised learning (SSL) [21, 59, 62, 177, 179]. We propose to use classical tools to suppress texture in images as a form of data augmentation to encourage deep neural networks to focus more on learning representations that are less dependent on textural cues. We use the Perona-Malik non-linear diffusion method [185], robust Anisotropic diffusion [12], and Bilateral filtering [224] to augment our training data. These methods

¹The contents of this work is in collaboration with Anshul Shah, Ankan Bansal, Jonghyun Choi, Abhinav Shrivastava, Abhishek Sharma and David Jacobs [167].

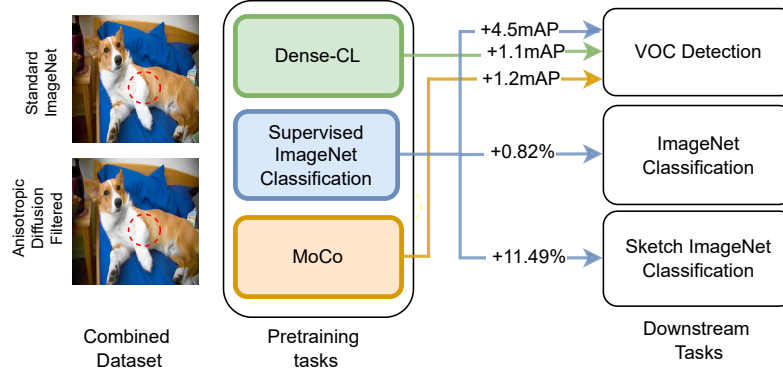


Figure 2.1: An overview of our approach. We propose to augment the ImageNet dataset by adding Anisotropic diffused images. The use of this augmentation helps the network rely less on texture information and increases performance in several settings.

suppress texture while retaining structure by preserving boundaries.

Our work is inspired by the observation that ImageNet pre-trained models fail to generalize well across datasets [76, 189], due to over-reliance on texture and low-level features. Stylized-ImageNet [76] attempted to modify the texture of images using style-transfer to render images in the style of randomly selected paintings from the Kaggle paintings dataset. However, this approach offers little control over exactly which cues are removed from the image. The resulting images sometimes retain texture and distort the original shape. Stylized-ImageNet especially doesn’t work well in the case of SSL since the network learns the texture and uses those textures to solve the SSL tasks (see Tab 2.2 Row1. In our approach (Fig. 5.1), we suppress the texture instead of modifying it. We empirically show that this helps in learning better higher-level representations and works better than CNN-based stylized augmentation. We compare our approach with Gaussian blur augmentation, recently used in [35, 39], and show that Anisotropic-filtering for texture suppression is better because isotropic Gaussian blur can potentially suppress edges and other

higher-level semantic information as well. Our proposed method works well in self-supervised and supervised learning tasks, and we outperform both Gaussian blurring and Stylized-ImageNet in both settings.

Anisotropic-filtering is simple to implement and can be integrated easily in various computer vision-based methods. In the case of supervised learning, we pre-train on ImageNet, and test on twelve different datasets including ImageNet, Pascal VOC [67], Synthetic-DTD [174], CIFAR 100 [101], Sketch ImageNet [241], etc. In the self-supervised setting, we use two learning frameworks: Dense-CL [246], and MoCoV2 [39] and pre-train on ImageNet and COCO. Our texture-suppressing augmentation consistently outperforms MoCoV2 and Dense-CL, which uses Gaussian blurring, on transfer learning experiments on VOC classification, detection, segmentation benchmarks, and also on classification tasks for other transfer learning datasets, including DTD [46], Cars [132], Aircraft [161], etc. With the help of qualitative and quantitative analysis, we show that our model is less reliant on high-frequency information in images and is more robust to common corruptions on datasets like ImageNet-C [100], and CIFAR-100 [101]. Our model also learns better shape bias than a Standard-ImageNet pretrained model and is also more confident in making correct predictions. Overall, we achieve significant improvements on several benchmarks:

- In a set of **twelve** diverse datasets, our method exhibits substantial improvements (as high as +11.49% on Sketch ImageNet and +10.41% on the DTD dataset) in learning visual representations across domains.

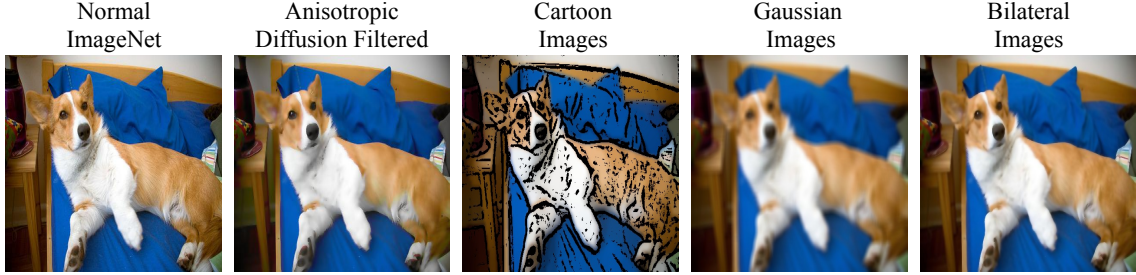


Figure 2.2: Four different methods for reducing texture in images.

- We also get improvements in the same domain visual recognition tasks on ImageNet validation (+0.82%), and on label corruption task [101].
- We achieve improved results in self-supervised learning on image classification transfer learning tasks and on VOC detection.

2.2 Related Work

In this section, we review relevant methods that aim to remove texture cues from images to reduce the dependency of CNNs on low-level cues. Since we also experiment with the application of our method in self-supervised learning, we review recent work in this area as well.

Reliance on Low-Level Texture Cues. Recent studies have highlighted that deep CNNs can leverage low-level texture information for classification on the ImageNet dataset. Contrary to popular belief that CNNs capture shape information of objects using hierarchical representations [139], the work in [76] revealed that CNNs trained on ImageNet are more biased towards texture than shape information. This dependency on texture not only affects generalization, but it can also limit the performance of CNNs on emerging real-world use-cases like few-shot image

classification [192]. [15] showed that a bag of CNNs with limited receptive field in the original image can *still* lead to excellent image classification performance. Intuitively, a small receptive field forces the CNNs to heavily rely on local cues vs learning hierarchical shape representations. This evidence strongly suggests that texture alone can yield competitive performance on ImageNet, and the fact that it’s relatively easier to learn vs hierarchical features may explain deep CNNs’ bias towards texture.

To reduce reliance on texture, Stylized-ImageNet [76] modified the ImageNet images into different styles taken from the Kaggle Painter by Numbers dataset. While trying to remove texture, this approach could also significantly affect the shape. Also, there isn’t an explicit control over the amount of removed texture. Moreover, this method may not be directly applicable to self-supervised learning because the fixed number of possible texture patterns result in images with strong low-level visual cues resulting in shortcuts. We show that the accuracy on downstream tasks, when MoCoV2 and Jigsaw are trained with Stylized-ImageNet, decreases dramatically Table 2.11. Some of the recent methods inspired by Stylized-ImageNet [76] that try to learn better shape representation [115, 146, 172] face similar issues. Info-Dropout [205] tries to learn shape information by using dropout based methods that zero out neurons with a high probability if the input patch contains less self-information.

On the other hand, we use Perona-Malik’s anisotropic diffusion [185] and bilateral filtering [224] as ways of suppressing texture in images. These methods remove texture without degrading the edge information. Consequently, the shape information

of the objects is better preserved. Also, these methods provide finer control over the level of texture suppression. Suppressing the texture in training images forces CNNs to build representations that put less emphasis on texture. We show that such data augmentation can lead to performance improvements in both supervised and self-supervised settings. We also distinguish our work from other data augmentation strategies like Auto-Augment [50] which uses Reinforcement Learning to automatically search for improved data augmentation policies and introduces Patch Gaussian Augmentation, which allows the network to interpolate between robustness and accuracy [157].

Self-Supervised Learning. To demonstrate the importance of removing texture in the self-supervised setting, we consider two pretext tasks. The first pretext task is Jigsaw [177], a patch-based self-supervised learning method that falls under the umbrella of visual permutation learning [48, 49]. Some of the most recent self-supervised methods are contrastive learning based methods [21, 22, 26, 35, 39, 91, 106, 113, 123, 141, 170, 221, 246]. In [21], the authors have proposed using contrastive losses on patches, where they learn representations by predicting representations of one patch from another. In MoCo [91], a dynamic dictionary is built as a queue along with a moving average encoder. Every image will be used as a positive sample for a query based on a jittered version of the image. The queue will contain a batch of negative samples for the contrastive losses. MoCo has two encoder networks. The momentum encoder has weights updated through backpropagation on the contrastive loss and a momentum update. In MoCoV2, Gaussian blur and linear projection layers were added that further improve the representations. MoCo

and MoCoV2 have shown competitive results on ImageNet classification and have outperformed supervised pre-trained counterparts on seven detection/segmentation tasks, including PASCAL VOC [67], and COCO [148].

Transfer Learning. Transfer learning is one of the most important problems in computer vision due to difficulty in collecting large datasets across all domains. In this work, we discuss transfer learning in the context of ImageNet. A lot of early datasets were shown to be too small to generalize well to other datasets [226]. Following this, many new large-scale datasets were released [56, 148], which are believed to transfer better. However, recent results show that these datasets do not generalize well in all cases [131, 189]. [131] showed that ImageNet features generally transfer well, but not to fine-grained tasks. We show results of transfer learning on some of the datasets that were used by [131].

2.3 Methods

CNN-based classifiers have been shown to exploit textures rather than shapes for classification [15, 76]. We aim to reduce the prominence of texture in images and thus encourage networks trained to learn representations that capture better higher-level representations.

Gaussian Blur. Gaussian blurring is one of the most popular smoothing methods in computer vision, and it has been recently proposed as data augmentation for SSL [35, 39]. However, along with low-level texture, Gaussian filtering also blurs across boundaries, diminishing edges and structural information.

2.3.1 Anisotropic diffusion

We propose to use Anisotropic Diffusion Filters (ADF) [185], which keep the shape information coherent and only alter low-level texture. Specifically, we use Perona-Malik diffusion [185]. These filters smooth the texture without degrading the edges and boundaries. Intuitively, this will encourage the network to extract high-level semantic features from the input patches.

Perona-Malik diffusion smooths the image using the differential diffusion equation:

$$\frac{\partial I}{\partial t} = c(x, y, t)\Delta I + \nabla c \cdot \nabla I \quad (2.1)$$

$$c(x, y, t) = e^{-(\|\nabla I(x, y, t)\|/K)^2} \quad (2.2)$$

where I is the image, t is the time of evolution, Δ is the Laplacian operator, K controls sensitivity to edges, ∇ is gradient and (x, y) is a location in the image. The amount of smoothing is modulated by the magnitude of the gradient in the image through c the diffusion coefficient. The larger the gradient, the smaller the smoothing at that location. Therefore, after applying Anisotropic diffusion, we obtain images with blurred regions, but edges are still prominent. Fig. 2.2 shows some examples of the application of the filter. Note that ADF reduces the texture in the image without replacing it, the domain gap between images is not large, while in the case of Stylized ImageNet, the domain shift will be large. Recently, there has been some work on removing textures using deep learning as well [153, 160, 258].

We find, though, that fast and simple classical methods work well on most tasks. For all our experiments, we create a dataset ‘Anisotropic ImageNet’ by adding ADF filtered ImageNet images to the standard ImageNet dataset.

2.3.2 Pix2Pix for suppressing texture

In addition to using classical tools, we also apply recent Image-to-Image translation models like Pix2Pix [116] to suppress texture. We train pix2pix model to produce images that are similar in style to anisotropic diffusion. Pix2pix model also helps in capturing different variations of texture, e.g., the amount of smoothing, according to the target task. The source domain for Pix2Pix model is normal ImageNet images, and the target domain is texture suppressed images. We only apply Pix2Pix model on Perona-Malik diffusion [185], which is our best performing anisotropic diffusion method. After adding Pix2Pix models and doing an end-to-end training for ImageNet classification, we are able to achieve better results and outperform Stylized-ImageNet on **top-1** accuracy. We call this model Pix2Pix-Anisotropic model. For MoCoV2 we train using ten different anisotropic diffusion variations where the iterations are randomly picked between 10 to 20. Since we are already able to capture these variations and also due to the huge compute requirements and time to train SSL and Pix2Pix models, we only show these results in the supervised learning setting and not for MoCoV2. In Fig 2.3 we show our complete pipeline of training pix2pix model. We pretrain the pix2pix network on ImageNet dataset for 8 epochs and use this network for the end to end classification.

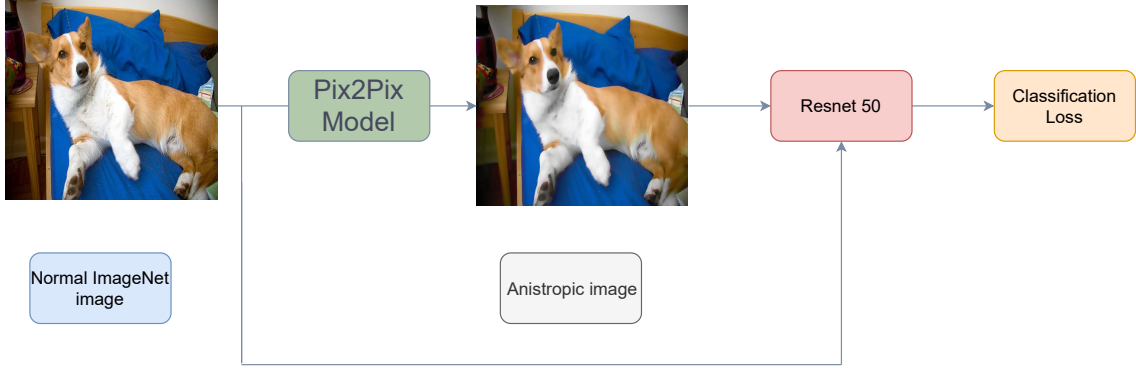


Figure 2.3: The figure shows the pipeline for training a Pix2Pix model with the classification loss. The input to the Pix2Pix network is a normal ImageNet image and the output is an Anisotropic image. Both of these images are used further for the classification task.

2.3.3 Other texture suppressing methods

We also experiment with a few other texture removing methods like robust Anisotropic diffusion [12], Bilateral filtering [224], and Cartoonization. However, empirically we find that the most simple Anisotropic diffusion method has the best results as discussed in Section 2.4.2. We will discuss these other texture removing methods briefly next.

Bilateral Filtering: [225] is an efficient method of anisotropic diffusion. In Gaussian filtering, each pixel is replaced by an average of neighbouring pixels, weighted by their spatial distance. Bilateral Filtering is its extension in which weights also depend on photometric distance. This also limits smoothing across edges, in which nearby pixels have quite different intensities.

Cartoonization: A more extreme method of limiting texture is to create cartoon images. To convert an image into a cartoonish image, we first apply bilateral filtering to reduce the image’s colour palette. In the second step, we convert the

actual image to grayscale and apply a median filter to reduce noise in the grayscale image. After this, we create an edge mask from the greyscale image using adaptive thresholding. Finally, we combine these two images to produce cartoonish looking images (see Fig. 2.2).

2.4 Experiments

We start by briefly describing the datasets used in our experiments. We then show the effectiveness of ADF for supervised and self-supervised learning. We find that ADF is particularly effective when there is a domain shift, supporting our hypothesis that variation in texture is a significant effect of domain shift. The effect is larger when we transfer from ImageNet to datasets such as Sketch Imagenet [241], and Synthetic-DTD [174], where the domain shift is larger. Our method is also able to outperform Stylized-ImageNet [76] and Gaussian Blur [35].

Datasets. In all experiments, we use the ImageNet training set as the source of our training data. For object detection and semantic segmentation, we evaluate on Pascal VOC 2007 and VOC 2012. For label corruption, we evaluate on CIFAR100. When the downstream task is classification we evaluate on Synthetic-DTD [174], Sketch-ImageNet [241], Birds [239], Aircraft [161], Stanford Dogs [124], Stanford Cars [132], DTD [46], ImageNet-C [100], and the ImageNet validation dataset.

Experimental Details. For SSL we build on MoCoV2 [39] and Dense-CL [246]. For supervised learning, we use the ResNet50 [94] model, closely following [76]. After training on Anisotropic ImageNet, we fine-tune our model on the

standard ImageNet training set following the procedure of [76]. We set the conduction coefficient (K) of Anisotropic Diffusion to 20, and a total of 20 iterations are used. We use the MedPy implementation. All other hyper-parameters are described in the sec 2.8.

2.4.1 Self-Supervision for Transfer Learning

We first experiment with Anisotropic ImageNet on Self-Supervised methods. We have doubled the number of images (Anisotropic images + normal images) as compared to normal ImageNet. So for a fair comparison, we only train our methods for half the number of epochs compared to training with just ImageNet. We then fine-tune the network pre-trained on the Anisotropic ImageNet for the downstream tasks, including image classification, object detection, and semantic segmentation on PASCAL VOC, and other transfer learning datasets. Since, we are removing low-level cues from the images, we expect to see better results when transferring to different datasets.

MoCo V2. We evaluate our method with MoCo V2 [39] and Dense-CL [246], which are the state-of-the-art methods in SSL. MoCoV2 and Dense-CL [246] used Gaussian blurring with 0.5 probability as data augmentation. In our case, we add Anisotropic diffusion on the images with 0.5 probability, and for the remaining 50% of the images, we apply Gaussian blurring with 0.5 probability. So, in our setup every image has 0.5 probability of coming from Anisotropic ImageNet, 0.25 of Gaussian blurring, and 0.25 of being normal ImageNet. Also, the number of

iterations on Anisotropic filtering is chosen randomly between 10 to 20. For object detection starting from a MoCoV2 initialization, we train a Faster R-CNN [190] with C4-backbone, which is fine-tuned end-to-end.

We show improvements over MoCoV2 and Dense-CL for object detection on the VOC Dataset. In the first setup, we show improvements on COCO-based evaluation metrics (i.e., AP_{50} , $AP_{0.05:0.05:0.95}$, AP_{75}) as shown in the first three columns of Table 2.1, achieving competitive results. We also observe an improvement of 1.3 mean IoU on semantic segmentation [155] over MoCo V2 baseline and 1.1 over Dense-CL baseline. We also see improved performances on Dense-CL when we pre-train on COCO dataset as well. These results show that in the case of transfer learning, we improve across different datasets. Our method is not bound to a particular pretext task and can be potentially added to any state-of-art method to achieve even further improvements. We also show that our method leads to improvements with the Jigsaw [179] task in Tab 2.8.

These results suggest that training the network on the Anisotropic ImageNet dataset forces it to learn better representations. This is consistent with our hypothesis that Anisotropic diffusion leads to smoothing of texture in images. This forces the network to be less reliant on lower-level information to solve the pretext task and, hence, learn representations that focus on higher-level concepts.

Experiments with Stylized ImageNet on MoCoV2 and Jigsaw. We now show experiments that indicate that, while effective in a supervised setting, Stylized ImageNet does not help with SSL. We train a model with MoCoV2 and Jigsaw as pretext tasks on the Stylized-ImageNet (SIN) dataset [76] and fine-tune on

Table 2.1: Comparison with MoCoV2 and Dense-CL in SSL. We note that using Anisotropic diffusion with improves performance on VOC detection and Semantic Segmentation (SS). We test on COCO-based metrics as used in [39]. We also improve performance over the baseline on the semantic segmentation (SS) task [155]. Although we have only focussed on MoCoV2 Dense-CL, our technique can potentially be extended to other state-of-art methods.

Methods	Dataset	AP ₅₀	AP	AP ₇₅	mIoU (SS)
Stylized ImageNet		43.5	28.80	33.7	-
Supervised ImageNet		81.6	54.2	59.8	59.8
MoCo V2 [39]	ImageNet	82.4	57.0	63.6	67.5
MoCo V2 Anistropic (Ours)	ImageNet	83.7	58.2	64.8	67.8
Dense-CL [246]	ImageNet	82.8	58.7	65.2	69.4
Dense-CL [246] Anistropic (Ours)	ImageNet	83.5	59.6	66.4	70.5
Dense-CL CC [246]	COCO	81.7	56.7	63.0	67.5
Dense-CL CC Anistropic (Ours) [246]	COCO	83.1	57.9	64.2	68.6

Table 2.2: Transfer learning across different datasets. Note that our approach leads to improvements in both supervised and SSL set-up.

Dataset	Aircraft [161]	Birds [239]	Dogs [124]	Cars [132]	DTD [46]
Supervised (Reproduced)	90.88	90.3	85.35	92.1	72.66
SimCLR [35]	88.1	-	-	92.1	73.2
BYOL [86]	88.1	-	-	91.7	76.2
MoCo V2 [39]	91.57	92.13	87.13	92.8	74.7
MoCo V2 Anistropic (Ours)	92.71	93.29	88.81	94.3	76.3

the downstream tasks of object detection and image classification on PASCAL VOC.

In Table 2.2 (and Table 2.11), we show that there is a huge drop in performance.

One reason for this failure using the SIN dataset could be that the model is able to memorize the textures in the stylized images since it only has 79,434 styles. This is not a problem in the original fully-supervised setting where the authors used SIN for supervised image classification. In that case, the network can learn to ignore texture to discriminate between classes.

Table 2.3: Experiments with Sketch-ImageNet. Use of Anisotropic ImageNet shows that our method is better at capturing representation that are less dependent on texture.

Method	Top-1 Acc	Top-5 Acc
ImageNet Baseline	13.00	26.24
Stylized Baseline	16.36	31.56
Pix2Pix-Anisotropic (Ours)	24.49	41.81

2.4.2 Transfer Learning for Supervised Learning

As shown in the last section, suppressing texture leads to performance improvements in the case of domain transfer with SSL. In this section, we also show improvements in supervised learning and domain transfer. In the case of supervised learning we also show results using Pix2Pix-Anisotropic model.

2.4.2.1 Across Domains

We hypothesize that learning with texture bias is most harmful to domain transfer. Thus, we first describe a challenging experimental setup for learning visual representation across domains.

Sketch-ImageNet. For a cross-domain supervised learning setup, we chose to use the Sketch-ImageNet dataset [241]. Sketch-ImageNet contains sketches collected by making Google image queries “sketch of X”, where “X” is chosen from the standard class names of ImageNet. The sketches have very little to no texture, so performance on Sketch-ImageNet is a strong indicator of how well the model can perform when much less texture is present. As shown in Table 2.3, the difference between the Pix2Pix-Anisotropic model and the baseline model is 11.49% for Top-1

Table 2.4: Comparison using different texture removing methods, with different hyper-parameters for Anisotropic diffusion methods. We observe that the most simple [185] performs the best, and removing more texture from images does not improve performance.

Method	# Iterations	Top-1 Acc	Top-5 Acc	Object Detection
Baseline Supervised	-	76.13	92.98	70.7
Stylized ImageNet [76]	-	76.72	93.27	75.1
Perona Malik [185]	20	76.95	93.36	75.21
Perona Malik [185]	50	76.32	92.96	73.80
Robust AD [12]	20	76.58	92.96	73.33
Robust AD [12]	50	76.64	93.09	73.57
Gaussian Blur	-	76.21	92.64	73.26
Cartoon ImageNet	-	76.22	93.12	72.31
Bilateral ImageNet	-	75.99	92.90	71.34

accuracy. This result implies that our model captures less dependent representations on texture than standard ImageNet and Stylized ImageNet.

Other Datasets - Aircraft, Birds, Dogs, and Cars. We further evaluate our method on image classification tasks using four different fine-grained classification datasets. We also observe improvement on image classification across five datasets in Table 2.2. This shows that in the case of domain shift, capturing higher level semantics helps in better transfer learning performance.

Object Detection. The biggest improvement we observe on transfer learning is on object detection on Faster-RCNN [190] as shown in Table 2.11. This improvement suggests that we are able to attend to more high-level semantics, which helps in transfer learning performance on object detection.

2.4.2.2 Same Domain

ImageNet: In Table 2.11, we show results using Anisotropic ImageNet for supervised classification. We observe that Anisotropic ImageNet improves performance in both ImageNet classification and object detection. We also use the Pix2Pix model for learning to suppress texture. Pix2Pix model converts a normal image to texture suppressed image. For Gaussian blurring experiments, we closely follow [39] and add a Gaussian blur operator with variance from 10 to 20 and train in a similar manner to Stylized ImageNet [76]. We can see that our proposed Pix2Pix Anisotropic ImageNet performs better than both Stylized ImageNet and Gaussian blurring. Hence, blurring the image completely without respecting boundaries and edges, or distorting the shape information by style transfer does not improve performance.

Different Texture Removing Methods. We also provide results using different texture removing methods and different hyper-parameters for Anisotropic diffusion in Table 2.11. We observe that as we increase the number of iterations and remove more and more texture from images, performance starts to degrade, possibly due to the difference that comes in the data distribution after removing texture information. The most simple texture removing method [185] has the best results.

2.4.3 Analysis

We now show qualitatively and quantitatively how our model is less dependent on texture information.

2.4.3.1 Synthetic-DTD Dataset:

To better demonstrate the effectiveness of less texture dependent representations, we used the dataset introduced by [174]. This dataset provides four variations in images: texture, color, lighting, and viewpoint. This dataset is created by taking 47 different textures from DTD dataset [174] and applying them to a 3D dataset of 10 classes, called ShapeNet [30] to yield the same object rendered with different viewpoints and multiple textures. It contains 480,000 training images and 72,000 testing images. In this dataset, we made sure that texture information during training and testing are completely different. So, the texture is not a cue when we use this dataset. We evaluate our Pix2Pix-Anisotropic model on this dataset and compare against the baseline normal ImageNet model. The Pix2Pix-Anisotropic model achieves a performance boost of 10.41% in classification which suggests that we are indeed able to learn texture agnostic feature representations.

2.4.3.2 Visual Analysis:

We now visually analyze the results by the saliency maps, which are produced by different networks. We use GradCam [199] to calculate the saliency maps. In Fig 2.9, we show the saliency maps produced by networks trained using the combined dataset and the original ImageNet dataset. We observe that Pix2Pix-Anisotropic model has saliency maps that spread out over a bigger area and that include the outlines of the objects. This suggests that it attends less to texture and more to overall holistic shape. In contrast, ImageNet trained models have narrower saliency

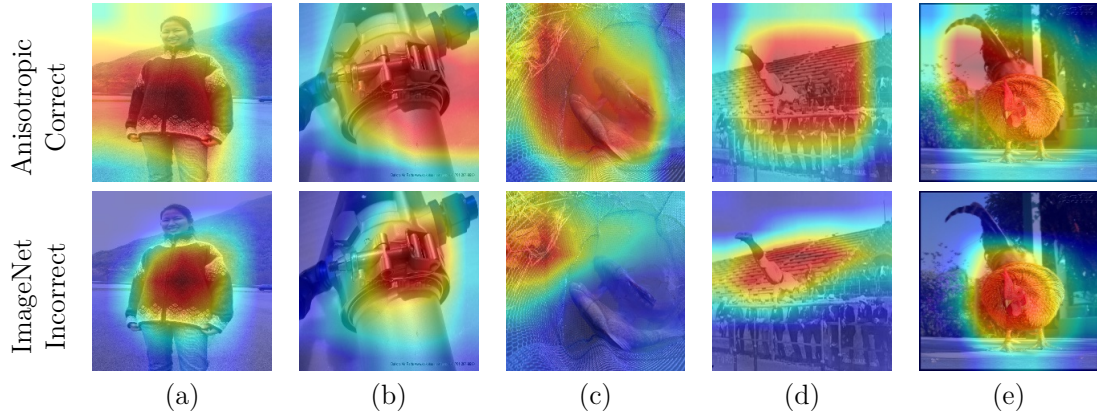


Figure 2.4: Saliency maps using GradCam. The text on the left of the row indicates whether the Anisotropic model or ImageNet model was used. The figure shows the saliency maps where the Anisotropic model gave correct predictions, and the ImageNet model gave wrong predictions. The failure of the ImageNet model might be due to it not attending to the whole object.

maps that miss the overall shape and focus on localized regions. In Fig. 2.9(a-e), we present the examples where the Pix2Pix-Anisotropic model gives the correct prediction, and the ImageNet model fails. For example in Fig. 2.9(e), we observe that the network trained on ImageNet alone is not focusing on the whole bird and is only focusing on the body to make the decision; whereas the one trained with Pix2Pix-Anisotropic ImageNet is focusing on complete bird to make a decision.

2.4.3.3 Experiments on testing shape bias:

To show the low reliance of our model on texture and greater reliance on high-level features like shape, we use the Geirhos Style-Transfer (GST) dataset [76]. It consists of 1248 images of 16 classes from ImageNet with shape and texture coming from different classes. This dataset teases apart the importance of shape vs texture for CNNs. We observe an improvement of 1.02% on identifying the class representing



Figure 2.5: Examples of images from Sketch-ImageNet. Images have very little or no texture, which implies texture will have little to no impact on object classification.

the shape over a ImageNet pretrained Resnet-50. This shows that we can successfully reduce the reliance on texture.

2.4.3.4 Results after removing high-frequency components:

CNNs tend to exploit the high-frequency components in an image which makes them less robust [114,242]. In this section, we show that our proposed Pix2Pix-Anisotropic model is less dependent on high-frequency components. To remove high-frequency components from an image, we first obtain the image spectrum using DFT. We then incrementally vary the masked region’s size to retain the more/less high frequency components with r denoting the mask region’s side. We can see from Figure 2.8 as we remove the highest frequency components, our model performs better than the baseline by +2.1%. The same difference in normal RGB images is 0.8%. Hence this shows that our model has less reliance on the highest frequency components. As we remove more high-frequency components, the error increases for both models, but we are still consistently better than the baseline.

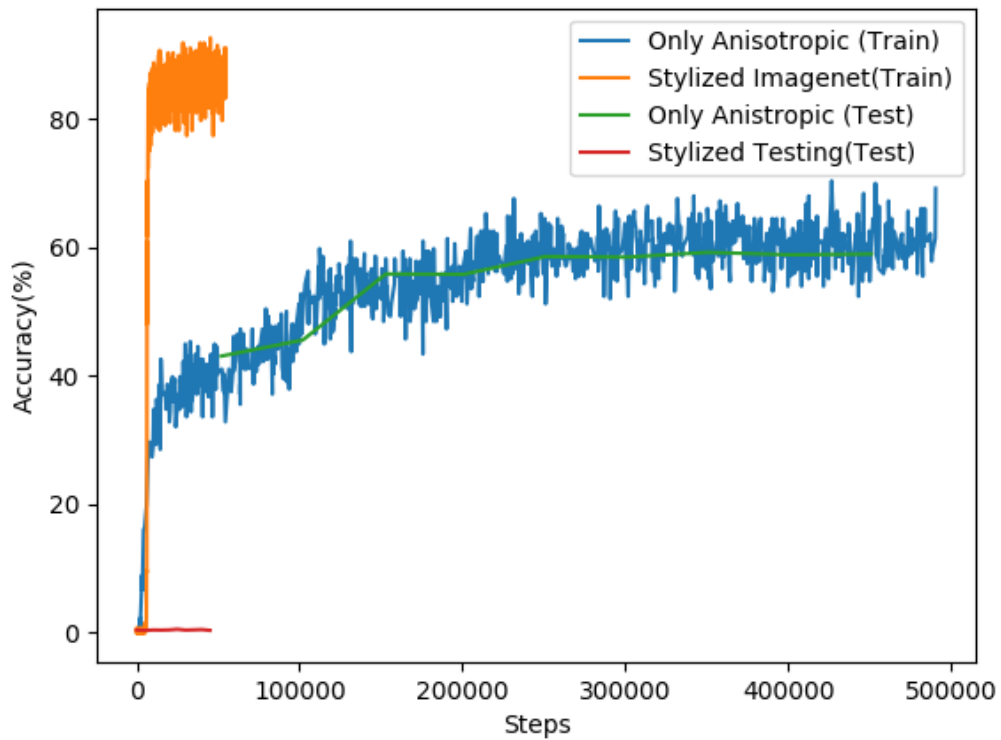


Figure 2.6: Training plots for the both Stylized ImageNet and Standard ImageNet. The plot shows that the model trained on Stylized ImageNet quickly overfits by finding shortcuts after around 6000 steps. Therefore, it gives poor performance on downstream tasks by relying on texture based shortcuts.

Table 2.6: We show additional experiments on dataset ImageNet-C [100], which evaluates model robustness to common corruptions. We can see that by focussing less on texture our model is consistently more robust than the baseline Resnet model pre-trained on ImageNet data.

Network	mCE	Noise				Blur				Weather				Digital		
		Gauss.	Shot	Impulse	Defocus	Glass	Motion	Zoom	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	JPEG
Baseline(ImageNet)	76.7	80	82	83	75	89	78	80	78	75	66	57	71	85	77	77
Anisotropic(Ours)	74.85	72.70	76.65	78.27	73.73	87.97	75.99	79.70	80.33	78.02	68.05	58.30	71.34	82.64	70.88	68

2.4.3.5 Confidence of Models:

In this section we compare the confidence and entropy of Anisotropic Model and ImageNet model when both the models have given correct predictions. To find confidence, we generate the probability scores of correct class. After this we calculated the mean of correct probability scores on both the models. As we can see from Table 2.12 that Anisotropic ImageNet has larger mean which means that Anisotropic ImageNet has better confidence as compared to Standard ImageNet. We also calculate the entropy of output probability distribution from both the models. We can see from Table 2.12 Anisotropic ImageNet has lower entropy scores as compared to Standard ImageNet.

2.4.3.6 Training with Stylized ImageNet

Figure 2.6 shows the training curves for both Stylized ImageNet and Standard ImageNet. We see that the model quickly saturates when using Stylized ImageNet. This leads to a low performance on downstream tasks. Our hypothesis is that the model rapidly learns to exploit some regularities in the texture introduced by the GANs to easily solve the MoCoV2 and Jigsaw tasks. This means that the self-supervised

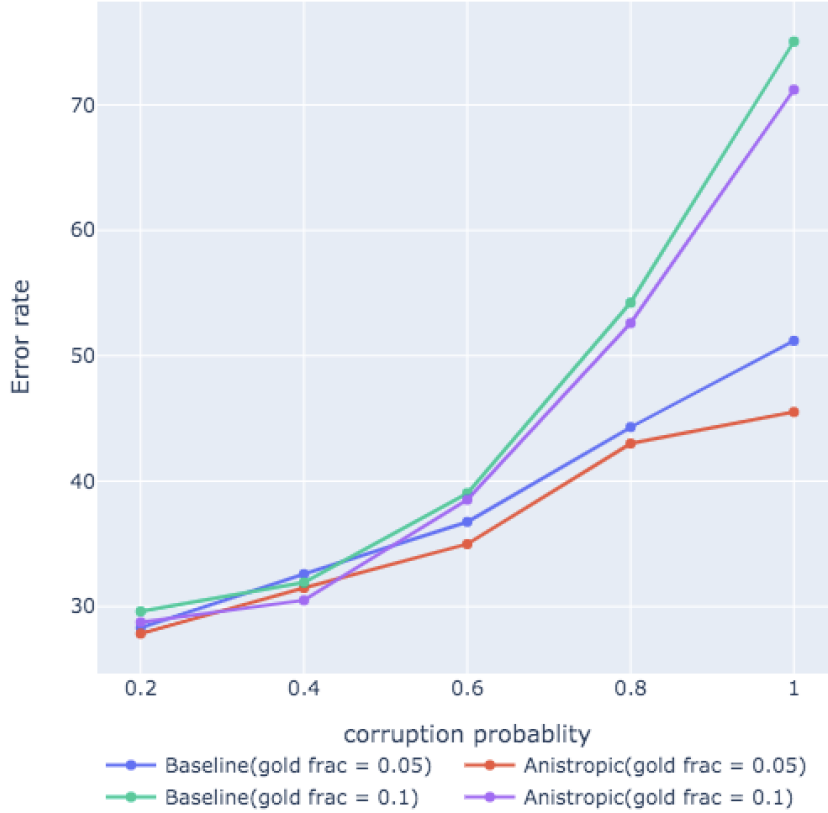


Figure 2.7: Results on Label corruption task. The bottom two line corresponds to when the gold fraction is 0.05, and the top two lines correspond to the case when the gold fraction is 0.1. We can see that our model consistently outperforms the baseline with a larger improvement upon increasing the corruption probability.

model has the tendency to take shortcuts in the presence of regular textures. The aim of this chapter has been to investigate such short-comings and provide appropriate solutions to the issues.

2.4.3.7 Results on model robustness

In this section, we show that suppressing texture not only helps in achieving better results but also makes our model more robust to common image corruptions.

Label Corruption Task: We show results on the label corruption task in

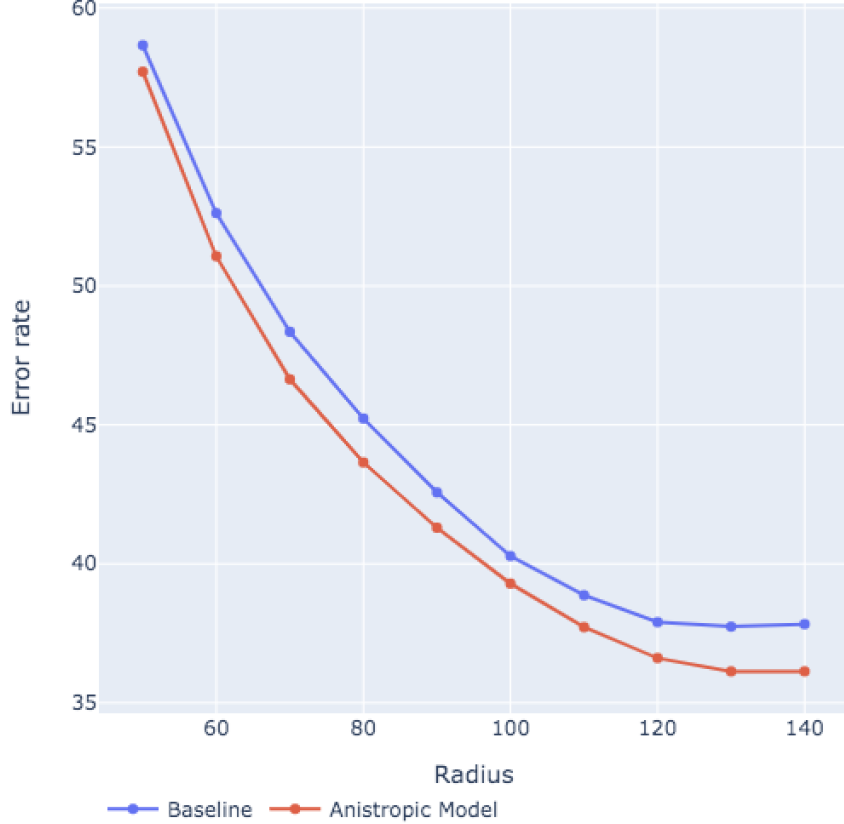


Figure 2.8: Results after removing high-frequency components from images. We can see that our proposed anisotropic filtering leads to models that are less reliant on high-frequency components in an image. When we remove the highest frequency components from the image(far-right), our model, outperforms baseline by close to 2.1%. As we remove more high-frequency components, both the model suffers, but our model consistently performs better.

Fig 2.7. In this task, [101] only some fraction(gold fraction) of data is clean, and the label of the rest of the data is corrupted with some probability. The task is to use both of these datasets for supervised classification. We use CIFAR100 as our dataset, and we augment the CIFAR100 dataset with Anisotropic diffused images. We create a dataset double the size of the original dataset and use it for the task of Label Corruption [101]. We can see from the results that we have consistent improvement over the baseline. With an increase in corruption probability, our results improve, even more, implying that focusing on higher-level features helps

even more in extreme corruptions.

Results on ImageNet-C: We show additional experiments on dataset ImageNet-C [100], which evaluates model robustness to common corruptions. ImageNet-C[1] dataset has a total of 15 corruptions, as shown in Table 2.6. We improve(+1.85%) upon the baseline Resnet model pre-trained on ImageNet data and perform especially well in the cases of noise corruptions since we train on anisotropic images, which removes high-frequency noise from images.

2.5 Jigsaw Task

In addition to MoCoV2 we also show results with Jigsaw pretext task.

2.5.1 Method

The goal in Jigsaw is to infer correct ordering of the given regions from an image. Following [177], the typical setting is to divide an image into nine non-overlapping square patches and randomly shuffle them. A CNN is trained to predict the original permutation of these patches in the image. Jigsaw++ [179] extended this idea and replaced some patches with random patches. These patch based methods come with their own issues and there has been some recent effort to solve them. [173] describe an easy short-cut CNNs take that utilizes the chromatic aberration produced due to different wavelengths exiting the lens at different angles. The authors provided a solution to the problem of chromatic aberration by removing cues from the images and also allowing the color pattern to be partially preserved. They also address the

problem of true spatial extent that network sees in patch based methods by yoking the patch jitter to create a random crop effect.

2.5.2 Results using Jigsaw Pretraining

The baseline model in the case of Jigsaw is pre-trained using the standard ImageNet dataset. Unlike [177], we use ResNet18 as our backbone instead of Alexnet [136] to take advantage of deeper layers and capture better image representations. We obtained these results by building on top of a publicly available implementation². Table 2.7 shows results for Jigsaw models trained and tested on different datasets. We observe that the Jigsaw model trained on the Cartoon dataset outperforms the baseline methods by 2.52 mAP and Anisotropic ImageNet outperforms the baseline methods by 1.8 mAP on the PASCAL VOC image classification dataset. On object detection Bilateral ImageNet outperforms the baseline Jigsaw model by 0.78 mAP. On semantic segmentation Anisotropic Imagenet outperforms the baseline Jigsaw models by 8.1 mAP. Traditionally semantic segmentation has been a difficult task for Self-Supervised methods [21, 177] and improvement of this order on semantic segmentation shows the effectiveness of removing texture.

We also show results on ImageNet classification as the downstream task. Due to its large scale, it is usually infeasible to fine-tune the whole network for the final task every time. Therefore, following prior work [21], we only fine-tune a linear classifier. The inputs to this classifier are the features from a convolution layer in the network. Note that while fine-tuning for the final task, we keep the backbone

²<https://github.com/bbrattoli/JigsawPuzzlePytorch>

frozen. Therefore, the performance of the linear classifier can be seen as a direct representation of the quality of the features obtained from the CNN. We report the results of this experiment on ImageNet in Table 2.8. Adding the Anisotropic ImageNet dataset to this model gives a further improvement of 0.5%.

2.5.3 Jigsaw using Alexnet as backbone

Our improvement when using Anisotropic ImageNet is not restricted to the backbone. Traditionally in Self-Supervised learning one of the most followed architectures is Alexnet [21, 62, 177]. Following these methods, we also show results on Alexnet backbone. In Table. 2.9 we show results on VOC Classification when using Alexnet as the backbone. We obtain an improvement of 0.67 mAP over the baseline.

2.5.4 Patch-wise Anisotropic Diffusion

In our best performing model, we considered all the patches for the jigsaw task to either come from the standard ImageNet or anisotropic diffusion filtered ImageNet. What if each of the 9 patches for the Jigsaw task could be either a standard patch or filtered patch? For this experiment we randomly choose a patch from the standard dataset or the filtered dataset, with equal probability. This is a much more extreme form of data augmentation and considerably increases the difficulty of the task. We got an improvement of 0.6 mAP over the baseline model for the classification task. However this is 1.1 mAP lower than the doing Anisotropic Diffusion on whole image.

Table 2.7: Comparison of our approach with Jigsaw baseline methods. Using our best model, we improve 2.52 mAP in VOC classification , 0.78 mAP on VOC detection and 8.1 mAP on VOC semantic segmentation(SS) over the baseline models. Note that Stylized ImageNet performs poorly on VOC classification due to the visual shortcuts.

Method	Dataset Size	VOC Cls.	VOC Det.	SS
Baseline	1.2M	74.82	61.98	27.1
Stylized [76]	1.2M	13.81	28.13	10.12
Gaussian ImageNet	2×1.2M	75.49	62.39	27.9
Bilateral ImageNet	2×1.2M	74.55	62.74	28.9
Only Anisotropic	1.2M	74.52	61.85	32.7
Anisotropic ImageNet	2×1.2M	76.77	61.59	35.2
Cartoon ImageNet	2×1.2M	77.34	59.31	34.1

Table 2.8: ImageNet classification by finetuning the last FC layer. Features from the conv layers are kept unchanged. This experiment helps evaluate the quality of features learnt by the convolutional layers.

Method	Dataset Size	VOC Cls	VOC Det.	ImageNet Cls.	Acc
Jigsaw Baseline	1.2M	74.82	61.98	26.17	
Jigsaw anisotropic	2×1.2M	76.77	61.59	26.67	

2.6 Anistropic Images

We show some more examples of Anistropic images obtained by applying Anisotropic diffusion filters to images from ImageNet [56] in figures 2.11 and 2.12. Notice how the images lose texture information. This makes it more difficult for models to find

Table 2.9: Experiments with Alexnet as the backbone. Ideas of anisotropic diffusion filter can extend to other architectures like Alexnet. The Anistropic ImageNet model improves over the baseline by 0.67 mAP

Method	VOC 2007 Classification
Jigsaw Baseline(Our Implementation)	65.21
Jigsaw anisotropic	65.88

shortcuts. This, in turn, leads to better semantic representations learned by the model which leads to higher performance on downstream tasks.

2.7 Saliency Maps

2.7.1 Sketch-ImageNet Saliency Maps

In Fig 2.13 we show some of saliency maps for Sketch-ImageNet images. We can see from saliency maps that Anistropic ImageNet has broader saliency map and has better coverage of the object as compared to ImageNet model.

2.7.2 Saliency Maps for Anistropic ImageNet and Standard ImageNet models

We also show some addtional saliency maps in Figure 2.14, Figure 2.16, Figure 2.15 and Figure 2.17 corresponding to both the models. We can see from the figures that Anistropic ImageNet has in general diffused saliency maps.

2.8 Implementation details

Training Details. With image classification as the downstream task, we train our network for 90,000 iterations with an initial learning rate of 0.003 following [21].

For object detection we report our results for Faster-RCNN [190] using our pre-trained model as backbone. We tune hyper-parameters using the validation set.

For object detection, we follow the details of [190] to train a model; 10 epochs with

Table 2.10: Results on Label corruption task. We can see that our model consistently outperforms the baseline with larger improvement upon increasing the corruption probability.

Corruption probability	Gold Fraction	Baseline(Error)	Anistropic Model(Error)
0.2	0.05	29.61	28.75
0.2	0.1	28.31	27.84
0.4	0.05	31.92	30.5
0.4	0.1	32.59	31.48
0.6	0.05	39.04	38.52
0.6	0.1	36.75	34.98
0.8	0.05	54.23	52.6
0.8	0.1	44.31	43
1.0	0.05	75.05	71.21
1.0	0.1	51.19	45.51

an initial learning rate of 0.001.

For semantic segmentation, we report our results on FCN [204] using our pretrained model as backbone. We train the FCN model for 30 epochs using an initial learning rate of 0.01.

ImageNet. We use ImageNet for all training and evaluation of image classification accuracy. For self-supervised learning, we follow [21, 91]; we train linear classifiers using features obtained from the final Residual block by freezing all convolutional layers. The performance of these linear classifiers is meant to evaluate the quality of the feature representations learnt by the convolutional layers, since the backbone is completely frozen and only fully-connected layers are being trained. We chose hyper-parameters using the validation set and report performance on the ImageNet validation set.

Note that since we use ImageNet to pre-train for self-supervised learning, there is no domain difference when we conduct inference on ImageNet, but with VOC there

is. With the VOC results, we validate that the gain by our method is particularly large when there is domain shift.

MoCo-v2 task. For MoCo-v2 we use higher lr of 0.3 as it gives better performance for our task.

Jigsaw task. In Jigsaw [177] the image is divided into 9 non-overlapping square patches. We select 1,000 from the $9!$ possible permutations. All of our primary experiments on Jigsaw use ResNet18 as the backbone [94]. We train the Jigsaw task for 90 epochs, with an initial learning rate of 0.01. The learning rate is reduced by a factor of 0.1 after (30, 30, 20, 10) epochs. We use the same data augmentation as in [177]. In MoCo [94], we use ResNet50 [94] as the backbone, following the same procedure as mentioned in [91].

PASCAL VOC. Following [21] and [39], we evaluate image classification and object detection on the PASCAL VOC dataset [67]. It contains about 5,000 images in the train-val set belonging to 20 classes. Note that the image classification task is multi-label. Therefore, the metric used for evaluating both image classification and object detection is the mean Average Precision (mAP).

Training Details for Object detection for COCO based metrics: We report object detection results on [190] C4 backbone which is finetuned end to end on VOC07+12 trainval dataset and evaluated on the VOC 07 test set using the COCO suite of metrics.

Other Details. We use 4 Nvidia GTX 1080 Ti for all experiments. Pretraining on Jigsaw takes 3 days on the standard ImageNet dataset. The SGD optimizer with

Table 2.11: Comparison between Stylized ImageNet and our Anisotropic ImageNet. Following [76], we use ResNet50 as our backbone. We finetune our models on only the ImageNet dataset. We can see that on ImageNet classification and object detection, Anisotropic ImageNet and Stylized ImageNet have very similar performance.

Method	Finetune	Top-1 Accuracy	Top-5 Accuracy	OBJ Detection
Stylized Imagenet	-	74.59	92.14	70.6
Stylized Imagenet	IN	76.72	93.28	75.1
Anisotropic Imagenet	-	68.38	87.19	-
Anisotropic Imagenet	IN	76.71	93.26	74.27
Cartoon Imagenet	IN	76.22	93.12	72.31

Table 2.12: Experiments discussing the confidence and entropy of Anisotropic ImageNet and Standard ImageNet

Method	Entropy	Mean Highest probability
Anisotropic ImageNet	0.81	0.93
Standard ImageNet	1.88	0.59

momentum was used for all experiments with momentum of 0.9 and weight decay of 5×10^{-4} . Cross-entropy loss was used for all experiments, mini-batch size was set to 256. Pretraining on MoCoV2 takes 6 days on 4 Nvidia P100 machines. We set all other hyperparameters following [39].

2.9 Label Corruption Task

We show the full results of the label corruption task in Table 2.10.

2.10 Saliency Maps

In Fig. 2.9 we show the saliency maps produced by networks trained using the combined dataset and the original ImageNet dataset. We use GradCam [199] to

calculate the saliency maps. We can see that Anisotropic ImageNet has saliency maps that spread out over a bigger area and that include the outlines of the objects. This suggests that it attends less to texture and more to overall holistic shape. In contrast, ImageNet trained models have narrower saliency maps that miss the overall shape and focus on localized regions, suggesting an attention to texture. In Fig. 2.9(f-j) we show these for the case where the Anisotropic model gives the correct prediction and the ImageNet model fails. For example in Fig. 2.9(j), we see that the network trained on ImageNet alone is not focusing on the whole bird and is only focusing on the body to make the decision whereas the one trained with Anisotropic ImageNet is focusing on complete bird to make a decision. We see a similar trend in the cases where both the models give the correct prediction (Fig. 2.9(a-e)). In the case where Anisotropic model makes incorrect predictions and ImageNet model (Fig. 2.9(k-o)) is correct we see the saliency maps are still diffused, but we fail to capture the whole object leading to incorrect predictions.

2.11 Conclusion

We propose to help a CNN focus on high-level cues instead of relying on texture by augmenting the ImageNet dataset with images filtered with Anisotropic diffusion, in which texture information is suppressed. Empirical results suggest that using the proposed data augmentation for pretraining self-supervised models and for training supervised models gives improvements across ten diverse datasets. Noticeably, the 11.4% improvement while testing the supervised model on Sketch ImageNet suggests

that the network is capturing more higher-level representations than the models trained on ImageNet alone.

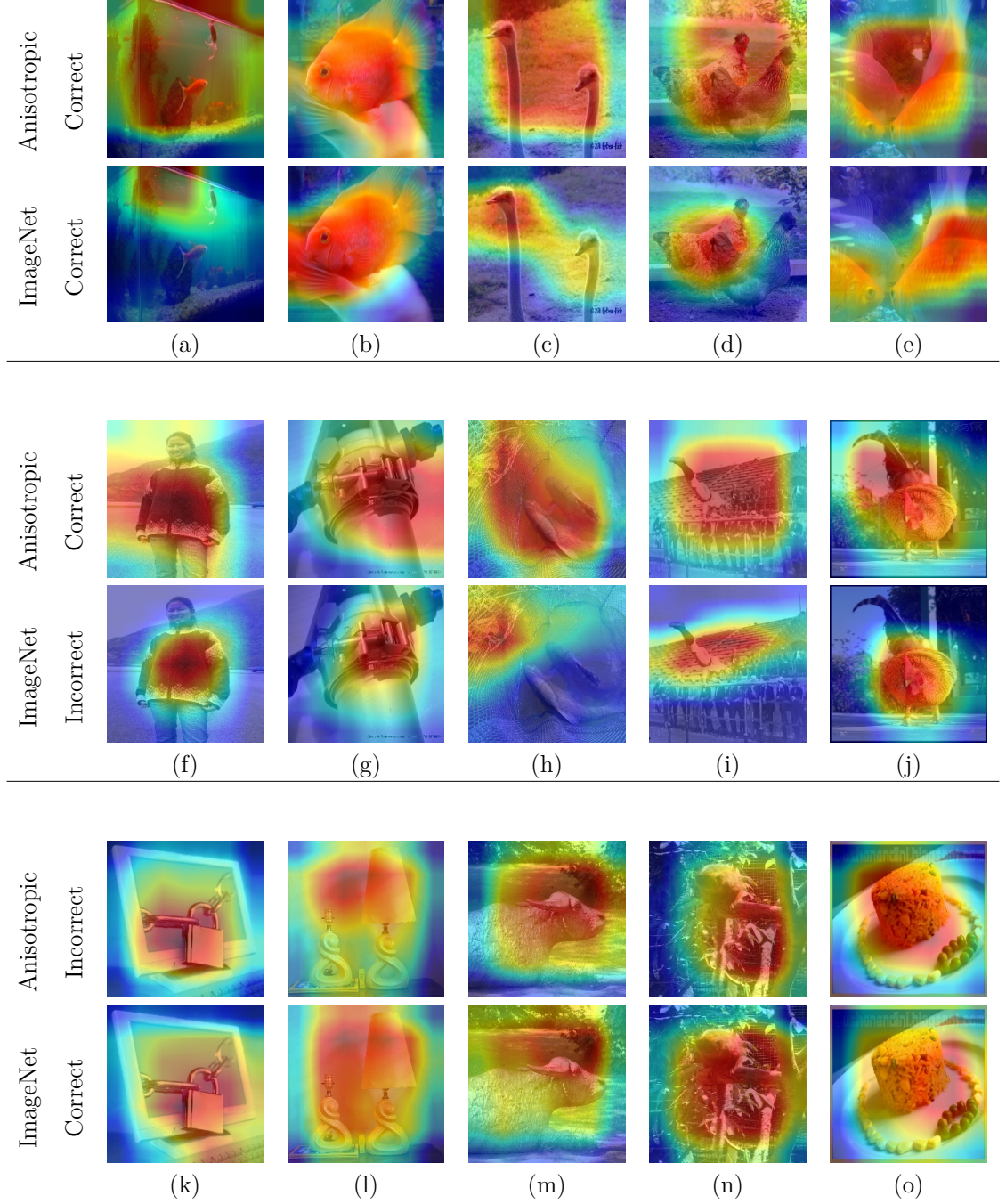


Figure 2.9: Saliency maps on three different set of images. The text on the left of the row indicates whether Anisotropic model or ImageNet model was used. The first two rows show the saliency maps where both model gave correct predictions. We can see from saliency maps that the Anisotropic model has more diffused saliency maps. The second two rows show the saliency maps where Anisotropic model gave correct predictions and ImageNet model gave wrong predictions. The failure of ImageNet model might be due to it not attending to whole object. The last two rows show the saliency maps where Anisotropic model gives incorrect predictions and ImageNet model gives correct predictions. Even in this failure mode, the Anisotropic model gives diffused saliency maps.

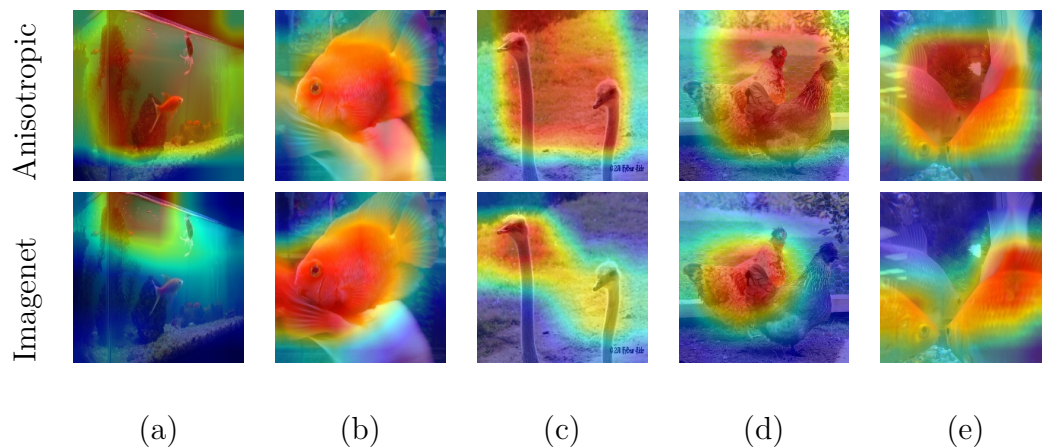


Figure 2.10: Saliency maps when our technique gives the correct prediction and baseline approach gives incorrect label. The top row gives the saliency maps for our model and the bottom one shows the corresponding saliency maps for the model trained on imagenet alone. We can see from saliency maps that Anisotropic model has bigger saliency maps which might be the reason for the correct prediction.

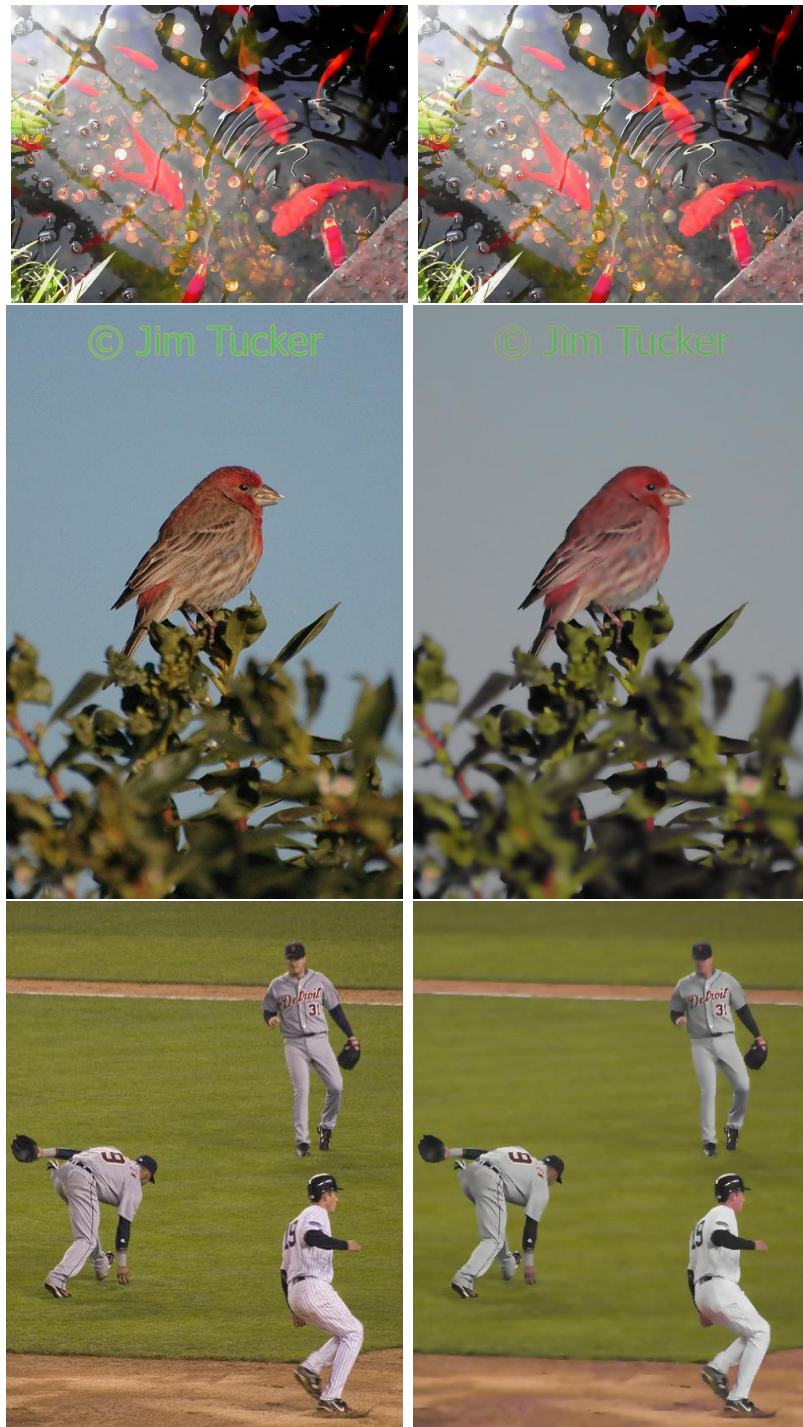


Figure 2.11: Original images (left) and images obtained after anisotropic diffusion (right). Most of the texture information in the images has been smoothed out by the filter while retaining the shape information. This forces the network to capture higher-level semantics without relying on low-level texture cues

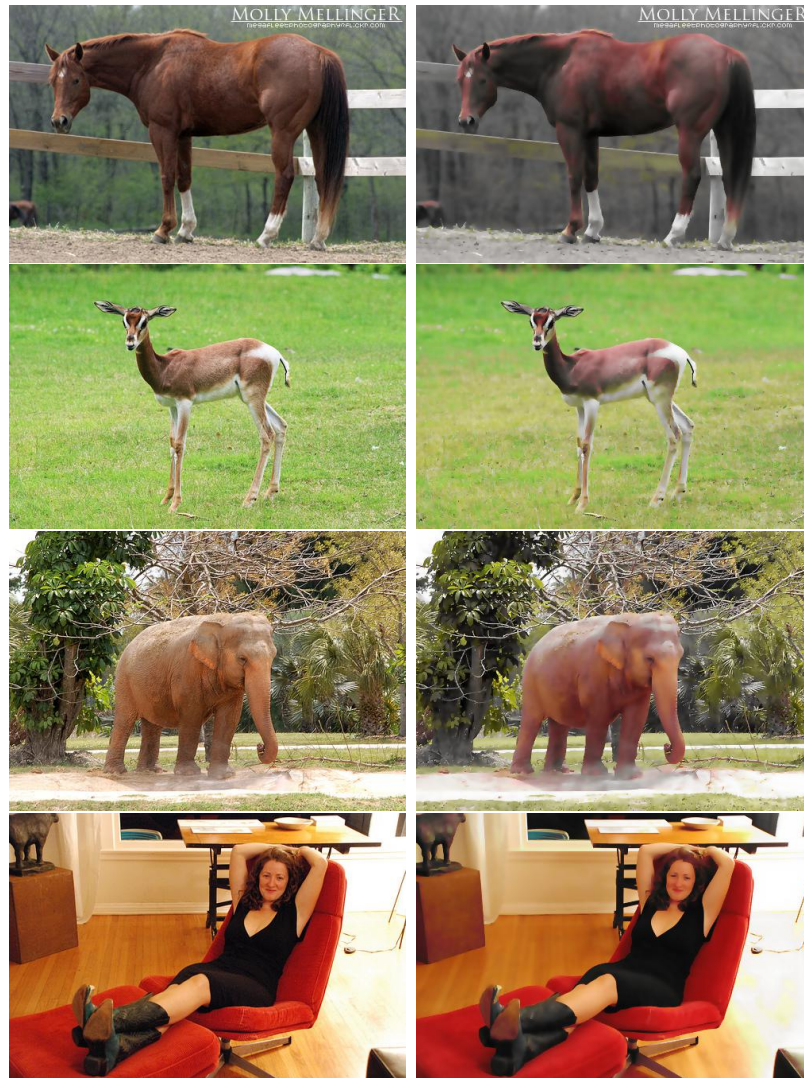


Figure 2.12: Original images (left) and images obtained after anisotropic diffusion (right). Most of the texture information in the images has been smoothed out by the filter while retaining the shape information. This forces the network to capture higher-level semantics without relying on low-level texture cues

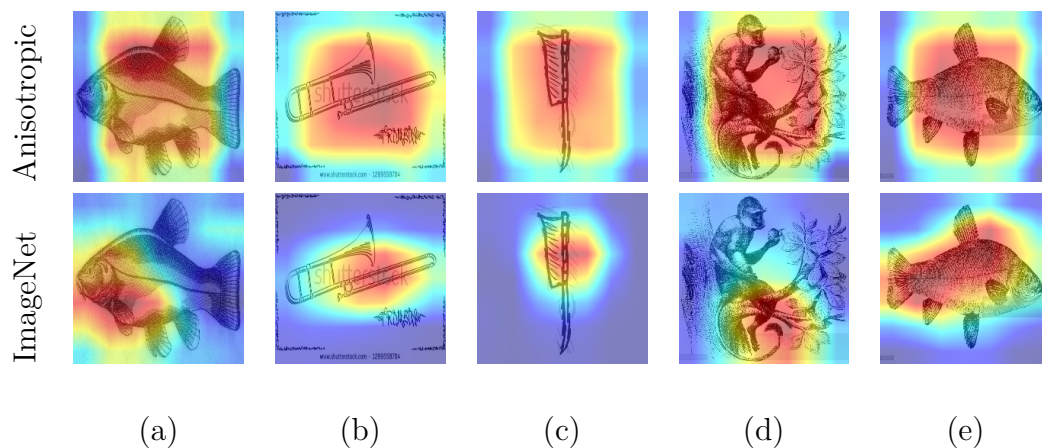


Figure 2.13: Saliency maps on few randomly selected images from Sketch-ImageNet. We can see from saliency maps that Anisotropic model has bigger saliency maps which might be the reason for the correct prediction.

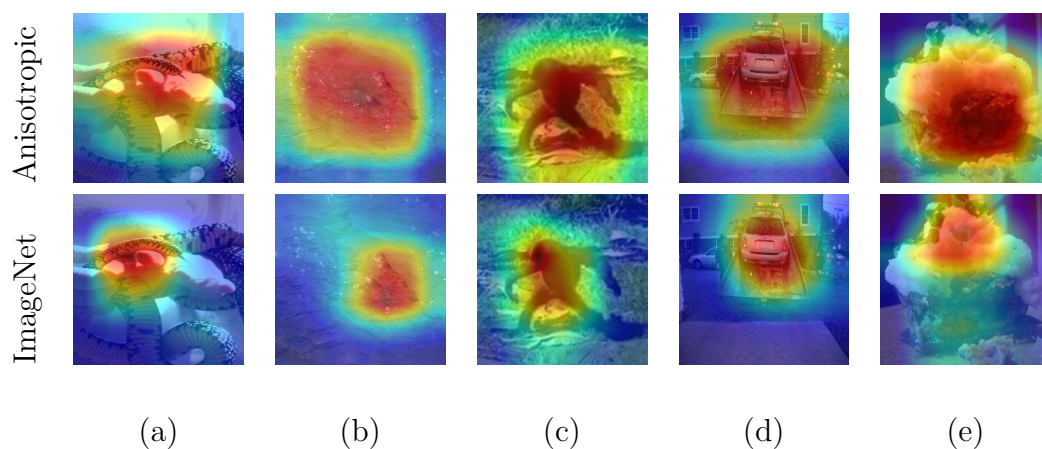


Figure 2.14: Saliency maps when Anisotropic Model had correct predictions and ImageNet model has wrong predictions.

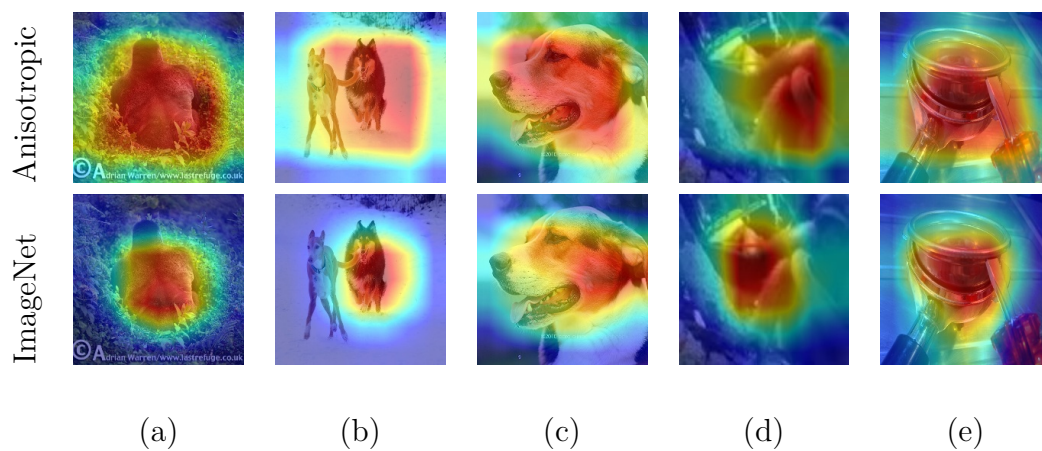


Figure 2.15: Saliency maps when both model have wrong predictions.

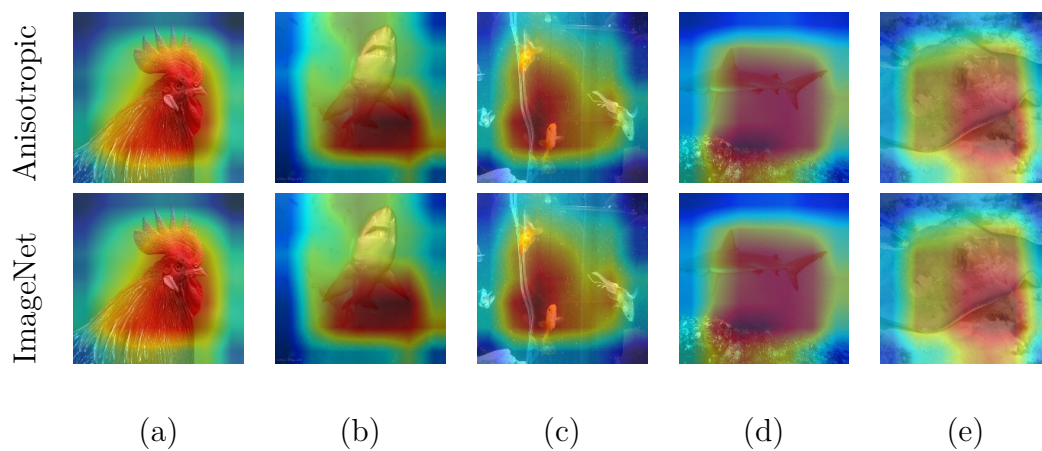


Figure 2.16: Saliency maps when both model have correct predictions.

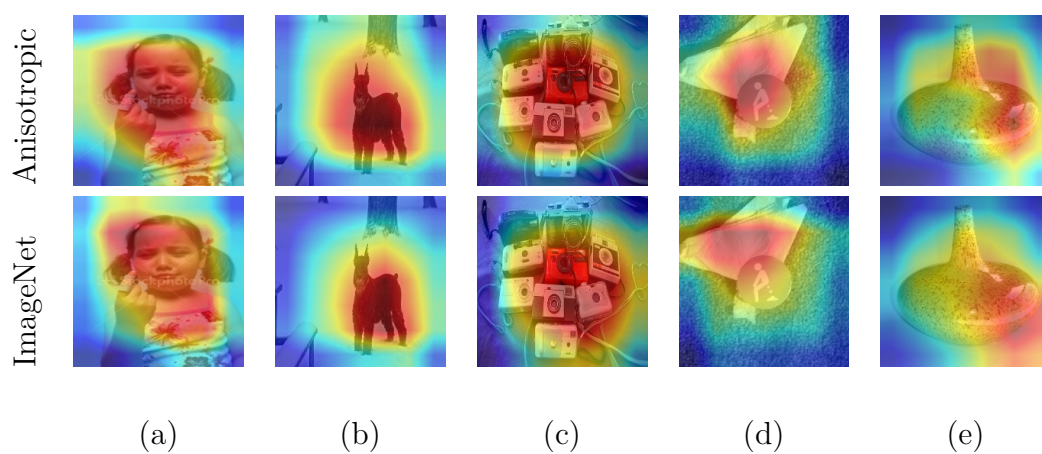


Figure 2.17: Saliency maps when ImageNet model has correct predictions and Anisotropic model has wrong predictions.

Chapter 3: Robust Contrastive Learning Using Negative Samples with Diminished Semantics

3.1 Introduction

Recent studies on self-supervised learning have shown great success in learning visual representations without human annotations. The gap between unsupervised and supervised learning has been progressively closed by contrastive learning [23, 35, 85, 92, 219, 222, 250, 266]. As we discussed in Chapter 2, CNNs trained in the supervised setting are known to learn correlations between labels and superfluous features such as local patches [16, 17], texture [77], high-frequency components [242], and even artificially added features [110], which has raised concerns about deploying these models in a real scenario [78, 120]. CNNs trained by contrastive learning methods are no exception [103]. In this chapter, we propose to construct negative samples that only preserve non-semantic features. We show that using contrastive learning methods trained with these negative samples can mitigate these concerns. ¹

Contrastive learning methods exploit carefully designed augmentations to construct positive pairs and adopt pretext tasks to pull together their representations.

¹The contents of this work is in collaboration with Songwei Ge, Haohan Wang, Chun-Liang Li, and David Jacobs [74].

These augmentations are crucial to the effectiveness of contrastive learning [23, 35]. A common assumption behind these augmentations is to preserve the semantic information of the input images while perturbing other superficial signals. This inspires us to generate negative samples and inject additional implicit biases on the visual features learned by the models. Specifically, we utilize augmentations that diminish the semantic features while keeping the undesired features such as texture. By pushing away the representations of such negative samples and input images, the models are expected to rely more on the semantic portion of the images and less on superficial features.

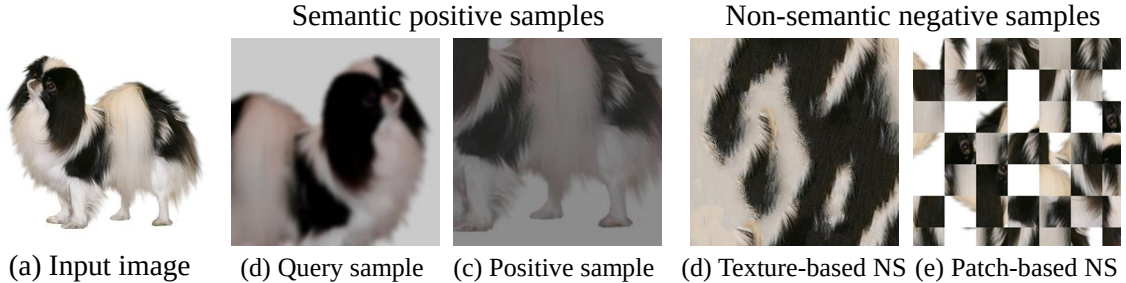


Figure 3.1: We propose to construct negative samples (NS) from input images for contrastive learning with augmentations that only preserve non-semantic information such as texture and local features.

Inspired by the non-semantic features, we propose two methods to craft negative samples. The first method relies on texture synthesis tools from classic computer vision [66, 247]. It generates realistic texture images based on two patches extracted from input images, as shown in Figure 5.1d. For each image in ImageNet, we generate its texture version and form a dataset which we call ImageNet-Texture. The second method constructs non-semantic images by tiling randomly sampled patches of different sizes from the input image, as shown in Figure 5.1e. Comparing the non-

semantic negative samples with two semantic positive samples in Figure 5.1b and Figure 5.1c, the dog from the input image is still recognizable in the positive samples but hard to understand from negative samples. Instead, some local statistics such as the fur and color of the dog are preserved in the negative samples.

The generated non-semantic samples can be readily used with existing contrastive learning methods that distinguish positive pairs from negative pairs such as MoCo [92] and SimCLR [35]. Despite their simplicity, we show that these non-semantic negative samples are generally harder than the standard negative samples used by these methods, which are inefficient at leveraging hard negatives [70, 122]. Further, our negative pairs can also be used for the contrastive learning methods that do not explicitly use negative samples, such as BYOL [85]. We evaluate our methods with two contrastive learning methods, MoCo [39, 92] and BYOL [85], on three datasets, ImageNet-100, ImageNet-1K and STL-10. When using our proposed augmentations to generate negative samples and minimizing their representation similarity to the input images, we notice a consistent improvement on the generalization performance over backbone methods [39, 85] and previous negative example generation strategies [122, 193], especially under out-of-distribution (OOD) settings.

With our proposed method and the ImageNet-Texture dataset, we conduct a systematic analysis of how the shape-texture trade-off influences model performance. We introduce a hyperparameter that controls the penalty strength on the similarity between the non-semantic negative examples and the query samples, and consequently impacts the trade-off of the dependence on the shape versus texture features. We find that the relative importance of texture and shape features varies

across different test datasets. For example, a large shape bias benefits more on the ImageNet-Sketch [240] dataset than the normal ImageNet validation set, while, as we will show later, a large texture bias benefits more for finer-grained classification such as classifying different dog breeds included in the ImageNet dataset, which complements to previous evidence showing the effectiveness of shape features in classifying 16 coarse classes [77]. Furthermore, such preference for one feature over the other is also observed intra-dataset: the texture is more important for some classes such as dishrag and plaque. These observations make us rethink the relationship between shape and texture features as the implicitly necessary bias of CNNs and advocate for an adaptive combination of both when deploying the model in real scenarios.

Our contributions are summarized below:

- We propose two methods, texture-based and patch-based augmentations, to generate negative samples from input images themselves, and show that with these negative samples contrastive learning can achieve better generalization.
- We introduce the ImageNet-Texture dataset, which contains texture versions of ImageNet images generated by texture synthesis tools.
- We provide fine-grained analysis on the shape-texture trade-off of CNNs based on our methods and the ImageNet-Texture dataset, and show different scenarios when one is preferred over the other.

3.2 Negative Samples with Diminished Semantics

CNNs are apt to learn low level features such as texture in a supervised setting [16, 77, 242]; this has been recently witnessed in the contrastive learning setting as well [103]. In this section, to mitigate this problem, we propose two methods, texture-based and patch-based augmentations, to generate negative samples for contrastive learning. Texture-based augmentation generates realistic images based on texture synthesis and patch-based augmentation exploit more comprehensive local features by sampling patches from input images. By penalizing a learned similarity between the representations of images and their non-semantic versions, the model is encouraged to place less value on the undesired features and focus more on the semantic features. In practice, we find the two negative samples play similar roles and the patch-based method works slightly better. In this section, we start by recapping the general frameworks for contrastive learning and show how non-semantic negatives are used in these frameworks. Then we elaborate on the two approaches to generate negative samples with diminished semantics.

3.2.1 Contrastive learning with non-semantic negatives

Given an encoder network f and image $\mathbf{x}$, we denote the output of the network as $\mathbf{z} = f(\mathbf{x})$. We use $\mathbf{z}_i$ and $\mathbf{z}_p$ to denote the representations of the query sample $\mathbf{x}_i$ and a positive sample $\mathbf{x}_p$ generated from the same input image with augmentations that preserve semantics. For contrastive learning methods like MoCo [92] and SimCLR [35], $\mathbf{z}_n$ denotes the representation of the standard negative sample

$\mathbf{x}_n$ extracted from the memory bank (MoCo) or other images in the current batch (SimCLR). $\mathbf{z}_{ns}$ is the representation of the proposed negative sample $\mathbf{x}_{ns}$ which contains particular non-semantic features of the input image with the semantic part weakened. We extend the noise-contrastive estimation (NCE) loss used by MoCo and SimCLR as below:

$$\mathcal{L}_{\text{NCE}} = - \sum_{i \in I} \log \frac{\exp(\mathbf{z}_i^T \mathbf{z}_p / \tau)}{\exp(\mathbf{z}_i^T \mathbf{z}_p / \tau) + \exp(\alpha \mathbf{z}_i^T \mathbf{z}_{ns} / \tau) + \sum_{n \in \mathcal{N}} \exp(\mathbf{z}_i^T \mathbf{z}_n / \tau)}, \quad (3.1)$$

where τ is a temperature parameter and α is an additional scaling parameter for non-semantic negatives. A larger α implies a stronger penalty on the similarity between the representations of query image and its non-semantic version. In section 3.7.1 we discuss other possible ways to apply α .

Methods like BYOL [85] do not explicitly rely on negative samples. Nevertheless, it adapts the loss to maximize the agreement of positive pairs. Therefore, we explicitly use the non-semantic negative sample with their loss to minimize its similarity to the query sample:

$$\mathcal{L}_{\text{BYOL}} = \|\mathbf{z}_i - \mathbf{z}_p\| - \alpha \|\mathbf{z}_i - \mathbf{z}_{ns}\| = 2 - 2\alpha - 2\mathbf{z}_i^T \mathbf{z}_p + 2\alpha \mathbf{z}_i^T \mathbf{z}_{ns}. \quad (3.2)$$

We overload α to be the parameter that controls the penalty on the similarity between the representations of input image and its non-semantic version under BYOL, with similar intention as MoCo and SimCLR above. To minimize either $\mathcal{L}_{\text{NCE}}$ or $\mathcal{L}_{\text{BYOL}}$, the encoder must learn features from $\mathbf{x}_i$ that are not contained in $\mathbf{x}_{ns}$ but

shared with $\mathbf{x}_p$.

3.2.2 Texture-based negative sample generation

We use texture synthesis tools to generate negative samples. Texture synthesis aims at generating realistic images that preserve as much local structure as possible from an example image [54, 95, 186]. For instance, as shown in Figure 5.1d, the texture of the input dog image preserves the fur and color of the dog. Notably, such local structure has often been recognized as highly correlated with the labels yet superfluous to generalization in the previous discussion of robustness [16, 240]. For example, under large domain shift due to lighting, motion, and even modality, the texture is more apt to change than the semantic features, such as shape. Furthermore, CNNs trained on ImageNet are more likely to classify images based on the texture features rather than the shape features which are instead preferred by humans due to their transferability [77]. To encourage the model to rely more on the shape features, we propose a two-step method to generate the texture image of input images as the negative samples for contrastive learning.

To be specific, we first sample two patches from given images as the input to the texture synthesis algorithms. One patch is extracted from the center of the image. This patch is expected to reflect the texture of the object according to the implicit bias contained in the ImageNet dataset that most of the objects are center-oriented in the images [11]. The other patch is extracted from a random location to reflect other possible textures of the image (e.g. background, peripheral region

of the object). In this work, we extract patches with size 96×96 when image size allows, otherwise 48×48 patches are extracted. Second, we adopt off-the-shelf texture synthesis algorithms [2, 66, 247] to generate texture images based on the two patches. These non-parametric algorithms iteratively sample pixels from given patches that share a similar neighborhood with the current pixel. Specifically, we use the open-source software built on these methods [2, 66, 247] with multi-threaded CPU support implemented in Rust ². For each sample in the ImageNet dataset, we generate one 224×224 texture image to construct a dataset that has the same training and validation size as the ImageNet dataset. We call this dataset ImageNet-Texture. More examples can be found in section 3.6.1.

3.2.3 Patch-based negative sample generation

To simulate the local information contained in the images [16, 17], we propose an efficient patch-based method to generate non-semantic images. Given an image and certain patch size d , we sample patches of size d from $(\lceil \frac{224}{d} \rceil)^2$ non-overlapping random locations that lie entirely in the image. The patches are then tiled and cropped into 224×224 as negative samples. Compared with the texture-based method, this generation process takes negligible time, therefore it can be implemented as part of the data loading process and paralleled with training. By doing so, each training sample can be paired with different negative samples generated from different patches every time it is used, compared with the two fixed patches selected when generating texture images.

²<https://github.com/EmbarkStudios/texture-synthesis>

Different from the texture-based method that generates realistic images, the patch-based method generates images with artificial lines as shown in Figure 5.1e. One might be concerned with possible degenerate solution where the model outputs a low similarity whenever it detects the repeated sharp changes in the horizontal or vertical directions, which could be done with a single layer of convolution. However, interestingly, we find that the model does not find such a simple solution in practice. This is also noticed in a previous study where the image and its copy with a patch cut out are non-trivially distinguished by the model [142]. To mitigate this potential issue, we randomly sample patch size d from a prior distribution instead of using a fixed d in practice, which allows the model to look at texture at different scales.

3.2.4 How hard are the texture-based and patch-based negative samples?

Contrastive learning methods are known to struggle with finding hard negative samples [70, 122] and researchers have proposed several ways to better leverage hard negatives [122, 193]. An intermediate question is how hard are our proposed negative samples compared with those standard negative samples used in previous contrastive learning methods [35, 92], i.e. random training samples.

We use the official MoCo-v2 model pretrained on the ImageNet-1K dataset for 200 epochs to calculate the cosine similarities between different kinds of pairs across ImageNet training set. We plot the histogram of these similarities in Figure 3.2. As shown in the Figures 3.2a and 3.2b, most positive pairs and negative pairs have

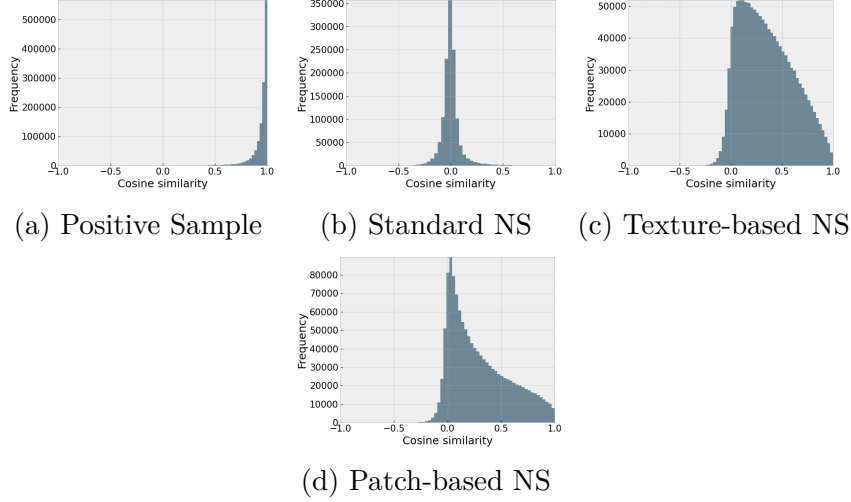


Figure 3.2: The histogram of cosine similarity between the representations of query sample and its paired samples, namely positive sample and standard, texture-based, and patch-based negative samples (NS), using MoCo-v2 model trained on the ImageNet-1K dataset for 200 epochs.

similarity close to 1 and 0 respectively. Specifically, the average similarities across the training samples are 0.94257 and 0.00018 for positive and negative pairs. As shown in the Figures 3.2d and 3.2c, the distributions of patch-based and texture-based negative samples are very different from those of standard negative samples; their similarity distributions have heavy tails in the positive region. Specifically, the distribution of patch-based and texture-based negative samples have average similarity 0.29503 and 0.35248 across the dataset, which shows that they remain difficult after training with standard negative examples.

3.3 Experiments

In this section, we evaluate the two proposed methods to generate non-semantic negative samples with two contrastive learning methods, MoCo and BYOL, on the ImageNet-1K dataset. We also experiment using its subset, the ImageNet-

100 dataset [122, 219], which allows us to perform more comprehensive experiments. We report accuracy on the out-of-domain (OOD) datasets including the ImageNet-C(orrupction) [99], ImageNet-S(ketch) [240], Stylized-ImageNet [77], and ImageNet-R(endition) [98] datasets as an evaluation of model’s robustness to corruptions and large domain shifts. ImageNet-C and Stylized-ImageNet contain images transformed from the images in the ImageNet validation set with common corruption and transferred style. ImageNet-S and ImageNet-R are collected independently from the ImageNet dataset and share all or a subset (200) of the classes in the ImageNet dataset with a focus on sketch and other rendering modalities. We show that with our proposed non-semantic negatives, contrastive learning can better generalize under corruptions and domain shifts. For patch-based negatives, it also improves the performance on the in-domain dataset.

3.3.1 ImageNet-100

We first follow the hyperparameters used in [122] to train MoCo-v2 on the ImageNet-100 dataset with a memory bank size $k = 16384$. We also use a similar hyperparameter configuration in which only memory bank size is halved. For patch-based augmentation parameters, we use patch size sampled from a uniform distribution $d \sim \mathcal{U}(16, 72)$. The parameter α is indicated behind each model name. We discuss the impact of α in detail in the next section. More ablations on the patch-based augmentations can be found in section 3.8.3. For ImageNet-C, we report the average accuracy across 5 levels of corruption severity.

Table 3.1: Top-1 accuracy on the ImageNet-100 dataset and its OOD variants. For our main comparison using MoCo-v2 and BYOL, we also report the standard deviation of 3 runs. For MoCo models, k represents the size of memory bank.

	ImageNet	ImageNet-C	ImageNet-S	Stylized-ImageNet	ImageNet-R
MoCo-v2 [39] - $k = 16384$	77.88 $\pm$ 0.28	43.08 $\pm$ 0.27	28.24 $\pm$ 0.58	16.20 $\pm$ 0.55	32.92 $\pm$ 0.12
+ Texture-based - $\alpha = 2$	77.76 $\pm$ 0.17	43.58 $\pm$ 0.33	29.11 $\pm$ 0.39	16.59 $\pm$ 0.17	33.36 $\pm$ 0.15
+ Patch-based - $\alpha = 2$	79.35 $\pm$ 0.12	45.13 $\pm$ 0.35	31.76 $\pm$ 0.88	17.37 $\pm$ 0.19	34.78 $\pm$ 0.15
+ Patch-based - $\alpha = 3$	75.58 $\pm$ 0.52	44.45 $\pm$ 0.15	34.03 $\pm$ 0.58	18.60 $\pm$ 0.26	36.89 $\pm$ 0.11
MoCo-v2 [39] - $k = 8192$	77.73 $\pm$ 0.38	43.22 $\pm$ 0.39	28.45 $\pm$ 0.36	16.83 $\pm$ 0.12	33.19 $\pm$ 0.44
+ Patch-based - $\alpha = 2$	79.54 $\pm$ 0.32	45.48 $\pm$ 0.20	33.36 $\pm$ 0.45	17.81 $\pm$ 0.32	36.31 $\pm$ 0.37
BYOL [85]	78.76 $\pm$ 0.28	44.43 $\pm$ 0.35	35.84 $\pm$ 0.38	15.01 $\pm$ 0.19	39.53 $\pm$ 0.51
+ Patch-based - $\alpha = 0.05$	78.81 $\pm$ 0.33	44.60 $\pm$ 0.21	36.76 $\pm$ 0.51	15.52 $\pm$ 0.22	41.16 $\pm$ 0.39
InsDis [250]	68.52	28.93	16.67	9.86	19.60
CMC [219]	79.34	39.28	24.04	13.88	32.68
InfoMin [222]	82.74	48.87	38.43	18.14	40.68
Supervised	86.26	49.17	34.95	21.20	39.76

Similar to [122], we repeat the experiments for 3 runs and report the mean and standard deviation in Table 3.1. As shown in the table, when following the previous memory bank size, using patch-based and texture-based negatives both improve the OOD generalizations. Specifically, patch-based augmentation increases the accuracy on ImageNet-S by 5.79% and ImageNet-R by 5.97% when $\alpha = 3$. When $\alpha = 2$, it also increases the in-domain accuracy by 1.47% and accuracy on ImageNet-C by 2.05%. A similar observation on standard ImageNet and ImageNet-C can be attributed to the resemblance of the images in the two dataset, especially those corrupted images with a lower level of severity. We show the performance of the model with $\alpha = 3$ is actually better on the highest corruption level as shown in section 3.8.2. The improvement achieved using texture-based negatives is less, probably because the information contained in the texture image is restricted due to the limited access to the two fixed patches. When the memory bank is halved

to be 8,192, the baseline MoCo model has slightly worse performance, decreasing from 77.88 to 77.73. But with patch-based hard negative samples, the MoCo-v2 model instead achieved the best accuracy 79.54 on the ImageNet-100 validation set, ImageNet-C and ImageNet-R. We discuss this more in a later section.

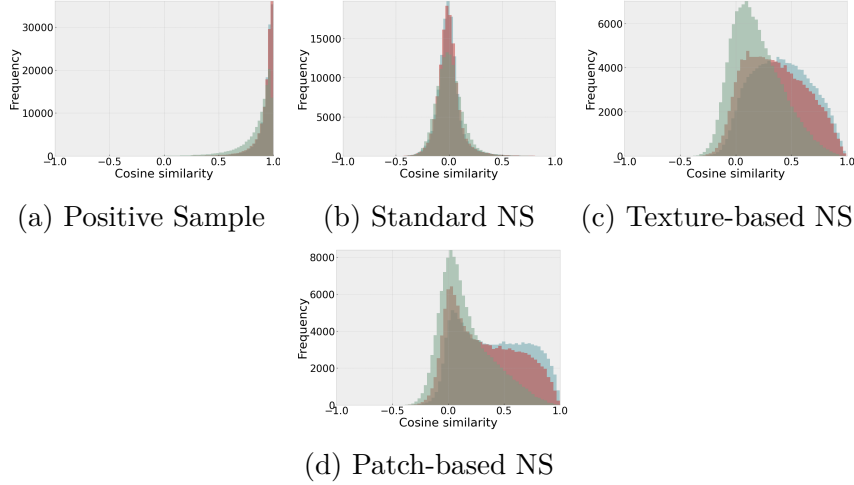


Figure 3.3: The histogram of cosine similarity between the representations of query sample and its paired positive sample, standard, texture-based, and patch-based negative samples (NS), using models trained without (blue) and with patch-based negative samples (red: $\alpha = 2$, green: $\alpha = 3$).

Similar to Figure 3.2, we visualize the distribution of the cosine similarity between the query sample and both semantic and non-semantic samples calculated based on the models trained with and without patch-based negative samples in Figure 3.3. The shift of the distribution towards the origin in Figure 3.3d meets the expectation that our method reduces the similarity of input images and patch-based negative samples. Specifically, the average similarity decreases from 0.4040 to 0.3252 to 0.1593 when α increases from 0, namely no patched-based negatives, to 2 to 3. Interestingly, we notice in Figure 3.3c that the similarity to texture-based negative samples also decreases, and the average similarity decreases from 0.4114

to 0.3541 to 0.1896, although we did not explicitly penalize it. This demonstrates the resemblance of patch-based negative samples to texture-based negative samples. Given the better performance achieved with patch-based negative samples, for the rest of experiments, we mainly focus on the patch-based methods. But we still conduct our analysis on the texture-based samples. We also find a marginal decrease in the positive similarity (-0.0068) and negative similarity (-0.0008) when $\alpha = 2$ and a substantial decrease in the positive similarity (-0.0737) and increase in the negative similarity (0.0036) when $\alpha = 3$. A similar figure for texture-based negative samples can be found in Fig 3.3.

3.3.2 ImageNet-1K

We follow the official hyperparameters [39] to train MoCo-v2 with our patch-based negative samples on the ImageNet-1K dataset. For the parameter α and patch size d , we follow the same configuration used on the ImageNet-100 dataset. We compare our results against the MoCo-v2 baseline [39] and the hard negative mixing algorithm, MoCHi [122]. Due to limited computational resources, we report the metrics evaluated with the official model without repeated runs. The results are shown in Table 3.2. More results can be found in section 3.8.6. Note that for ImageNet-C, we show the mCE metric [99], for which smaller is better. For the other datasets, we show the top-1 accuracy.

Table 3.2: Top-1 accuracy on the ImageNet-1K dataset and its sketch, stylized, rendition variants, and mCE on the ImageNet-C dataset.

	ImageNet	ImageNet-C	ImageNet-S	Stylized-ImageNet	ImageNet-R
MoCo-v2 [39]	67.60	87.7	17.47	5.55	27.81
+ MoCHi [122]	67.56	88.7	16.32	5.94	25.71
+ Patch-base NS - $\alpha = 2$	67.92	87.6	18.58	6.34	28.95

3.3.3 Memory bank size

Contrastive learning methods based on negative samples suffer from ineffective excavation of hard negatives [70, 122] and resort to large batch sizes [35] or memory bank [92]. In this section, we study whether our proposed negative samples can mitigate this problem on the STL-10 and ImageNet-100 datasets. We keep the hyperparameters intact and vary the memory bank size. We report the accuracy of the MoCo-v2 baseline with and without patch-based negative samples on STL-10 dataset in Table 3.3. We also compare with [193] which exploits hard negatives through reweighting. We found that with proper hyperparameters the MoCo-v2 baseline already beats the reweighting results with SimCLR.

SimCLR [35]	80.16
+ Debiased [44]	84.90 [§]
+ Hard [193]	87.42 [§]
MoCo-v2 [39]	88.00
+ Patch-based NS	89.36

Table 3.3: Top-1 accuracy on the STL-10 dataset.

[§] denotes results visually extracted from Figure 2 in [193].

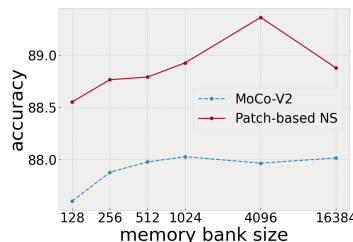


Figure 3.4: Top-1 accuracy on the STL-10 dataset with different memory bank sizes.

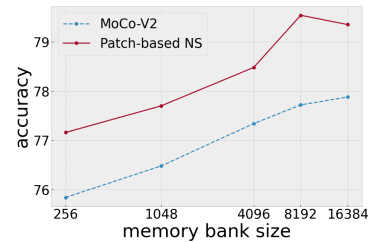


Figure 3.5: Top-1 accuracy on the ImageNet-100 dataset with different memory bank sizes.

As shown in Figures 3.4 and 3.5, using patch-based non-semantic negatives consistently improves the MoCo baseline when the number of standard negatives varies. Specifically, when slightly decreasing the memory bank sizes, the performance with patch-based negatives increases. This is probably because according to Eq. 3.1 and analysis in section 3.7.1, the smaller memory bank sizes indicate a larger contribution of non-semantic negatives to the loss, and consequently a larger regularization. When the memory bank size continues to decrease, the baseline performance decreases similar to the observations in the previous studies [92, 193]. However, the steady improvement caused by the non-semantic negatives effectively mitigates the problem. For example, on ImageNet-100, when using 1024 standard negatives with patch-based negatives, the model can achieve accuracy on par with 16384 standard negatives without patch-based negatives. On STL-10, the decrease caused by a smaller memory bank is less substantial and using patch-based negatives always beats the baseline.

3.4 Discussion

3.4.1 Controlling the shape-texture trade-off with α

There has been a growing interest in understanding the cause and impact of the trade-off between shape and texture bias of CNNs [77, 103, 115, 146]. CNNs trained on ImageNet are known to be over-reliant on the texture features [77]. Contrastive learning with our non-semantic negatives serve as not only an effective method to reduce such reliance, but a natural tool to study such a trade-off.

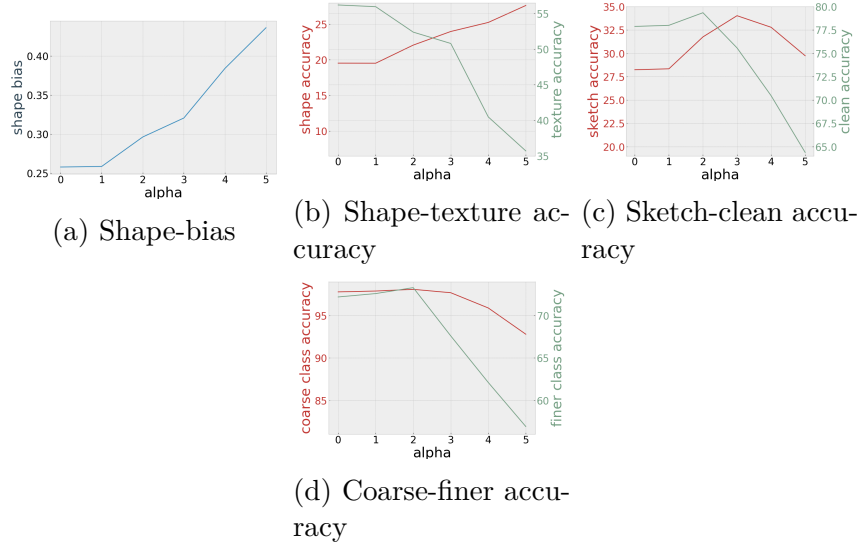


Figure 3.6: Larger α monotonically increases the model bias to shape features over texture features. Model performance is impacted by such a trade-off differently under different settings. In all scenarios, slightly calibrated shape bias improves model performance. The test settings represented in the red lines gain more from the increased shape bias than the settings represented in the green lines.

We train MoCo-v2 models with different α from 1 to 5 on the ImageNet-100 dataset. As shown in Figure 3.6a, we find that α effectively controls the trade-off on the model bias to the shape and texture features. $\alpha = 0$ is used to denote the baseline method. Specifically, a larger α in the loss function 3.1 leads to a larger penalty on the similarity between the representations of query samples and non-semantic samples, consequently a larger shape bias. We follow [77] to calculate shape bias on the stimuli images with conflicted shape and texture clues generated by style transfer. We show the corresponding accuracy on the shape and texture labels in Figure 3.6b.

As shown in Figure 3.6b, when α increases the texture accuracy on the stimuli dataset monotonically decreases while the shape accuracy monotonically increases. To further study how the trade-off between shape and texture bias impacts the model

performance, we first compare the accuracy on the ImageNet validation dataset and ImageNet-Sketch dataset [240] when α varies in Figure 3.6c. We find that on both datasets, slightly increased shape bias over baseline ($\alpha = 2$) improves performances. Interestingly, the accuracy peak on the ImageNet-Sketch appears at $\alpha = 3$ while the peak appears at $\alpha = 2$ on the ImageNet validation dataset. In addition, for even larger α the accuracy on the ImageNet-Sketch dataset still outperforms the baseline while the standard accuracy gets hurt. This shows that different downstream tasks may benefit differently from differently shape-biased models.

We plot the histograms of similarities calculated by the models trained with different α in Figure 3.12. We find that large α makes original pretext task challenging - the model cannot effectively pull together the representations of positive pairs. Specifically, when α increases from 1 to 5, the average similarity of the positive pairs decrease from 0.9267 to 0.7541. This demonstrates that it is hard for the model to learn representations that are completely independent of the texture features contained in the non-semantic images.

3.4.2 Rethinking the shape-texture trade-off through class-based analysis

The initial discussion on the shape-texture trade-off shows that humans rely more on the shape features while CNNs rely more on texture features and increasing shape bias can improve accuracy and robustness [77]. However, similar to [103, 172], we notice that increasing shape bias does not always improve the generalization and

robustness of the models. To better understand this phenomenon, we provide two observations based on the analysis of the ImageNet-Texture dataset and our method to explain why a texture-biased model is helpful with classification on the ImageNet dataset.



Figure 3.7: A ResNet-50 model trained on the ImageNet dataset achieve decent accuracy when only texture features are available on some classes. A normal image and its texture version are displayed for some of these classes. The caption indicates the class ID, name, and accuracy on texture images.

First, we find that an increasing shape bias often leads to more errors among the fine-grained classes. The initial discussion of the shape-bias [77] only pays attention to the selected 16 coarse classes. We thus compare the finer and coarse class accuracy on the dog images of ImageNet dataset as in Figure 3.6d. For coarse class accuracy, the predictions are counted to be correct whenever the image is classified as a dog class, no matter which dog class is predicted, while for the finer class accuracy, only those predictions of target dog classes are counted to be correct. We notice that the finer class accuracy drops more significantly when shape bias increases as opposed to the coarse class accuracy. For example, when $\alpha = 3$, the finer class accuracy drops from 72.9 to 67.6 while the coarse class accuracy slightly decreases from 97.8 to 97.7. Therefore, for datasets with numerous fine-grained classes like ImageNet, we conjecture that a texture-biased model is more helpful for a higher accuracy. Second, in Figure 3.9, we show a scatter plot of texture accuracy

vs. standard accuracy of different model architectures and a histogram of accuracy on individual ImageNet-Texture classes. We identify several classes where using only texture features is sufficient to achieve a high classification accuracy. These classes are all missing in the previous study [77]. As shown in Figure 3.7 and Figure 3.10, texture serves as a more important clue than the shape for these classes.

3.5 Related Work

Contrastive learning based self-supervised learning Recent contrastive learning based self-supervised learning methods including MoCo [92], SimCLR [35], InfoMin [222], SimSiam [40], BYOL [85], SwAV [23], Barlow Twins [266] have proven helpful in learning visual representations. These methods rely on different pretext tasks. The augmentations used to generate these views are essential to the success of these contrastive learning methods [23, 35] by preventing shortcuts such as the use of simple color histogram [35]. New state-of-the-art methods have suggested new types of data augmentation such as multi-crop in SwAV [23] and PIRL JigSaw [169] in InfoMin [222]. However, it is intractable to enumerate every augmentation and impossible to eliminate every shortcut. Different from these methods that focus on augmentations to create positive pairs, we show that augmentations that perturb the semantic features can be used to craft negative samples. By maximizing the difference between natural images and their non-semantic versions in the representation space, the models are coerced to avoid any potential shortcuts shared by them.

Methods like MoCo [92] and SimCLR [35] distinguish positive pairs from negative pairs that are picked from the rest of the dataset. However, most of the negatives prove to be unnecessary and insufficient [70, 122]. To excavate effective negative samples, these methods heavily depend on the large batch sizes [35] or memory bank [92]. Utilizing hard negative samples has long been recognized as an effective approach to boost model performance [88, 119, 208, 259]. In the contrastive learning studies, [44, 193] modify the contrastive learning loss to make it assign greater weights to the hard negative samples. [122] proposes to synthesize hard negative samples by taking linear combinations of the hardest negative samples. Our work is orthogonal to these ideas in the way that we propose to generate negative samples from given images themselves to reduce the reliance on the undesired features. In addition, two recent works [107, 126] study the application of adversarial examples as hard positive and negative samples in contrastive learning. [197] augments the images by manipulating their foregrounds and backgrounds to generate negative and positive samples. Compared with these studies, we mainly focus on the OOD evaluation of the models. In addition, our patch-based augmentation is also related to the self-Supervised learning methods that adopt the pretext task based on jigsaw [96, 169, 178], .

Robustness and out-of-domain generalization of CNNs High test accuracy provides no guarantee that a network learns high-level semantic features instead of low-level superfluous features that exist in both training and test dataset [120]. An increasing number of studies have corroborated such concerns and found that CNNs can rely

on local patches [16, 17], texture [77], high-frequency components [242] and even artificially added features [110] to achieve high test accuracy. These superficial correlations become brittle under large domain shifts [98, 99]. This still remains an unsolved problem [217] and is rarely discussed in the contrastive learning setting.

Among all these undesired features, the shape-texture bias has been widely discussed in recent studies [77, 162]. Previous work has shown that CNNs trained on the ImageNet dataset are biased to texture features and such over-reliance can hurt the generalization performance of CNNs [77, 104]. Several studies have aimed at mitigating this problem [146, 167] or providing a better understanding [103, 115]. In this chapter, we introduce a dataset called ImageNet-Texture, which can help future studies on these problems. Our method also effectively controls the trade-off between shape and texture bias. We provide new insights about this problem based on the analysis of our method and dataset.

3.6 ImageNet-Texture

See Figures 3.7 and 3.8 for examples of the ImageNet-Texture dataset and their counterparts in the original ImageNet dataset. The analysis of models pretrained on ImageNet and evaluated on ImageNet-Texture is presented in Section 3.6.2.

3.6.1 Examples

3.6.2 Analysis

The proposed ImageNet-Texture dataset contains images with preserved texture information and diminished shape information, which can be used to better understand how models behave when only texture features are available. We evaluate the pretrained models provided in Pytorch model zoo ³ on the ImageNet-Texture versus standard ImageNet and report the accuracy in Figure 3.9a. As shown in the figure, VGG networks generally have a higher accuracy on the ImageNet-Texture dataset. For the rest of the model architectures, we see a positive correlation between the standard accuracy and texture accuracy.

We also plot the histogram of the classes with different accuracy based on a pretrained ResNet-50 model in Figure 3.9b, which achieves a relatively high accuracy, 9.3%, among all the models. The class-based accuracy denotes the accuracy on the images of the same certain class. We find that on the 343 out of 100 classes, the model can only achieve an accuracy smaller than 2.5%, which shows that texture feature only is not sufficient to classify these classes. But we do notice a long tail of the distribution. Specifically, 15 classes have an accuracy larger than 50%. Sample images of these classes are demonstrated in Figure 3.10. Notably, texture plays an important role in distinguishing these classes. Shape is often less well-defined in these classes, for example in window screen and rapeseed.

³<https://pytorch.org/vision/stable/models.html>

3.7 Discussion on the contrastive learning with non-semantic negatives

3.7.1 Comparison of two ways to apply α in NCE loss

Since the non-semantic negative samples play different roles from the original negative samples, we introduce an additional parameter to control the penalty w.r.t. non-semantic negatives. There are two straightforward alternatives to implement α , which we call $\mathcal{L}_{in}$ and $\mathcal{L}_{out}$:

$$\mathcal{L}_{in} = - \sum_{i \in I} \log \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_p / \tau)}{\exp(\mathbf{z}_i \cdot \mathbf{z}_p / \tau) + \exp(\alpha \mathbf{z}_i \cdot \mathbf{z}_{ns} / \tau) + \sum_{n \in \mathcal{N}} \exp(\mathbf{z}_i \cdot \mathbf{z}_n / \tau)}$$

$$\mathcal{L}_{out} = - \sum_{i \in I} \log \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_p / \tau)}{\exp(\mathbf{z}_i \cdot \mathbf{z}_p / \tau) + \alpha \exp(\mathbf{z}_i \cdot \mathbf{z}_{ns} / \tau) + \sum_{n \in \mathcal{N}} \exp(\mathbf{z}_i \cdot \mathbf{z}_n / \tau)}$$

We compare the relative importance of different pairs play in the gradient w.r.t. $\mathbf{z}_i$:

$$\frac{\partial \mathcal{L}_{in}}{\partial \mathbf{z}_i} = \frac{\mathbf{z}_p / \tau \exp(\mathbf{z}_i \cdot \mathbf{z}_p / \tau) + \alpha \mathbf{z}_{ns} / \tau \exp(\alpha \mathbf{z}_i \cdot \mathbf{z}_{ns} / \tau) + \sum_{n \in \mathcal{N}} \mathbf{z}_n / \tau \exp(\mathbf{z}_i \cdot \mathbf{z}_n / \tau)}{\exp(\mathbf{z}_i \cdot \mathbf{z}_p / \tau) + \exp(\alpha \mathbf{z}_i \cdot \mathbf{z}_{ns} / \tau) + \sum_{n \in \mathcal{N}} \exp(\mathbf{z}_i \cdot \mathbf{z}_n / \tau)} - \mathbf{z}_p / \tau$$

$$\frac{\partial \mathcal{L}_{out}}{\partial \mathbf{z}_i} = \frac{\mathbf{z}_p / \tau \exp(\mathbf{z}_i \cdot \mathbf{z}_p / \tau) + \alpha \mathbf{z}_{ns} / \tau \exp(\mathbf{z}_i \cdot \mathbf{z}_{ns} / \tau) + \sum_{n \in \mathcal{N}} \mathbf{z}_n / \tau \exp(\mathbf{z}_i \cdot \mathbf{z}_n / \tau)}{\exp(\mathbf{z}_i \cdot \mathbf{z}_p / \tau) + \alpha \exp(\mathbf{z}_i \cdot \mathbf{z}_{ns} / \tau) + \sum_{n \in \mathcal{N}} \exp(\mathbf{z}_i \cdot \mathbf{z}_n / \tau)} - \mathbf{z}_p / \tau$$

Since the denominator normalizes the 3 kinds of pairs equally, we only pay attention to the numerator. The difference between $\mathcal{L}_{out}$ and $\mathcal{L}_{in}$ is that $\mathcal{L}_{out}$ has α inside the exponential. Because of the exponential tail, it applies a exponentially

larger weight to the negatives that are harder. Since non-semantic negatives are often harder as shown in Figure 3.2, $\mathcal{L}_{in}$ regularizes the non-semantic negatives more effectively than $\mathcal{L}_{out}$. Note that the implementation of $\mathcal{L}_{out}$ is equivalent to using α non-semantic negatives samples for each input image. In addition, the number of standard negative $|\mathcal{N}|$ in the loss also affects the relative importance of non-semantic. The smaller $|\mathcal{N}|$ is, the larger effect is played by the non-semantic negative.

3.7.2 Comparison of patch-based negatives and jigsaw-based pretext tasks

Our patch-based augmentation is also closely related to some of the self-supervised learning methods which solve jigsaw as the pretext task. Specifically, [178] proposes to learn meaningful representations through predicting the order of shuffled 3x3 patches of a given image. [169] further extends this idea to contrastive learning and learn representations that are invariant under such pre-text transformation. [96] exploit the connection between local patches to learn the representations by fusing it with contrastive predictive coding. The common idea behind these methods is to learn the representations based on the correct configuration of the patches in the natural images. However, our patch-based augmentation treats the entire image with unsorted patches as a wrong configuration of the a natural image and use it as the negative samples in the contrastive learning.

3.8 Experimental details and additional results

3.8.1 Implementation details

For ImageNet-1K dataset, we follow the official hyperparameters to MoCo-v2 [39]. For ImageNet-100 and STL-10 datasets, we follow [122] which uses a different memory bank size of 16384 and batch size 128 during the pretraining. For linear evaluation, we adopt a learning rate 10.0, batch size 128, and a learning schedule that decreases the learning rate by 0.1 at 30, 40, and 50 epochs. For BYOL pretraining, we follow [85] to utilize the LARS optimizer with cosine learning rate schedule with a global weight decay parameter of 10^{-6} and momentum of 0.9. We use a batch size of 1024 and an initial learning rate 4.8. For both MoCo and BYOL, the non-semantic negatives are input to the momentum branch. All of our models are trained on 4 GTX 1080 Ti gpus. Training single model takes around 1.5 and 12 days on the ImageNet-100 and ImageNet-1K datasets respectively.

3.8.2 Results on ImageNet-C with different levels of severity

We show the accuracy of different MoCo-v2 models on the ImageNet-C-100 dataset with different corruption severity levels in Table 3.4. Specifically, as we have seen in the main body of the paper, a larger α often favors larger domain shift. This is further demonstrated by the fact that the model with $\alpha = 3$ outperforms the model with $\alpha = 2$ on the highest corruption level 5.

Table 3.4: Top-1 accuracy on ImageNet-C-100 dataset with different levels of severity.

Severity level	1	2	3	4	5
MoCo-v2 - $k = 16384$	$65.58_{\pm 0.32}$	$53.41_{\pm 0.27}$	$42.31_{\pm 0.31}$	$31.87_{\pm 0.31}$	$22.22_{\pm 0.20}$
+ Texture-base NS	$66.00_{\pm 0.47}$	$54.11_{\pm 0.43}$	$42.96_{\pm 0.41}$	$32.23_{\pm 0.26}$	$22.60_{\pm 0.23}$
+ Patch-base NS - $\alpha = 2$	$67.84_{\pm 0.21}$	$55.99_{\pm 0.20}$	$44.65_{\pm 0.40}$	$33.74_{\pm 0.51}$	$23.41_{\pm 0.50}$
+ Patch-base NS - $\alpha = 3$	$65.82_{\pm 0.04}$	$54.95_{\pm 0.13}$	$44.10_{\pm 0.17}$	$33.67_{\pm 0.25}$	$23.71_{\pm 0.28}$
MoCo-v2 - $k = 8192$	$65.44_{\pm 0.52}$	$53.52_{\pm 0.49}$	$42.71_{\pm 0.42}$	$32.09_{\pm 0.33}$	$22.36_{\pm 0.19}$
+ Patch-base NS - $\alpha = 2$	$68.01_{\pm 0.01}$	$56.26_{\pm 0.12}$	$45.04_{\pm 0.28}$	$34.16_{\pm 0.34}$	$23.92_{\pm 0.27}$

3.8.3 Additional ablations on the patch-based augmentations

3.8.3.1 Patch sizes

We show more results on the patch-based augmentations with different patch sizes by reporting the test accuracy on the ImageNet-100 validation set and corresponding ImageNet-Sketch dataset in Figure 3.11. Specifically, we sample patch sizes from uniform distributions $d \sim \mathcal{U}(x, y)$ of different interval $[x, y]$. We find that for the performance on the standard validation set, both the large or smaller patch sizes cause a less desired accuracy. But for the ImageNet-Sketch dataset, the larger patch sizes generally provide larger performance improvement. Our conjecture on this observation is that using larger patch sizes prevents the model to learn some of the local features that are shared between training and validation set while absent from the sketch dataset.

3.8.3.2 Data augmentations on individual patches

We test whether common data augmentation methods can be used to augment individual patches so as to further improve the performance. We report the accuracy on the ImageNet-100 validation set with the model trained with the patch-based negative samples in Table 3.5. The patches are potentially augmented with horizontal flip or vertical flip with 50% probability, or rotation in $0^\circ, 90^\circ, 180^\circ$ or 270° with 25% probability respectively. We find that for a larger patch size, i.e. $d \sim \mathcal{U}(16, 72)$, such augmentations always downplay the accuracy. But for a smaller patch size, such augmentations, especially with horizontal flip only, can improve the performance. For the larger patch size, we conjecture that it is important to ensure that the patches to be identical to the certain part of the input image. Therefore the non-semantic features are best preserved. However, for a smaller patch size, such augmentations have a less impact on the captured non-semantic information.

Table 3.5: Accuracy on ImageNet-100 validation set with different configurations of augmentations on the individual patches. Augmentations are cumulative across the columns (e.g. the “+ Vertical flip” model used horizontal and vertical flip).

Patch size	No aug.	+ Horizontal flip	+ Vertical flip	+ Rotation
8 – 28	78.58	78.72	78.66	78.64
16 – 72	79.35	79.06	78.92	78.40

3.8.4 Hardness of texture-based negative samples

We show the distribution of similarities between the representations of input image versus different paired samples in Figure 3.12. The similarities are calculated by

the models trained without (blue) and with (red) texture-based negative samples. We use $\alpha = 2$ when texture-based negative samples are adopted. Comparing with Figure 3.3, the regularization of texture-based negative samples is generally weaker than the patch-based negative samples. For example, the average similarity w.r.t patch-based negative samples decrease from 0.4040 to 0.3677 when using texture-based negative samples for training, which is 0.3252 when using patch-based negative samples.

3.8.5 Hardness of patch-based negative samples using models trained with different α

We show the distribution of similarities calculated by the models trained with patch-based negative samples and different α in Figure 3.13. When α increases from 0 to 5, we see a larger penalty on the similarities to both patch-based and texture-based negative samples. When $\alpha = 5$, the distributions of two negative samples are close to a normal distribution like the distribution of the standard negative samples. Specifically, the average similarity w.r.t. the patch-based negative samples further decreases to 0.4040 to 0.1184. In terms of the original pretext task, we find that the distribution of both standard negative and positive samples become wider and less concentrated to 0 and 1 respectively. This shows that a larger α makes the optimization of the original objective more difficult.

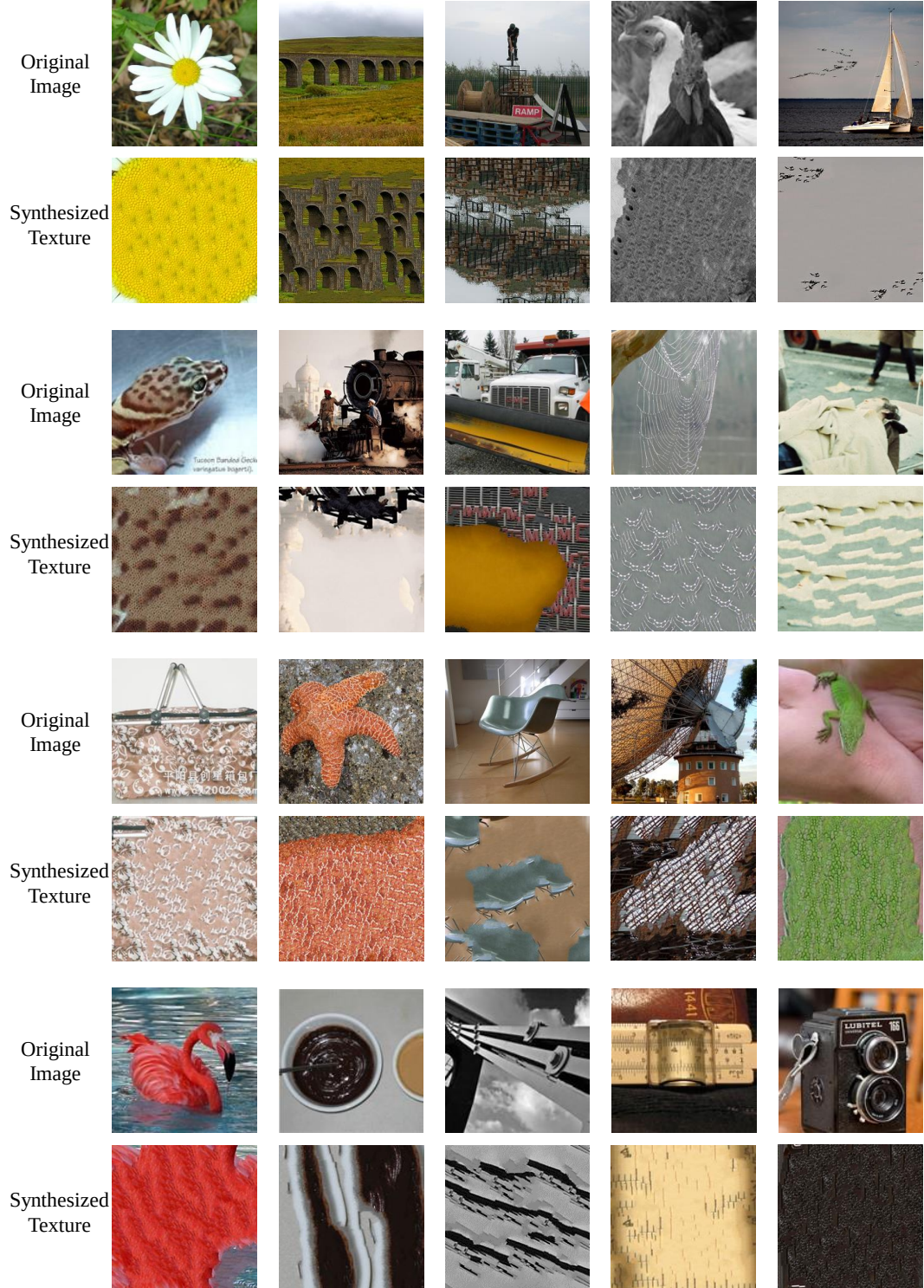


Figure 3.7: Randomly picked images from our ImageNet-Texture dataset and their corresponding original images from the ImageNet dataset.

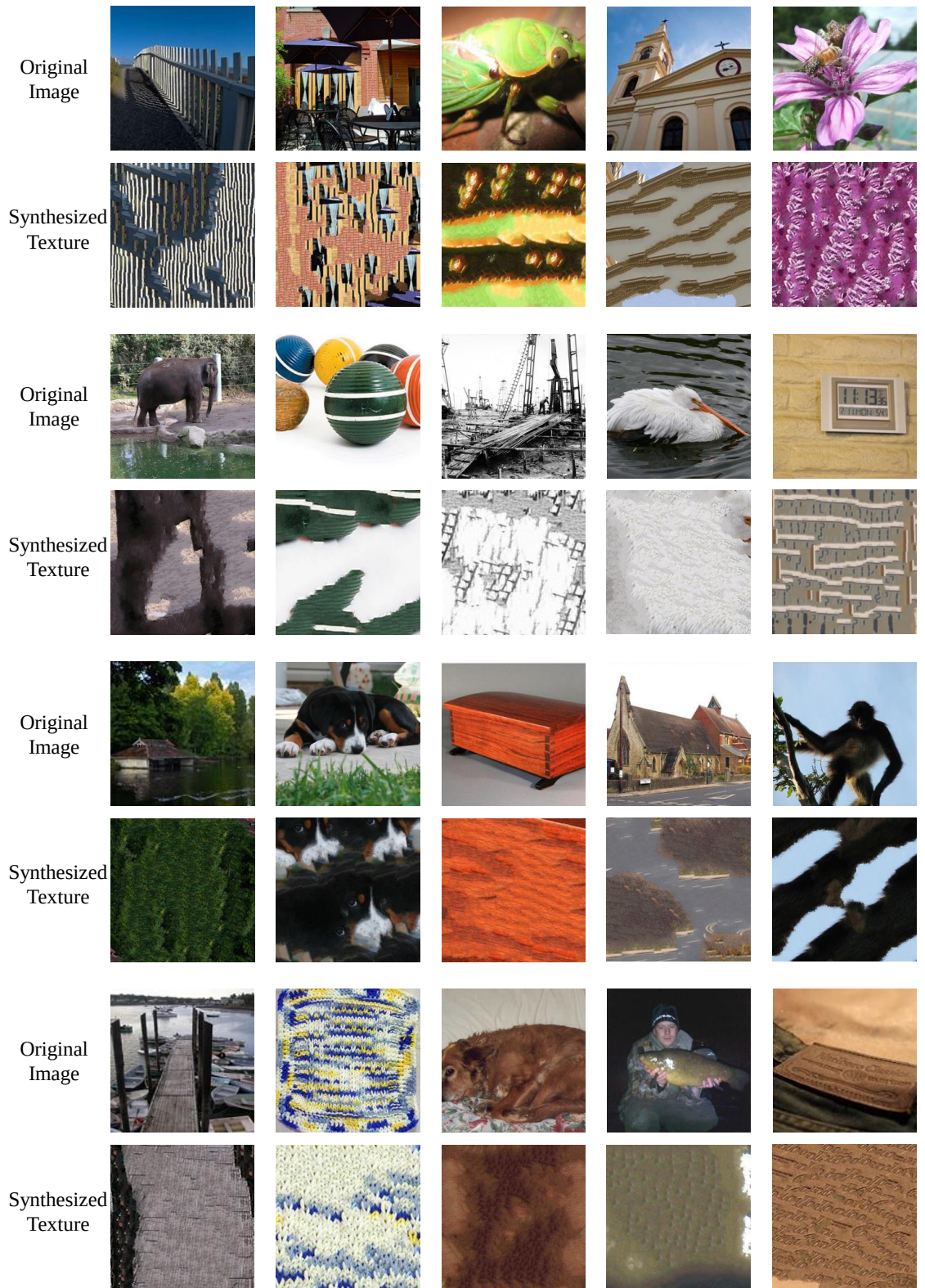
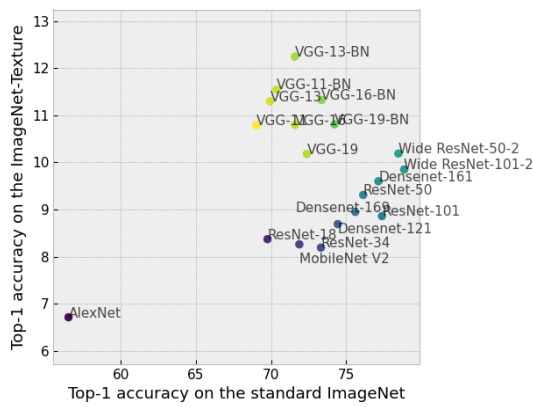
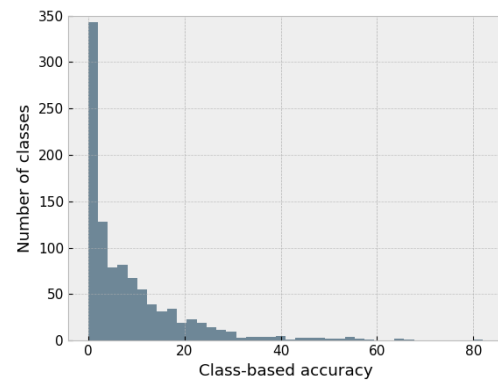


Figure 3.8: More randomly picked images from our ImageNet-Texture dataset and their corresponding original images from the ImageNet dataset.



(a)



(b)

Figure 3.9: (a) Accuracy on the ImageNet-Texture versus standard ImageNet with different model architectures. (b) Histogram of accuracy on the individual ImageNet-Texture classes with ResNet-50.

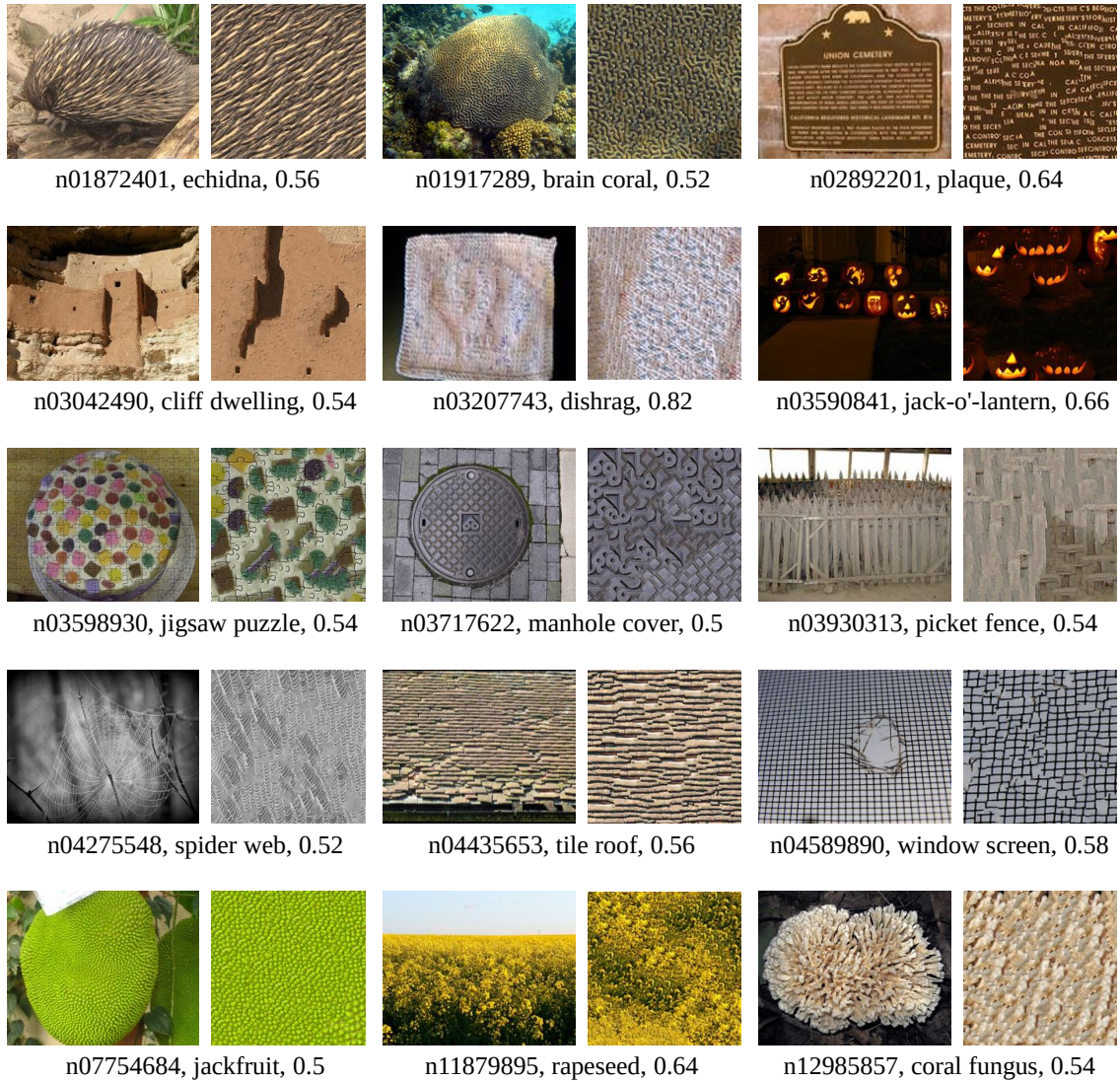


Figure 3.10: On some classes, a ResNet-50 model trained on the standard ImageNet dataset can achieve $> 50\%$ accuracy when only texture features are available. For each class, one sample image and its texture version are displayed. The caption of each image pair indicates the ImageNet class ID, class name and ResNet-50 accuracy on the texture images.

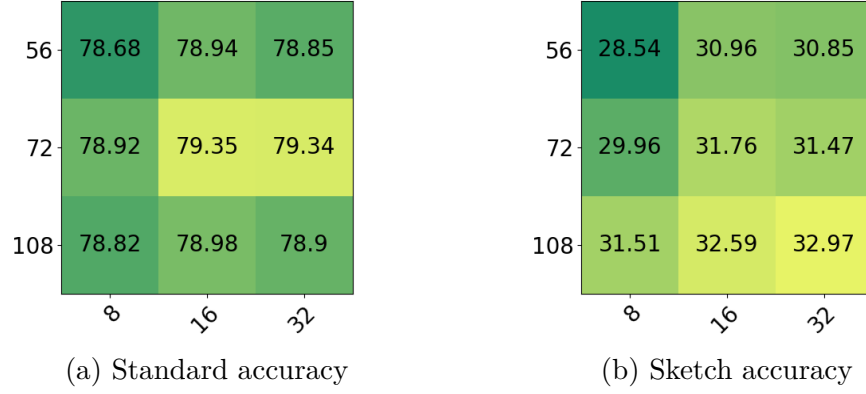


Figure 3.11: Test accuracy of patch-based non-semantic augmentations with patch sizes sampled from different uniform distribution. The x-axis indicates the lower boundary of the sampling interval and the y-axis indicates the higher boundary.

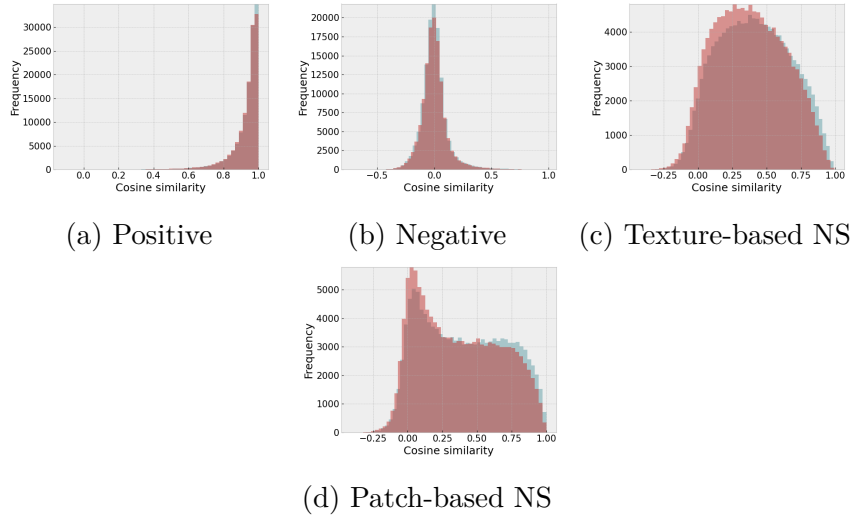


Figure 3.12: The histogram of cosine similarity between the representations of different kinds of pairs using models trained without (blue) and with (red) texture-based negative samples.

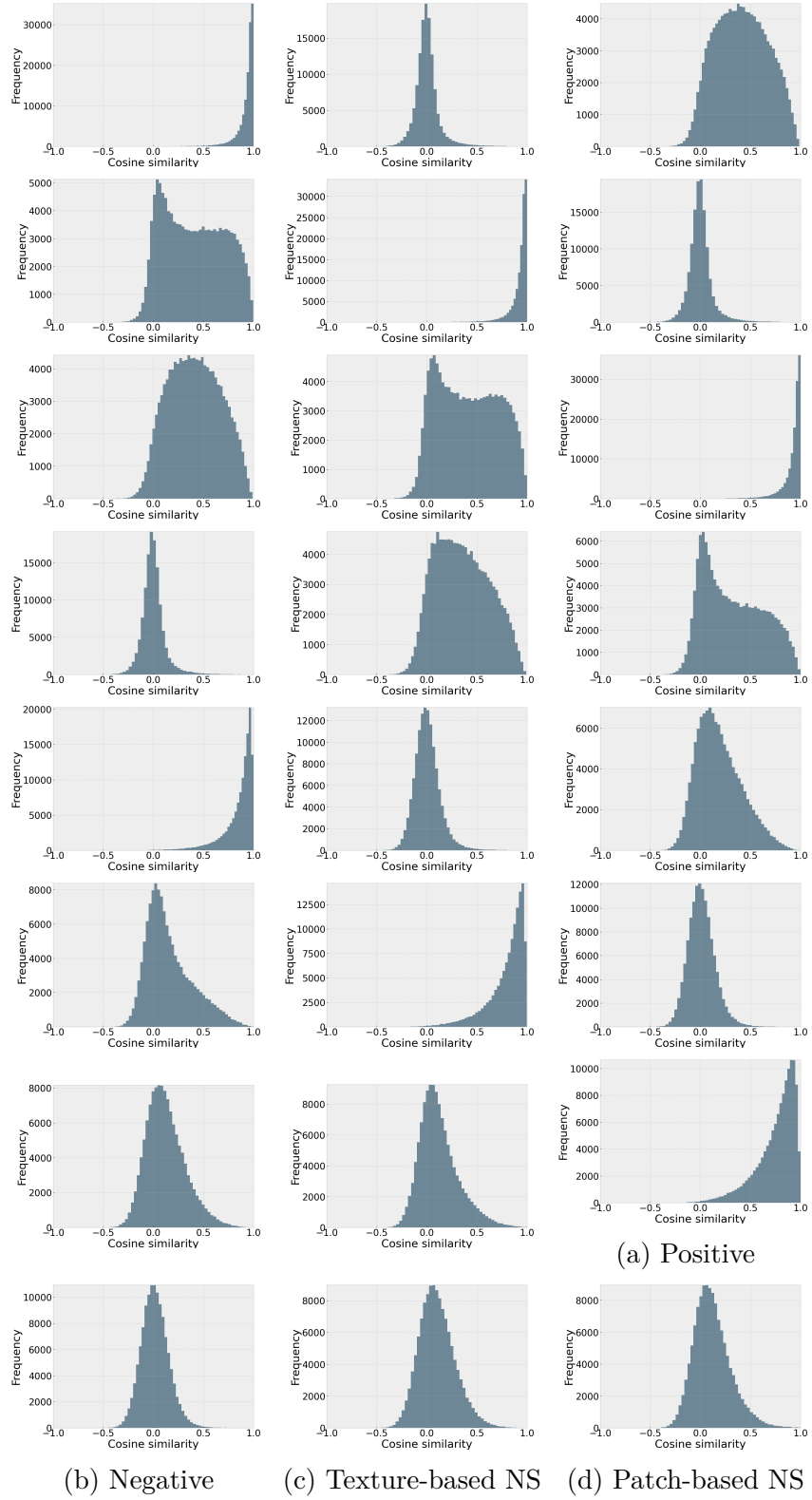


Figure 3.13: The histogram of cosine similarity between the representations of different kinds of pairs using models trained with patch-based negative samples and different α values from 0 (top) to 5 (bottom).

3.8.6 More results on ImageNet-1K

Table 3.6: Top-1 accuracy on the ImageNet-1K dataset and its sketch, stylized, rendition variants, and mCE on the ImageNet-C dataset.

	ImageNet	ImageNet-C	ImageNet-S	Stylized	ImageNet-R
MoCo-v2 [39]	67.60	87.7	17.47	5.55	27.81
+ MoChi (128,1024,512) [122]	66.62	90.0	16.00	5.32	25.29
+ MoChi (512,1024,512) [122]	67.56	88.7	16.32	5.94	25.71
+ Patch-base NS - $\alpha = 2, k = 65536$	67.92	87.6	18.58	6.34	28.95
+ Patch-base NS - $\alpha = 3, k = 65536$	60.80	92.1	18.79	6.80	28.11
+ Patch-base NS - $\alpha = 2, k = 32768$	67.83	87.2	18.70	6.06	28.50
+ Patch-base NS - $\alpha = 2, k = 16384$	67.34	87.5	19.49	6.49	29.45

We show more results on the ImageNet-1K dataset using MoCo-v2 models trained with patch-based negative samples in Table 3.6. We also report the other released MoChi model ⁴, and the different hyperparameters are indicated following the model name in the table. For our method, we focus on the models when a larger penalty on the similarity w.r.t. the patch-based negative samples is added. Specifically, we report the results with a larger $\alpha = 3$ and smaller memory bank sizes $k = 32768$ or $k = 16384$. As shown in the table, increasing the penalty stably improves the generalization under OOD settings. For example, when $\alpha = 2$ and $k = 16384$, the accuracy on ImageNet-S and ImageNet-R increase from 17.47 to 19.49 and 27.81 to 29.45 respectively.

3.9 Closing Remarks

Conclusion CNNs are prone to learn discriminative features that are vulnerable under domain shifts. In this chapter, we first demonstrate the regularization power

⁴<https://europe.naverlabs.com/research/computer-vision/mochi/>

of contrastive learning to discard any undesired features by generating appropriate negative samples. We explore two approaches, the patch-based and texture-based augmentations, to craft negative samples with only local features preserved. We show that the representations learned by contrastive learning with such negative samples depend less on the local features, and consequently generalize better under OOD settings. We hope this chapter can encourage people to rethink the role that negative samples play in contrastive learning, which hopefully leads to more efficient methods to generate negative samples. In the future we will aim to combine both anisotropic diffusion method as discussed in Chapter 2 and texture-based negative samples, since both of them could be complementary to each other.

Limitations: The problem of dependence on superficial features exists in various domains beyond vision, such as language [118, 164]. Therefore, it is intriguing to consider generalizing such an idea to other modalities. In addition, as the mechanism to ensure that contrastive learning models trained on large datasets to discard the bias of the datasets is yet to be invented, severe social issues in fairness or privacy may be raised as a result [18, 128]. In this chapter, we show how to calibrate the bias towards texture features using proposed negative samples. It is also worth considering whether contrastive learning can be used to address any of these negative effects triggered by bias in datasets.

Chapter 4: Object-aware Cropping for Self-Supervised Learning

4.1 Introduction

In recent works on self-supervised learning (SSL) of image representations, the most successful approaches have used data augmentation as a crucial tool [24, 35, 86, 91, 219]. Given a randomly chosen image sample, augmentations of the image are generated using common image transformations such as cropping and resizing a smaller region of the image, color transformations (hue, saturation, contrast), rotations etc. Of these augmentations, the use of cropping is clearly the most powerful (see [35], Fig. 5). This makes intuitive sense: cropping followed by resizing forces the representation to focus on different parts of an object with varying aspect ratios. This makes the representation robust to such natural transformations as scale and occlusion. The implicit assumption in this scheme is that the object of interest (classification or detection target) occupies most of the image and is fairly centered in the image, so that random crops of the image usually result in (most of) the object still being present in the cropped image. Such an assumption holds for “iconic” datasets such as ImageNet [135]. Forcing the resulting representations to be closer together maximizes the mutual information between the crops (also

called *views*) [219, 230].¹ However, in the case of “non-iconic” datasets such as OpenImages [137] and COCO [148], the objects of interest are small relative to the image size and rarely centered, see Fig. 4.1. These datasets are more representative of real-world uncurated data. We find that the default random cropping approach (which we call *scene* cropping) leads to a significant reduction in performance for self-supervised contrastive learning approaches. For example, using the default pipeline of MoCo-v2 [39], we find that there is a gap of 16.5% mean average precision (mAP) compared to fully supervised learning. Other state of the art methods such as BYOL [86], SwAV [24] and CMC [219] perform poorly as well (see Table 4.2). As we show, the core problem here is that random scene crops do not contain enough information about (small) objects, causing degraded representation quality.

However, merely switching from scene-level crops to purely object-level crops does not exploit the correlations that exist between scenes and objects in most natural images. These correlations are helpful for downstream tasks [252]. Keeping this in mind, we introduce *object-aware* cropping, which applies a simple pre-processing step using the unsupervised object proposal method LOD [238]. LOD outputs multiple object proposal rectangles, one of which we pick at random as the first candidate region. We then expand (dilate) this rectangle to create a second candidate region. We finally employ random cropping within each of these rectangles to create a final pair of “positive” views that are used for the SSL loss. The use of random crop-

¹The contents of this work is in collaboration with Anshul Shah, Ankan Bansal, Abhudhay Jaganntha, Abhishek Sharma, David Jacobs and Dilip Krishnan [168].

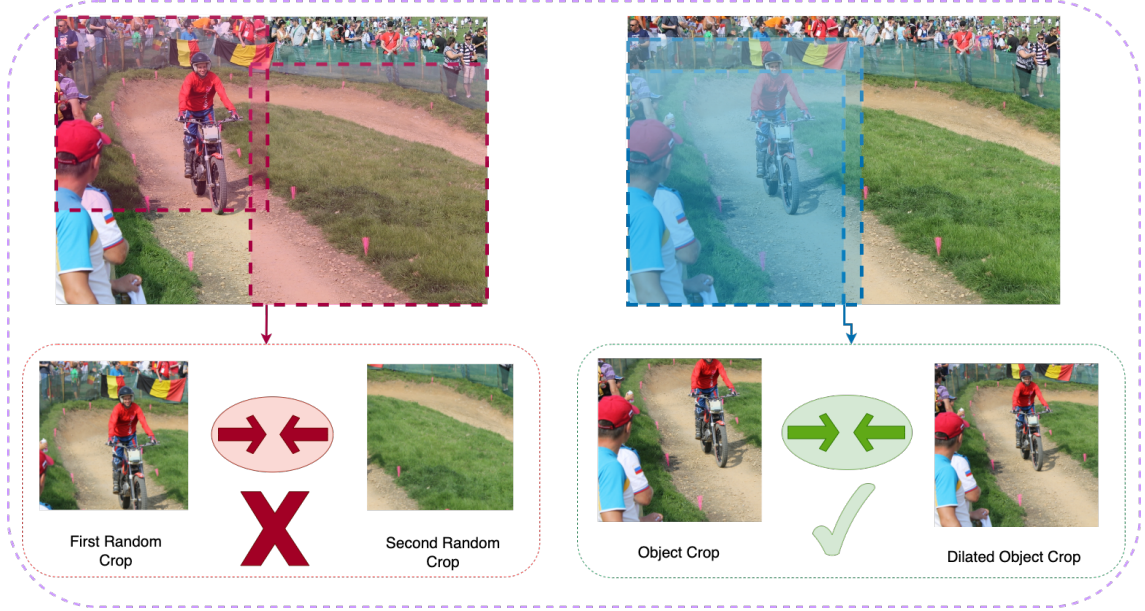


Figure 4.1: Illustration of object aware cropping. Top-Left: We show the original image with random crops overlaid. Bottom (red panel): Overlap between random crops tend to miss the object of interest. Top-Right: We show crops generated from the LOD [238] algorithm and also the dilated object crop. Bottom-Right (green panel): We use LOD-based object-aware crops. These incorporates both object and scene information into the MoCo-v2 (or other SSL frameworks).

ping reduces mutual information between the views, thereby making the pretext task harder for SSL losses, and improving final representation quality. We call this cropping approach “obj-obj+dilate” in the rest of the chapter.

In addition to an SSL loss that uses multiple views, we introduce two additional unsupervised losses which leverage the LOD proposal introduced above. The first loss, which we call object localization, encourages the network’s representations to carry information about objectness by predicting which patch features contain the object (labels are determined by the extent of the LOD proposal.

which we find to boost namely object localization and rotation prediction task. Use of these losses leads to further improvements to downstream performance.

We conduct a number of experiments incorporating object-aware cropping,

Method	Dataset	mAP
Supervised	-	53.26
InsDis([250])	ImageNet	48.82
MoCo([92])	ImageNet	50.51
PCL-v1([144])	ImageNet	53.93
PCL-v2([144])	ImageNet	53.92
MoCo-v2([40])	ImageNet	50.67
MoCHI([122])	ImageNet	52.61
MMCL([202])	ImageNet	50.73
MoCo-v2 + OAC (Ours)	ImageNet	53.96
MoCo-v2 + OAC (Ours)	OHMS	57.13

Table 4.1: We achieve superior performance on VOC detection when pre-training on our proposed **OpenImages Hard Multi-object Subset OHMS** dataset as compared to ImageNet trained models by 6.52mAP. Our proposed OAC also helps ImageNet (2nd last row), but it helps much more on the proposed OHMS multi-object dataset. ImageNet baselines has been trained for 100 epochs [202] and OpenImages model have been trained for same 100 ImageNet equivalent epochs.

finding consistent improvements on state of the art self-supervised methods such as MoCo-v2 [39], BYOL [86] and Dense-CL [245] across varied datasets and tasks. We also propose **OpenImages Hard Multi-object Subset OHMS**, which is a balanced subset of OpenImages and has at least two instances of different classes in an image. We show by pre-training on the OHMS dataset using our OAC can give superior performance on object detection task (+6.1mAP, Table 4.1) as compared to pre-training on ImageNet which has been used extensively in the literature [25, 35, 91].

4.2 Analysis of Self-Supervised Learning methods on the OpenImages Dataset

In this section, we first identify some limitations of state-of-the-art SSL methods such as MoCo-v2 [39, 91], SwAV [24] and BYOL [86] when pretraining on the Open-

Images dataset. These methods have nearly closed the performance gap with supervised learning methods when pre-trained and linear probed on ImageNet [56]. However, the performance of these methods on OpenImages dataset (where images contain multiple small objects) has not been extensively studied. OpenImages [137] encompasses images of complex scenes and several objects (containing, on average, 8 annotated objects per image). It consists of a total of 9.1 million images. To perform controlled experiments on the effectiveness of cropping on SSL method performance, we construct a subset of the OpenImages dataset called **OHMS Openimages Hard Multi-object Subset**. We construct the dataset as follows: We sample images that have labelled bounding boxes to enable comparisons with fully supervised learning. Secondly, we sample images with objects from at least 2 distinct classes to create a dataset that better reflects real-world uncured data. Finally, we only consider class categories with at least 900 images to mitigate effects of imbalanced class distribution. After this processing, we have 212,753 images present across 208 classes and approximately 12 objects per image on average. We provide further details in the Sec 4.6.

4.2.1 Performance of SSL methods

We pretrain several SSL methods MoCo-v2, CMC [219], SwAV [24] and BYOL [86] on OHMS dataset. MoCo-v2, BYOL, and other recent state of the art SSL approaches all relying on *scene-scene* cropping of the same image to generate positive samples.

Model	OHMS (mAP)	ImageNet (Top-1 %)
Supervised Performance	66.3	76.2
CMC [219]	48.7 (-17.6)	60.0 (-16.2)
BYOL [86]	50.2 (-16.1)	70.7 (-5.5)
SwAV [24]	51.3 (-15.0)	72.7 (-3.5)
MoCo-v2 (Scene-Scene crop)	49.8 (-16.5)	67.5 (-8.7)
MoCo-v2 (Object-Object+Dilate crop) (Ours)	58.6 (-7.7)	68.0 (-8.2)

Table 4.2: Classification results on OHMS and Imagenet. For each SSL method, we show in parentheses the gap to fully supervised training (same number of epochs). The last row shows that our proposed approach using *obj-obj+dilate* cropping reduces the gap on OHMS by nearly half compared to the baselines, improving over the *scene-scene* cropping based SSL methods by between 6.8 to 9.4 mAP points. We also observe improvements on ImageNet as well.

Table 4.2 shows our results. We see a significant difference in performance between fully supervised training and SSL approaches on the OHMS dataset, with a gap of 16.3 mAP points on average across the 4 SSL approaches considered. On ImageNet, the top-1 accuracy gap is considerably smaller with an average gap of only 8.5, nearly half that of OHMS. The last row shows the significant boost obtained by using our object-aware cropping approach, which we describe in the next section, with MOCO-v2. The gap between SSL and supervised training on OHMS is now the same as ImageNet.

4.2.2 Analysis and Motivation

We conduct further experiments to better analyze the results seen in Table 4.2, and to motivate our proposed approach. Our experiments help to narrow down scene cropping as one main cause of the poor performance of SSL on OHMS, rather than other differences with ImageNet, such as object size, class distributions or image

resolution.

Object Size: We compare MoCo-v2 performance to that of fully supervised learning, with both methods using scene-based cropping. Fig. 4.2 (left) shows that the performance gap between supervised learning and SSL methods does not vary significantly for objects of different sizes in OHMS. This suggests that once object sizes are below a threshold where scene cropping tends to ignore object information, MoCo-v2 performance is mostly independent of object scale.

Long-tail Distribution: Even after selecting at least 900 images per class, our OHMS subset has a significant variation in the number of images per class (from around 1000 to 60000). Fig. 4.2 (right) plots the performance of MoCo-v2 and supervised training as a function of the number of instances in each class. We do not see a significant change in relative performance as the number of instances in a class changes. This rules out long tails of the distribution as a cause for the poor absolute performance of MoCo-v2 on OHMS.

Can ImageNet pre-training help? We pre-trained a supervised model on ImageNet and then fine-tuned the final fully-connected layer on the OHMS dataset. We can see from the second column in Table 4.3 that this pre-training does not help to close the performance gap. One of the reasons that ImageNet pre-training does not help is that OHMS and ImageNet have significantly different class distributions (e.g. see [143] for a detailed analysis).

Can resizing the images help? We also experimented with resizing the images in OHMS to the same approximate size (384×384) and aspect ratio as ImageNet. The result is shown in the third column of Table 4.3 confirming that

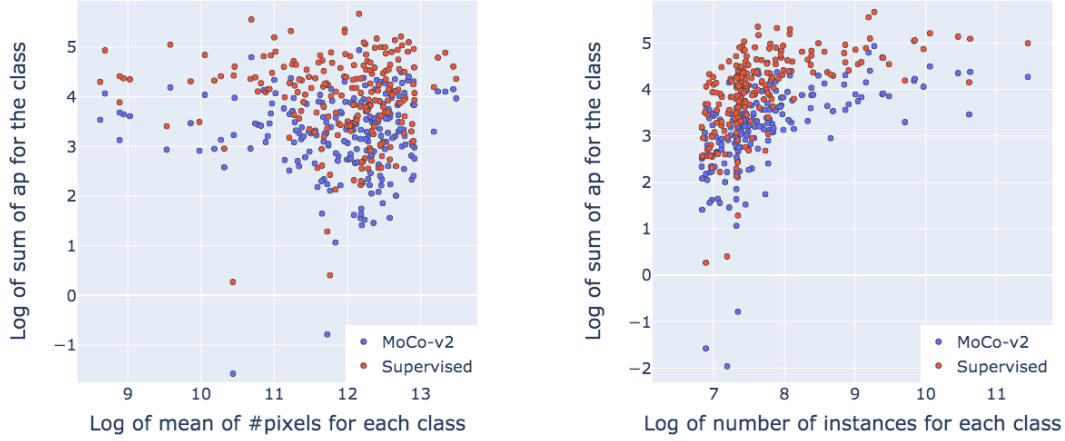


Figure 4.2: Analysis of OHMS data distribution. Left: Performance of supervised and MoCo-v2 pre-training as a function of the scale of the objects; we plot the log of the average of pixels against the sum of AP for each class. We see no discernible pattern of performance of MoCo-v2 or supervised learning as a function of object scale. Right: Performance of supervised learning and MoCo-v2 as a function of the number of instances in a class; we plot the log number of instances in a class against the sum of AP for that class. We do not see any discernible pattern of performance difference as a function of class size.

controlling for image size does not help to close the gap.

Can cropping on ground truth objects help? We see from column 4 of Table 4.3 that using random cropping on ground truth object boxes does not help reduce the performance gap either. As we show later in Section 4.3 that learning from both object and context is important for learning semantic information on multi-object datasets.

Varying the lower scale of random resized crop: MoCo-v2 ([39]) used scene crops whose size was chosen from a uniform distribution ranging from 20% to 100% of the ImageNet image size (384×384). Since OHMS images are bigger and objects generally occupy a smaller fraction compared to ImageNet, we vary the lower bound for scene crops to measure the impact. The last six columns of Table

Supervised	IN	Resize to	GT-Crop	Scene-Scene Crop					
	Pretraining	(384×384)		0.8-1.0	0.6-1.0	0.4-1.0	0.2-1.0	0.1-1.0	0.05-1.0
66.3	28.3	45.9	45.3	26.5	37.6	45.6	49.8	46.1	43.1

Table 4.3: Linear evaluation on our OHMS dataset with different pre-training strategies with MoCo-v2 (see Section 4.2 for details). Column 1 uses fully supervised learning on OHMS. We see that for self-supervised pre-training, no specific range of scene crops helps to close the large gap between SSL and supervised training. However, there is a sweet spot of scene crop range where MoCo-v2 performance is highest.

4.3 shows that varying the range of scene crop is insufficient to close the performance gap.

We conclude that the performance gap between supervised training and SSL training is likely due to the data augmentation, rather than characteristics of the image distribution. Further analysis experiments are provided in the (Sec 4.9).

4.3 Proposed Approach

In this section we first discuss how we add object-awareness to the loss, followed by two new loss functions which help in learning better object aware representations.

4.3.1 Object Proposals

To enable object-awareness, we consider an unsupervised object proposal model LOD [238]. LOD is a large scale unsupervised object proposal method [238]. The authors suggest a formulation of unsupervised object discovery as a ranking problem using distributed methods. In our experiments we use LOD to generate up to 10 proposals per image and select one object randomly among these proposals. The details of other faster semi supervised object proposals (which use object labels from

VOC dataset for training) are present in Section 4.8. Since, our method is completely Self-Supervised we focus on using LOD as our object proposal generation method.

Dilated object proposals (Obj-Obj+Dilate Crop): Scene pixels spatially close to the object are more likely to have a positive correlation with the object. With this intuition, we generate the second view by dilating the randomly selected LOD proposal. We dilate the box by 10% or 20% of the image size, followed by a random crop. Changing dilation gives us control over how much scene information is incorporated (a value of 10% works well in most cases). Note that the original and dilated boxes are both followed by a random crop, ensuring that the first view is not trivially included in the second view. The choice of which crop to use as query or key is arbitrary and either object or dilated object crops can be used as key and query.

Different projection heads for Object and Dilated Object crops: The projection head, introduced in [35] is an important component of most SSL methods. This is an MLP that maps representations from the encoder backbone to a lower-dimensional space where the loss function is applied. SimCLR [35] show that projection heads can remove information that may be useful for the downstream task, such as the color or orientation of objects. They show that by using this MLP, the last layer of network (i.e layer before mlp) can maintain more information. However [40] use a single projection head is used for both views, since both the views on ImageNet are object crops. We hypothesize that Object and Scene crops often contain different

object orientation and color information; hence we propose to use different projection heads for scene and objects.

4.3.2 Loss Objectives

Following [91], we use a momentum queue and optimize the model using the InfoNCE loss :

$$\mathcal{L}_{\text{moco}} = -\log \frac{\exp(q \cdot k^+ / \tau)}{\exp(q \cdot k^+ / \tau) + \sum_{k^-} \exp(q \cdot k^- / \tau)}. \quad (4.1)$$

Here q is a query representation, k^+ is a representation of the positive (similar) key sample, and $\{k^-\}$ are representations of the negative (dissimilar) key samples. τ is the temperature hyper-parameter. We augment the standard contrastive loss with two additional losses to learn richer features for both objects and scenes. Next we describe both these losses in detail.

4.3.2.1 Rotation task $\mathcal{L}_{\text{rot}}$

In the standard rotation prediction pretext task [81], an image crop is randomly rotated with an angle in set ϕ . An MLP is then tasked at correctly predicting the rotation of a patch given its features extracted using the feature extractor f . Our object cropping strategy generates an object and a scene crop from a given image. We modify the standard rotation task to estimate rotation of the object crop wrt the scene crop. Specifically, we randomly rotate the object crop and extract features z_{obj} . The scene crop is kept as is and its features z_{scene} are obtained by feeding

the scene crop through feature extractor f . Note that rotation augmentations are applied to object crop in addition to the standard MoCo-style augmentations. The rotation prediction MLP takes as input the concatenated features and estimates the relative rotation. Our rotation loss is $\mathcal{L}_{\text{rot}}$ is a standard cross entropy loss with rotation labels as the targets. Different from the standard rotation prediction task which estimates absolute rotation, our approach estimates rotation of the object relative to the scene and complements the contrastive loss. Note that since we deal with in-the-wild datasets, estimating absolute rotation of a crop is ill-posed since a particular object might occur in a variety of poses thus leading to incorrect gradients.

4.3.2.2 Object localization task $\mathcal{L}_{\text{loc}}$

Since we are working with images coming from non-iconic datasets, the object could potentially occupy a small region of the image. We propose to add a pretext task of localizing an object inside the scene using features alone. Specifically in this task, given an image, we predict the spatial location of the object in the image. Here we take original image and divide the image into $p * p$ patches. We then take an object crop and mark all the patches where the object is present as 1 and other patches as 0 which we use as our label Y . Similar to rotation task we first extract features for the object proposals z_{obj} . For the original image we apply all the same augmentations except random crop (since we want the full image and not the cropped version of the image). Then we obtain the features for the original image z_{ori} . We then pass the features through MLP layer. Finally the loss $\mathcal{L}_{\text{ol}}$ is two-class classification loss

on concatenation of object features, original image features and labels Y .

Why don't we see degenerate solution for object localization? A natural question is why doesn't our model cheat and remember all the object location for all the object proposals. The answer to the question is the random cropping on the object proposals. Since we are using random crops on the object proposals, the label Y will change every time according the random crop parameters. Hence the network cannot learn the object patch location and has to focus on the semantics of objects and scenes to figure out the spatial location of object in the scene.

Final loss function: Our final loss function is

$$\mathcal{L}_{\text{oac}} = \mathcal{L}_{\text{moco}} + \mathcal{L}_{\text{rot}} + \mathcal{L}_{\text{ol}} \quad (4.2)$$

We simply combine all the losses and call it $\mathcal{L}_{\text{oac}}$ (**Object-Aware Cropping**) and backpropagate on all of them.

4.4 Results

We created a subset of the OpenImages dataset with 212k images, as described in Section 4.2. We also experiment with the complete OpenImages (~ 1.9 million images.) In addition, we perform pre-training on ImageNet [56] and MS-COCO [148]. ImageNet (with 1.2M training images) has been extensively used and is the standard dataset used for benchmarking of SSL methods. MS-COCO has ~ 118 k training images and 896k labelled objects which is approximately 7 objects per image.

Model	Crops	Obj-Obj+Dilate	Scene-Scene	mAP
Supervised	-	-	-	66.3
MoCo-v2	-	-	✓	49.8
BYOL	-	-	✓	50.2
MoCo-v2	Ground Truth boxes	-	-	58.9
MoCo-v2	Ground Truth boxes	✓	-	60.2
MoCo-v2	LOD crops	✓	-	59.1
BYOL	LOD crops	✓	-	59.5

Table 4.4: Crop approaches on OHMS: using LOD crops to generate one view, and a dilated crop for the other positive, we are able to reduce the difference between SSL and Supervised Learning by close to 50% (compare the last two rows to the first row). Using ground-truth boxes to generate crops from OHMS improves the pre-training performance marginally compared to LOD crops. Our Obj-Obj+Dilate outperforms the Scene-Scene baseline by significant margin on the OHMS dataset.

For pre-training on MoCo-v2, we closely follow the standard protocol described in chen2020improved. We randomly select from 10 object proposals provided by LOD. All our training and evaluation is performed on a ResNet-50 [94]. For our baseline we use standard scene-scene crop, where we take two random crops in an image and treat them as positive views. This is the default approach used in MoCo-v2 and other SSL approaches. From our analysis in Section 4.2, this approach performs poorly on datasets such as OHMS.

As discussed, the Obj-Obj+Dilate uses a random crop on the object proposal or dilated version, to generate the final views. Since the object proposal itself is a small fraction of the image (e.g. in COCO, an object crop typically covers about 39% of the image), using the usual default lower value for the random crop range (usually 0.2) works poorly as it results in extremely small crops from the image. Therefore, we set the lower limit such that it matches the minimum sized crop in case of the usual scene crop ($s_{\min} = \frac{0.2}{\text{average Object proposal size}}$).

Description	AP	AP_{50}	AP_{75}	AP_s	AP_l	AP_m
Supervised (Random Initialization)	32.8	50.9	35.3	29.9	47.9	32.0
Supervised (ImageNet Pre-trained)	39.7	59.5	43.3	35.9	56.6	38.6
MoCo-v2 [39]	38.2	58.9	41.6	34.8	55.3	37.8
BYOL [112]	38.8	58.5	42.2	35.0	55.9	38.1
Dense-CL [245]	39.6	59.3	43.3	35.7	56.5	38.4
CAST [201] (180K steps)	39.4	60.0	42.8	35.8	57.1	37.6
Self-EMD [152] (Uses BYOL)	39.8	60.0	43.4	-	-	-
MoCo-v2 + OAC (Ours)	40.7	60.9	43.9	36.9	58.3	39.6
BYOL + OAC (Ours)	41.1	61.4	44.2	37.1	59.2	40.1
Dense-CL + OAC (Ours)	41.4	61.5	44.7	37.5	59.5	40.4

Table 4.5: Object detection (first 3 columns) and Semantic Segmentation (last 3 columns) results on COCO dataset. All SSL models have been pre-trained on COCO dataset and then finetuned on COCO. All other methods are run for 90K, finetuning iterations. For any SSL method, we compare (BYOL, Moco-v2, Dense-CL) adding **Object-Aware-Cropping (OAC)** crop improves performance.

We evaluate the pre-trained models on classification (linear evaluation), object detection and semantic segmentation. For VOC object detection, COCO object detection and COCO semantic segmentation, we closely follow the common protocols listed in Detectron2 [249]. For VOC object detection, we evaluate on the Faster-RCNN(C4-backbone) [190] detector on VOC `trainval07+12` dataset using the standard $1 \times$ standard protocol. For COCO-Object detection and semantic segmentation, we fine tune on the MaskRCNN detector (FPN-backbone) [93] on COCO `train2017` split (118k images) with the standard $1 \times$ schedule, evaluating on the COCO 5k `val2017` split. We compare to the state of the art SSL methods, including Self-EMD [152], DetCon [112], BYOL [191], DenseCL [245] and the default MoCo-v2 [39].

Table 4.2 and Table 4.4 shows results on OHMS dataset. We can see that Obj+Dilate crops outperform the baseline by 8.2 mAP, closing the gap between su-

Model	AP	AP_{50}	AP_{75}
Supervised (ImageNet-pretraining)	56.8	83.2	63.7
MoCo-v2 Scene-Scene crop chen2020improved	51.5	79.4	56.4
MoCo-v2 - Obj-Obj+Dilate crop ($\delta = 0.1$) (Ours)	53.4	80.1	59.1
MoCo-v2 - Obj-Obj+Dilate crop + Rotation ($\delta = 0.1$) (Ours)	53.8	79.6	60.1
MoCo-v2 - Obj-Obj+Dilate crop + Object Localization ($\delta = 0.1$) (Ours)	54.1	80.6	60.2
MoCo-v2 - Combined ($\delta = 0.1$) OAC (Ours)	54.6	81.0	60.6

Table 4.6: Object detection results on VOC dataset (OHMS pre-training, using LOD proposals). All models have been pre-trained on OHMS and then fine-tuned on VOC. We can see that all of our proposed components improve upon the baseline scene scene crop and combining all of them improves upon the baseline by 3.1 mAP.

Description	AP	AP_{50}	AP_{75}	AP_s	AP_l	AP_m
COCO: MoCo-v2 (Scene-Scene crop)	38.2	58.9	41.6	34.8	55.3	37.8
COCO: MoCo-v2 + OAC (Ours)	41.3	61.8	44.7	37.3	58.5	40.1
VOC: MoCo-v2 (Scene-Scene crop)	56.1	81.3	61.3	-	-	-
VOC: MoCo-v2 + OAC (Ours)	58.8	83.6	64.9	-	-	-

Table 4.7: Object detection (first 3 columns) and semantic segmentation (last 3 columns) results on COCO (first 2 rows) and VOC (last 2 rows). All SSL models have been pre-trained on complete OpenImages dataset(1.9 million images) for 75 epochs and then finetuned on COCO and VOC dataset.

pervised learning and MoCo-v2 baseline by almost 50%. In Obj-Obj+Dilate crops, a dilated object crop would potentially contain the entire object and more scene information; therefore, the representation from the dilated object-crop contains complementary information from both the object and the scene. We also show in Table 4.4 an ablation with ground truth bounding boxes being used to guide the object cropping. This performs marginally better than the use of LOD crop, suggesting that a tight fit around the object is not necessary for improved representations.

Table 4.5 shows results on object detection and semantic segmentation for COCO (by pre-training on COCO `trainval2017` datasets and finetuning on COCO). We train MoCo-v2, BYOL and Dense-CL models. Our MoCo-v2 Obj-Obj+Dilate cropping outperforms MoCo-v2 Scene-Scene baseline. Our proposed cropping is agnostic to the pre-training SSL method; we show results by adding our approach to Dense-CL [245]. We also outperform the CAST model [201] which also uses localized crops based on saliency maps: our approach is simpler and performs better by around 1.4 mAP. Table 4.8 (supplementary) shows results of object detection on PASCAL-VOC. We pre-train on COCO and then fine-tune on VOC. OAC cropping outperforms the MoCo-v2 baseline by 3.2 mAP and the BYOL baseline by 2.5 mAP. Improved results on iconic datasets like Aircraft, Birds and Cars can be found in Supplementary (Table 4.12). Additional results on varying number of proposals used can be found in Supplementary (Table 4.10).

Table 4.7 shows results on object detection and semantic segmentation for COCO and object-detection on VOC by pre-training on full OpenImages dataset [137] (all 1.9 million images) for 75 epochs. We show improved performance over the

baseline on both object detection and semantic segmentation tasks by using Obj-Obj+Dilate crops. Our proposed dilation method works not only for small multi-object datasets like COCO but also for datasets like OpenImages and performs well under a transfer learning setup.

Results on ImageNet: We also show improved results on ImageNet pre-training using object-aware cropping (Table 4.15 Supplementary) and MoCo-v2. For object detection on VOC2007, we see an improvement of 1.0 mAP; and a 0.5 mAP improvements for object detection on COCO. Our approach is thus adaptable to the pre-training dataset and SSL algorithm.

Use of Multiple Projection Heads: The use of different projection heads for each view on OpenImages classification gives us a boost of 1.1 mAP on Obj-Obj+Dilate crop. Pre-training on COCO and finetuning on VOC dataset for object-detection task gives a boost of 0.4 mAP. Hence using multiple projection heads results in a consistent improvement.

Varying Dilation Parameter: Table 4.11 shows the effect of varying the dilation parameter. A sweet spot exists at a moderate dilation value of $\delta = 0.1$ for COCO object detection.

4.5 Related Work

Recent progress in self-supervised learning, based on contrastive and non-contrastive approaches, has achieved excellent performance on various domains, datasets and tasks [35, 39, 80, 81, 86, 91, 138, 170, 179, 184, 219, 221, 230, 251]. The top-performing

methods have all used related ideas of pulling together “views” of a sample in representation space. Some of these approaches, in addition, use negative samples to add a “push” factor, and this is termed contrastive self-supervised learning. Theoretical and empirical studies have been published to better understand the behavior and limitations of these approaches [1, 19, 44, 122, 154, 174, 187, 227, 243, 253, 261].

A number of papers have observed that the default SSL approaches above (whether contrastive or not) perform poorly on uncured datasets such as Open-Images [137]. To address this, recent works have used different workarounds such as knowledge distillation [218], clustering [82], localization [201], unsupervised semantic segmentation masks [112], pixel-level pretext tasks [255], and local contrastive learning [152]. The common element among top-performing image-based SSL approaches, regardless of dataset, task or architecture, is their reliance on strong data augmentations such as random cropping, gaussian blurring, color jittering or rotations. These augmentations create meaningful positive views, and other randomly sampled images in the dataset are used to create negative views in the case of contrastive SSL methods. SSL data augmentation pipelines are adapted from the supervised learning literature [51, 51, 52, 60, 89, 135, 147, 207, 249, 264, 267, 272].

The closest work to our object cropping work is [201], which introduces a technique to choose crops around objects based on saliency maps [199], showing good improvements over the baseline of random crops for COCO datasets (see Table 4.5). As shown in our results, Obj-Obj+Dilate crop consistently performs better than [201] (Table 4.5). Our approach is also significantly simpler to incorporate into existing pipelines, requiring no change to the training, architecture or loss. van-

gansbeke2021revisiting show that constrained multi-cropping improves performance of SSL methods: our approach can be incorporated into their pipeline to further improve performance.

4.6 Details of OpenImages-Subset

We show in Fig. 4.3(left) the plot of the number of images and selected class ID's. The mean of the number of images per class is 3762.13. We show additional statistics by removing the classes with more than 10k instances present. We can see this in Fig. 4.3(middle). The mean of number of images per class also comes down from 3762 to 2203. We plot the width and height of all the images present in OpenImages-Subset. We can see from Fig. 4.3(right) that most images are in around 1000×1000 pixels. with some being significantly larger.

We created OpenImages-Subset by only keeping classes from OpenImages which had at least 900 images, resulting in a total of 208 classes with 3762 images per class on average. For each image in OpenImages-Subset we computed how many images have an extra class that is not present in these 208 classes: resulting in a very small number 0.002: mist objects in the subset are from the 208 classes. The total number of images is 212k. Finally we computed the number of images with N objects, where N varies from 2 to 7. These numbers are: 2/51008; 3/56131; 4/43426; 5/26011; 6/13693; 7/6593.

4.7 Implementation details for Object localization task

Let's assume the output of the object proposal method is $X_{o_1}, Y_{o_1}, X_{o_2}, Y_{o_2}$. We firstly feed this object image through random crop operation. The output of the random crop is denoted by $X_{r_1}, Y_{r_1}, X_{r_2}, Y_{r_2}$.

$$X_{f_1} = \frac{W_f}{W_b}(X_{o_1} + X_{r_1}), \quad X_{f_2} = \frac{W_f}{W_b}(X_{o_2} + X_{r_2} + W_b/2) \quad (4.3)$$

$$Y_{f_1} = \frac{H_f}{H_b}(Y_{o_1} + Y_{r_1}) \quad , \quad Y_{f_2} = \frac{Y_f}{Y_b}(Y_{o_2} + Y_{r_2} + Y_b/2) \quad (4.4)$$

$\frac{W_f}{W_b}$ and $\frac{Y_f}{Y_b}$ are used to scale the cropped image back to the original scale. So, the new location of the object wrt the original image would be $X_{f_1}, Y_{f_1}, X_{f_2}, Y_{f_2}$. The labels will be set by following condition :

$$Y_{ij} = 1 \iff X_{f_1} < X_{ij} < X_{f_2} \ \& \ Y_{f_1} < Y_{ij} < Y_{f_2}. \quad (4.5)$$

$$i, j \in [0, 1 * w/p, 2 * w/p \dots p * w/p]. \quad (4.6)$$

4.8 Other object proposal methods

We also experiment with three object proposal methods, i.e BING [42], Edge-Boxes [271] and an unsupervised object proposal method [237]. Here we discuss their methodology in more detail.

BING is trained with boxes from Pascal-VOC. BING [42] uses the norm of

gradients within a fixed window size as a simple feature that is fed into a cascaded SVM framework to make object proposals. This method is trained on the ground-truth bounding boxes of 2501 images of the PASCAL-VOC [68] training set. Note that BING does not use any class label, but only label independent bounding box information. BING has the advantage of extremely fast performance (125 fps on an image of size 300×300), generates many proposals, and generalizes well to many datasets, as shown in [42] and seen in our empirical results. This makes BING well suited as an object proposal algorithm.

Edge-Boxes [271] is based on a simple heuristic: an object is more likely to be contained in a box if the number of contours wholly inside the box exceed the contours that cross the boundary of the box. In our experiments, we found that EdgeBoxes was slower than BING, although the performance on the generated boxes was similar to that of BING.

We also consider an alternative unsupervised object proposal method [237], which uses a robust matching technique that relies on appearance and geometric consistency constraints to assign confidence scores to region proposals. While fully unsupervised, this approach performs similar to BING as shown in Table 4.4 and Table 4.8, but is significantly slower than BING. Our results therefore suggest that many object proposal methods provide sufficient quality and the choice may be mostly dependent on the speed of the method.

(c) Shifted object proposals (Obj-Obj+Shift Crop): Unlike Obj-Obj+Dilate, here, the second view is a box selected at random within a pre-specified distance range of the first box (the BING proposal). To choose a pre-specified distance, we

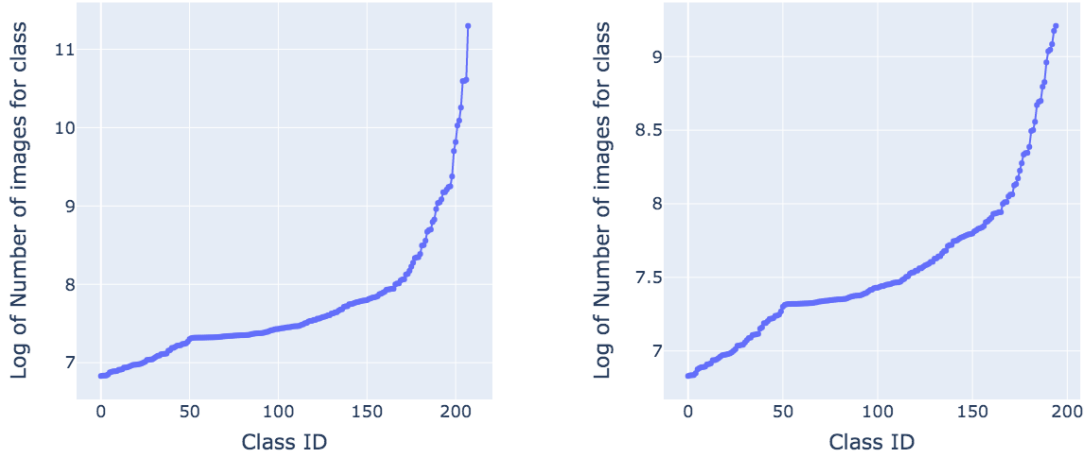


Figure 4.3: **Left:** distribution of number (log) of images in each class for each of 208 selected classes. **Middle:** distribution after removing classes which have more than 10k images. **Right:** scatter plot of the height and width.

choose a random value for the offset from a few ranges: 80-100 pixels; 100-120 pixels etc.

(d) Random crop (Obj-Scene Crop): The first view is the BING proposal; and a regular random crop at scene level is the second view. This method provides information at both object and scene levels, and we use it as a baseline for adding context information to the model.

4.9 More Analysis on OpenImages

Varying the temperature parameter: Some self-supervised methods are highly dependent on the temperature hyperparameter, which can be dataset dependent. The default temp parameter in MoCo-v2 [39] (tested on ImageNet) is 0.07. To ensure that this temperature setting was not a concern for OpenImages, we trained MoCo-

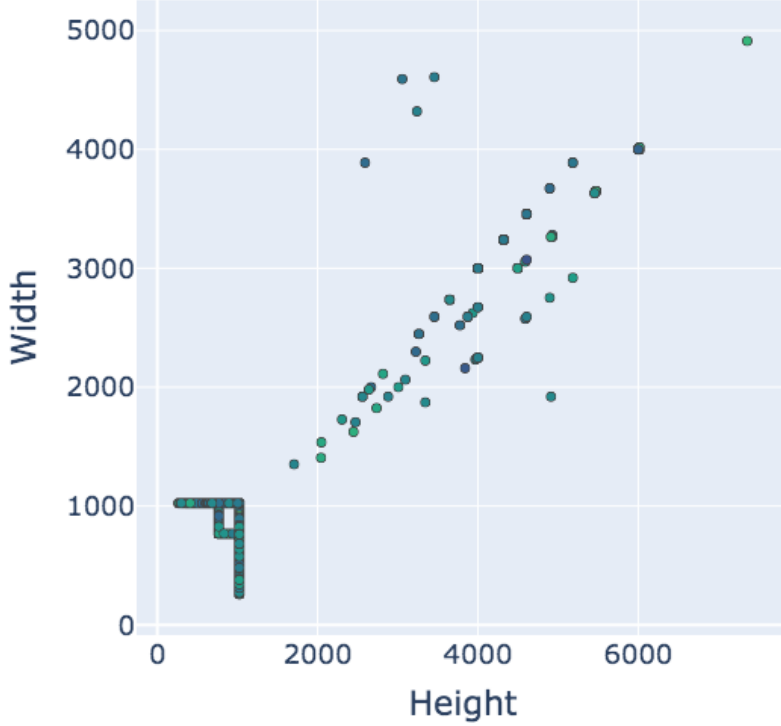


Figure 4.4: Scatter plot of the height and width of the OpenImages Dataset..

v2 models with a number of temperature hyperparameters i.e 0.05, 0.7, 0.1, 0.2 and 0.3. We found that a temperature setting of 0.2 gives the best result which is an increase of 1.2 mAP over the default value, which is significantly lower than the increase of 8mAP we get with object cropping.

MoCo-v2 training dynamics: Another potential concern with OpenImages is poor training dynamics of MoCo-v2 on OpenInmages. To verify that this was not the case, we computed the mean inner product positive samples and of negative samples after training MoCo-v2 for 200 epochs. We find that the mean of the score for positive samples is 0.81 and the mean of scores of negative samples is 0.005,

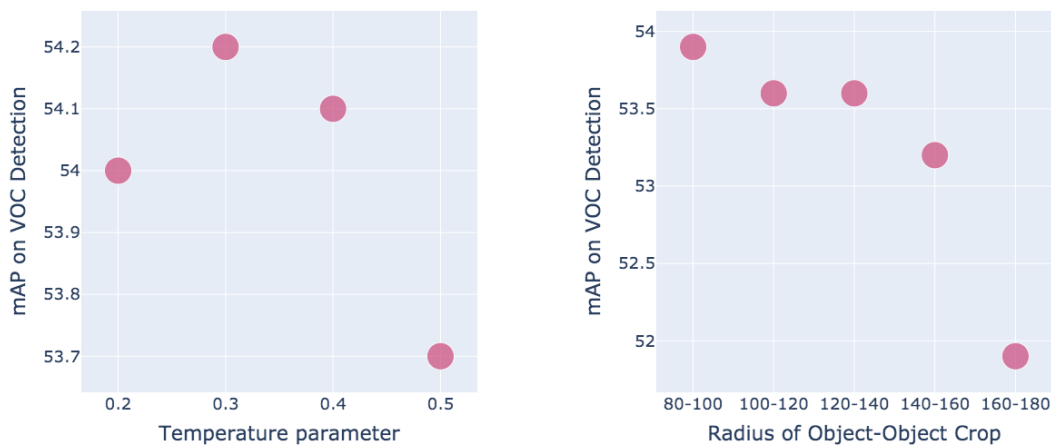


Figure 4.5: Results on COCO then transferred to VOC. **Left:** Varying the temperature parameter. **Right:** Varying radius for object-object crop.

very close to the ideal of 1 and 0 respectively. This indicates that MoCo-v2 trains properly, and the difference between the MoCo-v2 and supervised training is unlikely to be due to poor training dynamics.

4.10 Analysis on Object-Scene crops:

Overlap between crops and object of interest: In expectation, any two crops of a scene will overlap to some extent. If this intersection overlaps an object, the crops are more likely to contribute to representations that capture object properties. We measure the fraction of pixels in this intersection that belong to a ground truth object box, to get a sense of how this varies with different cropping strategies and data sets. We find that this fraction for COCO is 99% for object-scene crops and 92.1% for the scene-scene crop. In the case of OpenImages, the numbers are, respectively, 99.1% and 87.3%. This demonstrates that object-scene crops capture information about

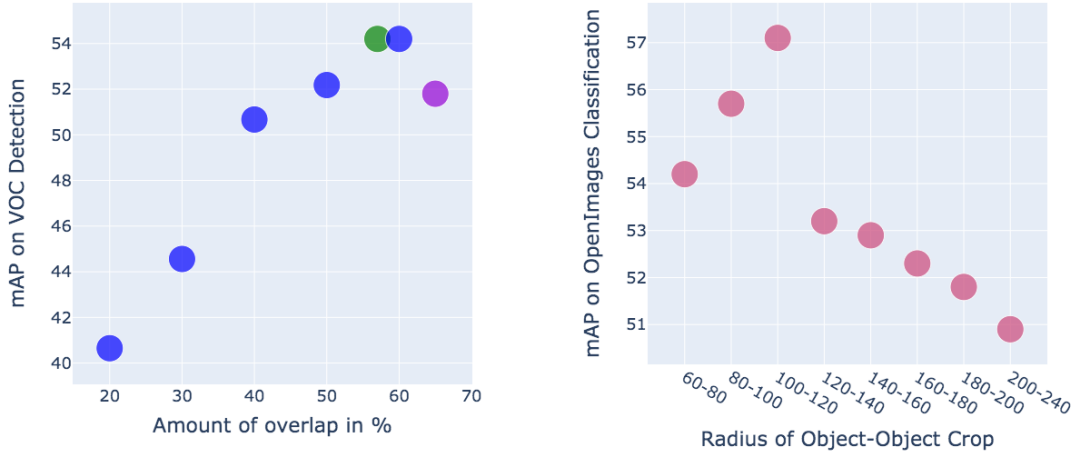


Figure 4.6: Left: We vary the amount of overlap between object and scene crops, as shown. Optimal VOC detection performance is achieved around 58% overlap, decreasing on either side. The purple dot indicates overlap for scene-scene crops (66%). Right: We vary the radius of the object-object crops and observe a similar phenomenon of a “sweet spot” distance between the object crops at which downstream performance is maximized.

objects much more than scene-scene crops, especially in the case of OpenImages.

Varying the radius in OpenImages and COCO: We also vary the radius, which is the distance of the second object-object crop in any random direction from the center of the object crop. We vary this radius under object-object crop when combining all three type of crops. Fig 4.6(right) shows that, for OpenImages, as we increase the distance of the two object crops, performance first increases and then decreases, again suggesting a sweet spot. We report similar results for COCO dataset in the supplementary.

Use of scene information: To test whether scene information is used by our models, we pre-train two models on the OpenImages dataset: one with object-object cropping and the other with object-scene cropping. Next, we create a new dataset by cropping ground truth bounding boxes from the OpenImages images.

Each resulting image contains exactly one object and minimal scene information. We fine-tune the two models on this dataset, and we find that they perform comparably (object-object fine-tuned gives 37 mAP and object-scene fine-tuned gives 36.4 mAP). However if we fine-tuned instead on the OpenImages dataset, the object-scene model outperforms by close to 4 mAP, thus verifying that object-scene model cropping learns features that rely on both object and scene information.

Varying radius: Fig. 4.5(right) shows the performance on VOC (pre-training on COCO) for MoCo-v2 as the radius object-object crops is changed. We see that there is a “sweet spot” for the radius: neither too large or too small is optimal.

4.11 Additional Results

Generating positive samples by randomly cropping on Dilated BING boxes: We also try to generate both the positive samples by randomly cropping on the Dilated BING boxes. We find that this method gives us a performance of 54.5 mAP for OpenImages classification. Hence cropping on object boxes and dilated crops are crucial to success of our method.

Varying temperature parameter: We also varied the temperature hyperparameter for training MoCo-v2 on COCO (transferring to VOC) as seen in Fig. 4.5(left). We find that temperature of 0.3 performs best in our setting on the COCO dataset.

Varying number of proposals used: We vary the number of proposals

Model	AP	AP_{50}	AP_{75}
Supervised	56.8	83.2	63.7
BYOL Scene-Scene crop [86]	50.0	75.8	54.8
MoCo-v2 Scene-Scene crop [39]	51.8	77.6	55.4
BYOL Obj-Scene crop using BING crops (Ours)	52.5	77.1	58.1
MoCo-v2 Obj-Scene crop using Unsupervised crops (Ours)	53.9	79.8	59.8
MoCo-v2 Obj-Scene crop using BING crops (Ours)	54.2	80.1	60.0
MoCo-v2 - Obj-Obj+Dilate crop ($\delta = 0.1$) (Ours)	55.0	80.9	60.7
Dene-CL [245]	56.7	82.5	63.8
Dense-CL with Obj-Obj+Dilate crop (Ours)	57.6	82.5	63.8

Table 4.8: Object detection results on VOC dataset (COCO pre-training). All models have been pre-trained on COCO and then fine-tuned on VOC. For both MoCo-v2 and BYOL, replacing the default scene crops with object-scene crops results in a consistent improvement.

Model	AP	AP_{50}	AP_{75}
Supervised	56.8	83.2	63.7
Baseline (Scene-Scene) [39]	51.5	79.4	56.1
Ours (Obj-Scene) [39]	53.0	79.3	58.3

Table 4.9: Object detection results on VOC dataset (OpenImages pre-training). All models have been pre-trained on OpenImages and then fine-tuned on VOC. Replacing the default scene crops with Obj-Scene crops results in a consistent improvement.

from BING as shown in Table 4.10. We find that using more proposals results in better performance and we tested upto 10 proposals per image, finding a consistent boost.

Impact of varying scene crop range in the object-scene crop: We show the effect of varying scene crop range in object-scene crop in Table 4.13. We can see that as we again reach a sweet spot when the lower crop range is 0.2.

Results after 200 epochs training: We show results after training 200 epochs of training in Table 5.2. We compare these results with 800 epochs and

Model	Proposals	AP	AP_{50}	AP_{75}
Supervised	-	56.8	83.2	63.7
MoCo-v2 Obj-Scene crop [39]	1	51.9	77.6	55.4
MoCo-v2 Obj-Scene crop [39]	5	53.5	79.4	56.4
MoCo-v2 Obj-Scene crops using BING crop (Ours)	10	54.2	80.1	60.0

Table 4.10: Varying the number of proposals generated by BING. All models are pre-trained on COCO and then fine-tuned on VOC Object Detection. Increasing the number of proposals provides a consistent boost. We used 10 proposals as this was close to the average number of objects per image in OpenImages.

Model	δ	AP	AP_{50}	AP_{75}
MoCo-v2	0	54.3	77.1	54.84
MoCo-v2	0.1	55.1	81.0	60.7
MoCo-v2	0.2	54.5	80.7	60.3
MoCo-v2	0.3	54.2	80.2	60.1

Table 4.11: Varying dilation i.e δ on the COCO dataset for Obj-Obj+Dilate crop strategy. All models have been pre-trained on COCO and then fine-tuned on VOC Object Detection. $\delta = 0$ corresponds to Obj-Obj cropping and larger δ is very similar to Obj-Scene. A sweet spot exists between the extremes.

Dataset	Obj-Scene (Top-1)	Obj-Obj+Dilate (Top-1)
Aircraft	87.8	88.7
Stanford Cars	89.1	90.7
Caltech-UCSD Birds	87.2	88.8

Table 4.12: Obj-Obj+Dilate crop pre-training consistently outperforms Obj-Scene for transfer to various downstream datasets for classification (COCO pre-training).

Model	mAP (Classification)
Supervised OpenImages	66.3
MoCo-v2: Scene Crop Range (Object-Scene) 0.8-1.0	34.2
MoCo-v2: Scene Crop Range (Object-Scene) 0.6-1.0	46.1
MoCo-v2: Scene Crop Range (Object-Scene) 0.4-1.0	54.3
MoCo-v2: Scene Crop Range (Object-Scene) 0.2-1.0	58.1
MoCo-v2: Scene Crop Range (Object-Scene) 0.1-1.0	56.3
MoCo-v2: Scene Crop Range (Object-Scene) 0.05-1.0	53.5

Table 4.13: Impact of varying scene crop range in the object-scene crop.

Description	epochs	AP	AP_{50}	AP_{75}	AP_s	AP_l	AP_m
MoCo-v2 Scene-Scene crop	200	34.6	53.5	37.0	30.4	50.1	32.3
MoCo-v2 Obj-scene crop (Ours)	200	35.6	54.3	38.3	31.3	51.2	33.4
MoCo-v2	800	38.2	58.9	41.6	34.8	55.3	37.8
MoCo-v2 Obj-Scene crop (Ours)	800	39.4	59.8	42.9	35.8	57.8	38.7

Table 4.14: Results after pre-training for 200 epochs. These results are highly indicative of results we get after longer training and can be useful for comparison in less compute settings.

Description	AP	AP_{50}	AP_{75}	AP_s	AP_l	AP_m
MoCo-v2	39.8	59.8	43.6	36.1	56.9	38.7
MoCo-v2 (Obj-Obj+Dilate $\delta = 0$) (Ours)	40.2	60.6	43.6	36.4	57.4	39.0
MoCo-v2	57.0	82.2	63.4	-	-	-
MoCo-v2 (Obj-Obj+Dilate $\delta = 0$)	57.8	82.9	64.2	-	-	-

Table 4.15: Object detection results on COCO (top 2 rows) and VOC (bottom 2 rows). All SSL models have been pre-trained on ImageNet for 200 epochs and then fine-tuned on COCO and VOC.

we can see their a good co-relation between results after 200 epochs and after 800 epochs. This could be useful in low compute regime settings, since training 200 epochs is highly indicative of performance after longer training schedules.

4.12 Visualizations

We show additional examples of object crops and scene (random) crops in Figures 4.7, 4.8, 4.9, 4.10, 4.11. We can see in all these images random scene crops often miss out objects of interest while BING tends to capture them. We also show visual comparison between OpenImages and ImageNet as shown in Fig 4.12. We can see that ImageNet images are centered and mostly occupy the center part of the image, which is not the case with OpenImages.

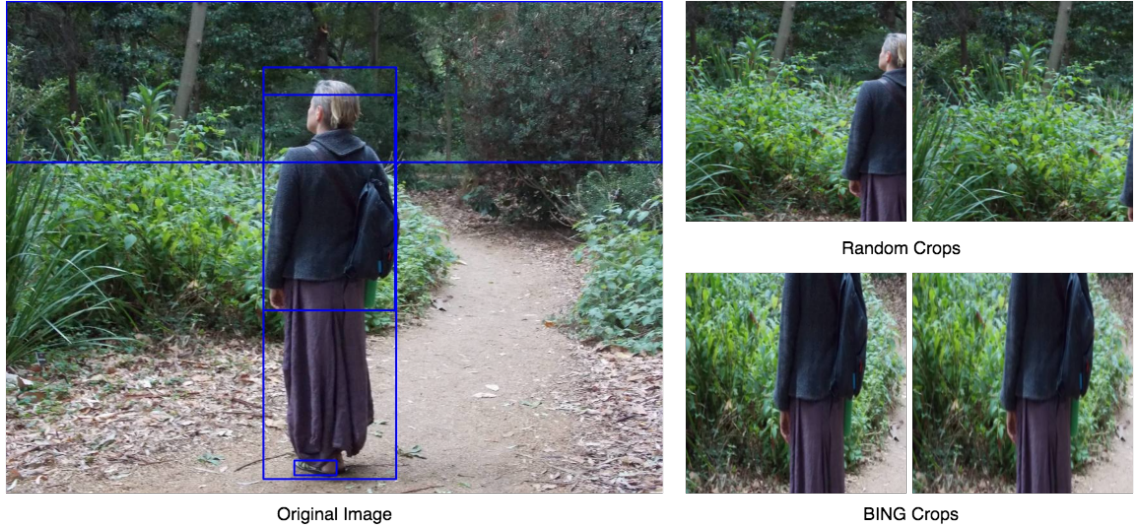


Figure 4.7: Visualization of BING and random crops.



Figure 4.8: Visualization of BING and random crops.

4.13 Implementation Details

We use a ResNet-50 [94] deep network as the backbone for all experiments. After pre-training, we freeze the backbone weights, and train a linear classifier or fine-tune on the downstream datasets and tasks. We also train a randomly initialized ResNet-50 network in a fully supervised manner using ground truth labels and a



Figure 4.9: Visualization of BING and random crops.

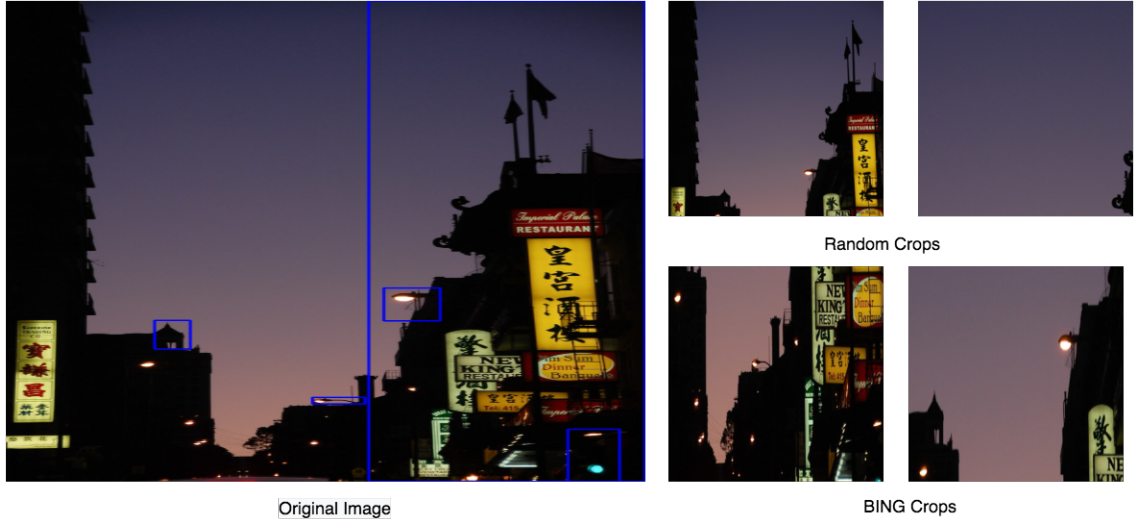


Figure 4.10: Visualization of BING and random crops.

multi-class logistic regression loss. We follow the mAP metric as described in section 4.2 of paper [234] (Eqn.(4)).

One additional change that we make to object crops is if the size of the object proposal is small, i.e less than 224×224 , we use a larger crop from the original image so that the size of the crop is at least 224×224 . To achieve this, we pick the center of the object-crop in the original image and cut a center crop of size 224×224 from

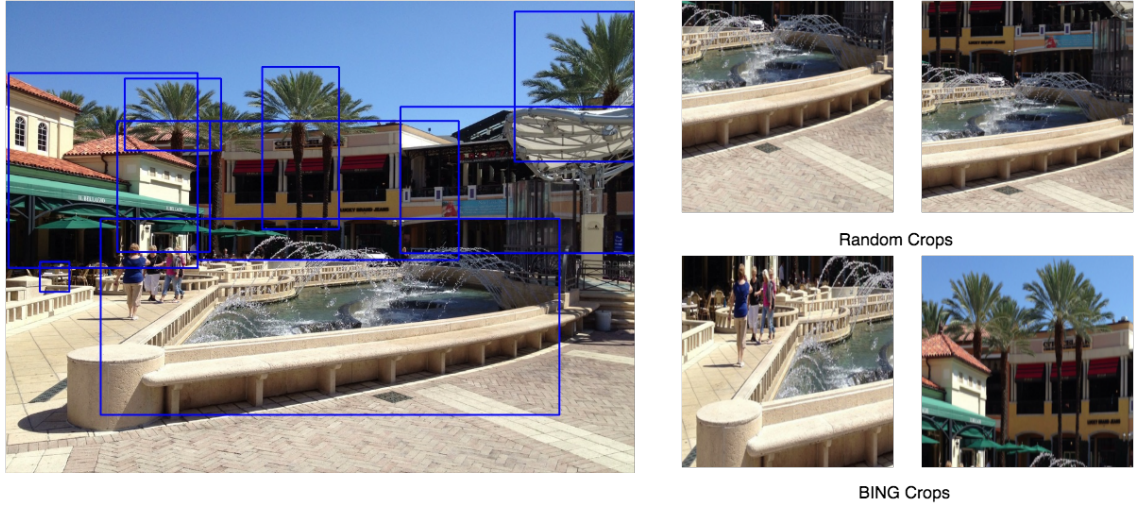


Figure 4.11: Visualization of BING and random crops.

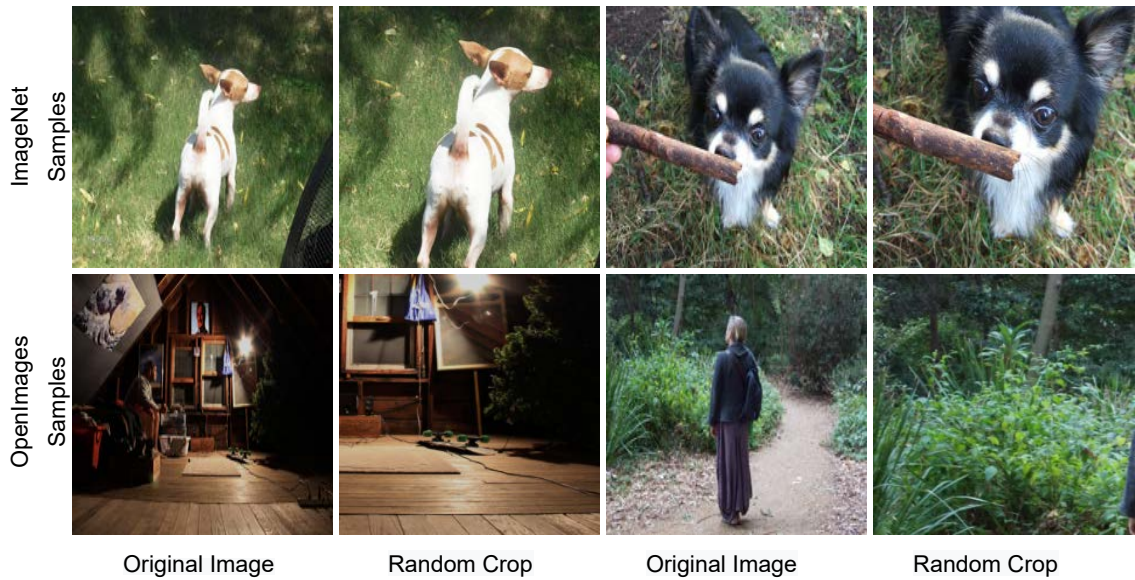


Figure 4.12: First column: Two samples from OpenImages dataset. Second and Third Columns: Random scene crops; Fourth and fifth columns: Random object crops generated by the BING [42] algorithm. We see that the object crop tends to center on objects in the image that are often missed by the scene crops.

original object-crop center. Hence the object-crops for images smaller than 224×224 are not tight bounding boxes but bigger crops with objects in it. Additionally we find that $\delta = 0.1$ works well for COCO and $\delta = 0.2$ works well for OpenImages.

4.13.1 Hyperparameters for different methods

We train all the SSL methods for 200 epochs on OpenImages-Subset unless mentioned otherwise.

MoCo-v2: For MoCo-v2 [39] we closely follow the standard hyper-parameters. We used a learning rate of 0.03 using SGD as our optimizer, weight decay of 10^{-4} and initial learning rate of 0.03. To train OpenImages-Subset, we trained for 200 epochs and for COCO, following DenseCL [245], we trained for 800 epochs.

Supervised Training on OpenImages: For supervised training on OpenImages, we use standard Adam optimizer with learning rate of 10^{-3} , patience of 5 epochs and trained the network for 90 epochs.

CMC: We follow the standard hyperparameters following CMC [219]: temperature 0.07, momentum 0.5 and learning rate 0.03.

BYOL: For BYOL, we used LARS optimizer with learning rate of 0.2, following the standard hyperparameters listed in [86].

SWaV: For SWaV the standard hyperparameters listed in SWaV [24] do not work well for OpenImages-Subset (the authors do not show results on this dataset in their paper). So following their FAQ's in the official github repository (<https://github.com/facebookresearch/swav>) we tried few different hyperparameters. Initially we started with epsilon of 0.05 and then decreased it to 0.03. We also decreased the LR to half and froze the prototypes during the first few iterations. We trained the network for 200 epochs.

4.14 Theoretical motivation

Our hypothesis is that uniformly random image cropping has a high likelihood of missing relevant objects or having other noisy signals. This is especially true when objects are small relative to the size of the image. This leads to suboptimal representations for multi-object datasets such as OpenImages due to the inclusion of data that is weakly correlated with the objects of interest. To overcome this, we propose using semantic information in the cropping to ensure a higher probability of selecting crops around relevant semantic objects in the scene. While we apply our techniques to several self-supervised training approaches, our intuition is best understood in the context of contrastive learning frameworks.

It has been shown [230] that objectives such as InfoNCE estimate a lower bound on $I(X; C)$, where $I(.,.)$ is the mutual information, X is the "image signal of interest" (X) and the correlated context (C). InfoNCE approximates the lower bound by maximizing the correlation between encoded image inputs $g_\theta(X)$ and $g_\theta(C)$, while using uncorrelated contexts to prevent code collapse. Here g_θ is the neural network parameterized by θ , and θ is optimized using stochastic gradient descent based optimization.

$I(X; C)$ can be rewritten as $\int_{x,c} p(x, c) \log(p(x|c)/p(x)) dx dc$. The statement for our approach is that *we plug in a new $p'(x)$ that concentrates the probability mass around regions of X that are strongly correlated with artefacts of interest in the downstream tasks*. Intuitively, this ensures that the training procedure for g_θ is more focused on X samples that are strongly correlated with objects of interest.

For downstream tasks such as object recognition and detection, we can approximate "strong correlation with artefacts of interest" in a crop with the presence of objects in that crop. Therefore our approach can be interpreted as plugging in a new distribution $p'(x)$ in the place of $p(x)$ in the following InfoNCE equation.

$$I(X; C) = \int_{x,c} p(x, c) \log \frac{p(x|c)}{p(x)} dx dc \geq \int_{(x,c,c'_{1:k})} -\log \frac{f(x, c)}{f(x, c) + \sum_k f(x, c'_k)} d(x, c, c'_{1:k}) \quad (4.7)$$

Here f is the function used to compute the density ratio [230] and $c'_{1:k}$ are the k negative samples. Our proposed $p'(x)$ should produce a set of samples that are more enriched with object-centric crops. To produce this enriched sample set, we use three object proposal models: BING [42] and Edge-Boxes [271], both of which are trained with boxes from Pascal-VOC; and an unsupervised object proposal method [237].

In the later sections of this chapter, we provide experimental result that validate the hypothesis that our plugged in $p'(x)$ distribution leads to improved downstream performance. Additionally, our schema also shows improvement over self-supervised approaches, such as BYOL and self-EMD, which cannot be decomposed into a gradient descent procedure for an objective function. Our approach also seems to complement methods like self-EMD that are already designed to improve performance on non-iconic image datasets such as COCO. **Analysis of Obj-Scene and Obj-Obj+Shift crops:** As shown in [219], the performance of contrastive self-supervised methods is have "sweet spot" which correspond to an optimal amount of MI between views. We perform a similar analysis on the Obj-Scene crop approach

by creating crops with progressively higher overlaps. Fig.4.6(left) (supplementary) shows the performance peaks around an overlap of 57.2%. This finding is consistent with that of [219]. Scene-Scene crop has a higher optimal overlap of 65.2%. Varying the radius of Obj-Obj+Shift crop (Fig 4.6(right) for OHMS; Fig.4.5(right) for COCO shows similar results. We show additional analysis on overlap between crops and object of interest in Section 4.10.

4.15 Conclusion

We have introduced object-aware cropping, a simple, fast and highly effective data augmentation alternative to random scene cropping. We conducted numerous experiments to show that object cropping significantly improves performance over scene cropping for self-supervised pre-training for classification, object detection and semantic segmentation on a number of datasets. The approach can be incorporated into most self-supervised learning pipelines in a seamless manner.

Chapter 5: Hyperbolic Contrastive Learning for Visual Representations beyond Objects

5.1 Introduction

Our visual world is diverse and structured. Imagine taking a close-up of a box of cereal in the morning. If we zoom out slightly, we may see different nearby objects such as a bowl of milk, a cup of hot coffee, today’s newspaper, or reading glasses. Zooming out further, we will probably recognize that these items are placed on a dining table with the kitchen as background rather than inside a bathroom. Such scene-object structure is diverse, yet not completely random. In this chapter, we aim at learning visual representations of both the cereal box (objects) and the entire dining table (scenes) in the same space while preserving such hierarchical structures.

[1](#)

Un-/self-supervised learning has become a standard method to learn visual representations [23, 27, 35, 36, 86, 92, 188]. Although these methods attain superior performance over the supervised pretraining on object-centric datasets such as ImageNet [23, 86], inferior results are observed on images depicting multiple objects such

¹The contents of this work is in collaboration with Songwei Ge, Simon Kornblith, Chun-Liang Li, and David Jacobs.

as OpenImages or COCO [254]. Several methods have been proposed to mitigate this issue [5, 152, 254, 256], but all focus on learning improved object representations or dense pixel representations, instead of explicitly modeling the representations for scene images. The object representations learned by these methods present a natural topology [250]. That is, the objects from visually similar classes lie close to each other in the representation space. However, it is not clear how the representations of scene images should fit into that topology. Naively applying existing contrastive learning results in sub-optimal topology of scenes and objects as well as unsatisfactory performance as we will show in the experiment. To this end, we argue that a hierarchical structure can be naturally adopted. Considering that the same kind of objects can be placed in different scenes, we can construct a hierarchical structure to describe such relationships, where the root nodes are the visually similar objects, and the scene images consisting of them are placed as the descendants. We call this structure the object-centric scene hierarchy.

The intermediate modeling difficulty induced by this structure is the combinatorial explosion. A finite number of objects leads to exponentially many different possible scenes. Hyperbolic space is known for its provably better capacity in modeling infinite trees with significantly lower distortion compared with Euclidean space [72, 87, 149]. Therefore, we propose to employ a hyperbolic objective to regularize the scene representations. Our framework builds upon MoCo [92], which has been shown to learn good object representations. To learn representations of scenes, we sample the co-occurring scene-object pairs as the positive pairs, and objects that are not part of that scene as the negative samples, and use these pairs to compute

r



Figure 5.1: Illustration of the representation space learned by our models. Object images of the same class tend to gather near the center around similar directions, while the scene images are far away in these directions with larger norms.

an auxiliary hyperbolic contrastive objective. Our model is trained to reduce the distance between positive pairs and push away the negative pairs in a hyperbolic space.

Contrastive learning models generally compute their objectives on a hypersphere [35, 92]. By discarding the norm information, these models effectively circumvent the shortcut of minimizing objectives by tuning the norms and obtain better downstream performance. At the same time, they also lose control of the representative power in the magnitude of the norm and leave the images disorganized. However, in hyperbolic space, it is the magnitude of the norm that is used to model the hypernymy of the hierarchical structure [175, 198, 223]. When projecting the representations to the hyperbolic space, the norm information is preserved and used to determine the Riemannian distance, which eventually affects our loss. Since the hyperbolic space is diffeomorphic and conformal to the Euclidean, our hyperbolic contrastive loss is completely differentiable and complementary to the original contrastive objective.

When training simultaneously with the original contrastive objective for objects and our proposed hyperbolic contrastive objective for scenes, the resulting representation space exhibits the desired hierarchical structure while keeping the object clustering topology intact as shown in Figure 5.1. We demonstrate the effectiveness of the learned representations on several downstream tasks, from image classification to object detection. We also show that the properties possessed by the representations allow us to perform various vision tasks in a zero-shot way, from label uncertainty quantification to out-of-context object detection. Our contributions

are summarized below:

1. We propose to learn representations for both object and scene images simultaneously using un-/self-supervised methods. We identify an object-centric scene hierarchy that the representations are expected to follow.
2. We propose a framework with a novel hyperbolic contrastive loss to regularize the scene representations with positive and negative pairs sampled from the hierarchy.
3. We show that the magnitude of representation norms effectively reflect the scene-objective hypernymy, and such representations transfer better to multiple downstream tasks.

5.2 Method

In this section, we elaborate upon our approach to learning visual representations of object and scene images. We start by describing the hierarchical structure between objects and scenes that we wish to enforce in the learned representation space.

5.2.1 Object-Centric Scene Hierarchy

From simple object co-occurrence statistics [71, 165] to finer object relationships [121, 133], using hierarchical relationships between objects and scenes to understand images is not new. Previous studies primarily work on an instance-level hierarchy by dividing an image into its lower-level elements recursively - a scene contains multiple

objects, an object has different parts, and each part may consist of even lower-level features [43, 181, 182]. While this is intuitive, it describes a hierarchical structure contained in the individual images. In our task, we would like to learn a single representation space that is shared by all objects and scenes across the entire dataset. To this end, we argue that it is more natural to consider an object-centric hierarchy.

It is known that when training an image classifier, though not being optimized directly, the objects from visually similar classes often lie close to each other in the representation space [250], which has become the cornerstone of contrastive learning [35, 92]. Motivated by this observation, we believe that the representation of each scene image should also be close to the object clusters it consists of. However, they require a much larger volume due to the exponential number of possible compositions. Another way to think about the object-centric hierarchy is through the generality and specificity as often discussed in the language literature [166, 175]. An object concept is general when standing alone in the visual world, and it will become specific when a certain context is given. For example, “a desk” is thought to be a more general concept than “a desk in a classroom with a boy sitting on it”.

Therefore, we propose to study an object-centric hierarchy across the entire dataset. Formally, given a set of images $\mathcal{S} = \{s_1, s_2, \dots, s_n\}$, $\mathcal{O}_i = \{o_i^1, o_i^2, \dots, o_i^{n_i}\}$ are the object bounding boxes contained in the image s_i . We define the regions of scene $\mathcal{R}_i = \{r_i^1, r_i^2, \dots, r_i^{m_i}\}$ to be partial areas of the image s_i that contain multiple objects such that $r_i^j = \cup_k o_i^k$, where $o_i^k \in \mathcal{O}_i$ and object k is in the region j . We define the object-centric forest $T = (V, E)$ to be that $V = \mathcal{S} \cup \mathcal{O} \cup \mathcal{R}$, where $\mathcal{R} = \mathcal{R}_1 \cup \dots \cup \mathcal{R}_n$ and $\mathcal{O} = \mathcal{O}_1 \cup \dots \cup \mathcal{O}_n$. For $u, v \in V$, $e = (u, v)$ is an edge of T

if $u \subseteq v$ or $v \subseteq u$. Note that the natural scene images $\mathcal{S}$ are always put as the leaf nodes.

5.2.2 Representation Learning beyond Objects

To describe our proposed model that is built on this hierarchy, we begin with a brief review of the hyperbolic space and its several properties that will be used in our model. For comprehensive introductions to the Riemannian geometry and hyperbolic space, we refer the readers to [61, 140].

5.2.2.1 Hyperbolic Space

A hyperbolic space $(\mathbb{H}^m, g)$ is a complete, connected Riemannian manifold with constant negative sectional curvature. These special manifolds are all isometric to each other with the isometries defined as $O^+(m, 1)$. Among these isometries, there are five common models that previous studies often work on [20]. In this chapter, we choose the Poincaré ball $\mathbb{D}^n := \{p \in \mathbb{R}^n \mid \|p\|^2 < r^2\}$ as our basic model [73, 175, 223], where $r > 0$ is the radius of the ball. The Poincaré ball is coupled with a Riemannian metric $g_{\mathbb{D}}(p) = \frac{4}{(1 - \|p\|^2/r^2)^2} g_{\mathbb{E}}$, where $p \in \mathbb{D}^n$ and $g_{\mathbb{E}}$ is the canonical metric of the Euclidean space. For $p, q \in \mathbb{D}$, the Riemannian distance on the Poincaré ball induced by its metric $g_{\mathbb{D}}$ is defined as follows:

$$d_{\mathbb{D}}(p, q) = 2r \tanh^{-1} \left(\frac{\| -p \oplus q \|}{r} \right), \quad (5.1)$$

where $\oplus$ is the Möbius addition and it is clearly differentiable. In addition, the Poincaré ball can be viewed as a natural counterpart of the hypersphere as it allows all directions, unlike the other models such as the halfspace or hemisphere models that have constraints on the directions. The hyperbolic space is globally diffeomorphic to the Euclidean space, which is stated in the theorem below:

Theorem 5.2.1. (*Cartan–Hadamard*). *For every point $p \in \mathbb{H}^n$ the exponential map $\exp_p : T_p\mathbb{H}^n \approx \mathbb{R}^n \rightarrow \mathbb{H}^n$ is a smooth covering map. Since $\mathbb{H}^n$ is simply connected, it is diffeomorphic to $\mathbb{R}^n$.*

Specifically, for $p \in \mathbb{D}^n$ and $v \in T_p\mathbb{D}^n \approx \mathbb{R}^n$, the exponential map of the Poincaré ball $\exp_p : T_p\mathbb{D}^n \rightarrow \mathbb{D}^n$ is defined as

$$\exp_p(v) := p \oplus \left(\tanh \left(\frac{r\|v\|}{r^2 - \|p\|^2} \right) \frac{rv}{\|v\|} \right), \quad (5.2)$$

The exponential map gives us a way to map the output of a network, which is in the Euclidean space, to the Poincaré ball. In practice, to avoid numerical issues, we clip the maximal norm of v with $r - \varepsilon$ before the projection, where $\varepsilon > 0$. During the backpropagation, we perform RSGD [13] by scaling the gradients by $g_{\mathbb{D}}(p)^{-1}$. Intuitively, this forces the optimizer to take a smaller step when p is closer to the boundary. The scaling factor is lower bounded by $\mathcal{O}(\varepsilon^2)$.

The immediate consequence of the negative curvature is that for any point $\mathbf{p} \in \mathbb{H}^m$, there are no conjugate points along any geodesic starting from $\mathbf{p}$. Therefore, the volume grows exponentially faster in hyperbolic space than in Euclidean space. Such a property makes it suitable to embed the hierarchical structure that has

constant branching factors and exponential number of nodes. This is formally stated in the theorem below:

Theorem 5.2.2. *[72, 87] Given a Poincaré ball $\mathbb{D}^n$ with an arbitrary dimension $n \geq 2$ and any set of points $p_1, \dots, p_m \in \mathbb{D}^n$, there exists a finite weighted tree (T, d_T) and an embedding $f : T \rightarrow \mathbb{D}^n$ such that for all i, j ,*

$$\left| d_T(f^{-1}(x_i), f^{-1}(x_j)) - d_{\mathbb{D}}(x_i, x_j) \right| = \mathcal{O}(\log(1 + \sqrt{2}) \log(m))$$

Intuitively, the theorem states that any tree can be embedded into a Poincaré disk ($n = 2$) with low distortion. On the contrary, it is known that the Euclidean space with unbounded number of dimensions is not able to achieve such a low distortion [149]. One useful intuition [198] to help understand the advantage of the hyperbolic space is given two points $p, q \in \mathbb{D}^n$ s.t. $\|p\| = \|q\|$,

$$d_{\mathbb{D}}(p, q) \rightarrow d_{\mathbb{D}}(p, 0) + d_{\mathbb{D}}(0, q), \text{ as } \|p\| = \|q\| \rightarrow r \quad (5.3)$$

This property basically reflects the fact that the shortest path in a tree is the path through the earliest common ancestor, and it is reproduced in the Poincaré when points are both close to the boundary.

5.2.2.2 Hyperbolic Contrastive Learning

Given the theoretical benefits of the hyperbolic space stated above, we propose a contrastive learning framework as shown in Figure 5.2. We adopt two losses to learn

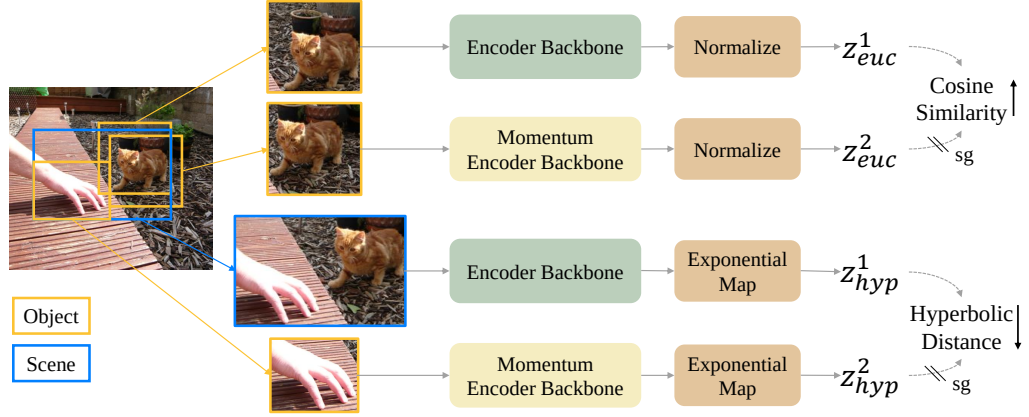


Figure 5.2: Our **H**yperbolic **C**ontrastive **L**earning (HCL) framework has two branches: given a scene image, two object regions are cropped to learn the object representations with a loss defined in the Euclidean space focusing on the representation directions. A scene region as well as a contained object region are used to learn the scene representations with a loss defined in the hyperbolic space that affects the representation norms.

the object and scene representations. First, to learn object representations, we use the standard normalized temperature-scaled cross-entropy loss, which operates on the hypersphere in Euclidean space. As shown in the top branch of Figure 5.2, we crop two views of a jittered and slightly expanded object region as the positive pairs and feed into the base and momentum encoders to calculate the object representations. We denote the output after the normalization to be $\mathbf{z}_{\text{euc}}^1$ and $\mathbf{z}_{\text{euc}}^2$. Given the computational cost of large batch sizes, we follow MoCo [39, 92] and leverage a memory bank to store the negative representations $\mathbf{z}_{\text{euc}}^n$, which are the features $\mathbf{z}_{\text{euc}}^2$ from the previous batches. The Euclidean loss for each image is then calculated as:

$$\mathcal{L}_{\text{euc}} = -\log \frac{\exp(\mathbf{z}_{\text{euc}}^1 \cdot \mathbf{z}_{\text{euc}}^2 / \tau)}{\exp(\mathbf{z}_{\text{euc}}^1 \cdot \mathbf{z}_{\text{euc}}^2 / \tau) + \sum_n \exp(\mathbf{z}_{\text{euc}}^1 \cdot \mathbf{z}_{\text{euc}}^n / \tau)},$$

where τ is a temperature parameter.

While the loss above aims to learn object representations, we also use a hyperbolic contrastive objective to learn the representations for scene images. We sample the positive region pairs u and v from object-centric scene hierarchy T such that $(u, v) \in E$. In other words, as shown in the bottom branch of Figure 5.2, the objects contained in one region are required to be a subset of the objects in the other. We sample the negative samples of u to be $\mathcal{N}_u = \{v | (u, v) \notin E\}$. However, building and sampling from the entire hierarchy explicitly is slow and memory consuming. Instead, we assume that there are exponentially more scenes than object classes. In practice, given an image s , we always sample $u \in \mathcal{R} \cup \{s\}$ to be a scene region, $v \in \mathcal{O}$ to be an object that occurs in u , and $\mathcal{N}_u$ to be the other objects that are not in u .

The pair of scene and object images are fed into the base and momentum encoders that share the weights with the Euclidean branch. However, instead of normalizing the output of the encoders, we use the exponential map defined in the equation 5.2 to project these features in the Euclidean space to the Poincaré ball, which are denoted as $\mathbf{z}_{\text{hyp}}^1$ and $\mathbf{z}_{\text{hyp}}^2$. Further, we replace the inner product in the cross-entropy loss with the negative hyperbolic distance as defined in equation 5.1. We calculate the hyperbolic contrastive loss as follows:

$$\mathcal{L}_{\text{hyp}} = -\log \frac{\exp(-d_{\mathbb{D}}(\mathbf{z}_{\text{hyp}}^1, \mathbf{z}_{\text{hyp}}^2)/\tau)}{\exp(-d_{\mathbb{D}}(\mathbf{z}_{\text{hyp}}^1, \mathbf{z}_{\text{hyp}}^2)/\tau) + \sum_n \exp(-d_{\mathbb{D}}(\mathbf{z}_{\text{hyp}}^1, \mathbf{z}_{\text{hyp}}^n)/\tau)},$$

When minimizing the distances of all the positive pairs, with the intuition from Equation 5.3, it would be beneficial to put the nodes near the root close

to the center to achieve a overall lower loss. The overall loss function of our model is as follows:

$$\mathcal{L} = \mathcal{L}_{\text{euc}} + \lambda \mathcal{L}_{\text{hyp}},$$

where λ is an scaling parameter to control the trade-off between hyperbolic and Euclidean losses.

5.3 Experiments

5.3.1 Implementation Details

Pre-training phase. We pre-train our method on two datasets: COCO [148] and a subset of OpenImages [168]. Both of these datasets are multi-object datasets; OpenImages [168] ($\sim 212\text{k}$ images) contains 12 objects on average per image and COCO ($\sim 118\text{k}$) contains 6 objects on average. We experiment with both the ground truth bounding box (GT) and using selective search [229] following the previous method [254] (SS) to acquire objects. For the optimizer setups and augmentation recipes, we follow the standard protocol described in MoCo-v2 [39] unless denoted otherwise. We find that a base learning rate of 0.3 works better for us as compared to 0.03. We adopt the linear learning rate scaling receipt that $lr = 0.3 \times \text{BatchSize}/256$ [83] and batch size of 128 by default on 4 NVIDIA p6000 gpus. To ensure fair comparison, we also pre-train the baselines with a learning rate of 0.3. We train our models on both datasets for 200 epochs. For the hyperparameters of our hyperbolic objective, we use $r = 4.5$, $\lambda = 0.1$, and $\varepsilon = 1e^{-5}$.

Method	Dataset	Bbox	VOC	IN-100	INPMix	IN1k
MoCo-v2	COCO	-	64.79	64.84	41.83	51.17
HCL/ $\mathcal{L}_{\text{hyp}}$	COCO	SS	73.13	73.84	51.28	54.21
HCL/ $\mathcal{L}_{\text{hyp}}$	COCO	GT	75.55	76.22	51.25	54.52
HCL	COCO	SS	74.19	75.16	51.35	55.03
HCL	COCO	GT	76.51	76.74	51.63	55.63
MoCo-v2	OpenImages	-	69.95	72.80	49.59	54.12
HCL/ $\mathcal{L}_{\text{hyp}}$	OpenImages	GT	73.79	77.36	52.96	57.57
HCL	OpenImages	SS	74.31	78.14	53.21	58.12
HCL	OpenImages	GT	75.40	79.08	53.82	58.51

Table 5.1: **Classification results with linear evaluation.** Our model improves scene-level classification on the VOC [68] and INPMix [269] datasets, and object-level classification on ImageNet-100 [219] and ImageNet-1k [56] datasets.

Downstream tasks. We evaluate our pre-trained models on image classification, object-detection and semantic segmentation. For classification, we show linear evaluation (lineval) accuracy, i.e we freeze the backbone and only train the final fc layer. We test on VOC [68], ImageNet-100 [219] and ImageNet-1k [56] datasets. To test the discriminative capacity of the representations on both objects and scenes, we create a dataset by mixing the ImageNet-100 and a subset of Place-205 [269] datasets, which we refer to as the INPMix dataset. More details of this dataset can be found in Chapter 4. For object detection and semantic segmentation, we show results on the COCO and Pascal VOC `trainval2017` datasets. For VOC object detection, COCO object detection and COCO semantic segmentation, we closely follow the common protocols listed in Detectron2 `wu2019detectron2`.

Detection	Dataset	AP	AP ₅₀	AP ₇₅
MoCo-v2	COCO	34.6	53.5	37.0
HCL/ $\mathcal{L}_{\text{hyp}}$	COCO	36.1	55.2	37.9
HCL	COCO	37.0	56.1	39.8
MoCo-v2	VOC	51.5	79.4	56.1
HCL/ $\mathcal{L}_{\text{hyp}}$	VOC	53.7	80.5	59.4
HCL	VOC	54.4	81.4	60.2
Segmentation	Dataset	AP _s	AP _l	AP _m
MoCo-v2	COCO	30.4	50.1	32.3
HCL/ $\mathcal{L}_{\text{hyp}}$	COCO	31.5	52.0	33.8
HCL	COCO	32.5	52.9	34.6

Table 5.2: **Object detection and Semantic Segmentation results.** Our model improves on both tasks on COCO [148] and VOC [68] datasets.

5.3.2 Main Results

This section discusses our main results on the downstream image classification, object detection, and semantic segmentation tasks. As the goal of this chapter is not to present another state-of-the-art self-supervised learning method, we primarily compare with the backbone model MoCo-v2 [92]. We also consider our model without the hyperbolic loss $\mathcal{L}_{\text{hyp}}$. This model, which we denote as HCL/ $\mathcal{L}_{\text{hyp}}$, learns only object representations are learned.

Image classification. As shown in Table 5.1, HCL improves image classification on both scene-level datasets (VOC and INPMix) and object-level datasets (ImageNet). When pretraining on OpenImages, HCL improves ImageNet lineal accuracy by 0.94% and VOC lineal classification accuracy by 1.61 mAP. We observe similar improvements when pretraining on COCO. HCL improves accuracy whether we use ground truth object bounding boxes or boxes generated by selective

search. In general, we observe a larger improvement of using HCL on OpenImages than COCO, which supports our observation that HCL would improve more on the dataset with more objects per images.

Object detection and semantic segmentation. Table 5.2 reports the object detection and semantic segmentation results using Mask R-CNN, following [39]. It shows consistent improvements over the baselines on VOC object detection, COCO object detection, and COCO semantic segmentation.

5.3.3 Properties of Models Trained with HCL

The visual representations learned by HCL have several useful properties. In this section, we evaluate the representation norm as an measure of the label uncertainty for image classification datasets, and evaluate the object-scene similarity in terms of out-of-context detection.

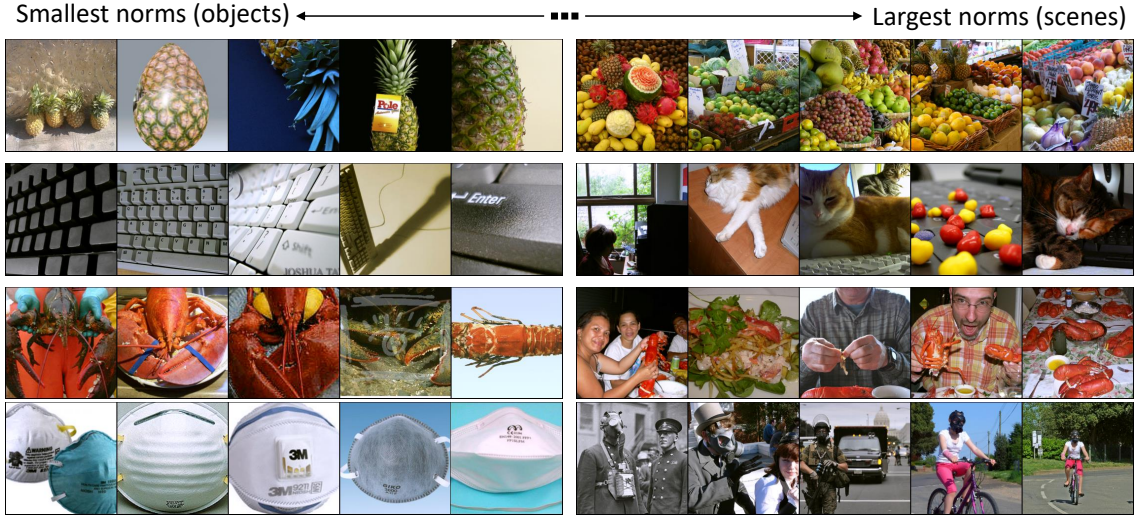


Figure 5.4: Images from ImageNet training set. The 5 images on the left have the smallest representation norms among all the images from the same class, and the 5 on the right have the largest norms.

5.3.3.1 Label Uncertainty Quantification

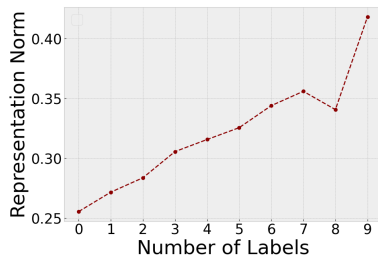


Figure 5.3: Average representation norms of images with different number of labels in ImageNet-Real [11].

Method	Indicator	Datasets	
		IN-Real	COCO
MoCo	Entropy	0.633	0.791
Supervised	Entropy	0.671	0.793
HCL	Norm	0.655	0.839
Ensemble	Entropy+Norm	0.717	0.823

Table 5.3: NDCG scores of the image rankings based on the different indicators and models, and evaluated by the the number of labels per image.

ImageNet [56] is an image classification dataset consisting of object-centered images, each of which has a single label. As the performance on this dataset has gradually saturated, the original labels have been scrutinized more carefully [11,

189, 203, 228, 232]. Prevailing labeling issues in the validation set have been recently identified [11, 203, 228], including labeling errors, multi-label images with only a single label provided, and so on. Although Beyer et al. [11] provide reassessed labels for the entire validation set, relabeling the entire training set can be infeasible.

Our learned representations provide a potential automatic way to identify images with multiple labels from datasets like ImageNet. Specifically, we first show in Figure 5.3 that there is a strong correlation between the representation norms and the number of labels per image according to the reassessed labels. For each class of the ImageNet training set, we rank the images according to their norms. The extreme images of some classes are shown in Figure 5.4. Images with smaller norms tend to capture a single object, while those with larger norms are likely to depict a scene.

To quantitatively evaluate this property, we report the NDCG metric on the ranked images as shown in Table 5.3. NDCG assesses how often the scene images are ranked at the top. As a baseline, we rank the images based on the entropy of the class probability predicted by a classifier, which is a widely adopted indicator of label uncertainty [33, 180]. We use both MoCo-v2 and supervised ResNet-50 as the classifier. As shown in Table 5.3, using norms with HCL achieves similar rank quality as using entropy with the supervised ResNet-50 on the ImageNet-ReaL dataset. In addition, when combining two ranks using simple ensemble methods such as Borda count, the score is further improved to 0.717. This shows that the entropy and the norm provide complimentary signals regarding the existence of multiple labels. For example, the entropy indicator can be affected by the bias of the model and the

norm indicator can be wrong on the images with multiple objects from the same class.

Compared to supervised indicators of label uncertainty, HCL has the additional advantage that it is dataset-agnostic and can be applied to new data without further training. To demonstrate this benefit, we report the same metric on the COCO validation, where we also have the number of labels for each image. Our method achieves much better NDCG scores than the supervised ResNet-50 as shown in Table 5.3. This finding can be potentially useful to guide label reassessment, or provide an extra signal for model training.

5.3.3.2 Out-of-Context Detection

Our hyperbolic loss $\mathcal{L}_{\text{hyp}}$ encourages the model to capture the similarity between the object and scene. We apply the resulting representations to detect out-of-context objects, which can be useful in designing data augmentation for object detection [65]. We are especially interested in out-of-context images with conflicting backgrounds. To this end, we use the out-of-context images proposed in the SUN09 dataset [43]. We first compute the representation of each object as well as the entire scene image with that object masked out. We then calculate the hyperbolic distance between the representations mapped to the Poincaré ball. Some example images from this dataset as well as the distance of each contained object are shown in Figure 5.5. We find that the out-of-context objects generally have a large distance, i.e. smaller similarity, to the overall scene image. To quantify this finding, we compute the mAP

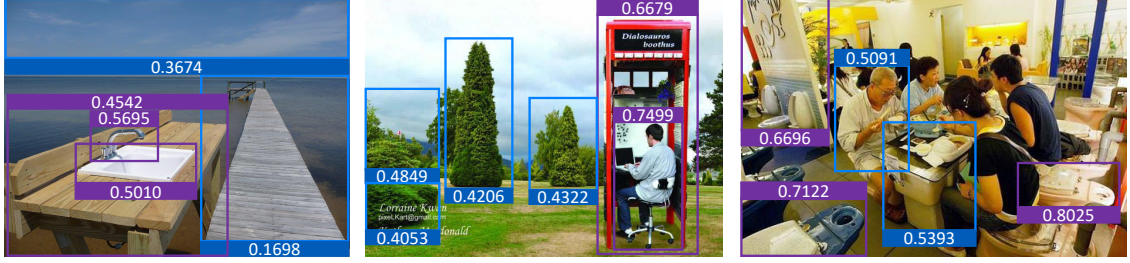


Figure 5.5: Out-of-context images from the SUN09 dataset [43]. The bounding box of each object, as well as its hyperbolic distance to the scene are displayed. The regular objects are in blue and the out-of-context objects are in purple. Note that the out-of-context objects tend to have large distances.

Dist.	Center	IN-100	IPS
-	-	77.36	52.96
Hyp.	Scene	79.08	53.82
Hyp.	Object	76.96	52.74
Euc.	Scene	76.68	52.58

Table 5.4: Ablation on the similarity measure and hierarchy center.

of the object ranking on each image and obtain 0.61 for HCL. As a comparison, the MoCo similarity gives $\text{mAP} = 0.52$ and the random ranking gives $\text{mAP} = 0.44$.

5.4 Main Ablation Studies

In this section, we report the results of several important ablation studies with respect to HCL. All the models are trained on the subset of the OpenImages dataset and linearly evaluated on the ImageNet-100 and our INPMix datasets. The top-1 accuracy is reported.

Similarity measure and the center of the scene-object hierarchy. We propose to use the negative hyperbolic distance as the similarity measure of the scene-object pairs.

As an alternative, one can use cosine similarity on the hypersphere as the measure

λ	IN-100	IPS
0.01	77.70	53.43
0.1	79.08	53.82
0.2	78.64	53.84
0.5	0	0

Table 5.5: Ablation on the losses trade-off.

Optim.	λ	IN-100	IPS
RSGD	0.1	79.08	53.82
RSGD	0.5	0	0
SGD	0.1	70.16	48.47
SGD	0.5	74.18	42.75

Table 5.6: Ablation on the RSGD versus SGD optimizers.

as in the original contrastive objective. However, this is basically minimizing the similarity between a single object and multiple objects. It is likely that these objects belong to different classes, and hence this strategy could impair the quality of the representation. As shown in Table 5.4, replacing the negative hyperbolic distance with the Euclidean similarity impairs downstream performance. The resulting model performs even worse than the baseline model that has no loss function on the scene-object pairs. We also validate our choice of an object-centric hierarchy by comparing its performance with that of a scene-centric hierarchy [181, 182] generated by sampling the negative pairs as objects and unpaired scenes. This scene-centric hierarchy leads to substantially lower accuracy (Table 5.4).

Trade-off between the Euclidean and hyperbolic losses. We adopt the Euclidean loss to learn object-object similarity and the hyperbolic loss to learn object-scene similarity. A hyperparameter λ controls the trade-off between them. As shown in

Table 5.5, we find that a smaller $\lambda = 0.01$ leads to marginal improvement. However, we also observe that larger λ s can lead to unstable and even stalled training. With careful inspection, we find that in the early stage of the training, the gradient provided by the hyperbolic loss can be inaccurate but strong, which pushes the representations to be close to the boundary. As a result, since Riemannian SGD divides gradients by the distance to the boundary, updates become small and training ceases to make progress.

Optimizer. Given the observation above, we ask whether RSGD is necessary for practical usage. We replace the RSGD optimizer with SGD. To avoid numerical issues when the representations are too close to the boundary, we increase ε from $1e^{-5}$ to $1e^{-1}$. This allows a larger λ to be used as opposed to the RSGD. However, SGD always yields inferior performance compared to RSGD.

5.5 Related Work

Representation Learning with Hyperbolic Space: Representations are typically learned in Euclidean space. Hyperbolic space has been adopted for its expressiveness in modeling tree-like structures existing in various domains such as language [72, 175, 176, 198, 223], graphs [6, 28, 29, 183], and vision [31, 125, 215]. The corresponding deep neural network modules have been designed to boost the progress of such applications [29, 73, 150, 206]. The hierarchical structure presented in the datasets can arise from three factors that together motivate the use of hyperbolic space. The first factor is generality: the hypernym-hyponym property is a natural

feature of words (e.g. WordNet [166]) and the hyperbolic space is extensively exploited to learn word embeddings that preserve that property [72, 175, 176, 198, 223]. Some image datasets also adopt the classes from WordNet for labeling, e.g. ImageNet [56], and consequently inherits the hierarchy in its labeling system. [151, 156, 260] take advantage of hyperbolic space to capture such information in the visual embeddings. The second factor is uncertainty: Several studies have found that applying hyperbolic neural network modules to different tasks leads to a natural modeling of the uncertainty [79, 125, 215]. The third factor is compositionality of different basic elements to form a natural hierarchy. Motivated by these factors, previous work in computer vision has applied hierarchical representations learned in the hyperbolic space to various tasks such as image classification [125] or segmentation [79, 248], zero-/few-shot learning [151, 156], action recognition [156], and video prediction [215]. In this chapter, we focus on learning the representations that capture the hierarchy between the objects and scenes with the goal of learning general-purpose image representations that can transfer to various downstream tasks.

Self-Supervised Learning on Scenes. Self-Supervised Learning (SSL) has made great strides in closing the performance with supervised methods [35, 39] when pre-trained on the object-centric datasets like ImageNet. However, recent work has shown that SSL is limited on multi-object datasets like COCO [200, 245] and Open-Images [168]. Several papers have tried to address this issue by proposing different techniques. Dense-CL [245] operates on pre-average pool features and uses dense features on pixel level to show improved performance on dense tasks such as seman-

tic segmentation. DetCon [112] uses unsupervised semantic segmentation masks to generate features for the corresponding objects in the two views. CAST [201] uses GradCAM [200] to identify correspondences between objects across views and applies a contrastive loss on these features. PixContrast [256] uses pixel-to-propagation consistency pretext task to build features for both dense downstream tasks and discriminative downstream tasks. Pixel-to-Pixel Contrast [244] uses pixel-level contrastive learning to learn better features for semantic segmentation. Self-EMD [152] uses earth mover distance with BYOL [86] for pretraining on the COCO dataset. ORL [254] uses selective search to generate object proposals, then applies object-level contrastive loss to enforce object-level consistency. ContraCAM [171] mitigates scene bias by performing self-supervised object localization and applying a contrastive loss on the objects. Below-par performance of SSL methods can be attributed to treating scenes and objects using similar techniques, which often results in similar representations. In our work, instead of treating scenes and objects similarly, we use a hyperbolic loss, which builds representation that disambiguates scenes and objects based on the norm of the embeddings. Our method not only separates scenes and objects, but also improves downstream tasks such as image classification.

5.6 Experiment setups

In this section, we provide additional details of our experiments.

Unsupervised object proposals. When pretraining on uncured datasets,

acquiring ground truth object bounding boxes using human annotations can be expensive. However, automatically generating unsupervised region proposal is well studied. We use Selective Search as the unsupervised proposal generation method. Following ORL [254] we first generate the proposals using selective search. Then we filter the proposals with 96 pixels as the minimal scale, maximum IOU of 0.5 and aspect ratio between 1/3 to 3. For every image we generate maximum of 100 proposals and randomly select any image as the object image.

OpenImages dataset. We use the OpenImages subset proposed in [168]. This is a subset created from the full OpenImages dataset where each image has at least 2 classes present and each class has at least 900 instances. This subset is a balanced subset of OpenImages with an average of 12 object present in an image, making it a good proxy for real-world multi-object images. Hence we used it as the main dataset in this chapter. More details of this dataset can be found in [168].

INPMix dataset. We sample a subset of the Place-205 [269] dataset to test the discriminative capacity of the representations on the scene images. Specifically, we randomly sample 1,300 training images from each of the 205 classes, and then combine them with the ImageNet-100 [219] to form a dataset of 305 classes in total. We provide the code to reproduce the dataset in the supplementary material.

Object and Scene image augmentations. We find that small objects are always detrimental to performance. Therefore, when sampling object bounding boxes, we drop bounding boxes with size $\text{width} \times \text{height} \leq 56 \times 56$. Further, when sampling objects for the Euclidean branch, if the size of a bounding box $\text{width} \times \text{height} \leq 256 \times 256$, we slightly expand it to either 256×256 or the maximal size

allowed by the original image size. We also apply a small jittering to the `width` and `height` to include different contexts around the objects. Next, we apply random cropping and resizing with the same scale (0.2, 1.) as in MoCo [92]. When sampling objects for the hyperbolic branch, we do not apply jittering and random cropping, but only filter the small boxes and resize to $\leq 224 \times 224$. To crop the scene images, we sample another 1 to 5 bounding boxes and merge with the selected object bounding box.

5.7 Additional experimental results

5.7.1 Robustness under Corruption.

We calculate the mCE error as in Hendrycks et al. [99]. We compare our HCL model trained on OpenImages and lineval on ImageNet dataset with the baseline model without using HCL loss. We see an improvement of 1.9 mCE over the baseline model, demonstrating that our HCL model learns more robust representations as compared to the vanilla MoCo.

5.7.2 Fine-grained class classification

Method	Cars [132]	DTD [46]	Food [14]
HCL/ $\mathcal{L}_{\text{hyp}}$	31.92	68.46	58.66
HCL	32.02	68.19	58.79

Table 5.7: Fine grained classification results.

In Table 5.7 we show results on fine-grained classification datasets. We can

see that on fine-grained classification our model provides little performance improvement. This could be due to the fact that all classes in these datasets have very similar scene contexts, and hence the hyperbolic objective does not help very much.

5.7.3 More ImageNet Examples

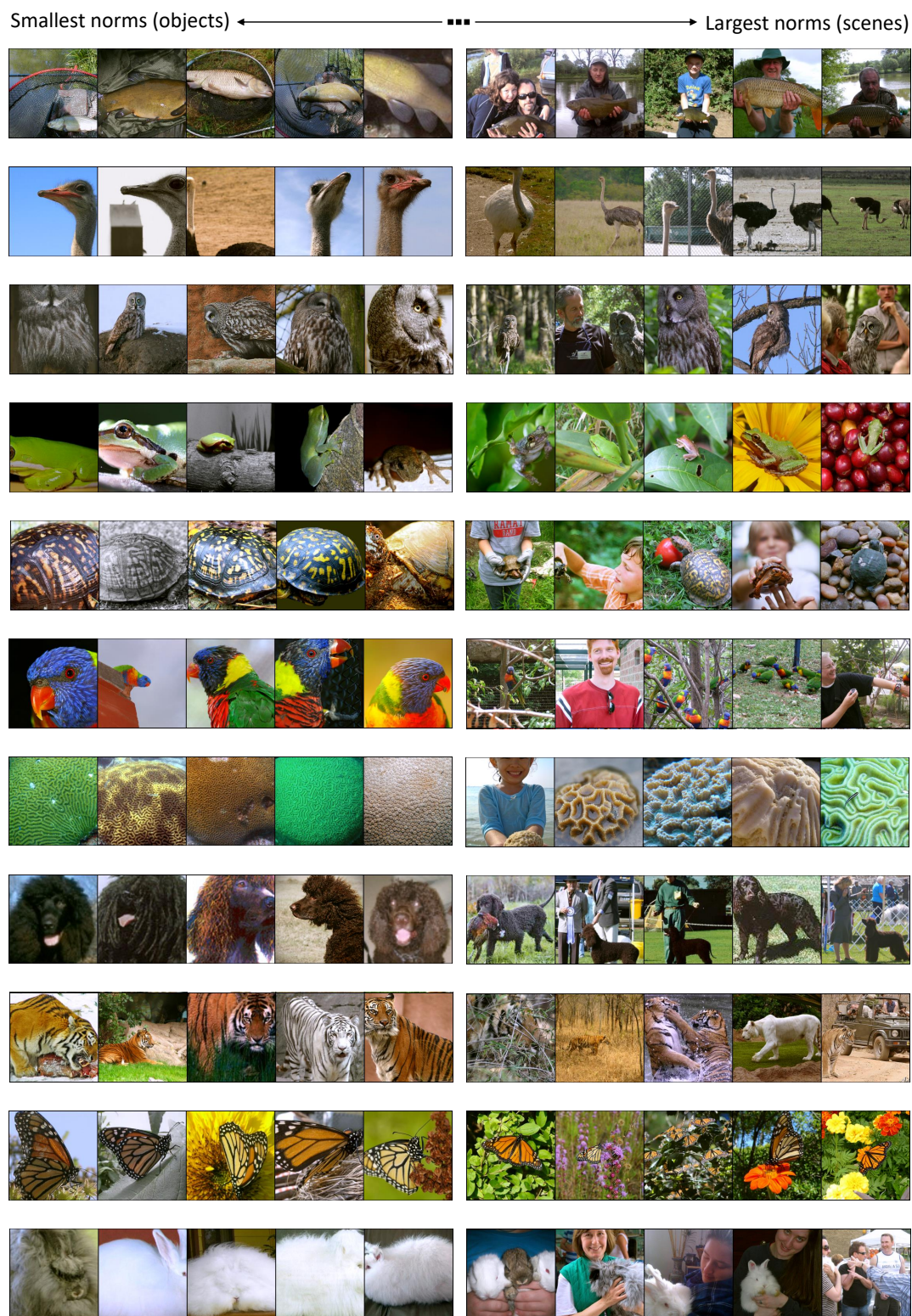


Figure 5.6: More images from ImageNet training set sorted by their representation norms.

Smallest norms (objects) ← ———— → Largest norms (scenes)

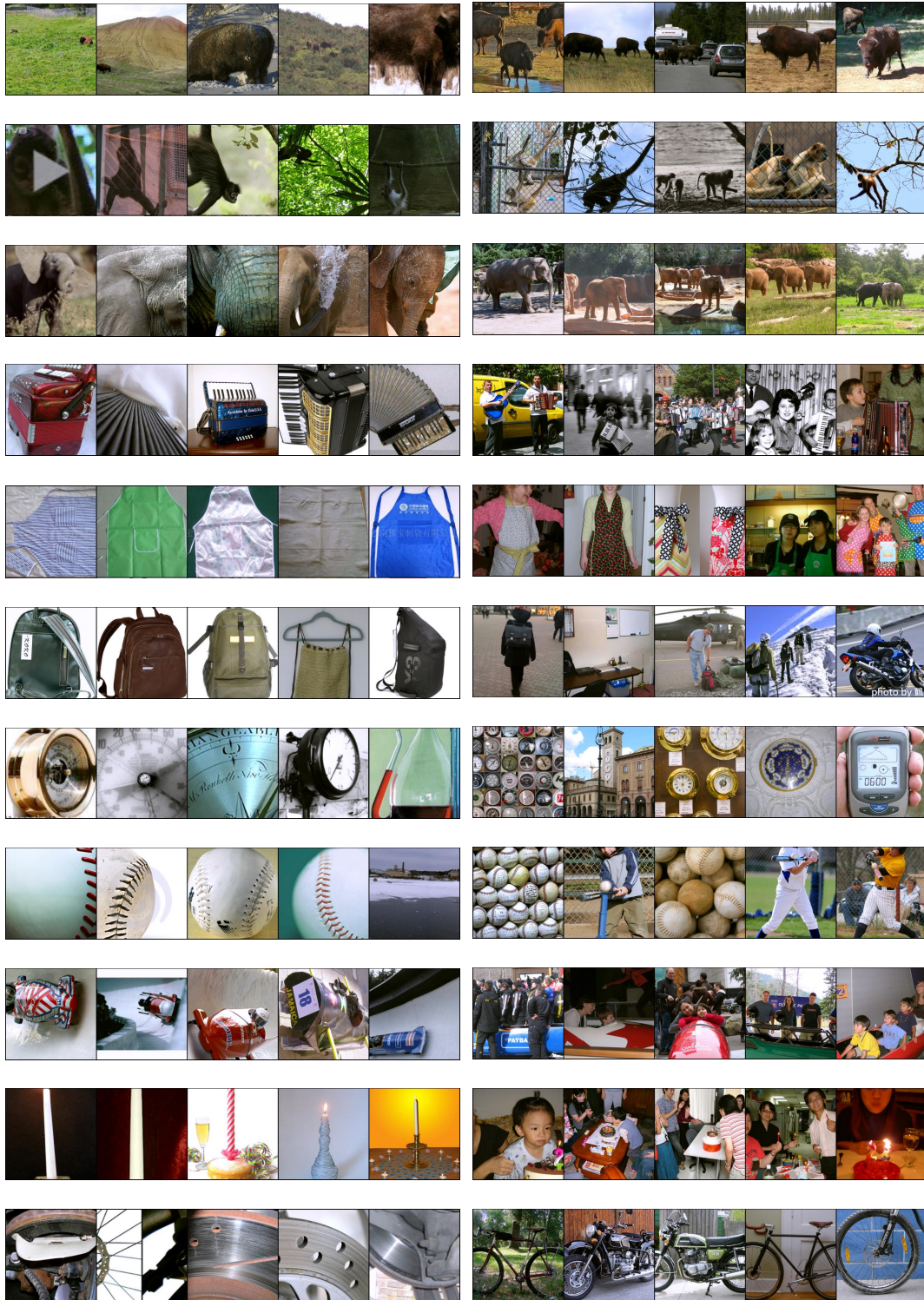


Figure 5.7: More images from ImageNet training set sorted by their representation norms.

5.8 Additional ablation studies

In this section, we provide more ablation experiments on hyperbolic linear evaluation, model architecture, and the radius of Poincare ball. All the models are trained on the OpenImages dataset and evaluated on the ImageNet-100 (IN-100) or ImageNet-1k (IN-1k) the top-1 accuracy reported.

Radius of the Poincare ball In Table 5.8 we show results by varying the radius of Poincaré ball. The hyperbolic objective improves the performance over all the tested radius. We find that a too small radius may lead to a smaller improvement due to the stronger regularization.

Configuration of the encoder head In our experiments, the Euclidean and hyperbolic branches share the weights in both the backbone and the head of the encoders. We also try using a separate head for the hyperbolic branch. As shown in Table 5.9, this leads to a more stable training when larger learning rate is applied. However, we did not see any improvements brought by this modification.

Hyperbolic linear evaluation Apart from the common linear evaluation in the Euclidean space, we show the hyperbolic linear evaluation results with different optimizers and learning rates in Table 5.10. The idea is to test if the representations are more linearly separable in the hyperbolic space. We follow the same setting of hyperbolic softmax regression [73] and train a single hyperbolic linear layer. However, we find the optimization with SGD can easily cause overflow. By contrast, Adam is much more stable with appropriate learning rates.

c	IN-1k
1	58.08
0.5	58.31
0.1	58.29
0.05	58.51
0.01	58.49

Table 5.8: Results by varying the radius r of Poincaré ball. $c = \frac{1}{r^2}$.

Head	λ	IN-100
N/A	0	77.36
shared	0.1	79.08
	0.5	0
split	0.1	77.88
	0.5	77.58

Table 5.9: Different configurations of the head in the Euclidean and hyperbolic branches.

SGD		Adam	
lr	IN-100	lr	IN-100
0.1	63.82	0.001	67.64
0.2	64.22	0.0005	70.32
0.3	1	0.0001	72.58
0.4	1	0.00005	70.5

Table 5.10: Results of hyperbolic linear evaluation with different optimizers and learning rates.

5.9 Closing Remarks

Conclusion We present HCL, a contrastive learning framework that learns visual representation for both objects and scenes in the same representation space. The major novelty of our method is a hyperbolic contrastive objective built on an object-centric scene hierarchy. We show the effectiveness of HCL on several benchmarks including image classification, object detection, and semantic segmentation. We

also demonstrate useful properties of the representations under several zero-shot settings, from detecting out-of-context objects to quantifying the label uncertainty in the datasets like ImageNet. More generally, we hope this chapter will encourage future work towards building a more holistic visual representation space, and draw attention to the power of non-Euclidean representation learning.

Limitations We have shown that our model improves classification performance on the ImageNet dataset, but gains on more fine-grained classification tasks are smaller, as shown in Section 5.7. We conjecture that our model primarily improves object representations by modeling context information, whereas most of these fine-grained datasets have similar contextual information. In addition, although we demonstrate the importance of Riemannian optimization, its underlying mechanism in the visual representation learning remains unclear. We conduct more experiments on training hyperbolic linear classifiers in Section 5.7. However, further efforts are needed to fully unleash the potential of non-Euclidean representation learning.

Chapter 6: A simple, efficient and scalable contrastive masked autoencoder for learning visual representations

6.1 Introduction

Self-supervised learning promises continued advances in the state of the art by enabling the use of increasingly large models and datasets without reliance on human annotations. However, interest in larger datasets has precipitated an increased reliance on web-scraped data collection processes, which result in heterogeneous, uncurated datasets [117, 188, 263]. Extreme image heterogeneity has made scaling vision models to large uncurated datasets a non-trivial challenge [47, 218]. There are two families of self-supervised methods for images which have both proven highly effective on curated datasets (e.g., ImageNet), and are therefore natural candidates for scaling to large, uncurated data. First, masked image models such as the masked autoencoder (MAE) [90] are a nascent approach based on a mask-and-reconstruct training mechanism. This classical idea [7] is enjoying a rejuvenation thanks to favourable efficiency when combined with the vision transformer architecture [63]. Second, contrastive learning [34, 92, 231] trains an encoder to distinguish between pairs of positive samples generated with data augmentations and negative pairs sam-

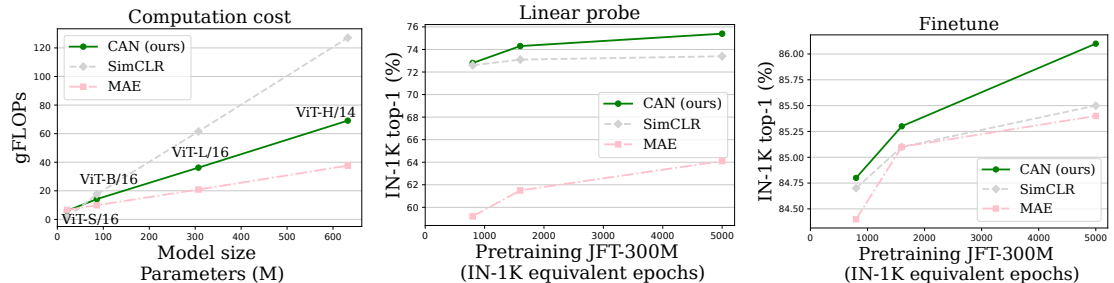


Figure 6.1: **Left:** CAN scales better than SimCLR since it uses masked inputs. **Middle and right:** CAN outperforms SimCLR and MAE on ImageNet linear probe and finetune evaluations for ViT-L models when pre-training on uncured data such as JFT-300M.

pled at random. Both approaches have proven to be very powerful self-supervised methods.

Contrastive learning and masked autoencoders (MAE) employ very different learning mechanisms: the former trains the encoder to be invariant to semantics-preserving data variations, while MAE learns spatial statistical correlations. Furthermore, MAE methods treat each sample independently in the loss function, while contrastive methods explicitly look at the relationship between all samples in the batch, by either reducing or increasing embedding distance. Given this, we hypothesize that these two approaches are *complementary*, extracting different discriminative features. If this hypothesis holds, then combining the two approaches presents itself as a promising way to build a reinforced training mechanism.

Advances in diffusion models [108, 210] have been driven by denoising models that predict the *noise* added to an input image. While denoising has not yet enjoyed wide adoption for self-supervised learning on its own, we believe it offers a promising *third* complementary learning mechanism to contrastive learning and masked autoencoding. Specifically, denoising forces the model to learn high-frequency in-

formation, whereas autoencoder reconstructions focus on low-frequency information [109].

This chapter presents CAN, a minimal fusion of contrastive learning, masked autoencoders, and the diffusion denoising loss. Our method enjoys stronger performance than its constituent parts on their own, especially when pre-training on large datasets such as JFT-300M and ImageNet-21K, which consist of 300M and 14M images, respectively. For instance, evaluating JFT-trained ViT-L models using the top-1 accuracy of an ImageNet-trained linear probe, MAE achieves 64.1% and SimCLR achieves 73.4%, while CAN achieves 75.4%. The advantages of CAN are:

1. **Simplicity.** CAN is a minimal synthesis of three powerful self-supervised learning methods.
2. **Efficiency.** CAN enjoys a favourable efficiency-performance trade-off (Figure 6.1), e.g., SimCLR uses 70% more FLOPs than CAN with ViT-L backbones.
3. **Scalability.** CAN scales well to training on large image datasets, such as JFT-300M and ImageNet-21K.

CAN is more efficient than SimCLR since it masks 50% of patches in each view. This also translates to faster run-times, with our largest training (ViT-L 5000 epochs) taking 2 weeks for SimCLR, and 1 week for CAN on our hardware.

6.2 Related Work

Masked image models with Vision Transformers. The advent of the Vision Transformer (ViT) [63] provoked a focused effort to develop strong self-supervised learning frameworks for ViT backbones. Works such as DINO [27] and MoCo-v3 [41] demonstrated that techniques developed with ConvNet backbones in mind could also perform competitively using ViTs after proper tuning to suit the new architecture. ViT-specific methods have emerged since then, particularly masked image modelling [8, 38, 257], which takes inspiration from pre-training methods used in NLP [58]. Notably MAE [90] showed that classical masked autoencoding approaches could be used to pre-train ViTs *without* passing masked tokens through the encoder. This provides a significant efficiency boost; our method similarly takes advantage of this.

Contrastive learning in computer vision. Self-supervision has received significant attention in computer vision as it offers a way to extract general purpose features without supervision. In particular, contrastive learning [34, 45, 92, 96, 112, 220, 231] has achieved state of the art performance by enforcing invariance to augmentations, whilst using negative samples [75, 194] to avoid trivial solutions by spreading the embedding out uniformly on the sphere [243]. The contrastive pre-training task is conceptually very different from masked image models such as MAE, which learn spatial statistical dependencies. Another distinction is that autoencoders encourage information preservation in latent representations, whilst contrastive learning could suppress features [37, 195]. This leads us to hypothesize that the two approaches learn different, complementary data features. This mo-

tivates us to combine contrastive learning and masked image modelling so as to develop a reinforced pre-training task that enjoys the merits of each.

Denoising diffusion models. Denoising autoencoders (DAE) [236] learn to reconstruct clean data given a noisy input. By learning to map low-density data regions to high-density regions, DAE learns the shape of the data manifold. This connection was made precise by [235], who showed that DAEs learn the score-function $s(\mathbf{x}) = \nabla_{\mathbf{x}} \log p(\mathbf{x})$. This key observation underpins the significant recent advances in generative diffusion models, which use an estimate of the score-function to generate samples [108, 210]. The recent success of DAEs in generative modelling has not yet translated to representation learning, with some exceptions [3, 265]. In this work we exploit a denoising autoencoder to eliminate the MAE inefficiency of reconstructing unmasked patches but never using them.

Siamese masked image modelling. Several recent works propose approaches that combine ideas from masked image modelling and Siamese self-supervised learning. For instance, [111] propose a combination of contrastive and masked reconstruction objectives using one masked view, and one full (unmasked) view. Other recent works [4, 38, 216] use similar asymmetric designs. The key distinction between CAN and these works is that we strike a different balance, focusing on developing a *simple*, and *efficient* method. For instance we use *two masked views* and no momentum encoder. We hope the simplicity and efficiency of CAN, and our experiments showing its scalability, will make it easy to adapt and modify in future work.

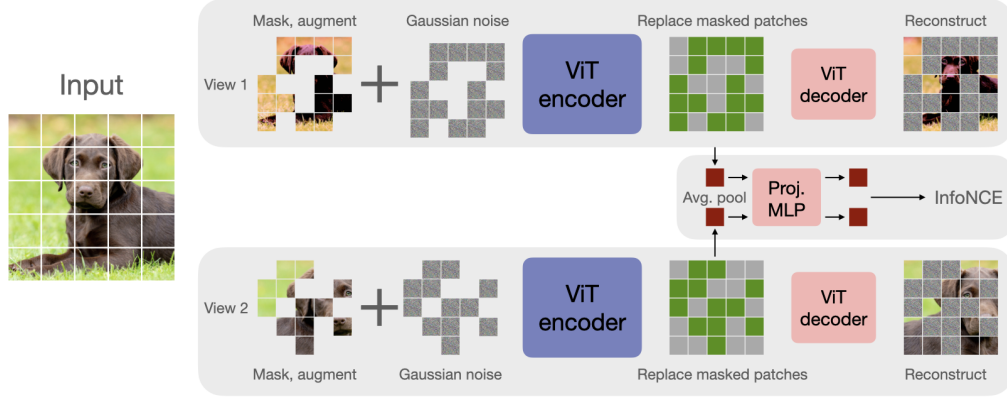


Figure 6.2: **The CAN framework:** Two views of an image are generated, 50% of patches randomly masked in each, and noise is added to patches. An encoder is trained to solve three tasks: 1) **Reconstruction:** encoded patches are passed to a decoder that reconstructs missing patches, 2) **Denoise:** reconstructs the noise added to unmasked patches, and 3) **Contrast:** pooled patches are passed to a contrastive loss, using in-batch samples as negatives [34].

6.3 A simple contrastive masked autoencoder framework

Our approach is a minimal synthesis of contrastive learning, the masked autoencoder (MAE) [90], and the denoising loss used in the training of diffusion models. We focus on simplicity and scalability, aiming to design a hybrid with as few complex or costly components as possible. We also aim to minimize *wasted* computation: in particular, the MAE decoder requires reconstructions of all patches, but only those of masked patches are used in the loss, a fact that CAN exploits. Below, first we detail the basic pipeline of generating views and passing masked inputs through the encoder and decoder, then explain the three objectives we use: contrastive, reconstruction, and denoising. The penultimate section in this chapter describes the combined objective, and the final section discusses scalability.

6.3.1 Overview of method

Given a batch of n images $\{\mathbf{x}\}_{i=1}^n$, we generate two views $\mathbf{x}_i^1, \mathbf{x}_i^2 \in \mathbb{R}^{h \times w \times 3}$ of each image without supervision using the same data augmentations as [34]. Each image is then split into $T = (h/p) \times (w/p)$ non-overlapping patches of size $p \times p$: $\mathbf{x}_{i,\text{patch}}^1, \mathbf{x}_{i,\text{patch}}^2 \in \mathbb{R}^{T \times p \times p \times 3}$ in preparation for input to the ViT encoder. We always assume that p divides h and w . Two masks $\mathbf{M}_i^1, \mathbf{M}_i^2 \in \{0, 1\}^T$ are independently generated, with a 1 in coordinate $t \in \{1, \dots, T\}$ indicating that the t -th patch is masked. Each patch is masked independently with probability r , conditioned on always having exactly $T' = r \cdot T$ patches masked, which we assume is an integer. In all CAN experiments our default masking rate is $r = 50\%$ unless explicitly stated otherwise (note that for all MAE results we follow the exact settings as in [90] using the default $r = 75\%$). Following [90], only the $T - T'$ *unmasked* patches are passed to the ViT encoder, which processes the two views in parallel. Masking a large fraction of patches from both views makes our method much more efficient (see Table 6.1) than contrastive methods that use two full views and recent works that use one full view and one masked view [4, 111]. Finally, we collect the embeddings of unmasked tokens $\mathbf{z}_i^1, \mathbf{z}_i^2 \in \mathbb{R}^{(T-T') \times d}$ and reshape into $T \times d$ tensors by adding a learned [M] embedding to positions corresponding to masked tokens. The result is passed through a comparatively lightweight ViT decoder to produce outputs $\hat{\mathbf{x}}_i^1, \hat{\mathbf{x}}_i^2$ in image space $\mathbb{R}^{h \times w \times 3}$.

6.3.2 Contrastive learning objective

The embeddings $\mathbf{z}_i^1, \mathbf{z}_i^2 \in \mathbb{R}^{(T-T') \times d}$ returned by the encoder are pooled via a simple mean along the first dimension to form d -dimensional embeddings. These embeddings are then passed through a lightweight MLP projection head that maps into a lower dimension space $\mathbb{R}^r$, $r < d$, and normalized to unit length to produce embeddings $\mathbf{u}_i^1, \mathbf{u}_i^2 \in \mathbb{R}^r$ for $i = 1, \dots, n$. For the i th batch item we collect the other $2n - 2$ samples in-batch $\mathcal{N}_i = \{\mathbf{u}_j^1, \mathbf{u}_j^2\}_{j \neq i}$ to use as negatives, and compute the $\mathcal{L}_{\text{InfoNCE}}$ loss:

$$\frac{1}{2n} \sum_{v=1,2} \sum_{i=1}^n -\log \frac{e^{\mathbf{u}_i^1 \top \mathbf{u}_i^2 / \tau}}{e^{\mathbf{u}_i^1 \top \mathbf{u}_i^2 / \tau} + \sum_{\mathbf{u}^- \in \mathcal{N}_i} e^{\mathbf{u}_i^v \top \mathbf{u}^- / \tau}}$$

where $\tau > 0$ is a temperature parameter, defaulting to 0.1. Our choice of InfoNCE objective is justified by recent work [130] that found that a simple InfoNCE objective as in SimCLR scales to large dataset better than methods such as BYOL [86] or DINO [23].

6.3.3 Patch reconstruction objective

The outputs $\hat{\mathbf{x}}_i^1, \hat{\mathbf{x}}_i^2$, $i = 1, \dots, n$ of the ViT decoder are trained to reconstruct the missing patches of each image. As in [90], we find it best to only compute the reconstruction loss on masked patches:

$$\mathcal{L}_{\text{rec}} = \frac{1}{2n} \sum_{v=1,2} \sum_{i=1}^n \|\mathbf{M}_i^v \circ (\mathbf{x}_i^v - \hat{\mathbf{x}}_i^v)\|_2^2$$

where $\circ$ multiplies all pixels in the t th patch of the residual image $\mathbf{x}_i^v - \hat{\mathbf{x}}_i^v$ by $(\mathbf{M}_i^v)_t \in \{0, 1\}$.

Whilst computing the loss only on masked patches gives better performance, it indicates wasted computation since the decoder also produces reconstructions for unmasked patches. To avoid waste we propose an alternative objective specifically for unmasked patches, which we discuss next.

6.3.4 Denoising objective

Inspired by the significant advances in diffusion modelling using *denoising* training objectives [108, 127] and their equivalent score-based counterparts [210, 235] we revisit the suitability of denoising for self-supervised learning. We add independent isotropic Gaussian noise to each image $\mathbf{x}_i^v \leftarrow \mathbf{x}_i^v + \sigma_i^v \mathbf{e}_i^v$ with $\mathbf{e}_i^v \sim \mathcal{N}(\mathbf{0}, I)$ and σ_i^v uniformly sampled from an interval $[0, \sigma_{\max}]$. This noisy input is masked and passed to the encoder as described in Section 6.3.1. When passing encoded patches to the decoder we make a small addition to the method in Section 6.3.1 to provide the decoder with information on the noise level σ_i^v to help it separate noise from the ground truth image. This is motivated by denoising diffusion methods, which pass both the noisy image and the noise level as inputs to the denoising model [108]. We approach this by using σ_i^v as a positional encoding in the decoder, similarly to [233]. First we produce a sinusoidal embedding of $\sigma_i^v \in \mathbb{R}^d$, which is passed through a lightweight 2 layer MLP with ReLU activations of constant width d to produce a (learnable) embedding $\mathbf{p}_i^v \in \mathbb{R}^d$, whose dimension matches the latent dimension of

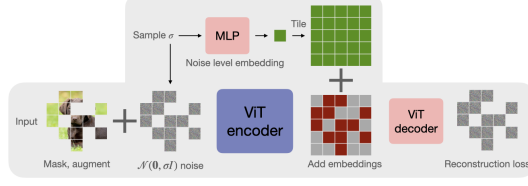


Figure 6.3: **Denoising:** Both the encoded patches and the noise level σ are passed to the decoder by passing σ through an MLP, and adding the result to each embedded token.

$\mathbf{z}_i^v \in \mathbb{R}^{T \times d}$. We add the result to each embedded token (including missing tokens [M]) to provide noise-level information: $(\mathbf{z}_i^v)_t \leftarrow (\mathbf{z}_i^v)_t + \mathbf{p}_i^v$ for $t = 1 \dots T$, and pass the result to the decoder producing $\hat{\mathbf{x}}_i^v$. We define our denoising loss function, which is computed only on unmasked pixels:

$$\mathcal{L}_{\text{denoise}} = \frac{1}{2n} \sum_{v=1,2} \sum_{i=1}^n \|(1 - \mathbf{M}_i^v) \circ (\sigma_i^v \mathbf{e}_i^v - \hat{\mathbf{x}}_i^v)\|_2^2$$

where, $\circ$ multiplies pixels by the patch-level masking as in Section 6.3.3. Note that the reconstruction loss $\mathcal{L}_{\text{rec}}$ still uses the *clean* input $\mathbf{x}$ as its target, with no noise added. The denoising loss is extremely lightweight, introducing only a very small overhead due to the MLP. We emphasize that the reconstruction of noise patches comes at zero additional cost since the decoder produces reconstructions of all patches, both masked and unmasked, but only reconstructions of masked patches are used in $\mathcal{L}_{\text{rec}}$. Finally, it has been observed in the diffusion modelling literature that although it is equivalent to train a denoising model to estimate the noise $\mathbf{e}$, or to estimate the clean input $\mathbf{x}$ [235], there is an empirical gap, with noise target faring better. While we do not pursue it further, our testing corroborates this.

6.3.5 The combined objective function

The overall CAN objective trains the encoder and decoder to optimize three losses combined:

$$\mathcal{L}_{\text{CAN}} = \lambda_{\text{InfoNCE}} \mathcal{L}_{\text{InfoNCE}} + \lambda_{\text{rec}} \mathcal{L}_{\text{rec}} + \lambda_{\text{denoise}} \mathcal{L}_{\text{denoise}}$$

where $0 \leq \lambda_{\text{InfoNCE}}, \lambda_{\text{rec}}, \lambda_{\text{denoise}}$, and $\lambda_{\text{InfoNCE}} + \lambda_{\text{rec}} + \lambda_{\text{denoise}} = 1$ weight the objectives. In practice we parameterize the weights by eliminating one variable using the equality constraint, taking: $\lambda_{\text{rec}} = (1 - \lambda_{\text{InfoNCE}}) \cdot \lambda$ and $\lambda_{\text{denoise}} = (1 - \lambda_{\text{InfoNCE}}) \cdot (1 - \lambda)$ where $0 \leq \lambda \leq 1$. This parameterization makes it easy to control the weighting between the two reconstruction losses $\mathcal{L}_{\text{rec}}, \mathcal{L}_{\text{denoise}}$ on the one hand, and the contrastive loss $\mathcal{L}_{\text{InfoNCE}}$ on the other. We find that performance is robust to the choice of λ , and many choices of λ_{InfoNCE} also work well (see Section 6.5).

6.3.6 Discussion on Efficiency

The efficiency of CAN arises from masking 50% of both views. We also omit certain design choices in the interests of efficiency: we do not use a momentum encoder or multiple views (multi-crop). Each of these components tends to add significant ($2\times$ or more) expense to training. Even without these components CAN achieves strong performance, outperforming its key constituent parts SimCLR and MAE.

	Architecture	Epochs	IN-1K top-1
MoCLR [218]	R50	5000	67.6
BYOL [86]	R50	5000	67.9
DnC [218]	R50	1000	67.9
DnC [218]	R50	4500	70.7
MoCLR [218]	R200×2	5000	74.2
DnC [218]	R200×2	3000	77.3
MAE [†] [90]	ViT-L	1600	50.5
MAE [†] [90]	ViT-L	5000	64.1
SimCLR [†] [34]	ViT-B	800	65.8
SimCLR [†] [34]	ViT-L	800	72.6
SimCLR [†] [34]	ViT-L	1600	73.1
SimCLR [†] [34]	ViT-L	5000	73.4
CAN (ours)	ViT-B	800	67.1
CAN (ours)	ViT-L	800	72.8
CAN (ours)	ViT-L	1600	74.3
CAN (ours)	ViT-L	3000	75.3
CAN (ours)	ViT-L	5000	75.4

Table 6.1: **JFT-300M pre-training:** Comparison to the state of the art on ImageNet linear probe. CAN outperforms all methods except DnC, which uses a complicated multi-stage training process. Computation is measured as ImageNet-equivalent epochs. [†]Our implementation of [34] and [90].

6.4 Results

6.4.1 Pre-training on uncurated data: JFT-300M

A key promise of self-supervised learning is to allow models to be trained on extremely large scale image datasets collected from the Web. Not only is such data likely to be *unannotated*, but also *uncurated*: images containing many objects, variable lighting, artifacts (e.g., watermarks) and so on. The large variation in images found online presents a major challenge to self-supervised learning, and it is not guaranteed that methods that work well on curated (and comparatively smaller) datasets such as ImageNet will work equally well on less curated data. To study how CAN scales to large datasets we use JFT-300M [213], a dataset of around 300 million images.

Setup. Training time is measured in ImageNet-equivalent epochs: 1 epoch equals $1281167/[\text{batch size}]$ steps, the number of steps in one IN-1K epoch. Models are evaluated using linear probe and finetuning on IN-1K. All hyperparameters were tuned on IN-1K, besides learning rate and weight decay which we cut by a factor of 4 and 2 respectively to stabilize training on JFT-300M. See Section 6.9 and Section 6.5 for details.

Results. Figure 6.1 compares CAN to SimCLR and MAE baselines using ViT-L models. CAN achieves a much better trade-off between efficiency (measured in FLOPs) and performance using ViT-L models for all three methods: SimCLR uses 70% more FLOPs than CAN, which consistently outperforms both SimCLR and MAE: for training ViT-L models for 5000 epochs, CAN achieves an IN-1K linear probe performance of 75.4%, compared to 73.4% for SimCLR and 64.1% for MAE. The relatively poorer linear probe performance of MAE on JFT-300M highlights the non-triviality of scaling from IN-1K to larger datasets and suggests that while MAE is scalable for *model size*, scalability to larger *datasets* requires further study. Figure 6.1 (right) gives finetuning results. CAN performs favourably: for a 5000 epoch pre-training schedule, CAN achieves an IN-1K linear probe performance of 86.1%, compared to 85.5% for SimCLR and 85.4% for MAE. CAN also enjoys better scaling with training schedule length than either MAE or SimCLR, with the difference in performance becoming *larger* for longer schedules. We hypothesize that this is not coincidental, and that strong pre-training tasks like CAN play an important role in scalability.

We also compare CAN to the current state of the art on JFT-300M pre-

training in Table 6.1. Our best performance, 75.4% with ViT-L outperforms all methods besides DnC, with 77.3% [218] with R200×2. However we note that CAN is *considerably* simpler than DnC, which involves training 10 separate “expert” models (each as large as the final model), and then using MoCLR (an improvement of SimCLR that adds a momentum encoder and more), using distillation to produce a single final model. Our calculations suggest that training a ViT-L with CAN is about $3\times$ faster than training the considerably smaller ResNet50 with DnC in terms of wall clock time (see Section 6.8 for explanation). CAN on ViT-L outperforms MoCLR with R200×2 backbone (similar parameter counts), where we note that MoCLR performs as well or better than BYOL and MoCo-v3 on IN-1K [218].

6.4.2 Pre-training on ImageNet-21K

We also consider the performance of CAN on pre-training on ImageNet-21K (IN-21K), a publicly available dataset of 14.2 million images [57]. We use the same hyperparameter settings as JFT-300M. We run a full set of evaluations on linear probe (Table 6.6), robustness (Figure 6.15), and few-shot learning (Figure 6.16) (see Sections 6.4.4 and 6.4.5 for details on few-shot and robustness evaluations). Results are reported in Section 6.7.1 due to space constraints. CAN also performs well with IN-21K pre-training, with CAN finetuned on IN-1K showing better robustness than MAE and SimCLR in 8 out of 8 cases, and CAN achieving best 25-shot performance on 6 out of 9 datasets.

Method	Pre-training epochs	Encoder	No Additional params.	Masked image	Finetune	Linear probe
<i>from scratch</i> [†]	100	ViT-B			79.1	—
MoCo-v3 [41]	300	ViT-B			83.0	76.7
DINO [27]	1600	ViT-B			82.8	78.2
CIM [69]	300	ViT-B			83.1	—
CAE [38]	800	ViT-B			83.8	68.6
CAE [38]	1600	ViT-B			83.9	70.4
BEiT [8]	800	ViT-B			83.2	37.6*
SimMIM [257]	800	ViT-B			83.8	56.7
MAE [90]	800	ViT-B			83.1	—
MAE [90]	1600	ViT-B			83.6	68.0
CAN (ours)	800	ViT-B			83.4	74.0
CAN (ours)	1600	ViT-B			83.6	74.8
SimCLR [†] [34]	800	ViT-L			83.4	73.9
MAE [90]	800	ViT-L			84.9	73.5
MAE [†] [90]	800	ViT-L			83.7	71.4
CAN (ours)	800	ViT-L			84.7	76.2

Table 6.2: **Finetune and linear probe results with pre-training on ImageNet-1K.** Note that CAN does not use multi-crop augmentation or momentum encoder. [†]Our implementation of [34] and [90]. *Quoted from [38].

6.4.3 Pre-training on ImageNet-1K

Next we evaluate our method using ImageNet (IN-1K) pre-training to verify that it is also competitive in this setting. Results in Table 6.2 record the top-1 accuracy on IN-1K classification of finetuned models and linear probes. Finetuning CAN achieves 83.6% with ViT-B, outperforming other contrastive approaches such as MoCo-v3 (83.0%), and is competitive with other state-of-the-art approaches such as CAE (83.9%). The linear probe performance of CAN is 74.8% using ViT-B, beating all masked image modelling methods, the best of which is CAE with 70.4% [38]. CAN is only outperformed by MoCo-v3 and DINO, which use momentum encoders and two full image views, and in the case of DINO 10 multi-crop views. Note that the *masked image* column indicates whether a method uses one or more full image views as input to the model, and the *no additional parameters* column indicates whether a method relies on other parameters besides the main encoder,

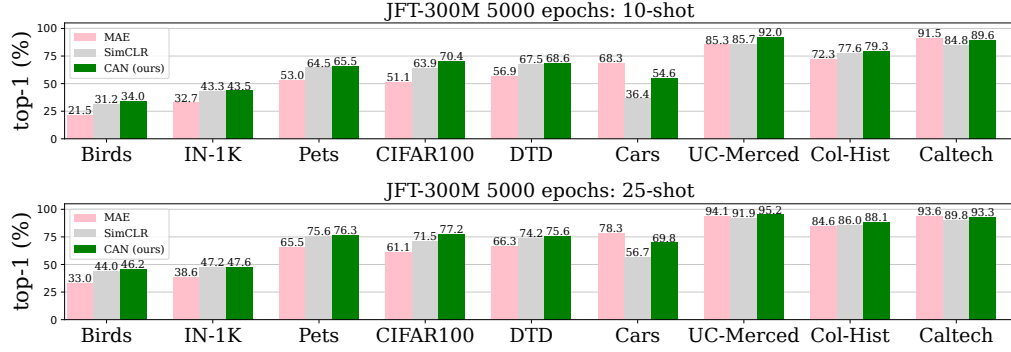


Figure 6.4: **Few-shot:** ViT-L models pre-trained on JFT-300M for 5000 epochs are evaluated on 9 datasets in few-shot setting (10-shot and 25-shot). CAN outperforms MAE and SimCLR.

e.g., from a pre-trained tokenizer, or a momentum updated target encoder. We also report results for our MAE implementation, which approximately matches the numbers reported in [90], validating our MAE results on JFT-300M.

6.4.4 Few-shot learning

We use linear probes to evaluate suitability of CAN for few-shot learning, following the protocol of [64]. We use the models pre-trained on JFT-300M for 5000 epochs whose ImageNet performance is recorded in Figure 6.1. Results in Figure 6.4 for few-shot transfer learning on 9 other datasets show that the superior performance on IN-1K translates to strong performance on other tasks. We also note that our 25-shot ViT-L models beat *full-shot* both DnC and BYOL ResNet50 models (also trained for 5000 epochs on JFT-300M) on 6 out of 8 datasets [218]. See Section 6.7 for many additional results, including pre-training on IN-21K.

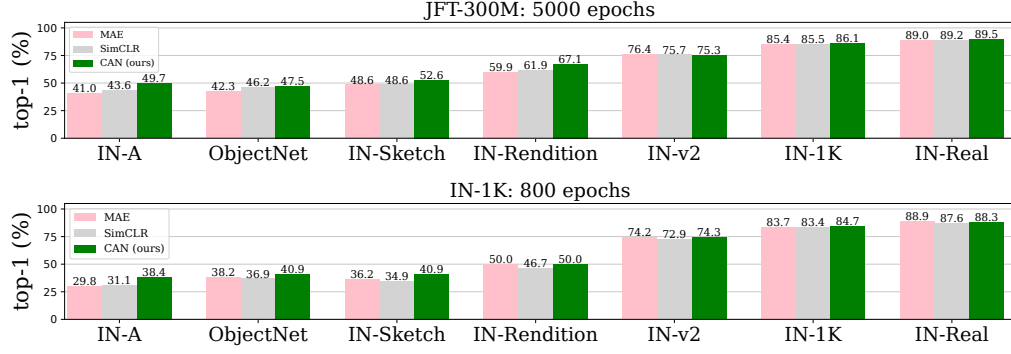


Figure 6.5: **Robustness:** Evaluating performance under distribution shifts with respect to models finetuned on IN-1K. Validation performance of ViT-L models is reported on 7 different datasets.

6.4.5 Robustness to distribution shift

Finally, we consider the robustness of CAN to distribution shifts. We use ViT-L backbones trained for 5000 epochs on JFT-300M, which have been finetuned on IN-1K. Model performance is evaluated on a number of different validation sets with the same 1000 classes as IN-1K [163]. Figure 6.5 reports results on the following 7 validation sets, which cover a large variety of distribution shifts: original IN-1K [57], IN-v2 [189], IN-ReaL [10], IN-Adversarial [102], IN-Rendition [97], ObjectNet [9]. CAN performs favourably under both JFT-300M, IN-21K and IN-1K pre-training, beating SimCLR and MAE baselines in nearly all cases. See Section 6.7 for additional results.

6.5 Hyperparameter analysis

We study the different components of CAN to better understand the effect of the different mechanisms, and to determine optimal parameter configurations. All ablations use ViT-B models trained for 100 epochs on IN-1K and evaluated with a

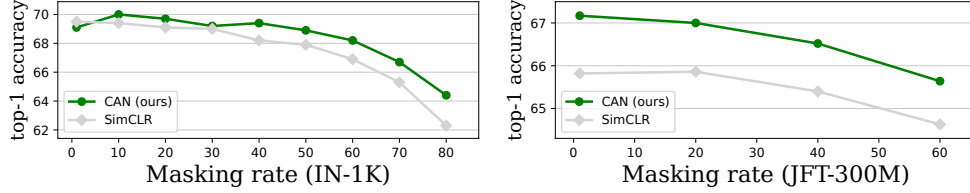


Figure 6.6: CAN and SimCLR with different masking rates. ViT-B models are pre-trained for 100 epochs on IN-1K (left), and 800 epochs on JFT-300M (right).

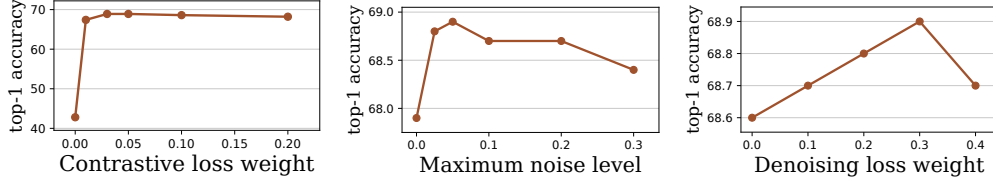


Figure 6.7: ViT-B models pre-trained on IN-1K for 100 epochs. **Left:** The best contrastive loss weight is small but non-negative. **Middle:** A wide range of $\sigma_{\max}$ values improve over no-noise. **Right:** Performance is not sensitive to the denoising loss weight.

linear probe on IN-1K unless explicitly said otherwise. We use the best loss weights and noise level in these experiments for experiments in Section 6.4.

Complementarity of contrastive and reconstruction losses. A key hypothesis motivating our work is that contrastive learning and masked autoencoder reconstruction may not only be compatible training objectives, but are *complementary* ones. Table 6.3 compares the final training value of the contrastive $\mathcal{L}_{\text{InfoNCE}}$ and reconstruction $\mathcal{L}_{\text{rec}}$ when jointly trained (i.e., CAN) compared to only optimizing $\mathcal{L}_{\text{InfoNCE}}$ (SimCLR) or only $\mathcal{L}_{\text{rec}}$ (MAE). The results support the hypothesis: joint training achieves a lower loss on *both* objectives compared to individual training.

Ablating CAN loss terms. CAN is comprised of three components: (C) contrastive, (A) masked autoencoder, and (N) denoising losses. We ablate each of the three components in Table 6.5, setting the loss weight to zero to “remove” a component. We use ViT-B models pre-trained for 100 epochs. Removing any

Method	Contrastive loss ↓	Reconstruction loss ↓
SimCLR	9.157	—
MAE	—	0.1658
CAN (ours)	9.143	0.1633

Table 6.3: **Loss complementarity.** CAN training achieves *lower* training loss for both contrastive and reconstruction than individual training. All methods use 50% masking for fair comparison.

component leads to worse performance, with contrastive loss hurting the most.

Denoising method. Table 6.4 studies the effect of each of the components of the denoising method. We use ViT-B models trained for 100 epochs on ImageNet, and consider four settings, each adding in more parts of the method: 1) CAN with no denoising, 2) adding noise to the input only, 3) adding noise and using the denoising loss, and 4) the full method with all of the described components, including using σ_i^v as a positional encoding in the decoder. Results show that simply adding noise as a data augmentation improves performance by 0.7%, which can be improved to 1% by adding a reconstruction loss with noise level passed as an argument. The noise level argument is necessary: the reconstruction loss without noise level argument performs worse (68.4%) than noise with no reconstruction at all (68.6%). We emphasize that the improvement from denoising comes at minimal run time and memory cost, since it uses reconstructions produced by the decoder, which in the case of MAE are simply thrown away unused. We also tried predicting the clean patches instead of noise, and found it worked poorly, corroborating similar findings in the diffusion literature.

Masking rate. Figure 6.6 reports the behavior of CAN and SimCLR under

None	+noise	+noise, +loss	Full
67.9	68.6	68.4	68.9

Table 6.4: **Denoising objective.** “Full” denotes the entire method as described in Section 6.3.4

AN	CN	CA	CAN (full)
42.8	68.5	67.9	68.9

Table 6.5: **CAN loss terms.** We remove each of the three loss terms in CAN one by one.

different masking rates on IN-1K and JFT-300M pre-training (for JFT-300M we use 800 epochs). The performance of SimCLR decreases as the masking rate increases, suggesting that masking is not an effective data augmentation. In contrast, performance of CAN peaks at a non-zero masking rate, but at a much lower rate than the 75% used by MAE on IN-1K. This occurs since very low masking rates are preferred by the contrastive part of CAN, but severely damage the autoencoder part as it can learn trivial solutions. The considerable efficiency improvement from masking 50% of patches more than compensates for the small drop in performance for a fixed number of epochs.

Contrastive loss weight. We vary the weighting λ_{InfoNCE} used to weight the contribution of the contrastive and reconstruction losses. Recall that larger λ_{InfoNCE} places higher weight on the contrastive loss. Results in Figure 6.7 show that the best weight is $\lambda_{\text{InfoNCE}} = 0.03$, which approximately balances the magnitudes of the two terms (see Table 6.3).

Denoising loss weight and noise level. We study the noise level interval $[0, \sigma_{\text{max}}]$ from which to sample input noise, and the weight λ balancing the denoising

and reconstruction losses. Results in Fig. 6.7 show that the best maximum noise level is $\sigma_{\max} = 0.05$, and that similar performance is attained for different weights on the denoising loss.

6.6 Discussion

We present CAN, a simple, efficient and scalable self-supervised method for visual representation learning. CAN combines ideas from contrastive learning, masked autoencoding, and diffusion denoising into a single high-performing method. Extensive empirical results show that CAN scales with minimal changes to the large uncured datasets, outperforming SimCLR and MAE methods on a wide range of downstream tasks and evaluations, including ImageNet linear probes, few-shot, robustness, and finetuning. Our results suggests that combining different self-supervised methods can produce better results than the constituent parts alone. Further exploration of this search space appears a promising avenue for future work.

6.7 Additional Transfer Learning Results

We report additional results for few-shot learning and robustness.

Robustness: Section 6.4.5 reports robustness results for ViT-L models pre-trained on JFT-300M for 5000 epochs, and ViT-L models pre-trained on IN-1K for 800 epochs. In both cases we report the performance of the models after finetuning on IN-1K.

Here we report the same robustness results for ViT-L models trained on JFT-

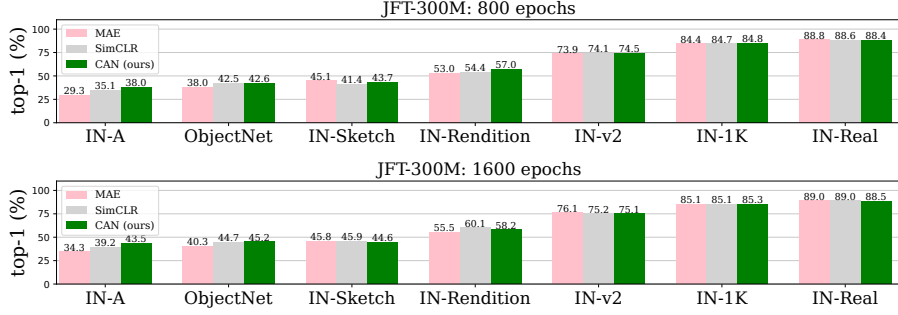


Figure 6.8: ViT-L models pre-trained on JFT-300M for 800 and 1600 epochs respectively, evaluated on 7 datasets with distribution shifts from IN-1K.

300M for 1600 and 800 epochs (Figure 6.8), and ViT-B models pre-trained for 800 epochs (Figure 6.9). Figure 6.9 also compares our ViT-B model to ViT-B models trained from scratch on ImageNet. We find that our model is considerably more robust than training with cross-entropy and Mixup from scratch, and also outperforms PyramidAT [105], an adversarial training method that introduces significant overheads compared to standard cross-entropy training. We emphasize that here there are two differences in the training: a) the training algorithm itself, and b) the data seen by the model. Our model sees extra JFT-300M data not seen by the other two approaches. This means that the methods are not exactly comparable. It is, however, a realistic setting showing the benefits to robustness of pre-training on large datasets.

Few shot: Section 6.4.4 reports 10- and 25-shot results for ViT-L models pre-trained on JFT-300M for 5000 epochs. Here we report 1- and 5-shot results for the same models in Figure 6.10. We additionally show the full set of {1, 5, 10, 25}-shot results for ViT-L models pre-trained on JFT-300M for 800 and 1600 epochs (Figures 6.11 and 6.12 respectively), ViT-B models pre-trained on JFT-300M for 800 epochs

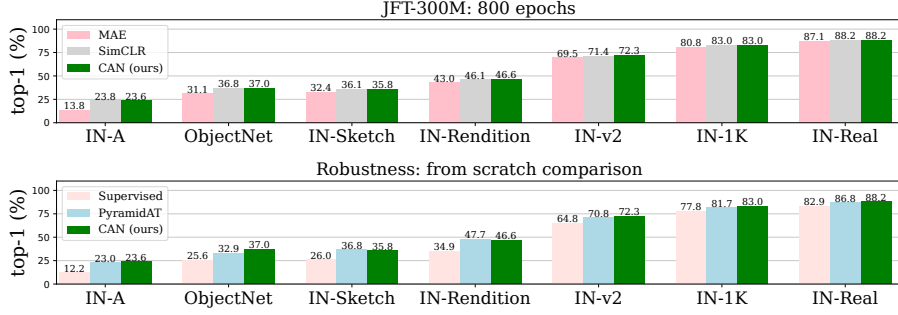


Figure 6.9: **Top:** ViT-B models pre-trained on JFT-300M for 800 epochs, evaluated on 7 datasets with distribution shifts from IN-1K. **Bottom:** Comparison of our JFT-300M pre-trained ViT-B model to training ViT-B from scratch on IN-1K. We compare to standard supervised cross-entropy training with Mixup, and to PyramidAT [105], which uses an adversarial training method. CAN considerably outperforms supervised training, and beats PyramidAT in 6 out of 7 cases without requiring adversarial training.

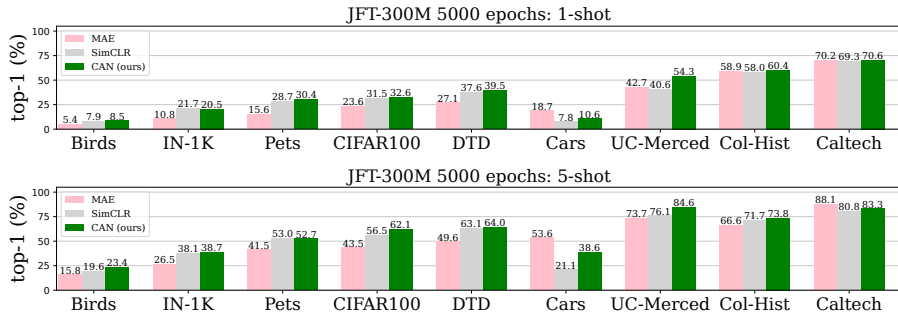


Figure 6.10: **Few shot:** ViT-L models pre-trained on JFT-300M for 5000 epochs evaluated on 9 few-shot learning tasks. Results accompany the 10- and 25-shot results in Figure 6.4.

(Figure 6.13), and ViT-L models pre-trained on IN-1K for 800 epochs (Figure 6.14).

We make a number of observations.

1. JFT-300M pre-training often outperforms IN-1K pre-training. Comparing Figures 6.11 and 6.14, JFT-300M yields better 25-shot CAN performance on 6 out of 9 datasets.
2. Model scale helps. Comparing Figures 6.13 and 6.11, ViT-L models perform best in nearly all cases.

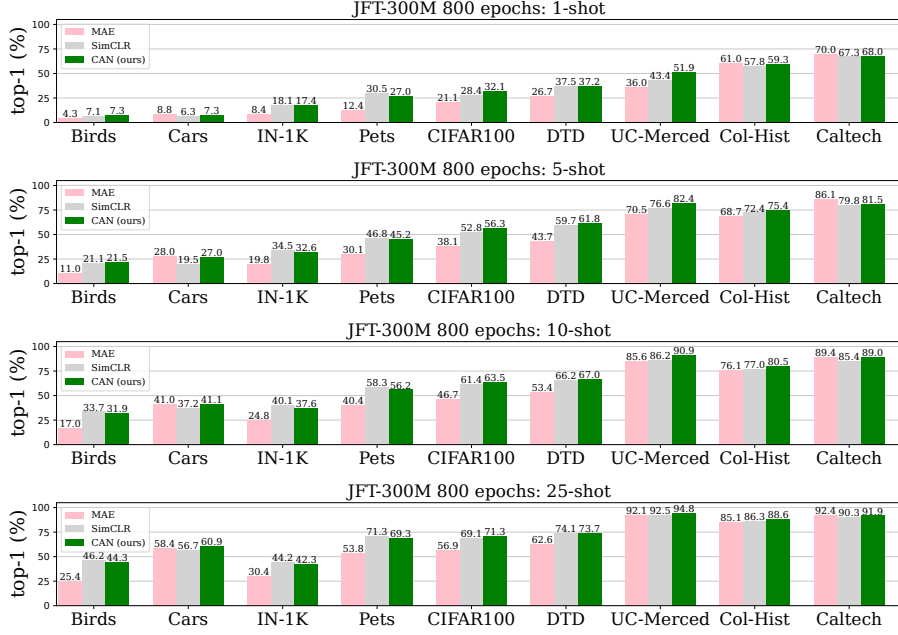


Figure 6.11: **Few shot:** ViT-L models pre-trained on JFT-300M for 800 epochs are evaluated on 9 few-shot learning tasks.

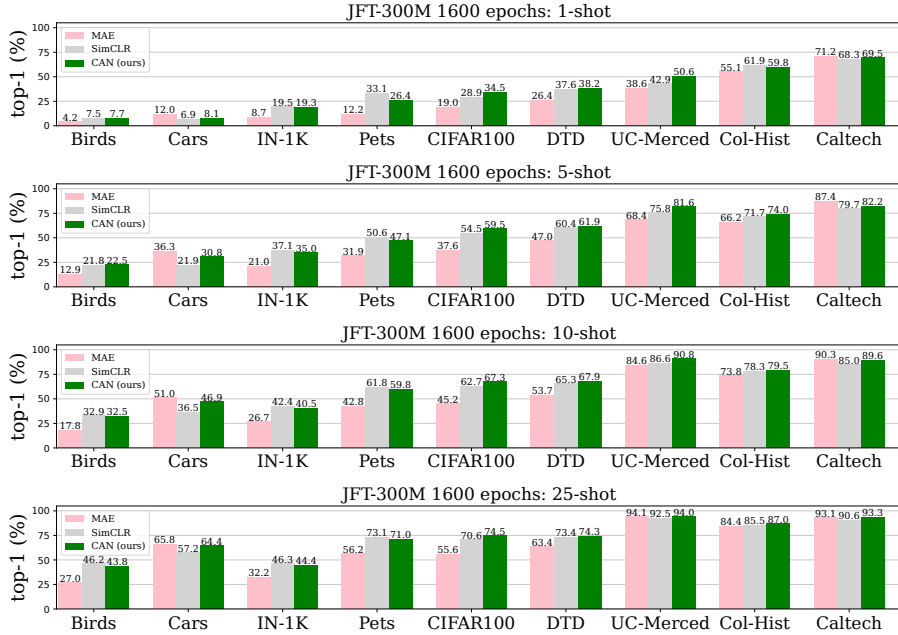


Figure 6.12: **Few shot:** ViT-L models pre-trained on JFT-300M for 1600 epochs are evaluated on 9 few-shot learning tasks.

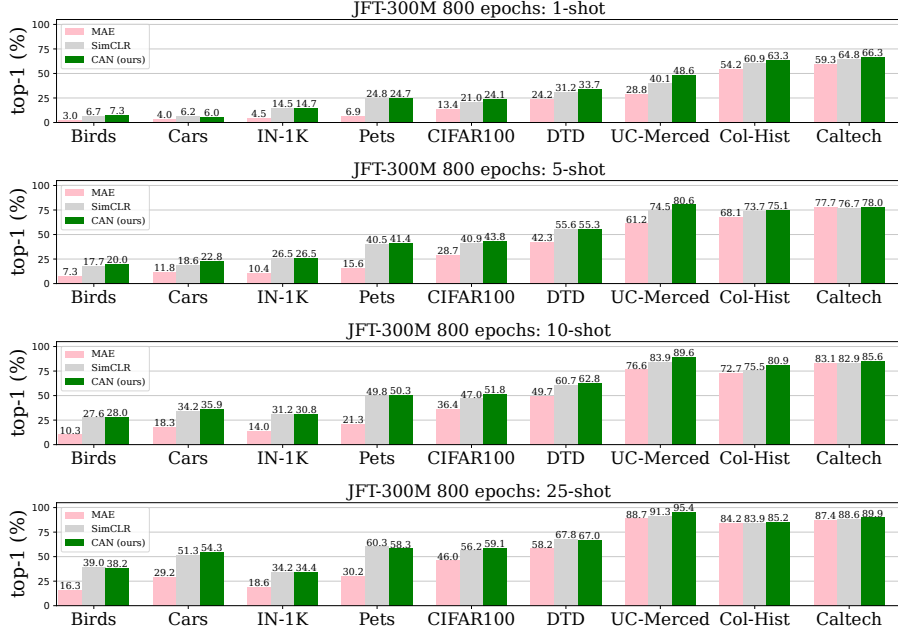


Figure 6.13: **Few shot:** ViT-B models pre-trained on JFT-300M for 800 epochs are evaluated on 9 few-shot learning tasks.

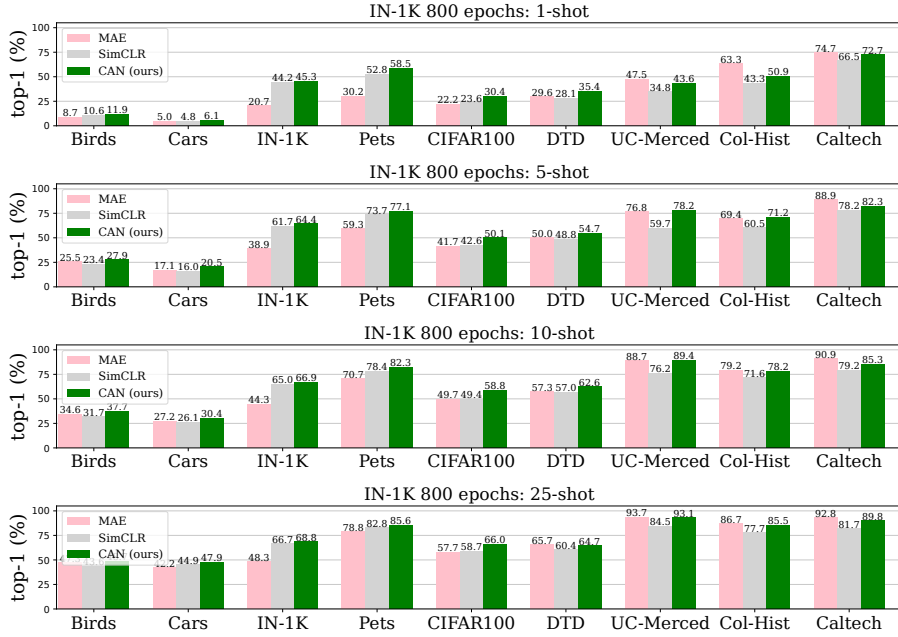


Figure 6.14: **Few shot:** ViT-L models pre-trained on IN-1K for 800 epochs are evaluated on 9 few-shot learning tasks.

3. Across all settings CAN generally performs the best on JFT-300M pre-training.
4. The situation is less consistent on IN-1K pre-training. For instance, although MAE has comparatively poor few-shot performance on IN-1K, it is competitive on others for 25-shot evaluation: in this setting CAN only beats MAE on 5 out of 9 datasets. However, on 10-shot CAN outperforms MAE and SimCLR in 7 out of 9 cases, showing a subtle picture.

6.7.1 ImageNet-21K pre-training.

We also consider the performance of CAN on pre-training on ImageNet-21K (IN-21K), a publicly available dataset of 14.2 million images, grouped into 21,000 different classes [57]. We use the same hyperparameter settings as JFT-300M training to train ViT-L models on IN-21K for 800 (IN-1K equivalent) epochs.

We run a full set of evaluations on finetuning, linear probe, robustness (Figure 6.15), and few-shot learning (Figure 6.16), and linear probe in Table 6.6. We note that while CAN and SimCLR achieve similar linear probe performance, CAN remains much more efficient (see Figure 6.1). Furthermore, we only ran IN-21K training for 800 epochs. This is a training schedule which also led to similar linear probe for CAN and SimCLR when pre-training on JFT-300M. The difference in linear probe between CAN and SimCLR emerged on JFT-300M with longer training runs. Despite this, CAN pre-training on IN-21K with 800 epoch schedule still performs favourably on other downstream robustness and few-shot evaluations.

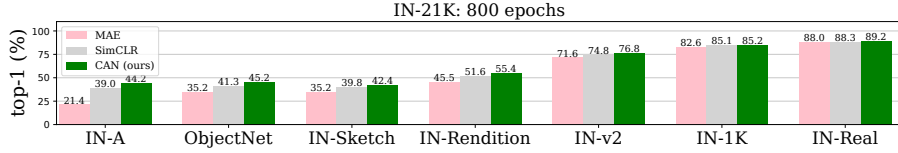


Figure 6.15: **Robustness:** ViT-L models pre-trained on IN-21K for 800 (IN-1K equivalent) epochs are first finetuned on IN-1K. The models are then evaluated on 7 test datasets with different distribution shifts from IN-1K.

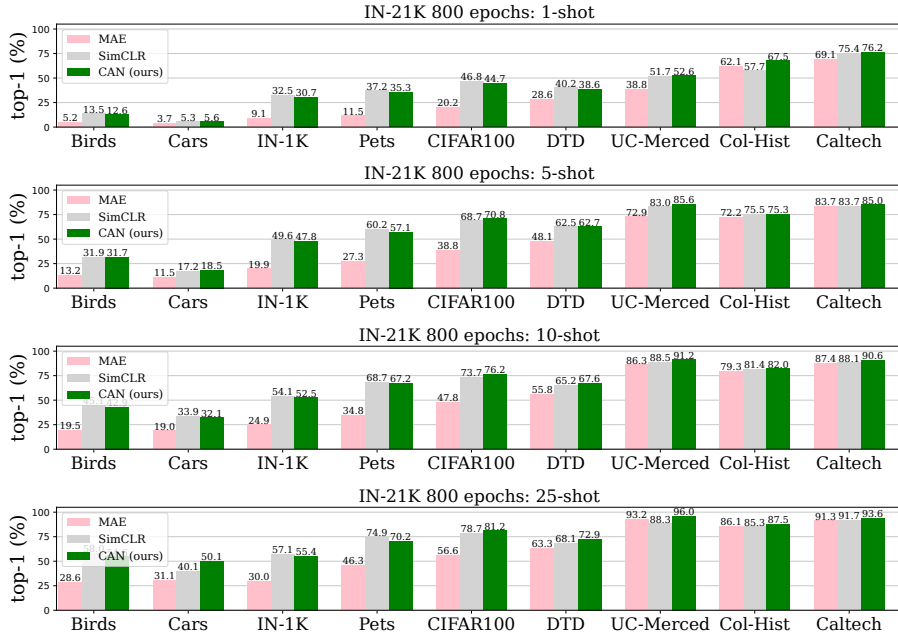


Figure 6.16: **Few shot:** ViT-L models pre-trained on IN-21K for 800 (IN-1K equivalent) epochs are evaluated on 9 few-shot learning tasks.

MAE	SimCLR	CAN
57.8	76.5	76.5

Table 6.6: **IN-21K pre-training.** Linear probe performance on IN-1K. ViT-B models pre-trained for 800 IN-1K equivalent epochs.

6.8 Runtime of CAN compared to DnC

We estimate our method is significantly faster than DnC [218]. We determined this approximate comparison from the following two pieces of information: 1) DnC reports that 3000 ImageNet epochs takes 29 hours on 512 TPUs for a ResNet-50 model (~ 25 M parameters), and 2) 3000 ImageNet epochs of CAN take 78 hours on 64 TPUs for a ViT-L model (~ 300 M parameters). We assume a linear relationship between number of TPUs and runtime. Under this assumption, we estimate that CAN would take approximately 10 hours to train with 512 TPUs, compared to the 29 hours reported by [218] for a model with 1/10th the number of parameters. We emphasize that this is far from an exact comparison and is only intended as a very approximate guide.

6.9 Hyperparameter settings

We list hyperparameters used for CAN pre-training in Table 6.7 and Table 6.8. For preprocessing we closely follow SimCLR [34]. We use the same hyperparameters for SimCLR pre-training. For MAE pre-training, we use the same hyperparameters as listed in [90], except for the use of Glorot uniform initialization instead of LeCun initialization as done in [90]. We found that this provided better performance for our JAX-based MAE implementation. Table 6.11 lists the hyperparameters for finetuning evaluations. We use the same set of hyperparameters for each finetuning each pre-training method, and for both ViT-B and ViT-L model sizes. For linear

probing we list the hyperparameters in Table 6.12 for which we followed the settings in [90]. We use global average pool of the final representation instead of the cls token.

MAE longer training: MAE pre-training for longer training (5000 epochs) on JFT becomes unstable after about 500k steps (training loss oscillates); this results in poorer fine-tuning performance. To overcome this, we decrease the base learning rate by 75% as shown in Table 6.10. However our model CAN is more stable and we use the same hyperparameters across different numbers of epochs.

Few shot training: For few-shot learning we use the same hyperparameters and pipeline as [64]. We use the same pre-processing as was done in [129]. We use a base learning rate of 0.01 and train for 2500 steps, using an input resolution of 384×384 .

Hardware details: We use TPU-v4 for all of our experiments. CAN on ViT-B uses 64 TPUs for a batch size of 4096. SimCLR, on the other hand, uses 128 TPUs for the same batch size, and is more compute intensive than CAN.

Decoder architecture: Our decoder architecture is the same as [90]. We use standard ViT with a decoder depth of 8 and decoder width of 512. We use 16 heads and 2048 as the dimension of the MLP.

Projection head architecture: We use 2 hidden layers in our projection heads. Each layer has a Fully-Connected (FC) layer (dim 4096) followed by BatchNorm (mo-

momentum=0.9) followed by ReLU. After these 2 layers we have a FC layer which transforms the features to 128 dimensions. We apply contrastive learning on top of these 128 dimensional features.

Noise positional encoding architecture: We use a 1 hidden layer MLP with ReLU activations of hidden width and output dimension of 768, equal to the input dimensions of the decoder. During testing we tried different depths and widths, finding performance to be robust to these variations. The input dimension is 256 sinusoidal features, exactly as in [233].

JFT-300M and IN-21K hyperparameters: All hyperparameters were determined by training on IN-1K, and directly transferred to JFT-300M and IN-21K pre-training, with the exception of learning rate and weight decay, which we found needed to be at a lower level for JFT-300M and IN-21K. For all methods we divided the learning rate by a factor of 4, and the weight decay by a factor of 2, except for MAE where we found that the original weight decay tuned on ImageNet worked better. We also tried dividing the learning rate by factors of 2 and 8, but found that 4 worked best for all methods. Specifically, for CAN and SimCLR we used following parameter choices: $wd = 0.1/2 = 0.05$ and $lr = 1.25 \times 10^{-4}/4 = 3.125 \times 10^{-5}$ and for MAE we used $lr = 1.5 \times 10^{-4}/4 = 3.75 \times 10^{-5}$, and tried $wd = 0.05/2 = 0.025$, but found that the original $wd = 0.05$ worked better, so kept this value.

6.9.1 Pre-training hyperparameters

Config	Value
optimizer	AdamW [158]
base learning rate (ViT-B)	2.5e-4
base learning rate (ViT-L)	1.25e-4
weight decay (ViT-B)	0.05
weight decay (ViT-L)	0.1
optimizer momentum	$\beta_1, \beta_2=0.9, 0.95$ [32]
batch size	4096
learning rate schedule	cosine decay [159]
warmup epochs [84]	40
augmentation	RandomResizedCrop, Color Jittering(strength=1.0), GrayScale(probability=0.2), Gaussian Blurring (probability=0.5)
masking rate	0.5

Table 6.7: Hyperparameters for CAN pre-training on ImageNet. Note that we use lower learning rate for ViT-L as compared to ViT-B, following [212]. We use the same hyper-parameters for SimCLR pre-training, except for masking rate that is set to 0.

Config	Value
optimizer	AdamW [158]
base learning rate (ViT-B)	2.5e-4
base learning rate (ViT-L)	3.125e-5
weight decay	0.05
optimizer momentum	$\beta_1, \beta_2=0.9, 0.95$ [32]
batch size	4096
learning rate schedule	cosine decay [159]
warmup epochs [84]	40
augmentation	RandomResizedCrop, Color Jittering(strength=1.0), GrayScale(probability=0.2), Gaussian Blurring (probability=0.5)
masking rate	0.5

Table 6.8: Hyperparameters for CAN pre-training on JFT-300M and IN-21K. Note that we use lower learning rate for ViT-L as compared to ViT-B, following [212]. We use the same hyper-parameters for SimCLR pre-training, except for masking rate that is set to 0.

Config	Value
optimizer	AdamW [158]
base learning rate (ViT-L)	1.5e-4
weight decay	0.05
optimizer momentum	$\beta_1, \beta_2=0.9, 0.95$ [32]
batch size	4096
learning rate schedule	cosine decay [159]
warmup epochs [84]	40
augmentation	RandomResizedCrop
masking rate	0.75

Table 6.9: Hyperparameters for MAE pre-training on IN-1K. These parameters are exactly the same as in [90].

Config	Value
optimizer	AdamW [158]
base learning rate (ViT-L)	3.75e-5
weight decay	0.05
optimizer momentum	$\beta_1, \beta_2=0.9, 0.95$ [32]
batch size	4096
learning rate schedule	cosine decay [159]
warmup epochs [84]	40
augmentation	RandomResizedCrop
masking rate	0.75

Table 6.10: Hyperparameters for MAE pre-training on JFT-300M and IN-21K with ViT-L models. The only difference from the IN-1K configuration is the learning rate, which we reduced to stabilize training.

6.9.2 Finetuning and linear probe hyperparameters

Table 6.11 shows the finetuning recipe we use for CAN and SimCLR. This is the same as the recipe reported for MAE, except for the learning rate, which we found beneficial to lower, and the augmentations, which we found marginally beneficial to strengthen ([90] uses only RandAug). The only other difference is that we always use layer-wise learning rate decay of 0.65, whereas MAE uses 0.65 for ViT-B, and 0.75 for ViT-L. For 5000 epochs of JFT-300M, and 800 epochs of IN-21K pre-training we noticed that CAN and SimCLR finetune was over-fitting, so we reduced the finetuning from 100 epochs to 40 epochs. For all MAE finetuning runs we follow the recipe of the original paper [90] as we found adjusting the learning rate and augmentations not to help for MAE. We did not observe any over-fitting in MAE, so kept the schedule fixed.

For linear probe training we follow the exact same recipe of [90] for all models (see Table 6.12 for details).

Config	Value
optimizer	AdamW [158]
base learning rate	5e-4
weight decay	0.005
optimizer momentum	$\beta_1, \beta_2=0.9, 0.999$ [32]
batch size	1024
learning rate schedule	cosine decay [159]
warmup epochs [84]	5
training epochs	100
label smoothing	0.1
drop path	0.1
layer-wise lr decay	0.65
augmentation	RandomResizedCrop, Flip, RandAug(layers=2, magnitude=9) [53], Random Erase [268](probability=0.25)

Table 6.11: Hyperparameters for finetuning CAN and SimCLR pre-trained models on ImageNet. We use the same hyperparameters for ViT-B and ViT-L, for JFT-300M, IN-21K and IN-1K pre-trained models. For MAE, we use the settings from Table 9 of [90].

Config	Value
optimizer	LARS [262]
base learning rate	0.1
weight decay	0
optimizer momentum	0.9
batch size	16384
learning rate schedule	cosine decay [159]
warmup epochs [84]	10
training epochs	100
batch norm momentum	0.9
label smoothing	0
augmentation	RandomResizedCrop

Table 6.12: Hyperparameters for linear probing pre-trained models on ImageNet. We use the same hyperparameters for ViT-B and ViT-L, and for JFT-300M, IN-21K and IN-1K pre-trained models. Note that these hyperparameters are same as reported in [90].

Chapter 7: Conclusion and Future work

7.1 Conclusion:

This manuscript shows five different ways of improving SSL by addressing issues such overreliance on texture, overfitting to single object images like ImageNet and how to scale up SSL methods to real world large scale datasets like JFT and ImageNet-21k. To reduce the over-reliance on texture we proposed two different methods: first we proposed to use Anisotropic diffusion, which suppress texture information. Empirical results highlight the efficacy of using Anisotropic diffusion, showing significant performance improvements across ten diverse datasets. Notably, a substantial 11.4% enhancement in the performance on Sketch ImageNet suggests that the network captures richer high-level representations when compared to models trained solely on ImageNet. Secondly we generate patch-based and texture-based augmentations, which we use for crafting negative samples while preserving local features. Our results demonstrate that representations learned through contrastive learning with these negative samples rely less on local features, leading to improved generalization under out-of-distribution (OOD) settings.

We then discuss two approaches which help in solving SSL on real-world multi object datasets like OpenImages. First, we introduce object-aware cropping as a

simple, fast, and highly effective data augmentation alternative to random scene cropping. Through a series of experiments, we establish that object cropping outperforms scene cropping in the context of self-supervised pre-training, benefiting tasks such as classification, object detection, and semantic segmentation across multiple datasets. Notably, this approach seamlessly integrates into most self-supervised learning pipelines, enhancing their effectiveness. Secondly we present the Hyperbolic Contrastive Learning (HCL) framework, a unique contrastive learning approach that learns visual representations for both objects and scenes within a unified representation space. The cornerstone of our method is a hyperbolic contrastive objective built on an object-centric scene hierarchy. HCL’s effectiveness is showcased across various benchmarks, including image classification, object detection, and semantic segmentation. Furthermore, we demonstrate the versatility of the representations in zero-shot settings, ranging from detecting out-of-context objects to quantifying label uncertainty in datasets like ImageNet. This work aims to inspire future endeavors in building a more comprehensive visual representation space, drawing attention to the potential of non-Euclidean representation learning.

We finally introduce CAN which is an simple method that scales really well on large scale datasets like ImageNet-21k and JFT-300m. CAN integrates concepts from contrastive learning, masked autoencoding, and diffusion denoising into a unified and high-performing approach. Through extensive empirical testing, we find that CAN exhibits scalability with minimal adjustments on large, uncured datasets, delivering substantial performance improvements compared to SimCLR and MAE methods across a diverse spectrum of downstream tasks and evaluations,

including linear probes, few-shot learning, robustness assessments, and fine-tuning. Our findings underscore the synergistic relationship between contrastive and reconstruction principles, demonstrating how they can mutually reinforce one another.

7.2 Future work:

Overall SSL has led to more robust and less biased representations, and the use of a various methods proposed in this manuscript has led to more effective object recognition and scene understanding. Now we discuss some of the future work in SSL:

7.2.1 Self-Supervised Learning and Generative Learning on multi-object scene datasets:

While we have made some progress in learning scene-level information using SSL, the generative capabilities of existing methods remain limited, with current state-of-the-art methods primarily focused on single objects [196]. To address this limitation, we propose extending MAGE [145] through two key modifications. Firstly, we will develop a more general-purpose tokenizer that is not dependent on any particular dataset, building on the approach described in [211]. Secondly, we will develop separate networks for object and scene generation, with the former focused on generating individual objects and the latter leveraging multiple scene networks to generate holistic scene images.

7.2.2 Self-Supervised Learning on Segment Anything:

Segment Anything Model represents a recent advancement, boasting approximately ~ 11 million images and 1 billion masks for semantic segmentation. This is a substantial foundational model in the realm of computer vision semantic segmentation. Ideally, the success of Segment Anything should have been achieved through SSL without the need for such extensive annotations. To tackle this issue, we can utilize the SAM dataset, harnessing robust object-level features from DiNo [27]. Unlike other methods outlined in this manuscript, we can bootstrap the DiNO features to create unsupervised masks. These bootstrapped masks could then be employed to develop a scalable SSL method that excels in the context of Segment Anything.

7.2.3 Self-Supervised Learning in Robotics:

The promise of SSL has yet to be fully realized in the realm of robotics. Robotics datasets remain relatively small [55, 270], often consisting of around 100 videos, primarily due to the challenges associated with heavy annotations. In this domain, SSL has encountered challenges, mainly stemming from the insufficient richness of features learned from real-world distributions and domain gaps that hinder their transferability to robotics datasets. Addressing these issues may involve identifying datasets in robotics that closely resemble the characteristics of these datasets [55, 270] and training SSL methods through fine-tuning on billion-scale images [209].

7.2.4 Image Text models and SSL:

Another line of SSL approaches that have become dominant is the use of image-text models. These models just use the text as pseduo-labels and learn features that have very good discriminative and generative features. We want to see whether adding contrastive loss and dense level features to these methods can help improve performance of these methods. Adding contrastive and MAE loss can help in learning better representation from the images and may help in learning on these large scale datasets.

Bibliography

- [1] S. Arora, H. Khandeparkar, M. Khodak, O. Plevrakis, and N. Saunshi. A theoretical analysis of contrastive unsupervised representation learning, 2019.
- [2] M. Ashikhmin. Synthesizing natural textures. In *Proceedings of the 2001 symposium on Interactive 3D graphics*, pages 217–226, 2001.
- [3] E. B. Asiedu, S. Kornblith, T. Chen, N. Parmar, M. Minderer, and M. Norouzi. Decoder denoising pretraining for semantic segmentation. *preprint arXiv:2205.11423*, 2022.
- [4] M. Assran, M. Caron, I. Misra, P. Bojanowski, F. Bordes, P. Vincent, A. Joulin, M. Rabbat, and N. Ballas. Masked siamese networks for label-efficient learning. In *preprint arXiv:2204.07141*, 2022.
- [5] Y. Bai, X. Chen, A. Kirillov, A. Yuille, and A. C. Berg. Point-level region contrast for object detection pre-training. *CVPR*, 2022.
- [6] I. Balazevic, C. Allen, and T. Hospedales. Multi-relational poincaré graph embeddings. *NeurIPS*, 2019.
- [7] D. H. Ballard. Modular learning in neural networks. In *Association for the Advancement of Artificial Intelligence (AAAI)*, volume 647, pages 279–284, 1987.
- [8] H. Bao, L. Dong, and F. Wei. BEiT: BERT pre-training of image transformers. In *Int. Conf. on Learning Representations (ICLR)*, 2022.
- [9] A. Barbu, D. Mayo, J. Alverio, W. Luo, C. Wang, D. Gutfreund, J. Tenenbaum, and B. Katz. ObjectNet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.
- [10] L. Beyer, O. J. Hénaff, A. Kolesnikov, X. Zhai, and A. van den Oord. Are we done with ImageNet? In *preprint arXiv:2006.07159*, 2020.
- [11] L. Beyer, O. J. Hénaff, A. Kolesnikov, X. Zhai, and A. van den Oord. Are we done with imagenet?, 2020.
- [12] M. J. Black, G. Sapiro, D. Marimont, and D. Heeger. Robust anisotropic diffusion. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, 7 3:421–32, 1998.
- [13] S. Bonnabel. Stochastic gradient descent on riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229, 2013.

- [14] L. Bossard, M. Guillaumin, and L. Van Gool. Food-101 – mining discriminative components with random forests. In *ECCV*, 2014.
- [15] W. Brendel and M. Bethge. Approximating cnns with bag-of-local-features models works surprisingly well on imagenet, 2019.
- [16] W. Brendel and M. Bethge. Approximating CNNs with bag-of-local-features models works surprisingly well on imagenet. In *International Conference on Learning Representations*, 2019.
- [17] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017.
- [18] J. Buolamwini and T. Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*, pages 77–91. PMLR, 2018.
- [19] T. T. Cai, J. Frankle, D. J. Schwab, and A. S. Morcos. Are all negatives created equal in contrastive instance discrimination?, 2020.
- [20] J. W. Cannon, W. J. Floyd, R. Kenyon, W. R. Parry, et al. Hyperbolic geometry. *Flavors of geometry*, 31(59-115):2, 1997.
- [21] M. Caron, P. Bojanowski, A. Joulin, and M. Douze. Deep clustering for unsupervised learning of visual features. In *ECCV*, 2018.
- [22] M. Caron, P. Bojanowski, J. Mairal, and A. Joulin. Unsupervised pre-training of image features on non-curated data. 2019.
- [23] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *NeurIPS*, 2020.
- [24] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 9912–9924. Curran Associates, Inc., 2020.
- [25] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *ArXiv*, abs/2006.09882, 2020.
- [26] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments, 2021.
- [27] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.
- [28] I. Chami, A. Wolf, D.-C. Juan, F. Sala, S. Ravi, and C. Ré. Low-dimensional hyperbolic knowledge graph embeddings. In *ACL*, 2020.
- [29] I. Chami, Z. Ying, C. Ré, and J. Leskovec. Hyperbolic graph convolutional neural networks. *NeurIPS*, 2019.

- [30] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. Shapenet: An information-rich 3d model repository, 2015.
- [31] J. Chen, J. Qin, Y. Shen, L. Liu, F. Zhu, and L. Shao. Learning attentive and hierarchical representations for 3d shape recognition. In *ECCV*, 2020.
- [32] M. Chen, A. Radford, J. Wu, H. Jun, P. Dhariwal, D. Luan, and I. Sutskever. Generative pretraining from pixels. In *Int. Conference on Machine Learning (ICML)*, 2020.
- [33] P. Chen, B. B. Liao, G. Chen, and S. Zhang. Understanding and utilizing deep neural networks trained with noisy labels. In *ICML*, 2019.
- [34] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *Int. Conference on Machine Learning (ICML)*, pages 1597–1607. PMLR, 2020.
- [35] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations, 2020.
- [36] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton. Big self-supervised models are strong semi-supervised learners. *NeurIPS*, 2020.
- [37] T. Chen, C. Luo, and L. Li. Intriguing properties of contrastive losses. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 11834–11845, 2021.
- [38] X. Chen, M. Ding, X. Wang, Y. Xin, S. Mo, Y. Wang, S. Han, P. Luo, G. Zeng, and J. Wang. Context autoencoder for self-supervised representation learning. In *preprint arXiv:2202.03026*, 2022.
- [39] X. Chen, H. Fan, R. Girshick, and K. He. Improved baselines with momentum contrastive learning, 2020.
- [40] X. Chen and K. He. Exploring simple siamese representation learning, 2020.
- [41] X. Chen, S. Xie, and K. He. An empirical study of training self-supervised vision transformers. In *ICCV*, 2021.
- [42] M.-M. Cheng, Z. Zhang, W.-Y. Lin, and P. Torr. Bing: Binarized normed gradients for objectness estimation at 300fps. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3286–3293, 2014.
- [43] M. J. Choi, J. J. Lim, A. Torralba, and A. S. Willsky. Exploiting hierarchical context on a large database of object categories. In *CVPR*, 2010.
- [44] C.-Y. Chuang, J. Robinson, Y.-C. Lin, A. Torralba, and S. Jegelka. Debiased contrastive learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 8765–8775. Curran Associates, Inc., 2020.
- [45] C.-Y. Chuang, J. Robinson, Y.-C. Lin, A. Torralba, and S. Jegelka. Debiased contrastive learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 8765–8775, 2020.

- [46] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.
- [47] E. Cole, X. Yang, K. Wilber, O. Mac Aodha, and S. Belongie. When does contrastive visual representation learning work? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14755–14764, 2022.
- [48] R. S. Cruz, B. Fernando, A. Cherian, and S. Gould. Deeppermnet: Visual permutation learning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6044–6052, 2017.
- [49] R. S. Cruz, B. Fernando, A. Cherian, and S. Gould. Visual permutation learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41:3100–3114, 2018.
- [50] E. D. Cubuk, B. Zoph, D. Mané, V. Vasudevan, and Q. V. Le. Autoaugment: Learning augmentation policies from data. *ArXiv*, abs/1805.09501, 2018.
- [51] E. D. Cubuk, B. Zoph, D. Mané, V. Vasudevan, and Q. V. Le. Autoaugment: Learning augmentation policies from data. *ArXiv*, abs/1805.09501, 2018.
- [52] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le. Randaugment: Practical automated data augmentation with a reduced search space, 2019.
- [53] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le. RandAugment: Practical automated data augmentation with a reduced search space. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3008–3017, 2020.
- [54] J. S. De Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 361–368, 1997.
- [55] M. Deitke, E. VanderBilt, A. Herrasti, L. Weihs, J. Salvador, K. Ehsani, W. Han, E. Kolve, A. Farhadi, A. Kembhavi, and R. Mottaghi. Proctor: Large-scale embodied ai using procedural generation. *ArXiv*, abs/2206.06994, 2022.
- [56] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.
- [57] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255. Ieee, 2009.
- [58] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [59] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

- [60] T. DeVries and G. W. Taylor. Dataset augmentation in feature space, 2017.
- [61] M. P. Do Carmo and J. Flaherty Francis. *Riemannian geometry*, volume 6. Springer, 1992.
- [62] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1422–1430, 2015.
- [63] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *Int. Conf. on Learning Representations (ICLR)*, 2021.
- [64] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *Int. Conf. on Learning Representations (ICLR)*, 2021.
- [65] N. Dvornik, J. Mairal, and C. Schmid. On the importance of visual context for data augmentation in scene understanding. *PAMI*, 43(6):2014–2028, 2019.
- [66] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1033–1038. IEEE, 1999.
- [67] M. Everingham, L. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88:303–338, 2009.
- [68] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.
- [69] Y. Fang, L. Dong, H. Bao, X. Wang, and F. Wei. Corrupted image modeling for self-supervised visual pre-training. *preprint arXiv:2202.03382*, 2022.
- [70] J. Frankle, D. J. Schwab, A. S. Morcos, et al. Are all negatives created equal in contrastive instance discrimination? *arXiv preprint arXiv:2010.06682*, 2020.
- [71] C. Galleguillos, A. Rabinovich, and S. Belongie. Object categorization using co-occurrence, location and appearance. In *CVPR*, 2008.
- [72] O. Ganea, G. Bécigneul, and T. Hofmann. Hyperbolic entailment cones for learning hierarchical embeddings. In *ICML*, 2018.
- [73] O. Ganea, G. Bécigneul, and T. Hofmann. Hyperbolic neural networks. *NeurIPS*, 2018.
- [74] S. Ge, S. Mishra, C.-L. Li, H. Wang, and D. Jacobs. Robust contrastive learning using negative samples with diminished semantics. *Advances in Neural Information Processing Systems*, 34, 2021.
- [75] S. Ge, S. K. Mishra, H. Wang, C.-L. Li, and D. Jacobs. Robust contrastive learning using negative samples with diminished semantics. In *Advances in*

- Neural Information Processing Systems (NeurIPS)*, volume abs/2110.14189, 2021.
- [76] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *ArXiv*, abs/1811.12231, 2018.
 - [77] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2019.
 - [78] R. Geirhos, C. R. M. Temme, J. Rauber, H. H. Schütt, M. Bethge, and F. A. Wichmann. Generalisation in humans and deep neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 7549–7561, 2018.
 - [79] M. GhadimiAtigh, J. Schoep, E. Acar, N. van Noord, and P. Mettes. Hyperbolic image segmentation. *arXiv preprint arXiv:2203.05898*, 2022.
 - [80] S. Gidaris, A. Bursuc, N. Komodakis, P. Pérez, and M. Cord. Learning representations by predicting bags of visual words, 2020.
 - [81] S. Gidaris, P. Singh, and N. Komodakis. Unsupervised representation learning by predicting image rotations. *ArXiv*, abs/1803.07728, 2018.
 - [82] P. Goyal, M. Caron, B. Lefaudeaux, M. Xu, P. Wang, V. Pai, M. Singh, V. Liptchinsky, I. Misra, A. Joulin, and P. Bojanowski. Self-supervised pre-training of visual features in the wild, 2021.
 - [83] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.
 - [84] P. Goyal, P. Dollár, R. B. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He. Accurate, large minibatch SGD: Training ImageNet in 1 hour. *preprint arXiv:1706.0267*, 2017.
 - [85] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, B. Piot, k. kavukcuoglu, R. Munos, and M. Valko. Bootstrap your own latent - a new approach to self-supervised learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21271–21284. Curran Associates, Inc., 2020.
 - [86] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko. Bootstrap your own latent: A new approach to self-supervised learning, 2020.
 - [87] M. Gromov. Hyperbolic groups. In *Essays in group theory*, pages 75–263. Springer, 1987.

- [88] B. Harwood, V. Kumar B G, G. Carneiro, I. Reid, and T. Drummond. Smart mining for deep metric learning. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
- [89] R. Hataya, J. Zdenek, K. Yoshizoe, and H. Nakayama. Faster autoaugment: Learning augmentation strategies using backpropagation, 2019.
- [90] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16000–16009, June 2022.
- [91] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning, 2019.
- [92] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.
- [93] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn, 2018.
- [94] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.
- [95] D. J. Heeger and J. R. Bergen. Pyramid-based texture analysis/synthesis. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 229–238, 1995.
- [96] O. Henaff. Data-efficient image recognition with contrastive predictive coding. In *International Conference on Machine Learning*, pages 4182–4192. PMLR, 2020.
- [97] D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *ICCV*, 2021.
- [98] D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo, D. Song, J. Steinhardt, and J. Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. *arXiv preprint arXiv:2006.16241*, 2020.
- [99] D. Hendrycks and T. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *ICLR*, 2019.
- [100] D. Hendrycks and T. G. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *ArXiv*, abs/1903.12261, 2019.
- [101] D. Hendrycks, M. Mazeika, S. Kadavath, and D. Song. Using self-supervised learning can improve model robustness and uncertainty. *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
- [102] D. Hendrycks, K. Zhao, S. Basart, J. Steinhardt, and D. Song. Natural adversarial examples. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15262–15271, 2021.

- [103] K. Hermann, T. Chen, and S. Kornblith. The origins and prevalence of texture bias in convolutional neural networks. *Advances in Neural Information Processing Systems*, 33, 2020.
- [104] K. L. Hermann and A. K. Lampinen. What shapes feature representations? exploring datasets, architectures, and training, 2020.
- [105] C. Herrmann, K. Sargent, L. Jiang, R. Zabih, H. Chang, C. Liu, D. Krishnan, and D. Sun. Pyramid adversarial training improves ViT performance. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13419–13429, 2022.
- [106] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, A. Trischler, and Y. Bengio. Learning deep representations by mutual information estimation and maximization. *ArXiv*, abs/1808.06670, 2018.
- [107] C.-H. Ho and N. Nvasconcelos. Contrastive learning with adversarial examples. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 17081–17093. Curran Associates, Inc., 2020.
- [108] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 6840–6851, 2020.
- [109] X. Hou, L. Shen, K. Sun, and G. Qiu. Deep feature consistent variational autoencoder. In *2017 IEEE winter conference on applications of computer vision (WACV)*, pages 1133–1141. IEEE, 2017.
- [110] H. Huang, X. Ma, S. M. Erfani, J. Bailey, and Y. Wang. Unlearnable examples: Making personal data unexploitable. In *International Conference on Learning Representations*, 2021.
- [111] Z. Huang, X. Jin, C. Lu, Q. Hou, M.-M. Cheng, D. Fu, X. Shen, and J. Feng. Contrastive masked autoencoders are stronger vision learners. *arXiv:2207.13532v1*, 2022.
- [112] O. J. Hénaff, S. Koppula, J.-B. Alayrac, A. van den Oord, O. Vinyals, and J. Carreira. Efficient visual pretraining with contrastive detection, 2021.
- [113] O. J. Hénaff, A. Srinivas, J. D. Fauw, A. Razavi, C. Doersch, S. M. A. Eslami, and A. van den Oord. Data-efficient image recognition with contrastive predictive coding, 2019.
- [114] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry. Adversarial examples are not bugs, they are features, 2019.
- [115] M. A. Islam, M. Kowal, P. Esser, S. Jia, B. Ommer, K. G. Derpanis, and N. Bruce. Shape or texture: Understanding discriminative features in {cnn}s. In *International Conference on Learning Representations*, 2021.
- [116] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks, 2018.

- [117] C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. Le, Y.-H. Sung, Z. Li, and T. Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021.
- [118] R. Jia and P. Liang. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, 2017.
- [119] S. Jin, A. RoyChowdhury, H. Jiang, A. Singh, A. Prasad, D. Chakraborty, and E. Learned-Miller. Unsupervised hard example mining from videos for improved object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 307–324, 2018.
- [120] J. Jo and Y. Bengio. Measuring the tendency of cnns to learn surface statistical regularities. *arXiv preprint arXiv:1711.11561*, 2017.
- [121] J. Johnson, R. Krishna, M. Stark, L.-J. Li, D. Shamma, M. Bernstein, and L. Fei-Fei. Image retrieval using scene graphs. In *CVPR*, 2015.
- [122] Y. Kalantidis, M. B. Sariyildiz, N. Pion, P. Weinzaepfel, and D. Larlus. Hard negative mixing for contrastive learning, 2020.
- [123] Y. Kalantidis, M. B. Sariyildiz, N. Pion, P. Weinzaepfel, and D. Larlus. Hard negative mixing for contrastive learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21798–21809. Curran Associates, Inc., 2020.
- [124] A. Khosla, N. Jayadevaprakash, B. Yao, and L. Fei-Fei. Novel dataset for fine-grained image categorization : Stanford dogs. 2012.
- [125] V. Khrulkov, L. Mirvakhabova, E. Ustinova, I. Oseledets, and V. Lempitsky. Hyperbolic image embeddings. In *CVPR*, 2020.
- [126] M. Kim, J. Tack, and S. J. Hwang. Adversarial self-supervised contrastive learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 2983–2994. Curran Associates, Inc., 2020.
- [127] D. Kingma, T. Salimans, B. Poole, and J. Ho. Variational diffusion models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 21696–21707, 2021.
- [128] S. Kiritchenko and S. M. Mohammad. Examining gender and race bias in two hundred sentiment analysis systems. *arXiv preprint arXiv:1805.04508*, 2018.
- [129] A. Kolesnikov, L. Beyer, X. Zhai, J. Puigcerver, J. Yung, S. Gelly, and N. Houlsby. Big transfer (bit): General visual representation learning. In *ECCV*, 2020.
- [130] S. Koppula, Y. Li, E. Shelhamer, A. Jaegle, N. Parthasarathy, R. Arandjelovic, J. Carreira, and O. Hénaff. Where should i spend my flops? efficiency evaluations of visual pre-training methods. *arXiv preprint arXiv:2209.15589*, 2022.

- [131] S. Kornblith, J. Shlens, and Q. V. Le. Do better imagenet models transfer better? *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2656–2666, 2019.
- [132] J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.
- [133] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, 2017.
- [134] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012.
- [135] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.
- [136] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60:84–90, 2017.
- [137] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Mallocci, A. Kolesnikov, et al. The open images dataset v4. *International Journal of Computer Vision*, pages 1–26, 2020.
- [138] G. Larsson, M. Maire, and G. Shakhnarovich. Colorization as a proxy task for visual understanding. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 840–849, 2017.
- [139] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.
- [140] J. M. Lee. *Introduction to Riemannian manifolds*. Springer, 2018.
- [141] K. Lee, Y. Zhu, K. Sohn, C.-L. Li, J. Shin, and H. Lee. i-mix: A strategy for regularizing contrastive representation learning. In *International Conference on Learning Representations*, 2021.
- [142] C.-L. Li, K. Sohn, J. Yoon, and T. Pfister. Cutpaste: Self-supervised learning for anomaly detection and localization. *arXiv preprint arXiv:2104.04015*, 2021.
- [143] H. Li, B. Singh, M. Najibi, Z. Wu, and L. S. Davis. An analysis of pre-training on object detection, 2019.
- [144] J. Li, P. Zhou, C. Xiong, and S. Hoi. Prototypical contrastive learning of unsupervised representations. In *International Conference on Learning Representations*, 2021.
- [145] T. Li, H. Chang, S. K. Mishra, H. Zhang, D. Katabi, and D. Krishnan. Mage: Masked generative encoder to unify representation learning and image synthesis. *CVPR*, 2023.

- [146] Y. Li, Q. Yu, M. Tan, J. Mei, P. Tang, W. Shen, A. Yuille, and cihang xie. Shape-texture debiased neural network training. In *International Conference on Learning Representations*, 2021.
- [147] S. Lim, I. Kim, T. Kim, C. Kim, and S. Kim. Fast autoaugment, 2019.
- [148] T.-Y. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.
- [149] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995.
- [150] Q. Liu, M. Nickel, and D. Kiela. Hyperbolic graph neural networks. *NeurIPS*, 2019.
- [151] S. Liu, J. Chen, L. Pan, C.-W. Ngo, T.-S. Chua, and Y.-G. Jiang. Hyperbolic visual embedding learning for zero-shot recognition. In *CVPR*, 2020.
- [152] S. Liu, Z. Li, and J. Sun. Self-emd: Self-supervised object detection without imagenet, 2021.
- [153] S. Liu, J. shan Pan, and M.-H. Yang. Learning recursive filters for low-level vision via a hybrid neural network. In *ECCV*, 2016.
- [154] X. Liu, F. Zhang, Z. Hou, Z. Wang, L. Mian, J. Zhang, and J. Tang. Self-supervised learning: Generative or contrastive, 2021.
- [155] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation, 2015.
- [156] T. Long, P. Mettes, H. T. Shen, and C. G. M. Snoek. Searching for actions on the hyperbole. In *CVPR*, 2020.
- [157] R. G. Lopes, D. Yin, B. Poole, J. Gilmer, and E. D. Cubuk. Improving robustness without sacrificing accuracy with patch gaussian augmentation. *ArXiv*, abs/1906.02611, 2019.
- [158] I. Loshchilov and F. Hutter. Fixing weight decay regularization in Adam. *preprint arXiv:1711.05101*, 2017.
- [159] I. Loshchilov and F. Hutter. SGDR: Stochastic gradient descent with warm restarts. In *Int. Conf. on Learning Representations (ICLR)*, 2017.
- [160] K. Lu, S. You, and N. Barnes. Deep texture and structure aware filtering network for image smoothing, 2018.
- [161] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. Technical report, 2013.
- [162] G. Malhotra and J. Bowers. The contrasting roles of shape in human vision and convolutional neural networks. In *CogSci*, 2019.
- [163] C. Mao, L. Jiang, M. Dehghani, C. Vondrick, R. Sukthankar, and I. Essa. Discrete representations strengthen vision transformer robustness. In *Int. Conf. on Learning Representations (ICLR)*, 2022.

- [164] T. McCoy, E. Pavlick, and T. Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, 2019.
- [165] T. Mensink, E. Gavves, and C. G. Snoek. Costa: Co-occurrence statistics for zero-shot classification. In *CVPR*, 2014.
- [166] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller. Introduction to wordnet: An on-line lexical database. *International journal of lexicography*, 3(4):235–244, 1990.
- [167] S. Mishra, A. Shah, A. Bansal, J. Choi, A. Shrivastava, A. Sharma, and D. Jacobs. Learning visual representations for transfer learning by suppressing texture. *arXiv preprint arXiv:2011.01901*, 2020.
- [168] S. K. Mishra, A. B. Shah, A. Bansal, A. N. Jagannatha, A. Sharma, D. Jacobs, and D. Krishnan. Object-aware cropping for self-supervised learning. *ArXiv*, abs/2112.00319, 2021.
- [169] I. Misra and L. v. d. Maaten. Self-supervised learning of pretext-invariant representations. In *CVPR*, 2020.
- [170] I. Misra and L. van der Maaten. Self-supervised learning of pretext-invariant representations, 2019.
- [171] S. Mo, H. Kang, K. Sohn, C.-L. Li, and J. Shin. Object-aware contrastive learning for debiased scene representation. In *NeurIPS*, 2021.
- [172] C. K. Mummadi, R. Subramaniam, R. Hutmacher, J. Vitay, V. Fischer, and J. H. Metzen. Does enhanced shape bias improve neural network robustness to common corruptions? In *International Conference on Learning Representations*, 2021.
- [173] T. N. Mundhenk, D. Ho, and B. Y. Chen. Improvements to context based self-supervised learning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9339–9348, 2017.
- [174] A. Newell and J. Deng. How useful is self-supervised pretraining for visual tasks? *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2020.
- [175] M. Nickel and D. Kiela. Poincaré embeddings for learning hierarchical representations. *NeurIPS*, 2017.
- [176] M. Nickel and D. Kiela. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *ICML*, 2018.
- [177] M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. *ArXiv*, abs/1603.09246, 2016.
- [178] M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, 2016.

- [179] M. Noroozi, A. Vinjimoor, P. Favaro, and H. Pirsiavash. Boosting self-supervised learning via knowledge transfer. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [180] C. Northcutt, L. Jiang, and I. Chuang. Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70:1373–1411, 2021.
- [181] D. Parikh and T. Chen. Hierarchical semantics of objects (hsos). In *ICCV*, 2007.
- [182] D. Parikh, C. L. Zitnick, and T. Chen. Unsupervised learning of hierarchical spatial structures in images. In *CVPR*, 2009.
- [183] J. Park, J. Cho, H. J. Chang, and J. Y. Choi. Unsupervised hyperbolic representation learning via message passing auto-encoders. In *CVPR*, 2021.
- [184] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2536–2544, 2016.
- [185] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12:629–639, 1990.
- [186] J. Portilla and E. P. Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *International journal of computer vision*, 40(1):49–70, 2000.
- [187] S. Purushwalkam and A. Gupta. Demystifying contrastive self-supervised learning: Invariances, augmentations and dataset biases, 2020.
- [188] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.
- [189] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar. Do imagenet classifiers generalize to imagenet?, 2019.
- [190] S. Ren, K. He, R. B. Girshick, and J. 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.
- [191] P. H. Richemond, J.-B. Grill, F. Althé, C. Tallec, F. Strub, A. Brock, S. Smith, S. De, R. Pascanu, B. Piot, et al. Byol works even without batch statistics. *arXiv preprint arXiv:2010.10241*, 2020.
- [192] S. Ringer, W. Williams, T. Ash, R. Francis, and D. MacLeod. Texture bias of cnns limits few-shot classification performance, 2019.
- [193] J. Robinson, C.-Y. Chuang, S. Sra, and S. Jegelka. Contrastive learning with hard negative samples. *International Conference on Learning Representations*, 2021.
- [194] J. Robinson, C.-Y. Chuang, S. Sra, and S. Jegelka. Contrastive learning with hard negative samples. In *Int. Conf. on Learning Representations (ICLR)*, 2021.

- [195] J. Robinson, L. Sun, K. Yu, K. Batmanghelich, S. Jegelka, and S. Sra. Can contrastive learning avoid shortcut solutions? In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 4974–4986, 2021.
- [196] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models, 2021.
- [197] C. K. Ryali, D. J. Schwab, and A. S. Morcos. Leveraging background augmentations to encourage semantic focus in self-supervised contrastive learning. *arXiv preprint arXiv:2103.12719*, 2021.
- [198] F. Sala, C. De Sa, A. Gu, and C. Ré. Representation tradeoffs for hyperbolic embeddings. In *ICML*, 2018.
- [199] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, 2016.
- [200] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017.
- [201] R. R. Selvaraju, K. Desai, J. Johnson, and N. Naik. Casting your model: Learning to localize improves self-supervised representations. *arXiv preprint arXiv:2012.04630*, 2020.
- [202] A. Shah, S. Sra, R. Chellappa, and A. Cherian. Max-margin contrastive learning, 2021.
- [203] V. Shankar, R. Roelofs, H. Mania, A. Fang, B. Recht, and L. Schmidt. Evaluating machine accuracy on imagenet. In *ICML*, 2020.
- [204] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):640–651, Apr 2017.
- [205] B. Shi, D. Zhang, Q. Dai, Z. Zhu, Y. Mu, and J. Wang. Informative dropout for robust representation learning: A shape-bias perspective, 2020.
- [206] R. Shimizu, Y. Mukuta, and T. Harada. Hyperbolic neural networks++. In *ICLR*, 2021.
- [207] P. Simard, D. Steinkraus, and J. Platt. Best practices for convolutional neural networks applied to visual document analysis. pages 958–962, 01 2003.
- [208] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 118–126, 2015.
- [209] M. Singh, Q. Duval, K. V. Alwala, H. Fan, V. Aggarwal, A. B. Adcock, A. Joulin, P. Doll’ar, C. Feichtenhofer, R. B. Girshick, R. Girdhar, and I. Misra. The effectiveness of mae pre-pretraining for billion-scale pretraining. *ArXiv*, abs/2303.13496, 2023.

- [210] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. In *Int. Conf. on Learning Representations (ICLR)*, 2021.
- [211] B. Sorscher, R. Geirhos, S. Shekhar, S. Ganguli, and A. S. Morcos. Beyond neural scaling laws: beating power law scaling via data pruning. *ArXiv*, abs/2206.14486, 2022.
- [212] A. Steiner, A. Kolesnikov, , X. Zhai, R. Wightman, J. Uszkoreit, and L. Beyer. How to train your ViT? Data, augmentation, and regularization in vision transformers. In *Transactions on Machine Learning Research (TMLR)*, 2021.
- [213] C. Sun, A. Shrivastava, S. Singh, and A. Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Int. Conference on Computer Vision (ICCV)*, pages 843–852, 2017.
- [214] C. Sun, A. Shrivastava, S. Singh, and A. K. Gupta. Revisiting unreasonable effectiveness of data in deep learning era. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 843–852, 2017.
- [215] D. Surís, R. Liu, and C. Vondrick. Learning the predictability of the future. In *CVPR*, 2021.
- [216] C. Tao, X. Zhu, G. Huang, Y. Qiao, X. Wang, and J. Dai. Siamese image modeling for self-supervised vision representation learning. In *preprint arXiv:2206.01204*, 2022.
- [217] R. Taori, A. Dave, V. Shankar, N. Carlini, B. Recht, and L. Schmidt. Measuring robustness to natural distribution shifts in image classification. *Advances in Neural Information Processing Systems*, 33, 2020.
- [218] Y. Tian, O. J. Henaff, and A. van den Oord. Divide and contrast: Self-supervised learning from uncurated data, 2021.
- [219] Y. Tian, D. Krishnan, and P. Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019.
- [220] Y. Tian, D. Krishnan, and P. Isola. Contrastive multiview coding, 2020.
- [221] Y. Tian, C. Sun, B. Poole, D. Krishnan, C. Schmid, and P. Isola. What makes for good views for contrastive learning? In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6827–6839. Curran Associates, Inc., 2020.
- [222] Y. Tian, C. Sun, B. Poole, D. Krishnan, C. Schmid, and P. Isola. What makes for good views for contrastive learning. *arXiv preprint arXiv:2005.10243*, 2020.
- [223] A. Tifrea, G. Bécigneul, and O.-E. Ganea. Poincaré glove: Hyperbolic word embeddings. In *ICLR*. OpenReview, 2018.
- [224] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Iccv*, volume 98, page 2, 1998.

- [225] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, pages 839–846, 1998.
- [226] A. Torralba and A. A. Efros. Unbiased look at dataset bias. *CVPR 2011*, pages 1521–1528, 2011.
- [227] C. Tosh, A. Krishnamurthy, and D. Hsu. Contrastive learning, multi-view redundancy, and linear models, 2021.
- [228] D. Tsipras, S. Santurkar, L. Engstrom, A. Ilyas, and A. Madry. From imagenet to image classification: Contextualizing progress on benchmarks. In *ICML*, 2020.
- [229] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *IJCV*, 104(2):154–171, 2013.
- [230] A. van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748, 2018.
- [231] A. van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *preprint arXiv:1807.03748*, 2018.
- [232] V. Vasudevan, B. Caine, R. Gontijo-Lopes, S. Fridovich-Keil, and R. Roelofs. When does dough become a bagel? analyzing the remaining mistakes on imagenet. *arXiv preprint arXiv:2205.04596*, 2022.
- [233] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- [234] A. Veit, N. Alldrin, G. Chechik, I. Krasin, A. Gupta, and S. Belongie. Learning from noisy large-scale datasets with minimal supervision, 2017.
- [235] P. Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.
- [236] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010.
- [237] H. V. Vo, F. Bach, M. Cho, K. Han, Y. LeCun, P. Perez, and J. Ponce. Unsupervised image matching and object discovery as optimization, 2019.
- [238] H. V. Vo, E. Sizikova, C. Schmid, P. P’erez, and J. Ponce. Large-scale unsupervised object discovery. In *NeurIPS*, 2021.
- [239] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.
- [240] H. Wang, S. Ge, Z. Lipton, and E. P. Xing. Learning robust global representations by penalizing local predictive power. In *NeurIPS*, 2019.
- [241] H. Wang, S. Ge, E. P. Xing, and Z. C. Lipton. Learning robust global representations by penalizing local predictive power. *ArXiv*, abs/1905.13549, 2019.

- [242] H. Wang, X. Wu, Z. Huang, and E. P. Xing. High frequency component helps explain the generalization of convolutional neural networks, 2020.
- [243] T. Wang and P. Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere, 2020.
- [244] W. Wang, T. Zhou, F. Yu, J. Dai, E. Konukoglu, and L. Van Gool. Exploring cross-image pixel contrast for semantic segmentation. In *ICCV*, 2021.
- [245] X. Wang, R. Zhang, C. Shen, T. Kong, and L. Li. Dense contrastive learning for self-supervised visual pre-training, 2021.
- [246] X. Wang, R. Zhang, C. Shen, T. Kong, and L. Li. Dense contrastive learning for self-supervised visual pre-training. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2021.
- [247] L.-Y. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 479–488, 2000.
- [248] Z. Weng, M. G. Ogut, S. Limonchik, and S. Yeung. Unsupervised discovery of the long-tail in instance segmentation using hierarchical self-supervision. In *CVPR*, 2021.
- [249] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick. Detectron2. <https://github.com/facebookresearch/detectron2>, 2019.
- [250] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, 2018.
- [251] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin. Unsupervised feature learning via non-parametric instance discrimination. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018.
- [252] K. Xiao, L. Engstrom, A. Ilyas, and A. Madry. Noise or signal: The role of image backgrounds in object recognition. *arXiv preprint arXiv:2006.09994*, 2020.
- [253] T. Xiao, X. Wang, A. A. Efros, and T. Darrell. What should not be contrastive in contrastive learning, 2021.
- [254] J. Xie, X. Zhan, Z. Liu, Y. S. Ong, and C. C. Loy. Unsupervised object-level representation learning from scene images. In *NeurIPS*, 2021.
- [255] Z. Xie, Y. Lin, Z. Zhang, Y. Cao, S. Lin, and H. Hu. Propagate yourself: Exploring pixel-level consistency for unsupervised visual representation learning. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16679–16688, 2021.
- [256] Z. Xie, Y. Lin, Z. Zhang, Y. Cao, S. Lin, and H. Hu. Propagate yourself: Exploring pixel-level consistency for unsupervised visual representation learning. In *CVPR*, 2021.

- [257] Z. Xie, Z. Zhang, Y. Cao, Y. Lin, J. Bao, Z. Yao, Q. Dai, and H. Hu. Simmim: A simple framework for masked image modeling. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9653–9663, 2022.
- [258] Y. Xu, T. Xiao, J. Zhang, K. Yang, and Z. Zhang. Scale-invariant convolutional neural networks. *ArXiv*, abs/1411.6369, 2014.
- [259] H. Xuan, A. Stylianou, X. Liu, and R. Pless. Hard negative examples are hard, but useful. In *European Conference on Computer Vision*, pages 126–142. Springer, 2020.
- [260] J. Yan, L. Luo, C. Deng, and H. Huang. Unsupervised hyperbolic metric learning. In *CVPR*, 2021.
- [261] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le. Xlnet: Generalized autoregressive pretraining for language understanding, 2020.
- [262] Y. You, I. Gitman, and B. Ginsburg. Large batch training of convolutional networks. In *ECCV*, 2017.
- [263] J. Yu, Z. Wang, V. Vasudevan, L. Yeung, M. Seyedhosseini, and Y. Wu. CoCa: Contrastive captioners are image-text foundation models. *preprint arXiv:2205.01917*, 2022.
- [264] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features, 2019.
- [265] S. Zaidi, M. Schaarschmidt, J. Martens, H. Kim, Y. W. Teh, A. Sanchez-Gonzalez, P. Battaglia, R. Pascanu, and J. Godwin. Pre-training via denoising for molecular property prediction. *preprint arXiv:2206.00133*, 2022.
- [266] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny. Barlow twins: Self-supervised learning via redundancy reduction. *arXiv preprint arXiv:2103.03230*, 2021.
- [267] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization, 2018.
- [268] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang. Random erasing data augmentation. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2020.
- [269] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. *NeurIPS*, 2014.
- [270] H. Zhu, R. Kapoor, S. Y. Min, W. Han, J. Li, K. Geng, G. Neubig, Y. Bisk, A. Kembhavi, and L. Weihs. Excalibur: Encouraging and evaluating embodied exploration. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14931–14942, 2023.
- [271] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014.
- [272] B. Zoph, G. Ghiasi, T.-Y. Lin, Y. Cui, H. Liu, E. D. Cubuk, and Q. V. Le. Rethinking pre-training and self-training, 2020.